As to claim 64-69, refer to claims 15-25 for rejection.

As to claim 70, MARTIN1 teaches the agent registry (agent library / list of agent capabilities) is a database accessible to all electronic agents (pg. 5, A collection of agents satisfies requests from users, or other agents...one or more facilitators."; "An agent satisfying a request may require supporting information...requesting data from other agents or from the user.").

As to claim 72, refer to claim 48 for rejection.

As to claims 73 and 74, refer to claims 49 and 50 for rejection.

As to claims 75-78, refer to claims 51-54 for rejection.

As to claims 79-83, refer to claims 54-60 for rejection.

As to claims 84 and 85, MARTIN2 teaches that facilitator engines (broker agents) are distributed across at least two computer processes (multiple broker agents in an architecture) (pg 7, pg. 16) wherein each stores a planning component (schema mapping rules) (pg. 8). It would be obvious that since the broker performs the delegation that it also has an execution component and therefore each broker agent has an execution component.

As to claim 87, MARTIN1 teaches a representation of a request for service in the inter-agent language from a first agent (client agent sending a query) to a second agent (facilitator) (pg. 5). It would be obvious and well known in the art that one skilled in the art would generate program code on a data wave carrier that would entail the method of MARTIN1 and thereby obvious that the method can be entailed in a data wave carrier.

As to claim 88, MARTIN1 teaches a representation of a goal dispatched to an agent for performance from a facilitator agent (every agent can request solutions for a set of goals / facilitator is responsible for breaking them down and for distributing sub-requests to the appropriate agent) (pg. 5). It would be obvious and well known in the art that one skilled in the art would generate program code on a data wave carrier that would entail the method of MARTIN1 and thereby obvious that the method can be entailed in a data wave carrier.

As to claim 89, It is well known in the art to one skilled in the art that an agent can send back a response after processing the request. It would be obvious and well known in the art that one skilled in the art would generate program code on a data wave carrier that would entail the method of MARTIN1 and thereby obvious that the method can be entailed in a data wave carrier.

Response to Arguments

4. Applicant's arguments with respect to claims 1-89 have been considered but are moot in view of the new ground(s) of rejection.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lewis A. Bullock, Jr. whose telephone number is (703) 305-0439. The examiner can normally be reached on Monday-Friday, 8:30 am - 5:00 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Alvin E. Oberley can be reached on (703) 305-9716. The fax phone numbers for the organization where this application or proceeding is assigned are (703) 746-7239 for regular communications and (703) 746-7238 for After Final communications.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-0286.

ALVIN OBERLEY
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

lab
February 21, 2003

Notice of References Cited	Application/Control No. 09/225,198	Applicant(s)/Patent Under Reexamination CHEYER ET AL.
	Examiner Lewis A. Bullock, Jr.	Art Unit 2126 / Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
	A	US-5,802,396	09-1998	Gray, Thomas A.	710/20
	B	US-5,638,494	06-1997	Pinard et al.	709/202
	C	US-			
	D	US-			
	E	US-			
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

Form PTO-1449 (Modified)
(Use several sheets if necessary)

Sheet	1	of	2

COMPLETE IF KNOWN	
Application Number	09/225,798
Confirmation Number	
Filing Date	January 5, 1999
First Named Inventor	Cheyer
Group Art Unit	2755
Examiner Name	Unassigned
Attorney Docket No.	59501-8016.US01

U.S. PATENT DOCUMENTS

Examiner Initials	Cite No.	U.S. Patent or Application NUMBER	Kind Code (if known)	Name of Patentee or Inventor of Cited Document	Date of Publication or Filing Date of Cited Document	Pages, Columns, Lines, Where Relevant Figures Appear
LAB	1	5,197,005		Schwartz et al.	3/23/93	
LAB	2	5,386,556		Hedin et al.	1/31/95	
LAB	3	5,434,777		Luciw	7/18/95	
LAB	4	5,519,608		Kupiec	5/21/96	
LAB	5	5,608,624		Luciw	3/4/97	
LAB	6	5,721,938		Stuckey	2/24/98	
LAB	7	5,729,659		Potter	3/17/98	
LAB	8	5,748,974		Johnson	5/5/98	
LAB	9	5,774,859		Houser et al.	6/30/98	
LAB	10	5,794,050		Dahlgren et al.	8/11/98	

FOREIGN PATENT DOCUMENTS

Examiner Initial	Cite No.	Foreign Patent or Application Office	NUMBER	Kind Code (if known)	Name of Patentee or Applicant of Cited Document	Date of Publication or Filing Date of Cited Document	Pages, Columns, Lines, Where Relevant Figures Appear	T
LAB	11	WO	00/11869		Ellis et al.	3/2/00		
LAB	12	EP	0 803 826 A2		Lindblad et al.	10/29/97		

OTHER PRIOR ART-NON PATENT LITERATURE DOCUMENTS

Examiner Initials	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume issue number(s), publisher, city and/or country where published.	T
LAB	13	Dowding, John et al., "Gemini: A Natural Language System For Spoken-Language Understanding", SRI International	
LAB	14	http://www.ai.sri.com/~oaa/infowiz.html, "InfoWiz: An Animated Voice Interactive Information System, May 8, 2000	
LAB	15	Dowding, John, "Interleaving Syntax and Semantics in an Efficient Bottom-up Parser", SRI International	
LAB	16	Moore, Robert et al., "Combining Linguistic and Statistical Knowledge Sources in a Natural-Language Processing for ATIS", SRI International	

EXAMINER	DATE CONSIDERED
Louis A. Bullock Jr	2/24/03

*EXAMINER: Initial if reference considered, whether or not criteria is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to application(s).

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

Form PTO-1449 (Modified)
(Use several sheets if necessary)

Sheet	2	of	2

COMPLETE IF KNOWN	
Application Number	09/225,198
Confirmation Number	
Filing Date	January 5, 1999
First Named Inventor	Cheyer
Group Art Unit	2755
Examiner Name	Unassigned
Attorney Docket No.	59501-8016.US01

U.S. PATENT DOCUMENTS

Examiner Initials	Cite No.	U.S. Patent or Application NUMBER	Kind Code (if known)	Name of Patentee or Inventor of Cited Document	Date of Publication or Filing Date of Cited Document	Pages, Columns, Lines, Where Relevant Figures Appear
fab	17	5,802,526		Fawcett et al.	9/1/98	
fab	18	6,192,338		Haszto et al.	2/2001	
fab	19	6,173,279		Levin et al.	1/2001	
fab	20	5,805,775		Eberman et al.	9/8/98	
fab	21	5,855,002		Armstrong	12/29/98	
fab	22	5,890,123		Brown et al.	3/30/99	
fab	23	5,963,940		Liddy et al.	10/5/99	
fab	24	6,003,072		Gerritsen et al	12/14/99	
fab	25	6,012,030		French-St. George et al.	1/4/00	
fab	26	6,026,388		Liddy et al.	2/15/00	
fab	27	6,080,202		Strickland et al.	6/27/00	
fab	28	6,021,427		Spagna et al.	1/1/00	
fab	29	6,338,081		Furusawa et al.		
fab	30	6,144,989		Hodjat et al.		
fab	31	6,226,666		Chang et al.		

OTHER PRIOR ART-NON PATENT LITERATURE DOCUMENTS

Examiner Initials	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume issue number(s), publisher, city and/or country where published.	T
fab	32	Stent, Amanda et al., "The CommandTalk Spoken Dialog System", SRI International	
fab	33	Moore, Robert et al., "CommandTalk: A Spoken-Language Interface for Battlefield Simulations:, October 23, 1997, SRI International	
fab	34	Dowding, John et al., "Interpreting Language in Context in CommandTalk", February 5, 1999, SRI International	

EXAMINER	DATE CONSIDERED
Lewis A. Bullock Jr	2/21/03

*EXAMINER: Initial if reference considered, whether or not criteria is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to application(s).

2126

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF:
ADAM CHEYER ET AL.
APPLICATION NO.: 09/225,198
FILING DATE: JANUARY 5, 1999
FOR: **SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS**

ATTORNEY DOCKET NO.: 59501.8016.US01
EXAMINER: LEWIS ALEXANDER BULLOCK JR.
ART UNIT: 2126

#9

Change of Address

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

Effective immediately, please direct all further communications in the above-identified patent application to the following address:

Brian R. Coleman
Patent Attorney
Perkins Coie LLP
P. O. Box 2168
Menlo Park, CA 95026-2168

Respectfully submitted,
Perkins Coie LLP

Date: April 24 2003

Brian R. Coleman
Registration No. 39,145

Correspondence Address:
Customer No. 22918
Perkins Coie LLP
P. O. Box 2168
Menlo Park, California 94026-2168
(650) 838-4300

PDL

i

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/225,198	01/05/1999	ADAM J. CHEYER	SRI1P016	2756

22918 7590 06/03/2003

PERKINS COIE LLP
P.O. BOX 2168
MENLO PARK, CA 94026

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	
ART UNIT	PAPER NUMBER
2126	10

DATE MAILED: 06/03/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 07-01)

PRG

Interview Summary

Application N . 09/225,198	**Applicant(s)** CHEYER ET AL.	
Examiner Lewis A. Bullock, Jr.	**Art Unit** 2126	

All participants (applicant, applicant's representative, PTO personnel):

(1) *Lewis A. Bullock, Jr.*. (3)_____.

(2) *Corina Tan*. (4)_____.

Date of Interview: *2/29/03*.

Type: a)☒ Telephonic b)□ Video Conference
c)□ Personal [copy given to: 1)□ applicant 2)□ applicant's representative]

Exhibit shown or demonstration conducted: d)□ Yes e)☒ No.
If Yes, brief description: _____.

Claim(s) discussed: *Claim 1*.

Identification of prior art discussed: *Martin*.

Agreement with respect to the claims f)□ was reached. g)☒ was not reached. h)□ N/A.

Substance of Interview including description of the general nature of what was agreed to if an agreement was reached, or any other comments: *See Continuation Sheet*.

(A fuller description, if necessary, and a copy of the amendments which the examiner agreed would render the claims allowable, if available, must be attached. Also, where no copy of the amendments that would render the claims allowable is available, a summary thereof must be attached.)

THE FORMAL WRITTEN REPLY TO THE LAST OFFICE ACTION MUST INCLUDE THE SUBSTANCE OF THE INTERVIEW. (See MPEP Section 713.04). If a reply to the last Office action has already been filed, APPLICANT IS GIVEN ONE MONTH FROM THIS INTERVIEW DATE TO FILE A STATEMENT OF THE SUBSTANCE OF THE INTERVIEW. See Summary of Record of Interview requirements on reverse side or on attached sheet.

Examiner Note: You must sign this form unless it is an Attachment to a signed Office action.

Lewis A. Bullock, Jr.
Examiner's signature, if required

Summary of Record of Interview Requirements

Manual of Patent Examining Procedure (MPEP), Section 713.04, Substance of Interview Must be Made of Record
A complete written statement as to the substance of any face-to-face, video conference, or telephone interview with regard to an application must be made of record in the application whether or not an agreement with the examiner was reached at the interview.

Title 37 Code of Federal Regulations (CFR) § 1.133 Interviews
Paragraph (b)
In every instance where reconsideration is requested in view of an interview with an examiner, a complete written statement of the reasons presented at the interview as warranting favorable action must be filed by the applicant. An interview does not remove the necessity for reply to Office action as specified in §§ 1.111, 1.135. (35 U.S.C. 132)

37 CFR §1.2 Business to be transacted in writing.
All business with the Patent or Trademark Office should be transacted in writing. The personal attendance of applicants or their attomeys or agents at the Patent and Trademark Office is unnecessary. The action of the Patent and Trademark Office will be based exclusively on the written record in the Office. No attention will be paid to any alleged oral promise, stipulation, or understanding in relation to which there is disagreement or doubt.

The action of the Patent and Trademark Office cannot be based exclusively on the written record in the Office if that record is itself incomplete through the failure to record the substance of interviews.

It is the responsibility of the applicant or the attomey or agent to make the substance of an interview of record in the application file, unless the examiner indicates he or she will do so. It is the examiner's responsibility to see that such a record is made and to correct material inaccuracies which bear directly on the question of patentability.

Examiners must complete an Interview Summary Form for each interview held where a matter of substance has been discussed during the interview by checking the appropriate boxes and filling in the blanks. Discussions regarding only procedural matters, directed solely to restriction requirements for which interview recordation is otherwise provided for in Section 812.01 of the Manual of Patent Examining Procedure, or pointing out typographical errors or unreadable script in Office actions or the like, are excluded from the interview recordation procedures below. Where the substance of an interview is completely recorded in an Examiners Amendment, no separate Interview Summary Record is required.

The Interview Summary Form shall be given an appropriate Paper No., placed in the right hand portion of the file, and listed on the "Contents" section of the file wrapper. In a personal interview, a duplicate of the Form is given to the applicant (or attomey or agent) at the conclusion of the interview. In the case of a telephone or video-conference interview, the copy is mailed to the applicant's correspondence address either with or prior to the next official communication. If additional correspondence from the examiner is not likely before an allowance or if other circumstances dictate, the Form should be mailed promptly after the interview rather than with the next official communication.

The Form provides for recordation of the following information:

- Application Number (Series Code and Serial Number)
- Name of applicant
- Name of examiner
- Date of interview
- Type of interview (telephonic, video-conference, or personal)
- Name of participant(s) (applicant, attomey or agent, examiner, other PTO personnel, etc.)
- An indication whether or not an exhibit was shown or a demonstration conducted
- An identification of the specific prior art discussed
- An indication whether an agreement was reached and if so, a description of the general nature of the agreement (may be by attachment of a copy of amendments or claims agreed as being allowable). Note: Agreement as to allowability is tentative and does not restrict further action by the examiner to the contrary.
- The signature of the examiner who conducted the interview (if Form is not an attachment to a signed Office action)

It is desirable that the examiner orally remind the applicant of his or her obligation to record the substance of the interview of each case. It should be noted, however, that the Interview Summary Form will not normally be considered a complete and proper recordation of the interview unless it includes, or is supplemented by the applicant or the examiner to include, all of the applicable items required below concerning the substance of the interview.

A complete and proper recordation of the substance of any interview should include at least the following applicable items:

1) A brief description of the nature of any exhibit shown or any demonstration conducted,
2) an identification of the claims discussed,
3) an identification of the specific prior art discussed,
4) an identification of the principal proposed amendments of a substantive nature discussed, unless these are already described on the Interview Summary Form completed by the Examiner,
5) a brief identification of the general thrust of the principal arguments presented to the examiner,
   (The identification of arguments need not be lengthy or elaborate. A verbatim or highly detailed description of the arguments is not required. The identification of the arguments is sufficient if the general nature or thrust of the principal arguments made to the examiner can be understood in the context of the application file. Of course, the applicant may desire to emphasize and fully describe those arguments which he or she feels were or might be persuasive to the examiner.)
6) a general indication of any other pertinent matters discussed, and
7) if appropriate, the general results or outcome of the interview unless already described in the Interview Summary Form completed by the examiner.

Examiners are expected to carefully review the applicant's record of the substance of an interview. If the record is not complete and accurate, the examiner will give the applicant an extendable one month time period to correct the record.

Examiner t Check for Accuracy

If the claims are allowable for other reasons of record, the examiner should send a letter setting forth the examiner's version of the statement attributed to him or her. If the record is complete and accurate, the examiner should place the indication, "Interview Record OK" on the paper recording the substance of the interview along with the date and the examiner's initials.

Continuati n She t (PTO-413) **Application N . 09/225,198**

Continuation of Substance of Interview including description of the general nature of what was agreed to if an agreement was reached, or any other comments: Applicant proposed amending the claims such that the goal satisfaction plan entails the facilitating engine using "reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms. Applicant argues this is quite different then the query execution plan as detailed in Martin. The examiner will consider the amendments in view of the prior art of record in responding in the subsequent action. The interview concluded.

2151
CD-Rom

Attorney Docket No. 59501-8016.US01

CERTIFICATE OF MAILING (37 CFR 1.8(a))

I hereby certify that this paper (along with any referred to as being attached or enclosed) is being deposited with the U.S. Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Date: June 3, 2003

Sharyl Brown
Sharyl Brown

Applicants:	CHEYER et al.
Application No.:	09/225,198
Filed:	January 5, 1999
Examiner:	L. A. Bullock, Jr.
Group Art Unit	2151
For:	SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL FOR AMENDMENT AND RESPONSE AND COMPUTER PROGRAM LISTING APPENDIX SUBMITTED ON COMPACT DISC

Sir:

1. Transmitted herewith are the following:

☒ Amendment and Response
☒ Copy 1 and Copy 2 of Compact Disc both containing the identical contents of Appendix A as filed with the patent application on January 5, 1999.
☒ Amended first page of Specification
☒ IDS, 1449 and 3 references

2. Machine format is ISO-9660 file system:

File Name	Size	Creation Date	Last Date
oaa.pl	159,613 bytes	1996/10/08	1998/12/23
fac.pl	52,733 bytes	1997/04/24	1998/05/06
compound.pl	42,937 bytes	1996/12/11	1998/04/10
com_tcp.pl	18,010 bytes	1998/02/10	1998/05/06

3. Fee Authorization

Applicants believe that there is no fee due, however, the Commissioner is authorized to charge any underpayment of fees to Deposit Account No. 50-2207. This paper is submitted in duplicate.

Respectfully submitted,
Perkins Coie LLP

Date: June 3, 2003

Carina M. Tan
Carina M. Tan
Registration No. 45,769

Correspondence Address:
Customer No. 22918
Perkins Coie LLP
P. O. Box 2168
Menlo Park, California 94026-2168
(650) 838-4300

Please forward to Group Art Unit 2157

Amended Compact Discs

EXAMINER NOTE: THIS PAPER IS AN INTERNAL WORKSHEET ONLY. DO NOT ENCLOSE WITH ANY COMMUNICATION TO THE APPLICANT. ITS PURPOSE IS ONLY THAT OF AN AID IN HIGHLIGHTING A PARTICULAR PROBLEM IN A COMPACT DISC.

THE ATTACHED CD (COPY 1) HAS BEEN REVIEWED BY OIPE FOR COMPLIANCE WITH 37 CFR 1.52(E). ***Please match this CD with the application listed below.***

Date:
Serial No./Control No. 09 225 198
Reviewed By: J Williams Phone: 305 3027

☑ The compact discs are readable and acceptable.

□ Copy 1 and Copy 2 of the compact discs are not the same.

□ The compact discs are unreadable.

□ The files on the compact discs are not in ASCII.

□ The compact discs contain at least one virus.

□ Other

12 B/Rwb
9-28-03

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450,

on June 3, 2003 by Sharyl Brown
Sharyl Brown

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Atty Dkt. No. 59501-8016.US01
CHEYER et al.	Group Art Unit No.: 2151
Serial No.: 09/225,198	Examiner: L. A. Bullock, Jr.
Filed on: January 5, 1999	

For: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

Commissioner of Patents
Washington, D.C. 20231

AMENDMENT AND RESPONSE

Sir:

This is in response to the Office Action mailed March 3, 2003, the shortened statutory period for which runs until June 3, 2003.

IN THE SPECIFICATION

Done

Enclosed is substitute Page 1 of the specification which has been amended to identify the compact disk and lists the file names, size, and creation date of each file.

IN THE CLAIMS

Please amend Claims 1, 29, 61, 71 and 86. The claim amendments are submitted in "revised amendment format" as described in A*MENDMENTS IN A REVISED FORMAT NOW PERMITTED*, signed January 31, 2003, and published in *Official Gazette* on February 25, 2003.

CLAIM AMENDMENTS

B1

1. (Currently Amended) A computer-implemented method for communication and cooperative task completion among a plurality of distributed electronic agents, comprising the acts of:

registering a description of each active client agent's functional capabilities as corresponding registered functional capabilities, using an expandable, platform-independent, inter-agent language;

receiving a request for service as a base goal in the inter-agent language, in the form of an arbitrarily complex goal expression; and

dynamically interpreting the arbitrarily complex goal expression, said act of interpreting further comprising:

generating one or more sub-goals expressed in the inter-agent language;

constructing a goal satisfaction plan ~~that includes said one or more sub-goals; and~~ , wherein the goal satisfaction plan includes:

a suitable delegation of sub-goal requests to best complete the requested service request-by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms; and

dispatching each of the sub-goals to a selected client agent for performance, based on a match between the sub-goal being dispatched and the registered functional capabilities of the selected client agent.

2. (Previously Amended) A computer-implemented method as recited in claim 1, further including the following acts of:

receiving a new request for service as a base goal using the inter-agent language, in the form of another arbitrarily complex goal expression, from at least one of the selected client agents in response to the sub-goal dispatched to said agent; and

recursively applying the step of dynamically interpreting the arbitrarily complex goal expression in order to perform the new request for service.

3. (Previously Amended) A computer-implemented method as recited in claim 2 wherein the act of registering a specific agent further includes:

B1

invoking the specific agent in order to activate the specific agent;

instantiating an instance of the specific agent; and

transmitting the new agent profile from the specific agent to a facilitator agent in response to the instantiation of the specific agent.

4. A computer-implemented method as recited in claim 1 further including the act of deactivating a specific client agent no longer available to provide services by deleting the registration of the specific client agent.

5. A computer-implemented method as recited in claim 1 further comprising the act of providing an agent registry data structure.

6. A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one symbolic name for each active agent.

7. A computer-implemented method of recited in claim 5 wherein the agent registry data structure includes at least one data declaration for each active agent.

8. A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one trigger declaration for one active agent.

9. A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one task declaration, and process characteristics for each active agent.

10. A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one process characteristic for each active agent.

11. A computer-implemented method as recited in claim 1 further comprising the act of establishing communication between the plurality of distributed agents.

B1

12. A computer-implemented method as recited in claim 1 further comprising the acts of:
receiving a request for service in a second language differing from the inter-agent language;
selecting a registered agent capable of converting the second language into the inter-agent language; and
forwarding the request for service in a second language to the registered agent capable of converting the second language into the inter-agent language, implicitly requesting that such a conversion be performed and the results returned.

13. A computer-implemented method as recited in claim 12 wherein the request includes a natural language query, and the registered agent capable of converting the second language into the inter-agent language service is a natural language agent.

14. A computer-implemented method as recited in claim 13 wherein the natural language query was generated by a user interface agent.

15. A computer-implemented method as recited in claim 1, wherein the base goal requires setting a trigger having conditional functionality and consequential functionality.

16. A computer-implemented method as recited in claim 15 wherein the trigger is an outgoing communications trigger, the computer implemented method further including the acts of:
monitoring all outgoing communication events in order to determine whether a specific outgoing communication event has occurred; and
in response to the occurrence of the specific outgoing communication event, performing the particular action defined by the trigger.

B1

17. A computer-implemented method as recited in claim 15 wherein the trigger is an incoming communications trigger, the computer implemented method further including the acts of:

monitoring all incoming communication events in order to determine whether a specific incoming communication event has occurred; and

in response to the occurrence of a specific incoming communication event satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

18. A computer-implemented method as recited in claim 15 wherein the trigger is a data trigger, the computer implemented method further including the acts of:

monitoring a state of a data repository; and

in response to a particular state event satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

19. A computer-implemented method as recited in claim 15 wherein the trigger is a time trigger, the computer implemented method further including the acts of:

monitoring for the occurrence of a particular time condition; and

in response to the occurrence of a particular time condition satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

20. A computer-implemented method as recited in claim 15 wherein the trigger is installed and executed within the facilitator agent.

21. A computer-implemented method as recited in claim 15 wherein the trigger is installed and executed within a first service-providing agent.

22. A computer-implemented method as recited in claim 15 wherein the conditional functionality of the trigger is installed on a facilitator agent.

23. A computer-implemented method as recited in claim 22 wherein the consequential functionality is installed on a specific service-providing agent other than a facilitator agent.

B1

24. A computer-implemented method as recited in claim 15 wherein the conditional functionality of the trigger is installed on specific service-providing agent other than a facilitator agent.

25. A computer-implemented method as recited in claim 15 wherein the consequential functionality of the trigger is installed on a facilitator agent.

26. A computer-implemented method as recited in claim 1 wherein the base goal is a compound goal having sub-goals separated by operators.

27. A computer-implemented method as recited in claim 26 wherein the type of available operators includes a conjunction operator, a disjunction operator, and a conditional execution operator.

28. A computer-implemented method as recited in claim 27 wherein the type of available operators further includes a parallel disjunction operator that indicates that disjunct goals are to be performed by different agents.

29. (Currently Amended) A computer program stored on a computer readable medium, the computer program executable to facilitate cooperative task completion within a distributed computing environment, the distributed computing environment including a plurality of autonomous electronic agents, the distributed computing environment supporting an Interagent Communication Language, the computer program comprising computer executable instructions for:

providing an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment;

B1

interpreting a service request in order to determine a base goal that may be a compound, arbitrarily complex base goal, the service request adhering to an Interagent Communication Language (ICL), the act of interpreting including the sub-acts of:

determining any task completion advice provided by the base goal, and

determining any task completion constraints provided by the base goal;

constructing a base goal satisfaction plan including the sub-acts of:

determining whether the requested service is available,

determining sub-goals required in completing the base goal by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms,

selecting service-providing electronic agents from the agent registry suitable for performing the determined sub-goals, and

ordering a delegation of sub-goal requests to best complete the requested service; and

implementing the base goal satisfaction plan.

30. A computer program as recited in claim 29 wherein the computer executable instruction for providing an agent registry includes the following computer executable instructions for registering a specific service-providing electronic agent into the agent registry:

establishing a bi-directional communications link between the specific agent and a facilitator agent controlling the agent registry;

providing a new agent profile to the facilitator agent, the new agent profile defining publicly available capabilities of the specific agent; and

registering the specific agent together with the new agent profile within the agent registry, thereby making available to the facilitator agent the capabilities of the specific agent.

31. A computer program as recited in claim 30 wherein the computer executable instruction for registering a specific agent further includes:

invoking the specific agent in order to activate the specific agent;

instantiating an instance of the specific agent; and transmitting the new agent profile from the specific agent to the facilitator agent in response to the instantiation of the specific agent.

B1

32. A computer program as recited in claim 29 wherein the computer executable instruction for providing an agent registry includes a computer executable instruction for removing a specific service-providing electronic agent from the registry upon determining that the specific agent is no longer available to provide services.

33. A computer program as recited in claim 29 wherein the provided agent registry includes a symbolic name, a unique address, data declarations, trigger declarations, task declarations, and process characteristics for each active agent.

34. Computer program as recited in claim 29 further including computer executable instructions for receiving the service request via a communications link established with a client.

35. A computer program as recited in claim 29 wherein the computer executable instruction for providing a service request includes instructions for:
receiving a non-ICL format service request;
selecting an active agent capable of converting the non-ICL formal service request into an ICL format service request;
forwarding the non-ICL format service request to the active agent capable of converting the non-ICL format service request, together with a request that such conversion be performed; and
receiving an ICL format service request corresponding to the non-ICL format service request.

36. A computer program as recited in claim 35 wherein the non-ICL format service request includes a natural language query, and the active agent capable of converting the non-ICL formal service request into an ICL format service request is a natural language agent.

37. A computer program as recited in claim 36 wherein the natural language query is generated by a user interface agent.

38. A computer program as recited in claim 29, the computer program further including computer executable instructions for implementing a base goal that requires setting a trigger having conditional and consequential functionality.

B1

39. A computer program as recited in claim 38 wherein the trigger is an outgoing communications trigger, the computer program further including computer executable instructions for:
monitoring all outgoing communication events in order to determine whether a specific outgoing communication event has occurred; and
in response to the occurrence of the specific outgoing communication event, performing the particular action defined by the trigger.

40. A computer program as recited in claim 38 wherein the trigger is an incoming communications trigger, the computer program further including computer executable instructions for:
monitoring all incoming communication events in order to determine whether a specific incoming communication event has occurred; and
in response to the occurrence of the specific incoming communication event, performing the particular action defined by the trigger.

41. A computer program as recited in claim 38 wherein the trigger is a data trigger, the computer program further including computer executable instructions for:
monitoring a state of a data repository; and
in response to a particular state event, performing the particular action defined by the trigger.

42. A computer program as recited in claim 38 wherein the trigger is a time trigger, the computer program further including computer executable instructions for:
monitoring for the occurrence of a particular time condition; and
in response to the occurrence of the particular time condition, performing the particular action defined by the trigger.

43. A computer program as recited in claim 38 further including computer executable instructions for installing and executing the trigger within the facilitator agent.

B1

44. A computer program as recited in claim 38 further including computer executable instructions for installing and executing the trigger within a first service-providing agent.

45. A computer program as recited in claim 29 further including computer executable instructions for interpreting compound goals having sub-goals separated by operators.

46. A computer program as recited in claim 45 wherein the type of available operators includes a conjunction operator, a disjunction operator, and a conditional execution operator.

47. A computer program as recited in claim 46 wherein the type of available operators further includes parallel disjunction operator that indicates that distinct goals are to be performed by different agents.

48. (Currently Amended) An Interagent Communication Language (ICL) providing a basis for facilitated cooperative task completion within a distributed computing environment having a facilitator agent and a plurality of autonomous service-providing electronic agents, wherein:
the ICL having one or more features from a set of features comprising:

enabling agents to perform queries of other agents;
enabling agents to exchange information with other agents; and
enabling agents to set triggers within other agents; and
the ICL having a syntax supporting compound goal expressions wherein said compound goal expressions are such that goals within a single request provided according to the ICL syntax may be coupled by one or more operators from a set of operators comprising:

~~a conjunctive operator;~~
a conditional execution operator; and
a parallel disjunctive operation that indicates that disjunct goals are to be performed by different agents.

49. An ICL as recited in claim 48, wherein the ICL is computer platform independent.

B1

50. An ICL as recited in claim 48 wherein the ICL is independent of computer programming languages which the plurality of agents are programmed in.

51. An ICL as recited in claim 48 wherein the ICL syntax supports explicit task completion constraints include use of specific agent constraints and response time constraints.

52. An ICL as recited in claim 51, wherein possible types of task completion constraints include use of specific agent constraints and response time constraints.

53. An ICL as recited in claim 51 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

54. An ICL as recited in claim 48 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

55. An ICL as recited in claim 48 wherein each autonomous service-providing electronic agent defines and publishes a set of capability declarations or solvables, expressed in ICL, that describes services provided by such electronic agent.

56. An ICL as recited in claim 55 wherein an electronic agent's solvables define an interface for the electronic agent.

57. An ICL as recited in claim 56 wherein the facilitator agent maintains an agent registry making available a plurality of electronic agent interfaces.

58. An ICL as recited in claim 57 wherein the possible types of solvables includes procedure solvables, a procedure solvable operable to implement a procedure such as a test or an action.

59. An ICL as recited in claim 58 wherein the possible types of solvables further includes data solvables, a data solvable operable to provide access to a collection of data.

B1

60. An ICL as recited in claim 58 wherein the possible types of solvables includes data solvables, a data solvable operable to provide access to a collection of data.

61. (Currently Amended) A facilitator agent arranged to coordinate cooperative task completion within a distributed computing environment having a plurality of autonomous service-providing electronic agents, the facilitator agent comprising:

an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment; and

a facilitating engine operable to parse a service request in order to interpret a compound goal set forth therein, the compound goal including both local and global constraints and control parameters, the service request formed according to an Interagent Communication Language (ICL), the facilitating engine further operable to construct a goal satisfaction plan by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms ~~specifying the coordination of a suitable delegation of sub-goal requests to complete the requested service satisfying both the local and global constraints and control parameters~~.

62. A facilitator agent as recited in claim 61, wherein the facilitating engine is capable of modifying the goal satisfaction plan during execution, the modifying initiated by events such as new agent declarations within the agent registry, decisions made by remote agents, and information provided to the facilitating engine by remote agents.

63. A facilitator agent as recited in claim 61 wherein the agent registry includes a symbolic name, a unique address, data declarations, trigger declarations, task declarations, and process characteristics for each active agent.

64. A facilitator agent as recited in claim 61 wherein the facilitating engine is operable to install a trigger mechanism requesting that a certain action be taken when a certain set of conditions are met.

31

65. A facilitator agent as recited in claim 64 wherein the trigger mechanism is a communication trigger that monitors communication events and performs the certain action when a certain communication event occurs.

66. A facilitator agent as recited in claim 64 wherein the trigger mechanism is a data trigger that monitors a state of a data repository and performs the certain action when a certain data state is obtained.

67. A facilitator agent as recited in claim 66 wherein the data repository is local to the facilitator agent.

68. A facilitator agent as recited in claim 66 wherein the data repository is remote from the facilitator agent.

69. A facilitator agent as recited in claim 64 wherein the trigger mechanism is a task trigger having a set of conditions.

70. A facilitator agent as recited in claim 61, the facilitator agent further including a global database accessible to at least one of the service-providing electronic agents.

71. (Currently Amended) A software-based, flexible computer architecture for communication and cooperation among distributed electronic agents, the architecture contemplating a distributed computing system comprising:
a plurality of service-providing electronic agents; and
a facilitator agent in bi-directional communications with the plurality of service-providing electronic agents, the facilitator agent including:

an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment;

B1

a facilitating engine operable to parse a service request in order to interpret an arbitrarily complex goal set forth therein, the facilitating engine further operable to construct a goal satisfaction plan including the coordination of a suitable delegation of sub-goal requests to best complete the requested service by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms.

72. A computer architecture as recited in claim 71, wherein the basis for the computer architect is an Interagent Communication Language (ICL) enabling agents to perform queries of other agents, exchange information with other agents, and set triggers within other agents, the ICL further defined by an ICL syntax supporting compound goal expressions such that goals within a single request provided according to the ICL syntax may be coupled by a conjunctive operator, a disjunctive operator, a conditional execution operator, and a parallel disjunctive operator parallel disjunctive operator that indicates that disjunct goals are to be performed by different agents.

73. A computer architecture as recited in claim 72, wherein the ICL is computer platform independent.

74. A computer architecture as recited in claim 73 wherein the ICL is independent of computer programming languages in which the plurality of agents are programmed.

75. A computer architecture as recited in claim 73 wherein the ICL syntax supports explicit task completion constraints within goal expressions.

76. A computer architecture as recited in claim 75 wherein possible types of task completion constraints include use of specific agent constraints and response time constraints.

77. A computer architecture as recited in claim 75 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

B1

78. A computer architecture as recited in claim 73 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

79. A computer architecture as recited in claim 73 wherein each autonomous service-providing electronic agent defines and publishes a set of capability declarations or solvables, expressed in ICL, that describes services provided by such electronic agent.

80. A computer architecture as recited in claim 79 wherein an electronic agent's solvables define an interface for the electronic agent.

81. A computer architecture as recited in claim 80 wherein the possible types of solvables includes procedure solvables, a procedure solvable operable to implement a procedure such as a test or an action.

82. A computer architecture as recited in claim 81 wherein the possible types of solvables further includes data solvables, a data solvable operable to provide access to a collection of data.

83. A computer architecture as recited in claim 82 wherein the possible types of solvables includes a data solvable operable to provide access to modify a collection of data.

84. (Previously Amended) A computer architecture as recited in claim 71 wherein a planning component of the facilitating engine are distributed across at least two computer processes.

85. (Previously Amended) A computer architecture as recited in claim 71 wherein an execution component of the facilitating engine is distributed across at least two computer processes.

B1

86. (Currently Amended) A data wave carrier providing a transport mechanism for information communication in a distributed computing environment having at least one facilitator agent and at least one active client agent, wherein said at least one facilitator agent is operable to construct a goal satisfaction plan by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms for satisfying one or more requests for service from said at least one active client agent, the data wave carrier comprising a signal representation of an inter-agent language description of an active client agent's functional capabilities.

87. (Previously Amended) A data wave carrier as recited in claim 86, the data wave carrier further comprising a corresponding signal representation of said one or more requests for service in the inter-agent language from a first agent to a second agent.

88. (Previously Amended) A data wave carrier as recited in claim 86, the data wave carrier further comprising a signal representation of a goal dispatched to an agent for performance from a facilitator agent.

89. A data wave carrier as recited in claim 88 wherein a later state of the data wave carrier comprises a signal representation of a response to the dispatched goal including results and/or a status report from the agent for performance to the facilitator agent.

REMARKS

The Examiner is thanked for the performance of a thorough search. By this amendment, Claims 1, 29, 61, 71 and 86 have been amended. No claims have been cancelled or added. Hence, Claims 1-89 are pending in the Application. It is respectfully submitted that the amendments to the claims as indicated herein do not add any new matter to this Application. Furthermore, amendments made to the claims as indicated herein have been made to improve readability and clarity of the claims. Applicants enclose a CD-ROM labeled as Copy 1 and an identical copy of the CD-ROM labeled as Copy 2 containing the identical contents of Appendix A as filed with the patent application on January 5, 1999. Also enclosed is substitute Page 1 of the specification which has been amended to identify the compact disc and list the file names, size, and creation date of each file.

SUMMARY OF REJECTIONS/OBJECTIONS

In the Office Action, Claims 1-89 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Developing Tools for the Open Agent Architecture" by Martin1 in view of "Information Brokering in an Agent Architecture" by Martin2.

REJECTIONS UNDER 35 U.S.C. § 103(a)

CLAIMS 1, 29, 61, 71 and 86

Claim 1 recites, in part, the features:

"constructing a goal satisfaction plan, wherein the goal satisfaction plan includes:

> a suitable delegation of sub-goal requests to best complete the requested service request **by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms;**"

Claim 1 has been amended to clarify that the facilitating engine uses sophisticated reasoning when delegating sub-goal requests to best complete the requested service request. The facilitating engine's use of reasoning is supported by the specification on page 10, lines 15 – 18. Amended Claim 1 requires that the facilitating engine use "reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms.

For purposes of explanation, assume that the facilitator receives a request such as, "Make Coffee". The facilitator's facilitating engine uses reasoning to generate the following goal satisfaction plan:

Sub-goal request A: Roast coffee beans
Sub-goal request B: Grind coffee beans
Sub-goal request C: Boil water, etc.

The facilitating engine is able to use reasoning to generate a plan to accomplish the base goal, "Make Coffee". The reasoning includes "one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms." For example, the facilitating engine uses its domain-specific reasoning based on domain-specific knowledge of symbols and axioms of the domain. In the above example, the facilitating engine uses its knowledge about domain symbols and axioms such as "coffee", "roast", and "beans" in order to generate a goal satisfaction plan by reasoning that making coffee entails roasting coffee beans, grinding coffee beans and boiling water, etc. Also, the coffee beans need to be roasted before the coffee beans can be ground and that only after the coffee beans are ground should water be boiled.

Further, the facilitating engine is able to use reasoning to delegate the sub-goals to service providing agents in such a way as "to best complete the requested service request." For example, assume that several agents are able to roast coffee. The facilitating engine is able to use reasoning to delegate the sub-goal task of roasting coffee to the service-providing agent that can roast beans in the least amount of time because the facilitating engine has reasoned that the least amount of time taken to make coffee is the best way to accomplish the base goal of making coffee.

Similarly, to use an example taken directly from the specification (see page 21, starting at line 29 to page 22, line 1-4), the facilitating engine accomplished the request "Remind Bob about lunch" by reasoning that all available message transfer agents (e.g., fax, phone, mail, pager) are to be enabled to **compete** for the opportunity to carry out the request. In other words, the base goal is carried out not by merely parsing the request into sub-goals **based on the syntax** of the request. Rather, the facilitating engine used reasoning to decide upon using **competing** message transfer agents to reminding Bob of lunch, in lieu of delegating the task to just one message transfer agent.

In contrast, *Martin's* "Development Tools for the Open Agent Architecture" (*Martin1*) and *Martin's* "Information Brokering in An Agent Architecture" fail to teach the goal satisfaction plan that entails the type of reasoning described above as performed by the facilitator agent. As mentioned by the Examiner in the Office Action, *Martin's* "Development Tools for the Open Agent Architecture" does not teach the act of constructing a goal satisfaction plan.

As for *Martin's* "Information Brokering in An Agent Architecture" (*Martin2*), it merely discloses query processing and a query execution plan which is NOT the same as a goal execution plan. Thus, *Martin2* is merely describing a method for information retrieval rather than fulfillment of a service request. Moreover, query execution plans are well-known in database systems. In database systems, query statements are made in query languages such as SQL. SQL statements are fulfilled according to a query execution plan based on the manner in which information is stored in the database. In contrast, the goal satisfaction plan is a plan that entails reasoning in its construction, rather than being based on the manner in which information is stored in a database.

Further, *Martin2* merely teaches that the queries are systematically broken based on syntax of the queries without any kind of reasoning for forming a goal satisfaction plan such as that of the "Make Coffee" example above. In *Martin2*, on page 11, *Martin2* teaches the construction of a query execution plan by analysis of "each predicate in the query" and the rewriting of the query for dispatch to information sources based on "a disjunction of translated subqueries. Therefore in *Martin2*, each request made of information sources **must have appeared syntactically** (albeit with language translation) **in the original query**.

Neither *Martin1* nor *Martin2*, either alone or in combination, disclose, teach, suggest or make obvious the novel features of claim 1. Thus, Claim 1 is allowable.

Claims 29, 61, 71 and 86, each contain similar features regarding the use "reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms. Thus, Claims 26, 61, 71 and 86 are allowable for at least the reasons provided herein in respect to Claim 1.

CLAIMS 2-28, 30-47, 62-70, 72-85 and 87-89

Claims 2-28 are either directly or indirectly dependent upon Claim 1 and include all the limitations of Claim 1 and therefore are allowable for at least the reasons provided herein in respect to Claim 1.

Claims 30-47 are either directly or indirectly dependent upon Claim 29 and include all the limitations of Claim 29 and therefore are allowable for at least the reasons provided herein in respect to Claim 29.

Claims 62-70 are either directly or indirectly dependent upon Claim 61 and include all the limitations of Claim 61 and therefore are allowable for at least the reasons provided herein in respect to Claim 61.

Claims 72-85 are either directly or indirectly dependent upon Claim 71 and include all the limitations of Claim 71 and therefore are allowable for at least the reasons provided herein in respect to Claim 71

Claims 87-89 are either directly or indirectly dependent upon Claim 86 and include all the limitations of Claim 86 and therefore are allowable for at least the reasons provided herein in respect to Claim 86.

CLAIM 48

Claim 48 as amended, recites in part:

> "the ICL having a syntax supporting compound goal expressions wherein said_compound goal expressions are such that **goals within a single request** provided according to the ICL syntax may **be coupled by one or more operators from a set of operators** comprising:
> **a conditional execution operator**; and
> **a parallel disjunctive operator** that indicates that disjunct goals are to be performed by different agents."

The novel method recited in Claim 48 as amended requires that "**goals within a single request**" are "**coupled by one or more operators from a set of operators**". In amended Claim 48, the set of operators comprise, **a conditional execution operator,** and **a parallel disjunctive operator**.

In the Office Action, the Examiner states that "the ICL having expression which may be coupled by a conjunctive operator". The claim has therefore been amended to clarify the applicant's invention. It is to be noted that *Martin2* does not suggest or mention **conditional execution operator,** and **a parallel disjunctive operators.**

None of the cited references disclose, suggest or render obvious the requirement that the "**goals within a single request**" be "coupled by one or more operators from a set of operators", such as **a conditional execution operator** (such as "if" and "when", allowing for particular actions to be predicated on the state, or outcomes of earlier actions), and **a parallel disjunctive operator** (allowing for alternative actions to be performed at the same time, if resources allow, and a first-to-respond strategy may be used in their competition to perform the goal at hand). Claim 48 is allowable over the art of record. Thus, it is respectfully submitted that Claim 48 be held in condition for allowance.

CLAIMS 49-60

Claims 49-60 are either directly or indirectly dependent upon independent Claim 48, and include all the features of Claim 48. Therefore, Claims 49-60 are allowable for at least the reasons provided herein with respect to Claim 48. Furthermore, it is respectfully submitted that Claims 49-60 recite additional features that independently render Claims 49-60 patentable over the art of record. Thus, it is respectfully submitted that Claims 49-60 be held in condition for allowance.

CONCLUSION

For the reasons set forth above, it is respectfully submitted that all of the pending claims are now in condition for allowance. Therefore, the issuance of a formal Notice of Allowance is believed next in order, and that action is most earnestly solicited.

If in the opinion of the Examiner a telephone conference would expedite the prosecution of the subject application, the Examiner is encouraged to call the undersigned at (650) 838-4311.

The Commissioner is authorized to charge any fees due to Applicants' Deposit Account No. 50-2207.

Respectfully submitted,
Perkins Coie LLP

Carina M. Tan

Date: June 3, 2003

Carina M. Tan
Registration No. 45,769

Correspondence Address:

Customer No. 22918
Perkins Coie LLP
P. O. Box 2168
Menlo Park, California 94026
(650) 838-4300

Marked-up version

Software-Based Architecture for Communication and Cooperation Among Distributed Electronic Agents

By:

Adam J. Cheyer and David L. Martin

A compact disk containing a computer program listing has been provided in duplicate (copy 1 and copy 2 of the compact disk are identical). The computer program listing in the compact disk is incorporated by reference herein. The compact disk contains files with their names, size and date of creation as follow:

File Name	Size	Creation Date	Last Date
oaa.pl	159,613 bytes	1996/10/08	1998/12/23
fac.pl	52,733 bytes	1997/04/24	1998/05/06
compound.pl	42,937 bytes	1996/12/11	1998/04/10
com_tcp.pl	18,010 bytes	1998/02/10	1998/05/06

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention is related to distributed computing environments and the completion of tasks within such environments. In particular, the present invention teaches a variety of software-based architectures for communication and cooperation among distributed electronic agents. Certain embodiments teach interagent communication languages enabling client agents to make requests in the form of arbitrarily complex goal expressions that are solved through facilitation by a facilitator agent.

Context and Motivation for Distributed Software Systems

The evolution of models for the design and construction of distributed software systems is being driven forward by several closely interrelated trends: the adoption of a *networked computing model*, rapidly rising expectations for *smarter, longer-lived, more autonomous software applications* and an ever increasing demand for *more accessible and intuitive user interfaces*.

Prior Art Figure 1 illustrates a *networked computing model* 100 having a plurality of client and server computer systems 120 and 122 coupled together over a physical transport mechanism 140. The adoption of the *networked computing model* 100 has lead to a greatly increased reliance on distributed sites for both data and processing resources. Systems such as the networked computing model 100 are based upon at least one physical transport mechanism 140 coupling the multiple computer systems 120 and 122 to support the transfer of information between these computers. Some of these computers basically support using the network and are known as *client*

Attorney Docket No. 59501-8016.US01

#11

I hereby certify that this correspondence is being deposited with the U.S. Postal Service with sufficient postage as First Class Mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA. 22313-1450, on:

Date: June 3, 2003 By: Sharyl Brown
Sharyl Brown

D. Bond
6/19/03

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF:
Cheyer *et al.*

APPLICATION NO.: 09/225,198

FILED: January 5, 1999

FOR: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

EXAMINER: L. A. BULLOCK, JR.

ART UNIT: 2151

Supplemental Information Disclosure Statement After First Office Action but Before Final Action or Notice of Allowance – 37 CFR 1.97(c)

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

1. Timing of Submission

The information transmitted herewith is being filed *after* three months of the filing date of this application or after the mailing date of the first Office action on the merits, whichever occurred last, but *before* the mailing date of either a final action under 37 CFR 1.113 or a Notice of Allowance under 37 CFR 1.311, whichever occurs first. The references listed on the enclosed Form PTO/SB/08A may be material to the examination of this application; the Examiner is requested to make them of record in the application.

2. Cited Information

☒ Copies of the following references are enclosed:

☒ All cited references
☐ References marked by asterisks
☐ The following:

3. Effect of Information Disclosure Statement (37 CFR 1.97(h))

This Information Disclosure Statement is not to be construed as a representation that: (i) a search has been made; (ii) additional information material to the examination of this application does not exist; (iii) the information, protocols, results and the like reported by third parties are accurate or enabling; or (iv) the cited information is, or is considered to be, material to patentability. In addition, applicant does not admit that any enclosed item of information constitutes prior art to the subject invention and specifically reserves the right to demonstrate that any such reference is not prior art.

4. Fee Payment (37 CFR 1.97(c)) or Certification (37 CFR 1.97(e))

☒ Applicant submits that no fee is due in light of the following certification under 37 CFR 1.97(e) (check only one):

□ In accordance with 37 CFR 1.97(e)(1), the undersigned hereby states that each item of information submitted herewith was cited in a communication from a foreign patent office in a counterpart foreign application not more than three months prior to this filing of this statement; or

☒ In accordance with 37 CFR 1.97(e)(2), the undersigned hereby states that no item of information submitted herewith was cited in a communication from a foreign patent office in a counterpart foreign application, or, to the knowledge of the person signing the certification after making reasonable inquiry, was known to any individual designated in 37 CFR 1.56(c), more than three months prior to the filing of this statement.

☒ Please charge any underpayment for timely filing of this paper to Deposit Account No. 50-2207.

5. Patent Term Adjustment (37 CFR 1.704(d))

□ The undersigned states that each item of information submitted herewith was cited in a communication from a foreign patent office in a counterpart application and that this communication was not received by any individual designated in 37 C.F.R. §1.56(c) more than thirty days prior to the filing of this statement. 37 C.F.R. §1.704(d).

Respectfully submitted,
Perkins Coie LLP

Date: 6/3/03

Carina M. Tan
Carina M. Tan
Registration No. 45,769

Correspondence Address:
Customer No. 22918
Perkins Coie LLP
P.O. Box 2168
Menlo Park, California 94026
(650) 838-4300

Software-Based Architecture for Communication and Cooperation Among Distributed Electronic Agents

By:

Adam J. Cheyer and David L. Martin

A compact disk containing a computer program listing has been provided in duplicate (copy 1 and copy 2 of the compact disk are identical). The computer program listing in the compact disk is incorporated by reference herein. The compact disk contains files with their names, size and date of creation as follow:

File Name	Size	Creation Date	Last Date
oaa.pl	159,613 bytes	1996/10/08	1998/12/23
fac.pl	52,733 bytes	1997/04/24	1998/05/06
compound.pl	42,937 bytes	1996/12/11	1998/04/10
com_tcp.pl	18,010 bytes	1998/02/10	1998/05/06

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention is related to distributed computing environments and the completion of tasks within such environments. In particular, the present invention teaches a variety of software-based architectures for communication and cooperation among distributed electronic agents. Certain embodiments teach interagent communication languages enabling client agents to make requests in the form of arbitrarily complex goal expressions that are solved through facilitation by a facilitator agent.

Context and Motivation for Distributed Software Systems

The evolution of models for the design and construction of distributed software systems is being driven forward by several closely interrelated trends: the adoption of a *networked computing model*, rapidly rising expectations for *smarter, longer-lived, more autonomous software applications* and an ever increasing demand for *more accessible and intuitive user interfaces*.

Prior Art Figure 1 illustrates a *networked computing model* 100 having a plurality of client and server computer systems 120 and 122 coupled together over a physical transport mechanism 140. The adoption of the *networked computing model* 100 has lead to a greatly increased reliance on distributed sites for both data and processing resources. Systems such as the networked computing model 100 are based upon at least one physical transport mechanism 140 coupling the multiple computer systems 120 and 122 to support the transfer of information between these computers. Some of these computers basically support using the network and are known as *client*

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/225,198	01/05/1999	ADAM J. CHEYER	SRI1P016	2756

22918 7590 11/28/2003

PERKINS COIE LLP
P.O. BOX 2168
MENLO PARK, CA 94026

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	
ART UNIT	PAPER NUMBER
2126	17

DATE MAILED: 11/28/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

M

Offic Action Summary	**Application N .** 09/225,198	**Applicant(s)** CHEYER ET AL.	
	Examin r Lewis A. Bullock, Jr.	**Art Unit** 2126	

-- The MAILING DATE of this communication appears on the cover sheet with th correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 03 June 2003.

2a)☒ This action is **FINAL**. 2b)□ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-86 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-86 is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)□ The specification is objected to by the Examiner.

10)□ The drawing(s) filed on _____ is/are: a)□ accepted or b)□ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)□ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. §§ 119 and 120

12)□ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)□ All b)□ Some * c)□ None of:

1.□ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____.

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

13)□ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application) since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

a) □ The translation of the foreign language provisional application has been received.

14)□ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121 since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) 11 .

4) □ Interview Summary (PTO-413) Paper No(s). _____ .

5) □ Notice of Informal Patent Application (PTO-152)

6) □ Other: .

DETAILED ACTION

Compact Disc Submission

1. The description portion of this application contains a computer program listing consisting of more than three hundred (300) lines. In accordance with 37 CFR 1.96(c), a computer program listing printout of more than three hundred lines must be submitted as a computer program listing appendix on compact disc conforming to the standards set forth in 37 CFR 1.96(c)(2) and must be appropriately referenced in the specification (see 37 CFR 1.77(b)(4)). Accordingly, applicant is required to cancel the computer program listing appearing in the specification on pages Appendix, file a computer program listing appendix on compact disc in compliance with 37 CFR 1.96(c) and insert an appropriate reference to the newly added computer program listing appendix on compact disc at the beginning of the specification. Applicant must include the Appendix A.V, source code file named translations.pl. with the other appendices on a compact disc.

* Applicant is also requested to delete the Brief Description of the Appendices on page 8, line 23 – page 9, line 3, since the amendment to page 1 is made.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-3, 5-11, 15-25, 29-34, 38-44, 61-71, and 86-89 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Development Tools for the Open Agent Architecture" by MARTIN1 in view of KISS (US 6,484,155).

As to claim 1, MARTIN1 teaches a computer-implemented method for communication and cooperative task completion among a plurality of distributed agents (sub-agents / agents), comprising the acts of: registering a description of each client agent's functional capabilities, using a platform independent inter-agent language (pg. 5, Each facilitator records the published capabilities of their subagents..."); receiving a request as a base goal in the inter-agent language (ICL form), in the form of an arbitrarily complex goal expression (request) (pg. 5, "...and when requests arrive.."); and dynamically interpreting the complex goal expression (request) comprising: generating one or more sub-goals (sub-request) expressed in the inter-agent language (ICL) (pg. 5, ...the facilitator is responsible for breaking them down and for distributing subrequest.."); and dispatching each of the sub-goals (sub-request) to a selected client agent (agent) for performance ("pg. 5, "...and when requests arrive (expressed in the Inter-agent Communication Language, described below), the facilitator is responsible for breaking them down and for distributing sub-requests to the appropriate agents; "For example, every agent can...and request solutions for a set of goals,..."). It would be inherent that since the functionalities of an agent are registered with the facilitator that they are stored registered functional capabilities of that agent and that the request is a complex goal since the facilitator can be requested to provide solutions for a set of goals (pg. 5). However, MARTIN1 does not teach the step of constructing a goal satisfaction plan.

KISS teaches an agent architecture for communicating and cooperation among distributed electronic agents (user agents / meta agents / and knowledge agents), wherein a facilitator agent (meta agent) is operable for generating / constructing a goal satisfaction plan (dynamic "solution plan") associated with the base goal (query) wherein the goal satisfaction plan includes a suitable delegation of sub-goal requests (sub-plans / tasks) to best complete the requested service request-by using domain-independent or domain –specific reasoning (col. 5, lines 14-45; col. 8, lines 21 – col. 9, line 26; col. 10, lines 10-38; col. 2, lines 50-67). Therefore, it would be obvious to combine the teachings of MARTIN1 with the teachings of KISS in order that inference be distributed and cooperative over a distributed environment (col. 3, lines 47 – col. 4, line 17).

As to claim 29, MARTIN1 teaches a method to facilitate cooperative task completion within a distributed computing environment supporting an Inter-agent Communication Language among a plurality of electronic agents (sub-agents / agents) comprising: providing an agent registry as disclosed (facilitator storage of published sub-agents capabilities); interpreting a service request in order to determine a base goal (via facilitator); determining whether the requested service is available, determining sub-goals required in completing the base goal (determine solutions for a set of goals) selecting suitable service-providing electronic agents for performing the sub-goals, and ordering a delegation of sub-goal requests to complete the requested service (pg. 5, "The facilitator is responsible for breaking them down and for distributing sub-requests to the appropriate agents."). It would be inherent that since an agent can request solutions for a goal to be satisfied under a variety of different control strategies (pg. 5) that the control strategies are the advice and constraints determined for the base goal. It would also be obvious to one skilled in the art to generate program code that would entail the method of MARTIN1 and thereby obvious that the method can be entailed in a computer program product. However, MARTIN1 does not teach the step of constructing a base goal satisfaction plan.

KISS teaches an agent architecture for communicating and cooperation among distributed electronic agents (user agents / meta agents / and knowledge agents), wherein a facilitator agent (meta agent) is operable for generating / constructing a goal satisfaction plan (dynamic "solution plan") associated with the base goal (query) wherein the goal satisfaction plan includes a suitable delegation of sub-goal requests (sub-plans / tasks) to best complete the requested service request-by using domain-independent or domain –specific reasoning (col. 5, lines 14-45; col. 8, lines 21 – col. 9, line 26; col. 10, lines 10-38; col. 2, lines 50-67). Therefore, it would be obvious to combine the teachings of MARTIN1 with the teachings of KISS in order that inference be distributed and cooperative over a distributed environment (col. 3, lines 47 – col. 4, line 17).

As to claim 61, MARTIN1 teaches a facilitator agent (facilitator) arranged to coordinate task completion (process coordination) within a distributed computing environment having a plurality of electronic agents (agents / clients) according to an Interagent Communication language, comprising: an agent registry (storage of records of published capabilities of their subagents) that declares capabilities of service-providing electronic agents (subagents) currently active within the distributed computing environment and that request have constraints and parameters (control strategies) (pg. 5, The Open Agent Architecture). However, MARTIN1 does not teach the facilitating engine constructs a goal satisfaction plan.

KISS teaches an agent architecture for communicating and cooperation among distributed electronic agents (user agents / meta agents / and knowledge agents), wherein a facilitator agent (meta agent) has a facilitating engine operable to parse a service request (query) in order to interpret a compound goal (goal statement), wherein the compound goal includes local and global constraints and parameters (col. 5, lines 33 – 64; col. 8, line 32 – col. 9, line 37) and the engine further operable for generating / constructing a goal satisfaction plan (dynamic “solution plan”) associated with the base goal (query) wherein the goal satisfaction plan includes a suitable delegation of sub-goal requests (sub-plans / tasks) to best complete the requested service request-by using domain-independent or domain –specific reasoning (col. 5, lines 14-45; col. 8, lines 21 – col. 9, line 26; col. 10, lines 10-38; col. 2, lines 50-67). Therefore, it would be obvious to combine the teachings of MARTIN1 with the teachings of KISS in order that inference be distributed and cooperative over a distributed environment (col. 3, lines 47 – col. 4, line 17).

As to claim 71, reference is made to an architecture that encompasses the agent of claim 61 above, and is therefore met by the rejection of claim 61 above. However claim 71, further details the facilitator agent in bi-directional communication with the electronic agents. MARTIN1 teaches the facilitator can distribute request to the agents and the agents can request information via the facilitator (pg. 5), therefore it would be obvious that the facilitator and agents are in bi-directional communication.

As to claim 86, MARTIN1 teaches a method for information communication in a distributed computing environment having at least one facilitator agent (facilitator) and at least one client agent (sub-agent / agents), comprising storing a representation of an inter-agent language description (ICL registration of capabilities) of a client agent's functional capabilities (pg. 5, "Each facilitator records the published capabilities of their subagents.."). However, MARTIN1 does not explicitly mention that the method is operable in a data wave carrier. It would be obvious and well known in the art that one skilled in the art would generate program code on a data wave carrier that would entail the method of MARTIN1 and thereby obvious that the method can be entailed in a data wave carrier. However, MARTIN1 does not teach the facilitator agent is operable to construct a goal satisfaction plan.

KISS teaches an agent architecture for communicating and cooperation among distributed electronic agents (user agents / meta agents / and knowledge agents), wherein a facilitator agent (meta agent) is operable for generating / constructing a goal satisfaction plan (dynamic "solution plan") associated with the base goal (query) wherein the goal satisfaction plan includes a suitable delegation of sub-goal requests (sub-plans / tasks) to best complete the requested service request-by using domain-independent or domain –specific reasoning (col. 5, lines 14-45; col. 8, lines 21 – col. 9, line 26; col. 10, lines 10-38; col. 2, lines 50-67). Therefore, it would be obvious to combine the teachings of MARTIN1 with the teachings of KISS in order that inference be distributed and cooperative over a distributed environment (col. 3, lines 47 – col. 4, line 17).

As to claim 2, MARTIN1 teaches receiving a new request for service as a base goal from at least one of the selected client agents in response to the sub-goal and recursively applying the dynamically interpreting step (pg. 5, "An agent satisfying a request may require supporting information, and the OAA provides numerous means of requesting data from other agents or from the user.").

As to claim 3, MARTIN1 teaches the act of registering and transmitting the new agent profile from the specific agent to the facilitator agent (pg. 5, "Every agent participating in an OAA-based system defines and publishes a set of capabilities specifications, expressed in the ICL, describing the services that it provides."). It would be obvious that an agent that is initially created is instantiated in memory before it is registered.

As to claims 5-10, MARTIN1 teaches providing an agent registry data structure that can comprise of symbolic names, data declarations, trigger declarations, and task and process characteristics (pg. 5, "For example, every agent can install local or remote triggers on data...").

As to claim 11, MARTIN1 teaches establishing communication between distributed agents (pg. 5, ...the facilitator is responsible for breaking them down and for distributing sub-requests to the appropriate agent.").

As to claims 15-25, MARTIN1 teaches the base goal requires setting a trigger having conditional functionality and consequential functionality which can be stored on the facilitator agent and/or the service providing agent (pg. 5, "For example, every agent can install local or remote triggers on data...").

As to claims 30 and 31, MARTIN1 teaches registering a specific agent (agent) into the agent registry (list of agents capabilities) comprising: establishing a bi-directional communications link between the specific agent and a facilitator agent controlling the agent registry; providing a new agent profile to the facilitator agent; and registering the specific agent with the profile thereby making the capabilities available to the facilitator agent (pg. 5, "Each facilitator records the published capabilities of their subagents..."; "Every agent participating in an OAA-based system...describing the services that it provides.").

As to claim 32, refer to claim 3 for rejection.

As to claim 33, refer to claim 5 for rejection.

As to claim 34, refer to claim 11 for rejection.

As to claims 38-44, refer to claims 15-25 for rejection.

As to claim 62, KISS teaches the facilitating engine is capable of modifying the goal satisfaction plan during execution, the modifying initiated by events such as new agent declarations within the agent registry, decisions made by remote agents, and information provided to the facilitating engine by remote agents (col. 5, line 20-64).

As to claim 63, refer to claim 5 for rejection.

As to claim 64-69, refer to claims 15-25 for rejection.

As to claim 70, MARTIN1 teaches the agent registry (agent library / list of agent capabilities) is a database accessible to all electronic agents (pg. 5, A collection of agents satisfies requests from users, or other agents...one or more facilitators."; "An agent satisfying a request may require supporting information...requesting data from other agents or from the user.").

As to claim 87, MARTIN1 teaches a representation of a request for service in the inter-agent language from a first agent (client agent sending a query) to a second agent (facilitator) (pg. 5). It would be obvious and well known in the art that one skilled in the art would generate program code on a data wave carrier that would entail the method of MARTIN1 and KISS and thereby obvious that the method can be entailed in a data wave carrier.

As to claim 88, MARTIN1 teaches a representation of a goal dispatched to an agent for performance from a facilitator agent (every agent can request solutions for a set of goals / facilitator is responsible for breaking them down and for distributing sub-requests to the appropriate agent) (pg. 5). It would be obvious and well known in the art that one skilled in the art would generate program code on a data wave carrier that would entail the method of MARTIN1 and KISS and thereby obvious that the method can be entailed in a data wave carrier.

As to claim 89, KISS teaches a response to the dispatched goal including results from the agent for performance to the facilitator agent (col. 5, line 65 – col. 6, line 28). It would be obvious and well known in the art that one skilled in the art would generate program code on a data wave carrier that would entail the method of MARTIN1 and KISS and thereby obvious that the method can be entailed in a data wave carrier.

4. Claims 4, 12-14, 26-28, 35-37, 45-47, and 72-85 are rejected under 35 U.S.C. 103(a) as being unpatentable over MARTIN1 in view of KISS as applied to claim 1 above, and further in view of "Information Brokering in an Agent Architecture" by MARTIN2.

As to claim 4, MARTIN1 and KISS substantially disclose the invention. However, neither reference teaches the cited deactivating. MARTIN2 teaches deactivating a client agent no longer available to provide services by deleting the registration (pg. 9, Source agents that need to go offline...so that it can unregister the source and retract its schema mapping rules."). Therefore, it would be obvious to combine the teachings of MARTIN1 with the teachings of KISS and MARTIN2 in order to facilitate the transparent delegation, translation, and relaying of the appropriate subqueries to the available source agents (pg. 7-8; pg. 1).

As to claims 12-14, MARTIN1 and KISS substantially disclose the invention. However, neither reference teaches the cited receiving. MARTIN2 teaches receiving a request for service in a second language (source schema); selecting a registered agent capable of converting the second language into the inter-agent language (broker schema); and forwarding the request for service in a second language to the registered agent for conversion to be performed and the results returned (pg. 12-13, Queries Expressed in a Source Schema). Refer to claim 4 for the motivation to combine.

As to claims 26-28, MARTIN1 teaches the base goal or request is expressed in the Interagent Communication Language and is broken down such that subrequests are distributed to the appropriate agents (pg. 5). However, combination does not teach that operators including a conjunction operator or a parallel disjunction operator separate the base goal.

MARTIN2 teaches the query is a base goal stored in as a compound goal having sub-goals (pg. 8, "Queries submitted to the Broker are expression...and backtracking in expressing and processing queries.") and the ICL having expression which may be coupled by a conjunctive operator and disjunction operator (pg. 10, "Although the body of the broker predicate rule is characterized as a conjunction of predicates....Disjunction, negation..."). It would be obvious that since the base goal (query) is broken down and distributed to as sub-requests to the appropriate agents or solutions are requested for a set of goals as disclosed in MARTIN1 that the base goal is a compound goal and is broken down based on operators disclosing where it can be broken down. Refer to claim 4 for the motivation to combine.

As to claims 35-37, refer to claims 12-14 for rejection.

As to claims 45-47, refer to claims 26-28 for rejection.

As to claim 72, MARTIN1 teaches an Inter-agent Communication Language (ICL) providing a basis for facilitated cooperative task completion within a distributed computing environment having a facilitator agent (facilitator) and a plurality of electronic agents (sub-agents / agents), the ICL having a feature for allowing the enabling agents (client / agent) to perform queries, exchange information, and set triggers with other agents (pg. 5, Agents share a common communication language...and may run on any network linked platform."; pg. 5, "The Open Agent Architecture"). It is inherent that since triggers are used in order for a message to be sent to an agent, that the trigger is a conditional execution operator. However, neither MARTIN1 nor KISS teach the ICL supporting compound goal expressions from a disjunction operation.

MARTIN2 teaches the query is a base goal stored in as a compound goal having sub-goals (pg. 8, "Queries submitted to the Broker are expression...and backtracking in expressing and processing queries.") and the ICL having expression which may be coupled by a parallel disjunctive operation or conditional execution operation or conjunctive operator (pg. 10, "Disjunction, negation (that is, Prolog-style negation as failure), and a few other control operators are also allowed."). It would be obvious that since the base goal (query) is broken down and distributed to as sub-requests to the appropriate agents or solutions are requested for a set of goals as disclosed in

MARTIN1 that the base goal as a compound goal is broken down based on operators disclosing where it can be broken down. Refer to claim 4 for the motivation to combine.

As to claim 73 and 74, MARTIN1 teaches the ICL is platform and language independent (pg. 5, "The OAA's Inter-agent Communication Language...they are programmed in.").

As to claims 75-78, MARTIN1 teaches the ICL supports task completion constraints (triggers) within goal expressions (pg. 5).

As to claims 79-83, MARTIN1 teaches each electronic agent defines and publishes a set of capability declarations or solvables that describe services and an interface to the electronic agent to be stored by the facilitator agent in a registry (pg. 5, "Every agent participating in an OAA-based system defines and publishes...we refer to these capabilities specifications as solvables.").

As to claims 84 and 85, MARTIN1 and KISS substantially disclose the invention. However, neither reference teaches the cited distribution. MARTIN2 teaches that facilitator engines (broker agents) are distributed across at least two computer processes (multiple broker agents in an architecture) (pg 7, pg. 16) wherein each stores a planning component (schema mapping rules) (pg. 8). It would be obvious that since the broker performs the delegation that it also has an execution component and therefore each broker agent has an execution component. Refer to claim 4 for the motivation to combine.

5. Claims 48-60 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Development Tools for the Open Agent Architecture" by MARTIN1 in view of "Information Brokering in an Agent Architecture" by MARTIN2.

As to claim 48, MARTIN1 teaches an Inter-agent Communication Language (ICL) providing a basis for facilitated cooperative task completion within a distributed computing environment having a facilitator agent (facilitator) and a plurality of electronic agents (sub-agents / agents), the ICL having a feature for allowing the enabling agents (client / agent) to perform queries, exchange information, and set triggers with other agents (pg. 5, Agents share a common communication language...and may run on any network linked platform."; pg. 5, "The Open Agent Architecture"). It is inherent that since triggers are used in order for a message to be sent to an agent, that the trigger is a conditional execution operator. However, MARTIN1 does not teach the ICL supporting compound goal expressions from a disjunction operation.

MARTIN2 teaches the query is a base goal stored in as a compound goal having sub-goals (pg. 8, "Queries submitted to the Broker are expression...and backtracking in expressing and processing queries.") and the ICL having expression which may be coupled by a parallel disjunctive operation or conditional execution operation (pg. 10, "Disjunction, negation (that is, Prolog-style negation as failure), and a few other control operators are also allowed."). It would be obvious that since the base goal (query) is broken down and distributed to as sub-requests to the appropriate agents or solutions are requested for a set of goals as disclosed in MARTIN1 that the base goal as a compound goal is broken down based on operators disclosing where it can be broken down. Refer to claim 1 for the motivation to combine.

As to claim 49 and 50, MARTIN1 teaches the ICL is platform and language independent (pg. 5, "The OAA's Inter-agent Communication Language…they are programmed in.").

As to claims 51-54, MARTIN1 teaches the ICL supports task completion constraints (triggers) within goal expressions (pg. 5).

As to claims 55-60, MARTIN1 teaches each electronic agent defines and publishes a set of capability declarations or solvables that describe services and an interface to the electronic agent to be stored by the facilitator agent in a registry (pg. 5, "Every agent participating in an OAA-based system defines and publishes…we refer to these capabilities specifications as solvables.").

Response to Arguments

6. Applicant's arguments with respect to claims 1-86 have been considered but are moot in view of the new ground(s) of rejection.

Conclusion

7. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lewis A. Bullock, Jr. whose telephone number is (703) 305-0439. The examiner can normally be reached on Monday-Friday, 8:30 am - 5:00 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John A Follansbee can be reached on (703) 305-8498. The fax phone number for the organization where this application or proceeding is assigned is (703) 746-7239.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-0286.

lab

JOHN FOLLANSBEE
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Notice of Referenc s Cited	Application/Control No. 09/225,198	Applicant(s)/Patent Under Reexamination CHEYER ET AL.	
	Examiner Lewis A. Bullock, Jr.	Art Unit 2126	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
	A	US-6,484,155	11-2002	Kiss et al.	706/46
	B	US-6,212,649	04-2001	Yalowitz et al.	714/31
	C	US-			
	D	US-			
	E	US-			
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	Moran et al. "Multimodal User Interfaces in the Open Agent Architecture." Proceedings of the International Conference on Intelligent User Interfaces. 6-9/1997.
	V	Martin, David et al. "The Open Agent Architecture: A Framework for Buidling Distributed Software Systems." October 19, 1998.
	W	Wilkins, David et al. "Multiagent Planning Architecture." SRI International. December 8, 1997.
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

Form PTO-1449 (Modified)
(Use several sheets if necessary)

Sheet 1 of 1

MPLETE IF KNOWN	
Application Number	09/225,198
Filed	January 5, 1999
First Named Inventor	Cheyer
Group Art Unit	2151
Examiner Name	L. A. Bullock, Jr.
Atty Dkt No.	59501-8016.US01

U.S. PATENT DOCUMENTS

Examiner Initials	Cite No.	U.S. Patent or Application NUMBER	Kind Code (if known)	Name of Patentee or Inventor of Cited Document	Date of Publication or Filing Date of Cited Document	Pages, Columns, Lines, Where Relevant Figures Appear
		6,047,053		Miner et al.	04/04/00	
		6,256,771	B1	O'Neil et al.	07/03/01	
		6,411,684	B1	Cohn et al.	06/25/02	

FOREIGN PATENT DOCUMENTS

Examiner Initial	Cite No.	Foreign Patent or Application Office	NUMBER	Kind Code (if known)	Name of Patentee or Applicant of Cited Document	Date of Publication or Filing Date of Cited Document	Pages, Columns, Lines, Where Relevant Figures Appear	T

OTHER PRIOR ART-NON PATENT LITERATURE DOCUMENTS

Examiner Initials	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume issue number(s), publisher, city and/or country where published.	T

EXAMINER	DATE CONSIDERED
L A Bullock Jr	11/20/03

*EXAMINER: Initial if reference considered, whether or not criteria is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to application(s).

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/225,198	01/05/1999	ADAM J. CHEYER	SRI1P016	2756

22918 7590 03/17/2004

PERKINS COIE LLP
P.O. BOX 2168
MENLO PARK, CA 94026

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	

ART UNIT	PAPER NUMBER
2126	14

DATE MAILED: 03/17/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

Interview Summary	**Application No.** 09/225,198	**Applicant(s)** CHEYER ET AL.	
	Examin r Lewis A. Bullock, Jr.	**Art Unit** 2126	

All participants (applicant, applicant's representative, PTO personnel):

(1) *Lewis A. Bullock, Jr.*. (3)*David Stringer-Calbert*.

(2) *Corina Tan*. (4)_____.

Date of Interview: *11 March 2004*.

Type: a)☒ Telephonic b)□ Video Conference
c)□ Personal [copy given to: 1)□ applicant 2)□ applicant's representative]

Exhibit shown or demonstration conducted: d)□ Yes e)☒ No.
If Yes, brief description: _____.

Claim(s) discussed: *1-89*.

Identification of prior art discussed: *Kiss and Inventors publications*.

Agreement with respect to the claims f)☒ was reached. g)□ was not reached. h)□ N/A.

Substance of Interview including description of the general nature of what was agreed to if an agreement was reached, or any other comments: *See Continuation Sheet*.

(A fuller description, if necessary, and a copy of the amendments which the examiner agreed would render the claims allowable, if available, must be attached. Also, where no copy of the amendments that would render the claims allowable is available, a summary thereof must be attached.)

THE FORMAL WRITTEN REPLY TO THE LAST OFFICE ACTION MUST INCLUDE THE SUBSTANCE OF THE INTERVIEW. (See MPEP Section 713.04). If a reply to the last Office action has already been filed, APPLICANT IS GIVEN ONE MONTH FROM THIS INTERVIEW DATE, OR THE MAILING DATE OF THIS INTERVIEW SUMMARY FORM, WHICHEVER IS LATER, TO FILE A STATEMENT OF THE SUBSTANCE OF THE INTERVIEW. See Summary of Record of Interview requirements on reverse side or on attached sheet.

Examiner Note: You must sign this form unless it is an Attachment to a signed Office action.

Examiner's signature, if required

Summary of Record of Interview Requirements

Manual of Patent Examining Procedure (MPEP), Section 713.04, Substance of Interview Must be Made of Record
A complete written statement as to the substance of any face-to-face, video conference, or telephone interview with regard to an application must be made of record in the application whether or not an agreement with the examiner was reached at the interview.

Title 37 Code of Federal Regulations (CFR) § 1.133 Interviews
Paragraph (b)
In every instance where reconsideration is requested in view of an interview with an examiner, a complete written statement of the reasons presented at the interview as warranting favorable action must be filed by the applicant. An interview does not remove the necessity for reply to Office action as specified in §§ 1.111, 1.135. (35 U.S.C. 132)

37 CFR §1.2 Business to be transacted in writing.
All business with the Patent or Trademark Office should be transacted in writing. The personal attendance of applicants or their attorneys or agents at the Patent and Trademark Office is unnecessary. The action of the Patent and Trademark Office will be based exclusively on the written record in the Office. No attention will be paid to any alleged oral promise, stipulation, or understanding in relation to which there is disagreement or doubt.

The action of the Patent and Trademark Office cannot be based exclusively on the written record in the Office if that record is itself incomplete through the failure to record the substance of interviews.

It is the responsibility of the applicant or the attorney or agent to make the substance of an interview of record in the application file, unless the examiner indicates he or she will do so. It is the examiner's responsibility to see that such a record is made and to correct material inaccuracies which bear directly on the question of patentability.

Examiners must complete an Interview Summary Form for each interview held where a matter of substance has been discussed during the interview by checking the appropriate boxes and filling in the blanks. Discussions regarding only procedural matters, directed solely to restriction requirements for which interview recordation is otherwise provided for in Section 812.01 of the Manual of Patent Examining Procedure, or pointing out typographical errors or unreadable script in Office actions or the like, are excluded from the interview recordation procedures below. Where the substance of an interview is completely recorded in an Examiners Amendment, no separate Interview Summary Record is required.

The Interview Summary Form shall be given an appropriate Paper No., placed in the right hand portion of the file, and listed on the "Contents" section of the file wrapper. In a personal interview, a duplicate of the Form is given to the applicant (or attorney or agent) at the conclusion of the interview. In the case of a telephone or video-conference interview, the copy is mailed to the applicant's correspondence address either with or prior to the next official communication. If additional correspondence from the examiner is not likely before an allowance or if other circumstances dictate, the Form should be mailed promptly after the interview rather than with the next official communication.

The Form provides for recordation of the following information:
- Application Number (Series Code and Serial Number)
- Name of applicant
- Name of examiner
- Date of interview
- Type of interview (telephonic, video-conference, or personal)
- Name of participant(s) (applicant, attorney or agent, examiner, other PTO personnel, etc.)
- An indication whether or not an exhibit was shown or a demonstration conducted
- An identification of the specific prior art discussed
- An indication whether an agreement was reached and if so, a description of the general nature of the agreement (may be by attachment of a copy of amendments or claims agreed as being allowable). Note: Agreement as to allowability is tentative and does not restrict further action by the examiner to the contrary.
- The signature of the examiner who conducted the interview (if Form is not an attachment to a signed Office action)

It is desirable that the examiner orally remind the applicant of his or her obligation to record the substance of the interview of each case. It should be noted, however, that the Interview Summary Form will not normally be considered a complete and proper recordation of the interview unless it includes, or is supplemented by the applicant or the examiner to include, all of the applicable items required below concerning the substance of the interview.

A complete and proper recordation of the substance of any interview should include at least the following applicable items:
1) A brief description of the nature of any exhibit shown or any demonstration conducted,
2) an identification of the claims discussed,
3) an identification of the specific prior art discussed,
4) an identification of the principal proposed amendments of a substantive nature discussed, unless these are already described on the Interview Summary Form completed by the Examiner,
5) a brief identification of the general thrust of the principal arguments presented to the examiner,
(The identification of arguments need not be lengthy or elaborate. A verbatim or highly detailed description of the arguments is not required. The identification of the arguments is sufficient if the general nature or thrust of the principal arguments made to the examiner can be understood in the context of the application file. Of course, the applicant may desire to emphasize and fully describe those arguments which he or she feels were or might be persuasive to the examiner.)
6) a general indication of any other pertinent matters discussed, and
7) if appropriate, the general results or outcome of the interview unless already described in the Interview Summary Form completed by the examiner.

Examiners are expected to carefully review the applicant's record of the substance of an interview. If the record is not complete and accurate, the examiner will give the applicant an extendable one month time period to correct the record.

Examiner to Check for Accuracy

If the claims are allowable for other reasons of record, the examiner should send a letter setting forth the examiner's version of the statement attributed to him or her. If the record is complete and accurate, the examiner should place the indication, "Interview Record OK" on the paper recording the substance of the interview along with the date and the examiner's initials.

Continuation of Substance of Interview including description of the general nature of what was agreed to if an agreement was reached, or any other comments: Applicants argued that the prior art teachings of Kiss did not accomplish the inventors goal of the faciliatator agent using the goal satisfaction plan that stored the intelligence of the order of the sub-goals since Kiss teaches that the solution plan can be dynamically modifed. The examiner alluded that the claims make no mention that the solution plan cannot be modified and that Kiss's solution plan accomplishes the limitations of the claims as disclosed. The examiner pointed out that all the rejections regarding this application were made with publications written by the Applicants. The examiner pointed out that there are limitations in the specification regarding the Interagent Communication Language that were not disclosed in any of the inventors publications that can distinguish the claims from the prior art of record. In particular, the examiner pointed to page 17, lines 7-11 which describe the ICL as including a layer of conversational protocol and a content layer that distinguish the claims from any teaching disclosed in the publications. The examiner also pointed out that this teaching distinguishes the Applicant's interagent communication language from the well known communication language KQML. Applicants will submit a response amending the claims to the examiners suggestions. The interview concluded..

EXPRESS MAIL LABEL NO. EV 099152888 US

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Atty Dkt. No. 59501-8016.US01
CHEYER et al.	Group Art Unit No.: 2126
Serial No.: 09/225,198	Examiner: L. A. Bullock, Jr.
Filed on: January 5, 1999	

For: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

Mail Stop AF
Commissioner of Patents
P. O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT AND RESPONSE

Sir:

This is in response to the Final Office Action mailed November 28, 2003, the shortened statutory period for which runs until February 28, 2004.

IN THE CLAIMS

1. (Currently amended) A computer-implemented method for communication and cooperative task completion among a plurality of distributed electronic agents, comprising the acts of:

registering a description of each active client agent's functional capabilities as corresponding registered functional capabilities, using an expandable, platform-independent, inter-agent language, wherein the inter-agent language includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

receiving a request for service as a base goal in the inter-agent language, in the form of an arbitrarily complex goal expression; and

dynamically interpreting the arbitrarily complex goal expression, said act of interpreting further comprising:

generating one or more sub-goals expressed in the inter-agent language;

constructing a goal satisfaction plan wherein the goal satisfaction plan includes:

a suitable delegation of sub-goal requests to best complete the requested service request-by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms; and

dispatching each of the sub-goals to a selected client agent for performance, based on a match between the sub-goal being dispatched and the registered functional capabilities of the selected client agent.

2. (Previously presented) A computer-implemented method as recited in claim 1, further including the following acts of:

receiving a new request for service as a base goal using the inter-agent language, in the form of another arbitrarily complex goal expression, from at least one of the selected client agents in response to the sub-goal dispatched to said agent; and recursively applying the step of dynamically interpreting the arbitrarily complex goal expression in order to perform the new request for service.

3. (Previously presented) A computer-implemented method as recited in claim 2 wherein the act of registering a specific agent further includes:
invoking the specific agent in order to activate the specific agent;
instantiating an instance of the specific agent; and
transmitting the new agent profile from the specific agent to a facilitator agent in response to the instantiation of the specific agent.

4. (original) A computer-implemented method as recited in claim 1 further including the act of deactivating a specific client agent no longer available to provide services by deleting the registration of the specific client agent.

5. original) A computer-implemented method as recited in claim 1 further comprising the act of providing an agent registry data structure.

6. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one symbolic name for each active agent.

7. (original) A computer-implemented method of recited in claim 5 wherein the agent registry data structure includes at least one data declaration for each active agent.

8. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one trigger declaration for one active agent.

9. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one task declaration, and process characteristics for each active agent.

10. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one process characteristic for each active agent.

11. (original) A computer-implemented method as recited in claim 1 further comprising the act of establishing communication between the plurality of distributed agents.

12. (original) A computer-implemented method as recited in claim 1 further comprising the acts of:
receiving a request for service in a second language differing from the inter-agent language;
selecting a registered agent capable of converting the second language into the inter-agent language; and
forwarding the request for service in a second language to the registered agent capable of converting the second language into the inter-agent language, implicitly requesting that such a conversion be performed and the results returned.

13. (original) A computer-implemented method as recited in claim 12 wherein the request includes a natural language query, and the registered agent capable of converting the second language into the inter-agent language service is a natural language agent.

14. (original) A computer-implemented method as recited in claim 13 wherein the natural language query was generated by a user interface agent.

15. (original) A computer-implemented method as recited in claim 1, wherein the base goal requires setting a trigger having conditional functionality and consequential functionality.

16. (original) A computer-implemented method as recited in claim 15 wherein the trigger is an outgoing communications trigger, the computer implemented method further including the acts of:

monitoring all outgoing communication events in order to determine whether a specific outgoing communication event has occurred; and

in response to the occurrence of the specific outgoing communication event, performing the particular action defined by the trigger.

17. (original) A computer-implemented method as recited in claim 15 wherein the trigger is an incoming communications trigger, the computer implemented method further including the acts of:

monitoring all incoming communication events in order to determine whether a specific incoming communication event has occurred; and

in response to the occurrence of a specific incoming communication event satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

18. (original) A computer-implemented method as recited in claim 15 wherein the trigger is a data trigger, the computer implemented method further including the acts of:

monitoring a state of a data repository; and

in response to a particular state event satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

19. (original) A computer-implemented method as recited in claim 15 wherein the trigger is a time trigger, the computer implemented method further including the acts of:

monitoring for the occurrence of a particular time condition; and

in response to the occurrence of a particular time condition satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

20. (original) A computer-implemented method as recited in claim 15 wherein the trigger is installed and executed within the facilitator agent.

21. (original) A computer-implemented method as recited in claim 15 wherein the trigger is installed and executed within a first service-providing agent.

22. (original) A computer-implemented method as recited in claim 15 wherein the conditional functionality of the trigger is installed on a facilitator agent.

23. (original) A computer-implemented method as recited in claim 22 wherein the consequential functionality is installed on a specific service-providing agent other than a facilitator agent.

24. (original) A computer-implemented method as recited in claim 15 wherein the conditional functionality of the trigger is installed on specific service-providing agent other than a facilitator agent.

25. (original) A computer-implemented method as recited in claim 15 wherein the consequential functionality of the trigger is installed on a facilitator agent.

26. (original) A computer-implemented method as recited in claim 1 wherein the base goal is a compound goal having sub-goals separated by operators.

27. (original) A computer-implemented method as recited in claim 26 wherein the type of available operators includes a conjunction operator, a disjunction operator, and a conditional execution operator.

28. (original) A computer-implemented method as recited in claim 27 wherein the type of available operators further includes a parallel disjunction operator that indicates that disjunct goals are to be performed by different agents.

29. (Currently amended) A computer program stored on a computer readable medium, the computer program executable to facilitate cooperative task completion within a distributed computing environment, the distributed computing environment including a plurality of autonomous electronic agents, the distributed computing environment supporting an Interagent Communication Language, the computer program comprising computer executable instructions for:

providing an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment;
interpreting a service request in order to determine a base goal that may be a compound, arbitrarily complex base goal, the service request adhering to an Interagent Communication Language (ICL), wherein the ICL includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

the act of interpreting including the sub-acts of:

determining any task completion advice provided by the base goal, and

determining any task completion constraints provided by the base goal;

constructing a base goal satisfaction plan including the sub-acts of:

determining whether the requested service is available,

determining sub-goals required in completing the base goal by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms,

selecting service-providing electronic agents from the agent registry suitable for performing the determined sub-goals, and

ordering a delegation of sub-goal requests to best complete the requested service; and

implementing the base goal satisfaction plan.

30. (original) A computer program as recited in claim 29 wherein the computer executable instruction for providing an agent registry includes the following computer executable instructions for registering a specific service-providing electronic agent into the agent registry:
establishing a bi-directional communications link between the specific agent and a facilitator agent controlling the agent registry;
providing a new agent profile to the facilitator agent, the new agent profile defining publicly available capabilities of the specific agent; and

registering the specific agent together with the new agent profile within the agent registry, thereby making available to the facilitator agent the capabilities of the specific agent.

31. (original) A computer program as recited in claim 30 wherein the computer executable instruction for registering a specific agent further includes:
invoking the specific agent in order to activate the specific agent;
instantiating an instance of the specific agent; and
transmitting the new agent profile from the specific agent to the facilitator agent in response to the instantiation of the specific agent.

32. (original) A computer program as recited in claim 29 wherein the computer executable instruction for providing an agent registry includes a computer executable instruction for removing a specific service-providing electronic agent from the registry upon determining that the specific agent is no longer available to provide services.

33. (original) A computer program as recited in claim 29 wherein the provided agent registry includes a symbolic name, a unique address, data declarations, trigger declarations, task declarations, and process characteristics for each active agent.

34. (original) Computer program as recited in claim 29 further including computer executable instructions for receiving the service request via a communications link established with a client.

35. (original) A computer program as recited in claim 29 wherein the computer executable instruction for providing a service request includes instructions for:
receiving a non-ICL format service request;
selecting an active agent capable of converting the non-ICL formal service request into an ICL format service request;
forwarding the non-ICL format service request to the active agent capable of converting the non-ICL format service request, together with a request that such conversion be performed; and

receiving an ICL format service request corresponding to the non-ICL format service request.

36. (original) A computer program as recited in claim 35 wherein the non-ICL format service request includes a natural language query, and the active agent capable of converting the non-ICL formal service request into an ICL format service request is a natural language agent.

37. (original) A computer program as recited in claim 36 wherein the natural language query is generated by a user interface agent.

38. (original) A computer program as recited in claim 29, the computer program further including computer executable instructions for implementing a base goal that requires setting a trigger having conditional and consequential functionality.

39. (original) A computer program as recited in claim 38 wherein the trigger is an outgoing communications trigger, the computer program further including computer executable instructions for:
monitoring all outgoing communication events in order to determine whether a specific outgoing communication event has occurred; and
in response to the occurrence of the specific outgoing communication event, performing the particular action defined by the trigger.

40. (original) A computer program as recited in claim 38 wherein the trigger is an incoming communications trigger, the computer program further including computer executable instructions for:
monitoring all incoming communication events in order to determine whether a specific incoming communication event has occurred; and
in response to the occurrence of the specific incoming communication event, performing the particular action defined by the trigger.

41. (original) A computer program as recited in claim 38 wherein the trigger is a data trigger, the computer program further including computer executable instructions for:
monitoring a state of a data repository; and
in response to a particular state event, performing the particular action defined by the trigger.

42. (original) A computer program as recited in claim 38 wherein the trigger is a time trigger, the computer program further including computer executable instructions for:
monitoring for the occurrence of a particular time condition; and
in response to the occurrence of the particular time condition, performing the particular action defined by the trigger.

43. (original) A computer program as recited in claim 38 further including computer executable instructions for installing and executing the trigger within the facilitator agent.

44. (original) A computer program as recited in claim 38 further including computer executable instructions for installing and executing the trigger within a first service-providing agent.

45. (original) A computer program as recited in claim 29 further including computer executable instructions for interpreting compound goals having sub-goals separated by operators.

46. (original) A computer program as recited in claim 45 wherein the type of available operators includes a conjunction operator, a disjunction operator, and a conditional execution operator.

47. (original) A computer program as recited in claim 46 wherein the type of available operators further includes parallel disjunction operator that indicates that distinct goals are to be performed by different agents.

48. (Currently amended) An Interagent Communication Language (ICL) providing a basis for facilitated cooperative task completion within a distributed computing environment having a facilitator agent and a plurality of autonomous service-providing electronic agents, wherein:

the ICL having one or more of:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

the ICL having one or more features from a set of features comprising:

enabling agents to perform queries of other agents;

enabling agents to exchange information with other agents; and

enabling agents to set triggers within other agents; and

the ICL having a syntax supporting compound goal expressions wherein said compound goal expressions are such that goals within a single request provided according to the ICL syntax may be coupled by one or more operators from a set of operators comprising:

a conditional execution operator; and

a parallel disjunctive operation that indicates that disjunct goals are to be performed by different agents.

49. (original) An ICL as recited in claim 48, wherein the ICL is computer platform independent.

50. (original) An ICL as recited in claim 48 wherein the ICL is independent of computer programming languages which the plurality of agents are programmed in.

51. (original) An ICL as recited in claim 48 wherein the ICL syntax supports explicit task completion constraints include use of specific agent constraints and response time constraints.

52. (original) An ICL as recited in claim 51, wherein possible types of task completion constraints include use of specific agent constraints and response time constraints.

53. (original) An ICL as recited in claim 51 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

54. (original) An ICL as recited in claim 48 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

55. (original) An ICL as recited in claim 48 wherein each autonomous service-providing electronic agent defines and publishes a set of capability declarations or solvables, expressed in ICL, that describes services provided by such electronic agent.

56. (original) An ICL as recited in claim 55 wherein an electronic agent's solvables define an interface for the electronic agent.

57. (original) An ICL as recited in claim 56 wherein the facilitator agent maintains an agent registry making available a plurality of electronic agent interfaces.

58. (original) An ICL as recited in claim 57 wherein the possible types of solvables includes procedure solvables, a procedure solvable operable to implement a procedure such as a test or an action.

59. (original) An ICL as recited in claim 58 wherein the possible types of solvables further includes data solvables, a data solvable operable to provide access to a collection of data.

60. (original) An ICL as recited in claim 58 wherein the possible types of solvables includes data solvables, a data solvable operable to provide access to a collection of data.

61. (Currently amended) A facilitator agent arranged to coordinate cooperative task completion within a distributed computing environment having a plurality of autonomous service-providing electronic agents, the facilitator agent comprising:

an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment; and

a facilitating engine operable to parse a service request in order to interpret a compound goal set forth therein, the compound goal including both local and global constraints and control parameters, the service request formed according to an Interagent Communication Language (ICL), wherein the ICL includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

the facilitating engine further operable to construct a goal satisfaction plan by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms.

62. (original) A facilitator agent as recited in claim 61, wherein the facilitating engine is capable of modifying the goal satisfaction plan during execution, the modifying initiated by events such as new agent declarations within the agent registry, decisions made by remote agents, and information provided to the facilitating engine by remote agents.

63. (original) A facilitator agent as recited in claim 61 wherein the agent registry includes a symbolic name, a unique address, data declarations, trigger declarations, task declarations, and process characteristics for each active agent.

64. (original) A facilitator agent as recited in claim 61 wherein the facilitating engine is operable to install a trigger mechanism requesting that a certain action be taken when a certain set of conditions are met.

65. (original) A facilitator agent as recited in claim 64 wherein the trigger mechanism is a communication trigger that monitors communication events and performs the certain action when a certain communication event occurs.

66. (original) A facilitator agent as recited in claim 64 wherein the trigger mechanism is a data trigger that monitors a state of a data repository and performs the certain action when a certain data state is obtained.

67. (original) A facilitator agent as recited in claim 66 wherein the data repository is local to the facilitator agent.

68. (original) A facilitator agent as recited in claim 66 wherein the data repository is remote from the facilitator agent.

69. (original) A facilitator agent as recited in claim 64 wherein the trigger mechanism is a task trigger having a set of conditions.

70. (original) A facilitator agent as recited in claim 61, the facilitator agent further including a global database accessible to at least one of the service-providing electronic agents.

71. (Currently amended) A software-based, flexible computer architecture for communication and cooperation among distributed electronic agents, the architecture contemplating a distributed computing system comprising:
a plurality of service-providing electronic agents; ~~and~~
an Interagent Communication Language (ICL), wherein the inter-agent language includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and
a content layer comprising one or more of goals, triggers and data elements associated with the events; and

a facilitator agent in bi-directional communications with the plurality of service-providing electronic agents, the facilitator agent including:

an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment;

a facilitating engine operable to parse a service request in order to interpret an arbitrarily complex goal set forth therein, the facilitating engine further operable to construct a goal satisfaction plan including the coordination of a suitable delegation of sub-goal requests to best complete the requested service by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms.

72. (Currently amended) A computer architecture as recited in claim 71, wherein the ~~basis for the computer architect is an~~ Interagent Communication Language (ICL) is for enabling agents to perform queries of other agents, exchange information with other agents, and set triggers within other agents, the ICL further defined by an ICL syntax supporting compound goal expressions such that goals within a single request provided according to the ICL syntax may be coupled by a conjunctive operator, a disjunctive operator, a conditional execution operator, and a parallel disjunctive operator parallel disjunctive operator that indicates that disjunct goals are to be performed by different agents.

73. (original) A computer architecture as recited in claim 72, wherein the ICL is computer platform independent.

74. (original) A computer architecture as recited in claim 73 wherein the ICL is independent of computer programming languages in which the plurality of agents are programmed.

75. (original) A computer architecture as recited in claim 73 wherein the ICL syntax supports explicit task completion constraints within goal expressions.

76. (original) A computer architecture as recited in claim 75 wherein possible types of task completion constraints include use of specific agent constraints and response time constraints.

77. (original) A computer architecture as recited in claim 75 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

78. (original) A computer architecture as recited in claim 73 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

79. (original) A computer architecture as recited in claim 73 wherein each autonomous service-providing electronic agent defines and publishes a set of capability declarations or solvables, expressed in ICL, that describes services provided by such electronic agent.

80. (original) A computer architecture as recited in claim 79 wherein an electronic agent's solvables define an interface for the electronic agent.

81. (original) A computer architecture as recited in claim 80 wherein the possible types of solvables includes procedure solvables, a procedure solvable operable to implement a procedure such as a test or an action.

82. (original) A computer architecture as recited in claim 81 wherein the possible types of solvables further includes data solvables, a data solvable operable to provide access to a collection of data.

83. (original) A computer architecture as recited in claim 82 wherein the possible types of solvables includes a data solvable operable to provide access to modify a collection of data.

84. (Previously presented) A computer architecture as recited in claim 71 wherein a planning component of the facilitating engine are distributed across at least two computer processes.

85. (Previously presented) A computer architecture as recited in claim 71 wherein an execution component of the facilitating engine is distributed across at least two computer processes.

86. (Currently amended) A data wave carrier providing a transport mechanism for information communication in a distributed computing environment having at least one facilitator agent and at least one active client agent, and an Interagent Communication Language (ICL), wherein the ICL includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

wherein said at least one facilitator agent is operable to construct a goal satisfaction plan by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms for satisfying one or more requests for service from said at least one active client agent, the data wave carrier comprising a signal representation of an inter-agent language description of an active client agent's functional capabilities.

87. (Previously presented) A data wave carrier as recited in claim 86, the data wave carrier further comprising a corresponding signal representation of said one or more requests for service in the inter-agent language from a first agent to a second agent.

88. (Previously presented) A data wave carrier as recited in claim 86, the data wave carrier further comprising a signal representation of a goal dispatched to an agent for performance from a facilitator agent.

89. (original) A data wave carrier as recited in claim 88 wherein a later state of the data wave carrier comprises a signal representation of a response to the dispatched goal including results and/or a status report from the agent for performance to the facilitator agent.

REMARKS

INTERVIEW:

A telephonic interview was conducted on March 11, 2004. The participants were Examiner Lewis A. Bullock, Jr., David Stringer-Calvert and Carina M. Tan. During the interview, an agreement with respect to all the claims were reached. Applicants argued that the prior art teachings of *KISS* did not disclose any intelligent reasoning when formulating a goal satisfaction plan. Applicants argued that *KISS* merely discloses a method of information retrieval from information repositories such as databases. The examiner disagreed. However, the examiner pointed out that certain features in Applicant's specification regarding ICL are novel. The Examiner indicated that the ICL features: 1) a conversational protocol layer, and 2) a content layer, would distinguish applicants' claims over the prior art. It was agreed that applicants would submit a response amending the claims to include the above novel ICL features.

The Examiner is thanked for the performance of a thorough search. By this response, claims 1, 29, 48, 61, 71, 72 and 86 have been amended. No claims have been cancelled or added. Hence, Claims 1-89 are pending in the Application.

IN THE SPECIFICATION

Compact Disc Containing Appendices

Applicants cancel the computer program listing appearing in the specification in Appendices A, B, C, D, and E. In compliance with 37 CFR 1.96(c), Applicants enclose a CD-ROM labeled as Copy 1 and an identical copy of the CD-ROM labeled as Copy 2 containing the identical contents of Appendices A, B, C, D and E as filed with the patent application on January 5, 1999.

Substitute Pages Of Specification

Enclosed are substitute Pages 1, 8 and 9. Substitute Page 1 of the specification has been amended to identify the compact disc and list the file names, size, and creation date of each file, and substitute Page 8 and Page 9 which have been amended to delete the "Brief Description of the Appendices." Also enclosed is a substitute ABSTRACT containing less than 150 words. The ABSTRACT as originally filed contained more than 150 words.

SUMMARY OF REJECTIONS/OBJECTIONS

In the Office Action, Claims 1-3, 5-11, 15-25, 29-34, 38-44, and 61-71 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Developing Tools for the Open Agent Architecture" by Martin1 in view of U.S. Patent No. 6,484,155 issued to Kiss.

Claims 4, 12-14, 26-28, 35-37, 45-47, and 72-85 are rejected under 35 U.S.C. 103(a) as being unpatentable over Martin1 in view of Kiss, and further in vie of "Information Brokering in an Agent Architecture" by Martin2.

Claims 48-60 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Development Tools for the Open Agent Architecture" by Martin1 in view of "Information Brokering in an Agent Architecture" by Martin2.

REJECTIONS UNDER 35 U.S.C. § 103(a)

CLAIMS 1, 29, 61, 71 and 86

Claim 1, as amended, recites in part, the features:

> "registering a description of each active client agent's functional capabilities as corresponding registered functional capabilities, using an expandable,

platform-independent, inter-agent language, **wherein the inter-agent language includes:**

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

constructing a goal satisfaction plan, wherein the goal satisfaction plan includes:

a suitable delegation of sub-goal requests to best complete the requested service request by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms;"

Claim 1 includes the limitation of a inter-agent language, wherein the inter-agent language includes 1) a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, and 2) a content layer comprising one or more of goals, triggers and data elements associated with the events. The cited references do not disclose or suggest such a conversational protocol and content layer.

Further, the Office Action states that the "dynamic solution plan" in *KISS* is the equivalent of the "goal satisfaction plan" of applicants' Claim 1 above. The Office Action points to col. 5, lines 14-45; col. 8, line 21 - col. 9, line 26; and col. 10, lines 10-38, and col. 2, lines 50-67 for support.

The method for forming the "dynamic solution plan" in *KISS* is irrelevant to the method of forming the goal satisfaction plan in Applicants' Claim 1. It is respectfully submitted that *KISS* is irrelevant because *KISS* is an invention involving accessing knowledge repositories. Such knowledge repositories are represented by "knowledge agents." The Abstract of *KISS* states that "the invention solicits accessible knowledge repositories, represented by knowledge agents, for relevant knowledge..."

In other words, *KISS* is merely a method of information retrieval from information repositories or data sources. For example, the meta agent can ask questions involving facts or data and the agents attempt to retrieve the facts or data from the corresponding data repository. In contrast, the goal satisfaction plan of Claim 1 involves asking service providing agents to perform **actions** such as boil water, roast coffee beans, grind the roasted coffee beans as opposed to merely asking the agents to retrieve information from an information repository.

To further explain why *KISS* is irrelevant and completely different from the method of Claim 1, see col. 5 lines 39-43 where "[t]he meta agent 119 is configured to begin executing the solution plan even before the plan is complete." This underscores the fact that the solution plan in *KISS* merely involves information retrieval rather than asking the agent to perform intelligent actions such as roast coffee beans. In *KISS*, it is not fatal to begin executing the solution plan even before the plan is complete because no real harm is done if the meta agent begins by asking the wrong questions. To explain, *KISS* teaches "the meta agent 119 is capable of backtracking or replanning to permit escape from a dead-end." In other words, it is not fatal if the search for data is proceeding down an incorrect search path, as explained in *KISS*. In contrast, the facilitator of Claim 1 cannot begin execution of the goal satisfaction plan before the goal satisfaction plan is complete. For example, it would be fatal for the facilitator to ask a service-providing agent to boil the coffee beans instead of requesting that the coffee beans be first roasted and then ground. Such an action of boiling the coffee beans would be **irreversible** and would produce soggy beans. In other words, the service-providing agents of Claim 1 perform actions and are not merely sources of information.

Further, *KISS* does not use reasoning for "formulating the dynamic solution plan." In other words, *KISS* does not use the inferencing schemes as described in column 7 for generating the solution plan. In fact, *KISS* teaches away from using reasoning or inferencing for generating the solution plan. Column 8, lines 58-61 of *KISS* states that "**[a]fter** the solution plan is formulated, the meta agent 119 implements a distributed inference process to perform the search and execution phases of solving the problem, while maintaining control of the process" (emphasis added). Thus, the inference process is what the solution plan in *KISS* accomplishes and is not what is used to generate the solution plan.

In contrast, Claim 1 shows that the facilitating engine uses sophisticated reasoning when delegating sub-goal requests to best complete the requested service request. The facilitating engine's use of reasoning is supported by the specification on page 13, lines 342-347.

Assume that the facilitator agent of Claim 1 receives a request such as, "Make Coffee". The facilitator agent's facilitating engine uses reasoning to generate the following goal satisfaction plan:

Sub-goal request A: Please perform the act of roasting coffee beans
Sub-goal request B: Please perform the act of grinding coffee beans
Sub-goal request C: Please perform the act of boiling water, etc.

The facilitating engine is able to use reasoning to accomplish the base goal, "Make Coffee" by asking an appropriate agents to first roast the coffee beans before asking the agent to grind the beans, etc.

Neither *Cohen* nor *KISS*, either alone or in combination, disclose, teach, suggest or make obvious the novel features of claim 1. Thus, Claim 1 is allowable.

Claims 29, 61, 71 and 86, each contain similar features regarding "using reasoning to determine sub-goal requests based on non-syntactic decomposition of the base goal and using said reasoning to co-ordinate and schedule efforts by the service-providing electronic agents for fulfilling the sub-goal requests in a cooperative completion of the base goal." Thus, Claims 29, 61, 71 and 86 are allowable for at least the reasons provided herein in respect to Claim 1.

CLAIMS 2-28, 30-47, 62-70, 72-85 and 87-89

Claims 2-28 are either directly or indirectly dependent upon Claim 1 and include all the limitations of Claim 1 and therefore are allowable for at least the reasons provided herein in respect to Claim 1.

Claims 30-47 are either directly or indirectly dependent upon Claim 29 and include all the limitations of Claim 29 and therefore are allowable for at least the reasons provided herein in respect to Claim 29.

Claims 62-70 are either directly or indirectly dependent upon Claim 61 and include all the limitations of Claim 61 and therefore are allowable for at least the reasons provided herein in respect to Claim 61.

Claims 72-85 are either directly or indirectly dependent upon Claim 71 and include all the limitations of Claim 71 and therefore are allowable for at least the reasons provided herein in respect to Claim 71

Claims 87-89 are either directly or indirectly dependent upon Claim 86 and include all the limitations of Claim 86 and therefore are allowable for at least the reasons provided herein in respect to Claim 86.

CLAIM 48

Claim 48 as amended, recites in part:

"the ICL having one or more of:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and
a content layer comprising one or more of goals, triggers and data elements associated with the events;

the ICL having a syntax supporting compound goal expressions wherein said compound goal expressions are such that goals within a single request provided according to the ICL syntax may be coupled by one or more operators from a set of operators comprising:

a conditional execution operator; and
a parallel disjunctive operator that indicates that disjunct goals are to be performed by different agents."

The novel method recited in Claim 48 as amended requires that the inter-agent language include 1) a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, and 2) a content layer comprising one or more of goals, triggers and data elements associated with the events. The cited references do not disclose or suggest such a conversational protocol and content layer.

Further, the novel method recited in Claim 48 as amended requires that "goals within a single request" are "coupled by one or more operators from a set of operators". In amended Claim 48, the set of operators comprise, a conditional execution operator, and **a parallel disjunctive operator**.

In the Office Action, the Examiner states that triggers are conditional operators. It is respectfully submitted that triggers are not conditional operators in the sense of an being a syntactical operator in an expression.

Further, the Office Action states that page 10 of *Martin2* discloses **parallel disjunctive operators**. *Martin2* does NOT disclose parallel disjunctive operators. The "disjunction" in *Martin2* is the run-of-the-mill Prolog style disjunction. The expression, "Do task A OR Do Task B," is an example of a *Martin2* type disjunction. In contrast, a "**parallel disjunctive operator** is an operator that indicates that disjunct goals are to be performed by different agents. An example of a **parallel disjunctive operator** expression is "Ask agent Bob to do task A OR Ask agent Fred to do task B concurrently.

None of the cited references disclose, suggest or render obvious the requirement that the "**goals within a single request**" be "coupled by one or more operators from a set of operators", such as **a conditional execution operator** (such as "if" and "when", allowing for particular actions to be predicated on the state, or outcomes of earlier actions), and **a parallel disjunctive operator** (allowing for alternative actions to be performed at the same time, if resources allow, and a first-to-respond strategy may be used in their competition to perform the goal at hand). Claim 48 is allowable over the art of record. Thus, it is respectfully submitted that Claim 48 be held in condition for allowance.

CLAIMS 49-60

Claims 49-60 are either directly or indirectly dependent upon independent Claim 48, and include all the features of Claim 48. Therefore, Claims 49-60 are allowable for at least the reasons provided herein with respect to Claim 48. Furthermore, it is respectfully submitted that Claims 49-60 recite additional features that independently render Claims 49-60 patentable over the art of record. Thus, it is respectfully submitted that Claims 49-60 be held in condition for allowance.

CONCLUSION

For the reasons set forth above, it is respectfully submitted that all of the pending claims are now in condition for allowance. Therefore, the issuance of a formal Notice of Allowance is believed next in order, and that action is most earnestly solicited.

If in the opinion of the Examiner a telephone conference would expedite the prosecution of the subject application, the Examiner is encouraged to call the undersigned at (650) 838-4311.

The Commissioner is authorized to charge any fees due to Applicants' Deposit Account No. 50-2207.

Respectfully submitted,
Perkins Coie LLP

Date: March 29 , 2004

Carina M. Tan

Carina M. Tan
Registration No. 45,769

Correspondence Address:

Customer No. 22918
Perkins Coie LLP
P. O. Box 2168
Menlo Park, California 94026
(650) 838-4300

Software-Based Architecture for Communication and Cooperation Among Distributed Electronic Agents

By:

Adam J. Cheyer and David L. Martin

A compact disk containing a computer program listing has been provided in duplicate (copy 1 and copy 2 of the compact disk are identical). The computer program listing in the compact disk is incorporated by reference herein. The compact disk contains files with their names, size and date of creation as follow:

File Name	Size	Creation Date	Last Date
oaa.pl	159,613 bytes	1996/10/08	1998/12/23
fac.pl	52,733 bytes	1997/04/24	1998/05/06
compound.pl	42,937 bytes	1996/12/11	1998/04/10
com_tcp.pl	18,010 bytes	1998/02/10	1998/05/06
translations.pl	19,583 bytes	1998/01/29	1998/12/23

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention is related to distributed computing environments and the completion of tasks within such environments. In particular, the present invention teaches a variety of software-based architectures for communication and cooperation among distributed electronic agents. Certain embodiments teach interagent communication languages enabling client agents to make requests in the form of arbitrarily complex goal expressions that are solved through facilitation by a facilitator agent.

Context and Motivation for Distributed Software Systems

The evolution of models for the design and construction of distributed software systems is being driven forward by several closely interrelated trends: the adoption of a *networked computing model*, rapidly rising expectations for *smarter, longer-lived, more autonomous software applications* and an ever increasing demand for *more accessible and intuitive user interfaces*.

Prior Art Figure 1 illustrates a *networked computing model* 100 having a plurality of client and server computer systems 120 and 122 coupled together over a physical transport mechanism 140. The adoption of the *networked computing model* 100 has lead to a greatly increased reliance on distributed sites for both data and processing resources. Systems such as the networked computing model 100 are based upon at least one physical transport mechanism 140 coupling the multiple computer systems 120 and 122 to support the transfer of information between these computers.

Some of these computers basically support using the network and are known as *client*

FIGURE 9 depicts operations involved in a client agent initiating a service request and receiving the response to that service request in accordance with a certain preferred embodiment of the present invention;

FIGURE 10 depicts operations involved in a client agent responding to a service request in accordance with another preferable embodiment of the present invention;

FIGURE 11 depicts operations involved in a facilitator agent response to a service request in accordance with a preferred embodiment of the present invention;

FIGURE 12 depicts an Open Agent Architecture^{TM} based system of agents implementing a unified messaging application in accordance with a preferred embodiment of the present invention;

FIGURE 13 depicts a map oriented graphical user interface display as might be displayed by a multi-modal map application in accordance with a preferred embodiment of the present invention;

FIGURE 14 depicts a peer to peer multiple facilitator based agent system supporting distributed agents in accordance with a preferred embodiment of the present invention;

FIGURE 15 depicts a multiple facilitator agent system supporting at least a limited form of a hierarchy of facilitators in accordance with a preferred embodiment of the present invention; and

FIGURE 16 depicts a replicated facilitator architecture in accordance with one embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

Figure 3 illustrates a distributed agent system 300 in accordance with one embodiment of the present invention. The agent system 300 includes a facilitator agent 310 and a plurality of agents 320. The illustration of Figure 3 provides a high level view of one simple system structure contemplated by the present invention. The facilitator agent 310 is in essence the "parent" facilitator for its "children" agents 320. The agents 320 forward service requests to the facilitator agent 310. The facilitator agent 310 interprets these requests, organizing a set of goals which are then delegated to appropriate agents for task completion.

The system 300 of Figure 3 can be expanded upon and modified in a variety of ways consistent with the present invention. For example, the agent system 300 can be distributed across a computer network such as that illustrated in Figure 1. The facilitator agent 310 may itself have its functionality distributed across several different computing platforms. The agents 320 may engage in interagent communication (also called peer to peer communications). Several different systems 300 may be coupled together for enhanced performance. These and a variety of other structural configurations are described below in greater detail.

Figure 4 presents the structure typical of a small system 400 in one embodiment of the present invention, showing user interface agents 408, several application agents 404 and meta-agents 406, the system 400 organized as a community of peers by their common relationship to a facilitator agent 402. As will be appreciated, Figure 4 places more structure upon the system 400 than shown in Figure 3, but both are valid representations of structures of the present invention. The facilitator 402 is a specialized server agent that is responsible for coordinating agent communications and cooperative problem-solving. The facilitator 402 may also provide a global data store for its client agents, allowing them to adopt a blackboard style of interaction. Note that certain advantages are found in utilizing two or more facilitator agents within the system 400. For example, larger systems can be assembled from multiple facilitator/client groups, each having the sort of structure

ABSTRACT

A highly flexible, software-based architecture is disclosed for constructing distributed systems. The architecture supports cooperative task completion by flexible and autonomous electronic agents. One or more facilitators are used to broker communication and cooperation among the agents. The architecture provides for the construction of arbitrarily complex goals by users and service-requesting agents. Additional features include agent-based provision of multi-modal interfaces, including natural language.

98

Please forward to Group Art Unit 2126

Amended Compact Discs

EXAMINER NOTE: THIS PAPER IS AN INTERNAL WORKSHEET ONLY. DO NOT ENCLOSE WITH ANY COMMUNICATION TO THE APPLICANT. ITS PURPOSE IS ONLY THAT OF AN AID IN HIGHLIGHTING A PARTICULAR PROBLEM IN A COMPACT DISC.

THE ATTACHED CD (COPY 1) HAS BEEN REVIEWED BY OIPE FOR COMPLIANCE WITH 37 CFR 1.52(E). ***Please match this CD with the application listed below.***

Date: 6/3/04
Serial No./Control No. 09/225198
Reviewed By: K. SMITH Phone: 308/215

☑ The compact discs are readable and acceptable.

□ Copy 1 and Copy 2 of the compact discs are not the same.

□ The compact discs are unreadable.

□ The files on the compact discs are not in ASCII.

□ The compact discs contain at least one virus.

□ Other

03-30-04

AF/2700

Attorney Docket No. 59501-8016.US01

EXPRESS MAIL LABEL NO. EV 099152888 US

Applicants: CHEYER et al.
Application No.: 09/225,198
Filed: January 5, 1999
Examiner: L. A. Bullock, Jr.
Group Art Unit 2151
For: **SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS**

Mail Stop AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL FOR AMENDMENT AND RESPONSE AND COMPUTER PROGRAM LISTING APPENDIX SUBMITTED ON COMPACT DISC

Sir:

This is in response to the Final Office Action mail by the U.S. Patent and Trademark Office on November 28, 2003. Applicants request a one month extension of time, thus allowing Applicants until March 28, 2004 to respond.

1. Transmitted herewith are the following:

☒ Check No. 2195 in the amount of $55.00
☒ Amendment and Response
☒ Copy 1 and Copy 2 of Compact Disc both containing the identical contents of Appendices A, B, C, D, and E as filed with the patent application on January 5, 1999.

2. Machine format is ISO-9660 file system:

File Name	Size	Creation Date	Last Date
oaa.pl	159,613 bytes	1996/10/08	1998/12/23
fac.pl	52,733 bytes	1997/04/24	1998/05/06
compound.pl	42,937 bytes	1996/12/11	1998/04/10
com_tcp.pl	18,010 bytes	1998/02/10	1998/05/06
translations.pl	19,583 bytes	1998/01/29	1998/12/23

3. Fee Authorization

Check No. 2195 in the amount of $55.00 is enclosed for the required fees for one month extension of time, however, the Commissioner is authorized to charge any underpayment of fees to Deposit Account No. 50-2207. This paper is submitted in duplicate.

Respectfully submitted,
Perkins Coie LLP

Date: March 29, 2004

Carina M. Tan

Carina M. Tan
Registration No. 45,769

Correspondence Address:
Customer No. 22918
Perkins Coie LLP
P. O. Box 2168
Menlo Park, California 94026-2168
(650) 838-4300

EXPRESS MAIL LABEL NO. EV 099152888 US

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Atty Dkt. No. 59501-8016.US01
CHEYER et al.	Group Art Unit No.: 2126
Serial No.: 09/225,198	Examiner: L. A. Bullock, Jr.
Filed on: January 5, 1999	

For: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

Mail Stop AF
Commissioner of Patents
P. O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT AND RESPONSE

Sir:

This is in response to the Final Office Action mailed November 28, 2003, the shortened statutory period for which runs until February 28, 2004.

IN THE CLAIMS

1. (Currently amended) A computer-implemented method for communication and cooperative task completion among a plurality of distributed electronic agents, comprising the acts of:

registering a description of each active client agent's functional capabilities as corresponding registered functional capabilities, using an expandable, platform-independent, inter-agent language, wherein the inter-agent language includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

receiving a request for service as a base goal in the inter-agent language, in the form of an arbitrarily complex goal expression; and

dynamically interpreting the arbitrarily complex goal expression, said act of interpreting further comprising:

generating one or more sub-goals expressed in the inter-agent language;

constructing a goal satisfaction plan wherein the goal satisfaction plan includes:

a suitable delegation of sub-goal requests to best complete the requested service request-by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms; and

dispatching each of the sub-goals to a selected client agent for performance, based on a match between the sub-goal being dispatched and the registered functional capabilities of the selected client agent.

2. (Previously presented) A computer-implemented method as recited in claim 1, further including the following acts of:

receiving a new request for service as a base goal using the inter-agent language, in the form of another arbitrarily complex goal expression, from at least one of the selected client agents in response to the sub-goal dispatched to said agent; and recursively applying the step of dynamically interpreting the arbitrarily complex goal expression in order to perform the new request for service.

3. (Previously presented) A computer-implemented method as recited in claim 2 wherein the act of registering a specific agent further includes:
invoking the specific agent in order to activate the specific agent;
instantiating an instance of the specific agent; and
transmitting the new agent profile from the specific agent to a facilitator agent in response to the instantiation of the specific agent.

4. (original) A computer-implemented method as recited in claim 1 further including the act of deactivating a specific client agent no longer available to provide services by deleting the registration of the specific client agent.

5. original) A computer-implemented method as recited in claim 1 further comprising the act of providing an agent registry data structure.

6. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one symbolic name for each active agent.

7. (original) A computer-implemented method of recited in claim 5 wherein the agent registry data structure includes at least one data declaration for each active agent.

8. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one trigger declaration for one active agent.

9. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one task declaration, and process characteristics for each active agent.

10. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one process characteristic for each active agent.

11. (original) A computer-implemented method as recited in claim 1 further comprising the act of establishing communication between the plurality of distributed agents.

12. (original) A computer-implemented method as recited in claim 1 further comprising the acts of:

receiving a request for service in a second language differing from the inter-agent language;

selecting a registered agent capable of converting the second language into the inter-agent language; and

forwarding the request for service in a second language to the registered agent capable of converting the second language into the inter-agent language, implicitly requesting that such a conversion be performed and the results returned.

13. (original) A computer-implemented method as recited in claim 12 wherein the request includes a natural language query, and the registered agent capable of converting the second language into the inter-agent language service is a natural language agent.

14. (original) A computer-implemented method as recited in claim 13 wherein the natural language query was generated by a user interface agent.

15. (original) A computer-implemented method as recited in claim 1, wherein the base goal requires setting a trigger having conditional functionality and consequential functionality.

16. (original) A computer-implemented method as recited in claim 15 wherein the trigger is an outgoing communications trigger, the computer implemented method further including the acts of:

monitoring all outgoing communication events in order to determine whether a specific outgoing communication event has occurred; and

in response to the occurrence of the specific outgoing communication event, performing the particular action defined by the trigger.

17. (original) A computer-implemented method as recited in claim 15 wherein the trigger is an incoming communications trigger, the computer implemented method further including the acts of:

monitoring all incoming communication events in order to determine whether a specific incoming communication event has occurred; and

in response to the occurrence of a specific incoming communication event satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

18. (original) A computer-implemented method as recited in claim 15 wherein the trigger is a data trigger, the computer implemented method further including the acts of:

monitoring a state of a data repository; and

in response to a particular state event satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

19. (original) A computer-implemented method as recited in claim 15 wherein the trigger is a time trigger, the computer implemented method further including the acts of:

monitoring for the occurrence of a particular time condition; and

in response to the occurrence of a particular time condition satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

20. (original) A computer-implemented method as recited in claim 15 wherein the trigger is installed and executed within the facilitator agent.

21. (original) A computer-implemented method as recited in claim 15 wherein the trigger is installed and executed within a first service-providing agent.

22. (original) A computer-implemented method as recited in claim 15 wherein the conditional functionality of the trigger is installed on a facilitator agent.

23. (original) A computer-implemented method as recited in claim 22 wherein the consequential functionality is installed on a specific service-providing agent other than a facilitator agent.

24. (original) A computer-implemented method as recited in claim 15 wherein the conditional functionality of the trigger is installed on specific service-providing agent other than a facilitator agent.

25. (original) A computer-implemented method as recited in claim 15 wherein the consequential functionality of the trigger is installed on a facilitator agent.

26. (original) A computer-implemented method as recited in claim 1 wherein the base goal is a compound goal having sub-goals separated by operators.

27. (original) A computer-implemented method as recited in claim 26 wherein the type of available operators includes a conjunction operator, a disjunction operator, and a conditional execution operator.

28. (original) A computer-implemented method as recited in claim 27 wherein the type of available operators further includes a parallel disjunction operator that indicates that disjunct goals are to be performed by different agents.

29. (Currently amended) A computer program stored on a computer readable medium, the computer program executable to facilitate cooperative task completion within a distributed computing environment, the distributed computing environment including a plurality of autonomous electronic agents, the distributed computing environment supporting an Interagent Communication Language, the computer program comprising computer executable instructions for:

providing an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment;

interpreting a service request in order to determine a base goal that may be a compound, arbitrarily complex base goal, the service request adhering to an Interagent Communication Language (ICL), wherein the ICL includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

the act of interpreting including the sub-acts of:

determining any task completion advice provided by the base goal, and

determining any task completion constraints provided by the base goal;

constructing a base goal satisfaction plan including the sub-acts of:

determining whether the requested service is available,

determining sub-goals required in completing the base goal by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms,

selecting service-providing electronic agents from the agent registry suitable for performing the determined sub-goals, and

ordering a delegation of sub-goal requests to best complete the requested service; and

implementing the base goal satisfaction plan.

30. (original) A computer program as recited in claim 29 wherein the computer executable instruction for providing an agent registry includes the following computer executable instructions for registering a specific service-providing electronic agent into the agent registry:

establishing a bi-directional communications link between the specific agent and a facilitator agent controlling the agent registry;

providing a new agent profile to the facilitator agent, the new agent profile defining publicly available capabilities of the specific agent; and registering the specific agent together with the new agent profile within the agent registry, thereby making available to the facilitator agent the capabilities of the specific agent.

31. (original) A computer program as recited in claim 30 wherein the computer executable instruction for registering a specific agent further includes:
invoking the specific agent in order to activate the specific agent;
instantiating an instance of the specific agent; and
transmitting the new agent profile from the specific agent to the facilitator agent in response to the instantiation of the specific agent.

32. (original) A computer program as recited in claim 29 wherein the computer executable instruction for providing an agent registry includes a computer executable instruction for removing a specific service-providing electronic agent from the registry upon determining that the specific agent is no longer available to provide services.

33. (original) A computer program as recited in claim 29 wherein the provided agent registry includes a symbolic name, a unique address, data declarations, trigger declarations, task declarations, and process characteristics for each active agent.

34. (original) Computer program as recited in claim 29 further including computer executable instructions for receiving the service request via a communications link established with a client.

35. (original) A computer program as recited in claim 29 wherein the computer executable instruction for providing a service request includes instructions for:
receiving a non-ICL format service request;
selecting an active agent capable of converting the non-ICL formal service request into an ICL format service request;
forwarding the non-ICL format service request to the active agent capable of converting the non-ICL format service request, together with a request that such conversion be performed; and

receiving an ICL format service request corresponding to the non-ICL format service request.

36. (original) A computer program as recited in claim 35 wherein the non-ICL format service request includes a natural language query, and the active agent capable of converting the non-ICL formal service request into an ICL format service request is a natural language agent.

37. (original) A computer program as recited in claim 36 wherein the natural language query is generated by a user interface agent.

38. (original) A computer program as recited in claim 29, the computer program further including computer executable instructions for implementing a base goal that requires setting a trigger having conditional and consequential functionality.

39. (original) A computer program as recited in claim 38 wherein the trigger is an outgoing communications trigger, the computer program further including computer executable instructions for:
monitoring all outgoing communication events in order to determine whether a specific outgoing communication event has occurred; and
in response to the occurrence of the specific outgoing communication event, performing the particular action defined by the trigger.

40. (original) A computer program as recited in claim 38 wherein the trigger is an incoming communications trigger, the computer program further including computer executable instructions for:
monitoring all incoming communication events in order to determine whether a specific incoming communication event has occurred; and
in response to the occurrence of the specific incoming communication event, performing the particular action defined by the trigger.

41. (original) A computer program as recited in claim 38 wherein the trigger is a data trigger, the computer program further including computer executable instructions for:
monitoring a state of a data repository; and
in response to a particular state event, performing the particular action defined by the trigger.

42. (original) A computer program as recited in claim 38 wherein the trigger is a time trigger, the computer program further including computer executable instructions for:
monitoring for the occurrence of a particular time condition; and
in response to the occurrence of the particular time condition, performing the particular action defined by the trigger.

43. (original) A computer program as recited in claim 38 further including computer executable instructions for installing and executing the trigger within the facilitator agent.

44. (original) A computer program as recited in claim 38 further including computer executable instructions for installing and executing the trigger within a first service-providing agent.

45. (original) A computer program as recited in claim 29 further including computer executable instructions for interpreting compound goals having sub-goals separated by operators.

46. (original) A computer program as recited in claim 45 wherein the type of available operators includes a conjunction operator, a disjunction operator, and a conditional execution operator.

47. (original) A computer program as recited in claim 46 wherein the type of available operators further includes parallel disjunction operator that indicates that distinct goals are to be performed by different agents.

48. (Currently amended) An Interagent Communication Language (ICL) providing a basis for facilitated cooperative task completion within a distributed computing environment having a facilitator agent and a plurality of autonomous service-providing electronic agents, wherein:

the ICL having one or more of:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

the ICL having one or more features from a set of features comprising:

enabling agents to perform queries of other agents;

enabling agents to exchange information with other agents; and

enabling agents to set triggers within other agents; and

the ICL having a syntax supporting compound goal expressions wherein said compound goal expressions are such that goals within a single request provided according to the ICL syntax may be coupled by one or more operators from a set of operators comprising:

a conditional execution operator; and

a parallel disjunctive operation that indicates that disjunct goals are to be performed by different agents.

49. (original) An ICL as recited in claim 48, wherein the ICL is computer platform independent.

50. (original) An ICL as recited in claim 48 wherein the ICL is independent of computer programming languages which the plurality of agents are programmed in.

51. (original) An ICL as recited in claim 48 wherein the ICL syntax supports explicit task completion constraints include use of specific agent constraints and response time constraints.

52. (original) An ICL as recited in claim 51, wherein possible types of task completion constraints include use of specific agent constraints and response time constraints.

53. (original) An ICL as recited in claim 51 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

54. (original) An ICL as recited in claim 48 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

55. (original) An ICL as recited in claim 48 wherein each autonomous service-providing electronic agent defines and publishes a set of capability declarations or solvables, expressed in ICL, that describes services provided by such electronic agent.

56. (original) An ICL as recited in claim 55 wherein an electronic agent's solvables define an interface for the electronic agent.

57. (original) An ICL as recited in claim 56 wherein the facilitator agent maintains an agent registry making available a plurality of electronic agent interfaces.

58. (original) An ICL as recited in claim 57 wherein the possible types of solvables includes procedure solvables, a procedure solvable operable to implement a procedure such as a test or an action.

59. (original) An ICL as recited in claim 58 wherein the possible types of solvables further includes data solvables, a data solvable operable to provide access to a collection of data.

60. (original) An ICL as recited in claim 58 wherein the possible types of solvables includes data solvables,.a data solvable operable to provide access to a collection of data.

61. (Currently amended) A facilitator agent arranged to coordinate cooperative task completion within a distributed computing environment having a plurality of autonomous service-providing electronic agents, the facilitator agent comprising:

an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment; and

a facilitating engine operable to parse a service request in order to interpret a compound goal set forth therein, the compound goal including both local and global constraints and control parameters, the service request formed according to an Interagent Communication Language (ICL), wherein the ICL includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

the facilitating engine further operable to construct a goal satisfaction plan by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms.

62. (original) A facilitator agent as recited in claim 61, wherein the facilitating engine is capable of modifying the goal satisfaction plan during execution, the modifying initiated by events such as new agent declarations within the agent registry, decisions made by remote agents, and information provided to the facilitating engine by remote agents.

63. (original) A facilitator agent as recited in claim 61 wherein the agent registry includes a symbolic name, a unique address, data declarations, trigger declarations, task declarations, and process characteristics for each active agent.

64. (original) A facilitator agent as recited in claim 61 wherein the facilitating engine is operable to install a trigger mechanism requesting that a certain action be taken when a certain set of conditions are met.

65. (original) A facilitator agent as recited in claim 64 wherein the trigger mechanism is a communication trigger that monitors communication events and performs the certain action when a certain communication event occurs.

66. (original) A facilitator agent as recited in claim 64 wherein the trigger mechanism is a data trigger that monitors a state of a data repository and performs the certain action when a certain data state is obtained.

67. (original) A facilitator agent as recited in claim 66 wherein the data repository is local to the facilitator agent.

68. (original) A facilitator agent as recited in claim 66 wherein the data repository is remote from the facilitator agent.

69. (original) A facilitator agent as recited in claim 64 wherein the trigger mechanism is a task trigger having a set of conditions.

70. (original) A facilitator agent as recited in claim 61, the facilitator agent further including a global database accessible to at least one of the service-providing electronic agents.

71. (Currently amended) A software-based, flexible computer architecture for communication and cooperation among distributed electronic agents, the architecture contemplating a distributed computing system comprising:

a plurality of service-providing electronic agents; ~~and~~

an Interagent Communication Language (ICL), wherein the inter-agent language includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events; and

a facilitator agent in bi-directional communications with the plurality of service-providing electronic agents, the facilitator agent including:

an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment;

a facilitating engine operable to parse a service request in order to interpret an arbitrarily complex goal set forth therein, the facilitating engine further operable to construct a goal satisfaction plan including the coordination of a suitable delegation of sub-goal requests to best complete the requested service by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms.

72. (Currently amended) A computer architecture as recited in claim 71, wherein the ~~basis for the computer architect is an~~ Interagent Communication Language (ICL) is for enabling agents to perform queries of other agents, exchange information with other agents, and set triggers within other agents, the ICL further defined by an ICL syntax supporting compound goal expressions such that goals within a single request provided according to the ICL syntax may be coupled by a conjunctive operator, a disjunctive operator, a conditional execution operator, and a parallel disjunctive operator parallel disjunctive operator that indicates that disjunct goals are to be performed by different agents.

73. (original) A computer architecture as recited in claim 72, wherein the ICL is computer platform independent.

74. (original) A computer architecture as recited in claim 73 wherein the ICL is independent of computer programming languages in which the plurality of agents are programmed.

75. (original) A computer architecture as recited in claim 73 wherein the ICL syntax supports explicit task completion constraints within goal expressions.

76. (original) A computer architecture as recited in claim 75 wherein possible types of task completion constraints include use of specific agent constraints and response time constraints.

77. (original) A computer architecture as recited in claim 75 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

78. (original) A computer architecture as recited in claim 73 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

79. (original) A computer architecture as recited in claim 73 wherein each autonomous service-providing electronic agent defines and publishes a set of capability declarations or solvables, expressed in ICL, that describes services provided by such electronic agent.

80. (original) A computer architecture as recited in claim 79 wherein an electronic agent's solvables define an interface for the electronic agent.

81. (original) A computer architecture as recited in claim 80 wherein the possible types of solvables includes procedure solvables, a procedure solvable operable to implement a procedure such as a test or an action.

82. (original) A computer architecture as recited in claim 81 wherein the possible types of solvables further includes data solvables, a data solvable operable to provide access to a collection of data.

83. (original) A computer architecture as recited in claim 82 wherein the possible types of solvables includes a data solvable operable to provide access to modify a collection of data.

84. (Previously presented) A computer architecture as recited in claim 71 wherein a planning component of the facilitating engine are distributed across at least two computer processes.

85. (Previously presented) A computer architecture as recited in claim 71 wherein an execution component of the facilitating engine is distributed across at least two computer processes.

86. (Currently amended) A data wave carrier providing a transport mechanism for information communication in a distributed computing environment having at least one facilitator agent and at least one active client agent, and an Interagent Communication Language (ICL), wherein the ICL includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

wherein said at least one facilitator agent is operable to construct a goal satisfaction plan by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms for satisfying one or more requests for service from said at least one active client agent, the data wave carrier comprising a signal representation of an inter-agent language description of an active client agent's functional capabilities.

87. (Previously presented) A data wave carrier as recited in claim 86, the data wave carrier further comprising a corresponding signal representation of said one or more requests for service in the inter-agent language from a first agent to a second agent.

88. (Previously presented) A data wave carrier as recited in claim 86, the data wave carrier further comprising a signal representation of a goal dispatched to an agent for performance from a facilitator agent.

89. (original) A data wave carrier as recited in claim 88 wherein a later state of the data wave carrier comprises a signal representation of a response to the dispatched goal including results and/or a status report from the agent for performance to the facilitator agent.

REMARKS

INTERVIEW:

A telephonic interview was conducted on March 11, 2004. The participants were Examiner Lewis A. Bullock, Jr., David Stringer-Calvert and Carina M. Tan. During the interview, an agreement with respect to all the claims were reached. Applicants argued that the prior art teachings of *KISS* did not disclose any intelligent reasoning when formulating a goal satisfaction plan. Applicants argued that *KISS* merely discloses a method of information retrieval from information repositories such as databases. The examiner disagreed. However, the examiner pointed out that certain features in Applicant's specification regarding ICL are novel. The Examiner indicated that the ICL features: 1) a conversational protocol layer, and 2) a content layer, would distinguish applicants' claims over the prior art. It was agreed that applicants would submit a response amending the claims to include the above novel ICL features.

The Examiner is thanked for the performance of a thorough search. By this response, claims 1, 29, 48, 61, 71, 72 and 86 have been amended. No claims have been cancelled or added. Hence, Claims 1-89 are pending in the Application.

IN THE SPECIFICATION

Compact Disc Containing Appendices

Applicants cancel the computer program listing appearing in the specification in Appendices A, B, C, D, and E. In compliance with 37 CFR 1.96(c), Applicants enclose a CD-ROM labeled as Copy 1 and an identical copy of the CD-ROM labeled as Copy 2 containing the identical contents of Appendices A, B, C, D and E as filed with the patent application on January 5, 1999.

Substitute Pages Of Specification

Enclosed are substitute Pages 1, 8 and 9. Substitute Page 1 of the specification has been amended to identify the compact disc and list the file names, size, and creation date of each file, and substitute Page 8 and Page 9 which have been amended to delete the "Brief Description of the Appendices." Also enclosed is a substitute ABSTRACT containing less than 150 words. The ABSTRACT as originally filed contained more than 150 words.

SUMMARY OF REJECTIONS/OBJECTIONS

In the Office Action, Claims 1-3, 5-11, 15-25, 29-34, 38-44, and 61-71 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Developing Tools for the Open Agent Architecture" by Martin1 in view of U.S. Patent No. 6,484,155 issued to Kiss.

Claims 4, 12-14, 26-28, 35-37, 45-47, and 72-85 are rejected under 35 U.S.C. 103(a) as being unpatentable over Martin1 in view of Kiss, and further in vie of "Information Brokering in an Agent Architecture" by Martin2.

Claims 48-60 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Development Tools for the Open Agent Architecture" by Martin1 in view of "Information Brokering in an Agent Architecture" by Martin2.

REJECTIONS UNDER 35 U.S.C. § 103(a)

CLAIMS 1, 29, 61, 71 and 86

Claim 1, as amended, recites in part, the features:

"registering a description of each active client agent's functional capabilities as corresponding registered functional capabilities, using an expandable, platform-independent, inter-agent language, **wherein the inter-agent language includes:**

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

constructing a goal satisfaction plan, wherein the goal satisfaction plan includes:

a suitable delegation of sub-goal requests to best complete the requested service request by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms;"

Claim 1 includes the limitation of a inter-agent language, wherein the inter-agent language includes 1) a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, and 2) a content layer comprising one or more of goals, triggers and data elements associated with the events. The cited references do not disclose or suggest such a conversational protocol and content layer.

Further, the Office Action states that the "dynamic solution plan" in *KISS* is the equivalent of the "goal satisfaction plan" of applicants' Claim 1 above. The Office Action points to col. 5, lines 14-45; col. 8, line 21 - col. 9, line 26; and col. 10, lines 10-38, and col. 2, lines 50-67 for support.

The method for forming the "dynamic solution plan" in *KISS* is irrelevant to the method of forming the goal satisfaction plan in Applicants' Claim 1. It is respectfully submitted that *KISS* is irrelevant because *KISS* is an invention involving accessing knowledge repositories.' Such knowledge repositories are represented by "knowledge agents." The Abstract of *KISS* states that "the invention solicits accessible knowledge repositories, represented by knowledge agents, for relevant knowledge..."

In other words, *KISS* is merely a method of information retrieval from information repositories or data sources. For example, the meta agent can ask questions involving facts or data and the agents attempt to retrieve the facts or data from the corresponding data repository. In contrast, the goal satisfaction plan of Claim 1 involves asking service providing agents to perform **actions** such as boil water, roast coffee beans, grind the roasted coffee beans as opposed to merely asking the agents to retrieve information from an information repository.

To further explain why *KISS* is irrelevant and completely different from the method of Claim 1, see col. 5 lines 39-43 where "[t]he meta agent 119 is configured to begin executing the solution plan even before the plan is complete." This underscores the fact that the solution plan in *KISS* merely involves information retrieval rather than asking the agent to perform intelligent actions such as roast coffee beans. In *KISS*, it is not fatal to begin executing the solution plan even before the plan is complete because no real harm is done if the meta agent begins by asking the wrong questions. To explain, *KISS* teaches "the meta agent 119 is capable of backtracking or replanning to permit escape from a dead-end." In other words, it is not fatal if the search for data is proceeding down an incorrect search path, as explained in *KISS*. In contrast, the facilitator of Claim 1 cannot begin execution of the goal satisfaction plan before the goal satisfaction plan is complete. For example, it would be fatal for the facilitator to ask a service-providing agent to boil the coffee beans instead of requesting that the coffee beans be first roasted and then ground. Such an action of boiling the coffee beans would be **irreversible** and would produce soggy beans. In other words, the service-providing agents of Claim 1 perform actions and are not merely sources of information.

Further, *KISS* does not use reasoning for "formulating the dynamic solution plan." In other words, *KISS* does not use the inferencing schemes as described in column 7 for generating the solution plan. In fact, *KISS* teaches away from using reasoning or inferencing for generating the solution plan. Column 8, lines 58-61 of *KISS* states that "**[a]fter** the solution plan is formulated, the meta agent 119 implements a distributed inference process to perform the search and execution phases of solving the problem, while maintaining control of the process" (emphasis added). Thus, the inference process is what the solution plan in *KISS* accomplishes and is not what is used to generate the solution plan.

In contrast, Claim 1 shows that the facilitating engine uses sophisticated reasoning when delegating sub-goal requests to best complete the requested service request. The facilitating engine's use of reasoning is supported by the specification on page 13, lines 342-347.

Assume that the facilitator agent of Claim 1 receives a request such as, "Make Coffee". The facilitator agent's facilitating engine uses reasoning to generate the following goal satisfaction plan:

Sub-goal request A: Please perform the act of roasting coffee beans
Sub-goal request B: Please perform the act of grinding coffee beans
Sub-goal request C: Please perform the act of boiling water, etc.

The facilitating engine is able to use reasoning to accomplish the base goal, "Make Coffee" by asking an appropriate agents to first roast the coffee beans before asking the agent to grind the beans, etc.

Neither *Cohen* nor *KISS*, either alone or in combination, disclose, teach, suggest or make obvious the novel features of claim 1. Thus, Claim 1 is allowable.

Claims 29, 61, 71 and 86, each contain similar features regarding "using reasoning to determine sub-goal requests based on non-syntactic decomposition of the base goal and using said reasoning to co-ordinate and schedule efforts by the service-providing electronic agents for fulfilling the sub-goal requests in a cooperative completion of the base goal." Thus, Claims 29, 61, 71 and 86 are allowable for at least the reasons provided herein in respect to Claim 1.

CLAIMS 2-28, 30-47, 62-70, 72-85 and 87-89

Claims 2-28 are either directly or indirectly dependent upon Claim 1 and include all the limitations of Claim 1 and therefore are allowable for at least the reasons provided herein in respect to Claim 1.

Claims 30-47 are either directly or indirectly dependent upon Claim 29 and include all the limitations of Claim 29 and therefore are allowable for at least the reasons provided herein in respect to Claim 29.

Claims 62-70 are either directly or indirectly dependent upon Claim 61 and include all the limitations of Claim 61 and therefore are allowable for at least the reasons provided herein in respect to Claim 61.

Claims 72-85 are either directly or indirectly dependent upon Claim 71 and include all the limitations of Claim 71 and therefore are allowable for at least the reasons provided herein in respect to Claim 71

Claims 87-89 are either directly or indirectly dependent upon Claim 86 and include all the limitations of Claim 86 and therefore are allowable for at least the reasons provided herein in respect to Claim 86.

CLAIM 48

Claim 48 as amended, recites in part:

"the ICL having one or more of:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events; and
a content layer comprising one or more of goals, triggers and data elements associated with the events;
the ICL having a syntax supporting compound goal expressions wherein said compound goal expressions are such that goals within a single request provided according to the ICL syntax may be coupled by one or more operators from a set of operators comprising:
a conditional execution operator; and
a parallel disjunctive operator that indicates that disjunct goals are to be performed by different agents."

The novel method recited in Claim 48 as amended requires that the inter-agent language include 1) a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, and 2) a content layer comprising one or more of goals, triggers and data elements associated with the events. The cited references do not disclose or suggest such a conversational protocol and content layer.

Further, the novel method recited in Claim 48 as amended requires that "goals within a single request" are "coupled by one or more operators from a set of operators". In amended Claim 48, the set of operators comprise, a conditional execution operator, and **a parallel disjunctive operator**.

In the Office Action, the Examiner states that triggers are conditional operators. It is respectfully submitted that triggers are not conditional operators in the sense of an being a syntactical operator in an expression.

Further, the Office Action states that page 10 of *Martin2* discloses **parallel disjunctive operators**. *Martin2* does NOT disclose parallel disjunctive operators. The "disjunction" in *Martin2* is the run-of-the-mill Prolog style disjunction. The expression, "Do task A OR Do Task B," is an example of a *Martin2* type disjunction. In contrast, a "**parallel disjunctive operator** is an operator that indicates that disjunct goals are to be performed by different agents. An example of a **parallel disjunctive operator** expression is "Ask agent Bob to do task A OR Ask agent Fred to do task B concurrently.

None of the cited references disclose, suggest or render obvious the requirement that the "**goals within a single request**" be "coupled by one or more operators from a set of operators", such as **a conditional execution operator** (such as "if" and "when", allowing for particular actions to be predicated on the state, or outcomes of earlier actions), and **a parallel disjunctive operator** (allowing for alternative actions to be performed at the same time, if resources allow, and a first-to-respond strategy may be used in their competition to perform the goal at hand). Claim 48 is allowable over the art of record. Thus, it is respectfully submitted that Claim 48 be held in condition for allowance.

CLAIMS 49-60

Claims 49-60 are either directly or indirectly dependent upon independent Claim 48, and include all the features of Claim 48. Therefore, Claims 49-60 are allowable for at least the reasons provided herein with respect to Claim 48. Furthermore, it is respectfully submitted that Claims 49-60 recite additional features that independently render Claims 49-60 patentable over the art of record. Thus, it is respectfully submitted that Claims 49-60 be held in condition for allowance.

CONCLUSION

For the reasons set forth above, it is respectfully submitted that all of the pending claims are now in condition for allowance. Therefore, the issuance of a formal Notice of Allowance is believed next in order, and that action is most earnestly solicited.

If in the opinion of the Examiner a telephone conference would expedite the prosecution of the subject application, the Examiner is encouraged to call the undersigned at (650) 838-4311.

The Commissioner is authorized to charge any fees due to Applicants' Deposit Account No. 50-2207.

Respectfully submitted,
Perkins Coie LLP

Date: March 29, 2004

Carina M. Tan
Carina M. Tan
Registration No. 45,769

Correspondence Address:

Customer No. 22918
Perkins Coie LLP
P. O. Box 2168
Menlo Park, California 94026
(650) 838-4300

Software-Based Architecture for Communication and Cooperation Among Distributed Electronic Agents

By:

Adam J. Cheyer and David L. Martin

A compact disk containing a computer program listing has been provided in duplicate (copy 1 and copy 2 of the compact disk are identical). The computer program listing in the compact disk is incorporated by reference herein. The compact disk contains files with their names, size and date of creation as follow:

File Name	Size	Creation Date	Last Date
oaa.pl	159,613 bytes	1996/10/08	1998/12/23
fac.pl	52,733 bytes	1997/04/24	1998/05/06
compound.pl	42,937 bytes	1996/12/11	1998/04/10
com_tcp.pl	18,010 bytes	1998/02/10	1998/05/06
translations.pl	19,583 bytes	1998/01/29	1998/12/23

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention is related to distributed computing environments and the completion of tasks within such environments. In particular, the present invention teaches a variety of software-based architectures for communication and cooperation among distributed electronic agents. Certain embodiments teach interagent communication languages enabling client agents to make requests in the form of arbitrarily complex goal expressions that are solved through facilitation by a facilitator agent.

Context and Motivation for Distributed Software Systems

The evolution of models for the design and construction of distributed software systems is being driven forward by several closely interrelated trends: the adoption of a *networked computing model*, rapidly rising expectations for *smarter, longer-lived, more autonomous software applications* and an ever increasing demand for *more accessible and intuitive user interfaces*.

Prior Art Figure 1 illustrates a *networked computing model* 100 having a plurality of client and server computer systems 120 and 122 coupled together over a physical transport mechanism 140. The adoption of the *networked computing model* 100 has lead to a greatly increased reliance on distributed sites for both data and processing resources. Systems such as the networked computing model 100 are based upon at least one physical transport mechanism 140 coupling the multiple computer systems 120 and 122 to support the transfer of information between these computers.

Some of these computers basically support using the network and are known as *client*

FIGURE 9 depicts operations involved in a client agent initiating a service request and receiving the response to that service request in accordance with a certain preferred embodiment of the present invention;

FIGURE 10 depicts operations involved in a client agent responding to a service request in accordance with another preferable embodiment of the present invention;

FIGURE 11 depicts operations involved in a facilitator agent response to a service request in accordance with a preferred embodiment of the present invention;

FIGURE 12 depicts an Open Agent Architecture^{TM} based system of agents implementing a unified messaging application in accordance with a preferred embodiment of the present invention;

FIGURE 13 depicts a map oriented graphical user interface display as might be displayed by a multi-modal map application in accordance with a preferred embodiment of the present invention;

FIGURE 14 depicts a peer to peer multiple facilitator based agent system supporting distributed agents in accordance with a preferred embodiment of the present invention;

FIGURE 15 depicts a multiple facilitator agent system supporting at least a limited form of a hierarchy of facilitators in accordance with a preferred embodiment of the present invention; and

FIGURE 16 depicts a replicated facilitator architecture in accordance with one embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

Figure 3 illustrates a distributed agent system 300 in accordance with one embodiment of the present invention. The agent system 300 includes a facilitator agent 310 and a plurality of agents 320. The illustration of Figure 3 provides a high level view of one simple system structure contemplated by the present invention. The facilitator agent 310 is in essence the "parent" facilitator for its "children" agents 320. The agents 320 forward service requests to the facilitator agent 310. The facilitator agent 310 interprets these requests, organizing a set of goals which are then delegated to appropriate agents for task completion.

The system 300 of Figure 3 can be expanded upon and modified in a variety of ways consistent with the present invention. For example, the agent system 300 can be distributed across a computer network such as that illustrated in Figure 1. The facilitator agent 310 may itself have its functionality distributed across several different computing platforms. The agents 320 may engage in interagent communication (also called peer to peer communications). Several different systems 300 may be coupled together for enhanced performance. These and a variety of other structural configurations are described below in greater detail.

Figure 4 presents the structure typical of a small system 400 in one embodiment of the present invention, showing user interface agents 408, several application agents 404 and meta-agents 406, the system 400 organized as a community of peers by their common relationship to a facilitator agent 402. As will be appreciated, Figure 4 places more structure upon the system 400 than shown in Figure 3, but both are valid representations of structures of the present invention. The facilitator 402 is a specialized server agent that is responsible for coordinating agent communications and cooperative problem-solving. The facilitator 402 may also provide a global data store for its client agents, allowing them to adopt a blackboard style of interaction. Note that certain advantages are found in utilizing two or more facilitator agents within the system 400. For example, larger systems can be assembled from multiple facilitator/client groups, each having the sort of structure

ABSTRACT

A highly flexible, software-based architecture is disclosed for constructing distributed systems. The architecture supports cooperative task completion by flexible and autonomous electronic agents. One or more facilitators are used to broker communication and cooperation among the agents. The architecture provides for the construction of arbitrarily complex goals by users and service-requesting agents. Additional features include agent-based provision of multi-modal interfaces, including natural language.

Perkins Coie

101 Jefferson Drive
Menlo Park, CA 94025-1114
PHONE: 650.838.4300
FAX: 650.838.4350
www.perkinscoie.com

Facsimile Cover Sheet

Confidential and Privileged

If there are any problems with this transmission, please call:

□ *Sender's name and phone number

Date: *June 8, 2004* Cover Sheet & 35 Page(s)

Client Number: *59501-8016.US01*

Return To: (Name) *Sharyl Brown* (Ext.) *4314* (Room No.) *Error! No document variable supplied.*

Original Document(s) will be: □ Sent to you ☒ Held in our files

Sender:	Telephone:	Facsimile:
Sharyl Brown	*(650) 838-4314*	(650) 838-4350

Recipient:	Company:	Telephone:	Facsimile:
Examiner L. A. Bullock, Jr.	*USPTO, Group Art Unit 2126*	*(703) 305-0439*	*(703)*872-9306

Re: *Serial No. 09/225,198*
Atty. Dkt. No. 59501-8016.US01

Dear Examiner Bullock:

Pursuant to your request, attached hereto is a copy of the Amendment and Response which was filed on March 29, 2994, including the return postcard stamped by the USPTO.

We would appreciate receiving status of the Notice of Allowance at your earliest convenience.

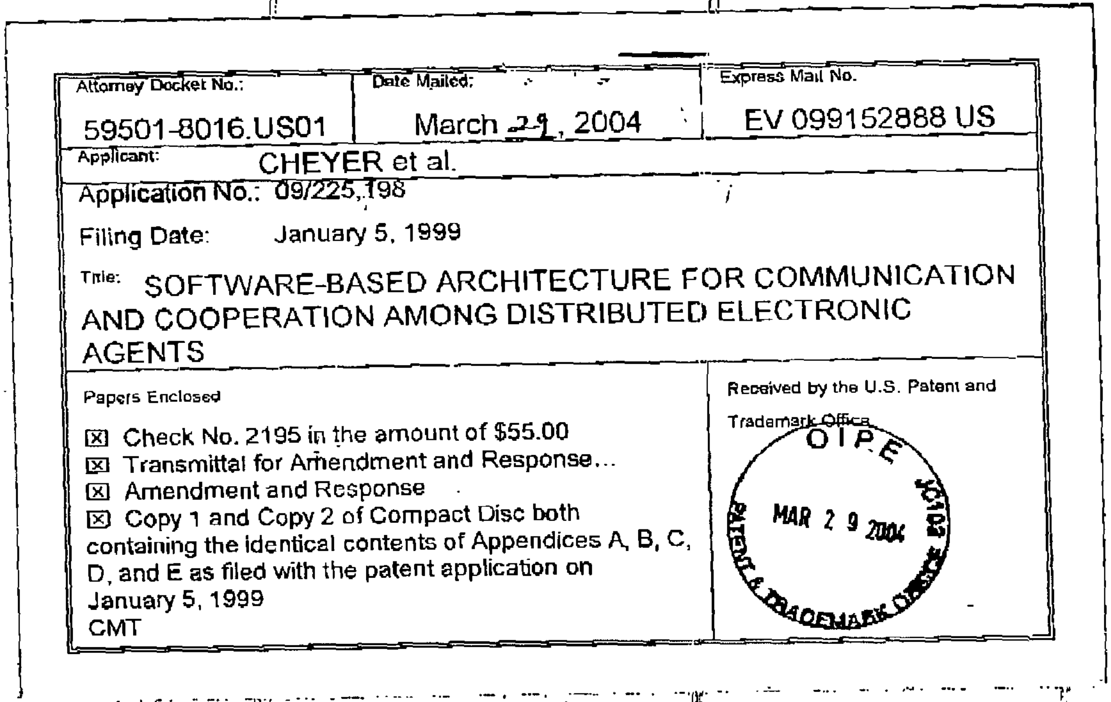
If you have any questions or comments, please contact Carina Tan, Reg. No. 45,769 at (650) 838-4311.

Sincerely,
PERKINS COIE LLP

Sharyl Brown

Sharyl Brown
Secretary to Carina M. Tan

This Fax contains confidential, privileged information intended only for the intended addressee. Do not read, copy or disseminate it unless you are the intended addressee. If you have received this Fax in error, please email it back to the sender at perkinscoie.com and delete it from your system or call us (collect) immediately at 650.838.4300, and mail the original Fax to Perkins Coie LLP, 101 Jefferson Drive, Menlo Park, CA 94025-1114.



Attorney Docket No.:	Date Mailed:	Express Mail No.
59501-8016.US01	March 29, 2004	EV 099152888 US

Applicant: CHEYER et al.

Application No.: 09/225,198

Filing Date: January 5, 1999

Title: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

Papers Enclosed

☒ Check No. 2195 in the amount of $55.00
☒ Transmittal for Amendment and Response...
☒ Amendment and Response
☒ Copy 1 and Copy 2 of Compact Disc both containing the identical contents of Appendices A, B, C, D, and E as filed with the patent application on January 5, 1999
CMT

Received by the U.S. Patent and Trademark Office

PERKINS COIE LLP
BAY AREA PATENT
101 JEFFERSON DRIVE
MENLO PARK, CA 94025-1114

US BANK
1420 5TH
SEATTLE, WA 98101
19-10/1250

2195

3/26/2004

PAY TO THE ORDER OF Commissioner for Patents $ **55.00

Fifty-Five and 00/100*** DOLLARS

UNITED STATES PATENT and TRADEMARK OFFICE

MEMO 59501-8016.US01

Shelina Berger MP

⑈002195⑈ ⑆125000105⑆153592271574⑈

PERKINS COIE LLP 2195

Commissioner for Patents		3/26/2004	
	Carina Tan		55.00
U. S. Bank	59501-8016.US01		55.00

PERKINS COIE LLP 2195

Commissioner for Patents		3/26/2004	
	Carina Tan		55.00
U. S. Bank	59501-8016.US01		55.00

Attorney Docket No. 59501-8016.US01

EXPRESS MAIL LABEL NO. EV 099152888 US

Applicants: CHEYER et al.
Application No.: 09/225,198
Filed: January 5, 1999
Examiner: L. A. Bullock, Jr.
Group Art Unit 2151
For: **SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS**

Mail Stop AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL FOR AMENDMENT AND RESPONSE AND COMPUTER PROGRAM LISTING APPENDIX SUBMITTED ON COMPACT DISC

Sir:

This is in response to the Final Office Action mail by the U.S. Patent and Trademark Office on November 28, 2003. Applicants request a one month extension of time, thus allowing Applicants until March 28, 2004 to respond.

1. Transmitted herewith are the following:

☒ Check No. 2195 in the amount of $55.00
☒ Amendment and Response
☒ Copy 1 and Copy 2 of Compact Disc both containing the identical contents of Appendices A, B, C, D, and E as filed with the patent application on January 5, 1999.

2. Machine format is ISO-9660 file system:

File Name	Size	Creation Date	Last Date
oaa.pl	159,613 bytes	1996/10/08	1998/12/23
fac.pl	52,733 bytes	1997/04/24	1998/05/06
compound.pl	42,937 bytes	1996/12/11	1998/04/10
com_tcp.pl	18,010 bytes	1998/02/10	1998/05/06
translations.pl	19,583 bytes	1998/01/29	1998/12/23

3. Fee Authorization

Check No. 2195 in the amount of $55.00 is enclosed for the required fees for one month extension of time, however, the Commissioner is authorized to charge any underpayment of fees to Deposit Account No. 50-2207. This paper is submitted in duplicate.

Respectfully submitted,
Perkins Coie LLP

Date: March 29, 2004

Carina M. Tan

Carina M. Tan
Registration No. 45,769

Correspondence Address:
Customer No. 22918
Perkins Coie LLP
P. O. Box 2168
Menlo Park, California 94026-2168
(650) 838-4300

RBy

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/225,198	01/05/1999	ADAM J. CHEYER	SRI1P016	2756

22918 7590 07/12/2004

PERKINS COIE LLP
P.O. BOX 2168
MENLO PARK, CA 94026

EXAMINER	
BULLOCK JR; LEWIS ALEXANDER	
ART UNIT	**PAPER NUMBER**
2126	

DATE MAILED: 07/12/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

Advisory Action	**Application No.** *09/225,198*	**Applicant(s)** CHEYER ET AL.
	Examiner Lewis A. Bullock, Jr.	**Art Unit** 2126

--The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

THE REPLY FILED 08 June 2004 FAILS TO PLACE THIS APPLICATION IN CONDITION FOR ALLOWANCE. Therefore, further action by the applicant is required to avoid abandonment of this application. A proper reply to a final rejection under 37 CFR 1.113 may only be either: (1) a timely filed amendment which places the application in condition for allowance; (2) a timely filed Notice of Appeal (with appeal fee); or (3) a timely filed Request for Continued Examination (RCE) in compliance with 37 CFR 1.114.

PERIOD FOR REPLY [check either a) or b)]

a) ☒ The period for reply expires 3 months from the mailing date of the final rejection.

b) □ The period for reply expires on: (1) the mailing date of this Advisory Action, or (2) the date set forth in the final rejection, whichever is later. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of the final rejection. ONLY CHECK THIS BOX WHEN THE FIRST REPLY WAS FILED WITHIN TWO MONTHS OF THE FINAL REJECTION. See MPEP 706.07(f).

Extensions of time may be obtained under 37 CFR 1.136(a). The date on which the petition under 37 CFR 1.136(a) and the appropriate extension fee have been filed is the date for purposes of determining the period of extension and the corresponding amount of the fee. The appropriate extension fee under 37 CFR 1.17(a) is calculated from: (1) the expiration date of the shortened statutory period for reply originally set in the final Office action; or (2) as set forth in (b) above, if checked. Any reply received by the Office later than three months after the mailing date of the final rejection, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

1.□ A Notice of Appeal was filed on _____. Appellant's Brief must be filed within the period set forth in 37 CFR 1.192(a), or any extension thereof (37 CFR 1.191(d)), to avoid dismissal of the appeal.

2.☒ The proposed amendment(s) will not be entered because:

(a) ☒ they raise new issues that would require further consideration and/or search (see NOTE below);

(b) □ they raise the issue of new matter (see Note below);

(c) □ they are not deemed to place the application in better form for appeal by materially reducing or simplifying the issues for appeal; and/or

(d) □ they present additional claims without canceling a corresponding number of finally rejected claims.

NOTE: *See Continuation Sheet*.

3.☒ Applicant's reply has overcome the following rejection(s): CD Requirements and Abstract objections.

4.□ Newly proposed or amended claim(s) _____ would be allowable if submitted in a separate, timely filed amendment canceling the non-allowable claim(s).

5.☒ The a)□ affidavit, b)□ exhibit, or c)☒ request for reconsideration has been considered but does NOT place the application in condition for allowance because: *See Continuation Sheet*.

6.□ The affidavit or exhibit will NOT be considered because it is not directed SOLELY to issues which were newly raised by the Examiner in the final rejection.

7.□ For purposes of Appeal, the proposed amendment(s) a)☒ will not be entered or b)□ will be entered and an explanation of how the new or amended claims would be rejected is provided below or appended.

The status of the claim(s) is (or will be) as follows:

Claim(s) allowed: _____.

Claim(s) objected to: _____.

Claim(s) rejected: *1-89*.

Claim(s) withdrawn from consideration: _____.

8.□ The drawing correction filed on _____ is a)□ approved or b)□ disapproved by the Examiner.

9.□ Note the attached Information Disclosure Statement(s)(PTO-1449) Paper No(s). _____.

10.□ Other: _____

Continuation of 2. NOTE: Applicant amended the claims to language that overcomes the prior art references, however, the examiner has been able to find references that meets the new claim limitations.

Continuation of 5. does NOT place the application in condition for allowance because: Applicant's arguments are unpersuasive. Applicants amendment of the agent language including a conversational protocol layer and a content layer would overcome the applied prior art references, however, the examiner has now found references that teach KQML having a a layer of conversational protocol defined by event types, i.e. a type of ask (ask one or ask_all primitive) along with parameters associated with the event types and a content layer comprising data elements associated with the event as disclosed in all independent claims. Also regarding claim 48, prior art references published by some of the Applicants detailed that ICL has either one of the layers, in particular the content layer, as disclosed in that claim however, the references do not allude to the ICL having both layers. Page 17, lines 12-30 attempts to illustrate that the events are different from the communication acts of KQML, however, the Examiner has not been able to ascertain how they are different from this portion of the specification or any other parts of the specification. It would seem that KQML's ask primitives are events that contain parameter information. Applicant would have to amend the claims or explain how the primitives of KQML would not represent events in order for the Examiner to not equate a layer of KQML primitives having parameter data to Applicant's conversational protocol layer defining events. In regards to claims 1-47 and 61-89, Applicant argues that the applied references, in particular Kiss, teaches the knowledge repository are represented by knowledge agents and merely ask the agents to retrieve information and is irrevelevant to Applicants method of forming the goal satisfaction plan in order to perform actions. The examiner disagrees. The examiner cannot find any language within the claims that details that the service is not a data retrieval service. Therefore, the plan generated to retrieve information is a satisfaction plan to perform actions, i.e. to retrieve the data. In addition, Applicant's example of actions such as boil water, roast coffee beans, and grind the roasted coffee beans are illustrated actions that the invention could perform when solving a goal. It is equally seen from the claim language that the actions can also be the tasks distributed by the meta agent when processing its solution plan to accomplish its overall goal. Applicant argues that the meta agent is capable of backtracking and replanning is another illustrations that Kiss does not teach the invention. In response, the Examiner cannot find any limitations that the plan can not be reevaluated or modified while being implemented. Therefore, the teachings of Kiss just adds another benefit, but still meets the limitations of the claims as disclosed. Applicant then argues that Kiss does not teach using reasoning to formulate the dynamic solution plan. The examiner disagrees. Column 5, lines 25-27 detail that the meta agent contains knowledge of problem solving methodologies and distributed inferencing procedures. Column 5, lines 30-32, detail that the meta agent may maintain the domain-specific knowledge necessary to answer the query itself. Column 5, lines 33-39 detail that meta agent formulates a solution plan and formulates sub-plans in order to perform iterative and recursive procedures. Therefore, the solution plan is generated by the planning component of the meta agent based on domain independent coordination strategies or domain specific reasoning. The cited paragraph Applicant refers to refute the teachings of Kiss refers to how the plan is replanned and backtracked. Applicant then argues that in regards to claim 48, the combination, i.e. Martin1 and Martin2, do not teach a single request are coupled by one or more operators from a set of operators comprising a conditional execution operator or a parallel disjunctive operator. The examiner disagrees. First, it is pointed out that only one operator has to be shown in order for the limitation to be met. Applicant discloses that a conditional execution operator is represented by an arrow (pg. 23, lines 2-5). Page 10, details a mapping rule (request) submitted in ICL format by an information agent which denotes an arrow as well as other control operators that affect the interpretation of a rule. Therefore, the cited reference teaches conditional execution operators and meets the claim language as disclosed.

EXPRESS MAIL LABEL NO. EV 099152888 US

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Atty Dkt. No. 59501-8016.US01
CHEYER et al.	Group Art Unit No.: 2126
Serial No.: 09/225,198	Examiner: L. A. Bullock, Jr.
Filed on: January 5, 1999	

For: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

Mail Stop AF
Commissioner of Patents
P. O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT AND RESPONSE

Sir:

This is in response to the Final Office Action mailed November 28, 2003, the shortened statutory period for which runs until February 28, 2004.

ENTER IN PART

ENTER AMENDMENTS TO SPECIFICATION & ABSTRACT

DO NOT ENTER AMENDMENT TO CLAIMS

7/7/04

CONFIDENTIAL PRIVILEGED

Perkins Coie LLP-Menlo Park

Facsimile Transmittal Sheet

Date: August 25, 2004

Please confirm receipt ☒

Total Number of Pages (including cover sheet): 24

Confirmation by mail □

Attorney Docket No.: 59501-8016.US01

To:	From:
Name: Examiner L. A. Bullock, Jr.	Name: Carina M. Tan
Company: USPTO	Company: Perkins Coie LLP
FAX No.: (703) 872-9306	Phone No.: 650 838-4311
	FAX No.: 650 838-4350

I HEREBY CERTIFY THAT THIS CORRESPONDENCE IS BEING TRANSMITTED VIA FACSIMILE TO (703) 872-9306, THE UNITED STATES PATENT AND TRADEMARK OFFICE, ALEXANDRIA, VA, ON:

Date: August 25, 2004 By: Sharyl Brown
Sharyl Brown

Re: Serial No.: 09/225,198
Filing Date: January 5, 1999

Dear Examiner Bullock:

Attached hereto please find a Transmittal for Supplemental Amendment and Response (in duplicate) and a Supplemental Amendment and Response for the above-identified patent application.

Respectfully submitted,
Perkins Coie LLP

Carina M. Tan

Carina M. Tan
Registration No. 45,769

NOTE: The information contained in this facsimile message may be attorney-client privileged and contains confidential information intended only for the use of the individual named above and others who have been specifically authorized to receive it. If you are not the intended recipient, you are hereby notified that any dissemination, distribution of copy of this communication is strictly prohibited. If you have received this communication in error, please notify us immediately by telephone (650/838-4300) and return the original transmission to us by mail without making a copy.

Perkins Coie LLP • 101 Jefferson Drive • Menlo Park, CA 94025

Attorney Docket No. 59501-8016.US01

CERTIFICATE OF FACSIMILE TRANSMISSION (37 CFR 1.8a)

I hereby certify that this correspondence is being transmitted to the United States Patent & Trademark Office, Central Fax Service Center via facsimile number (703) 872-9306 on August 25, 2004.

Date: August 25, 2004

By: Sharyl Brown
Sharyl Brown

Applicant: *CHEYER et al.*
Application No.: 09/225,198
Examiner L. A. Bullock, Jr.
Art Unit: 2151
Filed: January 5, 1999
For: **SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS**

Mail Stop AF
Commissioner for Patents
P. O. Box 1450
Alexandria, VA 22313-1450

Transmittal For Supplemental Amendment and Response

Sir:

1. Transmitted herewith are the following:
   ☒ Supplemental Amendment and Response
   ☒ Facsimile Cover Sheet

2. Entity Status

   ☒ Small Entity Status (37 CFR 1.9 and 1.27) has been established by a previously submitted Small Entity Statement.

3. Provisional Fee Authorization

   Applicants believe that no fees are due, however, the Commissioner is authorized to charge any underpayment in fees for timely filing to Deposit Account No. 50-2207.

Respectfully submitted,
Perkins Coie LLP

Date: August 25, 2004

Carina M. Tan
Registration No. 45,769

Correspondence Address:
Customer No. 22918
Perkins Coie LLP
P.O. Box 2168
Menlo Park, CA 94
(650) 838-4300

CERTIFICATE OF FACSIMILE TRANSMISSION (37 CFR 1.8a)

I hereby certify that this correspondence is being transmitted to the United States Patent & Trademark Office, Central Fax Service Center via facsimile number (703) 872-9306 on August 25, 2004.

Date: August 25, 2004 By: Sharyl Brown
Sharyl Brown

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	Atty Dkt. No. 59501-8016.US01
CHEYER et al.	Group Art Unit No.: 2126
Serial No.: 09/225,198	Examiner: L. A. Bullock, Jr.
Filed on: January 5, 1999	

For: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

Mail Stop AF
Commissioner of Patents
P. O. Box 1450
Alexandria, VA 22313-1450

SUPPLEMENTAL AMENDMENT AND RESPONSE

Sir:

This is a supplemental amendment to the Final Office Action mailed November 28, 2003, the shortened statutory period for which runs until February 28, 2004. A first amendment and response to Final Office Action mailed November 28, 2003 was filed on March 29, 2004.

IN THE CLAIMS

1. (Currently amended) A computer-implemented method for communication and cooperative task completion among a plurality of distributed electronic agents, comprising the acts of:

registering a description of each active client agent's functional capabilities as corresponding registered functional capabilities, using an expandable, platform-independent, inter-agent language, wherein the inter-agent language includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, wherein the parameter lists further refine the one or more events;

a content layer comprising one or more of goals, triggers and data elements associated with the events;

receiving a request for service as a base goal in the inter-agent language, in the form of an arbitrarily complex goal expression; and

dynamically interpreting the arbitrarily complex goal expression, said act of interpreting further comprising:

generating one or more sub-goals expressed in the inter-agent language;

constructing a goal satisfaction plan wherein the goal satisfaction plan includes:

a suitable delegation of sub-goal requests to best complete the requested service request-by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms; and

dispatching each of the sub-goals to a selected client agent for performance, based on a match between the sub-goal being dispatched and the registered functional capabilities of the selected client agent.

2. (Previously presented) A computer-implemented method as recited in claim 1, further including the following acts of:

receiving a new request for service as a base goal using the inter-agent language, in the form of another arbitrarily complex goal expression, from at least one of the selected client agents in response to the sub-goal dispatched to said agent; and

recursively applying the step of dynamically interpreting the arbitrarily complex goal expression in order to perform the new request for service.

3. (Previously presented) A computer-implemented method as recited in claim 2 wherein the act of registering a specific agent further includes:

invoking the specific agent in order to activate the specific agent;

instantiating an instance of the specific agent; and

transmitting the new agent profile from the specific agent to a facilitator agent in response to the instantiation of the specific agent.

4. (original) A computer-implemented method as recited in claim 1 further including the act of deactivating a specific client agent no longer available to provide services by deleting the registration of the specific client agent.

5. original) A computer-implemented method as recited in claim 1 further comprising the act of providing an agent registry data structure.

6. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one symbolic name for each active agent.

7. (original) A computer-implemented method of recited in claim 5 wherein the agent registry data structure includes at least one data declaration for each active agent.

8. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one trigger declaration for one active agent.

9. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one task declaration, and process characteristics for each active agent.

10. (original) A computer-implemented method as recited in claim 5 wherein the agent registry data structure includes at least one process characteristic for each active agent.

11. (original) A computer-implemented method as recited in claim 1 further comprising the act of establishing communication between the plurality of distributed agents.

12. (original) A computer-implemented method as recited in claim 1 further comprising the acts of:

receiving a request for service in a second language differing from the inter-agent language;

selecting a registered agent capable of converting the second language into the inter-agent language; and

forwarding the request for service in a second language to the registered agent capable of converting the second language into the inter-agent language, implicitly requesting that such a conversion be performed and the results returned.

13. (original) A computer-implemented method as recited in claim 12 wherein the request includes a natural language query, and the registered agent capable of converting the second language into the inter-agent language service is a natural language agent.

14. (original) A computer-implemented method as recited in claim 13 wherein the natural language query was generated by a user interface agent.

15. (original) A computer-implemented method as recited in claim 1, wherein the base goal requires setting a trigger having conditional functionality and consequential functionality.

16. (original) A computer-implemented method as recited in claim 15 wherein the trigger is an outgoing communications trigger, the computer implemented method further including the acts of:

monitoring all outgoing communication events in order to determine whether a specific outgoing communication event has occurred; and

in response to the occurrence of the specific outgoing communication event, performing the particular action defined by the trigger.

17. (original) A computer-implemented method as recited in claim 15 wherein the trigger is an incoming communications trigger, the computer implemented method further including the acts of:

monitoring all incoming communication events in order to determine whether a specific incoming communication event has occurred; and

in response to the occurrence of a specific incoming communication event satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

18. (original) A computer-implemented method as recited in claim 15 wherein the trigger is a data trigger, the computer implemented method further including the acts of:

monitoring a state of a data repository; and

in response to a particular state event satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

19. (original) A computer-implemented method as recited in claim 15 wherein the trigger is a time trigger, the computer implemented method further including the acts of:

monitoring for the occurrence of a particular time condition; and

in response to the occurrence of a particular time condition satisfying the trigger conditional functionality, performing the particular consequential functionality defined by the trigger.

20. (original) A computer-implemented method as recited in claim 15 wherein the trigger is installed and executed within the facilitator agent.

21. (original) A computer-implemented method as recited in claim 15 wherein the trigger is installed and executed within a first service-providing agent.

22. (original) A computer-implemented method as recited in claim 15 wherein the conditional functionality of the trigger is installed on a facilitator agent.

23. (original) A computer-implemented method as recited in claim 22 wherein the consequential functionality is installed on a specific service-providing agent other than a facilitator agent.

24. (original) A computer-implemented method as recited in claim 15 wherein the conditional functionality of the trigger is installed on specific service-providing agent other than a facilitator agent.

25. (original) A computer-implemented method as recited in claim 15 wherein the consequential functionality of the trigger is installed on a facilitator agent.

26. (original) A computer-implemented method as recited in claim 1 wherein the base goal is a compound goal having sub-goals separated by operators.

27. (original) A computer-implemented method as recited in claim 26 wherein the type of available operators includes a conjunction operator, a disjunction operator, and a conditional execution operator.

28. (original) A computer-implemented method as recited in claim 27 wherein the type of available operators further includes a parallel disjunction operator that indicates that disjunct goals are to be performed by different agents.

29. (Currently amended) A computer program stored on a computer readable medium, the computer program executable to facilitate cooperative task completion within a distributed computing environment, the distributed computing environment including a plurality of autonomous electronic agents, the distributed computing environment supporting an Interagent Communication Language, the computer program comprising computer executable instructions for:

providing an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment;

interpreting a service request in order to determine a base goal that may be a compound, arbitrarily complex base goal, the service request adhering to an Interagent Communication Language (ICL), wherein the ICL includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events , wherein the parameter lists further refine the one or more events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

the act of interpreting including the sub-acts of:

determining any task completion advice provided by the base goal, and

determining any task completion constraints provided by the base goal;

constructing a base goal satisfaction plan including the sub-acts of:

determining whether the requested service is available,

determining sub-goals required in completing the base goal by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms, selecting service-providing electronic agents from the agent registry suitable for performing the determined sub-goals, and

ordering a delegation of sub-goal requests to best complete the requested service; and

implementing the base goal satisfaction plan.

30. (original) A computer program as recited in claim 29 wherein the computer executable instruction for providing an agent registry includes the following computer executable instructions for registering a specific service-providing electronic agent into the agent registry:

establishing a bi-directional communications link between the specific agent and a facilitator agent controlling the agent registry;

providing a new agent profile to the facilitator agent, the new agent profile defining publicly available capabilities of the specific agent; and

registering the specific agent together with the new agent profile within the agent registry, thereby making available to the facilitator agent the capabilities of the specific agent.

31. (original) A computer program as recited in claim 30 wherein the computer executable instruction for registering a specific agent further includes:

invoking the specific agent in order to activate the specific agent;

instantiating an instance of the specific agent; and

transmitting the new agent profile from the specific agent to the facilitator agent in response to the instantiation of the specific agent.

32. (original) A computer program as recited in claim 29 wherein the computer executable instruction for providing an agent registry includes a computer executable instruction for removing a specific service-providing electronic agent from the registry upon determining that the specific agent is no longer available to provide services.

33. (original) A computer program as recited in claim 29 wherein the provided agent registry includes a symbolic name, a unique address, data declarations, trigger declarations, task declarations, and process characteristics for each active agent.

34. (original) Computer program as recited in claim 29 further including computer executable instructions for receiving the service request via a communications link established with a client.

35. (original) A computer program as recited in claim 29 wherein the computer executable instruction for providing a service request includes instructions for:

receiving a non-ICL format service request;

selecting an active agent capable of converting the non-ICL formal service request into an ICL format service request;

forwarding the non-ICL format service request to the active agent capable of converting the non-ICL format service request, together with a request that such conversion be performed; and

receiving an ICL format service request corresponding to the non-ICL format service request.

36. (original) A computer program as recited in claim 35 wherein the non-ICL format service request includes a natural language query, and the active agent capable of converting the non-ICL formal service request into an ICL format service request is a natural language agent.

37. (original) A computer program as recited in claim 36 wherein the natural language query is generated by a user interface agent.

38. (original) A computer program as recited in claim 29, the computer program further including computer executable instructions for implementing a base goal that requires setting a trigger having conditional and consequential functionality.

39. (original) A computer program as recited in claim 38 wherein the trigger is an outgoing communications trigger, the computer program further including computer executable instructions for:

monitoring all outgoing communication events in order to determine whether a specific outgoing communication event has occurred; and

in response to the occurrence of the specific outgoing communication event, performing the particular action defined by the trigger.

40. (original) A computer program as recited in claim 38 wherein the trigger is an incoming communications trigger, the computer program further including computer executable instructions for:

monitoring all incoming communication events in order to determine whether a specific incoming communication event has occurred; and

in response to the occurrence of the specific incoming communication event, performing the particular action defined by the trigger.

41. (original) A computer program as recited in claim 38 wherein the trigger is a data trigger, the computer program further including computer executable instructions for:

monitoring a state of a data repository; and

in response to a particular state event, performing the particular action defined by the trigger.

42. (original) A computer program as recited in claim 38 wherein the trigger is a time trigger, the computer program further including computer executable instructions for:

monitoring for the occurrence of a particular time condition; and

in response to the occurrence of the particular time condition, performing the particular action defined by the trigger.

43. (original) A computer program as recited in claim 38 further including computer executable instructions for installing and executing the trigger within the facilitator agent.

44. (original) A computer program as recited in claim 38 further including computer executable instructions for installing and executing the trigger within a first service-providing agent.

45. (original) A computer program as recited in claim 29 further including computer executable instructions for interpreting compound goals having sub-goals separated by operators.

46. (original) A computer program as recited in claim 45 wherein the type of available operators includes a conjunction operator, a disjunction operator, and a conditional execution operator.

47. (original) A computer program as recited in claim 46 wherein the type of available operators further includes parallel disjunction operator that indicates that distinct goals are to be performed by different agents.

48. (Currently amended) An Interagent Communication Language (ICL) providing a basis for facilitated cooperative task completion within a distributed computing environment having a facilitator agent and a plurality of autonomous service-providing electronic agents, wherein:

the ICL having:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, wherein the parameter lists further refine the one or more events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

the ICL having one or more features from a set of features comprising:

enabling agents to perform queries of other agents;

enabling agents to exchange information with other agents; and

enabling agents to set triggers within other agents; and

the ICL having a syntax supporting compound goal expressions wherein said compound goal expressions are such that goals within a single request provided according to the ICL syntax may be coupled by one or more operators from a set of operators comprising:

a conditional execution operator; and

a parallel disjunctive operation that indicates that disjunct goals are to be performed by different agents.

49. (original) An ICL as recited in claim 48, wherein the ICL is computer platform independent.

50. (original) An ICL as recited in claim 48 wherein the ICL is independent of computer programming languages which the plurality of agents are programmed in.

51. (original) An ICL as recited in claim 48 wherein the ICL syntax supports explicit task completion constraints include use of specific agent constraints and response time constraints.

52. (original) An ICL as recited in claim 51, wherein possible types of task completion constraints include use of specific agent constraints and response time constraints.

53. (original) An ICL as recited in claim 51 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

54. (original) An ICL as recited in claim 48 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

55. (original) An ICL as recited in claim 48 wherein each autonomous service-providing electronic agent defines and publishes a set of capability declarations or solvables, expressed in ICL, that describes services provided by such electronic agent.

56. (original) An ICL as recited in claim 55 wherein an electronic agent's solvables define an interface for the electronic agent.

57. (original) An ICL as recited in claim 56 wherein the facilitator agent maintains an agent registry making available a plurality of electronic agent interfaces.

58. (original) An ICL as recited in claim 57 wherein the possible types of solvables includes procedure solvables, a procedure solvable operable to implement a procedure such as a test or an action.

59. (original) An ICL as recited in claim 58 wherein the possible types of solvables further includes data solvables, a data solvable operable to provide access to a collection of data.

60. (original) An ICL as recited in claim 58 wherein the possible types of solvables includes data solvables, a data solvable operable to provide access to a collection of data.

61. (Currently amended) A facilitator agent arranged to coordinate cooperative task completion within a distributed computing environment having a plurality of autonomous service-providing electronic agents, the facilitator agent comprising:

an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment; ~~and~~

a facilitating engine operable to parse a service request in order to interpret a compound goal set forth therein, the compound goal including both local and global constraints and control parameters, the service request formed according to an Interagent Communication Language (ICL), wherein the ICL includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, wherein the parameter lists further refine the one or more events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events; and

the facilitating engine further operable to construct a goal satisfaction plan by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms.

62. (original) A facilitator agent as recited in claim 61, wherein the facilitating engine is capable of modifying the goal satisfaction plan during execution, the modifying initiated by events such as new agent declarations within the agent registry, decisions made by remote agents, and information provided to the facilitating engine by remote agents.

63. (original) A facilitator agent as recited in claim 61 wherein the agent registry includes a symbolic name, a unique address, data declarations, trigger declarations, task declarations, and process characteristics for each active agent.

64. (original) A facilitator agent as recited in claim 61 wherein the facilitating engine is operable to install a trigger mechanism requesting that a certain action be taken when a certain set of conditions are met.

65. (original) A facilitator agent as recited in claim 64 wherein the trigger mechanism is a communication trigger that monitors communication events and performs the certain action when a certain communication event occurs.

66. (original) A facilitator agent as recited in claim 64 wherein the trigger mechanism is a data trigger that monitors a state of a data repository and performs the certain action when a certain data state is obtained.

67. (original) A facilitator agent as recited in claim 66 wherein the data repository is local to the facilitator agent.

68. (original) A facilitator agent as recited in claim 66 wherein the data repository is remote from the facilitator agent.

69. (original) A facilitator agent as recited in claim 64 wherein the trigger mechanism is a task trigger having a set of conditions.

70. (original) A facilitator agent as recited in claim 61, the facilitator agent further including a global database accessible to at least one of the service-providing electronic agents.

71. (Currently amended) A software-based, flexible computer architecture for communication and cooperation among distributed electronic agents, the architecture contemplating a distributed computing system comprising:

a plurality of service-providing electronic agents;

an Interagent Communication Language (ICL), wherein the inter-agent language includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, wherein the parameter lists further refine the one or more events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events; and

a facilitator agent in bi-directional communications with the plurality of service-providing electronic agents, the facilitator agent including:

an agent registry that declares capabilities of service-providing electronic agents currently active within the distributed computing environment;

a facilitating engine operable to parse a service request in order to interpret an arbitrarily complex goal set forth therein, the facilitating engine further operable to construct a goal satisfaction plan including the coordination of a suitable delegation of sub-goal requests to best complete the requested service by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms.

72. (Previously presented) A computer architecture as recited in claim 71, wherein the Interagent Communication Language (ICL) is for enabling agents to perform queries of other agents, exchange information with other agents, and set triggers within other agents, the ICL further defined by an ICL syntax supporting compound goal expressions such that goals within a single request provided according to the ICL syntax may be coupled by a conjunctive operator, a disjunctive operator, a conditional execution operator, and a parallel disjunctive operator parallel disjunctive operator that indicates that disjunct goals are to be performed by different agents.

73. (original) A computer architecture as recited in claim 72, wherein the ICL is computer platform independent.

74. (original) A computer architecture as recited in claim 73 wherein the ICL is independent of computer programming languages in which the plurality of agents are programmed.

75. (original) A computer architecture as recited in claim 73 wherein the ICL syntax supports explicit task completion constraints within goal expressions.

76. (original) A computer architecture as recited in claim 75 wherein possible types of task completion constraints include use of specific agent constraints and response time constraints.

77. (original) A computer architecture as recited in claim 75 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

78. (original) A computer architecture as recited in claim 73 wherein the ICL syntax supports explicit task completion advisory suggestions within goal expressions.

79. (original) A computer architecture as recited in claim 73 wherein each autonomous service-providing electronic agent defines and publishes a set of capability declarations or solvables, expressed in ICL, that describes services provided by such electronic agent.

80. (original) A computer architecture as recited in claim 79 wherein an electronic agent's solvables define an interface for the electronic agent.

81. (original) A computer architecture as recited in claim 80 wherein the possible types of solvables includes procedure solvables, a procedure solvable operable to implement a procedure such as a test or an action.

82. (original) A computer architecture as recited in claim 81 wherein the possible types of solvables further includes data solvables, a data solvable operable to provide access to a collection of data.

83. (original) A computer architecture as recited in claim 82 wherein the possible types of solvables includes a data solvable operable to provide access to modify a collection of data.

84. (Previously presented) A computer architecture as recited in claim 71 wherein a planning component of the facilitating engine are distributed across at least two computer processes.

85. (Previously presented) A computer architecture as recited in claim 71 wherein an execution component of the facilitating engine is distributed across at least two computer processes.

86. (Currently amended) A data wave carrier providing a transport mechanism for information communication in a distributed computing environment having at least one facilitator agent and at least one active client agent, and an Interagent Communication Language (ICL), wherein the ICL includes:

a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, wherein the parameter lists further refine the one or more events; and

a content layer comprising one or more of goals, triggers and data elements associated with the events;

wherein said at least one facilitator agent is operable to construct a goal satisfaction plan by using reasoning that includes one or more of domain-independent coordination strategies, domain-specific reasoning, and application-specific reasoning comprising rules and learning algorithms for satisfying one or more requests for service from said at least one active client agent, the data wave carrier comprising a signal representation of an inter-agent language description of an active client agent's functional capabilities.

87. (Previously presented) A data wave carrier as recited in claim 86, the data wave carrier further comprising a corresponding signal representation of said one or more requests for service in the inter-agent language from a first agent to a second agent.

88. (Previously presented) A data wave carrier as recited in claim 86, the data wave carrier further comprising a signal representation of a goal dispatched to an agent for performance from a facilitator agent.

89. (original) A data wave carrier as recited in claim 88 wherein a later state of the data wave carrier comprises a signal representation of a response to the dispatched goal including results and/or a status report from the agent for performance to the facilitator agent.

REMARKS

INTERVIEW:

A telephonic interview was conducted on August 10, 2004. The participants were Examiner Lewis A. Bullock, Jr., and Carina M. Tan. During the interview, an agreement with respect to all the claims was reached. Applicants distinguished KQML from ICL.

The Examiner is thanked for the performance of a thorough search. By this response, claims 1, 29, 48, 61, 71, and 86 have been amended. No claims have been cancelled or added. Hence, Claims 1-89 are pending in the Application.

CONCLUSION

It is respectfully submitted that all of the pending claims are now in condition for allowance. Therefore, the issuance of a formal Notice of Allowance is believed next in order, and that action is most earnestly solicited.

If in the opinion of the Examiner a telephone conference would expedite the prosecution of the subject application, the Examiner is encouraged to call the undersigned at (650) 838-4311.

The Commissioner is authorized to charge any fees due to Applicants' Deposit Account No. 50-2207.

Respectfully submitted,
Perkins Coie LLP

Carina M. Tan

Date: August 25, 2004

Carina M. Tan
Registration No. 45,769

Correspondence Address:

Customer No. 22918
Perkins Coie LLP
P. O. Box 2168
Menlo Park, California 94026
(650) 838-4300

UNITED STATES PATENT AND TRADEMARK OFFICE

4

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

22918 7590 09/10/2004

PERKINS COIE LLP
P.O. BOX 2168
MENLO PARK, CA 94026

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	
ART UNIT	PAPER NUMBER
2126	

DATE MAILED: 09/10/2004

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/225,198	01/05/1999	ADAM J. CHEYER	SRI1P016	2756

TITLE OF INVENTION: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	NO	$1330	$0	$1330	12/10/2004

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE REFLECTS A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE APPLIED IN THIS APPLICATION. THE PTOL-85B (OR AN EQUIVALENT) MUST BE RETURNED WITHIN THIS PERIOD EVEN IF NO FEE IS DUE OR THE APPLICATION WILL BE REGARDED AS ABANDONED.

HOW TO REPLY TO THIS NOTICE:

I. Review the SMALL ENTITY status shown above.

If the SMALL ENTITY is shown as YES, verify your current SMALL ENTITY status:

A. If the status is the same, pay the TOTAL FEE(S) DUE shown above.

B. If the status above is to be removed, check box 5b on Part B - Fee(s) Transmittal and pay the PUBLICATION FEE (if required) and twice the amount of the ISSUE FEE shown above, or

If the SMALL ENTITY is shown as NO:

A. Pay TOTAL FEE(S) DUE shown above, or

B. If applicant claimed SMALL ENTITY status before, or is now claiming SMALL ENTITY status, check box 5a on Part B - Fee(s) Transmittal and pay the PUBLICATION FEE (if required) and 1/2 the ISSUE FEE shown above.

II. PART B - FEE(S) TRANSMITTAL should be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). Even if the fee(s) have already been paid, Part B - Fee(s) Transmittal should be completed and returned. If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed and an extra copy of the form should be submitted.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Utility patents issuing on applications filed on or after Dec. 12, 1980 may require payment of maintenance fees. It is patentee's responsibility to ensure timely payment of maintenance fees when due.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: Mail

Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

or Fax (703) 746-4000

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

22918 7590 09/10/2004

PERKINS COIE LLP
P.O. BOX 2168
MENLO PARK, CA 94026

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being facsimile transmitted to the USPTO (703) 746-4000, on the date indicated below.

(Depositor's name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/225,198	01/05/1999	ADAM J. CHEYER	SRI1P016	2756

TITLE OF INVENTION: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	NO	$1330	$0	$1330	12/10/2004

EXAMINER	ART UNIT	CLASS-SUBCLASS
BULLOCK JR, LEWIS ALEXANDER	2126	709-310000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

□ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

□ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list

(1) the names of up to 3 registered patent attorneys or agents OR, alternatively,

(2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 ____________________

2 ____________________

3 ____________________

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document has been filed for recordation as set forth in 37 CFR 3.11. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE ____________________ (B) RESIDENCE: (CITY and STATE OR COUNTRY) ____________________

Please check the appropriate assignee category or categories (will not be printed on the patent) : □ Individual □ Corporation or other private group entity □ Government

4a. The following fee(s) are enclosed:

□ Issue Fee

□ Publication Fee (No small entity discount permitted)

□ Advance Order - # of Copies ____________________

4b. Payment of Fee(s):

□ A check in the amount of the fee(s) is enclosed.

□ Payment by credit card. Form PTO-2038 is attached.

□ The Director is hereby authorized by charge the required fee(s), or credit any overpayment, to Deposit Account Number ____________________ (enclose an extra copy of this form).

5. **Change in Entity Status** (from status indicated above)

□ a. Applicant claims SMALL ENTITY status. See 37 CFR 1.27. □ b. Applicant is no longer claiming SMALL ENTITY status. See 37 CFR 1.27(g)(2).

The Director of the USPTO is requested to apply the Issue Fee and Publication Fee (if any) or to re-apply any previously paid issue fee to the application identified above.
NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Authorized Signature ____________________ Date ____________________

Typed or printed name ____________________ Registration No. ____________________

This collection of information is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/225,198	01/05/1999	ADAM J. CHEYER	SRI1P016	2756

22918 7590 09/10/2004

PERKINS COIE LLP
P.O. BOX 2168
MENLO PARK, CA 94026

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	
ART UNIT	PAPER NUMBER
2126	

DATE MAILED: 09/10/2004

Determination of Patent Term Extension under 35 U.S.C. 154 (b)

(application filed after June 7, 1995 but prior to May 29, 2000)

The Patent Term Extension is 0 day(s). Any patent to issue from the above-identified application will include an indication of the 0 day extension on the front page.

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Extension is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (703) 305-1383. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at (703) 305-8283.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/225,198	01/05/1999	ADAM J. CHEYER	SRI1P016	2756

22918 7590 09/10/2004

PERKINS COIE LLP
P.O. BOX 2168
MENLO PARK, CA 94026

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	
ART UNIT	PAPER NUMBER
2126	

DATE MAILED: 09/10/2004

Notice of Fee Increase on October 1, 2004

If a reply to a "Notice of Allowance and Fee(s) Due" is filed in the Office on or after October 1, 2004, then the amount due will be higher than that set forth in the "Notice of Allowance and Fee(s) Due" because some fees will increase effective October 1, 2004. See Revision of Patent Fees for Fiscal Year 2005; Final Rule, 69 Fed. Reg. 52604, 52606 (May 10, 2004).

The current fee schedule is accessible from WEB site (http://www.uspto.gov/main/howtofees.htm).

If the fee paid is the amount shown on the "Notice of Allowance and Fee(s) Due" but not the correct amount in view of the fee increase, a "Notice of Pay Balance of Issue Fee" will be mailed to applicant. In order to avoid processing delays associated with mailing of a "Notice of Pay Balance of Issue Fee," if the response to the Notice of Allowance is to be filed on or after October 1, 2004 (or mailed with a certificate of mailing on or after October 1, 2004), the issue fee paid should be the fee that is required at the time the fee is paid. See Manual of Patent Examining Procedure (MPEP), Section 1306 (Eighth Edition, Rev. 2, May 2004). If the issue fee was previously paid, and the response to the "Notice of Allowance and Fee(s) Due" includes a request to apply a previously-paid issue fee to the issue fee now due, then the difference between the issue fee amount at the time the response is filed and the previously-paid issue fee should be paid. See MPEP Section 1308.01.

Effective October 1, 2004, 37 CFR 1.18 is amended by revising paragraphs (a) through (c) to read as set forth below.

Section 1.18 Patent post allowance (including issue) fees.

(a) Issue fee for issuing each original or reissue patent, except a design or plant patent:	
By a small entity (Sec. 1.27(a))	$685.00
By other than a small entity	$1,370.00
(b) Issue fee for issuing a design patent:	
By a small entity (Sec. 1.27(a))	$245.00
By other than a small entity	$490.00
(c) Issue fee for issuing a plant patent:	
By a small entity (Sec. 1.27(a))	$330.00
By other than a small entity	$660.00

Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at (703) 305-8283.

14

Notice of Allowability	Application No. 09/225,198	Applicant(s) CHEYER ET AL.	
	Examiner Lewis A. Bullock, Jr.	Art Unit 2126	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☒ This communication is responsive to 8/25/04.

2. ☒ The allowed claim(s) is/are 1-89.

3. ☒ The drawings filed on 05 January 1999 are accepted by the Examiner.

4. □ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
   a) □ All b) □ Some* c) □ None of the:
   1. □ Certified copies of the priority documents have been received.
   2. □ Certified copies of the priority documents have been received in Application No. _____ .
   3. □ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____.

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5. □ A SUBSTITUTE OATH OR DECLARATION must be submitted. Note the attached EXAMINER'S AMENDMENT or NOTICE OF INFORMAL PATENT APPLICATION (PTO-152) which gives reason(s) why the oath or declaration is deficient.

6. □ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   (a) □ including changes required by the Notice of Draftsperson's Patent Drawing Review (PTO-948) attached
   1) □ hereto or 2) □ to Paper No./Mail Date _____.
   (b) □ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____.

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

7. □ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☒ Notice of References Cited (PTO-892)
2. □ Notice of Draftperson's Patent Drawing Review (PTO-948)
3. □ Information Disclosure Statements (PTO-1449 or PTO/SB/08), Paper No./Mail Date _____
4. □ Examiner's Comment Regarding Requirement for Deposit of Biological Material
5. □ Notice of Informal Patent Application (PTO-152)
6. ☒ Interview Summary (PTO-413), Paper No./Mail Date _____.
7. ☒ Examiner's Amendment/Comment
8. ☒ Examiner's Statement of Reasons for Allowance
9. □ Other _____.

LEWIS A. BULLOCK, JR.
PRIMARY EXAMINER

EXAMINER'S AMENDMENT

1. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

Authorization for this examiner's amendment was given in a telephone interview with Corina Tan on September 3, 2004.

The application has been amended as follows:

- The claims are amended as listed in the Attachment.

2. The following is an examiner's statement of reasons for allowance: All of the claims are allowable for at least the following reasons: All of the claims detail the inter-agent language including: a layer of conversational protocol defined by event types and parameter lists associated with one or more of the events, wherein the parameters lists further refine the one or more events; and a content layer comprising one or more goals, triggers and data elements associated with the events. The cited prior art of record do not teach the inter-agent language having the cited layers as disclosed. Prior Art article entitled, "Building Distributed Software Systems with the Open Agent Architecture", published by some of the inventors teaches the cited layers however, the reference has been disqualified by the 1.132 Affidavit filed on 11/25/02. In addition, prior art article "Software Agent Technologies" published by Nwana et al. teach an

agent communication language (KQML) that comprises three layers: a content layer, a message layer, and a communication layer. The content layer specifies the actual content of the message for which KQML standard itself has nothing to say about its structure (pg. 4). The message layer provides the performative that specifies the protocol for delivering the message that subsumes the content, i.e. the rules that agents must use when initiating and maintaining an exchange (pg. 5). The communication layer encodes low level communication parameters, such as the identities of the sender and the recipient, and unique identifiers for the particular speech act (pg. 5). The disclosed agent communication language does not read upon the cited agent language because the layer does not define an event type as well as the parameter lists that further refines the event. Nwana's language at best has separate layers for the event and the parameters associated with the event. By Applicant providing these parameters in the same layer as the event such that they further refine the event, a standard set of events are dynamically extensible based upon the parameter list which is not possible with the teachings of Nwana. Therefore, the claims are allowable over the prior art of record.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lewis A. Bullock, Jr. whose telephone number is (703)

305-0439. The examiner can normally be reached on Monday-Friday, 8:30 am - 5:00 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng An can be reached on (703) 305-9678. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

September 3, 2004

LEWIS A. BULLOCK, JR.
PRIMARY EXAMINER

Examiner-Initiated Interview Summary	**Application No.** 09/225,198	**Applicant(s)** CHEYER ET AL.
	Examiner Lewis A. Bullock, Jr.	**Art Unit** 2126

All Participants:

(1) Lewis A. Bullock, Jr..

(2) Corina Tan.

Status of Application: *Allowed*

(3) ____.

(4) ____.

Date of Interview: 2 September 2004

Time: ____

Type of Interview:
☒ Telephonic
☐ Video Conference
☐ Personal (Copy given to: ☐ Applicant ☐ Applicant's representative)

Exhibit Shown or Demonstrated: ☐ Yes ☒ No
If Yes, provide a brief description: .

Part I.

Rejection(s) discussed:
All

Claims discussed:
All

Prior art documents discussed:

Part II.

SUBSTANCE OF INTERVIEW DESCRIBING THE GENERAL NATURE OF WHAT WAS DISCUSSED:
See Continuation Sheet

Part III.

☒ It is not necessary for applicant to provide a separate record of the substance of the interview, since the interview directly resulted in the allowance of the application. The examiner will provide a written summary of the substance of the interview in the Notice of Allowability.

☐ It is not necessary for applicant to provide a separate record of the substance of the interview, since the interview did not result in resolution of all issues. A brief summary by the examiner appears in Part II above.

(Examiner/SPE Signature)

(Applicant/Applicant's Representative Signature – if appropriate)

Continuation Sheet (PTOL-413B) **Application No. 09/225,198**

Continuation of Substance of Interview including description of the general nature of what was discussed: In an informal interview, the examiner explained his position as disclosed in the 1st after final response. Applicant and the examiner agreed upon more language in the claims with the prior language that would place the application in condition for allowance as disclosed in the Reasons for allowance. The examiner also explained to Applicant that the after final response is non-compliant in that it is not readable in later pages, and the all new language is not underlined. The examiner will correct this defect by Examiner's Amendment..

Notice of References Cited	Application/Control No. 09/225,198	Applicant(s)/Patent Under Reexamination CHEYER ET AL.
	Examiner Lewis A. Bullock, Jr.	Art Unit 2126 — Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
	A	US-2003/0167247	09-2003	Masuoka, Ryusuke	706/46
	B	US-2001/0039562	11-2001	SATO, AKIRA	709/202
	C	US-			
	D	US-			
	E	US-			
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	Nwana, Hyacinth et al. "Software Agent Technologies". BT Technology Journal. 1996.
	V	Busetta, Paolo et al. "The BDIM Agent Toolkit Design." 1997.
	W	Mayfield, James et al. "Desiderata for Agent Communication Languages." March 27-29,1995.
	X	Khedro, Taha et al. "Concurrent Endineering through Interoperable Software Agents. August 1994.

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

CERTIFICATE OF FACSIMILE TRANSMISSION (37 CFR 1.8a)
I hereby certify that this correspondence is being transmitted to the United States Patent & Trademark Office, Central Fax Service Center via facsimile number (703) 872-9306 on August 25, 2004.

Date: August 25, 2004 By: Sharyl Brown
Sharyl Brown

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Atty Dkt. No. 59501-8016.US01

CHEYER et al. Group Art Unit No.: 2126

Serial No.: 09/225,198 Examiner: L. A. Bullock, Jr.

Filed on: January 5, 1999

For: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

Mail Stop AF
Commissioner of Patents
P. O. Box 1450
Alexandria, VA 22313-1450

SUPPLEMENTAL AMENDMENT AND RESPONSE

Do Not
ENTER
JLB

Sir:

This is a supplemental amendment to the Final Office Action mailed November 28, 2003, the shortened statutory period for which runs until February 28, 2004. A first amendment and response to Final Office Action mailed November 28, 2003 was filed on March 29, 2004.

SERIAL NUMBER	FILING DATE	CLASS	GROUP ART UNIT	ATTORNEY DOCKET NO.
09/225,198	01/05/99	~~395~~ 719	~~2755~~ 2126	SRI1P016

APPLICANT

ADAM J. CHEYER, PALO ALTO, CA; DAVID L. MARTIN, SANTA CLARA, CA.

CONTINUING DOMESTIC DATA********************
VERIFIED

None

371 (NAT'L STAGE) DATA********************
VERIFIED

None

FOREIGN APPLICATIONS**********
VERIFIED

None

FOREIGN FILING LICENSE GRANTED 01/28/99

	STATE OR COUNTRY	SHEETS DRAWING	TOTAL CLAIMS	INDEPENDENT CLAIMS
Foreign Priority claimed □yes ☒no 35 USC 119 (a-d) conditions met □yes ☒no □Met after Allowance Verified and Acknowledged ________ Examiner's Initials ________ Initials	CA	16	89	6

ADDRESS

BRIAN R COLEMAN
HICKMAN STEPHENS & COLEMAN
P O BOX 52037
PALO ALTO CA 94303-0746

TITLE

SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

FILING FEE RECEIVED		
$2,236	FEES: Authority has been given in Paper No. ______ to charge/credit DEPOSIT ACCOUN NO. ____________ for th following:	□ All Fees □ 1.16 Fees (Filing) .17 Fees (Processing Ext. of time) 8 Fees (Issue) r ____________ edit

Index of Claims

Application No. 09/225,198	**Applicant(s)** CHEYER ET AL.	
Examiner Lewis A. Bullock, Jr.	**Art Unit** 2126	

√	Rejected	—	(Through numeral) Cancelled	N	Non-Elected	A	Appeal
=	Allowed	÷	Restricted	I	Interference	O	Objected

Claim		Date								
Final	Original	7/17/02	3/3/03	11/28/03	9/3/04					
1	1	√	√	√	=					
2	2									
3	3									
4	4									
5	5									
6	6									
7	7									
8	8									
9	9									
10	10									
11	11									
12	12									
13	13									
14	14									
15	15									
16	16									
17	17									
18	18									
19	19									
20	20									
21	21									
22	22									
23	23									
24	24									
25	25									
26	26									
27	27									
28	28									
29	29									
30	30									
31	31									
32	32									
33	33									
34	34									
35	35									
36	36									
37	37									
38	38									
39	39									
40	40									
41	41									
42	42									
43	43									
44	44									
45	45									
46	46									
47	47									
48	48									
49	49									
50	50	√	√	√	=					

Claim		Date								
Final	Original	7/17/02	3/3/03	11/28/03	9/3/04					
51	51	√	√	√	=					
52	52									
53	53									
54	54									
55	55									
56	56									
57	57									
58	58									
59	59									
60	60									
61	61									
62	62									
63	63									
64	64									
65	65									
66	66									
67	67									
68	68									
69	69									
70	70									
71	71									
72	72									
73	73									
74	74									
75	75									
76	76									
77	77									
78	78									
79	79									
80	80									
81	81									
82	82									
83	83									
84	84									
85	85									
86	86									
87	87									
88	88									
89	89	√	√	√	=					
	90									
	91									
	92									
	93									
	94									
	95									
	96									
	97									
	98									
	99									
	100									

Claim		Date								
Final	Original									
	101									
	102									
	103									
	104									
	105									
	106									
	107									
	108									
	109									
	110									
	111									
	112									
	113									
	114									
	115									
	116									
	117									
	118									
	119									
	120									
	121									
	122									
	123									
	124									
	125									
	126									
	127									
	128									
	129									
	130									
	131									
	132									
	133									
	134									
	135									
	136									
	137									
	138									
	139									
	140									
	141									
	142									
	143									
	144									
	145									
	146									
	147									
	148									
	149									
	150									

U.S. Patent and Trademark Office

Part of Paper No. 20040903

Issue Classification	Application No.	Applicant(s)	
	09/225,198	CHEYER ET AL.	
	Examiner	Art Unit	
	Lewis A. Bullock, Jr.	2126	

ISSUE CLASSIFICATION

ORIGINAL							CROSS REFERENCE(S)						
CLASS				SUBCLASS	CLASS	SUBCLASS (ONE SUBCLASS PER BLOCK)							
719				317	709	202							
INTERNATIONAL CLASSIFICATION					717	114							
G	0	6	F	09/54									
				/									
				/									
				/									
				/									

(Assistant Examiner) (Date)	Lewis A. Bullock, Jr. 09/03/04 (Primary Examiner) (Date)	Total Claims Allowed: 89	
(Legal Instruments Examiner) (Date)		O.G. Print Claim(s) 1	O.G. Print Fig. 4

☒ Claims renumbered in the same order as presented by applicant □ CPA □ T.D. □ R.1.47

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original
	1		31		61		91		121		151		181
	2		32		62		92		122		152		182
	3		33		63		93		123		153		183
	4		34		64		94		124		154		184
	5		35		65		95		125		155		185
	6		36		66		96		126		156		186
	7		37		67		97		127		157		187
	8		38		68		98		128		158		188
	9		39		69		99		129		159		189
	10		40		70		100		130		160		190
	11		41		71		101		131		161		191
	12		42		72		102		132		162		192
	13		43		73		103		133		163		193
	14		44		74		104		134		164		194
	15		45		75		105		135		165		195
	16		46		76		106		136		166		196
	17		47		77		107		137		167		197
	18		48		78		108		138		168		198
	19		49		79		109		139		169		199
	20		50		80		110		140		170		200
	21		51		81		111		141		171		201
	22		52		82		112		142		172		202
	23		53		83		113		143		173		203
	24		54		84		114		144		174		204
	25		55		85		115		145		175		205
	26		56		86		116		146		176		206
	27		57		87		117		147		177		207
	28		58		88		118		148		178		208
	29		59		89		119		149		179		209
	30		60		90		120		150		180		210

U.S. Patent and Trademark Office

Part of Paper No. 20040903

Search Notes	Application No. 09/225,198	Applicant(s) CHEYER ET AL.
	Examiner Lewis A. Bullock, Jr.	Art Unit 2126

SEARCHED

Class	Subclass	Date	Examiner
719	317	8/31/2004	LAB
709	202	8/31/2004	LAB
717	114	8/31/2004	LAB

INTERFERENCE SEARCHED

Class	Subclass	Date	Examiner
719	**317**	**9/3/2004**	**LAB**
709	202	9/3/2004	LAB
717	114	9/3/2004	LAB

SEARCH NOTES (INCLUDING SEARCH STRATEGY)

	DATE	EXMR
EAST ACM IEEE CITESEER INTERNET	8/31/2004	LAB

U.S. Patent and Trademark Office

Part of Paper No. 20040903

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: Mail **Mail Stop ISSUE FEE**
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

or Fax **(703) 746-4000**

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

22918 7590 09/10/2004

PERKINS COIE LLP
P.O. BOX 2168
MENLO PARK, CA 94026

10/05/2004 GWORDOF2 00000107 09225198

01 FC:1501	1330.00 OP
02 FC:8001	30.00 OP

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being facsimile transmitted to the USPTO (703) 746-4000, on the date indicated below.

Sharyl Brown	(Depositor's name)
Sharyl Brown	(Signature)
September 29, 2004	(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/225,198	01/05/1999	ADAM J. CHEYER	SRI1P016	2756

TITLE OF INVENTION: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS

APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	NO	$1330	$0	$1330	12/10/2004

EXAMINER	ART UNIT	CLASS-SUBCLASS
BULLOCK JR, LEWIS ALEXANDER	2126	709-310000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

□ Change of correspondence address (or Change of Correspondence Address form PTO/SB/122) attached.

□ "Fee Address" indication (or "Fee Address" Indication form PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list

(1) the names of up to 3 registered patent attorneys or agents OR, alternatively,

(2) the name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 Perkins Coie LLP
2 ____
3 ____

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document has been filed for recordation as set forth in 37 CFR 3.11. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE	(B) RESIDENCE: (CITY and STATE OR COUNTRY)
SRI International	Menlo Park, CA

Please check the appropriate assignee category or categories (will not be printed on the patent) : □ Individual ☒ Corporation or other private group entity □ Government

4a. The following fee(s) are enclosed:

☒ Issue Fee

□ Publication Fee (No small entity discount permitted)

☒ Advance Order - # of Copies 10

4b. Payment of Fee(s):

☒ A check in the amount of the fee(s) is enclosed.

□ Payment by credit card. Form PTO-2038 is attached.

☒ The Director is hereby authorized by charge the required fee(s), or credit any overpayment, to Deposit Account Number 50-2207 (enclose an extra copy of this form).

5. **Change in Entity Status** (from status indicated above)

□ a. Applicant claims SMALL ENTITY status. See 37 CFR 1.27.

☒ b. Applicant is no longer claiming SMALL ENTITY status. See 37 CFR 1.27(g)(2).

The Director of the USPTO is requested to apply the Issue Fee and Publication Fee (if any) or to re-apply any previously paid issue fee to the application identified above.

NOTE: The Issue Fee and Publication Fee (if required) will not be accepted from anyone other than the applicant; a registered attorney or agent; or the assignee or other party in interest as shown by the records of the United States Patent and Trademark Office.

Authorized Signature Carina M. Tan Date September 29, 2004

Typed or printed name Carina M. Tan Registration No. 45,769

This collection of information is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 12 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450.

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PTOL-85 (Rev. 09/04) Approved for use through 04/30/2007. OMB 0651-0033 U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Attorney Docket No. 59501-8016.US01

I hereby certify that this correspondence is being deposited with the U.S. Postal Service with sufficient postage as First Class Mail in an envelope addressed to: Mail Stop Issue Fee, Commissioner for Patents, P. O. Box 1450, Alexandria, VA 22313-1450, on:

Date: September 29, 2004 By: Sharyl Brown
Sharyl Brown

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF:

CHEYER ET AL.

APPLICATION NO.: 09/225,198

FILED: January 5, 1999

FOR: **SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS**

EXAMINER: L. A. BULLOCK, JR.

ART UNIT: 2126

NOTICE OF
ALLOWABILITY: SEPTEMBER 10, 2004

Transmittal of Issue Fee and Advance Order

Mail Stop Issue Fee
Commissioner for Patents
P. O. Box 1450
Alexandria, VA 22313-1450

Sir:

In response to the Notice of Allowance dated September 10, 2004, applicants herewith submit the following:

☒ Form PTOL-85B (in duplicate)

☒ Check in the amount of $1,360.00 for:

1) Issue Fee ($1,330.00) – Large Entity
2) Fee ($30) for 10 advance copies of the printed patent.

☒ Please charge any additional fees necessary for consideration of this paper to Deposit Account No. 50-2207.

Respectfully submitted,
Perkins Coie LLP

Date: September 29, 2004

Carina M. Tan
Carina M. Tan
Registration No. 45,769

Correspondence Address:
Customer No. 22918
Perkins Coie LLP
P. O. Box 2168
Menlo Park, CA 94026-2168
(650) 838-4300

2126

CERTIFICATE OF MAILING (37 CFR 1.8(a))

I hereby certify that this paper (along with any referred to as being attached or enclosed) is being deposited with the U.S. Postal Service as first class mail in an envelope addressed to: Mail Stop Issue Fee, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on:

Date: September 29, 2004

Sharyl Brown
By: Sharyl Brown

UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Cheyer et al.	Docket No.:	59501-8016.US01
Serial No.:	09/225,198	Group Art Unit:	2126
Filing Date:	January 5, 1999	Examiner:	L.A. Bullock, Jr.

For: Software-Based Architecture For Communication And Cooperation Among Distributed Electronic Agents

Mail Stop Issue Fee
Commissioner for Patents
P. O. Box 1450
Alexandria, VA 22313-1450

NOTIFICATION OF ERROR IN PAYMENT OF FEE(S) AS A SMALL ENTITY (37 C.F. § 1.28(c))

1. The present application is no longer entitled to small entity status. On November 18, 2002 and on March 29, 2004, Applicants filed Amendment and Response to Office Actions, each requesting a one month extension of time.

Error

2. The error in the payment of fee(s) as a small entity was as follows:

☒ Applicant believed itself entitled to small entity status, and has discovered that it is no longer be entitled to small entity status.

Fee Payment for Deficiency

3. ☒ Payment is attached for the deficiency between the amount of fees paid and the amount due.

Fee Payment

4. ☒ The attached check in the amount of $110.00 includes fees for the deficiency of the filing of the one month extension of time filed on November 18, 2002 and on March 29, 2004.

10/05/2004 HLE333 00000072 09225198
01 FC:1251 110.00 OP

[/BY042730.087] 9/29/04

☒ In the event that: a) no check to cover the filing fee is enclosed, b) any above-referenced check is inadvertently omitted or lost, or c) any enclosed check is in an amount less than or greater than the required fee, the Commissioner is authorized to charge any required fees, additional fees, or credit any overpayment to Deposit Account 50-2207.

Further Status as a Small Entity

☒ Status as a small entity is hereby withdrawn.

☒ Attached is a postcard for date-stamped return as confirmation of receipt of these materials.

Date: September 29, 2004

Carina M. Tan
Carina M. Tan
Reg. No. 45,769

PERKINS COIE LLP
Customer No.: 22918
P.O. Box 2168
Menlo Park, CA 94026
Tel: (650) 838-4300
Fax: (650) 838-4350

PROVISIONAL APPLICATION COVER SHEET

A|prov

This transmittal and the documents and/or fees itemized hereon and attached hereto have been deposited as "Express Mail Post Office to Addressee" in accordance with 37 C.F.R. §1.10 with Express Mail Mailing Label Number EL285395885US

Attorney Docket No.: SRI1P024+

First Named Inventor: CHEYER, Adam J.

Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

□ Duplicate for fee processing

Sir: This is a request for filing a PROVISIONAL APPLICATION under 37 CFR 1.53(c).

INVENTOR(S)/APPLICANT(S)

LAST NAME	FIRST NAME	MIDDLE INITIAL	RESIDENCE (CITY AND EITHER STATE OR FOREIGN COUNTRY)
CHEYER	Adam	J.	Menlo Park, CA

TITLE OF INVENTION (280 characters max)

AN "INVISIBLE" USER INTERFACE PROVIDING A HIGH DEGREE OF INTEGRATION ACROSSS MULTIPLE APPLICATIONS INA PERSONAL COMPUTER ENVIRONMENT

CORRESPONDENCE ADDRESS

HICKMAN STEPHENS & COLEMAN, LLP
P.O. Box 52037
Palo Alto, CA 94303-0746
(650) 470-7430

ENCLOSED APPLICATION PARTS (check all that apply)

____ Specification Number of Pages ____________ _____ Small Entity Statement

____ Drawing(s) Number of Sheets ____________ X Other: Title, Abstract and Supp. Info. (8 Pages)

X A check or money order is enclosed to cover the Provisional filing fees. Provisional Filing Fee Amount $150

X The commissioner is hereby authorized to charge any additional fees which may be required or credit any overpayment to Deposit Account No. 50-0384 (Order No. SRI1P024+).

The inventions made by an agency of the United States Government or under a contract with an agency of the United States Government.

_________ No _________ Yes, the name of the U.S. Government agency and the contract number are:

Respectfully Submitted,

SIGNATURE ____________________ **DATE** 3/17/99

TYPED NAME Brian R. Coleman **REGISTRATION NO.** 39,145

PROVISIONAL APPLICATION FILING ONLY

TITLE OF THE INVENTION

An "Invisible" User Interface Providing a High Degree of Integration Across Multiple Applications in a Personal Computer Environment

ABSTRACT

Let's say a user is doing some interactive document authoring, and wants to insert some information that exists somewhere in electronic form but is currently external to the document. User wants this to be a highly integrated process, i.e., doesn't want to shift attention away from the working document and temporarily adjust to a different interaction mode, and shouldn't have to waste time on menial, repetitive steps simply to move information around that is already inside the computer.

In accordance with the present invention, an "OAA-style" architecture (facilitated collaboration among distributed agents with declared capabilities in a high-level interagent communication language) can be used to seamlessly ("invisibly") integrate the document authoring application and other auxiliary applications, such as information gathering applications. For example, here is an outline for one possible interactive scenario:

1) User is running an interactive document authoring application

2) User signals for OAA attention (e.g., function key, like invoking a "help" coach)

3) User requests insertion of desired information, e.g. (using natural language) "Insert the directory listings for Adam Cheyer").

4) Request is parsed as needed by NL agents.

5) Appropriate auxiliary agent(s) is(are) selected, and the Request is automatically dispatched in appropriate form to the selected agent(s) (e.g., local address book, Internet search engine, local file manager)

6) Designated agent(s) retrieves/processes the desired data.

7) The retrieved data is presented to the document authoring application and inserted into the working document.

Authoring applications featuring "invisible"-style integration of selected auxiliary applications -- that are pre-selected and "hardwired" for anticipated patterns of likely use -- are becoming quite popular, e.g., spelling and grammar checkers, more specialized citation checkers, and even the merger/integration of Web/desktop file management. But OAA-style architecture provides great potential benefit over such alternatives, in part because it is not a hardwired architecture. With OAA-style architecture, many different possible auxiliary application-agents can each declare their capabilities, and user requests are intelligently processed and delegated dynamically -- so that new agents can be plugged in over time. Such a structure, in accordance with the present invention, can effectively provide the primary UI to an entire personal computing environment rather than just for a single specialized application. In addition, with the likely future growth of speech-driven user interfaces, the value added by a UI-integration technology based on OAA-style architecture in accordance with the present invention is even more pronounced.

SUPPLEMENTAL INFORMATION

Sample source code excerpts for implementing a simple embodiment of the present invention are attached.

Pending patent application serial no. ___________ assigned to SRI (docket no. 3949-2) provides a detailed description of the underlying OAA platform architecture, and also specific descriptions of several applications including "multi-modal maps" which may be helpful in preferred embodiments. The referenced pending patent application is incorporated herein by reference in its entirety.

```
unit main;

(***********************************************************************
 * Unit: Main
 * ---------------------------------------------------------------------
 * Purpose: Program body for  "KeyCap Agent", which provides an "invisible"
 *          or "ubiquitous" user interface to a community of OAA agents.
 *          From any Windows program, a natural language query (e.g. "phone
 *          number of bill smith's manager") can be typed and copied to the
 *          Windows clipboard.  When a key (e.g. F12) is hit, this agent takes
 *          the English request, requests translation and execution from the
 *          OAA community, and then posts the result back on the clipboard with
 *          an audial confirmation ("ding").  If the result couldn't be
executed,
 *          the user hears a failure sound ("bong").  In this way, requests
 *          involving many agents and programs can be accessed transparently
 *          from ANY Windows application at any time, without having to call up
 *          separate user interfaces for each of the involved apps.
 * Authors: Adam Cheyer
 * Version: 1.0
 * Copyright 1998 by SRI International, all rights reserved.
 ***********************************************************************)


(* ===================================================================== *)
(* Interface: exported declarations                                      *)
(* ===================================================================== *)


interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, OleCtrls, isp3, OAAAgent, ComCtrls, FngKbdNtfy, StdCtrls, MPlayer;

type
  TfrmMain = class(TForm)
    MainMenu1: TMainMenu;
    oaa: TAgent;
    tcp: TTCP;
    mnuFile: TMenuItem;
    mnuExit: TMenuItem;
    mnuConnect: TMenuItem;
    Status: TStatusBar;
    keyCap: TFnugryKeyboardNotify;
    txtTest: TEdit;
    MediaPlayer1: TMediaPlayer;
    Label1: TLabel;
    procedure mnuConnectClick(Sender: TObject);
    procedure tcpClose(Sender: TObject);
    procedure tcpDataArrival(Sender: TObject; bytesTotal: Integer);
    procedure tcpConnect(Sender: TObject);
    procedure oaaSendData(Sender: TObject; data: String);
    procedure mnuExitClick(Sender: TObject);
    function oaaOAAEvent(Sender: TObject; ks, func, args: PChar;
      var ans: PChar): Boolean;
    procedure keyCapKeyboardMessage(Sender: TObject; Key, lParam: Integer;
      var fDiscard: Boolean);
    procedure FormCreate(Sender: TObject);
    procedure Play(fname: String);
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
    function ResolveArg(s: PChar) : PChar;
  public
    busy: Boolean;
    { Public declarations }
    procedure ReadCmdLine;
  end;

var
  frmMain: TfrmMain;

implementation

uses connect, clipbrd;

{$R *.DFM}

procedure TfrmMain.ReadCmdLine;
var
  i: Integer;
  StartOAA: Boolean;
  host: String;
  port: integer;
begin
  i := 1;
  StartOAA := False;
  oaa.ReadSetupFile(host, port);
  tcp.RemoteHost := host;
  tcp.RemotePort := port;
  While i <= ParamCount do begin
      if ParamStr(I) = '-oaa_host' then begin
         i := i + 1;
         StartOAA := True;
         tcp.RemoteHost := ParamStr(I)
      end else
      if ParamStr(I) = '-oaa_port' then begin
         I := I + 1;
         StartOAA := True;
         tcp.RemotePort := StrToInt(ParamStr(I))
      end else
      if ParamStr(I) = '-oaa' then
         StartOAA := True
      else
      if ParamStr(I) = '-oaa_name' then begin
         I := I + 1;
         oaa.AgentName := ParamStr(I)
      end;
      I := I + 1
    end;
    If StartOAA then begin
      frmConnect.txtHost.Text := tcp.RemoteHost;
      frmConnect.txtPort.Text := IntToStr(tcp.RemotePort);
      tcp.Connect(frmConnect.txtHost.Text,
         StrToInt(frmConnect.txtPort.Text));
    end;
 end;

 procedure TfrmMain.mnuConnectClick(Sender: TObject);
 begin
```

```
   if not oaa.Connected then begin
      frmConnect.txtHost.Text := tcp.RemoteHost;
      frmConnect.txtPort.Text := IntToStr(tcp.RemotePort);
      if frmConnect.ShowModal = mrOK Then begin
         tcp.RemoteHost := frmConnect.txtHost.Text;
         tcp.RemotePort := StrToInt(frmConnect.txtPort.Text);
         mnuConnect.Caption := 'Disconnect from OAA';
         Status.SimpleText := 'Connecting to ' + tcp.RemoteHost + ', ' +
IntToStr(tcp.RemotePort) + '...';
         tcp.Connect(tcp.RemoteHost, tcp.RemotePort)
      end
   end
   else begin
      mnuConnect.Caption := 'OAA &Connect';
      tcp.Close;
      Status.SimpleText := 'Disconnected'
   end
end;

procedure TfrmMain.oaaSendData(Sender: TObject; data: String);
begin
   tcp.SendData(data)
end;

procedure TfrmMain.tcpConnect(Sender: TObject);
begin
   Status.SimpleText :=  'Connected: ' + tcp.Remotehost + ', ' +
IntToStr(tcp.RemotePort) + '.';
   oaa.Connect;
end;

procedure TfrmMain.tcpDataArrival(Sender: TObject; bytesTotal: Integer);
var
   data: OleVariant;
begin
   tcp.GetData(data, VarOleStr, bytesTotal);
   oaa.OnDataRead(data)
end;

procedure TfrmMain.tcpClose(Sender: TObject);
begin
   Status.SimpleText := 'Disconnected.';
   oaa.Disconnect
end;

procedure TfrmMain.mnuExitClick(Sender: TObject);
begin
   Halt
end;


procedure TfrmMain.Play(fname: String);
begin
   try
      MediaPlayer1.FileName := fname;
      MediaPlayer1.Open;
      MediaPlayer1.Play;
   except
   end
end;
```

```
(* Returns true if this callback handles incoming event, false otherwise *)
(*   ans should be a list of solutions, with the empty list "[]"         *)
(*   representing failure.                                               *)
function TfrmMain.oaaOAAEvent(Sender: TObject; ks, func, args: PChar;
  var ans: PChar): Boolean;
var  goal, a, a2: PChar;
begin
   Result := True;
   a := nil;
   a2 := nil;
   goal := nil;
   (* Default answer: return success *)
   Ans := FStr('[%s(%s)]', [func, args], [func, args]);

   (* Results are returned asynchronously from the Facilitator in
      a "solved" message.  If there are results to a query, paste
      them to the clipboard and succeed, otherwise fail. *)
   if (StrComp(func, 'solved') = 0) and (ListLen(args) = 4) then begin
      NthElt(args, 2, goal);
      NthElt(args, 4, a);
      ListToTerms(a);
      if StrComp(a, '') <> 0 then begin
         Argument(a, 2, a2);
         RemoveQuotes(a2);
         Clipboard.SetTextBuf(UndoubleQuotes(a2));
         Play('good.wav');
      end
      else begin
         Play('bad.wav');
      end;
      StrFree(a);
      StrFree(a2);
      StrFree(goal);
      busy := False;
      keyCap.Enabled := true;
      Result := false
   end
   else Result := false
end;

function addVariable(s: PChar) : PChar;
var f, args, p: PChar;
begin
   Functor(s, f, args);
   if ListLen(args) = 1 then begin
      p := FStr('%s(%s, X)',[f,args],[f,args])
   end
   else p := StrNew(s);
   StrFree(f);
   StrFree(args);
   addVariable := p
end;

function TfrmMain.ResolveArg(s: PChar) : PChar;
var f, args, p, a: PChar;
begin
   s := StrNew(s);
   Functor(s, f, args);
```

```
      if ListLen(args) = 1 then begin
         if (StrPos(args, '(') <> nil) then begin
            p := AddVariable(args);
            oaa.Solve(p, [nil],[nil],'[]', a);
            StrFree(p);
            ListToTerms(a);
            if StrComp(a,'') <> 0 then begin
               StrFree(s);
               Term(a, p, s);
               StrFree(a);
               Argument(p, 2, a);
               s := FStr('%s(%s)',[f,a],[f,a]);
               StrFree(a)
            end
         end
      end;
      StrFree(f);
      StrFree(args);
      ResolveArg := s;
   end;

   (* Procedure executed when the Target key is pressed.
      Finds a query on the clipboard, tries to translate
      the English request to an ICL task, and then send
      the ICL to the Facilitator agent for execution.
      Fail if an ICL translation can't be found. *)
   procedure TfrmMain.keyCapKeyboardMessage(Sender: TObject; Key,
     lParam: Integer; var fDiscard: Boolean);
   var
      s: array[0..300] of char;
      answers, p, p2: PChar;
   begin
      answers := nil;
      p := nil;
      p2 := nil;
      fDiscard := False;
      if (Key = $7B) and  // F5 key
       not busy
       and (Clipboard.GetTextBuf(s, sizeof(s)) > 0)
       then begin
         keyCap.Enabled := False;
         busy := True;
         answers := nil;
         if oaa.Connected then begin
            OAAAgent.DoubleQuotes(s, p2);
            oaa.solve('convert_to_LF(''%s'',LF)',[p2],[p2],'[]', answers);
            StrFree(p2);
            if StrComp(answers,'[]') = 0 then begin
               Play('bad.wav');
               busy := False;
               keyCap.Enabled := true;
            end
            else begin
               Play('step.wav');
               ListToTerms(answers);
               Argument(answers, 2, p);
               StrFree(answers);
               if (StrPos(p, 'send(') = p) then begin
                  Argument(p, 1, answers);
                  txtTest.text := Strpas(answers);
               if (StrPos(answers, 'post_query(show(') = answers) then begin
                  StrFree(p);
                  Argument(answers, 1, p); StrFree(answers); // show(email())
                  Argument(p, 1, answers); StrFree(p);   // email('a');
                  p := ResolveArg(answers); StrFree(answers);
                  p2 := AddVariable(p); StrFree(p);
                  answers := FStr('post_query(%s,[])',[p2],[p2]);
                  StrFree(p2)
               end;
               oaa.PostEvent(answers,[nil],[nil])
            end
            else begin
               Status.Simpletext := 'Strange: LF not wrapped in send()';
               busy := False;
               keyCap.Enabled := true;
            end;
            StrFree(p)
         end;
         StrFree(answers)
      end
   end
end;

procedure TfrmMain.FormCreate(Sender: TObject);
begin
   busy := False;
end;

procedure TfrmMain.Button1Click(Sender: TObject);
var d: Boolean;
begin
  keyCapKeyboardMessage(Sender, $7B, 0, d);
end;

end.
```

PROVISIONAL APPLICATION COVER SHEET

A/prov

This transmittal and the documents and/or fees itemized hereon and attached hereto have been deposited as "Express Mail Post Office to Addressee" in accordance with 37 C.F.R. §1.10 with Express Mail Mailing Label Number EL285395899US

Attorney Docket No.: SRI1P025+

First Named Inventor: CHEYER ET AL.

Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

□ Duplicate for fee processing

Sir: This is a request for filing a PROVISIONAL APPLICATION under 37 CFR 1.53(c).

INVENTOR(S)/APPLICANT(S)

LAST NAME	FIRST NAME	MIDDLE INITIAL	RESIDENCE (CITY AND EITHER STATE OR FOREIGN COUNTRY)
CHEYER	Adam	J.	Menlo Park, California
JULIA	Luc	E.	Menlo Park, California
GUZZONI	Didier	None	Menlo Park, California

TITLE OF INVENTION (280 characters max)

USING A COMMUNITY OF DISTRIBUTED ELECTRONIC AGENTS TO DYNAMICALLY MONITOR AND SUPORT THE NEGOTIATION OF ELECTRONIC TRANSACTIONS

CORRESPONDENCE ADDRESS

HICKMAN STEPHENS & COLEMAN, LLP
P.O. Box 52037
Palo Alto, CA 94303-0746
(650) 470-7430

ENCLOSED APPLICATION PARTS (check all that apply)

____ Specification Number of Pages ____________ ____ Small Entity Statement

____ Drawing(s) Number of Sheets ____________ __X__ Other: White Paper + Cover Sheet (20 Pages)

__X__ A check or money order is enclosed to cover the Provisional filing fees. Provisional Filing Fee Amount $150

__X__ The commissioner is hereby authorized to charge any additional fees which may be required or credit any overpayment to Deposit Account No. 50-0384 (Order No. SRI1P025+).

The inventions made by an agency of the United States Government or under a contract with an agency of the United States Government.

________ No ________ Yes, the name of the U.S. Government agency and the contract number are:

____

Respectfully Submitted,

SIGNATURE ______________________ **DATE** 3/17/99

TYPED NAME Brian R. Coleman **REGISTRATION NO.** 39,145

PROVISIONAL APPLICATION FILING ONLY

TITLE OF THE INVENTION

Using a Community of Distributed Electronic Agents to Dynamically Monitor and Support the Negotiation of Electronic Transactions

ABSTRACT

Complex transactions entail multiple steps over time, possibly non-linear path, with dynamic decisions at each point among multiple alternatives. For example, consider the process of buying a product available at different simultaneous auctions; or obtaining an online mortgage loan; or buying or renting real estate. Currently dominant paradigm of automated support for electronic transactions is typically linear/static: search for prospects, display list, and user is then on his/her own. But numerous alternatives and decisions are possible during the period between the placing of a bid/reservation/application and the closing of the transaction, and there is a real need for on-going monitoring of changing alternatives and for analytical decision support.

The present invention uses "OAA-style" collaborating distributed agents (facilitated collaboration among distributed agents with declared capabilities in a high-level interagent communication language) to monitor and support the negotiation of electronic transactions. One way to outline the work flow in such a system could be:

- define a purchase order profile for a desired product;
- search multiple, heterogeneous database(s) for product availability
- enter one or more "bids" (a formal offer or application to transact business)
- automatically continue to periodically monitor databases for new information about alternatives and status
- analyze the new information with respect to triggers, notify user and present options
- sometimes interactively decide to change/withdraw bid or enter new bid, based on the new informatio/triggers ==> preferably this last step is fully automatic in some cases (bidding agents)

SUPPLEMENTAL INFORMATION

A detailed write-up of a preferred embodiment, in the context of online auctions, is attached.

Pending patent application serial no. ___________ assigned to SRI (docket no. 3949-2) provides a detailed description of the underlying OAA platform architecture, and also specific descrpitions of several applications including "multi-modal maps" which may be helpful in preferred embodiments. The referenced pending patent application is incorporated herein by reference in its entirety.

MetaAuction *Alpha* Preliminary Design

P-3970

SRI International

Overview

Online auctions are increasingly popular on the Internet – more than one hundred web sites exist simply to help buyers and sellers exchange goods through auction-based mechanisms. Although auction meta-sites are beginning to emerge (e.g. www.bidfind.com), these sites provide only search engine capabilities. Simply finding an auction site to participate in, although useful, is not enough: managing the auction process is a time-consuming effort. We believe that a more thorough automation of the auction-buying process is technically feasible, and could be provided as a mass-market service through portal such as the proposed MetaAuction.

SRI carried out an approximately 10 man-week background study and software prototyping effort to guide the MetaAuction design. The resulting system design, based on SRI's Open Agents Architecture™, will provide MetaAuction customers with the following services:

Product finding. Locate candidate products and their auction sites through familiar hierarchical and/or keyword searches. Provide links to manufacturer product descriptions, as well as pricing information, including average cost via auction and prices on fixed price sites for purposes of comparison.

Automated bidding. Execute multi-auction bidding strategy against user-selected products and sites. Automatically registers customers on auction sites and carries out bid placement.

Auction monitoring. Provide real-time reports on bid status and product availability. Monitors both specific items and auctions currently of interest to users, as well as global status of closing prices and other data of interest for all other items within *MetaAuction's* product categories.

Visual monitoring of auction status at the *MetaAuction* website.

- Phone/pager/e-mail notification of major change of status (e.g., when one successfully buys).
- Phone/pager/e-mail notification when items meeting interest criteria come available

P-3970

Functional Requirements

User Interactions

Registration

1. UI for profile entry/review/editing.
    - name/password
    - credit card
    - ship-to, bill-to
    - notification means (e-mail, pager, phone)
2. Secure (SSL) communications to user browser.

Product Finder

1. UI for product location
    - Hierarchical selection of products
    - keyword search (require search engine capability)
2. Rapid lookup & presentation of product information
    - description
    - sku
    - current bid price
    - current best fixed-price
    - auction sites
    - links to manufacturer information sites.
3. Bid control
    - product/auctions to pursue .
    - bidding behavior: start, max bid price; agressiveness

Monitoring/Notification Service

Multi-modal notification: webpage, banner, e-mail, pager, phone (text-to-speech)

1. Change of status of existing auction actions (buy, lose)
2. Periodic status updates (current bid price, auction sites)
3. As new items become available [optional for *Alpha*]

Data Management Tasks

Product Database

1. Information on preselected products and auction sites
2. Product query support
3. Periodic information updates via web crawler

Bid/monitoring management agents

1. Coordinate bids across multiple auction sites
2. Automatically schedule monitoring and bid updates

   Decisions about when a bid should be considered based on user-definable "aggressiveness" parameters, time remaining until the auction closes, importance of a particular auction is based on a prediction of potential success, etc.

User profile database

Maintain secure database of user profiles with links to current auction status.

1. User name/password
2. Credit card
3. Ship-to, Bill-to, account status
4. Historic (audit trail)
5. [user preferences, personalized homepage layout ..]

Information Extractors

1. Site-specific information extraction in support of bidding, monitoring and crawling (product database)
2. Multi-threaded (multiple simultaneous access)
3. Page and information caching

Other Design Issues

1. Browser compatibility (standards: HTML & *httpd* rev, use of Java applets and JavaScript vs. server-side HTML generation, etc.)
2. Modem/network load (bandwidth)
3. MetaAuction server scaling

P-3970

- 500,000-1,000,000 registered users
- 50,000-100,000 simultaneous bidders
- 10,000-100,000 monitored products

4. MetaAuction security
   - Customer (browser-server link)
   - Site security (site firewall, etc)
5. Extensibility for future services
   - New sites, products (mods probably on weekly basis)
   - Adaptations to new auction software (direct auction database access, XML, etc.)
   - Speech interface
     - Phone notification (text to speech)
     - Phone-based monitoring, eventually bidding (speech recognition, NLU)
     - Physical telephony requirements (T1 lines vs. Digital terminals, etc.)
   - Advanced bidding strategies
     - Optimized for single bidder
     - Optimized for blocks of bidders
   - Natural language understanding
     - Product information parsing
     - User description of product
   - Other.
6. Flexibility to support new processing architectures (e.g., highly distributed with significant client-side processing)

Proposed *Alpha* System Design

Goal: Rapid, low-risk / low-cost, rapid development of an operational MetaAuction site. Maximize the code carryover to *beta* version. Provide full suite of features for evaluation and demonstration.

Strategy: Make extensive use of OAA rapid-prototyping capability. Focus on new functionality for MetaAuction application, minimize system integration effort.

Scalability: Implement web page and site information caching to enhance performance for large-scale deployments. Modules compatible with later (*beta*) integration under OAA, CORBA, or blend where appropriate.

Security: *Alpha* architecture firewall-ready, relies on Proxy, SSL communications for sensitive information.

Theory of Operation

The user interactions in the proposed system design parallels the stages of the general buying process, prefaced with user re-entry or registration:

MetaAuction Site Entry. MetaAuction users will be greeted by a main page with MetaAuction branding, advertising, and either a login/password, or registration request for new users (including entry as a limited Guest). The opening page may also provide the product finder interface, major status information for that user (active bids, etc.), and links to help pages. Current users will be able to step into the main bidding channels; new users will be required to input name/password, credit card, ship-to/bill-to, and notification information (e-mail address, pager number, etc.).

Product Search. Locate products based on key-word search, hierarchical product definitions or combination thereof (similar to Junglee or Jango). MetaAuction will carry out a search of its *local* product/auction database and respond with a scrolling list of items meeting user objectives. List will include short product description, auction site, current bid prices, and links to vendor information. For reference, prices on similar items at fixed-cost sites can also be included (in this way, MetaAuction could essentially supersede Jango and Junglee). The system will allow the user to refine their search while viewing the current results in a fashion similar to the commercial search engines.

Bid launch. A user establishes what amounts to a Purchase Order (PO) for a product. The PO specifies the list of acceptable products (that is, what the user considers interchangeable) and auction sites. Parameters for aggressiveness, max bid price, and bidding end date will also be established.

Auction monitoring. The auction monitor provides status information on current active bids for this user, showing the site where recent bids have been placed, the corresponding bid amounts, and times. This view will also allow cancellation of any active PO (within limitations of auction site rules, and the status of that user's outstanding bids).

A more thorough Use Case analysis for these tasks will be carried out during the detailed design phase of the Alpha development program.

System Design Overview

Design Overview

The system design follows the client/server model standard for web applications. The client-side user interfaces provides monitoring and control over a server side that carries out the product/auction finding, extraction, and bidding services of the system.

The Open Agent Architecture^{TM} (OAA^{TM}) was selected as the integration framework for MetaAuction. OAA provides two major features relevant to intelligent, web-based applications. First, its use of independent, cooperating agents and an open interface enables new capabilities to be inserted with minimum effort, and without system reconfiguration or downtime. This feature will allows MetaAúction to add support for new auction sites, new bidding strategies, and new user interfaces as a matter of course.

Second, OAA is based on a collaborative problem-solving scheme, and has great long-term promise as more complex auction negotiations and user interactions are required. Finally, OAA also brings a wealth of existing tools (agents) to provide important user services, including multi-modal input, output (text-to-speech, paging, etc), natural language understanding, along with sophisticated planning/reasoning tools.

In short, OAA provides short-term development cost reductions, and in the longer term, the opportunity for more sophisticated capabilities to be quickly integrated as necessary.

General Architecture

The MetaAuction design, sketched in Figure 1, is based on the OAA™ model for collaborative problem solving. In this context the problem is locating and buying one or more user-selected item from a (potentially vast) list of suppliers. To achieve these ends, the system must elicit information from the user, maintain a database of information on current auction sites and their products, and place and monitor bids according to some buying strategy.

Two basic layouts are possible within the OAA framework. The first, and probably most practical for near-term application, is a coarse-grained architecture. In this design, each agent is a specialist in some aspect of the product monitoring and buying process, and simultaneously manages all purchase order requests. Multi-threaded processing allows each agent to efficiently manage thousands of POs in their various stages. If additional parallelism is required, multiple identical agents can be operated across processors on a network. This parallelism can be achieved at both a single site, or from multiple points of presence on the internet.

An alternative design approach achieves parallelism at the PO level. In this realization an independent family of cooperating agents is instantiated for each bidding request. The functionality of the agents would be identical to that in the coarse design, with the exception that each agent would be somewhat lighter as it need only manage a single buying thread. The primary advantage of this design, beyond general elegance, is its extensibility to highly distributed processing. For example, this would directly support client-side auction processing. This feature may one day be important as consumers are afforded full-time connectivity to the internet.

P-3970

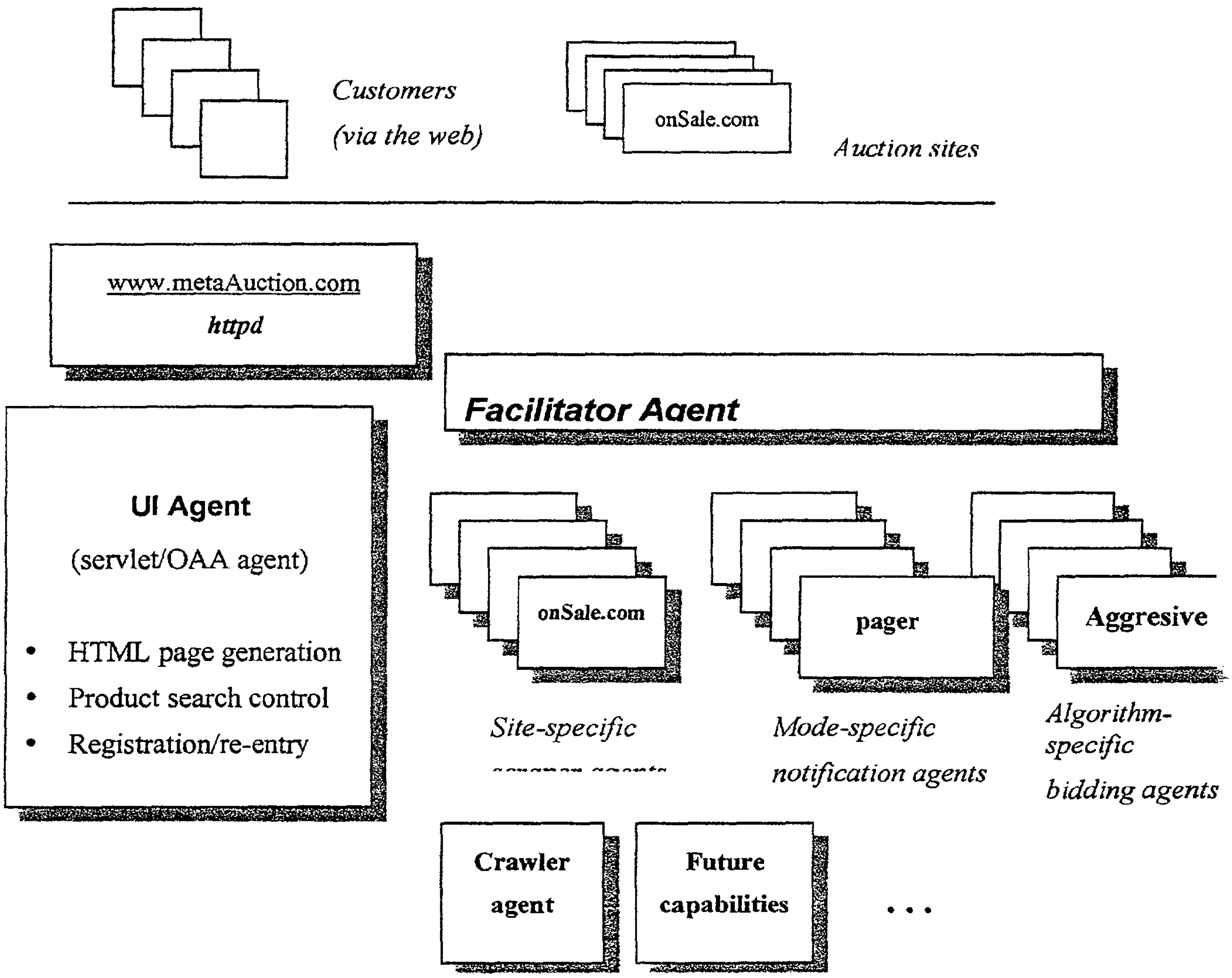


Figure 1: A coarse grained agent-based MetaAuction architecture.

Agents

Page Scrapers. Implements parsing and text submission required for interaction with specific auction sites. (1 per site)

Auction Crawler control. Creates the product information database. Maintains a list of auction sites that are visited periodically and searched for products of interest.

UI. Controls access and interaction with users. Implemented as both an OAA agent and Java Servlet, thus coupling *httpd* facilities directly within OAA framework.

Bid strategy. Implements bidding process. Expected to be a rather fluid module with frequent updates and augmented with adjunct agents as strategies are refined. (1 per product category and/or strategy)

Product/auction search. Controls searches of Web and local database to satisfy user requests for product availability information.

Notification. Automatically notifies metaAuction users of major changes in their bids or product availability, including successful purchases and newly available items. (1 per modality)

Major Data Stores

User database. User ID/passwords, bill-to/ship-to, account status, list of active bids.

Product database. Store of product and corresponding auction site information. Maintained for both the items under bid, as well as speculative requests for products of general interest posted by the crawler agent.

Both databases could be implemented by conventional commercial RDBMS products. Regardless of the parallel-processing architecture, the agents will place heavy demand on a central database of product information, user profiles, and cached web pages. Although we can take advantage of parallelism in data processing, a transaction processing bottleneck will still exist. Careful attention should be given to the choice of database product, the manner in which it is accessed, and the specific data models used. Maintaining speedy database access for common transactions and cached pages at the MetaAuction server site will reduce the apparent latency to the user and compensate for delays in accessing remote auction sites. This problem is not unique to the meta-auction problem, and a variety of commercial products and widely accepted approaches exist for tackling it. However, the success or failure of the final design will hinge on how well it is handled. Anticipating this fundamental problem in the design phase, and applying best practices to its solution, will smooth the way for scalability as the system grows.

Scaling Issues

Communications between extractors and their web sites. To address this problem we propose a two-level caching scheme that minimizes downloads from auction websites, as well as processing for 'scraping' (Figure 2). In this design, information gleaned from pages is time-stamped and stored in an intermediate database. A second level of cache provides rapid access to entire (time-stamped).web pages. While this strategy will provide limited benefit to light user loads, it should recoup significant gain as hundreds or thousands of simultaneous users review or bid on similar products.

OAA scalability. Our initial design will make full use of the OAA fast prototyping capabilities. After the *alpha* design gels, however, we may pursue performance gains by combining smaller agents and other streamlining. OAA will still provide the system framework for adding new functionality and carrying out more advanced problem-solving tasks.

Security Issues

Site Security. We recommend that a firewall stand between the MetaAuction system and the internet. The proxy web-server, used in conjunction with the firewall, will also improve site security.

MetaAuction-auction site and browser-MetaAuction communications. The Alpha system will rely on SSL to secure user information and bid transactions. SSL support is currently required by most auction sites, and is standard on all widely used commercial browsers. MetaAuction would be registered with VeriSign® or other verification/authentication services.

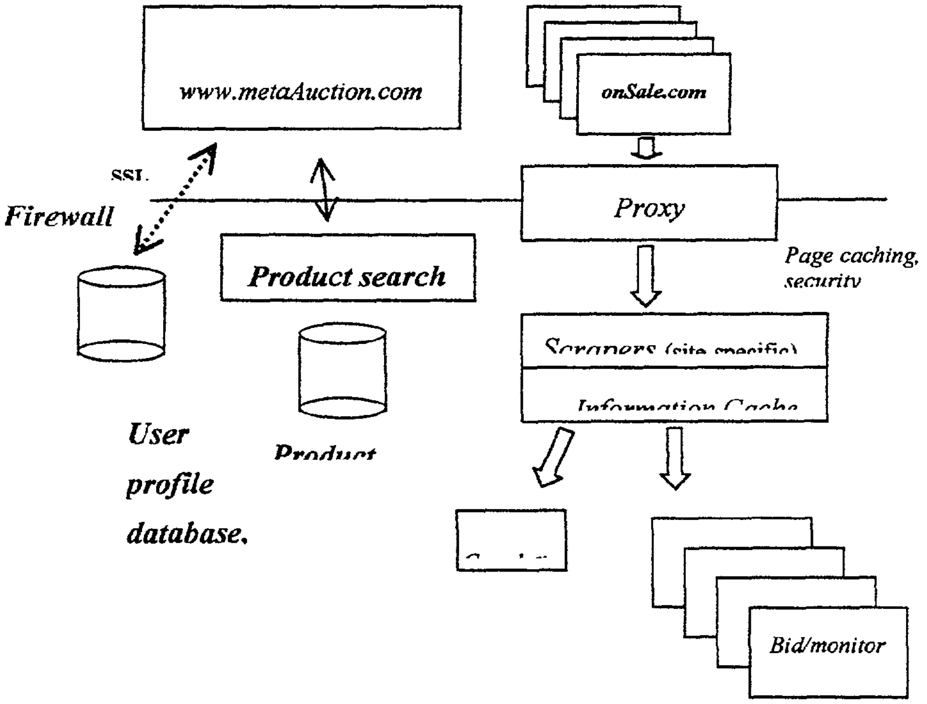


Figure 2: Major data storage components and data flow.

APPENDIX: A MetaAuction PROTOTYPE

Adam Cheyer, Luc Julia, Didier Guzzoni

To better understand the requirements, feasibility and design alternatives for MetaAuction, SRI implemented an experimental prototype known as FAAAB (for "Find All Auctions And Bid"). FAAAB was developed within the Open Agents Architecture™ (OAA™) and implements key elements for Product Finding and Auction Monitoring, and their corresponding user interface. This prototype demonstrated the general feasibility of automatically locating available products, as well as monitoring the bidding process. The FAAAB prototype also provided important experience on user interface issues, and provides the basis for the current UI design.

Problem Statement

The FAAAB prototype effort investigated the feasibility of applying agent technology to an electronic auction domain. Automated software agents would be responsible for providing the following capabilities to users of the service:

1. Find online auctions selling products the user is interested in.
2. Find the best market price available from commercial vendors for these products – this gives the user a baseline value against which to compare the value of an auctioned item.
3. Monitor relevant auction sites, acquiring knowledge about the patterns of buying and selling at that site (e.g., how often does a user truly find a bargain at a particular site).
4. Manage a collection of automated bidding agents who cooperate to achieve the user's objectives in obtaining products at a good price.

The ideas outlined here have been implemented in the FAAAB prototype system, which demonstrates (and allows experimentation with) many of the facets required for accomplishing the vision.

Requirements

To implement auction finding and bidding service, at least the following major components are required:

- web crawlers and information extractors to retrieve information about and from auction sites;
- a user interface to enable users to task and control monitoring and bidding agents;
- and a sophisticated agent scheduler to efficiently coordinate the efforts across auction agents.

P-3970

Crawlers and Extractors

There are at least two types of web data involved in our auction process: data which is highly dynamic in nature (e.g., an ongoing auction changes frequently as new bidders take action), and data which is more stable (e.g., the structure of a given auction site, or the set of auction sites themselves). The more static data can be efficiently cached by web-crawlers which refresh the cache every day or two, while the more dynamic data must be retrieved on demand from the source web page (by extractors). Crawlers generally have an associated database which caches their findings to allow rapid access.

Except for when the information to be extracted is generic in nature, such as a URL or email finder, or a keyword-based search index, knowledge will need to be generated about the format of the information to extract from a particular web site. Since most of the development time associated with this effort is related to encoding this knowledge, having the right tools and languages with which to do so is essential.

Domain-specific knowledge encodes two types of information : how to *navigate* web sites (e.g., go to URL X, find a button labeled Y and click on it, and then fill out a form at the resulting page with the specified information); and how to *extract* information found at the site. Encoding languages should be able to represent both sorts of knowledge in as readable and concise a form as possible.

It is desirable that the tools that interpret or execute the domain-specific knowledge have the following properties:

- Multi-threaded (or multi-tasking) to be able to manage many knowledge-extraction requests simultaneously
- Replicate-able and/or mobile, so that new instances can be created and distributed according to load requirements
- Able to communicate with other components of the system, such as databases for caching information, user interfaces, etc.

User Interface

The User Interface (UI) to the target system should have the following properties:

- Be portable and accessible from any modern web browser.
- Be rich enough to visually express a complex space of information: many agents will bid and monitor at auctions with changing prices, varied closing dates, etc. A user should have a global understanding of the current status of an entire multi-agent auction portfolio, and the ability to modify or control any aspect of the process.
- Intermittent operation: a user should be able to disconnect and reconnect at will.
- Lightweight: Additional low-profile UIs (e.g., banners) can update the user of portfolio events without requiring connection to the full user interface or focused attention by the user.

P-3970

Agent Scheduler

An agent-scheduler must be imbued with the ability to efficiently manage and schedule information retrieval tasks for the auction monitoring and bidding agents. The scheduler will make decisions about when a bid should be considered based on user-definable "aggressiveness" parameters, the amount of time remaining until the auction closes, how important a particular auction is based on a prediction of potential success, etc.

Implementation

The ideas and requirements have, for the most part, been implemented in a prototype system called FAAAB (for "Find All Auctions And Bid"). Features and issues not yet accomplished by this prototype are discussed in the next section.

Integration Framework

From our requirements section, it is clear that the implemented system must use a client/server model, with client user interfaces providing monitoring and control of a server side that provides the finding, extraction, and bidding services of the system.

The Open Agent Architecture™ (OAA™) was selected as the integration framework for FAAAB, as the OAA enables rapid development of both Java-based client user interfaces and complex server applications made out of distributed components.

Crawlers and Extractors

In the requirements section, we spoke of the need for both tools and languages for expressing domain-specific extraction and navigational logic. After evaluating several in-house (DIFF-parse, DCG-parse, plus web agents) and commercial (AgentSoft's LiveAgent Pro [2]), we chose Digital's WebL product [3] as the best tool and language for our needs. Implemented in Java, WebL provides powerful features (parallelism concepts, markup algrebra combining query sets over regular expressions and structured HTML and XML representations, specialized web-related exception handling, and so forth). In addition, source code is provided for free, allowing us to easily incorporate the technology as an OAA agent, and to make extensions to it as necessary (e.g., add mobility).

The WebLOAA agent provides a generic OAA-enabled tool which can dynamically load knowledge scripts encapsulating a particular web site or service; each script becomes an agent in the OAA sense. Scripts can serve both as extractors, and when used in conjunction with an OAA database agent, crawlers, which cache their results for fast retrieval.

Auction Finders

The first task of the FAAAB prototype requests a user to input a description of a product that they are interested in, and then attempts to find auctions which are selling comparable products. This task was accomplished in two ways:

1. an extractor agent for an existing auction search site, BidFind.com [4];
2. a web crawler for a site not currently indexed by BidFind (www.webauction.com [5]) to demonstrate that we need not be reliant on the BidFind service.

Both the WebAuction agent and the BidFind agent were rapidly implemented as WebL scripts managed by the WebLOAA agent. See Appendix A for source code of the BidFind extractor agent, to get a sense of the power and elegance of the WebL language for web wrapping and extraction tasks.

Market Price Finder

Once a list of interesting auctions have been returned and displayed to the user, he or she should choose which sites are to be managed by FAAAB auction agents. For each auction returned, the user may visit the website or may request a search for the real market price of the auctioned object. Note that even though most auctions returned for a given search will offer relatively similar products, the products may have varying brands, optional features, and so forth, so it might be desirable to find the market price for each individual auction and not just for the group.

Even though product search engines are starting to appear (e.g., Jango [6], Junglee [7]), finding a good guess for the real market price of an object given only its description is not an easy task. Here is the approach that we are using for the moment:

Given a description of an object for sale at an auction, we first try to guess the major category (e.g., desktop computer, camera, flowers, etc.) for the product. Jango, the best product finder currently available, uses a yahoo-like hierarchical category scheme, with pulldown menus for different choices. For instance, if looking for a laptop computer, you choose this category and then select criteria such as brand, model and processor speed from a preselected list. These criteria, both headings and values, are different for each category.

To guess the product category, we wrote a WebLOAA crawler that traverses all of the categories from Jango and pulls out the criteria and values for each category. Then for the given object description, we choose its category by taking the one which has the highest number of values present in the object description. The values are augmented by a hand-coded synonym list to increase the likelihood of positive matches. Note: a future enhancement would be to automatically generate pertinent keywords from the corpus of category items using statistical methods.

Once the major category has been determined, the findMarketPrice agent tries to fill out the search form for that category with relevant criteria taken from the target description. Resulting descriptions are then compared against the target description for similarity, and the price, description and URL of the best guess are returned to the user interface for display to the user. Note: this step is still under development, and in the meantime, a simple price-by-category result is returned as the answer.

Agent Scheduler

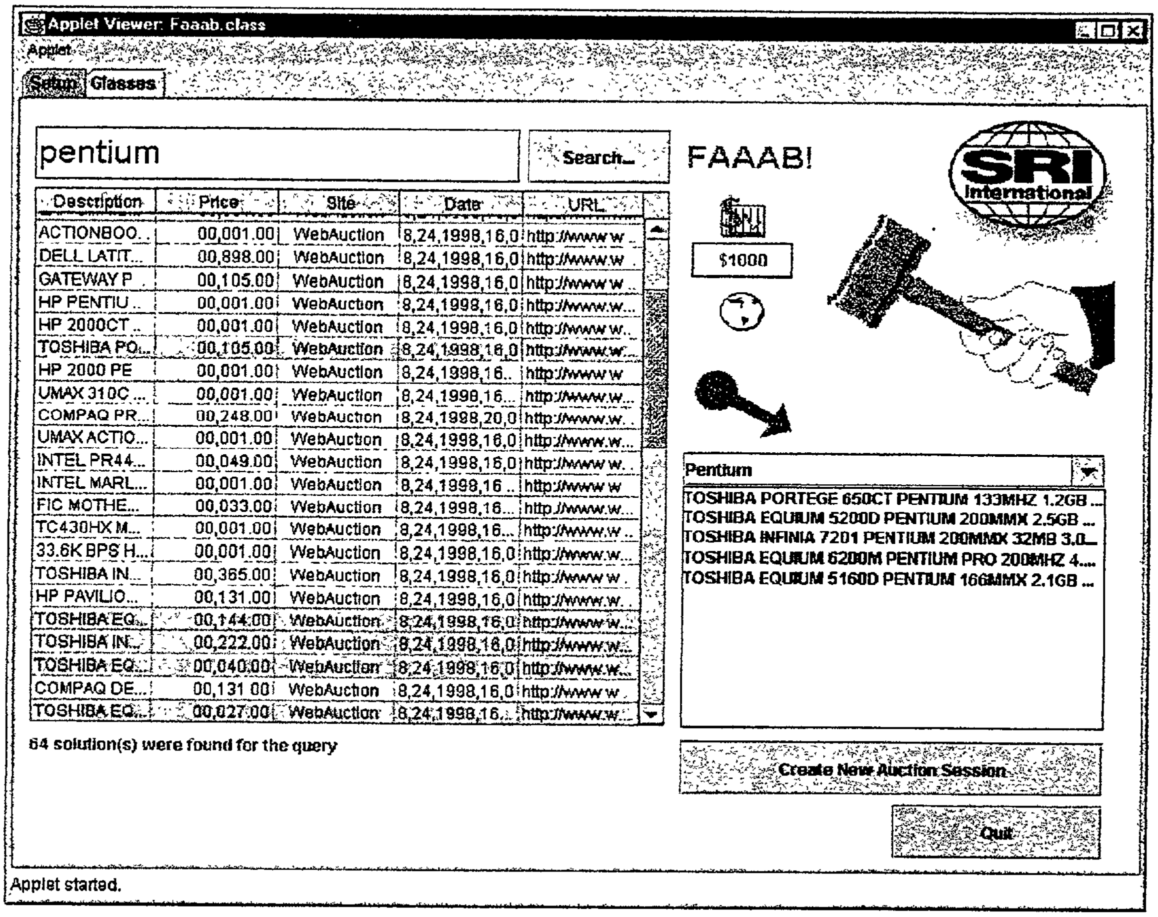
The agent scheduler, implemented in Prolog, is responsible for efficiently managing update requests for an entire community of auction bidding and monitoring agents and for webcrawler agents. As auction agents are created or modified, the agent scheduler plans future checkup times for the site based on:

- Closing date: scheduled checkup times are proportional to the amount of time remaining until the auction closes. If the auction closes in a week from now, it doesn't make sense to check the auction page every minute. However as the deadline approaches, more frequent checks are necessary.
- Auction importance: some auctions are more desirable than others for a variety of reasons. For instance, if one site has 500 copies of a product to sell and another site only has one, placing a winning bid at the first site is much more interesting because 500 other users will need to bid higher before your bid is surpassed.
- Users might also indicate a preference for a particular auction object over another.
- Agressiveness parameter: a user can tailor an aggresiveness parameter which influences how often an auction agent bids.
- Real-world notifications: some auctions send an email when someone has outbid you, and an email agent could reschedule an immediate counter-bid (not yet implemented).

User Interface

The user interface design reflects three key phases in the auction buying process: product location, bid selection, and monitoring. In the setup phase of the FAAAB prototype (Figure 1), users find and evaluate potential auctions of interest, and then create auction agents to monitor and bid on these auctions. The Setup tab of the user interface retrieves auction information from the auction site crawlers. Users may then choose to view the original web page featuring the auction or to search for the best online market price for the product offered by that specific auction. Note that multiple "tabs" can be created, each representing a group of agents (currently limited to 10 per group) acting upon auctions in a given "domain" (e.g., Pentium computers, cameras, sunglasses, etc.)

P-3970



pentium

Description	Price	Site	Date	URL
ACTIONBOO...	00,001.00	WebAuction	8,24,1998,16,0	http://www.w...
DELL LATIT...	00,898.00	WebAuction	8,24,1998,16,0	http://www.w...
GATEWAY P...	00,105.00	WebAuction	8,24,1998,16,0	http://www.w...
HP PENTIU...	00,001.00	WebAuction	8,24,1998,16,0	http://www.w...
HP 2000CT...	00,001.00	WebAuction	8,24,1998,16,0	http://www.w...
TOSHIBA PO...	00,105.00	WebAuction	8,24,1998,16,0	http://www.w...
HP 2000 PE	00,001.00	WebAuction	8,24,1998,16..	http://www.w
UMAX 310C ...	00,001.00	WebAuction	8,24,1998,16...	http://www.w...
COMPAQ PR...	00,248.00	WebAuction	8,24,1998,20,0	http://www.w...
UMAX ACTIO...	00,001.00	WebAuction	8,24,1998,16,0	http://www.w...
INTEL PR44...	00,049.00	WebAuction	8,24,1998,16,0	http://www.w...
INTEL MARL...	00,001.00	WebAuction	8,24,1998,16 ..	http://www.w
FIC MOTHE...	00,033.00	WebAuction	8,24,1998,16...	http://www.w...
TC430HX M...	00,001.00	WebAuction	8,24,1998,16...	http://www.w...
33.6K BPS H...	00,001.00	WebAuction	8,24,1998,16,0	http://www.w...
TOSHIBA IN...	00,365.00	WebAuction	8,24,1998,16,0	http://www.w.
HP PAVILIO...	00,131.00	WebAuction	8,24,1998,16,0	http://www.w..
TOSHIBA EQ...	00,144.00	WebAuction	8,24,1998,16,0	http://www.w...
TOSHIBA IN...	00,222.00	WebAuction	8,24,1998,16,0	http://www.w...
TOSHIBA EQ...	00,040.00	WebAuction	8,24,1998,16,0	http://www.w...
COMPAQ DE...	00,131 00	WebAuction	8,24,1998,16,0	http://www.w.
TOSHIBA EQ...	00,027.00	WebAuction	8,24,1998,16..	http://www.w...

64 solution(s) were found for the query

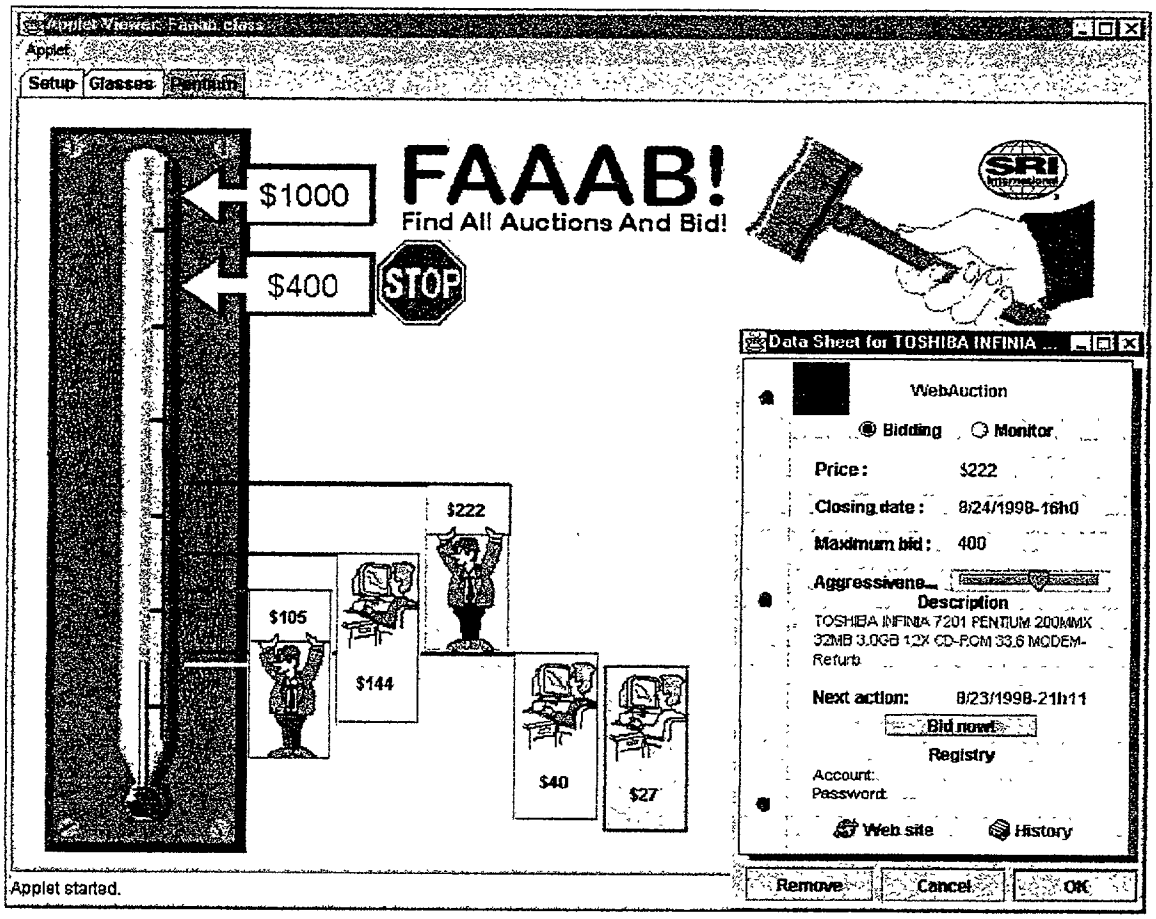
Pentium

TOSHIBA PORTEGE 650CT PENTIUM 133MHZ 1.2GB ...
TOSHIBA EQUIUM 5200D PENTIUM 200MMX 2.5GB ...
TOSHIBA INFINIA 7201 PENTIUM 200MMX 32MB 3.0...
TOSHIBA EQUIUM 6200M PENTIUM PRO 200MHZ 4....
TOSHIBA EQUIUM 5160D PENTIUM 166MMX 2.1GB ...

Figure 1. Creating auction agents for "Pentium" computers

For each domain tab created, a user can graphically view progress and results of auction agents (Figure 2). Agents are classified either as monitor agents, who simply record progress of a particular auction, or bidding agents, who autonomously make bids according to user instructions. The market price (actually, the highest market price for all auction products) and the (highest) max bid are displayed on a gauge. As the auctions unfold, the agents graphically move up the meter, displaying their current prices. Agents who have surpassed their max bid are colored with a red background.

An agent editor enables the user to tailor various properties of the auction agent, such as max bid and aggressiveness. Additional information is also available, such as the bidding history to the current moment, market price for the product, etc.



Data Sheet for TOSHIBA INFINIA ...

WebAuction

● Bidding ○ Monitor

Price: $222

Closing date: 8/24/1998-16h0

Maximum bid: 400

Aggressivene...

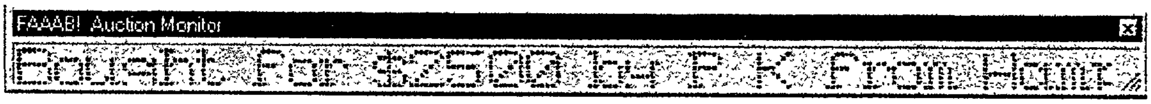
Description

TOSHIBA INFINIA 7201 PENTIUM 200MMX 32MB 3.0GB 12X CD-ROM 33.6 MODEM-Refurb

Next action: 8/23/1998-21h11

Figure 2. Bidding and monitoring agents for "pentium" auctions

Alternate interfaces are also possible. Figure 3 displays a lightweight "banner" interface which unobtrusively keeps the user informed as to updates by his or her auction agents.

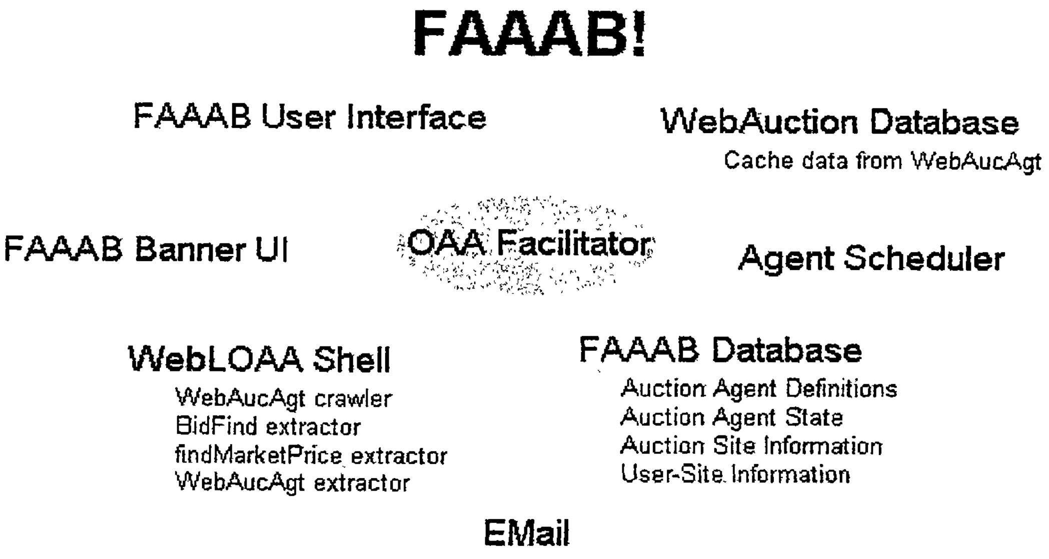


Bought For $2500 by P K From Hom

Figure 3. A lightweight banner interface displaying updates

Prototype Architecture

Figure 3 illustrates the architectural layout of the FAAAB components within the OAA. This section details a few notes on information flow and data storage choices implemented in the prototype.



FAAAB!

FAAAB User Interface

WebAuction Database
Cache data from WebAucAgt

FAAAB Banner UI

OAA Facilitator

Agent Scheduler

WebLOAA Shell
WebAucAgt crawler
BidFind extractor
findMarketPrice extractor
WebAucAgt extractor

FAAAB Database
Auction Agent Definitions
Auction Agent State
Auction Site Information
User-Site Information

EMail

Figure 3. Architecture of FAAAB Prototype

The main FAAAB user interface is accessed from a web browser. According to the FAAAB operational concept, a user will begin by issuing searches for auctions selling interesting products. The results of these searches come from cached data stored in the WebAuction database, and recalculated-on-demand data retrieved by the BidFind extractor agent. WebCrawler caches updates are managed by the Agent Scheduler.

For a given auction found by the above process, the user can query market price information about its products using the findMarketPrice extractor, and view full information about the auction using the UI browser.

A user then selects a subset of auctions to monitor using FAAAB agents -- information about each auction agent is stored in the FAAAB database. A user can edit and tailor agent specific information using the editor provided by the UI.

The Agent Scheduler is notified through OAA's trigger mechanisms of new or modified auction agent definitions. For each auction agent, the scheduler generates a monitoring plan based on auction closing date, importance of the auction site, user-tailorable agressiveness parameters, etc.

P-3970

When an auction agent "checkup" time arrives, the scheduler sends a request for an extractor to read the auction site and retrieve all information about it. If a bid should be made according to the optimal bidding strategies for the user's portfolio, a request is made to available bidding wrapper agents. The results of monitoring and bidding are written to the AucAgtState predicates in the FAAAB database.

The FAAAB user interface and banner user interface both receive update notifications about change in agent state, and display the results accordingly.

An email agent can be used for sending final reports about history and results when an auction closes, and for detecting real-world notifications that another user has outbid you.

Note: the system is extensible and can operate in disconnected mode. As new bidding and monitoring extractor and crawler agents are dynamically added to the system, they will automatically be integrated into the FAAAB process. Disconnected operation is available because both the user interface agents and agent scheduler store all state information in persistent databases and reload this information upon connection at a later time.

Conclusions

The FAAAB prototype illustrates that the construction of an automated meta-auction site management service is a feasible endeavor. The key contributions of the effort are:

- Design and implementation of multiple user interfaces that enable ubiquitous, disconnected access and control to the auction agents.
- Design and implementation of bidding and monitoring strategies and scheduling.
- Integration within a flexible architecture that facilitates light client UIs and complex, distributed, extensible server implementations.
- Selection of a representation language (WebL) for encoding navigational and extraction knowledge for web sites.

Related Work & Resources

1. MIT Media Lab's KASBAH experiment: multi-agent implementation of a commercial marketplace, where both buyers and sellers are represented by agents. What we can learn: parameters and algorithms for automated buying agents.
   http://ecommerce.media.mit.edu/Kasbah/

2. AgentSoft's LiveAgent Pro: A scripting language for automating the web. Semi-automatic generation of scripts through construction through example. Cumbersome to use…
   http://www.agentsoft.com/

3. Digital/Compaq's WebL language: A scripting language implemented in Java which contains powerful "Markup Algebra" and exception handling features. Free!
   http://www.research.digital.com/SRC/WebL

4. BidFinder: An auction meta-site search engine, allowing users to find current auctions for products (from keywords).
   http://www.bidfinder.com/

5. WebAuction: One of the most popular and large online auction sites.
   http://www.webauction.com/

6. Jango: Bought by Excite, the premier product finder on the web. Spinoff from University of Washington (Etzioni & Weld).
   http://www.jango.com/

7. Junglee: Similar to Jango but currently only for resume selling/buying.
   http://www.junglee.com/

Al prov

PROVISIONAL APPLICATION COVER SHEET

This transmittal and the documents and/or fees itemized ~~hereon~~ and attached hereto have been deposited as "Express Mail Post Office to Addressee" in accordance with 37 C.F.R. §1.10 with Express Mail Mailing Label Number EL285395871US

Attorney Docket No.: SRI1P023+
First Named Inventor: CHEYER, Adam J.

Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

□ Duplicate for fee processing

Sir: This is a request for filing a PROVISIONAL APPLICATION under 37 CFR 1.53(c).

INVENTOR(S)/APPLICANT(S)

LAST NAME	FIRST NAME	MIDDLE INITIAL	RESIDENCE (CITY AND EITHER STATE OR FOREIGN COUNTRY)
CHEYER	Adam	J.	Menlo Park, CA
JULIA	Luc	E.	Menlo Park, CA

TITLE OF INVENTION (280 characters max)

USING A COMMUNITY OF DISTRIBUTED ELECTRONIC AGENTS TO SUPPORT A HIGHLY MOBILE, AMBIENT COMPUTING ENVIRONMENT

CORRESPONDENCE ADDRESS

HICKMAN STEPHENS & COLEMAN, LLP
P.O. Box 52037
Palo Alto, CA 94303-0746
(650) 470-7430

ENCLOSED APPLICATION PARTS (check all that apply)

____ Specification Number of Pages ______________ _____ Small Entity Statement

____ Drawing(s) Number of Sheets ______________ __X__ Other (specify) 13 Pages of White Paper Article

__X__ A check or money order is enclosed to cover the Provisional filing fees. Provisional Filing Fee Amount $150

__X__ The commissioner is hereby authorized to charge any additional fees which may be required or credit any overpayment to Deposit Account No. 50-0384 (Order No. SRI1P023+).

The inventions made by an agency of the United States Government or under a contract with an agency of the United States Government.

________ No ________ Yes, the name of the U.S. Government agency and the contract number are:

____

Respectfully Submitted,

SIGNATURE ______________________ **DATE** 3/17/99

TYPED NAME Brian R. Coleman **REGISTRATION NO.** 39,145

PROVISIONAL APPLICATION FILING ONLY

TITLE OF THE INVENTION:

Using a Community of Distributed Electronic Agents to Support a Highly Mobile, Ambient Computing Environment

---

ABSTRACT

Douglas Engelbart asked 30 years ago, at SRI: How can knowledge workers (both individuals and groups) get maximum leverage from personal, networked, interactive computing devices? The twist in the present invention is to redirect this inquiry to the emerging post-desktop world of ubiquitous, highly mobile "information appliances" and PDA's. For example, what sort of computing environment will best serve the PDA-equipped knowledge worker away from the desktop in his/her car, airplane seat, or in a conference room with others? And what software architecture is required to provide that environment effectively? We believe that an "OAA-style" archictecture (facilitated collaboration among distributed agents with declared capabilities in a high-level interagent communication language) has tremendous potential for addressing this challenge. The present invention envisions a new application of this collaborative architecture to address the post-desktop, mobile/ubiquitous computing environment, by incorporating elements like: (a) GPS agents, (b) speech recognition (+ other hands-free UI, multi-modal UI), and (c) opportunistic connectivity among meeting participants (e.g., think of docked or IR-linked PDA's, not just Internet sites). In the specific context of such emerging, ambient computing environments, the distinctive advantages of OAA-style architecture (contrasted with lower-level distributed object approaches like CORBA standing alone), especially with respect to hands-free and multi-modal UI, are even more pronounced.

SUPPLEMENTAL INFORMATION

A November, 1988 OZCHI paper written by Adam Cheyer and Luc Julia entitled: "Cooperative Agents and Recognition Systems (CARS) for Drivers and Passengers" (copy attached) illustrates one example of a possible automobile-based realization of this invention, including GPS and multi-modal UI.

The attached OAA "Scenario" one-page PowerPoint slide illustrates some scenarios for potential interaction and collaboration among PDA-holders in a non-desktop environment like a car (or a conference room), using the technology of the present invention.

A description of multi-modal whiteboard-style collaboration (entitled "SCRIBE") is also attached and may be helpful in preferred embodiments of the present invention.

Pending patent application serial no. ___________ assigned to SRI (docket no. 3949-2) provides a detailed description of the underlying OAA platform architecture, and also specific descrpitions of several applications including "multi-modal maps" which may be helpful in preferred embodiments. The referenced pending patent application is incorporated herein by reference in its entirety.

The published paper Multimodal Maps: An Agent-based Approach, Cheyer & Julia, International Conference on Cooperative Multimodal Communication (CMC/95), 24-26 May 1995 (Eindhoven, The Netherlands), may also be useful for preferred embodiments, and is also incorporated herein by reference in its entirety.

Cooperative Agents and Recognition Systems (CARS) for Drivers and Passengers

Luc E. JULIA
STAR Laboratory
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
USA
julia@speech.sri.com

Adam J. CHEYER
Artificial Intelligence Center
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
USA
cheyer@ai.sri.com

Abstract

In this paper we present SRI's vision of the human-machine interface for a car environment. This interface leverages our work in human-computer interaction, speech, speaker and gesture recognition, natural language understanding, and intelligent agents architecture. We propose a natural interface that allows the driver to interact with the navigation system, control electronic devices, and communicate with the rest of the world much as would be possible in the office environment. Passengers would be able to use the system to watch TV or play games in their private spaces. The final prototype will be fully configurable (languages, voice output, and so forth), and will include speaker recognition technology for resetting preferences and/or for security.

Keywords

Multimodal Interfaces, Speech and Speaker Recognition, Gesture Recognition, Natural Language Understanding, Cooperative Agents.

1. Introduction

New technologies such as Global Positioning System (GPS), wireless phones or wireless internet and electronic controls inside cars are available to improve the way we drive and manage the time spent in our automobiles. To manage this heavy flow of data and to keep the cognitive load as low as possible for the driver, we propose a solution based on our previous developments: a small, speech-enabled, touch display device that provides a combination of the best features of several interfaces we have developed over the past few years. This device can be used according to the specific task that has to be completed by the driver or the passenger.

The interfaces we have developed are the front ends to SRI's powerful framework, the Open Agent Architecture™ (OAA) [18], which allows a community of intelligent agents to work together to achieve user goals. To build multimodal systems, the key agents are those that recognize human signals such as speech or gestures and those that extract the meaning: the natural language understanding agent and the multimodal interpretation agent, for instance.

2. Natural Interfaces

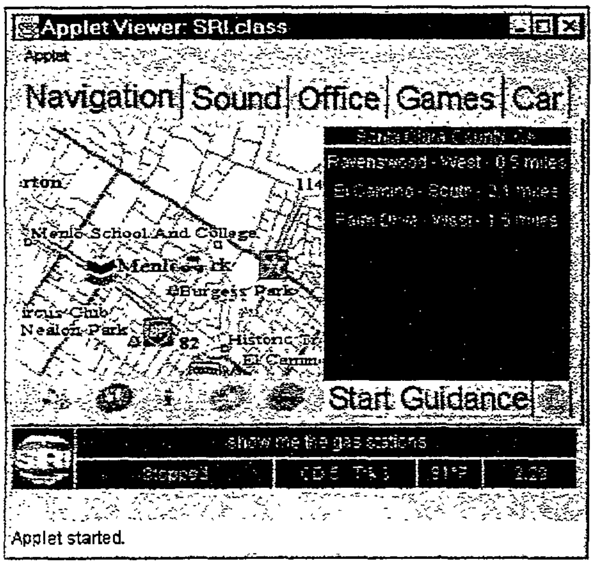
The first prototype we built using Java™ combines different reused interfaces that were chosen according to the task. For each section of the system, we reference the full project for which it was developed. The user can select the tabs using both speech or deictic gestures. Each panel provides its own vocabulary and set of commands in addition to the main commands that allow navigation between the tabs.

2.1. Navigation System

The Multimodal Maps [1] allow the user to navigate maps naturally and query associated databases using speech, handwriting, and 2D gestures in a synergistic fashion on a pen computer. Using the same interface (replacing the pen with the finger) to query the navigation system and to display the GPS information gives the driver or the passengers the ability to plan the route and to get information from local or remote databases displayed on the map ("I want to go to Menlo Park." "Show me the restaurants around here.") (Figure 1). The GPS system guides the car along the chosen route using both the map display and a text-to-speech output. The interactions among all the agents belonging to the system, even if they

P-3967

do not seem to be in use by the current visual interface, enables a great degree of proactivity from the system. For example, it could ask questions such as: "The tank is almost empty; would you like to find the nearest gas station?". As well as a multimodal synergistic input interface, the system provides multimedia outputs such as iconic sounds, discriminative talking voices, images, or videos.



Ravenswood - West - 0.5 miles
El Camino - South - 2.1 miles

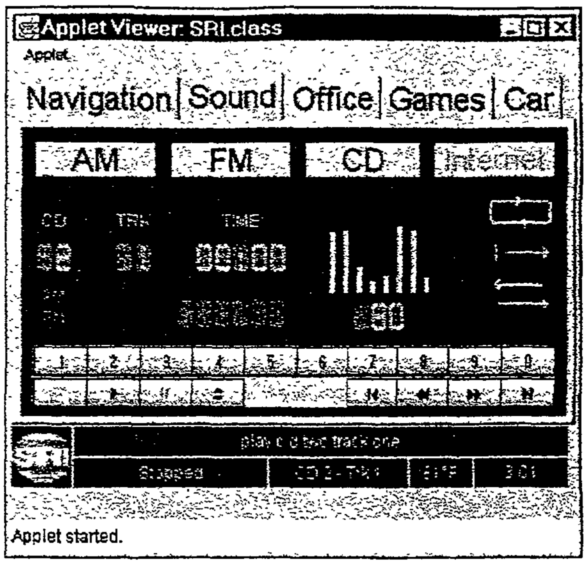
show me the gas stations

Figure 1. Navigation Panel

2.2. Electronic Device Control

Most of the cars will have numerous electronic devices that are accessible through a serial port using a predefined protocol. By connecting a computer and its multimodal interface, it will then be possible for the driver to control critical electronic devices such as cruise control or lights, and for everyone in the car to access comfort devices such as air conditioning, windows, sound, and entertainment ("Play CD 2, track 1.") (Figure 2).

Priority should be given to the driver, possibly through speaker identification. Moreover, an interesting study [11] has shown that it is also possible to use the touch screen in blind condition (for the driver) to enter simple command gestures (down arrow to turn the volume down for instance).

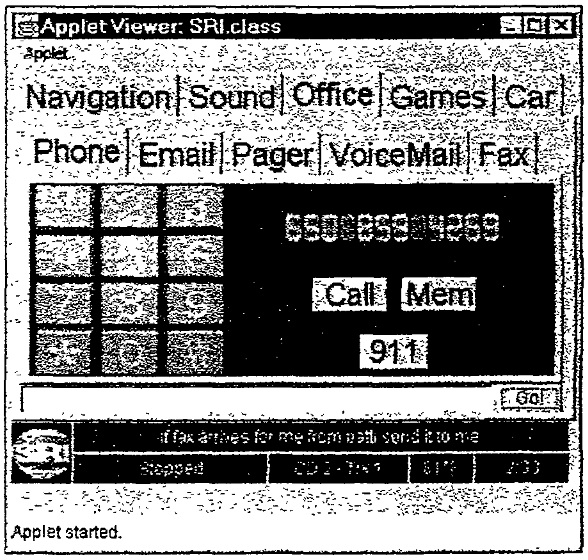


play c d two track one

Figure 2. Sound System Panel

2.3. Communication Center

The communication center is a remote office accessible by voice (Figure 3).



if fax arrives for me from patti send it to me

Figure 3. Communication Center Panel

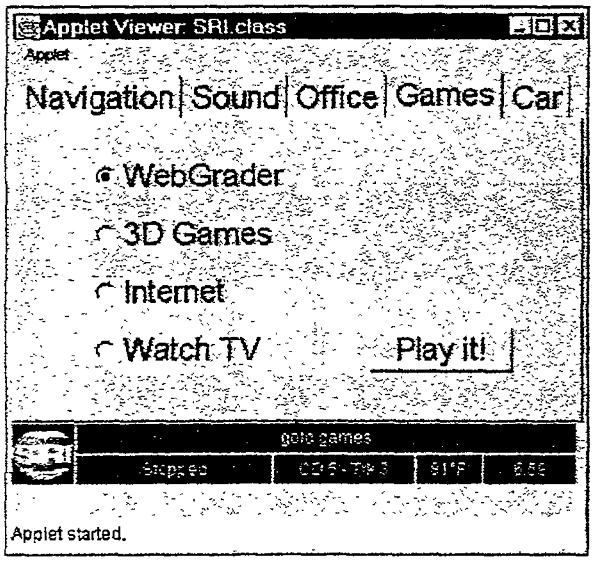
This is an instance of the Automated Office (Unified Messaging), developed to show some capabilities of OAA [18]. The driver or passengers are able to browse the incoming emails by voice (even multipart/multimedia MIME messages), make phone calls, or send spoken notes. As basic features of OAA, filtering and triggering capabilities are included in each connected agent: "If email arrives for me about OzCHI, read it to me." Plugging in an agent using speaker identification tech-

P-3967

niques allows commands such as "If voicemail arrives for me from Larry, send an email to Patti" [9]. Intelligent cross-media conversion and adaptability to the current set of available or desired output channels is a key characteristic of the Unified Messaging prototype.

2.4. Recreation Area

The recreation center gathers some of the innovative speech-enabled prototypes developed by SRI International. Passenger-oriented, it assumes that each passenger creates a private multimodal/multimedia area (close talking microphone, personal touch screen and headsets). The passenger can play impressive 3D games enhanced with speech commands, search the Internet by voice, talk to an animated avatar, and watch TV in a more interactive way by asking naturally for the available programs with specific features. In addition, speech-based educational systems, such as WebGrader™ [16], provide a fun and effective way to learn foreign languages and their pronunciations (Figure 4).



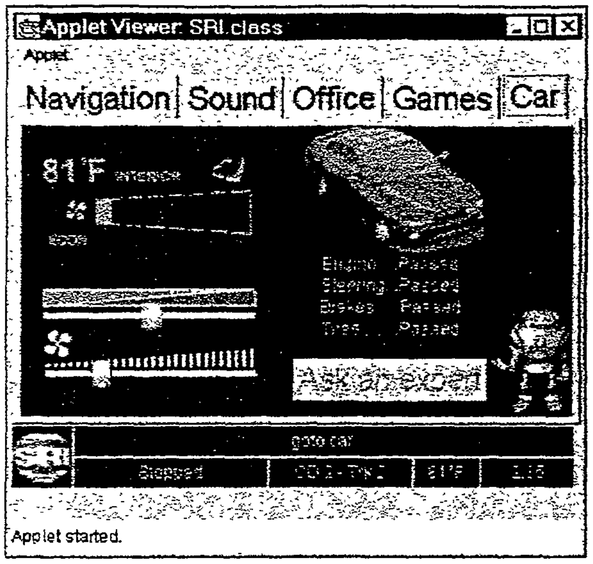
● WebGrader
○ 3D Games
○ Internet
○ Watch TV

Figure 4. Recreation Panel

2.5. Technical Information Access

The entire documentation of the car will be available on the Internet, making it easy to keep it up to date and possibly to personalize (via cookies) and control (via certificates) data for the car. This section extends the idea of the dialog with an avatar, or actor, implemented using the Microsoft^{TM} Agent graphics [19].

If a warning message appears from a monitored device, a dialog with an automobile expert, played by the actor, will help to diagnose and fix the problem (Figure 5). The expert may also answer common questions such as "How much air should I put in my tires?" or "How should I talk to you?"

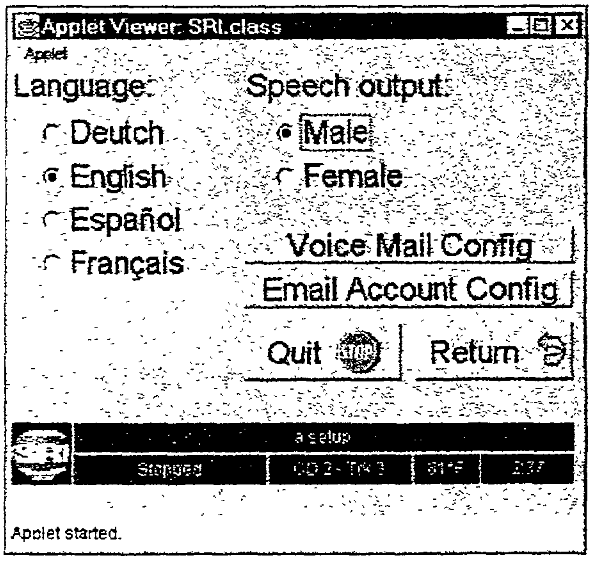


Engine Passed
Steering Passed
Brakes Passed
Tires Passed

Figure 5. Diagnostic Panel

2.6. Setups

Speaker verification techniques can be used to access the setup panel and private areas (to configure and define the passwords for email and voice mail accounts, for instance). It will also be used to automatically retrieve the preferences for the current driver with respect to seat position, radio selections, temperature, mirror direction, and so forth.



Language:
○ Deutch
● English
○ Español
○ Français

Speech output:
● Male
○ Female

Figure 6. Setup Panel

P-3967

3. Behind the Scene: the Agents

The functionality described above requires multiple Artificial Intelligence (AI) technologies (e.g., speech and gesture recognition, natural language understanding) to interact with each other and with commercial, off the shelf components such as email systems, map databases, and car electronics. SRI's Open Agent Architecture provides an infrastructure for integrating distributed components in a more flexible way than can be done through other distributed technologies such as CORBA, COM, or Java's RMI. The key difference in OAA's approach is that instead of components writing code to specify (and fix) their interactions and dependencies with other components, each agent (component) expresses its capabilities and needs in terms of a higher-level Interagent Communication Language (ICL). Each request for information or action is handled by one or more "facilitator agents," who break the request into subtasks, allocate subtasks to agents able to perform them, and then coordinate the flow of data and control among the participants. The architecture offers built-in support for creating natural user interfaces to the distributed services, since the logic-based ICL can be translated from and to natural language; users can speak a request in English, and the request can be acted upon by the community of agents, without requiring the user to specify or even know which agents are involved.

The advantages of the OAA approach include true plug-and-play, with new agents able to join the community of services at runtime; managed coordination of cooperative and competitive parallelism among components; heterogeneous computing, with components existing on diverse platforms. written in many programming languages; and enhance code reuse. Since components do not have hard-code dependencies written into them, we will be able to incorporate many existing agents from previous OAA-enabled systems [13, 18].

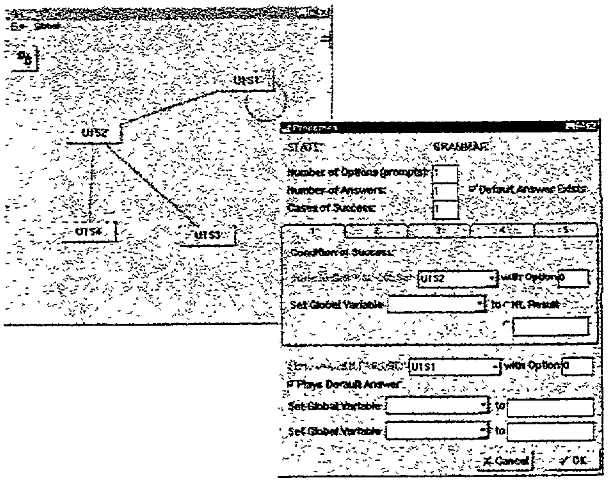
4. Recognition and Interpretation

4.1. Speech Recognition

Speech recognition, along with natural language, is a huge component of the multimodal user interface. While it is possible to use any speech recognition product available on the market to make an agent, we prefer the Nuance Communications[1] recognizer. Nuance is a real-time version of the SRI STAR Laboratory's continuous speech recognition system using context-dependent genonic hidden Markov models (HMMs) [4]. This technology recognizes natural speech without requiring the user to train the system in advance (i.e., speaker-independent recognition) and can be distinguished from the few other leading-edge speech recognition technologies by its detailed modeling of variations in pronunciation and its robustness to background noise and channel distortion. We plan to investigate automobile environments in more detail.

4.2. Natural Language Understanding

In most OAA-based systems, prototypes are initially constructed with relatively simple natural language (NL) components, and as the vocabulary and grammar complexities grow, more powerful technologies can be incrementally added. It is easy to integrate different levels of NL understanding, depending upon the requirements of the system, just by plugging in an adequate engine. The available engines are two of our low-end NL systems: Nuance's template-slot tools and DCG-NL, a Prolog-based top-down parser. SRI's GEMINI [5] and FASTUS [7] are more powerful tools, used for complex NL tasks. To design the dialog on the fly, a visual tool is under development (Figure 7). It simulates the behaviors of the NL engine and creates the necessary code and data for the final NL agent.



Number of Options (prompts): 1
Number of Answers: 1
☑ Default Answer Exists
Cases of Success: 1
Condition of Success:
UIS2 with Option 0
Set Global Variable: to ○ NL Result
UIS1 with Option 0
☑ Plays Default Answer
Set Global Variable: to
Set Global Variable: to

Figure 7. Visual Design Tool

4.3. Speaker Identification

Speaker identification technology has seen significant progress over the past several years. Although good performance can be achieved, several parameters affect accuracy. For example, systems trained on larger amounts of speech from the users will be more accurate. Similarly, variety in the training data (collecting over several days) will improve system robustness, and accuracy is higher for

[1] SRI spin-off: http://www.nuance.com

longer test utterances. Computational limitations of the onboard platform will also be a performance factor. Perhaps the most significant of the factors is the variety in training data. The effects of mismatches between training and testing conditions can be dramatic [17]. A severe example, in the context of cars, of mismatching conditions would occur when a user trains the system with the engine off, in the garage, and then uses it with the top down on the freeway at high speed. We have made significant progress in reducing adverse effects of mismatches between training and testing conditions. In particular, SRI has developed a technology that reduces the effect from a factor of 30 to a factor of less than 3 [6]. The technology enables the user to train in a single session (one acoustic environment).

4.4. Gesture Recognition

The gesture modality is usually used in conjunction with speech to add spatial or deictic data to issue commands to the system. But sometimes a gesture (like a crossout) can carry both a location and semantic content. A set of current gestures (Figure 8) can be recognized using algorithms developed in [8].

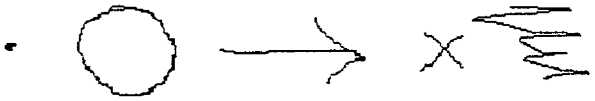


Figure 8. Gesture Set

In our experience [2], most gestures produced by users fall into this set. Since handwriting has rarely been used but we want to provide as many modalities as possible, we incorporated Communications Intelligence Corporation (CIC[2]) recognition routines. The handwriting recognizer is of interest in the navigation task where out-of-vocabulary names may appear, which are normally difficult for speech recognition systems to handle.. Both the gesture recognizer and the handwriting recognizer are competing on the same data to find the right meaning.

4.5. Multimodal Fusion

Even if we consider speech as a privileged modality [10], numerous user studies [e.g., 12] have shown that most subjects prefer combinations of spoken and gestural inputs. In such examples, whereas speech plays a strong role in the acquisition of commands, combining it with a pointing device provides significant (8%) improvement in performance (recognition and understanding) over the use of speech in isolation. Not surprisingly, gestures provide a fast and accurate means of locating specific objects, while voice commands are more appropriate for selecting describable sets of objects or for referring to objects not visible on the screen. Many of these studies also attempt to enumerate and classify the relationships between the modalities arriving for a single command (complementary, redundancy, transfer, equivalence, specialization, contradiction). To model interactions where blended and unsorted modalities may be combined in a synergistic fashion with little need for time stamping, we first proposed a three-slot model known as VO*V* (Figure 9), such that

V or Verb is a word or a set of words expressing the action part of a command.

O* or Object[s] is zero or more objects to which the verb applies (zero if it is a system command).

V* or Variable[s] is zero or more attributes or options necessary to complete the command.

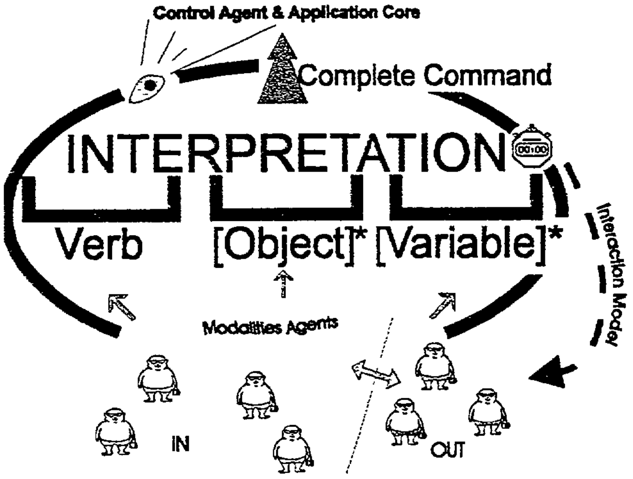


Figure 9. VO*V* Model

Input modalities produced by the user (handwriting, speech, gestures) fill slots in the model, and interpretation occurs as soon as the triplets produce a complete command. A multimedia prompting mechanism is also provided to assist the user in fulfilling an incomplete command. In addition, multiple information sources may compete in parallel for the right to fill a slot, given scored modality interpretations. This model has been shown to be easily generalizable, and has been applied to various application domains

[2] SRI spin-off: http://www.cic.com/

P-3967

5. Evaluation

When building complex systems, it is important to perform user experiments to validate the design and implementation of the application. As described in [2], we have developed a novel "hybrid Wizard-of-Oz" approach for evaluating how well the implemented system functions for an experienced user, while simultaneously gathering information about future extensions or improvements as dictated by new users. The technique promotes incremental development of a complex system, from initial prototype through tested product, and provides a means for logging user interactions and quantifying system improvements at every stage of development.

6. Conclusions and Future Work

The unique feature of the proposed approach is that we integrate several very distinctive pieces, thanks to the OAA, even though they were not intended for this purpose. Further, we unify those pieces through a common, natural, multimodal interface using as much as possible human-to-human communication to avoid adding cognitive overload to the user. We achieved most of this aim by using good recognition systems and effective fusion and presentation techniques But to improve the reliability and robustness of the speech recognizer in real cars, we still have to address considerable noise and speaker adaptation issues (see, e.g. [3, 14, 15]). Finally, within a short period of time we plan to hook up real GPS and navigation systems and install our system in a moving car so that we can conduct user testing in real-life conditions.

7. Acknowledgments

Many thanks to Patti Price, Director of the Speech Technology and Research lab who spent a lot of time helping to design the CARS system.

8. References

[1] A. CHEYER and L. JULIA, "Multimodal Maps: An Agent-based Approach," Proc. of Cooperative Multimodal Communication (CMC'95): Eindhoven, The Netherlands, 1995.

[2] A. CHEYER, L. JULIA and J. C. MARTIN, "A Unified Framework for Constructing Multimodal Experiments and Applications." Proc. of Cooperative Multimodal Communication (CMC'98): Tilburg, The Netherlands, 1998.

[3] V. DIGALAKIS and L. NEUMEYER, "Speaker Adaptation Using Combined Transformation and Bayesian Methods," Proc. of Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP'95): Detroit, USA, 1995

[4] V. DIGALAKIS, P. MONACO and H. MURVEIT, "Genones: Generalized Mixture Tying in Continuous Hidden Markov Model-Based Speech Recognizers," IEEE Transactions of Speech and Audio Processing, Vol.4, Num. 4, 1996

[5] J. DOWDING, J.M. GAWRON, D. APPELT, J. BEAR, L. CHERNY, R. MOORE and D. MORAN, "GEMINI: A natural language system for spoken-language understanding," Proc. of 31st Annual Meeting of the Association for Computational Linguistics (ACL'96): Columbus, USA, 1996.

[6] L. HECK and M. WEINTRAUB, "Handset dependent background models for robust text-independent speaker recognition," Proc. of Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP'97): Munich, Germany, 1997.

[7] J. HOBBS, D. APPELT, J. BEAR, D. ISRAEL, M. KAMEYAMA, M. STICKEL, and M. TYSON, "FASTUS: a cascaded finite-state transducer for extracting information from natural-language text," in Finite State Devices for Natural Language Processing (E. Roche and Y. Schabes, eds.) MIT Press, Cambridge, USA, 1996.

[8] L. JULIA and C. FAURE, "Pattern Recognition and Beautification for a Pen Based Interface," Proc. of Intl. Conference on Document Analysis and Recognition (ICDAR'95): Montréal, Canada, 1995.

[9] L. JULIA, L. HECK and A. CHEYER, "A Speaker Identification Agent," Proc. Audio and Video-based Biometric Person Authentication (AVBPA'97): Crans-Montana, Switzerland, 1997.

[10] L. JULIA and A. CHEYER, "Speech: A Privileged Modality," Proc. of EuroSpeech'97: Rhodes, Greece, 1997.

[11] J. F. KAMP, F. POIRIER and P. DOIGNON, "A New Idea to Efficiently Interact with In-Vehicle Systems. Study of the use of the Touchpad in "Blind Condition," Poster Proc. of Human Computer Interaction (HCI'97): San Francisco, USA, 1997.

[12] B.A. MELLOR, C. BABER and C. TUNLEY, "In goal-oriented multimodal dialogue systems," Proc. of Intl. Conference on Spoken Language Processing (ICSLP'96): Philadelphia, USA, 1996.

[13] D. MORAN, A. CHEYER, L. JULIA, D. MARTIN and S. PARK, "The Open Agent Architecture and Its Multimodal User Interface," Proc. of Intelligent User Interfaces (IUI'97): Orlando, USA, 1997.

[14] L. NEUMEYER and M. WEINTRAUB, "Probabilistic Optimum Filtering for Robust Speech Recognition," Proc. of Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP'94): Adelaide, Australia, 1994.

P-3967

[15] L. NEUMEYER and M. WEINTRAUB, "Robust Speech Recognition in Noise Using Adaptation and Mapping Techniques," Proc. of Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP'95): Detroit, USA, 1995.

[16] L. NEUMEYER, H. FRANCO, V. ABRASH, L. JULIA, O. RONEN, H. BRATT, J. BING and V. DIGALAKIS, "WebGrader: A multilingual pronunciation practice tool," Proc. of Speech Technology in Language Learning (STiLL'98): Stockholm, Sweden, 1998.

[17] NIST Speaker Recognition Workshop: Linthicum Heights, USA. 1996.

[18] SRI International web site on the Open Agent Architecture: http://www.ai.sri.com/~oaa/applications.html

[19] Microsoft web site about their agents and their animations: http://www.microsoft.com/workshop/imedia/agent/

P-3967

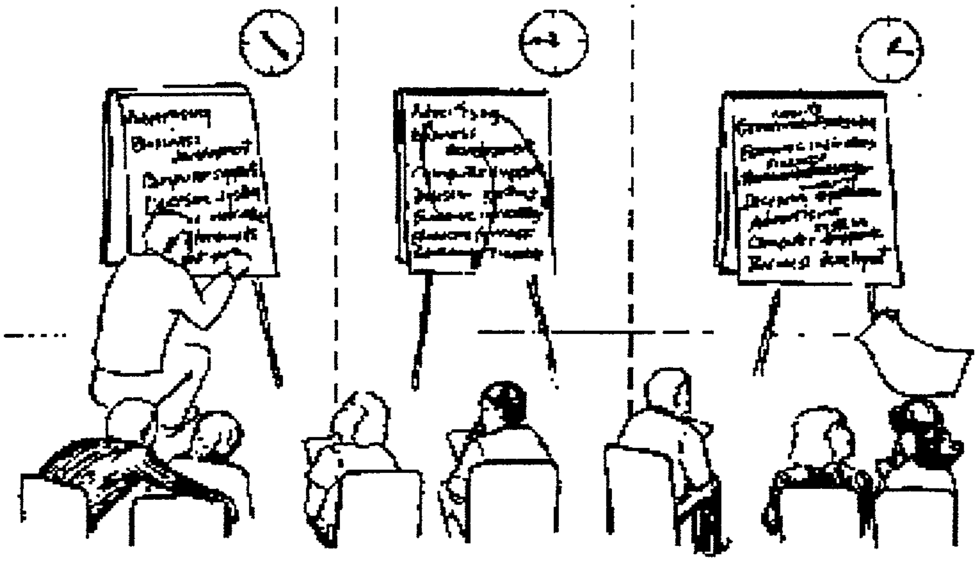
P-3967

A first step toward the MAGIC*:

SMARTMeetings

You donít have to be miles away to collaborate, but if you are thatís OK!

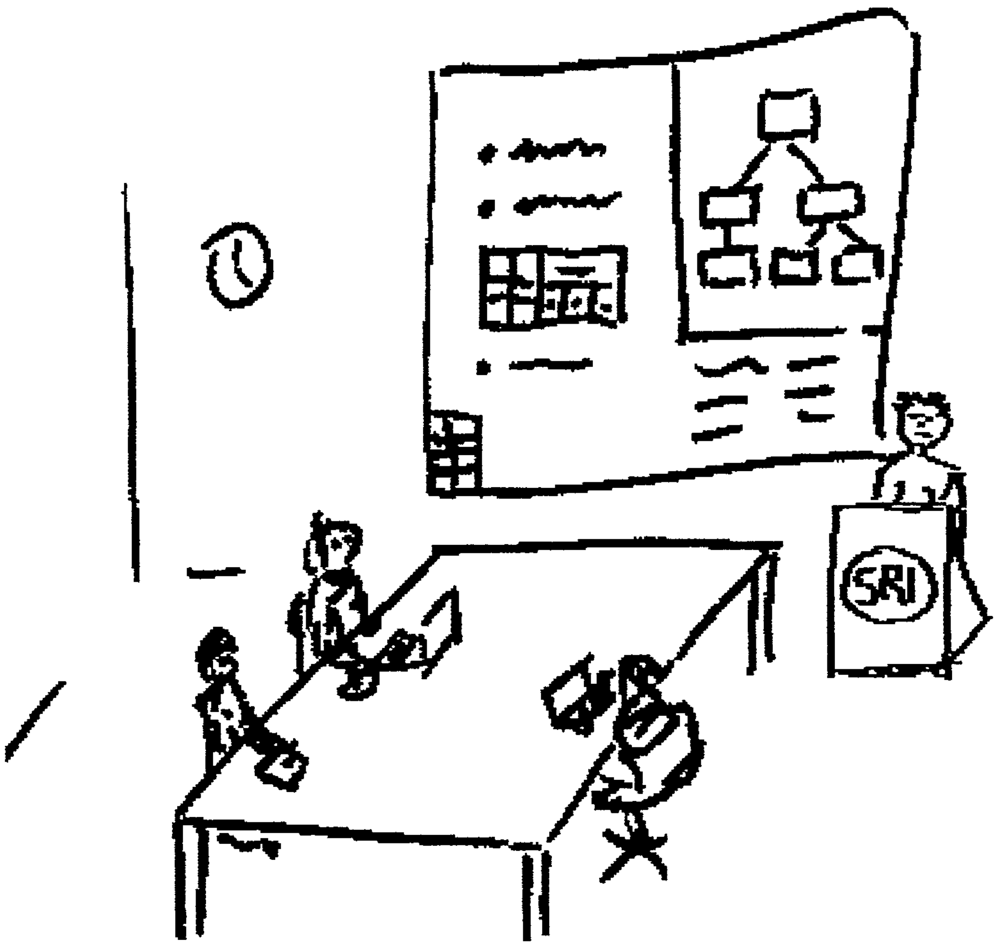
PAST



60124718.031799

P-3967

VISION



The SMARTMeetings room: each seat provides an internet connection and a pen tablet. The large screen display is the shared space. A broader vision accepts heterogeneous machines (PCs, PalmPilots, etc...)

The SCRIBE system, a reactive board metaphor. It gives immediate, beautified, feedback on the collaborative space for handwriting and drawings. Erase/Correction functions are available through natural gestures. Usage of colors possible.

The leader of the meeting, a privileged user, interacts with the board at the podium, and can also gather information from the other participants by giving them electronic access to the board in order to share their ideas. He or she is the facilitator.

More informal meetings as well as many meeting styles are falling in this paradigm. For instance, every attendant might be a privileged user who talks, writes and draws carefully around the table in order to be well understood by the other participants, and the recognizers. The shared, collaboratively built, document is projected

Location of the attendants is indifferent, they need a connected pen/multimedia computer.

SRI INTERNATIONAL

P-3967

Automatic production of clean documents, minutes, etc... History of production available. Immediate distribution, copies are stored on attendants computers. Users can import pre-meeting notes, specific backgrounds (maps, charts, etc...) in order to produce nicer documents. But we anticipate that most of the time all data will be produced on site, during the meeting.

TECHNOLOGIES

Collaborative Application.

Handwriting Recognition.

Drawing Recognition - Boxes, charts, tables... - Meta gestures (erase, move, etc...).

Speech Recognition (? Or limited) - Speaker ID.

(?Stereo) Vision - 3D gestures, meeting layout.

OUTCOMES

Working prototype.

Patent for the (ri's ollaborative, eactive and ntelligent oard nhancer).

BACKGROUND

The CDL is a good start. Tablets and Internet hub to add.

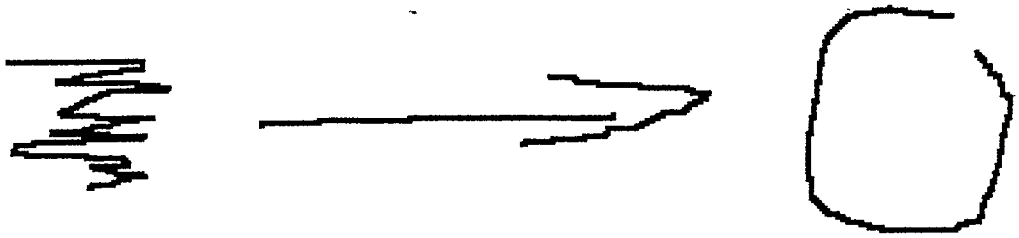
In Multimodal Maps, using OAA, we have some sharing/collaboration mechanisms.

Avery-Dennison "smart pen" idea. Pieces were there, but wasn't the technology too complex?

CIC provides good Handwriting Recognition engine. Already integrated in most projects.

STAR's Speech Recognition and Speaker ID expertise.

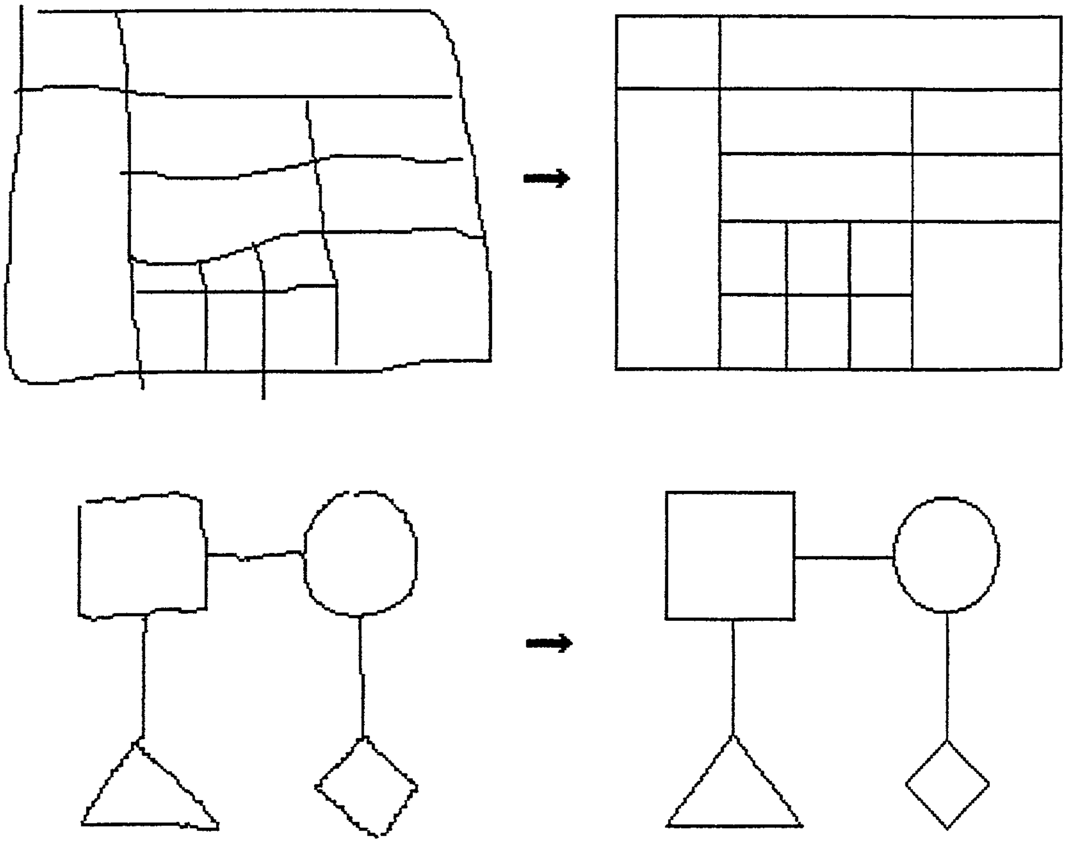
Gesture Recognition in use in all SRI's Multimodal projects.



Multimodal fusion, resolving ambiguities, algorithms competition and complementary

TAPAGE and DERAPAGE: on the fly graphics recognition and semantic interpretation.

P-3967



: ultimodal ccess and eneration for nteraction and ollaboration

CHIC!

Computer Human Interaction Crew

Send comments and suggestions to Dr. Luc JULIA Luc.Julia@sri.com

EXPERT DECLARATION OF DR. KEVIN NEGUS
FOR
INTER PARTES REVIEW OF U.S. PATENT NO. 6,757,718

TABLE OF CONTENTS

I. INTRODUCTION

1. I, Dr. Kevin Negus, submit this declaration in support of a Petition for *Inter Partes* Review of United States Patent Nos. 6,757,718 (the "'718 Patent"), owned by IPA Technologies L.L.C. ("IPA" or "Patent Owner"). I have been retained in this matter by Baker Botts LLP ("Counsel") on behalf of DISH Network L.L.C. ("Petitioner"). Petitioner DISH Network L.L.C. and DISH Network Corporation ("DISH") are the Real Parties-in-Interest to this Petition. DISH is a provider of direct broadcast satellite (DBS) services.

2. I make this declaration based upon my personal knowledge. I am over the age of 21 and am competent to make this declaration.

3. The statements herein include my opinions and the bases for those opinions, which relate to at least the following documents of the pending *inter partes* review petition:

- U.S. Patent No. 6,742,021 by Christine Halverson, Luc Julia, Dimitris Voutsas, and Aden J. Cheyer, entitled "Navigating Network-Based Electronic Information Using Spoken Input with Multimodal Error Feedback" (the "'021 Patent") (Ex. 1001).
- File History for U.S. Patent No. 6,742,021 (Ex. 1002).
- U.S. Patent No. 6,757,718 by Christine Halverson, Luc Julia, Dimitris Voutsas, and Adam Cheyer, entitled "Mobile Navigation of Network-Based Electronic Information Using Spoken Input" (the "'718 Patent") (Ex. 1003).
- File History for U.S. Patent No. 6,757,718 (Ex. 1004).
- U.S. Patent No. 6,523,061 by Christine Halverson, Luc Julia, Dimitris Voutsas, and Adam Cheyer, entitled "System, Method, and Article of Manufacture for Agent-Based Navigation in a Speech-Based Data Navigation System" (the "'061 Patent") (Ex. 1005).
- File History for U.S. Patent No. 6,523,061 (Ex. 1006).

- U.S. Patent No. 6,851,115 by Christine Halverson, Luc Julia, Dimitris Voutsas, and Adam Cheyer, entitled "Software-Based Architecture for Communication and Cooperation Among Distributed Electronic Agents" (the "'115 Patent") (Ex. 1007).
- File History for U.S. Patent No. 6,851,115 (Ex. 1008).
- File History for U.S. Patent Application No. 60/124,720 (Ex. 1009).
- File History for U.S. Patent Application No. 60/124,719 (Ex. 1010).
- File History for U.S. Patent Application No. 60/124,718 (Ex. 1011).
- U.S. Patent No. 5,500,920 by Julian M. Kupiec, entitled "Semantic Co-ocurrence Filtering for Speech Recognition and Signal Transcription Applications" ("Kupiec") (Ex. 1013).
- U.S. Patent No. 6,006,227 by Eric Freeman et al., entitled "Document Stream Operating System" ("Freeman") (Ex. 1014).
- U.S. Patent No. 5,247,580 by Toshiyuki Kimura et al., entitled "Voice-operated remote control system" ("Kimura") (Ex. 1015).
- Complaint, *IPA Technologies Inc. v. DISH Network Corp. et al.*, No. 1:16-cv-01170 (D. Del.) ("District Court Litigation") (Ex. 1016).
- Non-patent literature publication by Adam J. Cheyer and Luc Julia, entitled "Multimodal Maps: An Agent-based Approach" ("Cheyer") (Ex. 1019).

4. My materials considered for forming my opinions herein have included at least the above-referenced documents.

5. Although I am being compensated for my time at a rate of $500 per hour in preparing this declaration, the opinions herein are my own, and I have no stake in the outcome of

the review proceeding. My compensation does not depend in any way on the outcome of the Petitioner's petition.

II. QUALIFICATIONS

6. I am qualified by education and experience to testify as an expert in the field of telecommunications. Attached, as Attachment A, is a copy of my resume detailing my experience and education. Additionally, I provide the following overview of my background as it pertains to my qualifications for providing expert testimony in this matter.

7. I am a Full Professor of Electrical Engineering at Montana Tech University in Butte, MT. I lead a research program at Montana Tech to improve the delivery of mobile broadband communications services to rural and remote areas and to enable communications from sensors in challenging locations such as wildlife, drill holes, underground mines or long-haul electric transmission towers. I mentor, supervise and teach both senior undergraduate and graduate students of Electrical Engineering in the general fields of telecommunications and networking with an emphasis on wireless systems.

8. In 1988, I received my Ph.D. in Engineering from the University of Waterloo in Canada. The Departments of Electrical Engineering and Mechanical Engineering jointly supervised my Ph.D. research on the modeling of bipolar semiconductor devices. My graduate course work was primarily in Electrical Engineering and included such subjects as semiconductor device physics and fabrication, wireless circuit design, and wireless propagation analysis. For my Ph.D. work, I received the Faculty Gold Medal in 1988 for the best Ph.D. thesis in the entire Faculty of Engineering across all Departments for that year. My Ph.D. thesis research also formed the basis of a paper published in 1989 that won the award for Best Paper in 1989 for the IEEE (Institute of Electrical and Electronic Engineers) journal in which it was published.

9. In 1984 and 1985, respectively, I received the B.A.Sc. and M.A.Sc. Degrees in Mechanical Engineering from the University of Waterloo in Canada. My coursework and research work included, amongst many other topics, extensive embedded firmware development for automation applications and implementation of networks and communications protocols. For my M.A.Sc. Degree research and academic achievements, I received the prestigious University Gold Medal in 1985 for the best Masters thesis in the entire University of Waterloo for that year.

10. In 1986, I joined the Palo Alto Research Center of Fairchild Semiconductor in Palo Alto, CA. At Fairchild, I participated in the development of devices and products for high-speed applications such as wired networking, RISC microprocessors and wireless communications.

11. In 1988, I took the position of Member of the Technical Staff at Avantek, Inc. in Newark, CA. I was hired to develop products for both wireless and wired data networking applications. Some of the components I developed early in my career at Avantek were used for 1st generation wireless local area network (WLAN) products, voice band modem equipment, wired data networking both in the LAN and WAN and 1st generation cellular handsets and base stations based on AMPS or TACS.

12. In 1991, the Hewlett-Packard Company purchased Avantek, Inc. I continued to work for Hewlett-Packard until 1998 in such roles as IC Design Manager, Director of Chipset Development and Principal System Architect. In 1992, Hewlett-Packard assigned me to work on the "Field of Waves" project, which was a major multi-division effort to build WLAN products for mobile computers. The project was cancelled in 1993. However, the work I did on the project was leveraged into producing the world's first IEEE 802.11 chipset, which my division at Hewlett-Packard first offered for sale in 1994. I led the project to develop and market this

chipset for many early WLAN product companies including Proxim, Symbol (now part of Motorola) and Aironet (now part of Cisco). I also helped coordinate efforts within Hewlett-Packard to guide extensive research projects on WLAN protocols and technology at Hewlett-Packard's central research laboratories in Palo Alto, CA and Bristol, U.K.

13. I developed or led the development of multiple chips and chipsets for 2G cellular radio systems based on GSM, IS-54 (TDMA), and IS-95 (CDMA). A number of these chips were directed solely to cellular mobile stations and done specifically for major Hewlett-Packard customers and cellular handset and module manufacturers such as Motorola, Ericsson and Siemens. I was also involved in the development of power amplifier chips and modules for cellular mobile stations, cordless phones, wireless networking devices and cellular infrastructure products including those directed towards then emerging 3G cellular standards such as WCDMA, 1xRTT and EV-DO.

14. During my time at Avantek and Hewlett-Packard, I also developed or led development teams for numerous chipsets or general purpose chips used in other wired and wireless communications applications such as fiber optic transceivers, cordless telephones, cable set-top receivers, wired networking equipment, cellular infrastructure equipment, voice band and broadband wired modems and satellite TV receivers.

15. In 1998, I joined Proxim, Inc. in Mountain View, CA. At that time, Proxim was engaged in the development and sale of wired and wireless products for home and enterprise networking applications based on several different wired and wireless networking protocols. I stayed at Proxim through 2002 and was the Chief Technology Officer for this publicly-traded company at the time of my departure. During my career at Proxim, I led or participated in the development of many WLAN and WWAN products and/or chipsets for network adapters, OEM

design-in modules, access points, bridges, switches, and routers that used a wide variety of bus, LAN, or WAN wired interfaces. I have supervised many engineers including those responsible for embedded firmware development to implement various wired and wireless networking, reservation, and security protocols at the MAC layer and above, those responsible for HDL code creation of baseband chips to implement PHY and MAC algorithms, as well as other engineers that developed hardware reference designs, modem algorithms and chipsets.

16. During my many years of development of products providing voice, data and/or streaming media capabilities, I have acquired a deep understanding of the cellular radio system, the Public Switched Telephone Network (PSTN) and the public Internet network architectures and protocols. A partial list of networking and telephony protocols that I am familiar with includes DHCP, SNMP, TCP, UDP, IP, SIP, ICMP, SS7, ISDN, ISUP, TCAP, and MTP.

17. Over the past 30 years I have personally developed, modified, or analyzed numerous software or firmware modules for many different applications as well as supervised many engineers performing the same tasks. I have implemented or supervised the implementation of software and firmware code and/or hardware description language (HDL) code for many different communications protocols across all layers. I have developed or supervised the development of chips with both wireless baseband modem functionality and embedded processors including those licensed by ARM and MIPS. I have programmed with multiple high level languages for software and firmware code including C, C++, Fortran, Forth, BASIC, Pascal, Lisp and COBOL. I have developed products with HDL code including VHDL and Verilog. I also have firsthand experience with assembly language programming. I have personally designed a wide variety of analog, RF, and digital circuit elements at both the chip

and board level using various netlist-driven, schematic capture and manual or automated layout CAE/CAD tools.

18. Since 2002, I have been an independent consultant and have provided services to a number of companies including some that have developed IEEE 802.11 products. In particular, from 2002 until 2007 I was Chairman of WiDeFi, Inc. – a company that developed chips and embedded firmware for 802.11 repeater products based on 802.11a, b, g and draft n amendments. From 2007-2011, I was Chairman of Tribal Shout – a company that delivered IP voice and audio streaming media using VoIP to any cellular or landline phone including those reachable only by the circuit-switched connections such as the PSTN and 2nd generation cellular radio. From 2010-2016, I was Chairman and Chief Technology Officer of CBF Networks, Inc. (dba Fastback Networks) – a company that developed fiber extension products for backhaul of data networks including Wi-Fi, HSPA, CDMA2000, WiMAX and LTE cellular radio systems. I architected the products of Fastback Networks specifically around the re-use of chips originally developed and intended for LTE standards-based operation and for carrier-grade Ethernet network interface cards, switches and/or routers.

19. I have been a Board Observer on behalf of the venture capital firm Camp Ventures at two companies that develop semiconductor components including one that developed technology specifically to improve the system performance of HSPA and LTE cellular radio systems (Quantance) and another that provides system on a chip (SOC) microcontrollers, OEM design-in modules and firmware with 802.11 and wired interfaces for embedded applications (GainSpan). I have also been a technology and/or business strategy advisor to multiple early stage companies that are developing such products as new wireless

communications security systems (AirTight), RFID radio systems (Mojix) and time/frequency reference components (SiTime).

20. I have actively monitored or participated in the IEEE 802.11 standards process continuously since 1989. I am a listed contributor to the highly successful IEEE 802.11g standard published in 2003 that describes a wireless communications protocol in use worldwide by over 5 billion devices. In 2002 and 2003, I participated in the IEEE 802.11 Wireless Next Generation Committee that was responsible for launching the 802.11n standards development process.

21. In 1996, I was assigned the responsibility within the Hewlett-Packard Company for developing the HomeRF standard for WLANs specifically for home networking applications. I eventually became Chairman of the Technical Subcommittee of HomeRF that wrote the HomeRF standard. The HomeRF standard was essentially a modification of the IEEE 802.11 standard with significant changes to the PHY and MAC layers to lower cost and improve performance and security for home networking applications including integrated voice capability over both IP and circuit-switched connections. From 1998 to 2002, millions of wireless network adapters and access points from several different companies were shipped based upon compliance to the HomeRF standard.

22. I have specific experience with many wired and wireless networking standards including IEEE 802.1 and 802.3 (the "Ethernet" family of wired LANs), IEEE 802.11 (the "Wi-Fi" family of wireless LANs), IEEE 802.15 (personal area networks or "PAN"), IEEE 802.16 (also known as "WiMAX"), various cellular communications standards (such as IS-19, IS-41, IS-54, IS-95, IS-136, IS-826, IS-707, IS-856, IS-2000, CDPD, GSM, GPRS, EDGE, UMTS,

CAMEL, WCDMA, HSPA, and LTE), various cordless telephone standards (such as CT-2, DECT, and PHS), and other wired networking standards (such as DOCSIS, SONET and FDDI).

23. I am an author or co-author of many papers that have been published in distinguished engineering journals or conferences such as those of the IEEE or ASME. An exemplary list of these publications is included in my resume, and I believe that this list includes all publications I have authored at least in the past ten years.

24. I am also a former member of the Federal Communication Commission's Technological Advisory Committee as an appointee of then Chairman Michael Powell. I have also served on the Wyoming Telecommunications Council as an appointee of then Governor Jim Geringer after confirmation by the Wyoming State Senate.

25. I am named as an inventor on numerous U.S. patents all of which have related in at least some way to products for wired and/or wireless networks. I believe that the following is a complete list as of this date for my approximately 58 issued U.S. Patents: 4,839,717, 5,111,455, 5,150,364, 5,436,595, 5,532,655, 6,587,453, 7,035,283, 7,085,284, 7,187,904, 8,095,067, D704174, 8,238,318, 8,300,590, 8,311,023, 8,385,305, 8,422,540, 8,467,363, 8,502,733, 8,638,839, 8,649,418, 8,761,100, 8,811,365, 8,824,442, 8,830,943, 8,872,715, 8,897,340, 8,928,542, 8,942,216, 8,948,235, 8,982,772, 8,989,762, 9,001,809, 9,049,611, 9,055,463, 9,178,558, 9,179,240, 9,226,315, 9,226,295, 9,252,857, 9,282,560, 9,313,674, 9,325,398, 9,345,036, 9,350,411, 9,374,822, 9,408,215, 9,474,080, 9,490,918, 9,572,163, 9,577,700, 9,577,733, 9,578,643, 9,609,530, 9,655,133, 9,712,216, 9,713,019, 9,713,155, 9,713,157.

26. During the past several years, I have provided expert testimony, reports or declarations in the cases of *Agere v. Sony* (on behalf of plaintiff Agere), *Linex v. Belkin et al.* (on

behalf of defendant Cisco), *CSIRO v. Toshiba et al*. (multiple related cases on behalf of plaintiff CSIRO), *Freedom Wireless v. Cingular et al.* (on behalf of plaintiff Freedom Wireless), *Rembrandt v. HP et al*. (on behalf of defendant HP), *DNT v. Sprint et al.* (on behalf of the defendants Sprint, T-Mobile, US Cellular, Verizon and Novatel), *Teles v. Cisco* (on behalf of defendant Cisco), *WiAV v. HP* (on behalf of defendant HP), *SPH v. Acer et al*. (on behalf of defendants Sony, Nokia, Motorola, Novatel, Sierra and Dell), *LSI v. Funai* (on behalf of plaintiff LSI), *WiAV v. Dell and RIM* (on behalf of the defendants Dell and RIM), *Wi-LAN v. RIM* (on behalf of defendant RIM), *LSI v. Barnes & Noble* (on behalf of plaintiff LSI), *Novatel v. Franklin and ZTE* (on behalf of plaintiff Novatel), *LSI v. Realtek* (on behalf of plaintiff LSI), *Wi-LAN v. Apple et al.* (on behalf of defendants Apple, Sierra and Novatel), *EON v. Sensus et al*. (on behalf of defendants Motorola, US Cellular and Sprint), *M2M v Sierra et al*. (multiple related cases on behalf of defendants Sierra and Novatel), *Intellectual Ventures v. AT&T et al*. (on behalf of defendants AT&T, T-Mobile and Sprint), *Intellectual Ventures v. Motorola* (on behalf of defendant Motorola), *TQ Beta v. Dish et al*. (on behalf of defendant Dish), *Qurio v. Dish et al*. (on behalf of defendant Dish), *Fatpipe v. Talari* (on behalf of the defendant Talari), *EON v. Apple* (on behalf of defendant Apple), *Chrimar v. Dell* (on behalf of defendant Dell), *Nokia v. LGE* (on behalf of plaintiff Nokia), *PanOptis v. Blackberry* (on behalf of defendant Blackberry), *Customedia v. Dish et al.* (on behalf of defendant Dish), *Blackberry v. BLU* (on behalf of plaintiff Blackberry), *MTel v. Charter et al.* (on behalf of defendants Charter, Time Warner, Cox and Bright House), *Huawei v. Samsung* (on behalf of plaintiff Huawei) and *Alacritech v. Wistron* (on behalf of defendant Wistron). I believe that the preceding list includes all cases that I have testified in as an expert at trial or by deposition at least during the past four years.

III. PERSON OF ORDINARY SKILL IN THE ART

27. I understand that the content of a patent (including its claims) and prior art should be interpreted the way a person of ordinary skill in the art (or "POSITA") would have interpreted the material at the alleged time of invention.

28. I understand that the "alleged time of invention" here is no earlier than the date that the applicants for the '021, '718 and '061 Patents first filed an application related to the '021, '718 and '061 Patents in the United States Patent and Trademark Office, namely, Jan. 5, 1999, as further discussed herein.

29. It is my opinion that the person of ordinary skill in the art (or "POSITA") at the time of the filing date of the '021, '718 and '061 Patents would have had at least a Bachelor of Science in Computer Science, Computer Engineering, Electrical Engineering, or an equivalent field as well as at least 2 years of academic or industry experience in any type of network equipment field.

30. In addition to my testimony as an expert, I am prepared to testify as someone who actually practiced in the field from 1986 to present, who actually possessed at least the knowledge of a person of ordinary skill in the art in that time period, and who actually worked with others possessing at least the knowledge of a person of ordinary skill in the art in that time period.

31. I understand that the person of ordinary skill is a hypothetical person who is assumed to be aware of all the pertinent information that qualifies as prior art. In addition, the person of ordinary skill in the art makes inferences and takes creative steps.

IV. LEGAL UNDERSTANDING

32. I have a general understanding of validity based on my experience with patents and my discussions with counsel.

33. I have a general understanding of prior art and priority date based on my experience with patents and my discussions with counsel.

34. I understand that inventors are entitled to a priority date up to one year earlier than an actual date of filing of a patent application that provides written description support for a particular claim to the extent that they can show complete possession of such a particular claimed invention at such an earlier priority date and reasonable diligence to reduce such a particular claimed invention to practice between such an earlier priority date and such an actual date of filing. I understand that if the Patent Owner contends that particular claims are entitled to such an earlier priority date than such an actual date of filing, then the Patent Owner has the burden to prove this contention with specificity.

35. I understand that an invention by another must be made before the priority date of a particular patent claim in order to qualify as "prior art" under 35 U.S.C. § 102 or § 103, that a printed publication or a product usage must be publicly available before the priority date of a particular patent claim in order to qualify as "prior art" under 35 U.S.C. § 102(a), that a printed publication or a product usage or offer for sale must be publicly available more than one year prior to the actual date of filing of a patent application that provides written description support for a particular claim in the United States in order to qualify as "prior art" under 35 U.S.C. § 102(b), or that the invention by another must be described in an application for patent filed in the United States before the priority date of a particular patent claim in order to qualify as "prior art"

under 35 U.S.C. § 102(e). I understand that the Defendants have the burden of proving that any particular reference or product usage or offer for sale is prior art.

36. I have a general understanding of anticipation based on my experience with patents and my discussions with counsel.

37. I understand that anticipation analysis is a two-step process. The first step is to determine the meaning and scope of the asserted claims. Each claim must be viewed as a whole, and it is improper to ignore any element of the claim. For a claim to be anticipated under U.S. patent law: (1) each and every claim element must be identically disclosed, either explicitly or inherently, in a single prior art reference; (2) the claim elements disclosed in the single prior art reference must be arranged in the same way as in the claim; and (3) the identical invention must be disclosed in the single prior art reference, in as complete detail as set forth in the claim. Where even one element is not disclosed in a reference, the anticipation contention fails. Moreover, to serve as an anticipatory reference, the reference itself must be enabled, i.e., it must provide enough information so that a person of ordinary skill in the art can practice the subject matter of the reference without undue experimentation.

38. I further understand that where a prior art reference fails to explicitly disclose a claim element, the prior art reference inherently discloses the claim element only if the prior art reference must necessarily include the undisclosed claim element. Inherency may not be established by probabilities or possibilities. The fact that an element may result from a given set of circumstances is not sufficient to prove inherency. I have applied these principles in forming my opinions in this matter.

39. I have a general understanding of obviousness based on my experience with patents and my discussions with counsel.

40. I understand that a patent claim is invalid under 35 U.S.C. § 103 as being obvious only if the differences between the claimed invention and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person of ordinary skill in that art. An obviousness analysis requires consideration of four factors: (1) scope and content of the prior art relied upon to challenge patentability; (2) differences between the prior art and the claimed invention; (3) the level of ordinary skill in the art at the time of the invention; and (4) the objective evidence of non-obviousness, such as commercial success, unexpected results, the failure of others to achieve the results of the invention, a long-felt need which the invention fills, copying of the invention by competitors, praise for the invention, skepticism for the invention, or independent development.

41. I understand that a prior art reference is proper to use in an obviousness determination if the prior art reference is analogous art to the claimed invention. I understand that a prior art reference is analogous art if at least one of the following two considerations is met. First a prior art reference is analogous art if it is from the same field of endeavor as the claimed invention, even if the prior art reference addresses a different problem and/or arrives at a different solution. Second, a prior art reference is analogous art if the prior art reference is reasonably pertinent to the problem faced by the inventor, even if it is not in the same field of endeavor as the claimed invention.

42. I understand that it must be shown that one having ordinary skill in the art at the time of the invention would have had a reasonable expectation that a modification or combination of one or more prior art references would have succeeded. Furthermore, I understand that a claim may be obvious in view of a single prior art reference, without the need to combine references, if the elements of the claim that are not found in the reference can be

supplied by the knowledge or common sense of one of ordinary skill in the relevant art. However, I understand that it is inappropriate to resolve obviousness issues by a retrospective analysis or hindsight reconstruction of the prior art and that the use of "hindsight reconstruction" is improper in analyzing the obviousness of a patent claim.

43. I further understand that the law recognizes several specific guidelines that inform the obviousness analysis. First, I understand that a reconstructive hindsight approach to this analysis, i.e., the improper use of post-invention information to help perform the selection and combination, or the improper use of the listing of elements in a claim as a blueprint to identify selected portions of different prior art references in an attempt to show that the claim is obvious, is not permitted. Second, I understand that any prior art that specifically teaches away from the claimed subject matter, i.e., prior art that would lead a person of ordinary skill in the art to a specifically different solution than the claimed invention, points to non-obviousness, and conversely, that any prior art that contains any teaching, suggestion, or motivation to modify or combine such prior art reference(s) points to the obviousness of such a modification or combination. Third, while many combinations of the prior art might be "obvious to try", I understand that any obvious to try analysis will not render a patent invalid unless it is shown that the possible combinations are: (1) sufficiently small in number so as to be reasonable to conclude that the combination would have been selected; and (2) such that the combination would have been believed to be one that would produce predictable and well understood results. Fourth, I understand that if a claimed invention that arises from the modification or combination of one or more prior art references uses known methods or techniques that yield predictable results, then that factor also points to obviousness. Fifth, I understand that if a claimed invention that arises from the modification or combination of one or more prior art references is the result of known

work in one field prompting variations of it for use in the same field or a different one based on design incentives or other market forces that yields predicable variations, then that factor also points to obviousness. Sixth, I understand that if a claimed invention that arises from the modification or combination of one or more prior art references is the result of routine optimization, then that factor also points to obviousness. Seventh, I understand that if a claimed invention that arises from the modification or combination of one or more prior art references is the result of a substitution of one known prior art element for another known prior art element to yield predictable results, then that factor also points to obviousness.

44. I understand that a dependent claim incorporates each and every limitation of the claim from which it depends. Thus, my understanding is that if a prior art reference fails to anticipate an independent claim, then that prior art reference also necessarily fails to anticipate all dependent claims that depend from the independent claim. Similarly, my understanding is that if a prior art reference or combination of prior art references fails to render obvious an independent claim, then that prior art reference or combination of prior art references also necessarily fails to render obvious all dependent claims that depend from the independent claim.

V. THE ‘021, ‘718 AND ‘061 PATENTS

45. I note that the ‘021 Patent was filed on Mar. 13, 2000. I note that the ‘718 Patent was filed on Jun. 30, 2000 and claims priority to at least the ‘021 Patent and that the ‘061 Patent was filed on Jun. 30, 2000 and claims priority to at least the ‘021 Patent. Other than changes to the claims and the addition of text associated with filing a continuation patent, I am not aware at this time of any substantive changes to the specifications of the ‘061 and ‘718 Patents versus that of the ‘021 Patent. Thus, for purposes of my description herein, I will describe the ‘021, ‘718 and ‘061 Patent specification(s) together simultaneously with exemplary references only to the ‘021 Patent.

46. The ‘021, ‘718 and ‘061 Patents, respectively entitled “Navigating network-based electronic information using spoken input with multimodal error feedback” (‘021), “Mobile navigation of network-based electronic information using spoken input” (‘718), and “System, method, and article of manufacture for agent-based navigation in a speech-based data navigation system” (‘061) relate “generally to the navigation of electronic data by means of spoken natural language requests, and to feedback mechanisms and methods for resolving the errors and ambiguities that may be associated with such requests” (see, for example, Ex. 1001 at 1:15-19).

A. Overview of the ‘021, ‘718 and ‘061 Patents

47. In the “Background of the Invention” section, the ‘021, ‘718 and ‘061 Patents note that at the time of the invention that “As global electronic connectivity continues to grow, and the universe of electronic data potentially available to users continues to expand, there is a growing need for information navigation technology that allows relatively naïve users to navigate and access desired data by means of natural language input” (see, for example, Ex. 1001 at 1:20-25). The ‘021, ‘718 and ‘061 Patents further note that at the time of the invention that “existing navigational systems for browsing electronic databases and data warehouses (search

engines, menus, etc.), have been designed without navigation via spoken natural language as a specific goal" (see, for example, Ex. 1001 at 1:47-50)

48. Accordingly, the '021, '718 and '061 Patents claim that "What is needed is a methodology and apparatus for rapidly constructing a voice-driven front-end atop an existing, non-voice data navigation system, whereby users can interact by means of intuitive natural language input not strictly conforming to the step-by-step browsing architecture of the existing navigation system, and wherein any errors or ambiguities in user input are rapidly and conveniently resolved" (see, for example, Ex. 1001 at 2:12-19). However, the '021, '718 and '061 Patents also state that "a solution contemplating one-at-a-time user interactions at a single location is insufficient" (see, for example, Ex. 1001 at 2:22-24).

49. The '021, '718 and '061 Patents also describe in the "Summary of the Invention" section that "The present invention addresses the above needs by providing a system, method, and article of manufacture for navigating network-based electronic data sources in response to spoken input requests" such that "When a spoken input request is received from a user, it is interpreted, such as by using a speech recognition engine to extract speech data from acoustic voice signals, and using a language parser to linguistically parse the speech data" (see, for example, Ex. 1001 at 2:27-30). The '021, '718 and '061 Patents specifically distinguish the interpretation of such a "spoken request" between being performed "on a computing device locally with the user" versus that of being performed "remotely from the user" (see, for example, Ex. 1001 at 2:34-36).

50. In the "Detailed Description of the Invention" section for the '021, '718 and '061 Patents, "FIG. 1a" is described as "an illustration of a data navigation system driven by spoken

natural language input, in accordance with one embodiment of the present invention" (see, for example, Ex. 1001 at 3:44-46, FIG. 1a as reproduced herein).

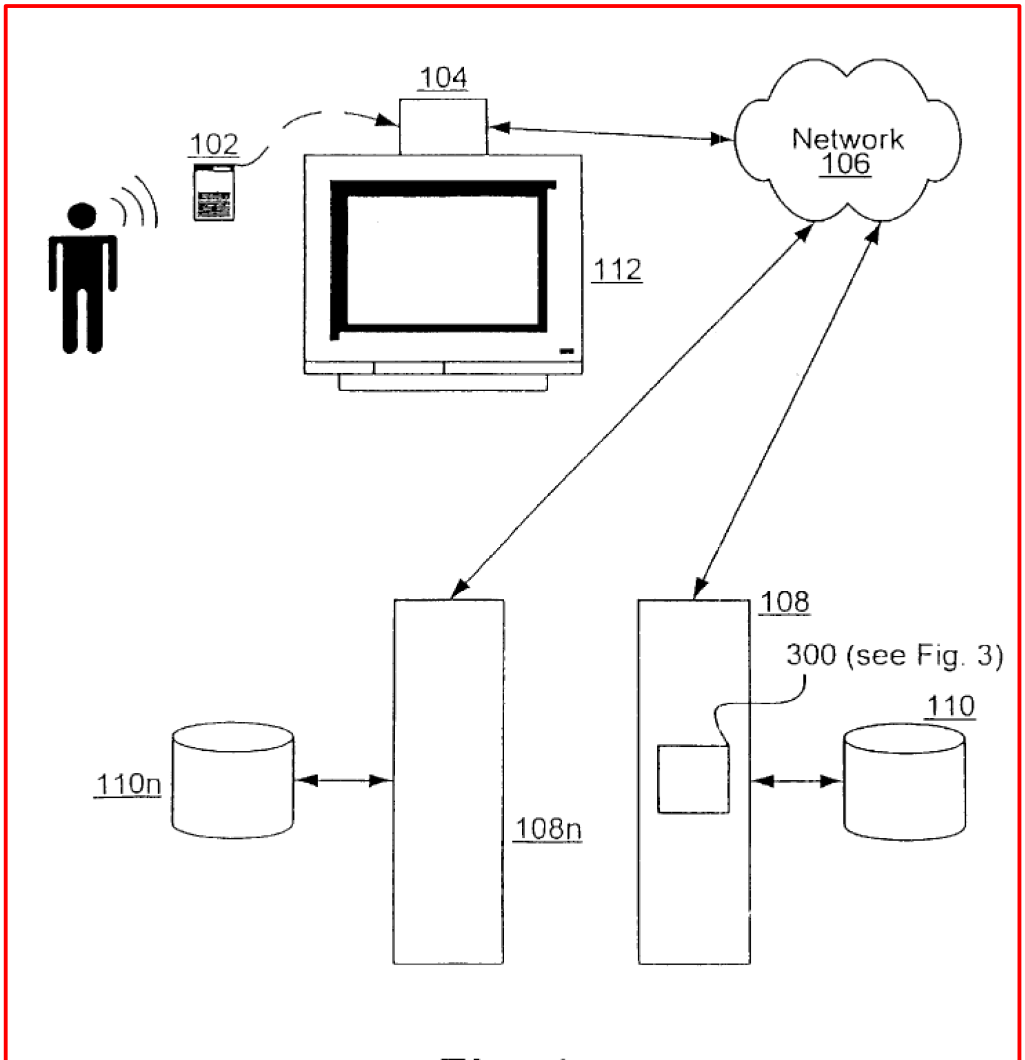


Fig. 1a

51. The '021, '718 and '061 Patents describe that "a user's voice input data is captured by a voice input device **102**", which in one example is "a portable remote control device with an integrated microphone" (see, for example, Ex. 1001 at 3:46-55). Additionally, the '021, '718 and '061 Patents describe that "the voice data is transmitted from device **102** preferably via infrared (or other wireless) link to communications box **104** (e.g., a set-top box or a similar communications device that is capable of retransmitting the raw voice data and/or

processing the voice data) local to the user's environment and coupled to communications network **106**" such that "The voice data is then transmitted across network **106** to a remote server or servers **108**" (see, for example, Ex. 1001 at 3:55-67). The '021, '718 and '061 Patents further describe that "At remote server **108**, the voice data is processed by request processing logic **300** in order to understand the user's request and construct an appropriate query or request for navigation of remote data source **110**" (see, for example, Ex. 1001 at 4:1-4).

52. The '021, '718 and '061 Patents describe this "request processing logic **300**" in reference to FIG. 3 as comprising "speech recognition engine **310**, natural language (NL) parser **320**, query construction logic **330**, and query refinement logic **340**" (see, for example, Ex. 1001 at 4:6-11, and FIG. 3 as reproduced below).

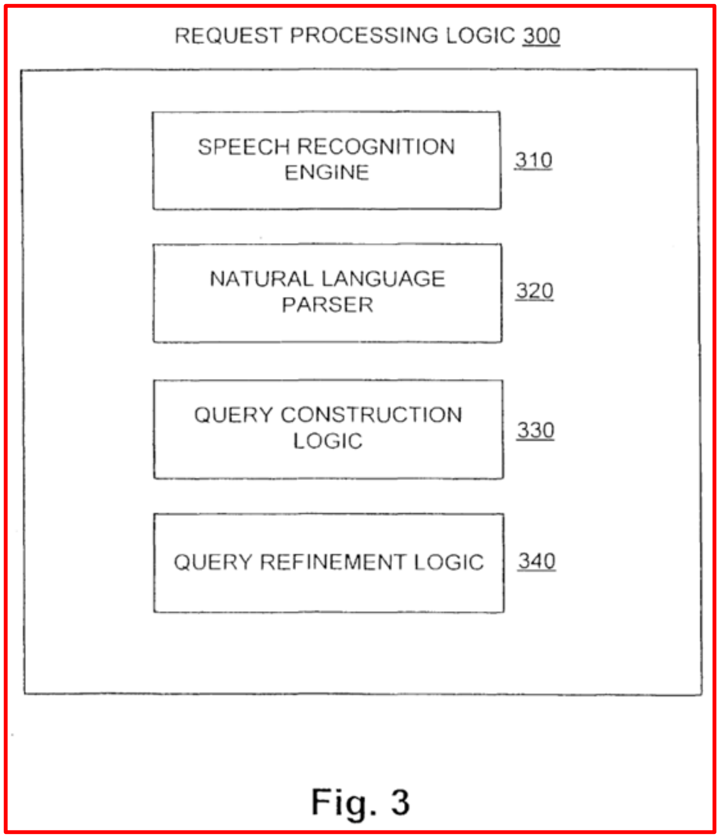
53. 

Fig. 3

54. Although FIG. 1a of the '021, '718 and '061 Patents depicts "request processing logic **300**" as being located at "a remote server or servers **108**", another embodiment of the '021,

‘718 and ‘061 Patents illustrated in FIG. 1b shows “request processing logic **300**” as being within a “local speech processor” that is “integrated as part of communications box **104**” or implemented in a “physically separate (but communicatively coupled) unit” as the ‘021, ‘718 and ‘061 Patents admit to be “readily apparent to those of skill in the art” (see, for example, Ex. 1001 at 4:58-66). Additionally, the ‘021, ‘718 and ‘061 Patents describe that “it is possible to divide and allocate the functional components of request processing logic **300** between client and server” such as “speech recognition—in entirety, or perhaps just early stages such as feature extraction—might be performed locally on the client end, perhaps to reduce bandwidth requirements, while natural language parsing and other necessary processing might be performed upstream on the server end, so that more extensive computational power need not be distributed locally to each client” (see, for example, Ex. 1001 at 6:39-49). However, the ‘021, ‘718 and ‘061 Patents do not provide enabling disclosure of a case where “early stages” of “request processing logic **300**” may be performed on a “remote server” followed by “natural language parsing and other necessary processing” being performed at a “client”, as would be opposite of the alleged benefits “to reduce bandwidth requirements” and enable “that more extensive computational power need not be distributed locally to each client”.

55. Also in reference to FIG. 1a, the ‘021, ‘718 and ‘061 Patents disclose that “Data source **110** may comprise database(s), Internet/web site(s), or other electronic information repositories, and preferably resides on a central server or servers” and “may include multimedia content, such as movies or other digital video and audio content, other various forms of entertainment data, or other electronic information” (see, for example, Ex. 1001 at 4:11-20). Additionally, the ‘021, ‘718 and ‘061 Patents describe that “Once the desired information has

been retrieved from data source **110**, it is electronically transmitted via network **106** to the user for viewing on client display device **112**" (see, for example, Ex. 1001 at 4:25-27).

56. According to the '021, '718 and '061 Patents, an example of "display device **112**" is "a television monitor or similar audiovisual entertainment device, typically in stationary position for comfortable viewing by users" that is "coupled to or integrated with a communications box (which is preferably the same as communications box **104**, but may also be a separate unit) for receiving and decoding/formatting the desired electronic information that is received across communications network **106**" (see, for example, Ex. 1001 at 4:27-37).

57. Also according to the '021, '718 and '061 Patents, "Network **106** is a two-way electronic communications network and may be embodied in electronic communication infrastructure including coaxial (cable television) lines, DSL, fiber-optic cable, traditional copper wire (twisted pair), or any other type of hardwired connection" and "may be part of the Internet and may support TCP/IP communications, or may be embodied in a proprietary network, or in any other electronic communications network infrastructure, whether packet-switched or connection-oriented" (see, for example, Ex. 1001 at 4:38-49).

58. The '021, '718 and '061 Patents describe that "as depicted in FIG. 2, a mobile variation in accordance with the server-side processing architecture illustrated in FIG. 1a may be implemented by replacing voice input device **102**, communications box **104**, and client display device **112**, with an integrated, mobile, information appliance **202** such as a cellular telephone" that "essentially performs the functions of the replaced components" (see, for example, Ex. 1001 at 5:56-64, FIG. 2 as reproduced below). The '021, '718 and '061 Patents admit that the "Data source", as a "network accessible information resource" can be "constructed to support access requests from simultaneous multiple network users, as known by practitioners of ordinary skill in

the art" (see, for example, Ex. 1001 at 6:20-28). Additionally, the '021, '718 and '061 Patents admit that "In the case of server-side speech processing", the "interpretation logic and error correction logic modules are also preferably designed and implemented to support queuing and multi-tasking of requests from multiple simultaneous network users, as will be appreciated by those of skill in the art" (see, for example, Ex. 1001 at 6:28-34).

59.

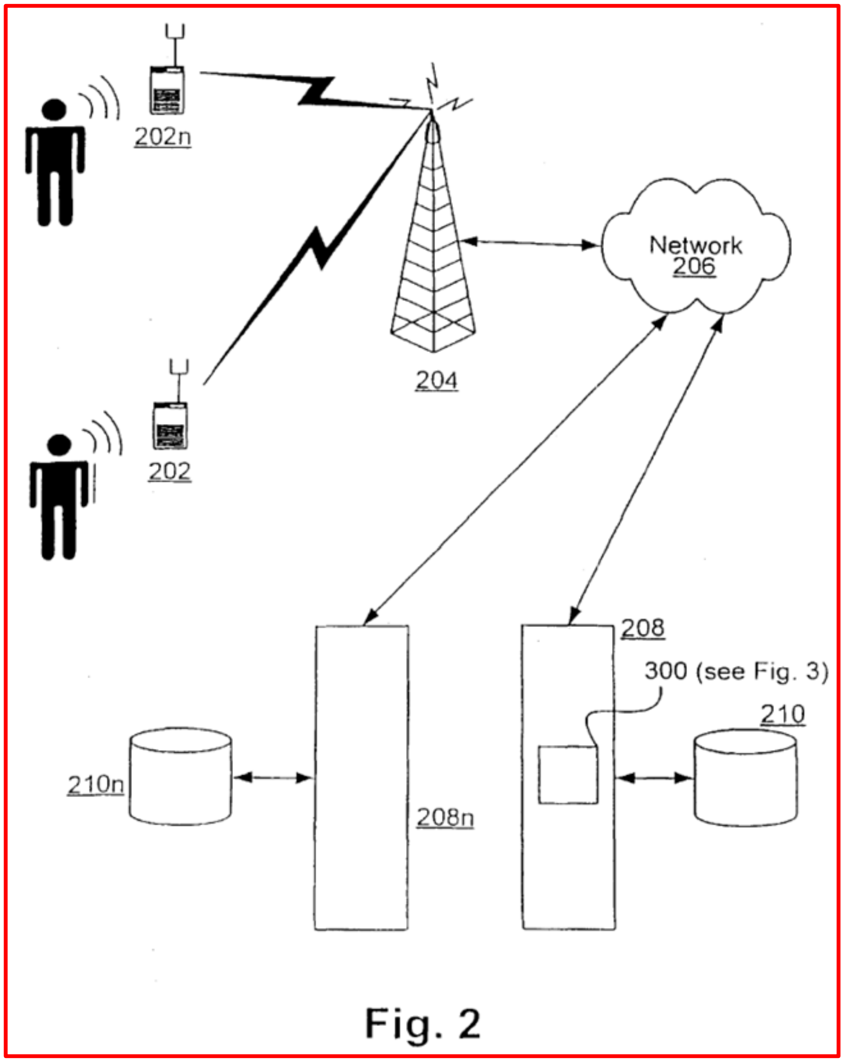


Fig. 2

60. The '021, '718 and '061 Patents describe FIG. 4 as "a process utilizing spoken natural language for navigating an electronic database" that begins "At step **402**, the user's spoken request for information is initially received in the form of raw (acoustic) voice data by a suitable input device, as previously discussed in connection with FIGS. 1-2" and then "At step

404 the voice data received from the user is interpreted in order to understand the user's request for information" (see, for example, Ex. 1001 at 3:29-30, 7:9-14, FIG. 4 as reproduced below).

61.

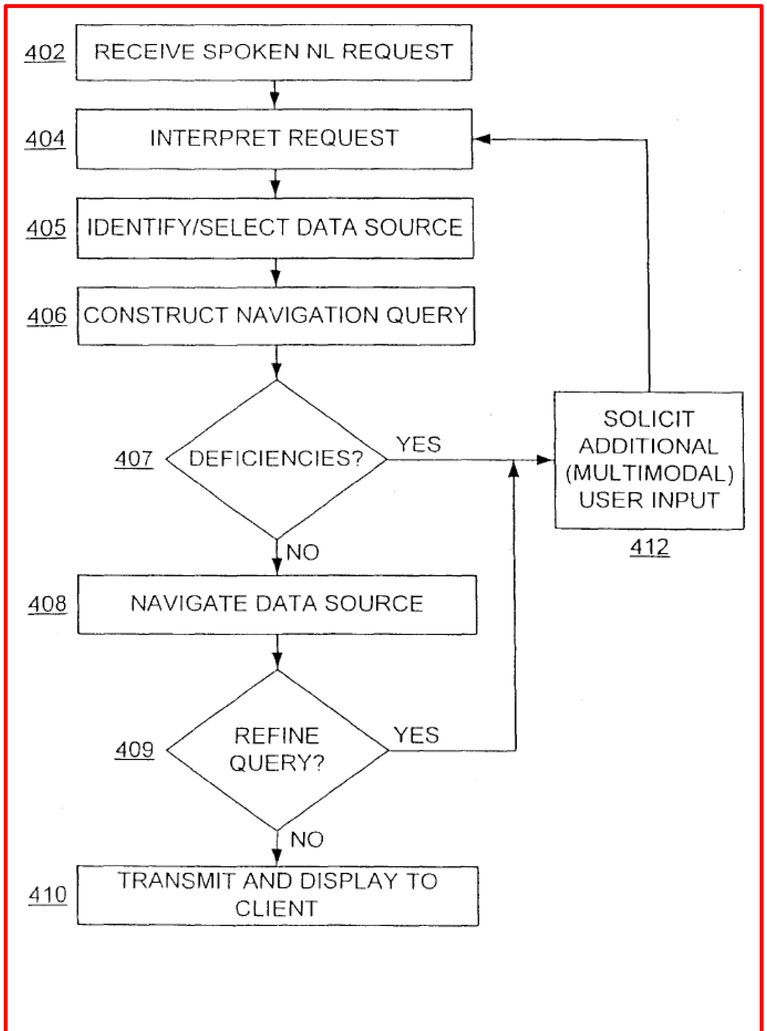

402
RECEIVE SPOKEN NL REQUEST
404
INTERPRET REQUEST
405
IDENTIFY/SELECT DATA SOURCE
406
CONSTRUCT NAVIGATION QUERY
407
DEFICIENCIES?
YES
NO
SOLICIT ADDITIONAL (MULTIMODAL) USER INPUT
412
408
NAVIGATE DATA SOURCE
409
REFINE QUERY?
YES
NO
410
TRANSMIT AND DISPLAY TO CLIENT

Fig. 4

62. According to the '021, '718 and '061 Patents, "a speech recognition engine processes acoustic voice data and attempts to generate a text stream of recognized words" wherein "A variety of commercial quality, speech recognition engines are readily available on the market, as practitioners will know" such as "Nuance Communications offers a suite of speech recognition engines, including Nuance 6, its current flagship product, and Nuance Express" and such as "IBM offers the ViaVoice speech recognition engine, including a low-cost

shrink-wrapped version available through popular consumer distribution channels" (see, for example, Ex. 1001 at 7:20-31).

63. Additionally, with respect to FIG. 4, the ‘021, ‘718 and ‘061 Patents describe "In step **405** request processing logic **300** identifies and selects an appropriate online data source where the desired information (in this case, current weather reports for a given city) can be found" from "a locally stored table, or possibly dynamic searching through an online search engine, or other online search techniques" (see, for example, Ex. 1001 at 8:41-46). The ‘021, ‘718 and ‘061 Patents also note that "For some applications, an embodiment of the present invention may be implemented in which only access to a particular data source (such as a particular vendor's proprietary content database) is supported; in that case, step **405** may be trivial or may be eliminated entirely" (see, for example, Ex. 1001 at 8:47-51).

64. The ‘021, ‘718 and ‘061 Patents describe that "Step **406** attempts to construct a navigation query, reflecting the interpretation of step **404**" wherein "This operation is preferably performed by query construction logic **330**" (see, for example, Ex. 1001 at 8:52-54). The ‘021, ‘718 and ‘061 Patents define "navigation query" as "an electronic query, form, series of menu selections, or the like; being structured appropriately so as to navigate a particular data source of interest in search of desired information" such that "it includes whatever content and structure is required in order to access desired information electronically from a particular database or data source of interest" (see, for example, Ex. 1001 at 8:55-62).

65. Moreover, the ‘021, ‘718 and ‘061 Patents admit that "Practitioners of ordinary skill in the art will be thoroughly familiar with the notion of database navigation through structured query, and will be readily able to appreciate and utilize the existing data structures and navigational mechanisms for a given database, or to create such structures and mechanisms

where desired" (see, for example, Ex. 1001 at 9:9-14). Additionally, the '021, '718 and '061 Patents require that "In accordance with the present invention, the query constructed in step **406** must reflect the user's request as interpreted by the speech recognition engine and the NL parser in step **404**" (see, for example, Ex. 1001 at 9:15-18).

66. According to the '021, '718 and '061 Patents, "Several problems can arise when attempting to perform searches based on spoken natural language input" and thus "As indicated at decision step **407** in the process of FIG. 4, certain deficiencies may be identified during the process of query construction, before search of the data source is even attempted" (see, for example, Ex. 1001 at 10:40-45). For example, the '021, '718 and '061 Patents describe that "the user's request may fail to specify enough information in order to construct a navigation query that is specific enough to obtain a satisfactory search result" or that "certain deficiencies and problems may arise following the navigational search of the data source at step **408**, as indicated at decision step **409** in FIG. 4" (see, for example, Ex. 1001 at 10:45-54).

67. According to the '021, '718 and '061 Patents, "In the event that one or more deficiencies in the user's spoken request, as processed, result in the problems described, either at step **407** or **409**, some form of error handling is in order" such as "soliciting additional input from the user in a manner taking advantage of the partial construction already performed and via user interface modalities in addition to spoken natural language ("multi-modality")" that is "preferably conducted through client display device **112** (**202**, in the embodiment of FIG. 2), and may include textual, graphical, audio and/or video media" (see, for example, Ex. 1001 at 10:64-11:13).

68. The '021, '718 and '061 Patents describe that "Query refinement logic **340** preferably carries out step **412**" such that "The additional input received from the user is fed into

and augments interpreting step **404**, and query construction step **406** is likewise repeated with the benefit of the augmented interpretation" and "These operations, and subsequent navigation step **408**, are preferably repeated until no remaining problems or deficiencies are identified at decision points **407** or **409**" (see, for example, Ex. 1001 at 11:14-21). The '021, '718 and '061 Patents admit that a "menu interface" for such a "query refinement process" is known in the "prior art" (see, for example, Ex. 1001 at 11:21-28).

69. The '021, '718 and '061 Patents also state that "Open Agent Architecture™ (OAA®) is a software platform, developed by the assignee of the present invention, that enables effective, dynamic collaboration among communities of distributed electronic agents", wherein "an agent registers with its parent facilitator a specification of the capabilities and services it can provide, using a highlevel, declarative Interagent Communication Language ("ICL") to express those capabilities" and that "OAA can provide an advantageous platform for constructing embodiments of the present invention" as in reference to FIG. 6 (see, for example, Ex. 1001 at 13:5-8, 13:25-29, 14:15-18, and FIG. 6 reproduced below).

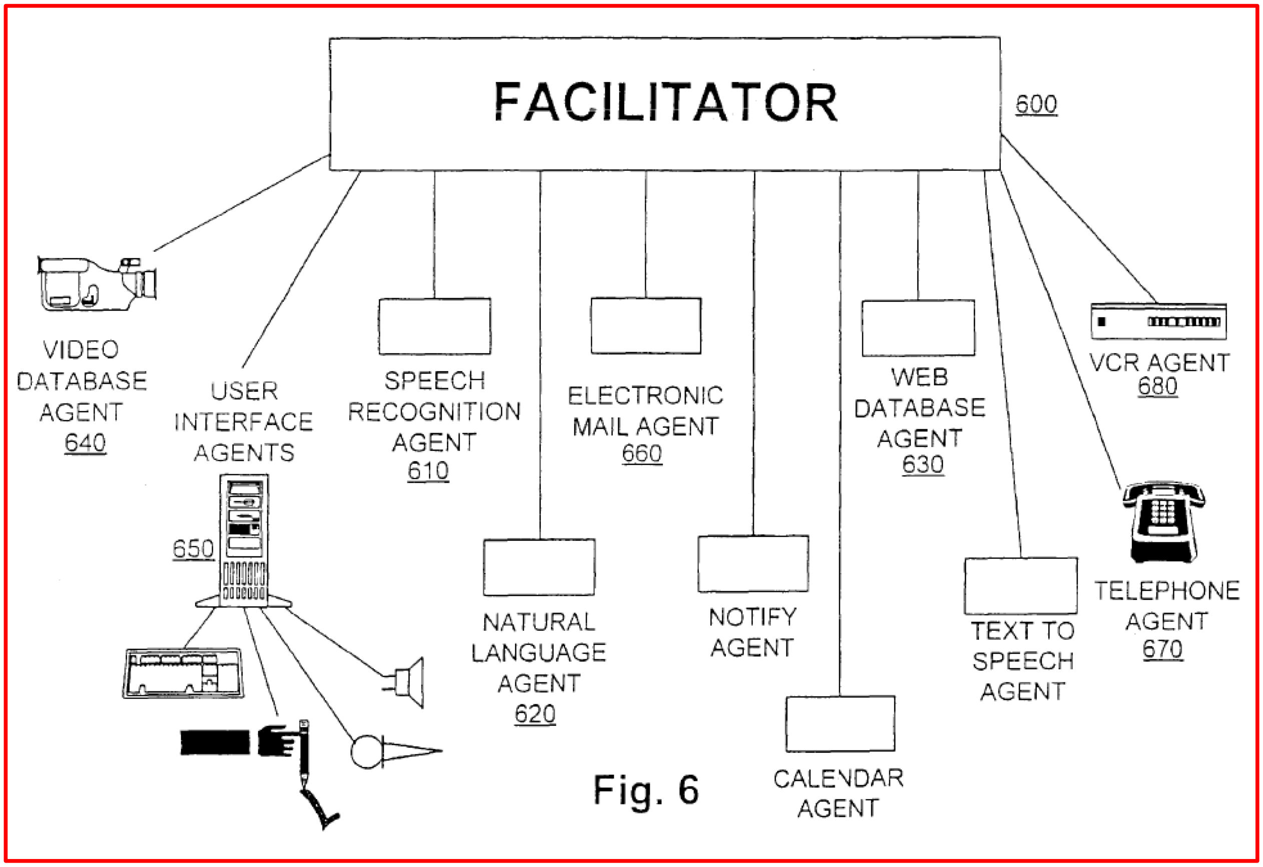


Fig. 6

70. According to the ‘021, ‘718 and ‘061 Patents, “If the statement “show me movies starring John Wayne” is spoken into the voice input device, the voice data for this request will be sent by UI agent **650** to facilitator **600**, which in turn will ask natural language (NL) agent **620** and speech recognition agent **610** to interpret the query and return the interpretation in ICL format” so that “The resulting ICL goal expression is then routed by the facilitator to appropriate agents—in this case, video-on-demand database agent **640**—to execute the request” (see, for example, Ex. 1001 at 14:18-27). Additionally, the ‘021, ‘718 and ‘061 Patents disclose that “Video database agent **640** preferably includes or is coupled to an appropriate embodiment of query construction logic **330** and query refinement logic **340**, and may also issue ICL requests to facilitator **600** for additional assistance—e.g., display of menus and capture of additional user input in the event that query refinement is needed—and facilitator **600** will delegate such requests to appropriate client agents in the community” so that “When the desired video content

is ultimately retrieved by video database agent **640**, UI agent **650** is invoked by facilitator **600** to display the movie" (see, for example, Ex. 1001 at 14:27-38).

B. Asserted Claims and Priority Date

71. The '021 Patent includes 132 claims. I understand that Claims 1-2, 5-13, 15-16, 18, 20-22, 24-28, 31-40, 42-47, 50-59, 61-63, 65-66, 68, 70-73, 76-85, 87-91, 94-103, 105-110, 113-122, 124-128 and 130-131 are asserted in the District Court litigation and are the subject of the *Inter Partes* Review petition(s).

72. The '718 Patent includes 27 claims. I understand that Claims 1-4, 6, 8-13, 15, 17-22, 24 and 26-27 are asserted in the District Court litigation and are the subject of the *Inter Partes* Review petition(s).

73. The '061 Patent includes 18 claims. I understand that Claims 1-5, 7-11 and 13-17 are asserted in the District Court litigation and are the subject of the *Inter Partes* Review petition(s).

74. I understand that in the District Court litigation that the Patent Owner alleges the priority date of the '021, '718 and '061 Patents to be Jan. 5, 1999 (see, for example, Ex. 1001 at 1:6-13).

C. Objective Indicia of Non-obviousness

75. I understand that in the District Court litigation, Patent Owner has not yet provided any information regarding this topic. As of this writing, I am unaware of any information that would provide objective indicia of non-obviousness for any of the asserted claims of the '021, '718 and '061 Patents. However, to the extent that Patent Owner (or its expert) provides opinions and/or analysis with respect to this topic, I reserve the right to supplement my opinions and analyses on this topic.

VI. CLAIM CONSTRUCTION

76. I understand that claim construction is a matter of law. However, I understand that in a review proceeding the claims are to be given their broadest reasonable interpretation consistent with the '021, '718 and '061 Patent specifications and file histories, and that claim terms are to be given their ordinary and customary meaning, as would be understood by a person of ordinary skill in the art in the context of the entire disclosure and intrinsic record. I also understand that limitations from the specification are not to be read into the claims. The specification, however, can inform a person of ordinary skill in the art as to the broadest reasonable interpretation of the claims. In addition, I understand that a person of ordinary skill in the art would look to explanations and arguments made by the applicants during their prosecution file histories to inform as to the broadest reasonable interpretation of the claims of the '021, '718 and '061 Patents. I further understand that the broadest reasonable interpretation of the claims as appropriate for a review proceeding may be different from that of the construction of such claims (or terms therein) as appropriate in a District Court litigation. I understand that after expiration of a patent, that the terms of the claims are subject to their plain and ordinary meaning as set forth in *Phillips v. AWH*. I understand that the '021, '718, and '061 Patents will expire during this IPR proceeding and I have therefore applied the *Phillips* standard.

77. I understand that indefiniteness, written description, and enablement are not issues that can be addressed as part of an *Inter Partes* Review proceeding. Therefore, solely for the purposes of my prior art invalidity analyses herein as relevant to this *Inter Partes* Review proceeding, I have used the proper interpretation as appropriate for an *Inter Partes* Review proceeding even for such claims that I may otherwise believe to be indefinite, lacking written description and/or non-enabled.

78. The term "**electronic data source [being] located at one or more network servers located remotely from a user**" appears in the preambles of Claims 1, 27, 46, 72, 90, 109, 127 and 130 of the '021 Patent and in Claims 1, 10 and 19 of the '718 Patent. I believe that under the proper interpretation, these preambles would be considered as limiting for at least the following reasons.

79. First, the '021 and '718 Patents recognize that there was a clear design decision between performing tasks locally and performing them on a remote server. In particular, the '021 and '718 Patents state that "The interpretation of the spoken request can be performed on a computing device locally with the user or remotely from the user" (see, for example, Ex. 1001 at 2:34-36). The '021 and '718 Patents further state that "The resulting interpretation of the request is thereupon used to automatically construct an operational navigation query to retrieve the desired information from one or more electronic network data sources, which is then transmitted to a client device of the user" (see, for example, Ex. 1001 at 2:37-41). The '021 and '718 Patents therefore teach that the electronic network data sources are remote from the user, thereby requiring transmission to reach the client device of the user.

80. Second, a POSITA would recognize that a remotely located electronic data source presents very different technical challenges from a local data source such as the need for networking or other techniques for transmitting and receiving the data between the local computer and the network data source. For example, a local computer typically has a much higher bandwidth connection to a local data source, compared with that of a remote data source. Thus, the preamble limitation that the "electronic data source [being] located at one or more network servers located remotely from a user" is necessary in my opinion to define the complete invention.

81. Third, I further note that the "electronic data source" is referenced in the body of claims at issue with respect to these preamble terms. For example, claim 1 of the '021 Patent requires "using the refined navigation query to select a portion of the electronic data source" and "transmitting the selected portion of the electronic data source from the network server to a client device of the user" (see, for example, Ex. 1001 at 15:29-33). The recited "network servers" are also the same "network servers" from the preamble, at least because they were preceded by the definite article in the body of the claim. Furthermore, the recitation of the "transmitting" step itself confirms the limitation that the electronic data sources be remote from the local device. The claims do not discuss transmission within the local device and therefore that there is a "transmitting" step further confirms that the electronic data source must be remote from the local device, as stated in the preamble.

82. The term "**electronic data source**" also appears in the preambles of Claims 1, 7 and 13 of the '061 Patent, and provides an antecedent basis for reference to such term in other claim elements. For purposes of my analyses herein, I have assumed that at least this term "**electronic data source**" may be considered as a limitation under the proper interpretation for analogous reasons to that listed in ¶ 81 above.

83. The term "**navigation query**" appears in Claims 1, 5, 6, 7, 8, 10, 11, 12, 13, 18, 21, 27, 34, 36, 38, 39, 46, 50, 51, 52, 53, 55, 57, 58, 68, 71, 72, 76, 77, 78, 79, 81, 82, 83, 84, 90, 97, 99, 101, 102, 109, 113, 114, 115, 116, 118, 120, 121, 127, and 130 of the '021 Patent, Claims 1, 4, 10, 13, 19, and 22 of the '718 Patent, and Claims 1, 4, 5, 7, 10, 11, 13, and 16 of the '061 Patent. The '021, '718 and '061 Patents each explicitly define "navigation query" as "an electronic query, form, series of menu selections, or the like; being structured appropriately so as to navigate a particular data source of interest in search of desired information" (see, for

example, Ex. 1001 at 8:55-58). Moreover, the ‘021, ‘718 and ‘061 Patents explain that “In other words, a navigation query is constructed such that it includes whatever content and structure is required in order to access desired information electronically from a particular database or data source of interest” (see, for example, Ex. 1001 at 8:58-62). Thus, for the purpose of my analysis herein, I consider the proper interpretation of the claim term “**navigation query**” to be “**an electronic query, form, series of menu selections, or the like; being structured appropriately so as to navigate a particular data source of interest in search of desired information**”.

84. The term “**mobile information appliance**” appears in Claims 1-3, 8, 10, 12, 17, 19 and 21 of the ‘718 Patent. The “mobile information appliance” is described by the ‘718 Patent as “integrated” from and “essentially performing the functions” of “replaced components” that include “voice input device **102**, communications box **104**, and client display device **112**” with examples “such as a cellular telephone or wireless personal digital assistant (wireless PDA)” (see, for example, Ex. 1003 at 5:66-6:7). Further, the “mobile information appliance” is configured to “receive[] spoken natural language input requests from the user in the form of voice data, and transmit[] that data (preferably via wireless data receiving station **204**) across communications network **206** for server-side interpretation of the request,” and then display the results on the “display of information appliance **202**,” and output any retrieved audio “through the appliance's speakers” (see, for example, Ex. 1003 at 6:8-19). During prosecution of the ‘718 Patent, the applicants argued that the prior art Levin reference failed to teach or suggest a “mobile information appliance” by stating that “the *very essence of a mobile appliance is its portability, small size*, and ease of use” and “As such, *unlike hard-wired appliances*, mobile appliances are not equipped with large bulky input devices” (emphasis added, see, for example,

Ex. 1004 at p. 101). In my opinion, a POSITA at the time of the alleged invention would have understood that an appliance (or "device") that was "*unlike hard-wired*" must be "*battery-powered*" such as the case for the "cellular telephone" and "wireless personal digital appliance" examples noted in the specification above and also in the prosecution file history (see, for example, Ex. 1003 at 6:4-6 and Ex. 1004 at p. 102). This ability to use the device away from a hard-wired power outlet led to the "very essence of a mobile appliance" that the applicants touted when contrasting such "mobile appliances" with those of "hard-wired appliances like a desktop computer" (see, for example, Ex. 1004 at p. 101). Thus, for the purpose of my analysis herein, I consider the proper interpretation of the claim term "**mobile information appliance**" to be "**a battery-powered and portable integrated information processing device**".

85. The term "**facilitator**" appears in Claims 1, 7 and 13 of the '061 Patent. The '061 Patent discloses that "functionality of each client agent is made available to the agent community through registration of the client agent's capabilities with a facilitator" and "When a facilitator determines that the registered capabilities of one of its client agents will help satisfy a current goal or sub-goal thereof, the facilitator delegates that sub-goal to the client agent" (see, for example, Ex. 1005 at 13:22-24, 41-44). Furthermore, the '061 Patent describes the "facilitator" as the entity that "coordinates and integrates the results received from different client agents on various sub-goals, in order to satisfy the overall goal" (see, for example, Ex. 1005 at 13:49-51). The parent application describes the "facilitator" as "a specialized server agent that is responsible for matching requests, from users and agents, with descriptions of the capabilities of other agents" (see, for example, Ex. 1007 at 6:32-37). Thus, for the purpose of my analysis herein, I consider the proper interpretation of the claim term "**facilitator**" to be "**a specialized server entity**".

86. I have applied a plain and ordinary meaning to all remaining claim terms for the purposes of this *Inter Partes* Review proceeding.

87. In the event that one or more of these constructions is changed, or in the event that additional terms not specifically construed herein receive a proposed construction, I reserve the right to revisit my analysis under such additional construction(s).

VII. STATE OF THE ART

88. As of Mar. 17, 1999, when the applications that became the '021, '718 and '061 Patents were filed, the state of the art in the field of navigation of electronic data by means of spoken natural language requests with feedback mechanisms and methods for resolving the errors and ambiguities that may be associated with such requests already fully encompassed the elements of the asserted claims of the '021, '718 and '061 Patents, as evidenced in even the small sample of the art described herein.

A. Kupiec (Ex. 1013)

89. For example, amongst the numerous prior art references in this field, U.S. Patent No. 5,500,920 by Julian M. Kupiec entitled "Semantic co-occurrence filtering for speech recognition and signal transcription applications" ("Kupiec") was filed on Sep. 30, 1994, and issued on Mar. 19, 1996, which is more than 1 year before the earliest priority date of the '021, '718 and '061 Patents (see, for example, Ex. 1013 at (22), (45)). Thus, I understand that Kupiec qualifies as prior art to the '021, '718 and '061 Patents at least under 35 U.S.C. §§ 102(a), (b), and (e).

90. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer" including "transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems and methods" and "more particularly to speech-recognition systems and methods appropriate for use in conjunction with computerized information-retrieval systems" (see, for example, Ex. 1013 at 1:36-45). Thus, Kupiec is clearly from the same field of art as the '021, '718 and '061 Patents

and is clearly addressing similar problems as those purportedly addressed by the ‘021, ‘718 and ‘061 Patents.

91. Kupiec notes that “The general problem of disambiguating the words contained in an error-prone transcription of user input arises in a number of contexts beyond speech recognition, including but not limited to handwriting recognition in pen-based computers and personal digital assistants” and that “Transcription of user input from a form convenient to the user into a form convenient for use by the computer has any number of applications, including but not limited to word processing programs, document analysis programs, and, as already stated, information retrieval programs” but for “speech recognition”, Kupiec observes that “Unfortunately, computerized transcription tends to be error-prone” (see, for example, Ex. 1013 at 1:56-67). Such “personal digital assistants” were known to a POSITA at the time of the alleged invention to be normally “battery-powered”. Kupiec discloses that the “present invention” can be “used in systems that accommodate natural-language utterances, Boolean/proximity queries, special commands, or any combination of these” (see, for example, Ex. 1013 at 4:32-35).

92. Kupiec provides a “Glossary” that lists “general meanings” for a variety of terms used in Kupiec (see, for example, Ex. 1013 at 4:63-5:38). For example, Kupiec discloses “**Corpus**” as meaning “A body of natural language text to be searched” (see, for example, Ex. 1013 at 4:66-67). For example, Kupiec discloses “**Hypothesis**” as meaning “A guess at the correct interpretation of the words of a user's question” (see, for example, Ex. 1013 at 5:7-8). For example, Kupiec discloses “**Information retrieval, IR**” as meaning “The accessing and retrieval of stored information, typically from a computer database” (see, for example, Ex. 1013 at 5:11-12). For example, Kupiec discloses “**Phonetic transcription**” as meaning “The process

of transcribing a spoken word or utterance into a sequence of constituent phones" wherein "**Phone**" means "A member of a collection of symbols that are used to describe the sounds uttered when a person pronounces a word" (see, for example, Ex. 1013 at 5:22-27). For example, Kupiec discloses "**Query**" as meaning "An expression that is used by an information retrieval system to search a corpus and return text that matches the expression" (see, for example, Ex. 1013 at 5:28-30). For example, Kupiec discloses "**Question**" as meaning "A user's information need, presented to the invention as input" (see, for example, Ex. 1013 at 5:31-32). For example, Kupiec discloses "**Utterance**" as meaning "Synonym for question in embodiments of the invention that accept spoken input" (see, for example, Ex. 1013 at 5:34-35). For example, Kupiec discloses "**Word index**" as meaning "A data structure that associates words found in a corpus with all the different places such words exist in the corpus" (see, for example, Ex. 1013 at 5:36-38).

93. Kupiec states that "FIG. 1 illustrates a system **1** that embodies the present invention" (see, for example, Ex. 1013 at 5:42-43, FIG. 1 reproduced below).

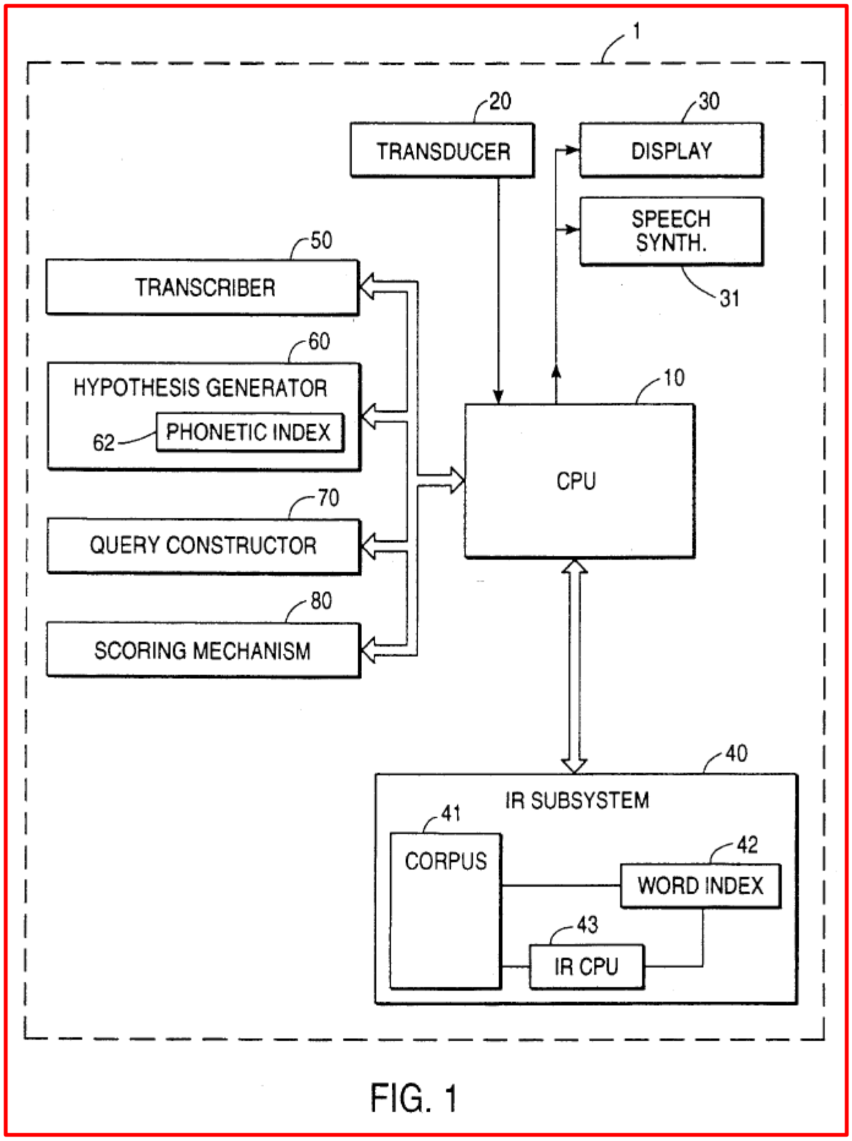
94. 

FIG. 1

95. According to Kupiec, "System **1** comprises a processor **10** coupled to an input audio transducer **20**, an output visual display **30**, an optional output speech synthesizer **31**, and an information retrieval (IR) subsystem **40** which accesses documents from corpus **41** using a word index **42**" as well as "a phonetic transcriber **50**, a hypothesis generator **60**, a phonetic index **62**, a query constructor **70**, and a scoring mechanism **80**" (see, for example, Ex. 1013 at 5:43-51).

96. Kupiec describes that "Processor **10** is a computer processing unit (CPU)" that typically is "part of a mainframe, workstation, or personal computer" but can comprise "multiple processing elements in some embodiments" (see, for example, Ex. 1013 at 5:52-55).

97. Kupiec also describes that “Transducer **20** converts a user's spoken utterance into a signal that can be processed by processor **10**” and can comprise “a microphone coupled to an analog-to-digital converter, so that the user's speech is converted by transducer **20** into a digital signal” and can further comprise “signal-conditioning equipment including components such as a preamplifier, a pre-emphasis filter, a noise reduction unit, a device for analyzing speech spectra (e.g., by Fast Fourier Transform), or other audio signal processing devices in some embodiments” (see, for example, Ex. 1013 at 5:56-6:7).

98. Kupiec discloses that “Display **30** provides visual output to the user”, which may be of the form of “alphanumeric display of the texts or titles”, such as for “documents retrieved from corpus **41**” and typically comprises “a computer screen or monitor” (see, for example, Ex. 1013 at 6:12-15). Kupiec also discloses that “Speech synthesizer **31** optionally can be included in system **1** to provide audio output, for example, to read aloud portions of retrieved documents to the user” (see, for example, Ex. 1013 at 6:16-18).

99. Kupiec further discloses that “IR subsystem **40** incorporates a processor that can process queries to search for documents in corpus **41**” and can “use processor **10** or, as shown in FIG. 1, can have its own processor **43**” (see, for example, Ex. 1013 at 6:22-25).

100. Kupiec specifically discloses that “IR subsystem **40** can be located at the same site as processor **10** or can be located at a remote site and connected to processor **10** via a suitable communication network” (see, for example, Ex. 1013 at 6:25-28).

101. Kupiec also discloses that “Corpus **41** comprises a database of documents that can be searched by IR subsystem **40**” wherein such documents comprise “for example, books, articles from newspapers and periodicals, encyclopedia articles, abstracts, office documents, etc.” (see, for example, Ex. 1013 at 6:29-33).

102. Kupiec further discloses that "Transcriber **50**, hypothesis generator **60**, phonetic index **62**, query constructor **70**, and scoring mechanism **80** are typically implemented as software modules executed by processor **10**" wherein the "function of these modules is described more fully below for specific embodiments, in particular with reference to the embodiments of FIGS. 2 and 8" (see, for example, Ex. 1013 at 6:44-50).

103. Kupiec provides a detailed example of how "IR subsystem **40** can perform certain IR query operations" when such "IR queries are formulated in a query language that expresses Boolean, proximity, and ordering or sequence relationships between search terms in a form understandable by IR subsystem **40**" as shown (see, for example, Ex. 1013 at 6:53-7:48).

104. Kupiec describes embodiments that employ "discrete-word speech input" where "the user is expected to pause between each spoken word, so that the system can readily determine where one spoken word ends and the next begins" and other embodiments that employ "continuous speech input and attempts to determine the word boundary positions for itself" (see, for example, Ex. 1013 at 7:50-58).

105. Kupiec discloses a "first specific embodiment" with reference to FIG. 2 of Kupiec (see, for example, Ex. 1013 at 9:17-18, FIG. 2 reproduced below).

106.

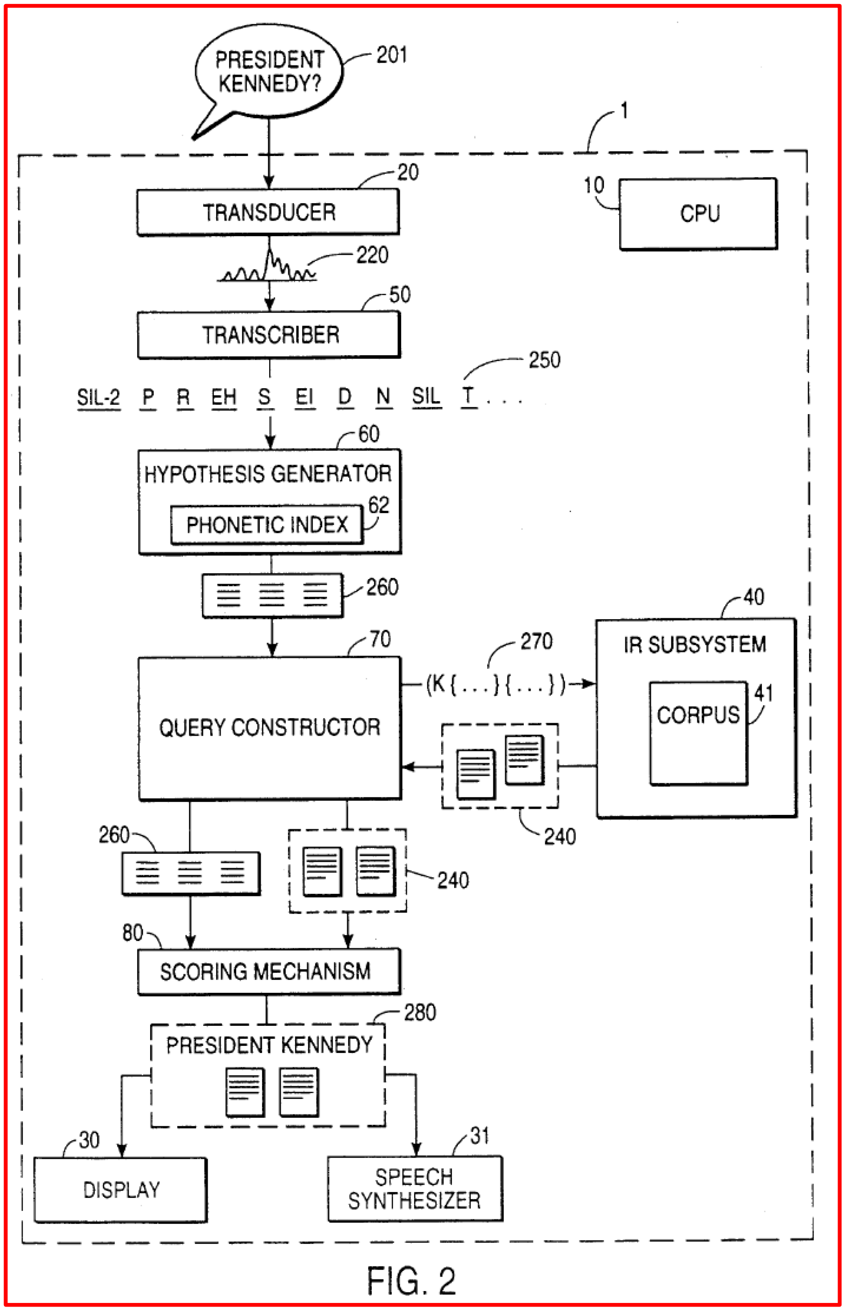


FIG. 2

107. With respect to FIG. 2, Kupiec discloses that "The user inputs a question **201** into system **1** by speaking into audio transducer **20**" and the "signal **220** produced by transducer **20** is fed to transcriber **50**, where it is converted into a phonetic transcription **250**" using "any of a variety of transcription techniques" that were "well-known among those of skill in the art" (see, for example, Ex. 1013 at 9:18-23). Kupiec explains that "phonetic transcription **250** is an

ordered sequence of phones, that is, of component sounds that can be used to form words" (see, for example, Ex. 1013 at 9:29-31). Kupiec also notes that "transcriber **50** is error-prone and produces a phonetic transcription **250** that is imperfect" (see, for example, Ex. 1013 at 9:35-37).

108. Kupiec discloses in reference to FIG. 2 that "The phonetic transcription **250** is provided to hypothesis generator **60** where it is matched using phonetic index **62** to generate a set of hypotheses **260**" and that "Because the transcriber **50** is known to be error-prone, hypothesis generator **60** develops alternative possible transcriptions for each word spoken by the user, in addition to the original phone sequences provided by transcriber **50**" such that "hypothesis generator **60** can attempt to correct mistakes commonly made by transcriber **50** by adding, deleting, or substituting one or more phones into the sequence of phones that represents the word as originally transcribed" (see, for example, Ex. 1013 at 9:38-61).

109. Kupiec further notes that "Occasionally, no candidates will be found for one or more words of the user's utterance" as "can happen, for example, if part of the utterance is garbled" wherein "hypothesis generator **60** can omit the unrecognized word from the generated hypotheses" or "Alternatively, hypothesis generator **60** can halt processing of the user's question and prompt the user to repeat the question" as "can be adopted, for example, if none of the user's words is recognized" (see, for example, Ex. 1013 at 11:1-9).

110. In continued reference to the embodiment of FIG. 2, Kupiec describes that "Once the set of hypotheses **260** has been generated, it is provided to query constructor **70**" such that "Query constructor **70** uses the hypotheses **260** to construct one or more queries **270** that will be sent to IR subsystem **40** for execution" (see, for example, Ex. 1013 at 11:10-13). Kupiec explains that "Queries **270** are Boolean queries with proximity and order constraints" and provides a specific example wherein the "user speaks two words" that lead to a "set of

hypotheses" such "an initial query" is constructed that "seeks occurrences of at least one of the words (search terms)" and is "sent to the IR subsystem **40** where it is executed" (see, for example, Ex. 1013 at 11:13-41).

111. Kupiec further describes that "Depending on the results obtained from execution of the initial query, additional queries can be constructed and executed, in a process called query reformulation" as in "For example, if no matches are found for the initial query, query constructor **70** can increase the proximity value" and "send the query thus modified back to IR subsystem **40** to be executed again" (see, for example, Ex. 1013 at 11:42-48). Alternatively, Kupiec describes that "query constructor **70** can drop one or more words from the query" to "be helpful, for example, if one of the user's intended words is not present in phonetic index **62**, so that none of the candidates for this word is correct" (see, for example, Ex. 1013 at 11:48-52). Additionally, Kupiec discloses that "Query reformulation is described in further detail with reference to FIG. 6 below" but "In general, a series of queries **270** is constructed by query constructor **70** and provided to IR subsystem **40**, which executes them by conducting searches in accordance with queries **270** over corpus **41**" such that "The execution of the initial and any additional queries causes a set of documents **240** to be retrieved from corpus **41**" (see, for example, Ex. 1013 at 11:53-60).

112. According to Kupiec, these "retrieved documents **240** and the query matches that they contain are provided along with hypotheses **260** to scoring mechanism **80**" that "assigns scores to the various hypotheses **260** according to probable relevance to the user's input question **201**" in order "to determine which hypothesis or hypotheses best match the user's intended utterance" (see, for example, Ex. 1013 at 12:1-9). Thus, "When scoring is finished, the results **280** can be presented to the user using processor **10** in conjunction with visual display **30**"

including “Excerpts of the documents showing the occurrence of the matched search terms” (see, for example, Ex. 1013 at 12:34-42).

113. Kupiec also notes that “The flowchart of FIG. 3 summarizes the method or processing steps performed by the system of FIG. 2” (see, for example, Ex. 1013 at 12:47-48, FIG. 3 reproduced below).

114.

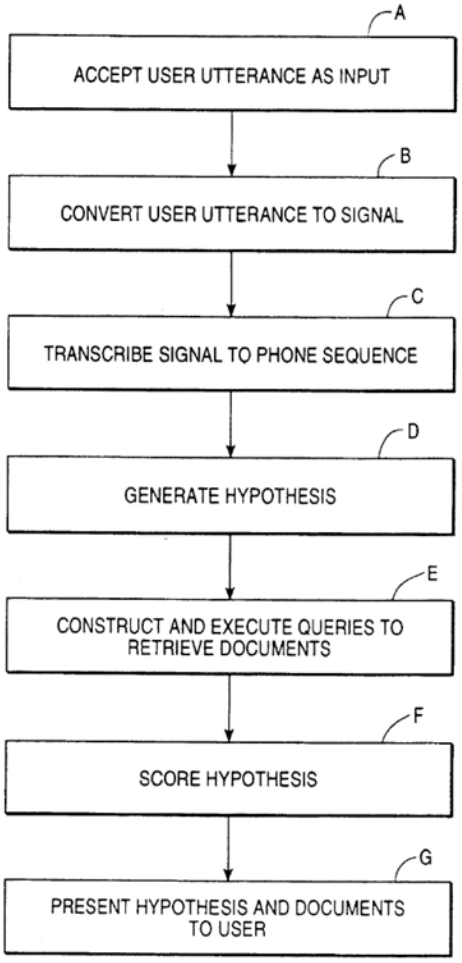

A
ACCEPT USER UTTERANCE AS INPUT
B
CONVERT USER UTTERANCE TO SIGNAL
C
TRANSCRIBE SIGNAL TO PHONE SEQUENCE
D
GENERATE HYPOTHESIS
E
CONSTRUCT AND EXECUTE QUERIES TO RETRIEVE DOCUMENTS
F
SCORE HYPOTHESIS
G
PRESENT HYPOTHESIS AND DOCUMENTS TO USER

FIG. 3

115. Kupiec succinctly summarizes FIG. 3 as “First the system accepts a user utterance as input (Step A). This utterance is converted to a signal (Step B) that is transcribed into a sequence of phones (Step C). The phone sequence is used to generate hypotheses (Step D). Boolean queries with proximity and order constraints are constructed based on the hypotheses

and are executed to retrieve documents (Step E). Hypotheses are scored in order of relevance (Step F). A relevant subset of the hypotheses and retrieved documents is presented to the user (Step G)" (see, for example, Ex. 1013 at 12:48-57).

116. As noted above, Kupiec's "FIG. 6 is an expansion of a portion of Step E of the flowchart of FIG. 3" whereby "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query" (see, for example, Ex. 1013 at 15:1-12, FIG. 6 reproduced below).

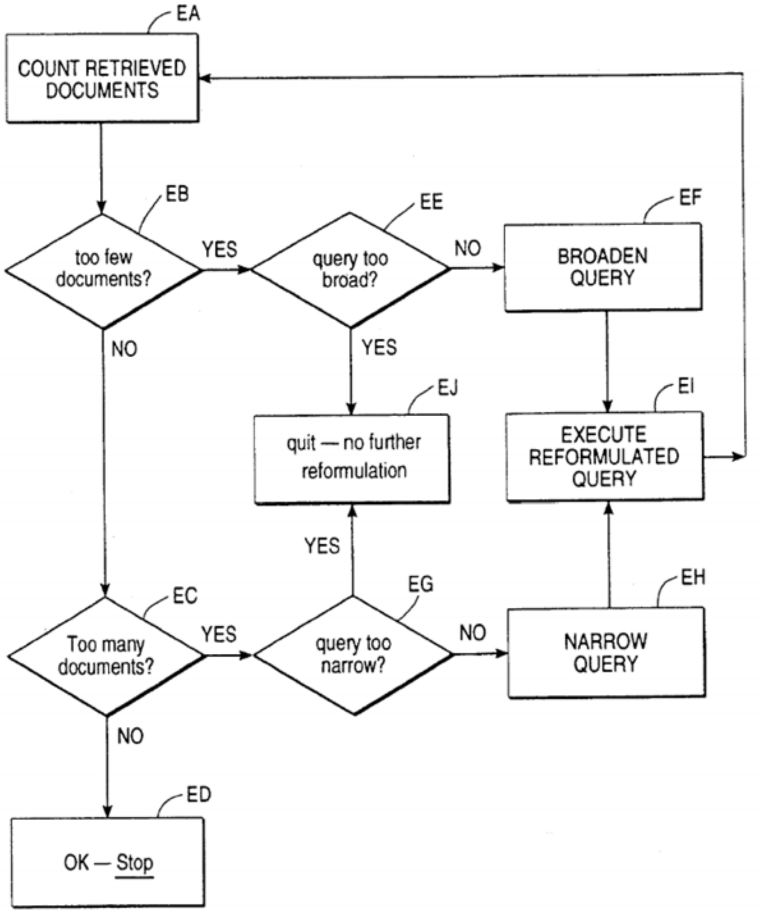


FIG. 6

117.

118. Kupiec discloses in reference to FIG. 6 that "query constructor 70 performs a preliminary analysis of the returned documents (Step EA)" followed by testing "against a predefined minimum value such as 15 (Step EB) and a predefined maximum value such as 50 (Step EC)" such that "If the number of documents is reasonable, that is, greater than or equal to the minimum value and less than the maximum value, then no additional queries are deemed necessary (Step ED)" (see, for example, Ex. 1013 at 15:15-24). Alternatively, Kupiec discloses that "To broaden a query is to modify it in such a way that the number of documents likely to be retrieved upon its execution increases" by "First, a check is made to see whether the query can helpfully be broadened further (Step EE)" to "ensure that queries are not broadened indefinitely, and to prevent an infinite loop of broadening and narrowing operations" such that "If the check succeeds, then the query is broadened (Step EF)" (see, for example, Ex. 1013 at 15:26-33). And similarly, Kupiec discloses that "If there are too many retrieved documents, then an attempt is made to narrow the query" by "First, a check is made to see whether the query can helpfully be narrowed further (Step EG)" to "ensure that queries are not narrowed indefinitely, and to prevent an infinite loop of broadening and narrowing operations" such that "If this check succeeds, then the query is narrowed (Step EH)" (see, for example, Ex. 1013 at 16:13-21).

119. Also in reference to FIG. 6, Kupiec describes that "Once the query has been broadened or narrowed, the query thus modified is sent to IR subsystem **40** where it is executed (Step EI)" such that the process returns to "Step EA" where "the count of returned documents is once again checked to see whether too few (Step EB) or too many (Step EC) documents have been returned" and then "If the number of documents is now reasonable, query reformulation stops (Step ED); otherwise, further broadening or narrowing occurs (Steps EE through EI)" and this "loop of broadening or narrowing proceeds until either a reasonable number of documents is

found (Step ED) or no further broadening or narrowing is deemed helpful (Step EJ)" (see, for example, Ex. 1013 at 16:37-49).

120. Finally, in reference to FIG. 6, Kupiec describes that "If query reformulation terminates successfully (in Step ED) with a reasonable number of documents retrieved, these documents are passed on to scoring mechanism **80** for scoring" but "If query reformulation terminates unsuccessfully (in Step EJ) with too few or too many documents retrieved, either such documents as have been retrieved can be passed on for scoring, or, alternatively, no documents can be passed on and an error message can be displayed to the user on visual display **30**" (see, for example, Ex. 1013 at 16:50-58). For the case where "query reformulation terminates successfully", Kupiec's "scoring mechanism" causes the "documents" to be "ranked in order from highest to lowest score" and "presented to the user on output" (see, for example, Ex. 1013 at 18:11-21).

121. Kupiec discloses a "second specific embodiment" with reference to FIG. 8 of Kupiec that is "similar in many respects to that shown in FIG. 2 for the first specific embodiment" but "This second embodiment incorporates user relevance feedback and supports keywords" (see, for example, Ex. 1013 at 18:31-42).

122. For example, Kupiec discloses the use of "Command words" that "The user can supply keywords that signify special IR operations to be carried out during query construction" such as "the Boolean keywords "and," "or," and "not," and also keywords to indicate that terms should be queried in strict order or within a certain proximity of one another" (see, for example, Ex. 1013 at 19:6-11). Kupiec specifically describes an example usage of such "commands" being "used to select among competing hypotheses" such as "if the invention determines that the two best interpretations of the user's utterance, as indicated by the highest-ranked hypotheses, are

"president kennedy" and "present-day canada," the user can indicate that "president kennedy" is the better choice" (see, for example, Ex. 1013 at 19:26-31).

123. For example, Kupiec discloses the use of "Relevance feedback commands" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, the user has the option of directing the invention to perform a follow-up search based on the retrieved results" (see, for example, Ex. 1013 at 19:32-36). Thus, Kupiec notes that "the best matching documents that correspond at any time to the words that the user has spoken so far can be displayed to the user on a screen" such that "Upon seeing the titles (or other descriptive content) the user can speak additional words to direct the search to particular documents or cause them to be excluded by invoking the NOT operation" (see, for example, Ex. 1013 at 19:64-20:2).

124. Kupiec describes that "The flowchart of FIG. 9 summarizes the processing steps performed according to the method of the second embodiment" (see, for example, Ex. 1013 at 20:3-5, FIG. 9 reproduced below).

125.

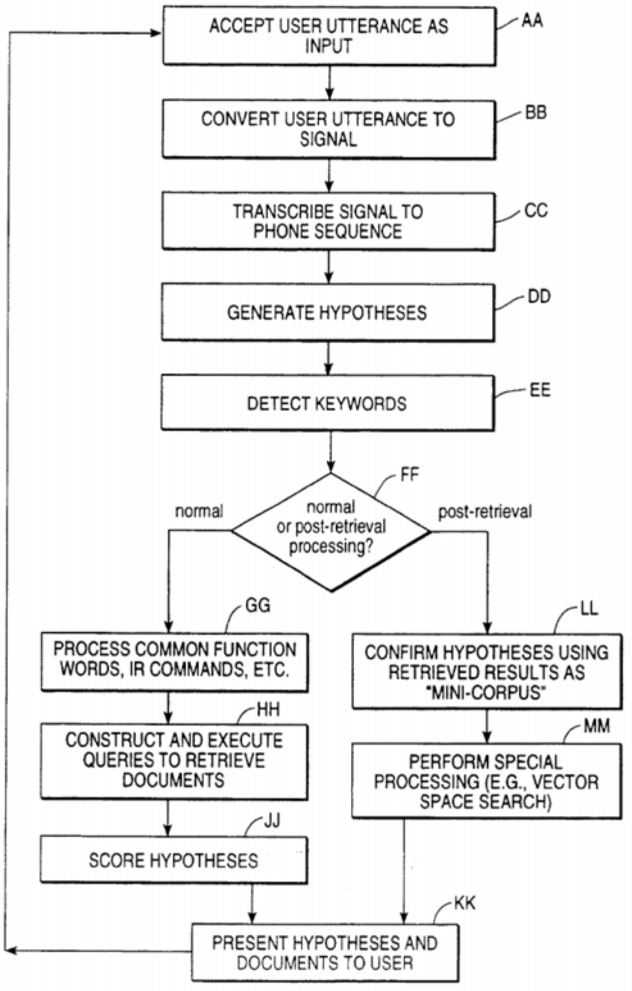

ACCEPT USER UTTERANCE AS INPUT
AA
CONVERT USER UTTERANCE TO SIGNAL
BB
TRANSCRIBE SIGNAL TO PHONE SEQUENCE
CC
GENERATE HYPOTHESES
DD
DETECT KEYWORDS
EE
FF
normal
normal or post-retrieval processing?
post-retrieval
GG
PROCESS COMMON FUNCTION WORDS, IR COMMANDS, ETC.
LL
CONFIRM HYPOTHESES USING RETRIEVED RESULTS AS "MINI-CORPUS"
HH
CONSTRUCT AND EXECUTE QUERIES TO RETRIEVE DOCUMENTS
MM
PERFORM SPECIAL PROCESSING (E.G., VECTOR SPACE SEARCH)
JJ
SCORE HYPOTHESES
KK
PRESENT HYPOTHESES AND DOCUMENTS TO USER

FIG. 9

126. In particular, Kupiec discloses in reference to FIG. 9 that “A test is made to determine whether documents have previously been retrieved since the last "new search" or similar command (step FF). If no documents have been retrieved, then a search for documents is made. Prior to query construction, keywords are processed (Step GG); in particular, common function words can be filtered out of the hypotheses, and IR command words are routed to the query constructor, where they can be interpreted and incorporated into queries. Thereafter, Boolean queries with proximity and order constraints are constructed based on the hypotheses and the keywords, and these queries are executed to retrieve documents (Step HH). A relevant

subset of the hypotheses and retrieved documents is presented to the user (Step KK). If documents have previously been retrieved, then user relevance feedback commands and search terms can be routed to the hypothesis generator, to instruct the hypothesis generator to use retrieved document titles as the basis for confirming hypotheses (Step LL), or to cease doing this upon a "new search" or similar command. The system then can perform operations such as a vector space search or the selection of one among several preferred hypotheses (Step MM). Results of these operations are presented to the user (Step KK)" (see, for example, Ex. 1013 at 20:21-32).

127. Kupiec also discloses that "the invention accepts continuous speech" because this "frees the user to speak more naturally" (see, for example, Ex. 1013 at 21:62-64). In addition, Kupiec notes that "Section 7 concerns an embodiment that is not limited to IR tasks" and "Section 8 concerns an embodiment in which the input can take forms besides speech" (see, for example, Ex. 1013 at 20:42-44). For example, in Kupiec's "Section 7" as illustrated by FIG. 10, the "general-purpose speech recognizer" serves "as a "front end" speech-to-text converter for an application program **120** that accepts text input, such as a word processor" and thus "When used in a non-IR context, the documents retrieved by the method of the invention are considered intermediate results that need not be displayed to the user" wherein the "output is fed to application program **120**" (see, for example, Ex. 1013 at 22:22-40).

128. Kupiec's "Section 8" is described in reference to FIG. 11 that "illustrates a specific embodiment of the invention that is adaptable to a range of input sources, transcription techniques, hypothesis generation techniques, information retrieval techniques, and analysis techniques" and "comprises a processor running appropriate software and coupled to a text

corpus, an input transducer, and an output facility such as an output channel, stream, or device" (see, for example, Ex. 1013 at 23:19-25, FIG. 11 reproduced below).

129.

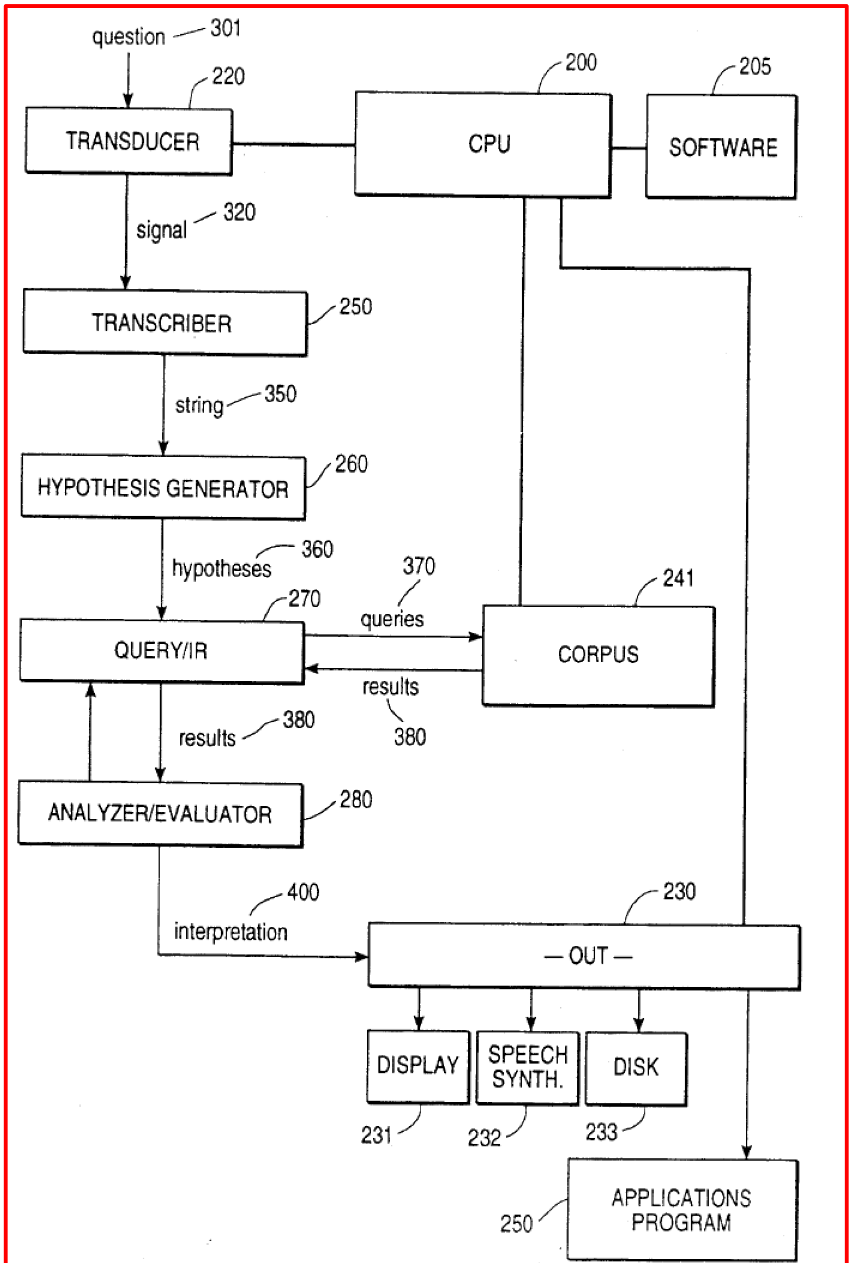


FIG. 11

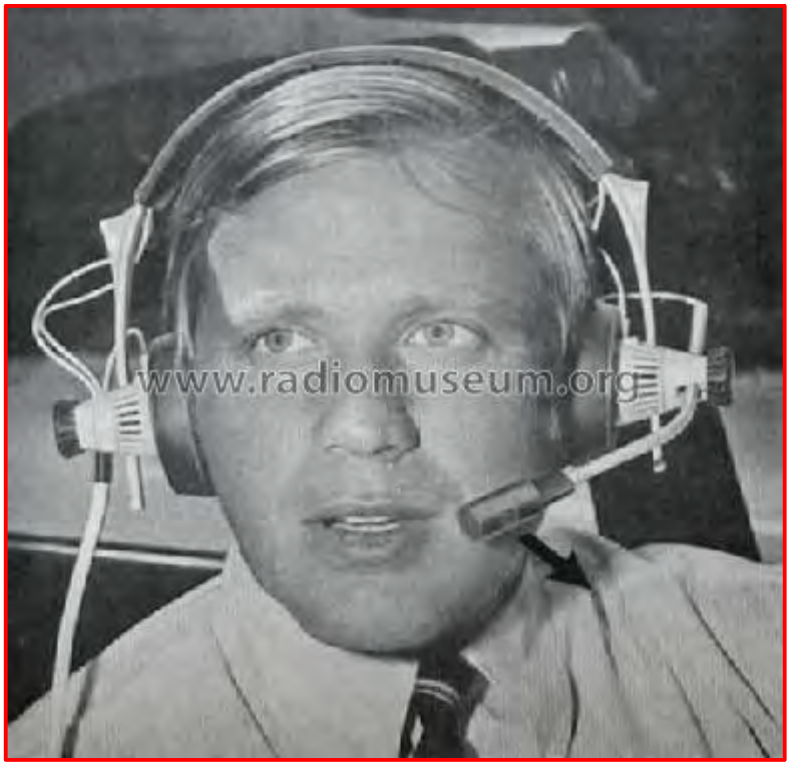
130. Kupiec discloses in reference to FIG. 11 that "processor **200** executes software **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator

280 are typically implemented as software modules that are part of software **205** and are executed by processor **200**" (see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "source code" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, for query construction, and for scoring" and the "second file includes source code for hypothesis generation" (see, for example, Ex. 1013 at 29:39-54).

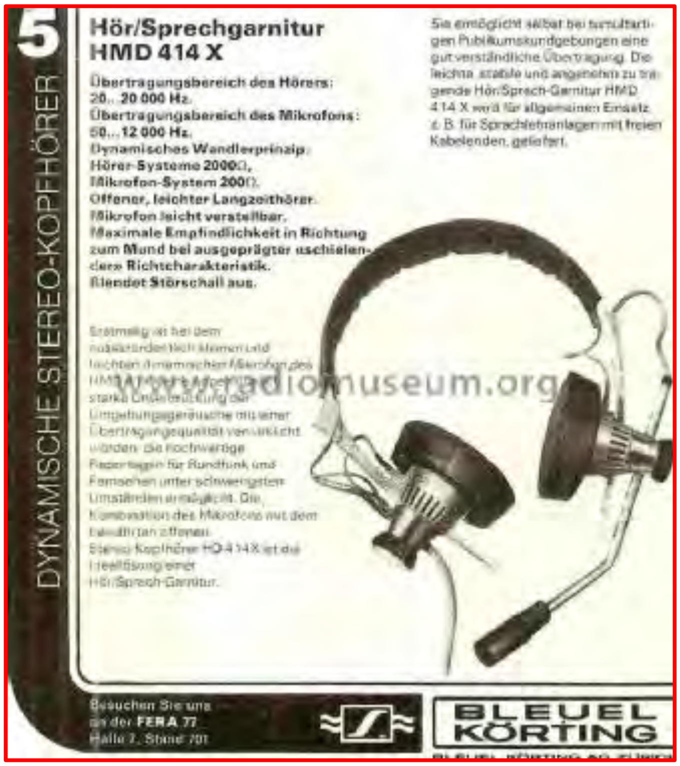
131. Kupiec further discloses that the system "has been demonstrated on a Sun SparcStation 10 workstation" and that "Discrete-word speech can be input using a Sennheiser HMD414 headset microphone and a Rane MS-1 preamplifier, with signal processing performed in software by the SparcStation" so that such "Input speech is transcribed into a phone sequence using hidden Markov model methods" as exemplified in a prior art 1989 IEEE paper (see, for example, Ex. 1013 at 29:45-46, 30:48-56).

132. Kupiec specifically references the well-known HMD 414 portable headset microphone combination from Sennheiser (see, for example, Ex. 1013 at 29:45-46), which was a popular product years before the alleged priority date of the '021, '718 and '061 Patents, but Kupiec does not provide a visual depiction of this commercially-available product. I personally recall the Sennheiser portable headset microphone combinations of this time frame, and I provide these image from the Internet that provides exemplary views of the HMD 414 device (from https://www.radiomuseum.org/r/sennheiser_hmd_414.html):

133.



134.


5
DYNAMISCHE STEREO-KOPFHÖRER
www.radiomuseum.org

Hör/Sprechgarnitur HMD 414 X

Übertragungsbereich des Hörers: 20...20 000 Hz.
Übertragungsbereich des Mikrofons: 50...12 000 Hz.
Dynamisches Wandlerprinzip: Hörer-Systeme 2000Ω, Mikrofon-System 200Ω.
Offener, leichter Langzeithörer.
Mikrofon leicht verstellbar.
Maximale Empfindlichkeit in Richtung zum Mund bei ausgeprägter »schielender« Richtcharakteristik.
Blendet Störschall aus.

Erstmalig ist bei dem außerordentlich kleinen und leichten dynamischen Mikrofon des HMD 414 X eine [illegible] starke Unterdrückung der Umgebungsgeräusche mit einer Übertragungsqualität verwirklicht worden, die hochwertige Reportagen für Rundfunk und Fernsehen unter schwierigsten Umständen ermöglicht. Die Kombination des Mikrofons mit dem bewährten offenen Stereo-Kopfhörer HD 414 X ist die Ideallösung einer Hör/Sprech-Garnitur.

Sie ermöglicht selbst bei tumultartigen Publikumskundgebungen eine gut verständliche Übertragung. Die leichte, stabile und angenehm zu tragende Hör/Sprech-Garnitur HMD 414 X wird für allgemeinen Einsatz z. B. für Sprachlehranlagen mit freien Kabelenden, geliefert.

Besuchen Sie uns
an der **FERA 77**
Halle 7, Stand 701

BLEUEL
KÖRTING

BLEUEL KÖRTING AG ZÜRICH

135.

136. Kupiec also discloses in reference to FIG. 11 that "In operation, transducer **220** accepts an input question **301** and converts it into a signal **320**" wherein "input question **301** can be a spoken utterance, in which case transducer **220** comprises audio signal processing equipment that converts the spoken utterance to signal **320**" as well as other input modalities such as "handwritten" or "typewritten" wherein "transducer **220** comprises a digitizing tablet or input-sensitive display screen as is typical of pen-based computers" or "transducer **220** comprises a conventional computer keyboard" (see, for example, Ex. 1013 at 24:22-41).

137. Kupiec further describes that "Transducer **220** provides signal **320** to transcriber **250**" such that "Transcriber **250** converts signal **320** to a string **350** that represents a transcription of the input question **301**" but Kupiec specifically notes that "transcriber **250** is error-prone, and string **350** does not necessarily represent a correct transcription of what the user intended to say in question **301**" (see, for example, Ex. 1013 at 24:66-25:4). Additional elements and functionality associated with FIG. 11 of Kupiec for the "hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280"** are described in relation to "corpus **241**" analogously to the similarly numbered elements of FIG. 1 of Kupiec (see, for example, ¶¶ 95-112 above and Ex. 1013 at 25:13-27:6).

138. For example, Kupiec describes that "Transducer **220** provides signal **320** to transcriber **250**" that "converts signal **320** to a string **350** that represents a transcription of the input question **301**" such that "Hypothesis generator **260** converts string **350** and any alternatives to a set of hypotheses **360**" provided to "Query/IR mechanism **270**" that "converts the hypotheses **360** to one or more information retrieval queries **370**" that are "in a format that can be searched by processor **200** (or a separate IR processor that communicates with processor **200**) using corpus **241**" (see, for example, Ex. 1013 at 24:66-25:61).

139. With respect to FIG. 11, Kupiec additionally discloses that "Analyzer/evaluator **280** provides the hypothesis or hypotheses most likely to correctly interpret question **301** and, if appropriate, query results relevant to this hypothesis or hypotheses, as an interpretation **400** that is output via output channel **230**" wherein such "hypotheses can be represented, for example, as ASCII text", thereby enabling that "Output channel **230** can send interpretation **400** to be displayed using a visual display **231**" such that "If the appropriate command keywords are supported, the user can provide relevance feedback based on displayed or speech-synthesized output" in order "to facilitate the understanding of the inputs that the user provides as relevance feedback" (see, for example, Ex. 1013 at 26:66-27:18). In a specific embodiment example, Kupiec discloses that such "displayed output" can be in the "form of an IR query is then presented, followed by a list of matching documents, in alphabetical title order" (see, for example, Ex. 1013 at 28:25-27).

B. Cheyer (Ex. 1019)

140. For example, amongst the numerous prior art references in this field, a published article by Adam Cheyer and Luc Julia entitled "Multimodal Maps: An Agent-Based Approach" ("Cheyer", see, for example, Ex. 1019 at p. 2) was first published by distribution of copies to attendees at the International Conference on Cooperative Multimodal Communication CMC/95

(see, for example, Ex. 1019 at p. 1) on May 24-26, 1995, which is more than 1 year before the earliest priority date of the ‘021, ‘718 and ‘061 Patents. I further understand that *Proceedings of the International Conference on Cooperative Multimodal Communication*, a printed publication that included Cheyer, was published, cataloged, indexed, and available to those of skill in the art in at least one library by September 13, 1996, which is more than 1 year before the earliest priority date of the ‘021, ‘718 and ‘061 Patents. Thus, I understand that Cheyer qualifies as prior art to the ‘021, ‘718 and ‘061 Patents at least under 35 U.S.C. § 102 (b).

141. I understand that the authors of Cheyer, Adam Cheyer and Luc Julia of SRI International in Menlo Park, CA (see, for example, Ex. 1019 at p. 2), are the same individuals as two of the named inventors of the ‘021, ‘718 and ‘061 Patents (see, for example, Ex. 1001 at [73], [75]).

142. As Cheyer summarizes in its “Abstract” section, Cheyer presents “a prototype map-based application for a travel planning domain” that is “distinguished by a synergistic combination of handwriting, gesture and speech modalities; access to existing data sources including the World Wide Web; and a mobile handheld interface” and that is implemented using “a hierarchical distributed network of heterogeneous software agents” that were “augmented by appropriate functionality for developing synergistic multimodal applications” (see, for example, Ex. 1019 at p. 2). Thus, Cheyer is clearly from the same field of art as the ‘021, ‘718 and ‘061 Patents and is clearly addressing similar problems as those purportedly addressed by the ‘021, ‘718 and ‘061 Patents.

143. Cheyer describes “Direct manipulation interface technologies” that are “currently the most widely used techniques for creating user interfaces” as comprising “the use of menus and a graphical user interface” such that “users are presented with sets of discrete actions and the

objects on which to perform them" (see, for example, Ex. 1019 at p. 2). Cheyer further notes that "Pointing devices such as a mouse facilitate selection of an object or action, and drag and drop techniques allow items to be moved or combined with other entities or actions" and "Gestures allow users to communicate a surprisingly wide range of meaningful requests with a few simple strokes" (see, for example, Ex. 1019 at pp. 2-3).

144. According to Cheyer, "Direct manipulation interactions possess many desirable qualities: communication is generally fast and concise; input techniques are easy to learn and remember; the user has a good idea about what can be accomplished, as the visual presentation of the available actions is generally easily accessible" (see, for example, Ex. 1019 at p. 3). Cheyer further discloses that "Limitations of direct manipulation style interfaces can be addressed by another interface technology, that of natural language interfaces" because such "Natural language interfaces excel in describing entities that are not currently displayed on the monitor, in specifying temporal relations between entities or actions, and in identifying members of sets" (see, for example, Ex. 1019 at p. 3). According to Cheyer, "These strengths are exactly the weaknesses of direct manipulation interfaces, and concurrently, the weaknesses of natural language interfaces (ambiguity, conceptual coverage, etc.) can be overcome by the strengths of direct manipulation" (see, for example, Ex. 1019 at p. 3).

145. Cheyer also describes that "Natural language content can be entered through different input modalities, including typing, handwriting, and speech" and that "the same textual content can be provided by the three modalities" wherein "Spoken language is the modality used first and foremost in human-human interactive problem solving", "Typing is the most common way of entering information into a computer, because it is reasonably fast, very accurate, and

requires no computational resources", and "Handwriting has been shown to be useful for certain types of tasks" (see, for example, Ex. 1019 at p. 3).

146. Cheyer observes that "direct manipulation and natural language seem to be very complementary modalities" and admits that "It is therefore not surprising that a number of multimodal systems combine the two" with reference to several prior art systems (see, for example, Ex. 1019 at p. 3). For example, Cheyer notes that "A number of systems have focused on combining the speed of speech with the reference provided by direct manipulation of a mouse pointer" such as "CUBRICON", which combines "complex spoken input with mouse clicks, using several knowledge sources for reference identification" (see, for example, Ex. 1019 at p. 4). Cheyer further notes that the prior art CUBRICON addresses "a map-based task, making it similar to the application developed in [Cheyer]" and that CUBRICON can "use direct manipulation to indicate a specific item" (see, for example, Ex. 1019 at p. 4). Cheyer also observes that prior art "TAPAGE is another system that allows true synergistic combination of spoken input with direct manipulation" and notes that "TAPAGE, selected as a building block for our map application, will be described more in detail in section 4.2" (see, for example, Ex. 1019 at p. 4).

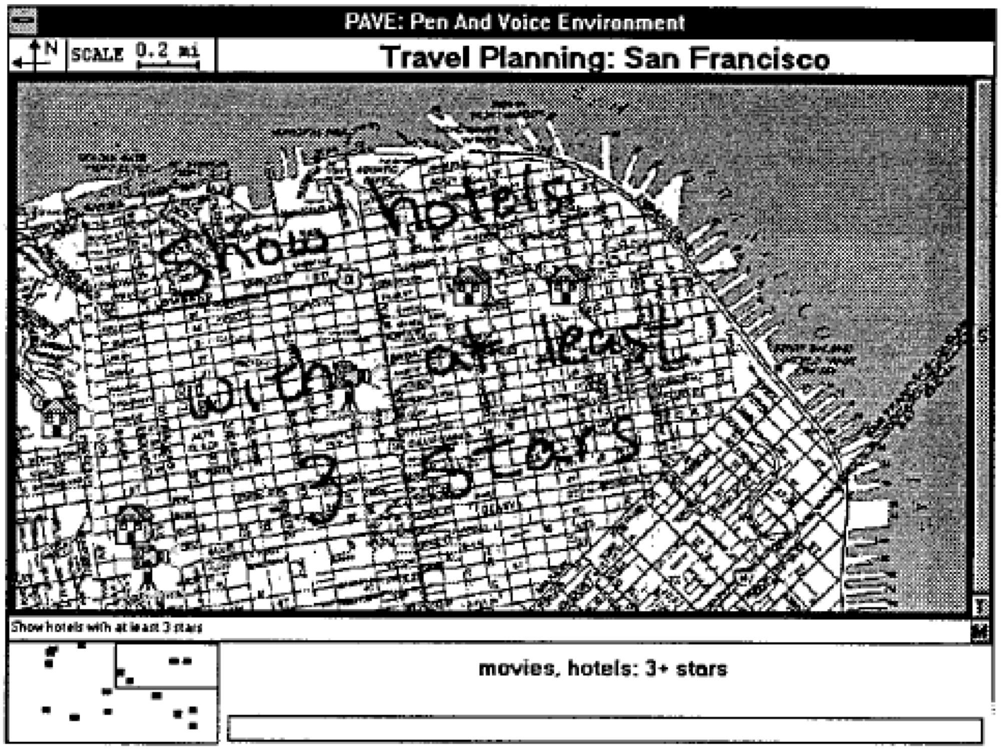
147. In the section entitled "A Multimodal Map Application", Cheyer describes "a prototype map-based application for a travel planning domain" wherein "the system permits the user to simultaneously combine direct manipulation, gestural drawings, handwritten, typed and spoken natural language" (see, for example, Ex. 1019 at pp. 4-5).

148. More specifically, Cheyer describes the system design criteria as including a "user interface" that is "light and fast enough to run on a handheld PDA while able to access applications and data that may require a more powerful machine" (see, for example, Ex. 1019 at

p. 5). Cheyer also describes that the system uses "Existing commercial or research natural language and speech recognition systems" (see, for example, Ex. 1019 at p. 5). Cheyer further describes the system as enabling "Through the multimodal interface" that "a user" can "transparently access a wide variety of data sources, including information stored in HTML form on the World Wide Web" (see, for example, Ex. 1019 at p. 5).

149. Cheyer also describes this example application via Figure 1, entitled "Multimodal application for travel planning" (see, for example, Ex. 1019 at p. 4, Figure 1 reproduced below).

150.


PAVE: Pen And Voice Environment
N
SCALE 0.2 mi
Travel Planning: San Francisco
Show hotels with at least 3 stars

Show hotels with at least 3 stars

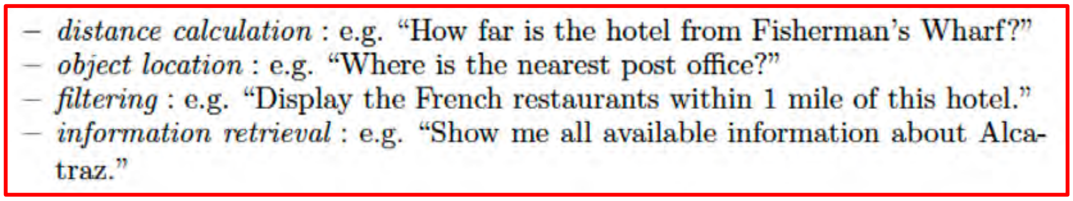
movies, hotels: 3+ stars

Figure 1: Multimodal Application for Travel Planning

151. In reference to Figure 1, Cheyer discloses that "the user is presented with a pen sensitive map display on which drawn gestures and written natural language statements may be combined with spoken input" such that "content presented by the map change, according to the requests of the user", "Objects of interest" are "displayed as icons" and the "user may ask the

map to perform various actions" such as the examples in the excerpt of Cheyer given below (see, for example, Ex. 1019 at p. 5, equivalent excerpt reproduced below):

152.



- *distance calculation* : e.g. "How far is the hotel from Fisherman's Wharf?"
- *object location* : e.g. "Where is the nearest post office?"
- *filtering* : e.g. "Display the French restaurants within 1 mile of this hotel."
- *information retrieval* : e.g. "Show me all available information about Alcatraz."

153. Cheyer further notes that "The application also makes use of multimodal (multimedia) output as well as input: video, text, sound and voice can all be combined when presenting an answer to a query" (see, for example, Ex. 1019 at p. 5). For example, Cheyer discloses that "During input, requests can be entered using gestures (see Figure 2 for sample gestures)" (see, for example, Ex. 1019 at p. 5, and Figure 2 reproduced below).

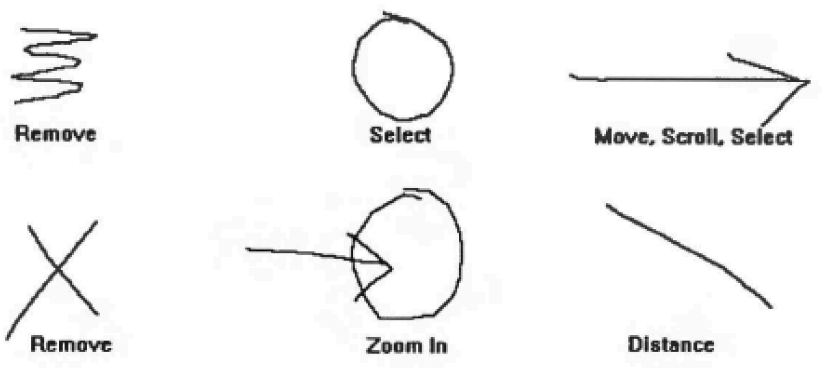


Figure 2: Sample gestures

154. Cheyer describes the system as also having a "user interface" that "runs on pen-equipped PC's or a Dauphin handheld PDA" using "either a microphone or a telephone for voice input" (see, for example, Ex. 1019 at p. 5). Cheyer continues this system description by disclosing that "The interface is connected either by modem or ethernet to a server machine which will manage database access, natural language processing and speech recognition for the application" such that "The result is a mobile system that provides a synergistic pen/voice interface to remote databases" (see, for example, Ex. 1019 at pp. 5-6).

155. Cheyer specifically references the prior art user manual for the "Dauphin handheld PDA" (see, for example, Ex. 1019 at pp. 5, 11), which was a very well-known handheld portable computing device in the mid-1990s, but Cheyer does not provide a visual depiction of this commercially-available product. Such "personal digital assistants" were known to a POSITA at the time of the alleged invention to be normally "battery-powered". I personally recall the Dauphin handheld computer quite well from that timeframe, and I provide this image from the Internet that provides an exemplary view of this device (from http://oldcomputers.net/pics/dauphin-dtr-1-front.jpg):

156.

157. In the section entitled "Approach", Cheyer describes that "In order to implement the application described in the previous section, we chose to augment a proven agent- based architecture with functionalities developed for a synergistically multimodal application" (see, for example, Ex. 1019 at p. 6).

158. More specifically, Cheyer describes as prior art from 1994 an "Open Agent Architecture (OAA)" (see, for example, Ex. 1019 at pp. 6, 11) that "provides a framework for

coordinating a society of agents which interact to solve problems for the user" and "provides distributed access to commercial applications, such as mail systems, calendar programs, databases, etc." (see, for example, Ex. 1019 at p. 6). According to Cheyer, this "Open Agent Architecture possesses several properties which make it a good candidate for our needs" including, for example, "An Interagent Communication Language (ICL) and Query Protocol" that allows "agents to communicate among themselves" and "a speech recognition agent" to "provide transparent access to the Corona speech recognition system" (see, for example, Ex. 1019 at pp. 6-7).

159. Cheyer further explains that the "architecture" for this prior art "Open Agent Architecture (OAA)" is "based loosely" on an even older prior art "FLiPSiDE system" from 1993 (see, for example, Ex. 1019 at pp. 7, 12) that "uses a hierarchical configuration where client agents connect to a "facilitator" server" (see, for example, Ex. 1019 at p. 7).

160. According to Cheyer, such "Facilitators provide content-based message routing, global data management, and process coordination for their set of connected agents" and "Facilitators can, in turn, be connected as clients of other facilitators" (see, for example, Ex. 1019 at p. 7). Furthermore, Cheyer discloses that "Each facilitator records the published functionality of their sub-agents, and when queries arrive in Interagent Communication Language form, they are responsible for breaking apart any complex queries and for distributing goals to the appropriate agents" such that "An agent solving a goal may require supporting information and the agent architecture provides numerous means of requesting data from other agents or from the user" (see, for example, Ex. 1019 at p. 7).

161. Cheyer continues the discussion of "Building Blocks" for its "Approach" by describing another prior art system "TAPAGE" from 1994 (see, for example, Ex. 1019 at pp. 7,

11) that can “capture signals emitted during a user's interaction” and also “integrates a set of modality agents, each responsible for a very specialized kind of signal” (see, for example, Ex. 1019 at p. 7). Cheyer further explains in reference to TAPAGE that the “modality agents are connected to an ‘interpret agent’ which is responsible for combining the inputs across all modalities to form a valid command for the application” wherein this “interpret agent receives filtered results from the modality agents, sorts the information into the correct fields, performs type-checking on the arguments, and prompts the user for any missing information” (see, for example, Ex. 1019 at p. 7).

162. Additionally, Cheyer discloses in reference to TAPAGE that “The interpret agent is also responsible for merging the data streams sent by the modality agents, and for resolving ambiguities among them, based on its knowledge of the application's internal state” and that the “system can accept multimodal input” (see, for example, Ex. 1019 at p. 7).

163. Cheyer continues with a section entitled “Synthesis” by describing that “In the Open Agent Architecture, agents are distributed entities that can run on different machines, and communicate together to solve a task for the user” (see, for example, Ex. 1019 at p. 8). More specifically Cheyer discloses “Macro Agents”, which “contain some knowledge and ability to reason about a domain, and can answer or make queries to other macro agents using the Interagent Communication Language”, and “Micro Agents”, which “are responsible for handling a single input or output data stream, either filtering the signal to or from a hierarchically superior ‘interpret’ agent” (see, for example, Ex. 1019 at p. 8).

164. According to Cheyer, the “network architecture” used was “hierarchical at two resolutions” wherein “micro agents are connected to a superior macro agent” and “macro agents

are connected in turn to a facilitator agent" but "In both cases, a server is responsible for the supervision of its client sub-agents" (see, for example, Ex. 1019 at p. 8).

165. Cheyer also specifically describes, from a 1990 prior art article, a "Speech Recognition (SR) Agent" that "provides a mapping from the Interagent Communication Language to the API for the Decipher (Corona) speech recognition system", which Cheyer describes as "a continuous speech speaker independent recognizer based on Hidden Markov Model technology" (see, for example, Ex. 1019 at p. 8). According to Cheyer, "This macro agent is also responsible for supervising a child micro agent whose task is to control the speech data stream" and this "SR agent can provide feedback to an interface agent about the current status and progress of the micro agent (e.g. "listening", "end of speech detected", etc.) (see, for example, Ex. 1019 at p. 8). Cheyer further specifically describes, with reference to a 1994 prior art article, a "Natural Language (NL) Parser Agent" that "translates English expressions into the Interagent Communication Language (ICL)" (see, for example, Ex. 1019 at p. 8).

166. According to Cheyer, "Database Agents" can "reside at local or remote locations and can be grouped hierarchically according to content" wherein such "databases" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the World Wide Web (WWW)" as well as "information" that is "extracted by an HTML reading database agent" such as a "list of current movie times and reviews" (see, for example, Ex. 1019 at p. 8).

167. Cheyer also discloses a "Reference Resolution Agent" that is "responsible for merging requests arriving in parallel from different modalities, and for controlling interactions between the user interface agent, database agents and modality agents" (see, for example, Ex. 1019 at pp. 8-9). Cheyer explains that "the reference resolution agent is domain specific" and

the "agent can verify argument types, supply default values, and resolve argument references" (see, for example, Ex. 1019 at p. 9). In some cases, these "argument references are descriptive" such that "a domain agent will try to resolve the definite reference by sending database agent requests" but "Other references, particularly when contextual or deictic, are resolved by the user interface agent" (see, for example, Ex. 1019 at p. 9). However, "Once arguments to a query have been resolved", then this Reference Resolution Agent "coordinates the actions and calculations necessary to produce the result of the request" (see, for example, Ex. 1019 at p. 9).

168. Cheyer further discloses an "Interface Agent" as a "macro agent" that is "responsible for managing what is currently being displayed to the user, and for accepting the user's multimodal input" (see, for example, Ex. 1019 at p. 9). According to Cheyer, this "Interface Agent also coordinates client modality agents and resolves ambiguities among them" such that "handwriting and gestures are interpreted locally by micro agents and combined with results from the speech recognition agent, running on a remote speech server" where the "handwriting micro-agent interfaces with the Microsoft PenWindows API and accesses a handwriting recognizer by CIC Corporation" (see, for example, Ex. 1019 at p. 9).

169. Additionally, Cheyer notes that "An important task for the interface agent is to record which objects of each type are currently salient, in order to resolve contextual references such as "the hotel" or "where I was before"" wherein such "Deictic references are resolved by gestural or direct manipulation commands" (see, for example, Ex. 1019 at p. 9). Cheyer informs that "If no such indication is currently specified, the user interface agent waits long enough to give the user an opportunity to supply the value, and then prompts the user for it" (see, for example, Ex. 1019 at p. 9).

170. Cheyer illustrates in Figure 3 an "Agent Architecture for Map Application" as shown below (from Ex. 1019 at p. 9, Figure 3):

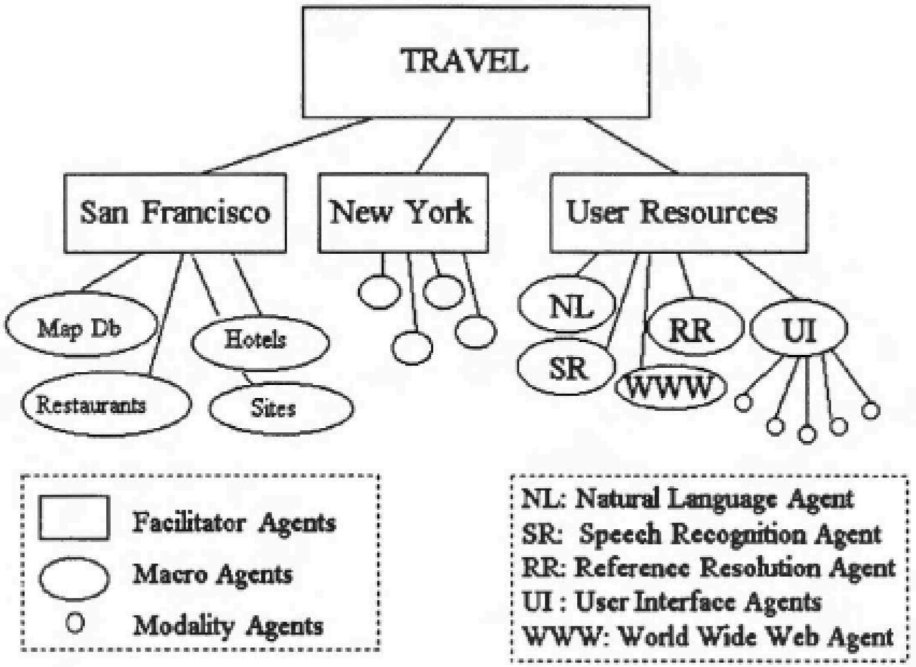
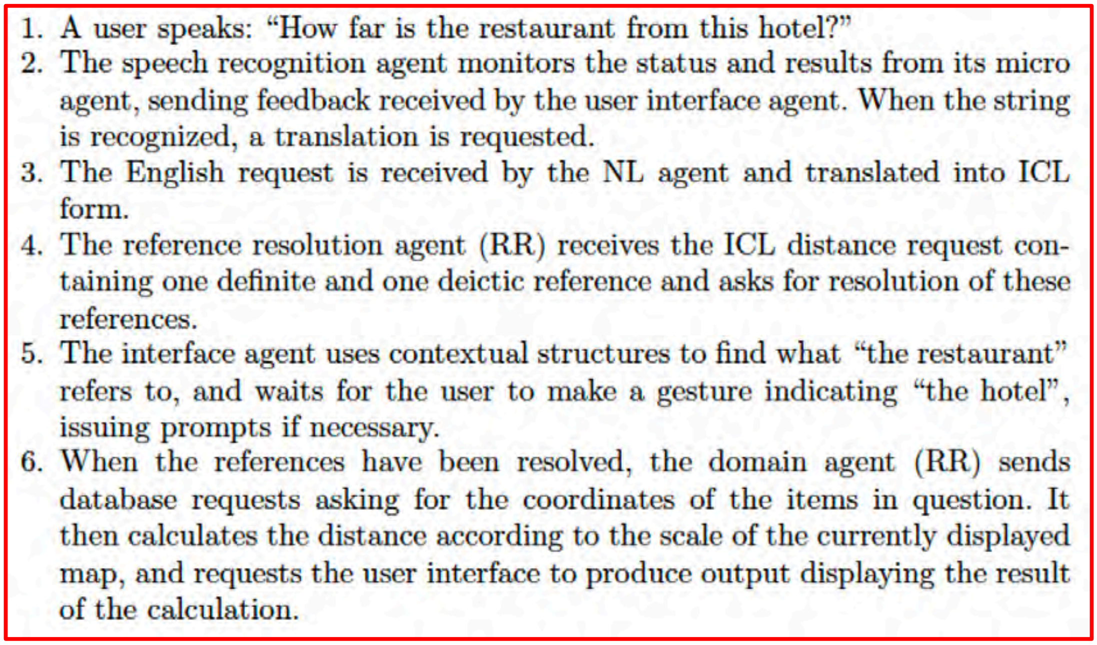

TRAVEL
San Francisco
New York
User Resources
Map Db
Hotels
Restaurants
Sites
NL
RR
UI
SR
WWW
Facilitator Agents
Macro Agents
Modality Agents
NL: Natural Language Agent
SR: Speech Recognition Agent
RR: Reference Resolution Agent
UI : User Interface Agents
WWW: World Wide Web Agent

Figure 3: Agent Architecture for Map Application

171.

172. Cheyer also describes "an example of the distributed interaction of agents for a specific query" wherein "all communication among agents passes transparently through a facilitator agent in an undirected fashion" as summarized in the excerpt below (from Ex. 1019 at pp. 9-10, equivalent excerpt reproduced below):

173.



1. A user speaks: "How far is the restaurant from this hotel?"
2. The speech recognition agent monitors the status and results from its micro agent, sending feedback received by the user interface agent. When the string is recognized, a translation is requested.
3. The English request is received by the NL agent and translated into ICL form.
4. The reference resolution agent (RR) receives the ICL distance request containing one definite and one deictic reference and asks for resolution of these references.
5. The interface agent uses contextual structures to find what "the restaurant" refers to, and waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary.
6. When the references have been resolved, the domain agent (RR) sends database requests asking for the coordinates of the items in question. It then calculates the distance according to the scale of the currently displayed map, and requests the user interface to produce output displaying the result of the calculation.

174. Cheyer's "Conclusions" section summarizes that "By augmenting an existing agent-based architecture with concepts necessary for synergistic multimodal input", Cheyer has disclosed "a mobile, synergistic pen/voice interface providing good natural language access to heterogeneous distributed knowledge sources" with an "approach" that "should provide a for developing synergistic multimodal applications for other domains" (see, for example, Ex. 1019 at p. 10).

C. Kimura (Ex. 1015)

175. For example, amongst the numerous prior art references in this field, U.S. Patent No. 5,247,580 by Toshiyuki Kimura and Kazuo Yabe entitled "Voice-operated remote control system" ("Kimura") was filed on Jul. 22, 1992, and issued on Sep. 21, 1993, which is more than 1 year before the earliest priority date of the '021, '718 and '061 Patents (see, for example, Ex. 1015 at (22), (45)). Thus, I understand that Kimura qualifies as prior art to the '021, '718 and '061 Patents at least under 35 U.S.C. §§ 102(a), (b), and (e).

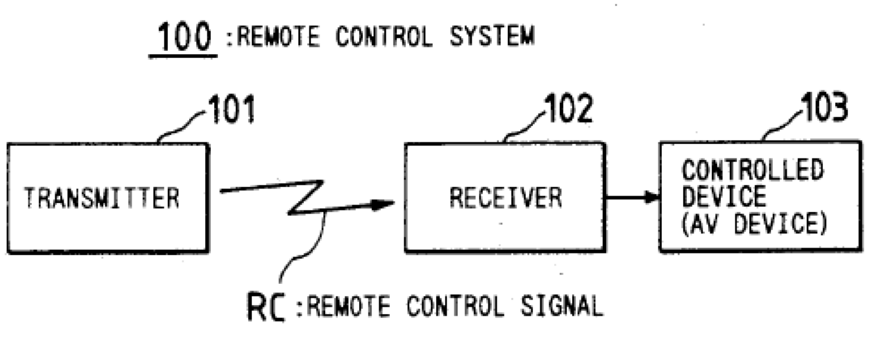
176. As Kimura summarizes in its "Background of the Invention" section, the Kimura patent "relates to a remote control system for remotely controlling various electronic devices,

and more particularly to a remote control system for remotely controlling devices such as AV (audio visual) devices by way of voice commands" including such "AV devices" as "television receivers" (see, for example, Ex. 1015 at 1:8-14). Additionally, Kimura is directed to a "voice-operated remote control system which can vary a speech recognition process depending on the degree of importance of a control command" in view of the fact that "the magnitudes of effects caused by erroneous recognition, may not necessarily be the same" (see, for example, Ex. 1015 at 1:41-42, 1:54-57). Thus, Kimura is clearly from the same field of art as the '021, '718 and '061 Patents and is clearly addressing similar problems as those purportedly addressed by the '021, '718 and '061 Patents.

177. According to Kimura, "FIG. 1 is a block diagram of a general remote control system" that "comprises a transmitter **101** for transmitting a remote control signal from a position remote from a controlled device **103** such as an AV device, and a receiver **102** for receiving the transmitted remote control signal, decoding the remote control signal, and sending the decoded information to the controlled device **103**" (see, for example, Ex. 1015 at 2:40-41, 3:10-15 and FIG. 1 reproduced below).

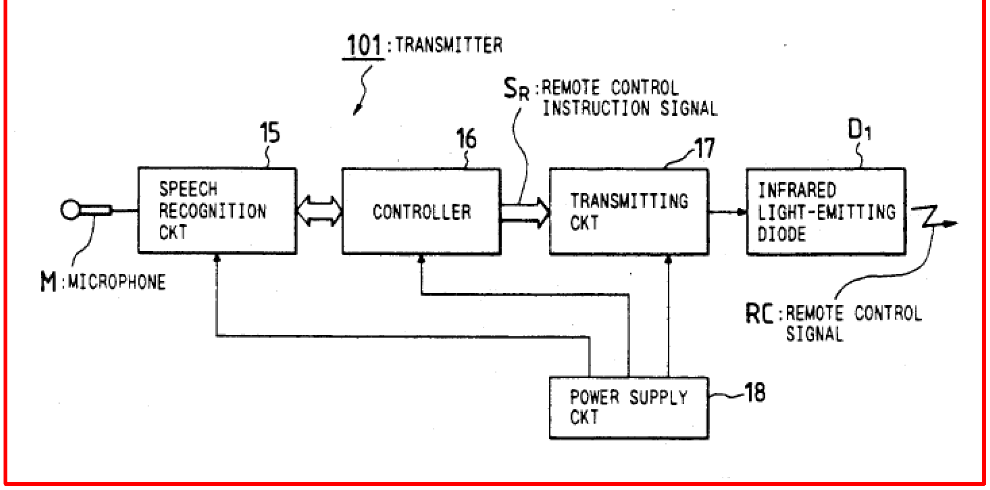
178.

FIG. 1


100 :REMOTE CONTROL SYSTEM
101
102
103
TRANSMITTER
RECEIVER
CONTROLLED DEVICE (AV DEVICE)
RC :REMOTE CONTROL SIGNAL

179. More specifically, Kimura discloses that "FIG. 3 is a block diagram of the transmitter of a general voice-operated remote control system", wherein the "transmitter **101** has a microphone **M** for converting a voice command into an electric signal" that "is applied to a speech recognition circuit **15** in the form of a speech recognition LSI circuit or the like which includes a microprocessor" and "produces command data corresponding to the recognized contents", and further wherein "The transmitter **101** also has a controller **16** comprising a microprocessor" (see, for example, Ex. 1015 at 2:44-45, 3:27-36 and FIG. 3 reproduced below).

FIG. 3



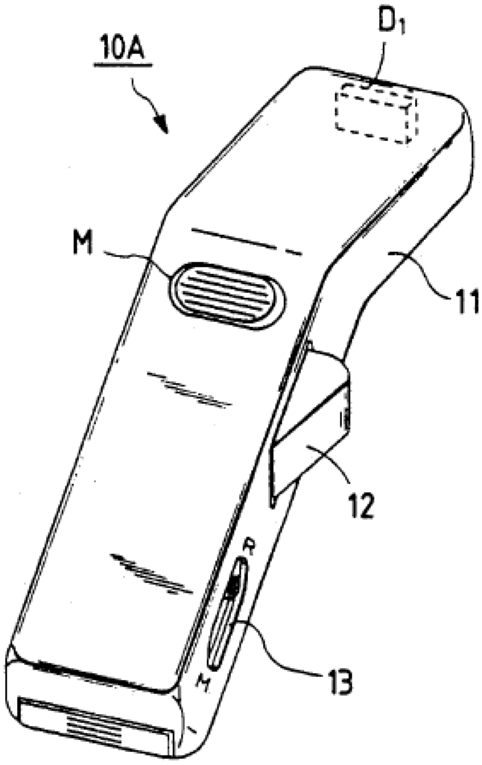
180.

181. Kimura discloses that "When a voice command is received through the microphone **M**, the speech recognition circuit **15** converts the voice command into pattern data" and then "compares the voice command pattern data with a plurality of standard pattern data which are stored therein, and determines the distance between the voice command data and the standard pattern data, and outputs command data corresponding to the standard pattern data, the distance of which from the voice command pattern data is smallest", thereby causing that "The command data thus produced are applied to the controller **16**" (see, for example, Ex. 1015 at 3:46-62).

182. Kimura further discloses that "The controller **16** sends a remote control signal SR corresponding to the applied command data to the transmitting circuit **17**" that "drives the infrared light-emitting diode D1 to transmit a remote control signal RC" such that "The controlled device **103** is therefore remotely controlled by the remote control signal RC" (see, for example, Ex. 1015 at 3:63-4:2).

183. According to Kimura, "FIG. 4 is a perspective view of the transmitter of a voice-operated remote control system according to a first embodiment of the present invention" (see, for example, Ex. 1015 at 2:46-48 and FIG. 4 reproduced below).

FIG. 4



184.

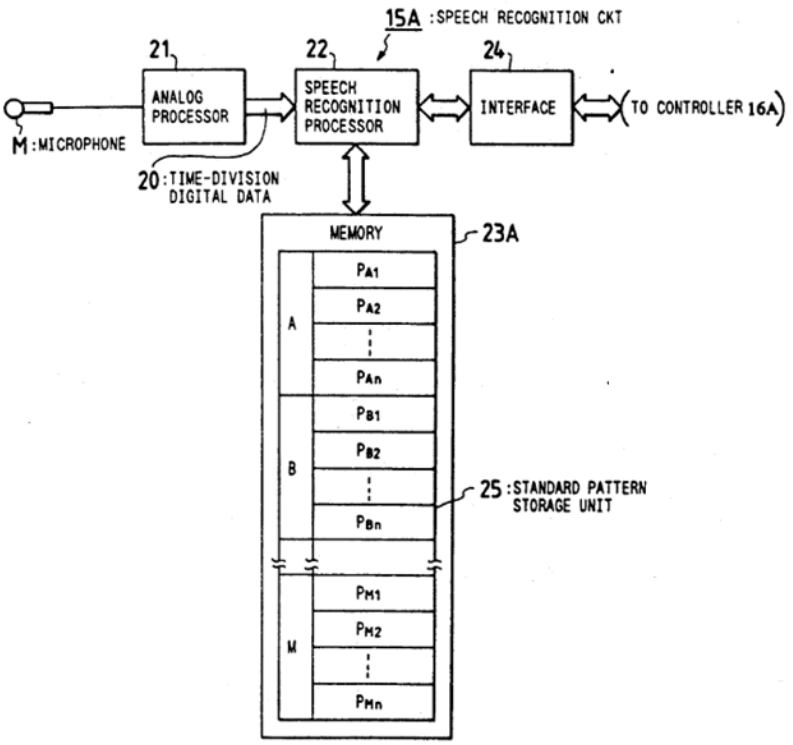
185. Kimura describes in reference to FIG. 4 that "transmitter **10A** of the voice-operated remote control system has a unitary casing **11** which allows the operator to carry the transmitter freely around" (see, for example, Ex. 1015 at 4:9-12). Kimura further describes that

"casing **11** supports a microphone **M** on an upper panel thereof" wherein "microphone **M** converts a voice command given by the operator into an electric signal" and on "one side of the casing **11**, there is disposed a voice input switch (hereinafter referred to as a "talk switch") **12** which is closed when pressed and can automatically be released or opened when released" so that when a "voice command is to be entered, the talk switch **12** is closed to operate the transmitter **10A**" or "Otherwise, the talk switch **12** is open keeping the transmitter **10A** out of operation" (see, for example, Ex. 1015 at 4:12-28). Kimura also discloses that "casing **11** also supports, on its side, a mode selector switch **13** in the form of a slide-type switch" wherein such "mode selector switch **13** serves to select one of modes at a time" that "include a speech registration mode in which a voice command is registered in the transmitter **10A** and a speech recognition mode in which a voice command is recognized" (see, for example, Ex. 1015 at 4:29-35).

186. According to Kimura, "FIG. 7 is a block diagram of a speech recognition circuit according to the first embodiment" (see, for example, Ex. 1015 at 2:54-55 and FIG. 7 reproduced below).

187.

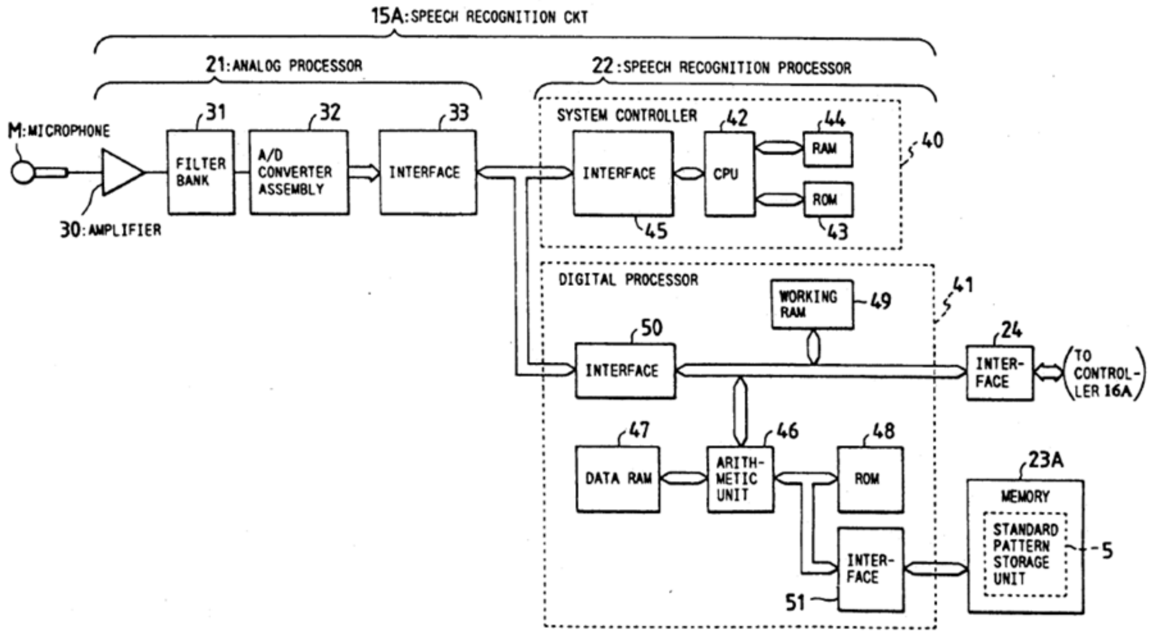
FIG. 7


15A :SPEECH RECOGNITION CKT
21
22
24
ANALOG PROCESSOR
SPEECH RECOGNITION PROCESSOR
INTERFACE
TO CONTROLLER 16A
M :MICROPHONE
20 :TIME-DIVISION DIGITAL DATA
MEMORY
23A
A
PA1
PA2
PAn
B
PB1
PB2
PBn
25 :STANDARD PATTERN STORAGE UNIT
M
PM1
PM2
PMn

188. Kimura describes in reference to FIG. 7 that "the speech recognition circuit **15A** comprises an analog processor **21** for processing an analog voice command signal which is received through the microphone **M** and outputting the processed analog voice command signal as a time-division digital data **20**, a speech recognition processor **22** for recognizing the voice command based on the time-division digital data **20** from the analog processor **21**, a memory **23A** for storing standard pattern data for speech recognition, and an interface **24** for transmitting signals to and receiving signals from the controller **16A**" and the "memory **23A** includes a standard pattern data storage unit **25** which stores a plurality of different standard pattern data through PAn, PB1 through PBn, . . . , PM1 through PMn with respect to respective voice commands" (see, for example, Ex. 1015 at 5:3-18).

189. According to Kimura, "FIG. 8 is a detailed block diagram of the speech recognition circuit according to the first embodiment" (see, for example, Ex. 1015 at 2:56-57 and FIG. 8 reproduced below).

FIG. 8



190. Kimura describes in reference to FIG. 8 that "the speech recognition processor **22** comprises a system controller **40** for analyzing and processing control commands from the controller **16** and also for controlling the entire operation of the speech recognition processor **22**, and a digital processor **41** for effecting distance calculations and controlling the memory **23A**" wherein the "system controller **40** comprises a CPU (Central Processing Unit) **42** for controlling the overall operation of the transmitter **1**" (see, for example, Ex. 1015 at 6:21-30).

D. Freeman (Ex. 1014)

191. For example, amongst the numerous prior art references in this field, U.S. Patent No. 6,006,227 by Eric Freeman et al., entitled "Document Stream Operating System" ("Freeman") was filed on Jun. 28, 1996, which is more than 1 year before the earliest priority

date of the ‘021, ‘718 and ‘061 Patents, and issued on Dec. 21, 1999 (see, for example, Ex. 1014 at (22), (45)). Thus, I understand that Freeman qualifies as prior art to the ‘021, ‘718 and ‘061 Patents at least under 35 U.S.C. § 102(e).

192. As Freeman summarizes in its “Background of the Invention” section, the Freeman patent “relates to an operating system in which documents are stored in a chronologically ordered "stream"” (see, for example, Ex. 1014 at 1:4-6). Additionally, Freeman is directed to “documents” that “can contain any type of data including but not limited to pictures, correspondence, bills, movies, voice mail and software programs” (see, for example, Ex. 1014 at 4:16-18) and is directed to “streams” that “can be controlled by a voice-interface as well as a computer and thereby be accessed via a conventional phone” (see, for example, Ex. 1014 at 11:38-40). Thus, Freeman is clearly from the same field of art as the ‘021, ‘718 and ‘061 Patents and is clearly addressing similar problems as those purportedly addressed by the ‘021, ‘718 and ‘061 Patents.

193. Freeman describes that “This invention is a new model and system for managing personal electronic information” wherein “streams and filters provide a unified framework that subsumes many separate desktop applications to accomplish and handle personal communication, scheduling, and search and retrieval tasks” (see, for example, Ex. 1014 at 3:62-4:2). Freeman discloses that in this system “location and nature of file storage is transparent to the user” such that “computers using the operating system of the present invention need not be independent data storage devices” but instead can be “viewpoints” to “data stored and maintained on external systems such as the INTERNET” (see, for example, Ex. 1014 at 2:20-22, 2:52-56). Freeman states that “in accordance with the present invention users can access their personal document streams from any available platform such as a UNIX machine, a Macintosh

or IBM-compatible personal computer, a personal digital assistant (PDA), or a set-top box via cable” (see, for example, Ex. 1014 at 2:56-61).

194. Freeman disclose that “streams can be organized on the fly with the find operation” wherein “Find prompts for a search query, such as "all E-mail I haven't responded to," or "all faxes I've sent to Schwartz" and creates a substream” that “according to the present invention contains all documents that are relevant to the search query” (see, for example, Ex. 1014 at 4:48-56). Additionally, Freeman discloses that “The find operation creates a substream” that can be “based on, for example, a boolean attribute-and-keyword expression or a `chronological expression`, for example, "my last letter to Schwartz"” or “may point to the future, for example, "my next appointment"” (see, for example, Ex. 1014 at 4:62-67).

195. Freeman also describes an “embodiment of the present invention” that “is implemented in a client/server architecture running over the Internet” as well as embodiments that implement “a client viewport using graphically based X Windows”, “a client viewport solely with text in standard ASCII”, and “a client viewport for the NEWTON personal digital assistant (PDA)” (see, for example, Ex. 1014 at 6:8-9, 6:17-23). Furthermore, Freeman notes that “The X Windows viewport provides the full range of functionalities including picture and movie display” and that “The X Windows viewport embodiment is shown in FIG. 1” (see, for example, Ex. 1014 at 6:23-24, 6:30-31, FIG. 1 reproduced below).

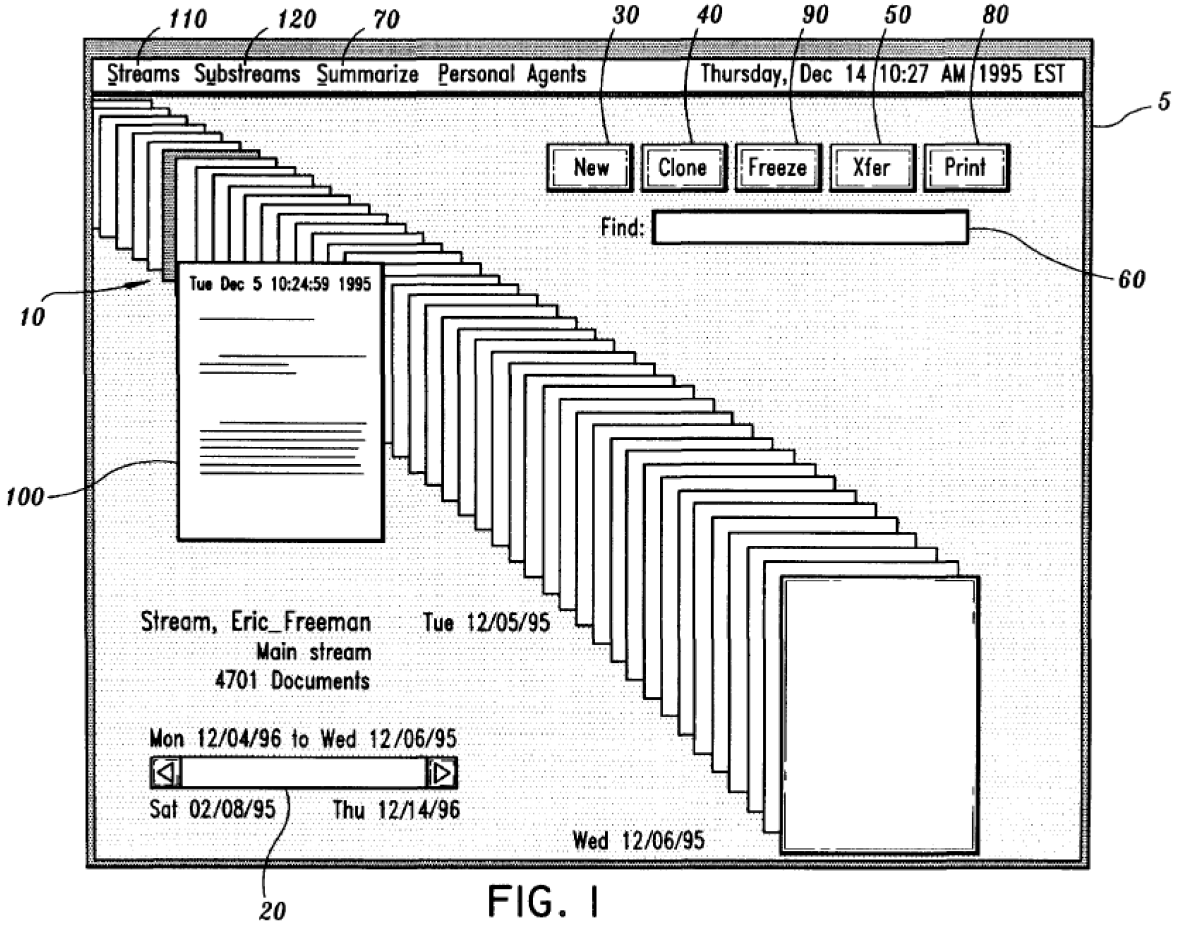


FIG. 1

196. In reference to FIG. 1, Freeman describes that “The interface is based on a visual representation of the stream metaphor **5**” wherein “Users can slide the mouse pointer **10** over the document representations to "glance" at each document, or use the scroll bar **20** in the lower left-hand corner to move through time, either into the past or into the future portion of the stream” (see, for example, Ex. 1014 at 6:31-36). Additionally, Freeman describes that “Pulldown menus are used to select documents from streams or existing substreams, create summaries, initiate personal agents and change the clock” and that “The Streams menu **110** allows the user to select from a list of locally available streams” (see, for example, Ex. 1014 at 7:8-12).

197. Freeman’s FIG.1 also illustrates a “Personal Agents” menu that “lists a number of available software agent types” where such “Personal software agents can be added to the user interface in order to automate common tasks” (see, for example, Ex. 1014 at 7:36-38). Freeman

also states that “any software agent with the necessary access can ride your stream” and thus “streams can be the basis of groupware systems implemented for example as a flock of agents” (see, for example, Ex. 1014 at 10:52-55).

198. Freeman further describes an embodiment wherein “a phone conversation is stored as a time-ordered sequence of spoken sounds or as electronic representations”, thereby enabling “two users” to “have a phone conversation” where “the users can use software such as a software agent” and “Each user's `phone agent` tosses digitized representations of speech frames onto the stream and grabs each new frame that appears, turning each speech frame into sound” (see, for example, Ex. 1014 at 11:16-23).

199. Freeman discloses another embodiment wherein “a television source can be stored as a time-ordered sequence of sound-and-image frames” such that “television information is an archive as well as a realtime source and can be searched and substreamed” and a “television set is merely a viewport”, thereby enabling “scheduling information” to be “stored in the television stream's future and tuning into a television station” that “only requires double-clicking on the appropriate calling card” (see, for example, Ex. 1014 at 11:28-36). Freeman also states that “Similar embodiments can provide for radio stations, music sources, etc.” (see, for example, Ex. 1014 at 11:36-37).

200. Freeman specifically discloses that “A stream according to the present invention can be controlled by a voice-interface as well as a computer and thereby be accessed via a conventional phone” wherein this “voice interface would allow: (1) the stream to be searched and manipulated; (2) new objects to be installed; (3) objects to be transferred; and (4) other capability” (see, for example, Ex. 1014 at 11:38-43).

201. Freeman observes that “A stream is a data structure that can be examined and to the extent possible manipulated by many processes simultaneously” wherein “A stream must support simultaneous access because: (1) a user creates many software agents which may need to examine the stream concurrently; and (2) a user may have granted other users limited access to the user's stream, and the user will want access to this stream even while the other users access the stream” (see, for example, Ex. 1014 at 13:50-52, 13:59-64).

202. Freeman further discloses that “One embodiment of the present invention is configured such that each server may support three to four simultaneous users with stream sizes on the order of 100,000 documents (perhaps a year or two of documents for the average user)” but “In another embodiment, the operating system is configured such that lifestreams may have millions of documents or more” (see, for example, Ex. 1014 at 13:65-14:4). Freeman also discloses that “The substreaming aspect of one embodiment of the present invention is efficiently implemented using an inverse index of the document collection maintained by the server” such that “No real performance problems with respect to retrieval have occurred” and “Given the very large indices that are being used on the Internet the retrieval scheme is expected to scale to large document collections” (see, for example, Ex. 1014 at 14:4-10).

203. Freeman notes that “Since a user is unlikely look at 10,000 documents at once and discern any usable information, the present invention does not provide the user with an entire document collection at once” but “Instead "cursors" are used to allow the user to view segments of the document collection and to load in more segments as needed” (see, for example, Ex. 1014 at 14:11-16). Additionally, Freeman discloses “embodiments of the present invention utilize a multi-server and multi-threaded approach which provides a more scalable architecture” (see, for example, Ex. 1014 at 14:19-21).

204. With respect to the term "agent" Freeman notes that "this term refers one of three kinds of embedded computations: personal agents, document agents, and stream agents" wherein "Personal agents are typically attached to the user interface and can automate tasks or can learn from the user's interactions with streams", "Document agents live on documents and are spawned by various events, for example, the first time that a document is accessed", and "Stream agents are attached to streams and execute whenever the stream changes in some way, for example, a new document appears on the stream" (see, for example, Ex. 1014 at 14:22-33).

205. With respect to the term "document" Freeman notes that "this term includes traditional text based files, electronic mail files, binary files, audio data, video data, and multimedia data" (see, for example, Ex. 1014 at 14:34-37).

VIII. OBVIOUSNESS OF THE '718 PATENT UNDER 35 U.S.C. § 103 DUE TO KUPIEC IN VIEW OF CHEYER AND KIMURA OR FREEMAN

206. In my opinion, Kupiec in view of Cheyer renders obvious at least Claims 1-4, 6, 8-13, 15, 17-22, 24 and 26-27 of the '718 Patent for at least the reasons described herein.

207. In my opinion, Kupiec in view of Cheyer further in view of Kimura renders obvious at least Claims 1-4, 6, 8-13, 15, 17-22, 24 and 26-27 of the '718 Patent for at least the reasons described herein.

208. In my opinion, Kupiec in view of Cheyer further in view of Freeman renders obvious at least Claims 6, 15 and 24 of the '718 Patent for at least the reasons described herein.

209. A general overview of Kupiec is given at ¶¶ 89-139 above.

210. A general overview of Cheyer is given at ¶¶ 140-174 above.

211. A general overview of Kimura is given at ¶¶ 175-190 above.

212. A general overview of Freeman is given at ¶¶ 191-205 above.

213. My specific analyses of Kupiec in view of Cheyer and of Kupiec in view of Cheyer further in view of Kimura, with respect to every claim element of Claims 1-4, 6, 8-13, 15, 17-22, 24 and 26-27 of the '718 Patent, as well as Kupiec in view of Cheyer further in view of Freeman, with respect to Claims 6, 15 and 24 of the '718 Patent, are given herein.

<u>'718 Patent: Claim 1</u>

> 1. A method for speech-based navigation of an electronic data source located at one or more network servers located remotely from a user, wherein a data link is established between a mobile information appliance of the user and the one or more network servers, comprising the steps of:
>
> (a) receiving a spoken request for desired information from the user utilizing the mobile information appliance of the user, wherein said mobile information appliance comprises a portable remote control device or a set-top box for a television;
>
> (b) rendering an interpretation of the spoken request;
>
> (c) constructing a navigation query based upon the interpretation;
>
> (d) utilizing the navigation query to select a portion of the electronic data source; and
>
> (e) transmitting the selected portion of the electronic data source from the network server to the mobile information appliance of the user.

1. A method for speech-based navigation of an electronic data source located at one or more network servers located remotely from a user, wherein a data link is established between a mobile information appliance of the user and the one or more network servers, comprising the steps of:

214. In my opinion, this preamble claim element is a claim limitation at least because I believe that this preamble recites essential structure and/or steps that give life, meaning, and vitality to the claim as opposed to *only* stating a purpose or intended use for the alleged invention. See also ¶¶ 78-81 above.

215. See ¶ 84 above regarding claim construction for this claim element.

216. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "*relates to systems and methods for* transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer" including "transcription systems and methods appropriate for use in conjunction with computerized *information-retrieval (IR) systems* and methods" and "more particularly to *speech-recognition systems and methods appropriate for use in conjunction with computerized information-retrieval systems*" (emphasis added, see, for example, Ex. 1013 at 1:36-45).

217. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

218. Kupiec describes that "*Processor **10** is* a computer processing unit (CPU)" that typically is "*part of a* mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

219. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

220. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

221. Thus, Kupiec discloses a "method for speech-based navigation of an electronic data source" (the operation of a speech-recognition system in conjunction with an information-retrieval system) wherein such an "electronic data source" (the information-retrieval system or IR subsystem) is "located at one or more network servers located remotely from a user" (at least when the IR subsystem is located at a remote site and connected to the speech-recognition system via a suitable communication network), and wherein "a data link is established between a mobile information appliance of the user and the one or more network servers" (at least when a personal computer such as a disclosed portable personal digital assistant would be configured to receive spoken input and to connect to an information-retrieval system over a communications network).

222. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

223. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" (emphasis added, see, for example, Ex. 1019 at p. 2).

224. As Cheyer summarizes in its "Conclusions" section, "By augmenting an existing agent-based architecture with concepts necessary for synergistic multimodal input", Cheyer

discloses "a mobile, synergistic pen/*voice interface providing good natural language access to heterogeneous distributed knowledge sources*" (emphasis added, see, for example, Ex. 1019 at p. 10).

225. Cheyer specifically discloses that "*the system permits the user* to simultaneously combine direct manipulation, gestural drawings, handwritten, typed and *spoken natural language*" and that the system uses "Existing commercial or research natural language and *speech recognition systems*", thereby enabling "a user" to "transparently *access* a wide variety of data sources, including *information stored in HTML form on the World Wide Web*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

226. Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" wherein "The *interface is connected either by modem or ethernet to a server machine which will manage database access, natural language processing and speech recognition* for the application" such that "The result is a *mobile system that provides* a synergistic pen/voice *interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

227. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

228. Thus, Cheyer discloses a "method for speech-based navigation of an electronic data source" (the operation of a speech-recognition and navigation system in conjunction with remote databases and/or the World Wide Web) wherein such an "electronic data source" (such as remote databases and/or the World Wide Web) is "located at one or more network servers located remotely from a user" (at least because the World Wide Web is located across numerous

network servers remote from any individual user and remote databases are remote), and wherein "a data link is established between a mobile information appliance of the user and the one or more network servers" (at least when a handheld PDA is connected by modem or ethernet to a server machine which will manage database access, natural language processing and speech recognition).

229. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

230. At least because each of Kupiec and Cheyer discloses the limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein.

231. To the extent that Kupiec's disclosures are considered to not include the recited "mobile information appliance" of this claim (or any of its elements below, or its dependent claims herein), then in my opinion, a POSITA would have modified the system of Kupiec in view of the system of Cheyer specifically to arrive at a combination that meets the limitations of this claim element (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

232. First, Kupiec and Cheyer are both addressing the same basic problem of building a system for retrieval of information from remote electronic sources based upon an initial user inquiry made via spoken language that upon transcription is prone to errors and/or ambiguities (see, for example, ¶¶ 90, 91, 144 and 173 above). Moreover, Cheyer's subsequent patents cite to subject-matter related earlier material of Kupiec (see, for example, Ex. 1001 at (56), Ex. 1003 at (56), Ex. 1005 at (56), citing the Kupiec US Patent No. 5,519,608), thereby further motivating a

POSITA to combine disclosures by Cheyer with disclosures by Kupiec within the same field of related art.

233. Second, Kupiec and Cheyer each describe systems with substantial similarities to each other, as well as this claim, as evident at least by the fact that each of Kupiec and Cheyer discloses many of the same limitations of this entire claim, as evident from my analysis herein. Additionally, as described herein, Kupiec discloses implementation of functionality for this claim in the form of software modules while Cheyer discloses implementation of functionality for this claim in the form of software agents, thereby informing a POSITA that the likelihood of success in combining the software agent approach of Cheyer with the software module approach of Kupiec would be very high and very predictable.

234. Third, Kupiec and Cheyer both rely upon the same fundamental speech transcription technology (the hidden Markov model) and thus both are susceptable to similar kinds of transcription errors for which convenient user input via a portable interface device is likely to be needed (see, for example, ¶¶ 131 and 165 above).

235. Fourth, Kupiec and Cheyer both arrive at the conclusion that information retrieval from remote electronic sources based upon an initial user inquiry in a natural language modality can benefit from usage of handheld personal digital assistants (see, for example, ¶¶ 91 and 154 above). Additionally, a POSITA at the time of the alleged invention of the ‘718 Patent would have been well aware of a general trend to migrate numerous applications from desktop or fixed client devices to mobile information appliances such as personal digital assistants. For example, at the time of the alleged invention of the ‘718 Patent, wireless local area networking technologies such as HomeRF and Wi-Fi were coming into rapid adoption (see, for example, ¶ 21 above), thereby enabling numerous applications, including information retrieval, that had

previously been known to operate over Ethernet to become wireless applications operating on a mobile information appliance such as a personal digital assistant. Similarly, at the time of the alleged invention of the '718 Patent, wireless wide area data networking technologies such as CDPD, IS-95B, GPRS and 3G were coming into rapid adoption (see, for example, https://en.wikipedia.org/wiki/History_of_mobile_phones), thereby enabling numerous applications, including information retrieval, that had previously been known to operate over a dial-up modem to become wireless applications operating on a mobile information appliance such as what today is more commonly called a smartphone.

236. Fifth, Kupiec does not teach away from or exclude the use of a mobile information appliance for a speech recognition system by its disclosure of at least using a personal digital assistant (or pen-based personal computer) for handwriting recognition. Because as taught in Cheyer that "Natural language content can be entered through different input modalities, including typing, handwriting, and speech" (see, for example, ¶ 145 above), it would have been obvious to a POSITA to use the personal digital assistant (or pen-based personal computer) disclosed in Kupiec as "a portable computing device configured to receive spoken input" (or "mobile information appliance" per its proper interpretation) as explicitly taught in Cheyer to be applicable to both handwriting and spoken input. Additionally, Kupiec already discloses the combination of spoken input via a portable voice input device (Sennheiser headset) and a workstation computer (SparcStation 10), thereby further illustrating the high likelihood of success for applying the approach put forth in Cheyer to the system of Kupiec.

237. Therefore, in my opinion, Kupiec in view of Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

1(a) receiving a spoken request for desired information from the user utilizing the mobile information appliance of the user, wherein said mobile information appliance comprises a portable remote control device or a set-top box for a television;

238. See ¶ 84 above regarding claim construction for this claim element.

239. In reference to FIG. 1, Kupiec describes that "Transducer **20** converts a *user's spoken utterance into a signal that can be processed* by processor **10**" and can comprise "*a microphone coupled to an analog-to-digital converter, so that the user's speech is converted by transducer* ***20*** *into a digital signal*" (emphasis added, see, for example, Ex. 1013 at 5:56-6:7).

240. Also, in reference to FIG. 2, Kupiec describes that "The *user inputs a question* ***201*** *into system* ***1*** *by speaking into audio transducer* ***20***" (emphasis added, see, for example, Ex. 1013 at 9:18-23). FIG. 2 of Kupiec and its detailed description explicitly illustrate that "question **201**" is not only "spoken" but is also a "request for desired information" to be found by "a series of queries" provided "to IR subsystem **40**, which executes them by conducting searches in accordance with queries **270** over corpus **41**" such that "The execution of the initial and any additional queries causes a set of documents **240** to be retrieved from corpus **41**" (see, for example, Ex. 1013 at 11:53-60, and see also, for example, ¶¶ 106-112 above).

241. Similarly, in reference to FIG. 11, Kupiec describes that "In operation, *transducer* ***220*** *accepts an input question* ***301*** and converts it into a signal **320**" wherein "*input question* ***301*** *can be a spoken utterance, in which case transducer* ***220*** *comprises audio signal processing equipment* that converts the spoken utterance to signal **320**" (emphasis added, see, for example, Ex. 1013 at 24:22-41).

242. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

243. Kupiec describes that "*Processor **10** is* a computer processing unit (CPU)" that typically is "*part of a* mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

244. Kupiec further discloses that the system "has been demonstrated on a Sun SparcStation 10 workstation" and that "*Discrete-word speech can be input using a Sennheiser HMD414 headset microphone* and a Rane MS-1 preamplifier, *with signal processing performed in software by the SparcStation*" so that such "Input speech is transcribed into a phone sequence using hidden Markov model methods" as exemplified in a prior art 1989 IEEE paper (emphasis added, see, for example, Ex. 1013 at 29:45-46, 30:48-56).

245. See also, for example, ¶¶ 94, 106, 114, 115, 125 and 129 above with respect to Kupiec and this claim element.

246. Thus, Kupiec discloses the recited method step of "receiving a spoken request" (at least when the microphone/transducer accepts an input question) that is "for desired information" (such as the material to be searched and retrieved from the IR subsystem) and that is "from the user" (at least when the input question is a user's spoken utterance), and further is "utilizing the mobile information appliance of the user" (at least when a personal computer such as a disclosed portable personal digital assistant would be configured to receive spoken input), wherein "said mobile information appliance comprises a portable remote control device or a set-top box for a television" (at least because in a speech recognition controlled system a portable headset microphone constitutes a portable remote control device).

247. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

248. However, to the extent that alleged non-disclosure of the antecedent “the mobile information appliance” of this claim element for Kupiec alone were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent “the mobile information appliance” of this claim element (see, for example, ¶¶ 231-237 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent “the mobile information appliance” (see, for example, ¶¶ 238-247 above).

249. In the section entitled “A Multimodal Map Application”, Cheyer describes “a prototype map-based application for a travel planning domain” wherein “the *system permits the user to* simultaneously combine direct manipulation, gestural drawings, handwritten, typed and *spoken natural language*” and this system uses “Existing commercial or research natural language and *speech recognition systems*” (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

250. In reference to Fig. 1, Cheyer discloses that “the *user*” is “*presented with a* pen sensitive map *display*” and can provide “*spoken input*” and the “*user may ask the map to perform various actions*” such as “*information retrieval*” (emphasis added, see, for example, Ex. 1019 at p. 5).

251. Cheyer further notes that “The application also makes use of *multimodal (multimedia) output* as well as input: *video, text, sound and voice can all be combined when presenting an answer to a query*” (emphasis added, see, for example, Ex. 1019 at p. 5).

252. Cheyer describes the system as also having a “*user interface*” that “*runs on pen-equipped PC's or a Dauphin handheld PDA*” using “either *a microphone or a telephone for*

voice input" (emphasis added, see, for example, Ex. 1019 at p. 5). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

253. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

254. Thus, Cheyer discloses the recited method step of "receiving a spoken request" (at least when the microphone or telephone accepts spoken or voice input) that is "for desired information" (such as for information retrieval) and that is "from the user" (at least when the spoken input is provided by a user), and further is "utilizing the mobile information appliance of the user" (at least when the handheld PDA receives the spoken input), wherein "said mobile information appliance comprises a portable remote control device or a set-top box for a television" (at least because the handheld PDA is itself a portable remote control device for controlling access to the information retrieval system).

255. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

256. Additionally, a POSITA would understand that Kupiec's disclosure of a workstation or a personal computer within the described system is essentially the same component as a "set-top box for a television". Moreover, a POSITA would be aware of prior art wherein a system that provides voice-driven navigation for information retrieval can connect to documents retrieved on the Internet using a "set-top box via cable" (see, for example, ¶¶ 193 and 200 above). Furthermore, a POSITA would understand that Cheyer's disclosure of "multimodal (multimedia) output" that includes "video" and "sound" indicates that the disclosed output "display" controlled by the handheld PDA can be a "television". Also, a POSITA would understand from Cheyer's disclosure of a handheld PDA that controls access to retrieving

information in remote databases such as server machines that such a handheld PDA acts as a portable remote control device for the system and further that this portable remote control device would be similarly applicable to the speech-recognition driven system of Kupiec to control access to Kupiec's remote information retrieval subsystem. In my opinion, a POSITA would be specifically motivated to combine the handheld PDA as a portable remote control device in Cheyer with the remote information retrieval subsystem of Kupiec at least because such a combination was obvious to try given Kupiec's own disclosure of a personal digital assistant and such a combination would lead to a predictable and successful result.

257. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, because the additional limitations of this claim element were well known to a POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein.

258. Kimura "relates to a *remote control system for remotely controlling various electronic devices*, and *more particularly to a remote control system for remotely controlling devices such as AV (audio visual) devices by way of voice commands*" including such "AV devices" as "*television receivers*" (emphasis added, see, for example, Ex. 1015 at 1:8-14). Additionally, Kimura is directed to a "*voice-operated remote control system which can vary a speech recognition process depending* on the degree of importance of a control command" in view of the fact that "the magnitudes of *effects caused by erroneous recognition*, may not necessarily be the same" (emphasis added, see, for example, Ex. 1015 at 1:41-42, 1:54-57).

259. Kimura discloses a "general *remote control system*" that "comprises a *transmitter* ***101*** *for transmitting a remote control signal from a position remote from a controlled device* ***103***

such as an AV device, and a receiver **102** for receiving the transmitted remote control signal, decoding the remote control signal, and sending the decoded information to the controlled device **103**" (emphasis added, see, for example, Ex. 1015 at 3:10-15).

260. More specifically, Kimura discloses that "FIG. 3 is a block diagram of the transmitter of a general voice-operated remote control system", wherein the "*transmitter **101** has a microphone **M** for converting a voice command* into an electric signal" that "is *applied to a speech recognition circuit **15*** in the form of a speech recognition LSI circuit or the like *which includes a microprocessor*" and "*produces command data corresponding to the recognized contents*", and further wherein "The transmitter **101** also has a controller **16** comprising a microprocessor" (emphasis added, see, for example, Ex. 1015 at 3:27-36).

261. Kimura describes in reference to FIG. 4 that "*transmitter **10A** of the voice-operated remote control system* has a unitary casing **11** which *allows the operator to carry the transmitter freely around*" (emphasis added, see, for example, Ex. 1015 at 4:9-12). Kimura further describes that "casing **11** supports a microphone **M** on an upper panel thereof" wherein "microphone **M** converts a voice command given by the operator into an electric signal" and on "one side of the casing **11**, there is disposed a *voice input switch* (hereinafter referred to as a "talk switch") **12** which is closed when pressed and can automatically be released or opened when released" so that when a "voice command is to be entered, the talk switch **12** is closed to operate the transmitter **10A**" or "Otherwise, the talk switch **12** is open keeping the transmitter **10A** out of operation" (emphasis added, see, for example, Ex. 1015 at 4:12-28).

262. Kimura describes in reference to FIG. 7 that "the *speech recognition circuit **15A** comprises* an analog processor **21** for processing an analog voice command signal which is received through the microphone **M** and outputting the processed analog voice command signal

as a time-division digital data **20**, *a speech recognition processor* ***22*** *for recognizing the voice command* based on the time-division digital data **20** from the analog processor **21**, a memory **23A** for storing standard pattern data for speech recognition, and an interface **24** for transmitting signals to and receiving signals from the controller **16A**" (emphasis added, see, for example, Ex. 1015 at 5:3-18).

263. Additionally, Kimura describes in reference to FIG. 8 that "the *speech recognition processor* ***22*** *comprises a system controller* ***40*** for analyzing and processing control commands from the controller **16** and also for controlling the entire operation of the speech recognition processor **22**, and a digital processor **41** for effecting distance calculations and controlling the memory **23A**" wherein the "*system controller* ***40*** *comprises a CPU (Central Processing Unit)* ***42*** *for controlling the overall operation of the transmitter* **1**" (emphasis added, see, for example, Ex. 1015 at 6:21-30).

264. See also, for example, ¶¶ 178, 180, 184, 187 and 189 above with respect to Kimura and this claim element.

265. Thus, Kimura discloses "a portable remote control device" that is "for a television" (such as the portable transmitter with a CPU that provides a remote control signal to a television receiver) and is part of a system for "receiving a spoken request" (at least because the transmitter includes a microphone and a speech recognition processor).

266. Although Kupiec in view of Cheyer renders obvious the limitations of this claim element per my analysis herein under the proper interpretation, to the extent that the limitation of "said mobile information appliance comprises a portable remote control device or a set-top box for a television" were construed to mean "said mobile information appliance comprises a *portable remote control device* or a set-top box *for a television*", as opposed to a "*portable*

remote control device" that can be for purposes that may not necessarily be "*for a television*", then in my opinion, a POSITA would have modified the system of Kupiec in view Cheyer further in view of the system of Kimura specifically to arrive at a combination that meets the limitations of this claim element under this alternative claim construction (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

267. First, Kupiec and Cheyer are both addressing the same basic problem of building a system for retrieval of information from remote electronic sources based upon an initial user inquiry made via spoken language that upon transcription is prone to errors and/or ambiguities (see, for example, ¶¶ 90, 91, 144 and 173 above), and Kimura is addressing a related problem of using spoken language input that upon transcription is prone to errors for the control of AV devices such as television receivers (see, for example, ¶ 176 above).

268. Second, Kupiec and Cheyer each describe systems with substantial similarities to each other, as well as this claim, and Kimura also describes a system with similarities regarding the use of portable handheld devices that use voice input and speech recognition, as evident from my analysis herein. For example, Kupiec and Cheyer individually and combined disclose at least a system that can access information at remote and/or local databases using either or both of spoken and non-spoken input modalities as described extensively herein. Analogously, Kimura discloses a system that enables access to local AV content and/or devices using a spoken input modality but with a portable remote control device that has additional capabilities such as a "pushbutton switch" that a POSITA would understand could be used for non-spoken user input in any system that can make use of such non-spoken user input (see, for example, ¶¶ 183-185 above, or Ex. 1015 at 4:23-24). Thus, it would have been obvious to a POSITA at the time of the alleged invention to apply the portable remote control device of Kimura to modify the

capabilities of the combined spoken and non-spoken input modality system of Kupiec and/or Cheyer such that input device options include the use of a portable remote control device that is specifically "for a television" as disclosed in Kimura. Additionally, Kimura explicitly discloses the additional limitations of this claim element under the alternative claim construction discussed above.

269. Third, Kupiec, Cheyer and Kimura each rely upon speech transcription techniques that are susceptable to transcription errors for which convenient user input via a portable interface device is likely to be needed (see, for example, ¶¶ 131, 165 and 183 above).

270. Fourth, Kupiec and Cheyer both arrive at the conclusion that information retrieval from remote electronic sources based upon an initial user inquiry in a natural language modality can benefit from usage of handheld personal digital assistants, and Kimura illustrates that a portable handheld device with a CPU and speech recognition can be used for a television application (see, for example, ¶¶ 91, 154 and 183 above).

271. Fifth, neither of Kupiec or Cheyer teaches away from or excludes the use of a mobile information appliance for a speech recognition system that comprises a "*portable remote control device*" specifically "*for a television*" by its disclosure of at least using a personal digital assistant (or pen-based personal computer) for handwriting recognition. Because as taught in Kimura that a handheld "voice-operated remote control system" can be used with a CPU and speech recognition to control AV devices such as television receivers, it would have been obvious to a POSITA to use the personal digital assistant (or pen-based personal computer) disclosed in Kupiec and/or Cheyer as a "mobile information appliance" that "comprises a *portable remote control device* or a set-top box *for a television*" as explicitly taught in Kimura. Additionally, Kupiec and Cheyer each already discloses the combination of spoken input and a

portable voice input device, thereby further illustrating the high likelihood of success for applying the approach put forth in Kimura to the system of either Kupiec and/or Cheyer.

272. Therefore, in my opinion, Kupiec in view of Cheyer further in view of Kimura discloses the limitations of this claim element for the alternative construction of this claim element as described herein.

1(b) rendering an interpretation of the spoken request;

273. In reference to FIG. 2, Kupiec describes that "*signal **220** produced by transducer **20** is fed to transcriber **50**, where it is converted into a phonetic transcription **250***" using "any of a variety of transcription techniques" that were "well-known among those of skill in the art" (emphasis added, see, for example, Ex. 1013 at 9:18-23). Kupiec explains that "*phonetic transcription **250** is an ordered sequence* of phones, that is, *of component sounds that can be used to form words*" (emphasis added, see, for example, Ex. 1013 at 9:29-31).

274. Also, in reference to FIG. 3, Kupiec describes that "*First the system accepts a user utterance* as input (Step A)" and that "This *utterance is converted to a signal (Step B) that is transcribed into a sequence of phones* (Step C)" (emphasis added, see, for example, Ex. 1013 at 12:48-57).

275. Similarly, in reference to FIG. 11, Kupiec describes that "Transducer **220** provides signal **320** to transcriber **250**" such that "*Transcriber **250** converts signal **320** to a string **350** that represents a transcription of the input question **301***" (emphasis added, see, for example, Ex. 1013 at 24:66-25:4).

276. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

277. Thus, Kupiec discloses the recited method step of "rendering an interpretation" (at least when the transcriber produces a phonetic transcription of component sounds that can be

used to form words) that is "of the spoken request" (at least because the transcriber operates upon the user's spoken utterance).

278. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

279. However, to the extent that alleged non-disclosure of the antecedent "the spoken request" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the spoken request" of this claim element (see, for example, ¶ 248 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the spoken request" (see, for example, ¶¶ 273-278 above).

280. In the section entitled "A Multimodal Map Application", Cheyer describes "a prototype map-based application for a travel planning domain" wherein "the *system permits the user to* simultaneously combine direct manipulation, gestural drawings, handwritten, typed and *spoken natural language*" and this system uses "*Existing commercial* or research natural language and *speech recognition systems*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

281. Cheyer also specifically describes, from a 1990 prior art article, a "*Speech Recognition (SR) Agent*" that "provides a mapping from the Interagent Communication Language to the API *for the Decipher (Corona) speech recognition system*", which Cheyer describes as "*a continuous speech speaker independent recognizer based on Hidden Markov Model technology*" (emphasis added, see, for example, Ex. 1019 at p. 8). Cheyer further

specifically describes, with reference to a 1994 prior art article, a "*Natural Language (NL) Parser Agent*" that "*translates English expressions into the Interagent Communication Language (ICL)*" (emphasis added, see, for example, Ex. 1019 at p. 8).

282. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

283. Thus, Cheyer discloses the recited method step of "rendering an interpretation" (at least when speech recognition system and parser creates natural language English expressions) that is "of the spoken request" (at least because the speech recognition system and parser operates upon the user's spoken natural language).

284. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

285. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

1(c) constructing a navigation query based upon the interpretation;

286. See ¶ 83 above regarding claim construction for this claim element.

287. In reference to FIG. 2, Kupiec describes that "The *phonetic transcription **250** is provided to hypothesis generator **60*** where it is matched using phonetic index **62** *to generate a set of hypotheses **260***" (emphasis added, see, for example, Ex. 1013 at 9:38-61) and further that "Once the set of *hypotheses **260*** has been generated, it is *provided to query constructor **70***" such that "Query constructor **70** uses the hypotheses **260** *to construct one or more queries **270** that*

*will be sent to IR subsystem **40** for execution*" (emphasis added, see, for example, Ex. 1013 at 11:10-13).

288. Additionally, Kupiec specifically discloses that "Queries **270** are Boolean queries with proximity and order constraints" and provides a specific example wherein the "user speaks two words" that lead to a "set of hypotheses" such "an initial *query*" is constructed that "*seeks occurrences of* at least one of the words (*search terms*)" and is "*sent to the IR subsystem **40*** where it is executed" (emphasis added, see, for example, Ex. 1013 at 11:13-41).

289. Also, in reference to FIG. 3, Kupiec describes that "First the system accepts a user utterance" that is "transcribed" and then "*used to generate hypotheses* (Step D)" such that "*Boolean queries with proximity and order constraints are constructed based on the hypotheses and are executed to retrieve documents* (Step E)" (emphasis added, see, for example, Ex. 1013 at 12:48-57).

290. Similarly, in reference to FIG. 11, Kupiec describes that "Transducer **220** provides signal **320** to transcriber **250**" that "converts signal **320** to a string **350** that represents a transcription of the input question **301**" such that "Hypothesis generator **260** converts string **350** and any alternatives to a set of hypotheses **360**" provided to "*Query/IR mechanism* **270**" that "*converts the hypotheses **360** to one or more information retrieval queries* **370**" that are "in a format *that can be searched* by processor **200** (or a separate IR processor that communicates with processor **200**) using corpus **241**" (emphasis added, see, for example, Ex. 1013 at 24:66-25:61).

291. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

292. Thus, Kupiec discloses the recited method step of "constructing a navigation query" (at least when the query constructor constructs one or more queries that can be used to search the IR subsystem) that is "based upon the interpretation" (at least because the query constructor operates upon the hypotheses formed from the transcription of the user's spoken utterance).

293. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

294. However, to the extent that alleged non-disclosure of the antecedent "the interpretation" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the interpretation" of this claim element (see, for example, ¶ 279 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the interpretation" (see, for example, ¶¶ 286-293 above).

295. In reference to Figure 1, Cheyer discloses that "the *user*" can provide "*spoken input*" and the "*user may ask the map to perform various actions*" such as "*information retrieval*" (emphasis added, see, for example, Ex. 1019 at p. 5).

296. Cheyer further describes that the system can "capture signals emitted during a user's interaction" and also "integrates a set of modality agents, each responsible for a very specialized kind of signal" wherein the "modality agents are connected to an '*interpret agent*' which is *responsible for combining the inputs across all modalities to form a valid command* for the application" (emphasis added, see, for example, Ex. 1019 at p. 7).

297. Cheyer discloses "Database Agents" wherein exemplary "*databases*" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning *HTML pages from the World Wide Web (WWW)*" as well as "information" that is "extracted by an HTML reading database agent" (emphasis added, see, for example, Ex. 1019 at p. 8).

298. Cheyer also discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*" wherein this "reference resolution agent (RR)" in an exemplary embodiment "sends database requests asking for the coordinates of the items in question" (emphasis added, see, for example, Ex. 1019 at pp. 8-10).

299. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

300. Thus, Cheyer discloses the recited method step of "constructing a navigation query" (at least when the reference resolution agent sends database requests for data on the World Wide Web) that is "based upon the interpretation" (at least because the reference resolution agent creates database requests from the output of agents that respond to spoken input).

301. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

302. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

1(d) utilizing the navigation query to select a portion of the electronic data source;

303. See ¶ 83 above regarding claim construction for this claim element.

304. Kupiec describes that "Depending on the results obtained from execution of the initial query, *additional queries can be constructed and executed, in a process called query reformulation*" in order to "*send the query thus modified back to IR subsystem **40** to be executed again*" (emphasis added, see, for example, Ex. 1013 at 11:42-48).

305. Kupiec also states that "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "*The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query*" (emphasis added, see, for example, Ex. 1013 at 15:1-12).

306. Kupiec discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search* based on the retrieved results" (emphasis added, see, for example, Ex. 1013 at 19:32-36).

307. Kupiec further discloses that "*IR subsystem **40*** incorporates a processor that *can process queries to search for documents in corpus **41***" (emphasis added, see, for example, Ex. 1013 at 6:22-25). According to Kupiec, "a series of queries **270** is constructed by query constructor **70** and provided to IR subsystem **40**, which executes them by *conducting searches in accordance with queries* **270** over corpus **41**" such that "The execution of the initial and any additional *queries causes a set of documents **240** to be retrieved from corpus **41***" (emphasis added, see, for example, Ex. 1013 at 11:53-60).

308. Kupiec discloses in reference to FIG. 9 that "A relevant subset of the hypotheses and retrieved documents is presented to the user (Step KK). If documents have previously been

retrieved, then user relevance feedback commands and search terms can be routed to the hypothesis generator, to instruct the hypothesis generator to use retrieved document titles as the basis for confirming hypotheses (Step LL), or to cease doing this upon a "new search" or similar command. The *system then can perform operations such as a vector space search or the selection of one among several preferred hypotheses* (Step MM). Results of these operations are presented to the user (Step KK)" (emphasis added, see, for example, Ex. 1013 at 20:21-32).

309. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

310. Thus, Kupiec discloses the recited method step of "utilizing the navigation query" (at least by searching the information retrieval subsystem or a subset thereof with an initial or modified query) specifically "to select a portion of the electronic data source" (at least when the documents from the information retrieval subsystem search are selected via a preferred hypothesis based on the query).

311. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

312. However, to the extent that alleged non-disclosure of the antecedent "the navigation query" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the navigation query" of this claim element (see, for example, ¶ 294 above) and Kupiec discloses the subsequent additional limitations recited by this

claim element in view of the antecedent "the navigation query" (see, for example, ¶¶ 303-311 above).

313. According to Cheyer, "*Database Agents*" can "*reside at local or remote locations and can be grouped hierarchically according to content*" wherein such "databases" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the World Wide Web (WWW)" as well as "information" that is "extracted by an HTML reading database agent" such as a "list of current movie times and reviews" (emphasis added, see, for example, Ex. 1019 at p. 8).

314. Cheyer discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*" (emphasis added, see, for example, Ex. 1019 at pp. 8-9).

315. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

316. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

317. Thus, Cheyer discloses the recited method step of "utilizing the navigation query" (at least by sending database requests after user resolution of an ambiguous reference)

specifically "to select a portion of the electronic data source" (at least when such refined database requests return the selected information about the items in question).

318. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

319. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

1(e) transmitting the selected portion of the electronic data source from the network server to the mobile information appliance of the user.

320. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor* **10** *via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

321. Kupiec discloses that "*Display* **30** *provides visual output to the user*" such as for "*documents retrieved from corpus* **41**" and typically comprises "*a computer screen* or monitor", and further that "*Corpus* **41** *comprises a database of documents* that can be searched by IR subsystem **40**" wherein such documents comprise "for example, *books, articles from newspapers and periodicals, encyclopedia articles, abstracts, office documents*, etc." (emphasis added, see, for example, Ex. 1013 at 6:12-15, 29-33). Similarly, Kupiec additionally discloses that "*Output channel* **230** *can send interpretation* **400** *to be displayed using a visual display* **231**" in order "to facilitate the understanding of the inputs that the *user provides as relevance feedback*" (emphasis added, see, for example, Ex. 1013 at 27:7-18).

322. In reference to FIG. 3 of Kupiec, "A *relevant subset of the* hypotheses and *retrieved documents is presented to the user* (Step G)" (emphasis added, see, for example, Ex. 1013 at 12:48-57). Similarly, in reference to FIG. 9 of Kupiec, "The *system* then *can perform operations such as a vector space search or the selection of one among several preferred hypotheses* (Step MM)" such that "*Results of these operations are presented to the user* (Step KK)" (emphasis added, see, for example, Ex. 1013 at 20:21-32).

323. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

324. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

325. Thus, Kupiec discloses the recited method step of "transmitting the selected portion of the electronic data source from the network server to the mobile information appliance of the user" (at least by retrieving documents from an information retrieval subsystem at a remote site over a communications network and displaying the selected documents on a computer screen such as that of a personal digital assistant).

326. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

327. However, to the extent that alleged non-disclosure of the antecedent "the selected portion" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element or alleged non-disclosure of the antecedent "the mobile information appliance" were considered to constitute non-disclosure of this claim element by Kupiec, then

Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the selected portion" and "the mobile information appliance" of this claim element (see, for example, ¶¶ 231-237 above and ¶ 312 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the selected portion" and "the mobile information appliance" (see, for example, ¶¶ 320-326 above).

328. Cheyer is "distinguished by a synergistic combination of handwriting, gesture and speech modalities; *access to existing data sources including the World Wide Web*; and a *mobile handheld interface*" (emphasis added, see, for example, Ex. 1019 at p. 2). Cheyer further describes the system as enabling "Through the multimodal interface" that "a user" can "*transparently access a wide variety of data sources, including information stored in HTML form on the World Wide Web*" and Cheyer describes the system as also having a "user interface" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" (emphasis added, see, for example, Ex. 1019 at p. 5). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

329. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" and then subsequently "*requests the user interface to produce output displaying the result* of the calculation" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

330. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

331. Thus, Cheyer discloses the recited method step of "transmitting the selected portion of the electronic data source from the network server to the mobile information appliance of the user" (at least by retrieving information in databases on the World Wide Web that gets displayed on the user interface of mobile pen-equipped personal computers such as the Dauphin handheld PDA).

332. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

333. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

<u>'718 Patent: Claim 2</u>

> 2. The method of claim 1, wherein the step of rendering the interpretation of the spoken request is performed by the mobile information appliance.

2. The method of claim 1, wherein the step of rendering the interpretation of the spoken request is performed by the mobile information appliance.

334. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 1 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 214-333 above.

335. Kupiec specifically discloses that "<u>*IR subsystem* ***40*** *can be located at the same site as processor* ***10***</u> or can be located at a remote site and connected to processor **10** via a suitable communication network" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

336. According to Kupiec, “*System* ***1*** *comprises a processor* ***10*** coupled to an input audio transducer **20**, an output visual display **30**, an optional output speech synthesizer **31**, and an information retrieval (IR) subsystem **40** which accesses documents from corpus **41** using a word index **42**” as well as “*a phonetic transcriber* ***50***, *a hypothesis generator* ***60***, a phonetic index **62**, a *query constructor* ***70***, and a scoring mechanism **80**” (emphasis added, see, for example, Ex. 1013 at 5:43-51).

337. Kupiec describes that “*Processor* ***10*** *is a computer processing unit (CPU)*” that typically is “*part of a* mainframe, *workstation, or personal computer*” but can comprise “multiple processing elements in some embodiments” (emphasis added, see, for example, Ex. 1013 at 5:52-55).

338. Kupiec notes that “*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*” (emphasis added, see, for example, Ex. 1013 at 1:56-67).

339. Kupiec further discloses that the system “has been *demonstrated on a Sun SparcStation 10 workstation*” and that “Discrete-word speech can be input using a Sennheiser HMD414 headset microphone and a Rane MS-1 preamplifier, *with signal processing performed in software by the SparcStation*” so that such “*Input speech is transcribed* into a phone sequence using hidden Markov model methods” as exemplified in a prior art 1989 IEEE paper (emphasis added, see, for example, Ex. 1013 at 29:45-46, 30:48-56). In my opinion, a POSITA would understand such a SparcStation 10 workstation with a headset microphone discrete-word speech input at the time of the alleged invention to be an example of a computing device that can be located locally with the user, and therefore would inherently disclose the limitation that such

rendering be performed at “the mobile information appliance” to the extent that “the mobile information appliance” is the computing device, such as a personal computer or a personal digital assistant, that is located locally with the user as in the case of Kupiec or Kupiec in view of Cheyer as discussed herein.

340. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

341. Thus, Kupiec discloses the “method of claim 1” (as described herein), wherein “the step of rendering the interpretation of the spoken request is performed by the mobile information appliance” (at least when the phonetic transcriber is implemented on a workstation or a portable personal computer or personal digital assistant with a headset microphone input that would normally be located locally with the user).

342. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

343. However, to the extent that alleged non-disclosure of the antecedent “method of claim 1” of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element or alleged non-disclosure of the antecedent “the mobile information appliance” were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent “method of claim 1” and “the mobile information appliance” of this claim element (see, for example, ¶¶ 214-333 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent “method of claim 1” and “the mobile information appliance” (see, for example, ¶¶ 334-342 above).

344. Cheyer describes its system as implemented with an "*Open Agent Architecture (OAA)*" that "*provides a framework for coordinating a society of agents which interact to solve problems for the user*" and "*provides distributed access to* commercial applications, such as mail systems, calendar programs, *databases*, etc." (emphasis added, see, for example, Ex. 1019 at p. 6).

345. Cheyer also describes that "In the *Open Agent Architecture, agents are distributed entities that can run on different machines*, and communicate together to solve a task for the user" (see, for example, Ex. 1019 at p. 8). More specifically Cheyer discloses "Macro Agents", which "contain some knowledge and ability to reason about a domain, *and can answer or make queries to other macro agents using the Interagent Communication Language*", and "Micro Agents", which "are responsible for handling a single input or output data stream, either filtering the signal to or from a hierarchically superior 'interpret' agent" (emphasis added, see, for example, Ex. 1019 at p. 8).

346. According to Cheyer, the "network architecture" used was "hierarchical at two resolutions" wherein "micro agents are connected to a superior macro agent" and "macro agents are connected in turn to a facilitator agent" but "In both cases, *a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

347. Cheyer also specifically describes a "*Speech Recognition (SR) Agent*" that "provides a mapping from the Interagent Communication Language to the API for the Decipher (Corona) speech recognition system", which Cheyer describes as "a continuous speech speaker independent recognizer based on Hidden Markov Model technology" (emphasis added, see, for example, Ex. 1019 at p. 8). Cheyer further specifically describes a "Natural Language (NL)

Parser Agent" that "translates English expressions into the Interagent Communication Language (ICL)" (emphasis added, see, for example, Ex. 1019 at p. 8).

348. Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either *a microphone or a telephone for voice input*" (emphasis added, see, for example, Ex. 1019 at p. 5). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

349. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

350. Thus, Cheyer discloses the "method of claim 1" (as described herein), wherein "the step of rendering the interpretation of the spoken request is performed by a **computing device**" (at least when the speech recognition, parser and database agents are implemented on a server or different machines) but not specifically where such steps are performed by a "**computing device**" that is "*the mobile information appliance*". However, Cheyer does not teach away from or preclude such implementation on a "**computing device**" that is "*the mobile information appliance*" as disclosed by Cheyer, and thereby does not provide any motivation not to combine Cheyer with Kupiec with respect to this claim element, or otherwise.

351. Additionally, a POSITA at the time of the alleged invention would have known that for systems such as Kupiec, Cheyer, or Kupiec in view of Cheyer (as described herein), that the step of "rendering an interpretation" could be performed "*by the mobile information appliance*". For example, the '021 Patent admits that "It will be *apparent to those skilled in the art* that additional implementations, permutations and combinations of the embodiments set forth in FIGS. 1a, 1b, and 2 may be created" and that "*practitioners will understand*" that "*it is possible to divide and allocate the functional components of request processing logic* ***300***

between client and server" such as "*speech recognition*—in entirety, or perhaps just early stages such as feature extraction—*might be performed locally on the client end*, perhaps to reduce bandwidth requirements" (emphasis added, see, for example, Ex. 1001 at 6:35-46). Moreover, a POSITA would understand that for systems such as Kupiec, Cheyer, or Kupiec in view of Cheyer (as described herein), that the only logical choices for performing such step of "rendering an interpretation" would be either "*on a computing device located locally with the user*" or "*on a network computing device located remotely from the user*". Thus, to the extent that Kupiec, Cheyer, or Kupiec in view of Cheyer (as described herein) were considered to not disclose the additional limitations of this claim element with respect to performing recited steps "*by the mobile information appliance*" (i.e. "*on a computing device located locally with the user*"), in my opinion, a POSITA at the time of the alleged invention would already be knowledgeable regarding such a topic and would find it obvious to try such an approach in view of the finite number of possibilities (effectively only two) and the high predictability that such an approach would lead to a successful outcome. Furthermore, a POSITA would have known from the prior art that systems that provide voice-driven input to electronic systems that the step of "rendering an interpretation" could be performed "*by the mobile information appliance*" (see, for example, ¶¶ 179-181 above).

352. At least because Kupiec discloses the additional limitations of this claim element, and Cheyer discloses some of the additional limitations of this claim element, because the additional limitations of this claim element were well known to a POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this

claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

353. Kimura "relates to a *remote control system for remotely controlling various electronic devices*, and *more particularly to a remote control system for remotely controlling devices such as AV (audio visual) devices by way of voice commands*" including such "AV devices" as "*television receivers*" (emphasis added, see, for example, Ex. 1015 at 1:8-14).

354. Kimura discloses a "general *remote control system*" that "comprises a *transmitter **101** for transmitting a remote control signal from a position remote from a controlled device **103** such as an AV device*, and a receiver **102** for receiving the transmitted remote control signal, decoding the remote control signal, and sending the decoded information to the controlled device **103**" (emphasis added, see, for example, Ex. 1015 at 3:10-15).

355. More specifically, Kimura discloses that "FIG. 3 is a block diagram of the transmitter of a general voice-operated remote control system", wherein the "*transmitter **101** has a microphone **M** for converting a voice command* into an electric signal" that "is *applied to a speech recognition circuit **15** in the form of a speech recognition LSI circuit* or the like which includes a microprocessor" and "*produces command data corresponding to the recognized contents*", and further wherein "The transmitter **101** also has a controller **16** comprising a microprocessor" (emphasis added, see, for example, Ex. 1015 at 3:27-36).

356. Kimura describes in reference to FIG. 4 that "*transmitter **10A** of the voice-operated remote control system* has a unitary casing **11** which *allows the operator to carry the transmitter freely around*" (emphasis added, see, for example, Ex. 1015 at 4:9-12).

357. Kimura describes in reference to FIG. 7 that "the *speech recognition circuit **15A** comprises* an analog processor **21** for processing an analog voice command signal which is

received through the microphone **M** and outputting the processed analog voice command signal as a time-division digital data **20**, *a speech recognition processor **22** for recognizing the voice command* based on the time-division digital data **20** from the analog processor **21**, a memory **23A** for storing standard pattern data for speech recognition, and an interface **24** for transmitting signals to and receiving signals from the controller **16A**" (emphasis added, see, for example, Ex. 1015 at 5:3-18).

358. See also, for example, ¶¶ 178, 180, 184, 187 and 189 above with respect to Kimura and this claim element.

359. Thus, Kimura discloses a "step of rendering an interpretation of a spoken request" that is "performed by a mobile information appliance" (at least when the handheld portable transmitter with microphone uses a speech recognition processor within such transmitter to render an interpretation of a spoken request).

360. Although Kupiec in view of Cheyer renders obvious the limitations of this claim element per my analysis herein under the proper interpretation, to the extent that the disclosures of Kupiec in view of Cheyer regarding this claim element were considered to not include the recited step of this claim element, then in my opinion, a POSITA would have modified the system of Kupiec in view Cheyer further in view of the system of Kimura specifically to arrive at a combination that meets the limitations of this claim element under the proper interpretation (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

361. First, see ¶ 267 above.

362. Second, see ¶ 268 above. Additionally, Kimura explicitly discloses the limitation of this claim element regarding a "step of rendering an interpretation of a spoken request" that is "performed by a mobile information appliance" as discussed above.

363. Third, see ¶ 269 above.

364. Fourth, see ¶ 270 above.

365. Fifth, see ¶ 271 above. Additionally, neither of Kupiec or Cheyer teaches away from or excludes the use of a mobile information appliance that performs a "step of rendering an interpretation of a spoken request" within the mobile information appliance itself. Because as taught in Kimura that a handheld "voice-operated remote control system" can perform speech recognition within such a remote control device using a speech recognition processor, it would have been obvious to a POSITA to use the personal digital assistant (or pen-based personal computer) disclosed in Kupiec and/or Cheyer as a "mobile information appliance" wherein "the step of rendering the interpretation of the spoken request is performed by the mobile information appliance" as explicitly taught in Kimura. Additionally, Kupiec and Cheyer each already discloses the combination of "rendering an interpretation of a spoken request" and a "mobile information appliance" with local computing capabilities, thereby further illustrating the high likelihood of success for applying the approach put forth in Kimura to the system of either Kupiec and/or Cheyer.

366. Therefore, in my opinion, Kupiec in view of Cheyer further in view of Kimura discloses the limitations of this claim element for the alternative construction of this claim element as described herein.

<u>'718 Patent: Claim 3</u>

> 3. The method of claim 1, wherein the step of rendering the interpretation of the spoken request is performed by the mobile information appliance.

3. The method of claim 1, wherein the step of rendering the interpretation of the spoken request is performed by the mobile information appliance.

367. See ¶¶ 334-366 above, which apply here at least because this Claim 3 is identical to Claim 2.

<u>'718 Patent: Claim 4</u>

> 4. The method of claim 1, further comprising the steps of soliciting additional input from the user, including user interaction in a modality different than the original request; refining the navigation query, based upon the additional input; and using the refined navigation query to select a portion of the electronic data source.

4. The method of claim 1, further comprising the steps of

368. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 1 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 214-333 above.

4(a) soliciting additional input from the user, including user interaction in a modality different than the original request;

369. With respect to FIG. 2, Kupiec discloses that "*transcriber **50** is error-prone and produces a phonetic transcription **250** that is imperfect*" (emphasis added, see, for example, Ex. 1013 at 9:35-37) and that "Because the transcriber **50** is known to be error-prone, *hypothesis generator **60** develops alternative possible transcriptions for each word spoken by the user*, in addition to the original phone sequences provided by transcriber **50**" (emphasis added, see, for example, Ex. 1013 at 9:38-61), thereby occasionally causing "*hypothesis generator **60***" to "halt processing of the user's question and *prompt the user to repeat the question*" (emphasis added, see, for example, Ex. 1013 at 11:1-9).

370. Kupiec also discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search based on the retrieved results*" (emphasis added, see, for example, Ex. 1013 at 19:32-36). Thus, Kupiec notes that "the *best matching documents that correspond at any time* to the words that the user has spoken so far *can be displayed to the user on a screen*" such that "Upon seeing the titles (or other descriptive content) the *user can speak additional words to direct the search to particular documents or cause them to be excluded* by invoking the NOT operation" (emphasis added, see, for example, Ex. 1013 at 19:64-20:2). See also ¶ 121 above.

371. Kupiec discloses that "*Section 8* concerns an embodiment in which the *input can take forms besides speech*" (emphasis added, see, for example, Ex. 1013 at 20:42-44), and describes "Section 8" in reference to FIG. 11 that "illustrates a specific embodiment of the invention that is *adaptable to a range of input sources*, transcription techniques, hypothesis generation techniques, information retrieval techniques, and analysis techniques" (emphasis added, see, for example, Ex. 1013 at 23:19-25).

372. Kupiec also discloses in reference to FIG. 11 that "In operation, *transducer **220** accepts an input question **301*** and converts it into a signal **320**" wherein "*input question **301** can be a spoken utterance*, in which case *transducer **220** comprises audio signal processing equipment* that converts the spoken utterance to signal **320**" as well as other input modalities such as "*handwritten*" or "*typewritten*" wherein "*transducer **220** comprises a digitizing tablet* or input-sensitive display screen as is typical of pen-based computers" or "*transducer **220** comprises a conventional computer keyboard*" (emphasis added, see, for example, Ex. 1013 at 24:22-41).

373. With respect to FIG. 11, Kupiec additionally discloses that "Output channel **230** can send interpretation **400** to be displayed using a visual display **231**" such that "the *user can provide relevance feedback based on displayed or speech-synthesized output*" in order "to facilitate the *understanding of the inputs that the user provides as relevance feedback*" (emphasis added, see, for example, Ex. 1013 at 27:7-18).

374. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

375. Thus, Kupiec discloses the recited method step of "soliciting additional input from the user" (at least when the system prompts a user to repeat a question or when the system

accepts additional words provided as relevance feedback to direct a search) that is "including user interaction in a **non-spoken** modality" (at least because the system accepts user inputs in the form of handwritten or typewritten modalities).

376. Kupiec does not explicitly disclose the totality of this claim element because Kupiec's disclosed step of "soliciting additional input from the user, including user interaction in a **non-spoken** modality" does not necessarily require that such "**non-spoken** modality" be "*<u>different</u>* than the original request" as recited by this claim limitation. However, Kupiec does not teach away or exclude the case wherein Kupiec's disclosed step of "soliciting additional input from the user, including user interaction in a **non-spoken** modality" would be "*<u>different</u>* than the original request". Instead, Kupiec is silent with respect to this particular limitation within this claim element, and thereby does not provide any motivation not to combine Cheyer with Kupiec with respect to this claim element, or otherwise.

377. Cheyer states that "*<u>direct manipulation and natural language</u>* seem to be very *<u>complementary modalities</u>*" and further that "It is therefore not surprising that a number of multimodal systems combine the two" including Cheyer's description that "A number of systems have focused on *<u>combining the speed of speech with</u>* the reference provided by *<u>direct manipulation of a mouse pointer</u>*" (emphasis added, see, for example, Ex. 1019 at pp. 3-4).

378. Cheyer describes a "*<u>system</u>*" that "permits the user to *<u>simultaneously combine direct manipulation</u>*, gestural drawings, handwritten, typed *<u>and spoken natural language</u>*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

379. In reference to Figure 1, Cheyer discloses that "the user is presented with a *<u>pen sensitive map display on which drawn gestures</u>* and written natural language statements *<u>may be combined with spoken input</u>*" such that "content presented by the map change, according to the

requests of the user" and the "*user may ask the map to perform various actions*" (emphasis added, see, for example, Ex. 1019 at p. 5).

380. According to Cheyer, this "system" has "*modality agents*" that are "*connected to an 'interpret agent'* which is responsible for *combining the inputs across all modalities* to form a valid command for the application" wherein this "interpret agent receives filtered results from the modality agents, sorts the information into the correct fields, performs type-checking on the arguments, *and prompts the user for any missing information*" (emphasis added, see, for example, Ex. 1019 at p. 7).

381. Cheyer further discloses an "*Interface Agent*" as "responsible for managing what is currently being displayed to the user, and *for accepting the user's multimodal input*" wherein this "Interface Agent also coordinates client modality agents and resolves ambiguities among them" such that "*handwriting and gestures are* interpreted locally by micro agents and *combined with results from the speech recognition agent*, running on a remote speech server" (emphasis added, see, for example, Ex. 1019 at p. 9).

382. Additionally, Cheyer notes that "An important task for the *interface agent* is to record which objects of each type are currently salient, in order *to resolve contextual references* such as "the hotel" or "where I was before"" wherein such "Deictic references are resolved *by gestural or direct manipulation commands*" and further wherein "*If no such indication is currently specified, the user interface agent* waits long enough to give the user an opportunity to supply the value, and then *prompts the user for it*" (emphasis added, see, for example, Ex. 1019 at p. 9).

383. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent*

uses contextual structures to find what "the restaurant" refers to, and ***waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary***" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

384. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

385. Thus, Cheyer discloses the recited method step of "soliciting additional input from the user" (at least when the system prompts a user to provide missing information) that is "including user interaction in a modality different than the original request" (at least because the system prompts the user that had provided spoken input for additional information to be returned by non-spoken modalities such as handwriting, gestures or direct manipulation by mouse pointer or typing).

386. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

387. In my opinion, a POSITA would have modified the system of Kupiec in view of the system of Cheyer specifically to arrive at a combination that meets the limitations of this claim element for at least the following reasons.

388. First, see ¶ 232 above.

389. Second, see ¶ 233 above.

390. Third, Kupiec and Cheyer both rely upon the same fundamental speech transcription technology (the hidden Markov model) and thus both are susceptable to similar kinds of transcription errors for which additional input is likely to correct (see, for example, ¶¶ 131 and 165 above).

391. Fourth, Kupiec and Cheyer both arrive at the conclusion that information retrieval from remote electronic sources based upon an initial user inquiry made via spoken language can benefit from additional solicted user input subsequent to the initial spoken request in the form of relevance feedback in response to initial search results or in the form of resolution to contextual references or ambiguities (see, for example, ¶¶ 123 and 167-169 above). Kupiec provides examples using spoken "command words" or "keywords" for such additional input that include the use of "NOT" in order to exclude possibilities (see, for example, ¶ 122 above), while Cheyer provides examples using non-spoken "gestures" for such additional input that include the use of "Remove" in order to exclude possibilities (see, for example, ¶¶ 153 and 173 above). Since exemplary ones of Kupiec's spoken additional inputs (such as "NOT") are directly analogous to exemplary ones of Cheyer's non-spoken additional inputs (such as "Remove"), then a POSITA would have understood substitution of the spoken additional inputs of Kupiec that follow an original spoken input by non-spoken additional inputs of Cheyer that follow an original spoken input to be obvious to try and to produce predictable and successful results.

392. Fifth, Cheyer specifically teaches that for a system otherwise highly similar to the system of Kupiec, a solicitation for additional input specifically in a "non-spoken modality" that is "different than the original request" (in a spoken modality) is the preferred approach for resolving deficiencies in an original spoken request for retrieving information from remote databases (see, for example, ¶¶ 168 and 173 above).

393. Sixth, Cheyer also recites that this specific combination of spoken original input and non-spoken additional solicited input was well known in the prior art (see, for example, ¶¶ 146 and 161 above), thereby indicating to a POSITA that such a specific combination of input modalities would be highly likely to achieve a successful result.

394. Seventh, Kupiec is silent as to whether or not Kupiec's additional solicted user input subsequent to the initial spoken request should be provided in a "non-spoken modality" that is "different than the original request". However, because there exist only two basic possibilities ("different modality" or "same modality") for this issue upon which Kupiec is silent, then a POSITA would be highly motivated to try the recommended one of these two basic possibilities taught by Cheyer ("different modality"). Furthermore, given the specific teaching of Cheyer showing the success of this "different modality" approach as recited by this claim element in a system otherwise highly similar to the system of Kupiec, a POSITA would also view the likelihood of success in combining the "different modality" approach of Cheyer with the system of Kupiec as very high and very predictable.

395. Therefore, in my opinion, Kupiec in view of Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

4(b) refining the navigation query, based upon the additional input;

396. See ¶ 83 above regarding claim construction for this claim element.

397. Kupiec describes that "Depending on the results obtained from execution of the initial query, *additional queries can be constructed and executed, in a process called query reformulation*" in order to "send the query thus modified back to IR subsystem **40** to be executed again" (emphasis added, see, for example, Ex. 1013 at 11:42-48).

398. Kupiec also states that "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "*The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query*" (emphasis added, see, for example, Ex. 1013 at 15:1-12).

399. Kupiec discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search* based on the retrieved results" (emphasis added, see, for example, Ex. 1013 at 19:32-36). See also ¶ 121 above.

400. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

401. Thus, Kupiec discloses the recited method step of "refining the navigation query" (at least by the query reformulation process that is used for searching the information retrieval subsystem) wherein such step is "based upon the additional input" (at least when the user provides relevance feedback commands that cause the system to perform a follow-up search).

402. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

403. However, to the extent that non-disclosure of the specifically-limited antecedent "the additional input" of this claim element as discussed above for Kupiec were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the specifically-limited antecedent "the additional input" of this claim element (see, for example, ¶¶ 387-395 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the additional input" (see, for example, ¶¶ 396-402 above).

404. Cheyer discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between*

the user interface agent, database agents and modality agents" (emphasis added, see, for example, Ex. 1019 at pp. 8-9).

405. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

406. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

407. Thus, Cheyer discloses the recited method step of "refining the navigation query" (at least by the interactions of the reference resolution, user interface, database and modality agents as described herein) wherein such step is "based upon the additional input" (at least when the user provides additional input such as gesture following an original spoken request).

408. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

409. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

4(c) using the refined navigation query to select a portion of the electronic data source.

410. See ¶ 83 above regarding claim construction for this claim element.

411. Kupiec describes that "Depending on the results obtained from execution of the initial query, *additional queries can be constructed and executed, in a process called query reformulation*" in order to "*send the query thus modified back to IR subsystem **40** to be executed again*" (emphasis added, see, for example, Ex. 1013 at 11:42-48).

412. Kupiec also states that "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "*The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query*" (emphasis added, see, for example, Ex. 1013 at 15:1-12).

413. Kupiec discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search* based on the retrieved results" (emphasis added, see, for example, Ex. 1013 at 19:32-36). See also ¶ 121 above.

414. Kupiec further discloses that "*IR subsystem **40*** incorporates a processor that *can process queries to search for documents in corpus **41***" (emphasis added, see, for example, Ex. 1013 at 6:22-25). According to Kupiec, "a series of queries **270** is constructed by query constructor **70** and provided to IR subsystem **40**, which executes them by *conducting searches in accordance with queries* **270** over corpus **41**" such that "The execution of the initial and any additional *queries causes a set of documents **240** to be retrieved from corpus **41***" (emphasis added, see, for example, Ex. 1013 at 11:53-60).

415. Kupiec discloses in reference to FIG. 9 that "A relevant subset of the hypotheses and retrieved documents is presented to the user (Step KK). If documents have previously been

retrieved, then user relevance feedback commands and search terms can be routed to the hypothesis generator, to instruct the hypothesis generator to use retrieved document titles as the basis for confirming hypotheses (Step LL), or to cease doing this upon a "new search" or similar command. The *system then can perform operations such as a vector space search or the selection of one among several preferred hypotheses* (Step MM). Results of these operations are presented to the user (Step KK)" (emphasis added, see, for example, Ex. 1013 at 20:21-32).

416. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

417. Thus, Kupiec discloses the recited method step of "using the refined navigation query" (at least by searching the information retrieval subsystem or a subset thereof with the relevance feedback modified query) specifically "to select a portion of the electronic data source" (at least when the documents from the information retrieval subsystem search are selected via a preferred hypothesis based on the user's relevance feedback commands).

418. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

419. However, to the extent that alleged non-disclosure of the antecedent "the refined navigation query" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the refined navigation query" of this claim element (see, for example, ¶ 403 above) and Kupiec discloses the subsequent additional limitations recited by

this claim element in view of the antecedent “the refined navigation query” (see, for example, ¶¶ 410-418 above).

420. Cheyer discloses a “*Reference Resolution Agent*” that is “responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*” (emphasis added, see, for example, Ex. 1019 at pp. 8-9).

421. Cheyer also provides a specific example of operation of the system wherein “A *user speaks*: “How far is the restaurant from this hotel?””, and subsequently “The *interface agent* uses contextual structures to find what “the restaurant” refers to, and *waits for the user to make a gesture indicating “the hotel”, issuing prompts if necessary*” and then the “*reference resolution agent (RR)*” in an exemplary embodiment “*sends database requests asking for the coordinates of the items in question*” (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

422. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

423. Thus, Cheyer discloses the recited method step of “using the refined navigation query” (at least by sending database requests after user resolution of an ambiguous reference) specifically “to select a portion of the electronic data source” (at least when such refined database requests return the selected information about the items in question).

424. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

425. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this

claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

'718 Patent: Claim 6

> 6. The method of claim 1, wherein steps (a)-(d) are performed with respect to multiple users.

6. The method of claim 1, wherein steps (a)-(d) are performed with respect to multiple users.

426. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 1 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 214-333 above.

427. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer" including "*transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems*" (emphasis added, see, for example, Ex. 1013 at 1:36-45).

428. Kupiec notes that "The *general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to *handwriting recognition in pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

429. Kupiec describes that "Processor **10** is a computer processing unit (CPU)" that typically is "part of a mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

430. Kupiec discloses that "*Display **30** provides visual output to the user*", which may be of the form of "alphanumeric display of the texts or titles", such as for "documents retrieved

from corpus **41**” and typically comprises “a _*computer screen or monitor*_” (emphasis added, see, for example, Ex. 1013 at 6:12-15).

431. Kupiec specifically discloses that “_*IR subsystem*_ **40** can be located at the same site as processor **10** or _*can be located at a remote site and connected to processor **10** via a suitable communication network*_” (emphasis added, see, for example, Ex. 1013 at 6:25-28).

432. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

433. Thus, Kupiec discloses the “method of claim 1” (as described herein), wherein “steps (a)-(d) are performed with respect to multiple users” (at least when the system operates with multiple computerized information-retrieval (IR) systems and multiple computers and personal digital assistants over a suitable communications network).

434. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

435. However, to the extent that alleged non-disclosure of the antecedent “method of claim 1” of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent “method of claim 1” of this claim element (see, for example, ¶¶ 214-333 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent “method of claim 1” (see, for example, ¶¶ 426-434 above).

436. Cheyer describes “Direct manipulation interface technologies” as comprising “the use of menus and a graphical user interface” such that “_*users*_ are presented with sets of discrete

actions and the objects on which to perform them" (see, for example, Ex. 1019 at p. 2) and further notes that "Gestures allow *users* to communicate a surprisingly wide range of meaningful requests with a few simple strokes" (emphasis added, see, for example, Ex. 1019 at p. 3).

437. Cheyer describes the system design criteria as including a "*user interface*" that is "light and fast enough to *run on a handheld PDA while able to access applications and data that may require a more powerful machine*" (emphasis added, see, for example, Ex. 1019 at p. 5). Similarly, Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" (emphasis added, see, for example, Ex. 1019 at p. 5).

438. Cheyer also discloses that "The *interface is connected either by modem or ethernet to a server machine* which will manage database access, natural language processing and speech recognition for the application" such that "The *result is a mobile system that provides a synergistic pen/voice interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

439. Cheyer describes that "In the Open Agent Architecture, *agents are distributed entities that can run on different machines, and communicate together to solve a task for the user*" and "*a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

440. According to Cheyer, "*Database Agents*" can "*reside at local or remote locations* and can be grouped hierarchically according to content" wherein such "*databases*" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the *World Wide Web (WWW)*" as well as a "*Reference Resolution Agent*" that

is "*responsible for merging requests arriving in parallel*" (emphasis added, see, for example, Ex. 1019 at p. 8).

441. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

442. Thus, Cheyer discloses the "method of claim 1" (as described herein), wherein "steps (a)-(d) are performed with respect to multiple users" (at least because the described mobile system supports multiple users operating multiple handheld computers connected via modem or Ethernet to remote databases managed by agents distributed on multiple different machines to handle multiple requests arriving in parallel).

443. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

444. Additionally, a POSITA at the time of the alleged invention would have known that for systems such as Kupiec, Cheyer, or Kupiec in view of Cheyer (as described herein), that such systems could be operated "*with respect to multiple users*". For example, the '021 Patent admits that "*Data source* **210** (or **100**), being a network accessible information resource, has *typically* already been *constructed to support access requests from simultaneous multiple network users, as known by practitioners of ordinary skill in the art*" and "the interpretation logic and error correction logic modules are also preferably *designed and implemented to support queuing and multi-tasking of requests from multiple simultaneous network users, as will be appreciated by those of skill in the art*" (emphasis added, see, for example, Ex. 1001 at 6:24-34). Moreover, a POSITA would understand that for systems such as Kupiec, Cheyer, or Kupiec in view of Cheyer (as described herein), that the only logical choices for such systems would be to operate either "*with respect to multiple users*" or "*with respect to one user*". Thus, to the extent

that Kupiec, Cheyer, or Kupiec in view of Cheyer (as described herein) were considered to not disclose the additional limitations of this claim element with respect such systems operating "*with respect to multiple users*", in my opinion, a POSITA at the time of the alleged invention would already be knowledgeable regarding such a topic and would find it obvious to try such an approach in view of the finite number of possibilities (effectively only two) and the high predictability that such an approach would lead to a successful outcome. Furthermore, a POSITA would have known from the prior art that systems that provide voice-driven navigation for information retrieval can operate "*with respect to multiple users*" (see, for example, ¶¶ 200-202 above).

445. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, because the additional limitations of this claim element were well known to a POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

446. Freeman describes that "This invention is a new model and *system for managing personal electronic information*" wherein "*streams and filters provide a unified framework* that subsumes many separate desktop applications to accomplish and handle personal communication, scheduling, and *search and retrieval tasks*" (emphasis added, see, for example, Ex. 1014 at 3:62-4:2). Freeman states that "in accordance with the present invention *users can access their personal document streams from any available platform such as* a UNIX machine, a

Macintosh or IBM-compatible *personal computer*, a *personal digital assistant (PDA)*, or a *set-top box via cable*" (emphasis added, see, for example, Ex. 1014 at 2:56-61).

447. Freeman also describes an "embodiment of the present invention" that "is implemented in *a client/server architecture running over the Internet*" as well as embodiments that implement "a *client viewport using graphically based X Windows*", "a *client viewport solely with text* in standard ASCII", and "a *client viewport for the NEWTON personal digital assistant (PDA)*" (emphasis added, see, for example, Ex. 1014 at 6:8-9, 6:17-23).

448. Freeman specifically discloses that "*A stream according to the present invention can be controlled by a voice-interface* as well as a computer and thereby be *accessed via a conventional phone*" wherein this "*voice interface would allow: (1) the stream to be searched and manipulated*; (2) new objects to be installed; (3) objects to be transferred; and (4) other capability" (emphasis added, see, for example, Ex. 1014 at 11:38-43).

449. Freeman observes that "A *stream is a data structure that can be* examined and to the extent possible *manipulated by many processes simultaneously*" wherein "A *stream must support simultaneous access* because: (1) a *user creates many software agents which may need to examine the stream concurrently*; and (2) a user may have granted other users limited access to the user's stream, and the user will want access to this stream even while the other users access the stream" (emphasis added, see, for example, Ex. 1014 at 13:50-52, 13:59-64).

450. Freeman further discloses that "One embodiment of the *present invention is configured such that each server may support three to four simultaneous users with stream sizes on the order of 100,000 documents* (perhaps a year or two of documents for the average user)" (emphasis added, see, for example, Ex. 1014 at 13:65-14:4). Additionally, Freeman discloses "embodiments of the present invention utilize a *multi-server and multi-threaded approach which*

provides a more scalable architecture" (emphasis added, see, for example, Ex. 1014 at 14:19-21).

451. Thus, Freeman discloses a voice-driven navigation system for information retrieval that is relevant to the "method of claim 1", and wherein such "method" is "performed with respect to multiple users" (at least because the described system operates with multiple simultaneous users, each with a corresponding client device such as a personal computer, PDA, or set-top box via cable).

452. In my opinion, a POSITA would have modified the system of Kupiec in view of Cheyer further in view of the system of Freeman specifically to arrive at a combination that meets the limitations of this claim element (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

453. First, Kupiec and Cheyer are both addressing the same basic problem of building a system for retrieval of information from remote electronic sources based upon user input made via spoken language (see, for example, ¶¶ 90, 91, 144 and 173 above), and Freeman is addressing a related problem of using spoken language input retrieve documents from streams in an Internet-based client/server architecture with client devices such as a personal computer, PDA, or set-top box via cable (see, for example, ¶¶ 193 and 200 above).

454. Second, Kupiec and Cheyer each describe systems with substantial similarities to each other, as well as this claim, and Freeman also describes a system with similarities regarding the use of client devices that use voice input and speech recognition, as evident from my analysis herein. Additionally, Freeman explicitly discloses the additional limitations of this claim element under the proper interpretation discussed above.

455. Third, Kupiec and Cheyer both describe systems where users can operate client devices to retrieve information in documents from network-connected sources based on spoken-input queries as described herein, and Freeman describes a system specifically where multiple simultaneous users, each with a corresponding client device such as a personal computer, PDA, or set-top box via cable can retrieve information in documents from network-connected sources based on spoken-input queries as described herein.

456. Fourth, neither of Kupiec or Cheyer teaches away from or excludes the operation of a system with "multiple users". Additionally, operation of such a system was within the admitted prior art and/or knowledge of a POSITA at the time of the alleged invention (see, for example, ¶ 444 above).

457. Therefore, in my opinion, Kupiec in view of Cheyer further in view of Freeman discloses the limitations of this claim element under the proper interpretation proposed herein.

<u>'718 Patent: Claim 8</u>

> 8. The method of claim 1, wherein the mobile information appliance is a portable computing device.

8. The method of claim 1, wherein the mobile information appliance is a portable computing device.

458. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 1 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 214-333 above.

459. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer" including "*<u>transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems</u>*" (emphasis added, see, for example, Ex. 1013 at 1:36-45).

460. Kupiec notes that "The *general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to *handwriting recognition in pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

461. Kupiec describes that "Processor **10** is a computer processing unit (CPU)" that typically is "part of a mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

462. Kupiec discloses that "*Display **30** provides visual output to the user*", which may be of the form of "alphanumeric display of the texts or titles", such as for "documents retrieved from corpus **41**" and typically comprises "a *computer screen or monitor*" (emphasis added, see, for example, Ex. 1013 at 6:12-15).

463. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

464. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

465. Thus, Kupiec discloses the "method of claim 1" (as described herein), wherein "the mobile information appliance is a portable computing device" (at least when the system operates with personal computers and personal digital assistants).

466. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

467. However, to the extent that alleged non-disclosure of the antecedent "method of claim 1" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "method of claim 1" of this claim element (see, for example, ¶¶ 214-333 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "method of claim 1" (see, for example, ¶¶ 458-466 above).

468. Cheyer describes the system design criteria as including a "*user interface*" that is "light and fast enough to *run on a handheld PDA while able to access applications and data that may require a more powerful machine*" (emphasis added, see, for example, Ex. 1019 at p. 5). Similarly, Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" (emphasis added, see, for example, Ex. 1019 at p. 5). In my opinion, a POSITA at the time of the alleged invention of the '718 Patent would understand Cheyer's disclosure of "PDA" to mean "personal digital assistant".

469. Cheyer also discloses that "The *interface is connected either by modem or ethernet to a server machine* which will manage database access, natural language processing and speech recognition for the application" such that "The *result is a mobile system that provides a synergistic pen/voice interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

470. Cheyer describes that "In the Open Agent Architecture, *agents are distributed entities that can run on different machines, and communicate together to solve a task for the*

user" and "*a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

471. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

472. Thus, Cheyer discloses the "method of claim 1" (as described herein), wherein "the mobile information appliance is a portable computing device" (at least because the described mobile system supports users operating handheld computers or personal digital assistants).

473. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

474. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

'718 Patent: Claim 9

9. The method of claim 8, wherein the portable computing device is a personal digital assistant.

9. The method of claim 8, wherein the portable computing device is a personal digital assistant.

475. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 8 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 458-474 above.

476. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an

applications program executed by a computer" including "*transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems*" (emphasis added, see, for example, Ex. 1013 at 1:36-45).

477. Kupiec notes that "The *general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to *handwriting recognition in pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

478. Kupiec describes that "Processor **10** is a computer processing unit (CPU)" that typically is "part of a mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

479. Kupiec discloses that "*Display **30** provides visual output to the user*", which may be of the form of "alphanumeric display of the texts or titles", such as for "documents retrieved from corpus **41**" and typically comprises "a *computer screen or monitor*" (emphasis added, see, for example, Ex. 1013 at 6:12-15).

480. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

481. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

482. Thus, Kupiec discloses the "method of claim 8" (as described herein), wherein "the portable computing device is a personal digital assistant" (at least when the system operates with personal digital assistants).

483. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

484. However, to the extent that alleged non-disclosure of the antecedent "method of claim 8" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "method of claim 8" of this claim element (see, for example, ¶¶ 458-474 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "method of claim 8" (see, for example, ¶¶ 475-483 above).

485. Cheyer describes the system design criteria as including a "*user interface*" that is "light and fast enough to *run on a handheld PDA while able to access applications and data that may require a more powerful machine*" (emphasis added, see, for example, Ex. 1019 at p. 5). Similarly, Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" (emphasis added, see, for example, Ex. 1019 at p. 5). In my opinion, a POSITA at the time of the alleged invention of the '718 Patent would understand Cheyer's disclosure of "PDA" to mean "personal digital assistant".

486. Cheyer also discloses that "The *interface is connected either by modem or ethernet to a server machine* which will manage database access, natural language processing and speech recognition for the application" such that "The *result is a mobile system that provides a synergistic pen/voice interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

487. Cheyer describes that "In the Open Agent Architecture, *agents are distributed entities that can run on different machines, and communicate together to solve a task for the user*" and "*a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

488. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

489. Thus, Cheyer discloses the "method of claim 8" (as described herein), wherein "the portable computing device is a personal digital assistant" (at least because the described mobile system supports users operating personal digital assistants).

490. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

491. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

'718 Patent: Claim 10

> 10. A computer program embodied on a computer readable medium for speech-based navigation of an electronic data source located at one or more network servers located remotely from a user, wherein a data link is established between a mobile information appliance of the user and the one or more network servers, comprising:
>
> (a) a code segment that receives a spoken request for desired information from the user utilizing the mobile information appliance of the user, wherein said mobile information appliance comprises a portable remote control device or a set-top box for a television;
>
> (b) a code segment that renders an interpretation of the spoken request;
>
> (c) a code segment that constructs a navigation query based upon the interpretation;
>
> (d) a code segment that utilizes the navigation query to select a portion of the electronic data source; and
>
> (e) a code segment that transmits the selected portion of the electronic data source from the network server to the mobile information appliance of the user.

10. A computer program embodied on a computer readable medium for speech-based navigation of an electronic data source located at one or more network servers located remotely from a user, wherein a data link is established between a mobile information appliance of the user and the one or more network servers, comprising:

492. In my opinion, this preamble claim element is a claim limitation at least because I believe that this preamble recites essential structure and/or steps that give life, meaning, and vitality to the claim as opposed to *only* stating a purpose or intended use for the alleged invention. See also ¶¶ 78-81 above.

493. See ¶ 84 above regarding claim construction for this claim element.

494. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "*relates to systems and methods for* transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer" including "transcription systems and methods appropriate for use in conjunction with computerized *information-retrieval (IR) systems* and methods" and "more particularly to *speech-recognition systems and methods appropriate for use in conjunction with computerized information-retrieval systems*" (emphasis added, see, for example, Ex. 1013 at 1:36-45).

495. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

496. Kupiec describes that "*Processor* ***10*** *is* a computer processing unit (CPU)" that typically is "*part of a* mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

497. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor* ***10*** *via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

498. Kupiec discloses in reference to FIG. 11 that "*processor* ***200*** *executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, *hypothesis generator* ***260****, query/IR mechanism* ***270****, and analyzer/evaluator* ***280*** *are typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, *for query construction, and for scoring*" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

499. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

500. Thus, Kupiec discloses a "computer program embodied on a computer readable medium" (such as software modules that execute on a processor) for "speech-based navigation of an electronic data source" (the operation of a speech-recognition system in conjunction with an information-retrieval system) wherein such an "electronic data source" (the information-retrieval system or IR subsystem) is "located at one or more network servers located remotely from a user" (at least when the IR subsystem is located at a remote site and connected to the speech-recognition system via a suitable communication network), and wherein "a data link is established between a mobile information appliance of the user and the one or more network servers" (at least when a personal computer such as a disclosed portable personal digital assistant

would be configured to receive spoken input and to connect to an information-retrieval system over a communications network).

501. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

502. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

503. As Cheyer summarizes in its "Conclusions" section, "By augmenting an existing agent-based architecture with concepts necessary for synergistic multimodal input", Cheyer discloses "a mobile, synergistic pen/*voice interface providing good natural language access to heterogeneous distributed knowledge sources*" (emphasis added, see, for example, Ex. 1019 at p. 10).

504. Cheyer specifically discloses that "*the system permits the user* to simultaneously combine direct manipulation, gestural drawings, handwritten, typed and *spoken natural language*" and that the system uses "Existing commercial or research natural language and *speech recognition systems*", thereby enabling "a user" to "transparently *access* a wide variety of data sources, including *information stored in HTML form on the World Wide Web*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

505. Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" wherein "The *interface is connected either by modem or ethernet to a server machine*

which will manage database access, natural language processing and speech recognition for the application" such that "The result is a *mobile system that provides* a synergistic pen/voice *interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

506. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

507. Thus, Cheyer discloses a "computer program embodied on a computer readable medium" (such as a network of heterogeneous software agents) for "speech-based navigation of an electronic data source" (the operation of a speech-recognition and navigation system in conjunction with remote databases and/or the World Wide Web) wherein such an "electronic data source" (such as remote databases and/or the World Wide Web) is "located at one or more network servers located remotely from a user" (at least because the World Wide Web is located across numerous network servers remote from any individual user and remote databases are remote), and wherein "a data link is established between a mobile information appliance of the user and the one or more network servers" (at least when a handheld PDA is connected by modem or ethernet to a server machine which will manage database access, natural language processing and speech recognition).

508. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

509. At least because each of Kupiec and Cheyer discloses the limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein.

510. To the extent that Kupiec's disclosures are considered to not include the recited "mobile information appliance" of this claim (or any of its elements below, or its dependent claims herein), then in my opinion, a POSITA would have modified the system of Kupiec in view of the system of Cheyer specifically to arrive at a combination that meets the limitations of this claim element (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

511. First, see ¶ 232 above.

512. Second, see ¶ 233 above.

513. Third, see ¶ 234 above.

514. Fourth, see ¶ 235 above.

515. Fifth, see ¶ 236 above.

516. Therefore, in my opinion, Kupiec in view of Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

10(a) a code segment that receives a spoken request for desired information from the user utilizing the mobile information appliance of the user, wherein said mobile information appliance comprises a portable remote control device or a set-top box for a television;

517. See ¶ 84 above regarding claim construction for this claim element.

518. In reference to FIG. 1, Kupiec describes that "Transducer **20** converts a *user's spoken utterance into a signal that can be processed* by processor **10**" and can comprise "*a microphone coupled to an analog-to-digital converter, so that the user's speech is converted by transducer **20** into a digital signal*" (emphasis added, see, for example, Ex. 1013 at 5:56-6:7).

519. Also, in reference to FIG. 2, Kupiec describes that "The *user inputs a question **201** into system **1** by speaking into audio transducer **20***" (emphasis added, see, for example, Ex. 1013 at 9:18-23). FIG. 2 of Kupiec and its detailed description explicitly illustrate that "question

201" is not only "spoken" but is also a "request for desired information" to be found by "a series of queries" provided "to IR subsystem **40**, which executes them by conducting searches in accordance with queries **270** over corpus **41**" such that "The execution of the initial and any additional queries causes a set of documents **240** to be retrieved from corpus **41**" (see, for example, Ex. 1013 at 11:53-60, and see also, for example, ¶¶ 106-112 above).

520. Similarly, in reference to FIG. 11, Kupiec describes that "In operation, *transducer **220** accepts an input question **301*** and converts it into a signal **320**" wherein "*input question **301** can be a spoken utterance, in which case transducer **220** comprises audio signal processing equipment* that converts the spoken utterance to signal **320**" (emphasis added, see, for example, Ex. 1013 at 24:22-41).

521. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "*Transcriber **250***, hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are *typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21).

522. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

523. Kupiec describes that "*Processor **10** is* a computer processing unit (CPU)" that typically is "*part of a* mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

524. Kupiec further discloses that the system "has been demonstrated on a Sun SparcStation 10 workstation" and that "*Discrete-word speech can be input using a Sennheiser HMD414 headset microphone* and a Rane MS-1 preamplifier, *with signal processing performed in software by the SparcStation*" so that such "Input speech is transcribed into a phone sequence using hidden Markov model methods" as exemplified in a prior art 1989 IEEE paper (emphasis added, see, for example, Ex. 1013 at 29:45-46, 30:48-56).

525. See also, for example, ¶¶ 94, 106, 114, 115, 125 and 129 above with respect to Kupiec and this claim element.

526. Thus, Kupiec discloses a "code segment" (at least the transcriber software module and/or the signal processing software) that "receives a spoken request" (at least when the microphone/transducer accepts an input question) that is "for desired information" (such as the material to be searched and retrieved from the IR subsystem) and that is "from the user" (at least when the input question is a user's spoken utterance), and further is "utilizing the mobile information appliance of the user" (at least when a personal computer such as a disclosed portable personal digital assistant would be configured to receive spoken input), wherein "said mobile information appliance comprises a portable remote control device or a set-top box for a television" (at least because in a speech recognition controlled system a portable headset microphone constitutes a portable remote control device).

527. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

528. However, to the extent that alleged non-disclosure of the antecedent "the mobile information appliance" of this claim element for Kupiec alone were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses

the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the mobile information appliance" of this claim element (see, for example, ¶¶ 510-516 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the mobile information appliance" (see, for example, ¶¶ 517-527 above).

529. In the section entitled "A Multimodal Map Application", Cheyer describes "a prototype map-based application for a travel planning domain" wherein "the *system permits the user to* simultaneously combine direct manipulation, gestural drawings, handwritten, typed and *spoken natural language*" and this system uses "Existing commercial or research natural language and *speech recognition systems*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

530. In reference to Fig. 1, Cheyer discloses that "the *user*" is "*presented with a* pen sensitive map *display*" and can provide "*spoken input*" and the "*user may ask the map to perform various actions*" such as "*information retrieval*" (emphasis added, see, for example, Ex. 1019 at p. 5).

531. Cheyer further notes that "The application also makes use of *multimodal (multimedia) output* as well as input: *video, text, sound and voice can all be combined when presenting an answer to a query*" (emphasis added, see, for example, Ex. 1019 at p. 5).

532. Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either *a microphone or a telephone for voice input*" (emphasis added, see, for example, Ex. 1019 at p. 5). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

533. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

534. Thus, Cheyer discloses a "code segment" (at least the speech recognition software agent and/or Decipher/Corona speech recognition system software) that "receives a spoken request" (at least when the microphone or telephone accepts spoken or voice input) that is "for desired information" (such as for information retrieval) and that is "from the user" (at least when the spoken input is provided by a user), and further is "utilizing the mobile information appliance of the user" (at least when the handheld PDA receives the spoken input), wherein "said mobile information appliance comprises a portable remote control device or a set-top box for a television" (at least because the handheld PDA is itself a portable remote control device for controlling access to the information retrieval system).

535. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

536. Additionally, see ¶ 256 above.

537. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, because the additional limitations of this claim element were well known to a POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein.

538. Kimura "relates to a *remote control system for remotely controlling various electronic devices*, and *more particularly to a remote control system for remotely controlling devices such as AV (audio visual) devices by way of voice commands*" including such "AV devices" as "*television receivers*" (emphasis added, see, for example, Ex. 1015 at 1:8-14).

Additionally, Kimura is directed to a “*voice-operated remote control system which can vary a speech recognition process depending* on the degree of importance of a control command” in view of the fact that “the magnitudes of *effects caused by erroneous recognition*, may not necessarily be the same” (emphasis added, see, for example, Ex. 1015 at 1:41-42, 1:54-57).

539. Kimura discloses a “general *remote control system*” that “comprises a *transmitter **101** for transmitting a remote control signal from a position remote from a controlled device **103** such as an AV device*, and a receiver **102** for receiving the transmitted remote control signal, decoding the remote control signal, and sending the decoded information to the controlled device **103**” (emphasis added, see, for example, Ex. 1015 at 3:10-15).

540. More specifically, Kimura discloses that “FIG. 3 is a block diagram of the transmitter of a general voice-operated remote control system”, wherein the “*transmitter **101** has a microphone **M** for converting a voice command* into an electric signal” that “is *applied to a speech recognition circuit **15*** in the form of a speech recognition LSI circuit or the like *which includes a microprocessor*” and “*produces command data corresponding to the recognized contents*”, and further wherein “The transmitter **101** also has a controller **16** comprising a microprocessor” (emphasis added, see, for example, Ex. 1015 at 3:27-36).

541. Kimura describes in reference to FIG. 4 that “*transmitter **10A** of the voice-operated remote control system* has a unitary casing **11** which *allows the operator to carry the transmitter freely around*” (emphasis added, see, for example, Ex. 1015 at 4:9-12). Kimura further describes that “casing **11** supports a microphone **M** on an upper panel thereof” wherein “microphone **M** converts a voice command given by the operator into an electric signal” and on “one side of the casing **11**, there is disposed a *voice input switch* (hereinafter referred to as a "talk switch") **12** which is closed when pressed and can automatically be released or opened when

released" so that when a "voice command is to be entered, the talk switch **12** is closed to operate the transmitter **10A**" or "Otherwise, the talk switch **12** is open keeping the transmitter **10A** out of operation" (emphasis added, see, for example, Ex. 1015 at 4:12-28).

542. Kimura describes in reference to FIG. 7 that "the *speech recognition circuit **15A** comprises* an analog processor **21** for processing an analog voice command signal which is received through the microphone **M** and outputting the processed analog voice command signal as a time-division digital data **20**, *a speech recognition processor **22** for recognizing the voice command* based on the time-division digital data **20** from the analog processor **21**, a memory **23A** for storing standard pattern data for speech recognition, and an interface **24** for transmitting signals to and receiving signals from the controller **16A**" (emphasis added, see, for example, Ex. 1015 at 5:3-18).

543. Additionally, Kimura describes in reference to FIG. 8 that "the *speech recognition processor **22** comprises a system controller **40*** for analyzing and processing control commands from the controller **16** and also for controlling the entire operation of the speech recognition processor **22**, and a digital processor **41** for effecting distance calculations and controlling the memory **23A**" wherein the "*system controller **40** comprises a CPU (Central Processing Unit) **42** for controlling the overall operation of the transmitter* **1**" (emphasis added, see, for example, Ex. 1015 at 6:21-30).

544. See also, for example, ¶¶ 178, 180, 184, 187 and 189 above with respect to Kimura and this claim element.

545. Thus, Kimura discloses "a portable remote control device" that is "for a television" (such as the portable transmitter with a CPU that provides a remote control signal to a

television receiver) and is part of a system for "receiving a spoken request" (at least because the transmitter includes a microphone and a speech recognition processor).

546. Although Kupiec in view of Cheyer renders obvious the limitations of this claim element per my analysis herein under the proper interpretation, to the extent that the limitation of "said mobile information appliance comprises a portable remote control device or a set-top box for a television" were construed to mean "said mobile information appliance comprises a *portable remote control device* or a set-top box *for a television*", as opposed to a "*portable remote control device*" that can be for purposes that may not necessarily be "*for a television*", then in my opinion, a POSITA would have modified the system of Kupiec in view Cheyer further in view of the system of Kimura specifically to arrive at a combination that meets the limitations of this claim element under this alternative claim construction (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

547. First, see ¶ 267 above.

548. Second, see ¶ 268 above.

549. Third, see ¶ 269 above.

550. Fourth, see ¶ 270 above.

551. Fifth, see ¶ 271 above.

552. Therefore, in my opinion, Kupiec in view of Cheyer further in view of Kimura discloses the limitations of this claim element for the alternative construction of this claim element as described herein.

10(b) a code segment that renders an interpretation of the spoken request;

553. In reference to FIG. 2, Kupiec describes that "*signal **220** produced by transducer **20** is fed to transcriber **50**, where it is converted into a phonetic transcription **250***" using "any of a variety of transcription techniques" that were "well-known among those of skill in the art"

(emphasis added, see, for example, Ex. 1013 at 9:18-23). Kupiec explains that "*phonetic transcription* ***250*** *is an ordered sequence* of phones, that is, *of component sounds that can be used to form words*" (emphasis added, see, for example, Ex. 1013 at 9:29-31).

554. Also, in reference to FIG. 3, Kupiec describes that "*First the system accepts a user utterance* as input (Step A)" and that "This *utterance is converted to a signal (Step B) that is transcribed into a sequence of phones* (Step C)" (emphasis added, see, for example, Ex. 1013 at 12:48-57).

555. Similarly, in reference to FIG. 11, Kupiec describes that "Transducer **220** provides signal **320** to transcriber **250**" such that "*Transcriber* ***250*** *converts signal* ***320*** *to a string* ***350*** *that represents a transcription of the input question* ***301***" (emphasis added, see, for example, Ex. 1013 at 24:66-25:4).

556. Kupiec discloses in reference to FIG. 11 that "*processor* ***200*** *executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "*Transcriber* ***250***, hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are *typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21).

557. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

558. Thus, Kupiec discloses a "code segment" (at least the transcriber that can be implemented as a software module executing on a processor) that "renders an interpretation" (at least when the transcriber produces a phonetic transcription of component sounds that can be used to form words) that is "of the spoken request" (at least because the transcriber operates upon the user's spoken utterance).

559. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

560. However, to the extent that alleged non-disclosure of the antecedent "the spoken request" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the spoken request" of this claim element (see, for example, ¶ 528 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the spoken request" (see, for example, ¶¶ 553-559 above).

561. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; access to existing data sources including the World Wide Web; and a mobile handheld interface" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

562. In the section entitled "A Multimodal Map Application", Cheyer describes "a prototype map-based application for a travel planning domain" wherein "the *system permits the user to* simultaneously combine direct manipulation, gestural drawings, handwritten, typed and *spoken natural language*" and this system uses "*Existing commercial* or research natural language and *speech recognition systems*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

563. Cheyer also specifically describes, from a 1990 prior art article, a "*Speech Recognition (SR) Agent*" that "provides a mapping from the Interagent Communication

Language to the API *for the Decipher (Corona) speech recognition system*", which Cheyer describes as "*a continuous speech speaker independent recognizer based on Hidden Markov Model technology*" (emphasis added, see, for example, Ex. 1019 at p. 8). Cheyer further specifically describes, with reference to a 1994 prior art article, a "*Natural Language (NL) Parser Agent*" that "*translates English expressions into the Interagent Communication Language (ICL)*" (emphasis added, see, for example, Ex. 1019 at p. 8).

564. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

565. Thus, Cheyer discloses a "code segment" (at least the speech recognition and/or natural language parser software agents in combination with the Decipher/Corona speech recognition system) that "renders an interpretation" (at least when speech recognition system and parser creates natural language English expressions) that is "of the spoken request" (at least because the speech recognition system and parser operates upon the user's spoken natural language).

566. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

567. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

10(c) a code segment that constructs a navigation query based upon the interpretation;

568. See ¶ 83 above regarding claim construction for this claim element.

569. In reference to FIG. 2, Kupiec describes that "The *phonetic transcription **250** is provided to hypothesis generator **60*** where it is matched using phonetic index **62** *to generate a set of hypotheses **260***" (emphasis added, see, for example, Ex. 1013 at 9:38-61) and further that "Once the set of *hypotheses **260*** has been generated, it is *provided to query constructor **70***" such that "Query constructor **70** uses the hypotheses **260** *to construct one or more queries **270** that will be sent to IR subsystem **40** for execution*" (emphasis added, see, for example, Ex. 1013 at 11:10-13).

570. Additionally, Kupiec specifically discloses that "Queries **270** are Boolean queries with proximity and order constraints" and provides a specific example wherein the "user speaks two words" that lead to a "set of hypotheses" such "an initial *query*" is constructed that "*seeks occurrences of* at least one of the words (*search terms*)" and is "*sent to the IR subsystem **40*** where it is executed" (emphasis added, see, for example, Ex. 1013 at 11:13-41).

571. Also, in reference to FIG. 3, Kupiec describes that "First the system accepts a user utterance" that is "transcribed" and then "*used to generate hypotheses* (Step D)" such that "*Boolean queries with proximity and order constraints are constructed based on the hypotheses and are executed to retrieve documents* (Step E)" (emphasis added, see, for example, Ex. 1013 at 12:48-57).

572. Similarly, in reference to FIG. 11, Kupiec describes that "Transducer **220** provides signal **320** to transcriber **250**" that "converts signal **320** to a string **350** that represents a transcription of the input question **301**" such that "Hypothesis generator **260** converts string **350** and any alternatives to a set of hypotheses **360**" provided to "*Query/IR mechanism* **270**" that "*converts the hypotheses **360** to one or more information retrieval queries* **370**" that are "in a format *that can be searched* by processor **200** (or a separate IR processor that communicates

with processor **200**) using corpus **241**" (emphasis added, see, for example, Ex. 1013 at 24:66-25:61).

573. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, *hypothesis generator **260**, query/IR mechanism **270**,* and analyzer/evaluator **280** *are typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, *for query construction*, and for scoring" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

574. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

575. Thus, Kupiec discloses a "code segment" (at least the hypothesis generator and the query constructor that can be implemented as software modules executing on a processor) that "constructs a navigation query" (at least when the query constructor constructs one or more queries that can be used to search the IR subsystem) that is "based upon the interpretation" (at least because the query constructor operates upon the hypotheses formed from the transcription of the user's spoken utterance).

576. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

577. However, to the extent that alleged non-disclosure of the antecedent "the interpretation" of this claim element for Kupiec alone in view of another antecedent limitation in

a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the interpretation" of this claim element (see, for example, ¶ 560 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the interpretation" (see, for example, ¶¶ 568-576 above).

578. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

579. In reference to Figure 1, Cheyer discloses that "the *user*" can provide "*spoken input*" and the "*user may ask the map to perform various actions*" such as "*information retrieval*" (emphasis added, see, for example, Ex. 1019 at p. 5).

580. Cheyer further describes that the system can "capture signals emitted during a user's interaction" and also "integrates a set of modality agents, each responsible for a very specialized kind of signal" wherein the "modality agents are connected to an '*interpret agent*' which is *responsible for combining the inputs across all modalities to form a valid command* for the application" (emphasis added, see, for example, Ex. 1019 at p. 7).

581. Cheyer discloses "Database Agents" wherein exemplary "*databases*" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning *HTML pages from the World Wide Web (WWW)*" as well as "information" that is "extracted by an HTML reading database agent" (emphasis added, see, for example, Ex. 1019 at p. 8).

582. Cheyer also discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*" wherein this "reference resolution agent (RR)" in an exemplary embodiment "sends database requests asking for the coordinates of the items in question" (emphasis added, see, for example, Ex. 1019 at pp. 8-10).

583. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

584. Thus, Cheyer discloses a "code segment" (at least the reference resolution software agent and/or user interface, database and modality software agents) that "constructs a navigation query" (at least when the reference resolution agent sends database requests for data on the World Wide Web) that is "based upon the interpretation" (at least because the reference resolution agent creates database requests from the output of agents that respond to spoken input).

585. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

586. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

10(d) a code segment that utilizes the navigation query to select a portion of the electronic data source;

587. See ¶ 83 above regarding claim construction for this claim element.

588. Kupiec describes that "Depending on the results obtained from execution of the initial query, *additional queries can be constructed and executed, in a process called query reformulation*" in order to "*send the query thus modified back to IR subsystem **40** to be executed again*" (emphasis added, see, for example, Ex. 1013 at 11:42-48).

589. Kupiec also states that "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "*The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query*" (emphasis added, see, for example, Ex. 1013 at 15:1-12).

590. Kupiec discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search* based on the retrieved results" (emphasis added, see, for example, Ex. 1013 at 19:32-36).

591. Kupiec further discloses that "*IR subsystem **40*** incorporates a processor that *can process queries to search for documents in corpus **41***" (emphasis added, see, for example, Ex. 1013 at 6:22-25). According to Kupiec, "a series of queries **270** is constructed by query constructor **70** and provided to IR subsystem **40**, which executes them by *conducting searches in accordance with queries* **270** over corpus **41**" such that "The execution of the initial and any additional *queries causes a set of documents **240** to be retrieved from corpus **41***" (emphasis added, see, for example, Ex. 1013 at 11:53-60).

592. Kupiec discloses in reference to FIG. 9 that "A relevant subset of the hypotheses and retrieved documents is presented to the user (Step KK). If documents have previously been retrieved, then user relevance feedback commands and search terms can be routed to the

hypothesis generator, to instruct the hypothesis generator to use retrieved document titles as the basis for confirming hypotheses (Step LL), or to cease doing this upon a "new search" or similar command. The *system then can perform operations such as a vector space search or the selection of one among several preferred hypotheses* (Step MM). Results of these operations are presented to the user (Step KK)" (emphasis added, see, for example, Ex. 1013 at 20:21-32).

593. Similarly, in reference to FIG. 11, Kupiec describes that "Transducer **220** provides signal **320** to transcriber **250**" that "converts signal **320** to a string **350** that represents a transcription of the input question **301**" such that "Hypothesis generator **260** converts string **350** and any alternatives to a set of hypotheses **360**" provided to "*Query/IR mechanism* **270**" that "*converts the hypotheses **360** to one or more information retrieval queries* **370**" that are "in a format *that can be searched* by processor **200** (or a separate IR processor that communicates with processor **200**) using corpus **241**" (emphasis added, see, for example, Ex. 1013 at 24:66-25:61).

594. Kupiec specifically discloses that "*IR subsystem* **40** *can be located at the same site as processor **10*** or can be located at a remote site and connected to processor **10** via a suitable communication network" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

595. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "*Transcriber **250**, hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software* **205** and are *executed by processor* **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code *for reading a phonetic index file, for query*

construction, and for scoring" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

596. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

597. Thus, Kupiec discloses a "code segment" (at least the analyzer/evaluator and/or query constructor that can be implemented as software modules executing on a processor in combination with the information retrieval subsystem software that conducts searches) that "utilizes the navigation query" (at least by searching the information retrieval subsystem or a subset thereof with an initial or modified query) specifically "to select a portion of the electronic data source" (at least when the documents from the information retrieval subsystem search are selected via a preferred hypothesis based on the query).

598. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

599. However, to the extent that alleged non-disclosure of the antecedent "the navigation query" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the navigation query" of this claim element (see, for example, ¶ 577 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the navigation query" (see, for example, ¶¶ 587-598 above).

600. As Cheyer summarizes in its “Abstract” section, Cheyer presents “a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface” that is implemented using “a *hierarchical distributed network of heterogeneous software agents*” (emphasis added, see, for example, Ex. 1019 at p. 2).

601. According to Cheyer, “*Database Agents*” can “*reside at local or remote locations and can be grouped hierarchically according to content*” wherein such “databases” can include “Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the World Wide Web (WWW)” as well as “information” that is “extracted by an HTML reading database agent” such as a “list of current movie times and reviews” (emphasis added, see, for example, Ex. 1019 at p. 8).

602. Cheyer discloses a “*Reference Resolution Agent*” that is “responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*” (emphasis added, see, for example, Ex. 1019 at pp. 8-9).

603. Cheyer also provides a specific example of operation of the system wherein “A *user speaks*: “How far is the restaurant from this hotel?””, and subsequently “The *interface agent* uses contextual structures to find what “the restaurant” refers to, and *waits for the user to make a gesture indicating “the hotel”, issuing prompts if necessary*” and then the “*reference resolution agent (RR)*” in an exemplary embodiment “*sends database requests asking for the coordinates of the items in question*” (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

604. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

605. Thus, Cheyer a "code segment" (at least the database and/or reference resolution software agents) that "utilizes the navigation query" (at least by sending database requests after user resolution of an ambiguous reference) specifically "to select a portion of the electronic data source" (at least when such refined database requests return the selected information about the items in question).

606. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

607. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

10(e) a code segment that transmits the selected portion of the electronic data source from the network server to the mobile information appliance of the user.

608. See ¶ 84 above regarding claim construction for this claim element.

609. Kupiec describes that "*Processor **10** is* a computer processing unit (CPU)" that typically is "*part of a* mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

610. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

611. Kupiec discloses that "*Display **30** provides visual output to the user*" such as for "*documents retrieved from corpus **41***" and typically comprises "*a computer screen* or monitor",

and further that "*Corpus **41** comprises a database of documents* that can be searched by IR subsystem **40**" wherein such documents comprise "for example, *books, articles from newspapers and periodicals, encyclopedia articles, abstracts, office documents*, etc." (emphasis added, see, for example, Ex. 1013 at 6:12-15, 29-33). Similarly, Kupiec additionally discloses that "*Output channel **230** can send interpretation **400** to be displayed using a visual display **231***" in order "to facilitate the understanding of the inputs that the *user provides as relevance feedback*" (emphasis added, see, for example, Ex. 1013 at 27:7-18).

612. In reference to FIG. 3 of Kupiec, "A *relevant subset of the* hypotheses and *retrieved documents is presented to the user* (Step G)" (emphasis added, see, for example, Ex. 1013 at 12:48-57). Similarly, in reference to FIG. 9 of Kupiec, "The *system* then *can perform operations such as a vector space search or the selection of one among several preferred hypotheses* (Step MM)" such that "*Results of these operations are presented to the user* (Step KK)" (emphasis added, see, for example, Ex. 1013 at 20:21-32).

613. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

614. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "*Transcriber **250**, hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software* **205** and are *executed by processor* **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp"

wherein the "first file includes source code *for reading a phonetic index file, for query construction, and for scoring*" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

615. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

616. Thus, Kupiec discloses a "code segment" (at least the software associated with the suitable communication network that connects the information retrieval subsystem at a remote site to the workstation or personal computer at the user) that "transmits the selected portion of the electronic data source from the network server to the mobile information appliance of the user" (at least by retrieving documents from an information retrieval subsystem at a remote site over a communications network and displaying the selected documents on a computer screen such as that of a personal digital assistant).

617. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

618. However, to the extent that alleged non-disclosure of the antecedent "the selected portion" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element or alleged non-disclosure of the antecedent "the mobile information appliance" were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the selected portion" and "the mobile information appliance" of this claim element (see, for example, ¶¶ 510-516 above and ¶ 599 above) and Kupiec discloses the subsequent

additional limitations recited by this claim element in view of the antecedent "the selected portion" and "the mobile information appliance" (see, for example, ¶¶ 608-617 above).

619. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and speech modalities; *access to existing data sources including the World Wide Web*; and a *mobile handheld interface*" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

620. Cheyer further describes the system as enabling "Through the multimodal interface" that "a user" can "*transparently access a wide variety of data sources, including information stored in HTML form on the World Wide Web*" and Cheyer describes the system as also having a "user interface" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" (emphasis added, see, for example, Ex. 1019 at p. 5). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

621. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" and then subsequently "*requests the user interface to produce output displaying the result* of the calculation" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

622. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

623. Thus, Cheyer a "code segment" (at least the software associated with the modem or ethernet that connects data sources such as the World Wide Web to the user's PDA) that "transmits the selected portion of the electronic data source from the network server to the mobile information appliance of the user" (at least by retrieving information in databases on the World Wide Web that gets displayed on the user interface of mobile pen-equipped personal computers such as the Dauphin handheld PDA).

624. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

625. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

<u>'718 Patent: Claim 11</u>

> 11. The computer program of claim 10, wherein the rendering of the interpretation of the spoken request is performed at the one or more network servers.

11. The computer program of claim 10, wherein the rendering of the interpretation of the spoken request is performed at the one or more network servers.

626. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 10 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 492-625 above.

627. According to Kupiec, "*<u>System **1** comprises a processor **10**</u>* coupled to an input audio transducer **20**, an output visual display **30**, an optional output speech synthesizer **31**, and an information retrieval (IR) subsystem **40** which accesses documents from corpus **41** using a word index **42**" as well as "*<u>a phonetic transcriber **50**, a hypothesis generator **60**</u>*, a phonetic

index **62**, a query constructor **70**, and a scoring mechanism **80**" (emphasis added, see, for example, Ex. 1013 at 5:43-51).

628. Kupiec describes that "*Processor **10** is a computer processing unit (CPU)*" that typically is "*part of a mainframe*, workstation, or personal *computer*" but can comprise "*multiple processing elements in some embodiments*" (emphasis added, see, for example, Ex. 1013 at 5:52-55). In my opinion, a POSITA at the time of alleged invention would consider an operation performed in a "mainframe computer" to be an example of at least performing such an operation "at the one or more network servers".

629. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "*Transcriber **250**, hypothesis generator **260**,* query/IR mechanism **270**, and analyzer/evaluator **280** *are typically implemented as software modules that are part of software* **205** and are *executed by processor* **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, for query construction, and for scoring" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

630. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

631. Thus, Kupiec discloses the "computer program of claim 10" (as described herein), wherein "the rendering of the interpretation of the spoken request is performed at the one or more network servers" (at least when the phonetic transcriber and hypothesis generator are

implemented on a mainframe computer that would normally be located remotely from the user such as at one or more network servers).

632. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

633. However, to the extent that alleged non-disclosure of the antecedent "computer program of claim 10" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element or alleged non-disclosure of the antecedent "the interpretation" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "computer program of claim 10" and "the interpretation" of this claim element (see, for example, ¶¶ 492-625 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "computer program of claim 10" and "the interpretation" (see, for example, ¶¶ 626-631 above).

634. Cheyer describes the system as also having a "*user interface*" that "runs *on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" wherein "The interface is *connected either by modem or ethernet to a server machine* which will *manage database access, natural language processing and speech recognition* for the application" (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

635. Cheyer describes its system as implemented with an "*Open Agent Architecture (OAA)*" that "*provides a framework for coordinating a society of agents which interact to solve problems for the user*" and "*provides distributed access to* commercial applications, such as mail

systems, calendar programs, *databases*, etc." (emphasis added, see, for example, Ex. 1019 at p. 7).

636. Cheyer also describes that "In the *Open Agent Architecture, agents are distributed entities that can run on different machines*, and communicate together to solve a task for the user" (see, for example, Ex. 1019 at p. 8). More specifically Cheyer discloses "Macro Agents", which "contain some knowledge and ability to reason about a domain, *and can answer or make queries to other macro agents using the Interagent Communication Language*", and "Micro Agents", which "are responsible for handling a single input or output data stream, either filtering the signal to or from a hierarchically superior 'interpret' agent" (emphasis added, see, for example, Ex. 1019 at p. 8).

637. According to Cheyer, the "network architecture" used was "hierarchical at two resolutions" wherein "micro agents are connected to a superior macro agent" and "macro agents are connected in turn to a facilitator agent" but "In both cases, *a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

638. Cheyer also specifically describes a "*Speech Recognition (SR) Agent*" that "provides a mapping from the Interagent Communication Language to the API for the Decipher (Corona) speech recognition system", which Cheyer describes as "a continuous speech speaker independent recognizer based on Hidden Markov Model technology" (emphasis added, see, for example, Ex. 1019 at p. 8). Cheyer further specifically describes a "Natural Language (NL) Parser Agent" that "translates English expressions into the Interagent Communication Language (ICL)" (emphasis added, see, for example, Ex. 1019 at p. 8).

639. Cheyer also specifically discloses the "*Speech Recognition (SR) Agent*" and the "*Reference Resolution (RR) Agent*" as examples of "macro agents" (see, for example, Ex. 1019 at p. 9, Figure 3).

640. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

641. Thus, Cheyer discloses the "computer program of claim 10" (as described herein), wherein "the rendering of the interpretation of the spoken request is performed at the one or more network servers" (at least when the speech recognition, language parser and database agents are implemented on a server or different machines that are networked together via a modem or ethernet).

642. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

643. Additionally, a POSITA at the time of the alleged invention would have known that for systems such as Kupiec, Cheyer, or Kupiec in view of Cheyer (as described herein), that "rendering an interpretation" could be performed "*at the one or more network servers*". For example, the '718 Patent admits that "It will be *apparent to those skilled in the art* that additional implementations, permutations and combinations of the embodiments set forth in FIGS. 1a, 1b, and 2 may be created" and that "*practitioners will understand*" that "*it is possible to divide and allocate the functional components of request processing logic* ***300*** *between client and server*" such as "*natural language parsing and other necessary processing might be performed upstream on the server end,* so that more extensive computational power need not be distributed locally to each client" (emphasis added, see, for example, Ex. 1003 at 6:45-59). Moreover, a POSITA would understand that for systems such as Kupiec, Cheyer, or Kupiec in view of Cheyer (as

described herein), that the only logical choices for performing such steps of "rendering an interpretation" and "constructing a navigation query" would be either "*on a computing device located locally with the user*" or "*on a network computing device located remotely from the user*". Thus, to the extent that Kupiec, Cheyer, or Kupiec in view of Cheyer (as described herein) were considered to not disclose the additional limitations of this claim element with respect to performing recited steps "*at the one or more network servers*" (i.e. "*on a network computing device located remotely from the user*"), in my opinion, a POSITA at the time of the alleged invention would already be knowledgeable regarding such a topic and would find it obvious to try such an approach in view of the finite number of possibilities (effectively only two) and the high predictability that such an approach would lead to a successful outcome.

644. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, because the additional limitations of this claim element were well known to a POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

'718 Patent: Claim 12

> 12. The computer program of claim 10, wherein the rendering of the interpretation of the spoken request is performed by the mobile information appliance.

12. The computer program of claim 10, wherein the rendering of the interpretation of the spoken request is performed by the mobile information appliance.

645. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 10 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 492-625 above.

646. Kupiec specifically discloses that "*IR subsystem* ***40*** *can be located at the same site as processor* ***10*** or can be located at a remote site and connected to processor **10** via a suitable communication network" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

647. According to Kupiec, "*System* ***1*** *comprises a processor* ***10*** coupled to an input audio transducer **20**, an output visual display **30**, an optional output speech synthesizer **31**, and an information retrieval (IR) subsystem **40** which accesses documents from corpus **41** using a word index **42**" as well as "*a phonetic transcriber* ***50****, a hypothesis generator* ***60***, a phonetic index **62**, a *query constructor* ***70***, and a scoring mechanism **80**" (emphasis added, see, for example, Ex. 1013 at 5:43-51).

648. Kupiec describes that "*Processor* ***10*** *is a computer processing unit (CPU)*" that typically is "*part of a* mainframe*, workstation, or personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

649. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

650. Kupiec further discloses that the system "has been *demonstrated on a Sun SparcStation 10 workstation*" and that "Discrete-word speech can be input using a Sennheiser HMD414 headset microphone and a Rane MS-1 preamplifier, *with signal processing performed in software by the SparcStation*" so that such "*Input speech is transcribed* into a phone sequence using hidden Markov model methods" as exemplified in a prior art 1989 IEEE paper (emphasis added, see, for example, Ex. 1013 at 29:45-46, 30:48-56). In my opinion, a POSITA would

understand such a SparcStation 10 workstation with a headset microphone discrete-word speech input at the time of the alleged invention to be an example of a computing device that can be located locally with the user, and therefore would inherently disclose the limitation that such rendering be performed at "the mobile information appliance" to the extent that "the mobile information appliance" is the computing device, such as a personal computer or a personal digital assistant, that is located locally with the user as in the case of Kupiec or Kupiec in view of Cheyer as discussed herein.

651. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

652. Thus, Kupiec discloses the "computer program of claim 10" (as described herein), wherein "the rendering the interpretation of the spoken request is performed by the mobile information appliance" (at least when the phonetic transcriber is implemented on a workstation or a portable personal computer or personal digital assistant with a headset microphone input that would normally be located locally with the user).

653. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

654. However, to the extent that alleged non-disclosure of the antecedent "computer program of claim 10" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element or alleged non-disclosure of the antecedent "the mobile information appliance" were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "computer program of claim 10" and "the mobile information appliance" of this

claim element (see, for example, ¶¶ 492-625 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "computer program of claim 10" and "the mobile information appliance" (see, for example, ¶¶ 645-653 above).

655. Cheyer describes its system as implemented with an "*Open Agent Architecture (OAA)*" that "*provides a framework for coordinating a society of agents which interact to solve problems for the user*" and "*provides distributed access to* commercial applications, such as mail systems, calendar programs, *databases*, etc." (emphasis added, see, for example, Ex. 1019 at p. 6).

656. Cheyer also describes that "In the *Open Agent Architecture, agents are distributed entities that can run on different machines*, and communicate together to solve a task for the user" (see, for example, Ex. 1019 at p. 8). More specifically Cheyer discloses "Macro Agents", which "contain some knowledge and ability to reason about a domain, *and can answer or make queries to other macro agents using the Interagent Communication Language*", and "Micro Agents", which "are responsible for handling a single input or output data stream, either filtering the signal to or from a hierarchically superior 'interpret' agent" (emphasis added, see, for example, Ex. 1019 at p. 8).

657. According to Cheyer, the "network architecture" used was "hierarchical at two resolutions" wherein "micro agents are connected to a superior macro agent" and "macro agents are connected in turn to a facilitator agent" but "In both cases, *a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

658. Cheyer also specifically describes a "*Speech Recognition (SR) Agent*" that "provides a mapping from the Interagent Communication Language to the API for the Decipher (Corona) speech recognition system", which Cheyer describes as "a continuous speech speaker

independent recognizer based on Hidden Markov Model technology" (emphasis added, see, for example, Ex. 1019 at p. 8). Cheyer further specifically describes a "Natural Language (NL) Parser Agent" that "translates English expressions into the Interagent Communication Language (ICL)" (emphasis added, see, for example, Ex. 1019 at p. 8).

659. Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either *a microphone or a telephone for voice input*" (emphasis added, see, for example, Ex. 1019 at p. 5). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

660. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

661. Thus, Cheyer discloses the "computer program of claim 10" (as described herein), wherein "the rendering the interpretation of the spoken request is performed by a **computing device**" (at least when the speech recognition, parser and database agents are implemented on a server or different machines) but not specifically where such rendering is performed by a "**computing device**" that is "*the mobile information appliance*". However, Cheyer does not teach away from or preclude such implementation on a "**computing device**" that is "*the mobile information appliance*" as disclosed by Cheyer, and thereby does not provide any motivation not to combine Cheyer with Kupiec with respect to this claim element, or otherwise.

662. Additionally, see ¶ 351 above.

663. At least because Kupiec discloses the additional limitations of this claim element, and Cheyer discloses some of the additional limitations of this claim element, because the additional limitations of this claim element were well known to a POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of

this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

664. Kimura "relates to a *remote control system for remotely controlling various electronic devices*, and *more particularly to a remote control system for remotely controlling devices such as AV (audio visual) devices by way of voice commands*" including such "AV devices" as "*television receivers*" (emphasis added, see, for example, Ex. 1015 at 1:8-14).

665. Kimura discloses a "general *remote control system*" that "comprises a *transmitter **101** for transmitting a remote control signal from a position remote from a controlled device **103** such as an AV device*, and a receiver **102** for receiving the transmitted remote control signal, decoding the remote control signal, and sending the decoded information to the controlled device **103**" (emphasis added, see, for example, Ex. 1015 at 3:10-15).

666. More specifically, Kimura discloses that "FIG. 3 is a block diagram of the transmitter of a general voice-operated remote control system", wherein the "*transmitter **101** has a microphone **M** for converting a voice command* into an electric signal" that "is *applied to a speech recognition circuit **15** in the form of a speech recognition LSI circuit* or the like which includes a microprocessor" and "*produces command data corresponding to the recognized contents*", and further wherein "The transmitter **101** also has a controller **16** comprising a microprocessor" (emphasis added, see, for example, Ex. 1015 at 3:27-36).

667. Kimura describes in reference to FIG. 4 that "*transmitter **10A** of the voice-operated remote control system* has a unitary casing **11** which *allows the operator to carry the transmitter freely around*" (emphasis added, see, for example, Ex. 1015 at 4:9-12).

668. Kimura describes in reference to FIG. 7 that "the *speech recognition circuit **15A** comprises* an analog processor **21** for processing an analog voice command signal which is received through the microphone **M** and outputting the processed analog voice command signal as a time-division digital data **20**, *a speech recognition processor **22** for recognizing the voice command* based on the time-division digital data **20** from the analog processor **21**, a memory **23A** for storing standard pattern data for speech recognition, and an interface **24** for transmitting signals to and receiving signals from the controller **16A**" (emphasis added, see, for example, Ex. 1015 at 5:3-18).

669. See also, for example, ¶¶ 178, 180, 184, 187 and 189 above with respect to Kimura and this claim element.

670. Thus, Kimura discloses "rendering an interpretation of a spoken request" that is "performed by a mobile information appliance" (at least when the handheld portable transmitter with microphone uses a speech recognition processor within such transmitter to render an interpretation of a spoken request).

671. Although Kupiec in view of Cheyer renders obvious the limitations of this claim element per my analysis herein under the proper interpretation, to the extent that the disclosures of Kupiec in view of Cheyer regarding this claim element were considered to not include the recited step of this claim element, then in my opinion, a POSITA would have modified the system of Kupiec in view Cheyer further in view of the system of Kimura specifically to arrive at a combination that meets the limitations of this claim element under the proper interpretation (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

672. First, see ¶ 267 above.

673. Second, see ¶ 362 above.

674. Third, see ¶ 269 above.

675. Fourth, see ¶ 270 above.

676. Fifth, see ¶ 365 above.

677. Therefore, in my opinion, Kupiec in view of Cheyer further in view of Kimura discloses the limitations of this claim element for the alternative construction of this claim element as described herein.

‘718 Patent: Claim 13

> 13. The computer program of claim 10, further comprising a code segment that solicits additional input from the user, including user interaction in a modality different than the original request; a code segment that refines the navigation query, based upon the additional input; and a code segment that uses the refined navigation query to select a portion of the electronic data source.

13. The computer program of claim 10, further comprising a code segment that solicits additional input from the user, including user interaction in a modality different than the original request; a code segment that refines the navigation query, based upon the additional input; and a code segment that uses the refined navigation query to select a portion of the electronic data source.

678. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 10 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 492-625 above.

13(a) a code segment that solicits additional input from the user, including user interaction in a modality different than the original request;

679. With respect to FIG. 2, Kupiec discloses that “*transcriber **50** is error-prone and produces a phonetic transcription **250** that is imperfect*” (emphasis added, see, for example, Ex. 1013 at 9:35-37) and that “Because the transcriber **50** is known to be error-prone, *hypothesis generator **60** develops alternative possible transcriptions for each word spoken by the user*, in addition to the original phone sequences provided by transcriber **50**” (emphasis added, see, for example, Ex. 1013 at 9:38-61), thereby occasionally causing “*hypothesis generator **60***” to “halt processing of the user's question and *prompt the user to repeat the question*” (emphasis added, see, for example, Ex. 1013 at 11:1-9).

680. Kupiec also discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search based on the retrieved results*" (emphasis added, see, for example, Ex. 1013 at 19:32-36). Thus, Kupiec notes that "the *best matching documents that correspond at any time* to the words that the user has spoken so far *can be displayed to the user on a screen*" such that "Upon seeing the titles (or other descriptive content) the *user can speak additional words to direct the search to particular documents or cause them to be excluded* by invoking the NOT operation" (emphasis added, see, for example, Ex. 1013 at 19:64-20:2). See also ¶ 121 above.

681. Kupiec discloses that "*Section 8* concerns an embodiment in which the *input can take forms besides speech*" (emphasis added, see, for example, Ex. 1013 at 20:42-44), and describes "Section 8" in reference to FIG. 11 that "illustrates a specific embodiment of the invention that is *adaptable to a range of input sources*, transcription techniques, hypothesis generation techniques, information retrieval techniques, and analysis techniques" (emphasis added, see, for example, Ex. 1013 at 23:19-25).

682. Kupiec also discloses in reference to FIG. 11 that "In operation, *transducer **220** accepts an input question **301*** and converts it into a signal **320**" wherein "*input question **301** can be a spoken utterance*, in which case *transducer **220** comprises audio signal processing equipment* that converts the spoken utterance to signal **320**" as well as other input modalities such as "*handwritten*" or "*typewritten*" wherein "*transducer **220** comprises a digitizing tablet* or input-sensitive display screen as is typical of pen-based computers" or "*transducer **220** comprises a conventional computer keyboard*" (emphasis added, see, for example, Ex. 1013 at 24:22-41).

683. With respect to FIG. 11, Kupiec additionally discloses that "*Analyzer/evaluator **280** provides the hypothesis or hypotheses most likely to correctly interpret question **301*** and, if appropriate, query results relevant to this hypothesis or hypotheses, *as an interpretation **400*** that is output via output channel **230**" wherein such "hypotheses can be represented, for example, as ASCII text", thereby enabling that "*Output channel **230** can send interpretation **400** to be displayed using a visual display* **231**" such that "If the appropriate command keywords are supported, the *user can provide relevance feedback based on displayed or speech-synthesized output*" in order "to facilitate the understanding of the inputs that the user provides as relevance feedback" (see, for example, Ex. 1013 at 26:66-27:18).

684. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, *hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, *for query construction*, and for scoring" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

685. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

686. Thus, Kupiec discloses a "code segment" (at least the analyzer/evaluator and/or the hypothesis generator and query constructor that can be implemented as software modules executing on a processor) that "solicits additional input from the user" (at least when the system

prompts a user to repeat a question or when the system accepts additional words provided as relevance feedback to direct a search) that is "including user interaction in a **non-spoken** modality" (at least because the system accepts user inputs in the form of handwritten or typewritten modalities).

687. Kupiec does not explicitly disclose the totality of this claim element because Kupiec's disclosed functionality for "soliciting additional input from the user, including user interaction in a **non-spoken** modality" does not necessarily require that such "**non-spoken** modality" be "*different* than the original request" as recited by this claim limitation. However, Kupiec does not teach away or exclude the case wherein Kupiec's disclosed functionality for "soliciting additional input from the user, including user interaction in a **non-spoken** modality" would be "*different* than the original request". Instead, Kupiec is silent with respect to this particular limitation within this claim element, and thereby does not provide any motivation not to combine Cheyer with Kupiec with respect to this claim element, or otherwise.

688. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

689. Cheyer states that "*direct manipulation and natural language* seem to be very *complementary modalities*" and further that "It is therefore not surprising that a number of multimodal systems combine the two" including Cheyer's description that "A number of systems have focused on *combining the speed of speech with* the reference provided by *direct manipulation of a mouse pointer*" (emphasis added, see, for example, Ex. 1019 at pp. 3-4).

690. Cheyer describes a "*system*" that "permits the user to *simultaneously combine direct manipulation*, gestural drawings, handwritten, typed *and spoken natural language*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

691. In reference to Fig. 1, Cheyer discloses that "the user is presented with a *pen sensitive map display on which drawn gestures* and written natural language statements *may be combined with spoken input*" such that "content presented by the map change, according to the requests of the user" and the "*user may ask the map to perform various actions*" (emphasis added, see, for example, Ex. 1019 at p. 5).

692. According to Cheyer, this "system" has "*modality agents*" that are "*connected to an 'interpret agent'* which is responsible for *combining the inputs across all modalities* to form a valid command for the application" wherein this "interpret agent receives filtered results from the modality agents, sorts the information into the correct fields, performs type-checking on the arguments, *and prompts the user for any missing information*" (emphasis added, see, for example, Ex. 1019 at p. 7).

693. Cheyer further discloses an "*Interface Agent*" as "responsible for managing what is currently being displayed to the user, and *for accepting the user's multimodal input*" wherein this "Interface Agent also coordinates client modality agents and resolves ambiguities among them" such that "*handwriting and gestures are* interpreted locally by micro agents and *combined with results from the speech recognition agent*, running on a remote speech server" (emphasis added, see, for example, Ex. 1019 at p. 9).

694. Additionally, Cheyer notes that "An important task for the *interface agent* is to record which objects of each type are currently salient, in order *to resolve contextual references* such as "the hotel" or "where I was before"" wherein such "Deictic references are resolved *by*

gestural or direct manipulation commands" and further wherein "*If no such indication is currently specified, the user interface agent* waits long enough to give the user an opportunity to supply the value, and then *prompts the user for it*" (emphasis added, see, for example, Ex. 1019 at p. 9).

695. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

696. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

697. Thus, Cheyer discloses a "code segment" (at least the user interface software agent and/or the reference resolution and modality software agents) that "solicits additional input from the user" (at least when the system prompts a user to provide missing information) that is "including user interaction in modality different than the original request" (at least because the system prompts the user that had provided spoken input for additional information to be returned by non-spoken modalities such as handwriting, gestures or direct manipulation by mouse pointer or typing).

698. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

699. In my opinion, a POSITA would have modified the system of Kupiec in view of the system of Cheyer specifically to arrive at a combination that meets the limitations of this claim element for at least the following reasons.

700. First, see ¶ 232 above.

701. Second, see ¶ 233 above.

702. Third, see ¶ 390 above.

703. Fourth, see ¶ 391 above.

704. Fifth, see ¶ 392 above.

705. Sixth, see ¶ 393 above.

706. Seventh, see ¶ 394 above.

707. Therefore, in my opinion, Kupiec in view of Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

13(b) a code segment that refines the navigation query, based upon the additional input;

708. See ¶ 83 above regarding claim construction for this claim element.

709. Kupiec describes that "Depending on the results obtained from execution of the initial query, *additional queries can be constructed and executed, in a process called query reformulation*" in order to "send the query thus modified back to IR subsystem **40** to be executed again" (emphasis added, see, for example, Ex. 1013 at 11:42-48).

710. Kupiec also states that "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "*The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query*" (emphasis added, see, for example, Ex. 1013 at 15:1-12).

711. Kupiec discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up*

search based on the retrieved results" (emphasis added, see, for example, Ex. 1013 at 19:32-36). See also ¶ 121 above.

712. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, *hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, *for query construction*, and for scoring" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

713. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

714. Thus, Kupiec discloses a "code segment" (at least the analyzer/evaluator and/or the hypothesis generator and query constructor that can be implemented as software modules executing on a processor) that "refines the navigation query" (at least by the query reformulation process that is used for searching the information retrieval subsystem) wherein such step is "based upon the additional input" (at least when the user provides relevance feedback commands that cause the system to perform a follow-up search).

715. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

716. However, to the extent that non-disclosure of the specifically-limited antecedent "the additional input" of this claim element as discussed above for Kupiec were considered to

constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the specifically-limited antecedent "the additional input" of this claim element (see, for example, ¶¶ 699-707 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the additional input" (see, for example, ¶¶ 708-715 above).

717. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

718. Cheyer discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*" (emphasis added, see, for example, Ex. 1019 at pp. 8-9).

719. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

720. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

721. Thus, Cheyer discloses a "code segment" (at least the reference resolution software agent and/or user interface, database and modality software agents) that "refines the navigation query" (at least by the interactions of the reference resolution, user interface, database and modality agents as described herein) wherein such step is "based upon the additional input" (at least when the user provides additional input such as gesture following an original spoken request).

722. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

723. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

13(c) a code segment that uses the refined navigation query to select a portion of the electronic data source.

724. See ¶ 83 above regarding claim construction for this claim element.

725. Kupiec describes that "Depending on the results obtained from execution of the initial query, *additional queries can be constructed and executed, in a process called query reformulation*" in order to "*send the query thus modified back to IR subsystem* ***40*** *to be executed again*" (emphasis added, see, for example, Ex. 1013 at 11:42-48).

726. Kupiec also states that "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "*The initial query can be modified and re-run once, many times, or not*

at all, depending on the results obtained at each from executing each intermediate query" (emphasis added, see, for example, Ex. 1013 at 15:1-12).

727. Kupiec discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search* based on the retrieved results" (emphasis added, see, for example, Ex. 1013 at 19:32-36). See also ¶ 121 above.

728. Kupiec further discloses that "*IR subsystem **40*** incorporates a processor that *can process queries to search for documents in corpus **41***" (emphasis added, see, for example, Ex. 1013 at 6:22-25). According to Kupiec, "a series of queries **270** is constructed by query constructor **70** and provided to IR subsystem **40**, which executes them by *conducting searches in accordance with queries* **270** over corpus **41**" such that "The execution of the initial and any additional *queries causes a set of documents **240** to be retrieved from corpus **41***" (emphasis added, see, for example, Ex. 1013 at 11:53-60).

729. Kupiec discloses in reference to FIG. 9 that "A relevant subset of the hypotheses and retrieved documents is presented to the user (Step KK). If documents have previously been retrieved, then user relevance feedback commands and search terms can be routed to the hypothesis generator, to instruct the hypothesis generator to use retrieved document titles as the basis for confirming hypotheses (Step LL), or to cease doing this upon a "new search" or similar command. The *system then can perform operations such as a vector space search or the selection of one among several preferred hypotheses* (Step MM). Results of these operations are presented to the user (Step KK)" (emphasis added, see, for example, Ex. 1013 at 20:21-32).

730. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, *hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, *for query construction, and for scoring*" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

731. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

732. Thus, Kupiec discloses a "code segment" (at least the analyzer/evaluator and/or query constructor that can be implemented as software modules executing on a processor in combination with the information retrieval subsystem software that conducts searches) that "uses the refined navigation query to select a portion of the electronic data source" (at least by searching the information retrieval subsystem or a subset thereof with the relevance feedback modified query wherein the documents from the information retrieval subsystem search are selected via a preferred hypothesis based on the user's relevance feedback commands).

733. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

734. However, to the extent that alleged non-disclosure of the antecedent "the refined navigation query" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim

element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the refined navigation query" of this claim element (see, for example, ¶ 716 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the refined navigation query" (see, for example, ¶¶ 724-733 above).

735. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

736. According to Cheyer, "*Database Agents*" can "*reside at local or remote locations and can be grouped hierarchically according to content*" wherein such "databases" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the World Wide Web (WWW)" as well as "information" that is "extracted by an HTML reading database agent" such as a "list of current movie times and reviews" (emphasis added, see, for example, Ex. 1019 at p. 8).

737. Cheyer discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*" (emphasis added, see, for example, Ex. 1019 at pp. 8-9).

738. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent*

uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

739. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

740. Thus, Cheyer discloses a "code segment" (at least the database and/or reference resolution software agents) that "uses the refined navigation query to select a portion of the electronic data source" (at least by sending database requests after user resolution of an ambiguous reference wherein such refined database requests return the selected information about the items in question).

741. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

742. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

'718 Patent: Claim 15

> 15. The computer program of claim 10, wherein code segments (a)-(d) are executed with respect to multiple users.

15. The computer program of claim 10, wherein code segments (a)-(d) are executed with respect to multiple users.

743. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 10 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 492-625 above.

744. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer" including "*transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems*" (emphasis added, see, for example, Ex. 1013 at 1:36-45).

745. Kupiec notes that "The *general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to *handwriting recognition in pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

746. Kupiec describes that "Processor **10** is a computer processing unit (CPU)" that typically is "part of a mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

747. Kupiec discloses that "*Display **30** provides visual output to the user*", which may be of the form of "alphanumeric display of the texts or titles", such as for "documents retrieved from corpus **41**" and typically comprises "a *computer screen or monitor*" (emphasis added, see, for example, Ex. 1013 at 6:12-15).

748. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

749. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

750. Thus, Kupiec discloses the "computer program of claim 10" (as described herein), wherein "code segments (a)-(d) are executed with respect to multiple users" (at least when the system operates with multiple computerized information-retrieval (IR) systems and multiple computers and personal digital assistants over a suitable communications network).

751. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

752. However, to the extent that alleged non-disclosure of the antecedent "computer program of claim 10" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "computer program of claim 10" of this claim element (see, for example, ¶¶ 492-625 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "computer program of claim 10" (see, for example, ¶¶ 743-751 above).

753. Cheyer describes "Direct manipulation interface technologies" as comprising "the use of menus and a graphical user interface" such that "*users* are presented with sets of discrete actions and the objects on which to perform them" (see, for example, Ex. 1019 at p. 2) and further notes that "Gestures allow *users* to communicate a surprisingly wide range of meaningful requests with a few simple strokes" (emphasis added, see, for example, Ex. 1019 at p. 3).

754. Cheyer describes the system design criteria as including a "*user interface*" that is "light and fast enough to *run on a handheld PDA while able to access applications and data that may require a more powerful machine*" (emphasis added, see, for example, Ex. 1019 at p. 5).

Similarly, Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" (emphasis added, see, for example, Ex. 1019 at p. 5).

755. Cheyer also discloses that "The *interface is connected either by modem or ethernet to a server machine* which will manage database access, natural language processing and speech recognition for the application" such that "The *result is a mobile system that provides a synergistic pen/voice interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

756. Cheyer describes that "In the Open Agent Architecture, *agents are distributed entities that can run on different machines, and communicate together to solve a task for the user*" and "*a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

757. According to Cheyer, "*Database Agents*" can "*reside at local or remote locations* and can be grouped hierarchically according to content" wherein such "*databases*" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the *World Wide Web (WWW)*" as well as a "*Reference Resolution Agent*" that is "*responsible for merging requests arriving in parallel*" (emphasis added, see, for example, Ex. 1019 at p. 8).

758. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

759. Thus, Cheyer discloses the "computer program of claim 10" (as described herein), wherein "code segments (a)-(d) are executed with respect to multiple users" (at least because the described mobile system supports multiple users operating multiple handheld computers

connected via modem or Ethernet to remote databases managed by agents distributed on multiple different machines to handle multiple requests arriving in parallel).

760. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

761. Additionally, see ¶ 444 above.

762. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, because the additional limitations of this claim element were well known to a POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

763. Freeman describes that "This invention is a new model and *system for managing personal electronic information*" wherein "*streams and filters provide a unified framework* that subsumes many separate desktop applications to accomplish and handle personal communication, scheduling, and *search and retrieval tasks*" (emphasis added, see, for example, Ex. 1014 at 3:62-4:2). Freeman states that "in accordance with the present invention *users can access their personal document streams from any available platform such as* a UNIX machine, a Macintosh or IBM-compatible *personal computer*, a *personal digital assistant (PDA)*, or a *set-top box via cable*" (emphasis added, see, for example, Ex. 1014 at 2:56-61).

764. Freeman also describes an "embodiment of the present invention" that "is implemented in *a client/server architecture running over the Internet*" as well as embodiments that implement "a *client viewport using graphically based X Windows*", "a *client viewport solely*

with text in standard ASCII", and "a *client viewport for the NEWTON personal digital assistant (PDA)*" (emphasis added, see, for example, Ex. 1014 at 6:8-9, 6:17-23).

765. Freeman specifically discloses that "*A stream according to the present invention can be controlled by a voice-interface* as well as a computer and thereby be *accessed via a conventional phone*" wherein this "*voice interface would allow: (1) the stream to be searched and manipulated*; (2) new objects to be installed; (3) objects to be transferred; and (4) other capability" (emphasis added, see, for example, Ex. 1014 at 11:38-43).

766. Freeman observes that "A *stream is a data structure that can be* examined and to the extent possible *manipulated by many processes simultaneously*" wherein "A *stream must support simultaneous access* because: (1) a *user creates many software agents which may need to examine the stream concurrently*; and (2) a user may have granted other users limited access to the user's stream, and the user will want access to this stream even while the other users access the stream" (emphasis added, see, for example, Ex. 1014 at 13:50-52, 13:59-64).

767. Freeman further discloses that "One embodiment of the *present invention is configured such that each server may support three to four simultaneous users with stream sizes on the order of 100,000 documents* (perhaps a year or two of documents for the average user)" (emphasis added, see, for example, Ex. 1014 at 13:65-14:4). Additionally, Freeman discloses "embodiments of the present invention utilize a *multi-server and multi-threaded approach which provides a more scalable architecture*" (emphasis added, see, for example, Ex. 1014 at 14:19-21).

768. Thus, Freeman discloses a voice-driven navigation system for information retrieval that is relevant to the "computer program of claim 10", and wherein software agents and threads are "executed with respect to multiple users" (at least because the described system

operates with multiple simultaneous users, each with a corresponding client device such as a personal computer, PDA, or set-top box via cable).

769. In my opinion, a POSITA would have modified the system of Kupiec in view of Cheyer further in view of the system of Freeman specifically to arrive at a combination that meets the limitations of this claim element (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

770. First, see ¶ 453 above.

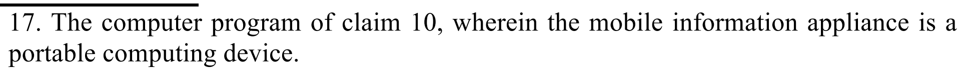
771. Second, see ¶ 454 above.

772. Third, see ¶ 455 above.

773. Fourth, see ¶ 456 above.

774. Therefore, in my opinion, Kupiec in view of Cheyer further in view of Freeman discloses the limitations of this claim element under the proper interpretation proposed herein.

'718 Patent: Claim 17



17. The computer program of claim 10, wherein the mobile information appliance is a portable computing device.

17. The computer program of claim 10, wherein the mobile information appliance is a portable computing device.

775. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 10 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 492-625 above.

776. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer" including "*transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems*" (emphasis added, see, for example, Ex. 1013 at 1:36-45).

777. Kupiec notes that "The *general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to *handwriting recognition in pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

778. Kupiec describes that "Processor **10** is a computer processing unit (CPU)" that typically is "part of a mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

779. Kupiec discloses that "*Display **30** provides visual output to the user*", which may be of the form of "alphanumeric display of the texts or titles", such as for "documents retrieved from corpus **41**" and typically comprises "a *computer screen or monitor*" (emphasis added, see, for example, Ex. 1013 at 6:12-15).

780. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

781. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

782. Thus, Kupiec discloses the "computer program of claim 10" (as described herein), wherein "the mobile information appliance is a portable computing device" (at least when the system operates with personal computers and personal digital assistants).

783. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

784. However, to the extent that alleged non-disclosure of the antecedent “computer program of claim 10” of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent “computer program of claim 10” of this claim element (see, for example, ¶¶ 492-625 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent “computer program of claim 10” (see, for example, ¶¶ 775-783 above).

785. Cheyer describes the system design criteria as including a “*user interface*” that is “light and fast enough to *run on a handheld PDA while able to access applications and data that may require a more powerful machine*” (emphasis added, see, for example, Ex. 1019 at p. 5). Similarly, Cheyer describes the system as also having a “*user interface*” that “*runs on pen-equipped PC's or a Dauphin handheld PDA*” using “either a microphone or a telephone for voice input” (emphasis added, see, for example, Ex. 1019 at p. 5). In my opinion, a POSITA at the time of the alleged invention of the ‘718 Patent would understand Cheyer’s disclosure of “PDA” to mean “personal digital assistant”.

786. Cheyer also discloses that “The *interface is connected either by modem or ethernet to a server machine* which will manage database access, natural language processing and speech recognition for the application” such that “The *result is a mobile system that provides a synergistic pen/voice interface to remote databases*” (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

787. Cheyer describes that "In the Open Agent Architecture, *agents are distributed entities that can run on different machines, and communicate together to solve a task for the user*" and "*a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

788. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

789. Thus, Cheyer discloses the "computer program of claim 10" (as described herein), wherein "the mobile information appliance is a portable computing device" (at least because the described mobile system supports users operating handheld computers or personal digital assistants).

790. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

791. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

'718 Patent: Claim 18

18. The computer program of claim 17, wherein the portable computing device is a personal digital assistant.

18. The computer program of claim 17, wherein the portable computing device is a personal digital assistant.

792. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 17 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 775-791 above.

793. As Kupiec summarizes in its “Background of the Invention” section, the Kupiec patent “relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer” including “*transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems*” (emphasis added, see, for example, Ex. 1013 at 1:36-45).

794. Kupiec notes that “The *general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to *handwriting recognition in pen-based computers and personal digital assistants*” (emphasis added, see, for example, Ex. 1013 at 1:56-67).

795. Kupiec describes that “Processor **10** is a computer processing unit (CPU)” that typically is “part of a mainframe, workstation, or *personal computer*” but can comprise “multiple processing elements in some embodiments” (emphasis added, see, for example, Ex. 1013 at 5:52-55).

796. Kupiec discloses that “*Display **30** provides visual output to the user*”, which may be of the form of “alphanumeric display of the texts or titles”, such as for “documents retrieved from corpus **41**” and typically comprises “a *computer screen or monitor*” (emphasis added, see, for example, Ex. 1013 at 6:12-15).

797. Kupiec specifically discloses that “*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*” (emphasis added, see, for example, Ex. 1013 at 6:25-28).

798. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

799. Thus, Kupiec discloses the "computer program of claim 17" (as described herein), wherein "the portable computing device is a personal digital assistant" (at least when the system operates with personal digital assistants).

800. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

801. However, to the extent that alleged non-disclosure of the antecedent "computer program of claim 17" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "computer program of claim 17" of this claim element (see, for example, ¶¶ 775-791 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "computer program of claim 17" (see, for example, ¶¶ 792-800 above).

802. Cheyer describes the system design criteria as including a "*<u>user interface</u>*" that is "light and fast enough to *<u>run on a handheld PDA while able to access applications and data that may require a more powerful machine</u>*" (emphasis added, see, for example, Ex. 1019 at p. 5). Similarly, Cheyer describes the system as also having a "*<u>user interface</u>*" that "*<u>runs on pen-equipped PC's or a Dauphin handheld PDA</u>*" using "either a microphone or a telephone for voice input" (emphasis added, see, for example, Ex. 1019 at p. 5). In my opinion, a POSITA at the time of the alleged invention of the '718 Patent would understand Cheyer's disclosure of "PDA" to mean "personal digital assistant".

803. Cheyer also discloses that "The *interface is connected either by modem or ethernet to a server machine* which will manage database access, natural language processing and speech recognition for the application" such that "The *result is a mobile system that provides a synergistic pen/voice interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

804. Cheyer describes that "In the Open Agent Architecture, *agents are distributed entities that can run on different machines, and communicate together to solve a task for the user*" and "*a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

805. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

806. Thus, Cheyer discloses the "computer program of claim 17" (as described herein), wherein "the portable computing device is a personal digital assistant" (at least because the described mobile system supports users operating personal digital assistants).

807. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

808. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

'718 Patent: Claim 19

> 19. A system for speech-based navigation of an electronic data source located at one or more network servers located remotely from a user, comprising:

> (a) a mobile information appliance operable to receive a spoken request for desired information from the user, wherein said mobile information appliance comprises a portable remote control device or a set-top box for a television;
>
> (b) spoken language processing logic, operable to render an interpretation of the spoken request;
>
> (c) query construction logic, operable to construct a navigation query based upon the interpretation;
>
> (d) navigation logic, operable to select a portion of the electronic data source using the navigation query, and
>
> (e) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to the mobile information appliance of the user.

19. A system for speech-based navigation of an electronic data source located at one or more network servers located remotely from a user, comprising:

809. In my opinion, this preamble claim element is a claim limitation at least because I believe that this preamble recites essential structure and/or steps that give life, meaning, and vitality to the claim as opposed to <u>*only*</u> stating a purpose or intended use for the alleged invention. See also ¶¶ 78-81 above.

810. See ¶ 84 above regarding claim construction for this claim element.

811. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "<u>*relates to systems and methods for*</u> transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer" including "transcription systems and methods appropriate for use in conjunction with computerized <u>*information-retrieval (IR) systems*</u> and methods" and "more particularly to <u>*speech-recognition systems and methods appropriate for use in conjunction with computerized information-retrieval systems*</u>" (emphasis added, see, for example, Ex. 1013 at 1:36-45).

812. Kupiec notes that "<u>*The general problem*</u> of disambiguating the words contained in an error-prone transcription of user input <u>*arises in a number of contexts*</u> beyond speech recognition, <u>*including*</u> but not limited to handwriting recognition in <u>*pen-based computers and personal digital assistants*</u>" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

813. Kupiec describes that "*Processor **10** is* a computer processing unit (CPU)" that typically is "*part of a* mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

814. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

815. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

816. Thus, Kupiec discloses a "system for speech-based navigation of an electronic data source" (the operation of a speech-recognition system in conjunction with an information-retrieval system) wherein such an "electronic data source" (the information-retrieval system or IR subsystem) is "located at one or more network servers located remotely from a user" (at least when the IR subsystem is located at a remote site and connected to the speech-recognition system via a suitable communication network).

817. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

818. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" (emphasis added, see, for example, Ex. 1019 at p. 2).

819. As Cheyer summarizes in its "Conclusions" section, "By augmenting an existing agent-based architecture with concepts necessary for synergistic multimodal input", Cheyer

discloses "a mobile, synergistic pen/*voice interface providing good natural language access to heterogeneous distributed knowledge sources*" (emphasis added, see, for example, Ex. 1019 at p. 10).

820. Cheyer specifically discloses that "*the system permits the user* to simultaneously combine direct manipulation, gestural drawings, handwritten, typed and *spoken natural language*" and that the system uses "Existing commercial or research natural language and *speech recognition systems*", thereby enabling "a user" to "transparently *access* a wide variety of data sources, including *information stored in HTML form on the World Wide Web*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

821. Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" wherein "The *interface is connected either by modem or ethernet to a server machine which will manage database access, natural language processing and speech recognition* for the application" such that "The result is a *mobile system that provides* a synergistic pen/voice *interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

822. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

823. Thus, Cheyer discloses a "system for speech-based navigation of an electronic data source" (the operation of a speech-recognition and navigation system in conjunction with remote databases and/or the World Wide Web) wherein such an "electronic data source" (such as remote databases and/or the World Wide Web) is "located at one or more network servers

located remotely from a user" (at least because the World Wide Web is located across numerous network servers remote from any individual user and remote databases are remote).

824. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

825. At least because each of Kupiec and Cheyer discloses the limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

19(a) a mobile information appliance operable to receive a spoken request for desired information from the user, wherein said mobile information appliance comprises a portable remote control device or a set-top box for a television;

826. See ¶ 84 above regarding claim construction for this claim element.

827. In reference to FIG. 1, Kupiec describes that "Transducer **20** converts a *user's spoken utterance into a signal that can be processed* by processor **10**" and can comprise "*a microphone coupled to an analog-to-digital converter, so that the user's speech is converted by transducer **20** into a digital signal*" (emphasis added, see, for example, Ex. 1013 at 5:56-6:7).

828. Also, in reference to FIG. 2, Kupiec describes that "The *user inputs a question **201** into system **1** by speaking into audio transducer **20***" (emphasis added, see, for example, Ex. 1013 at 9:18-23). FIG. 2 of Kupiec and its detailed description explicitly illustrate that "question **201**" is not only "spoken" but is also a "request for desired information" to be found by "a series of queries" provided "to IR subsystem **40**, which executes them by conducting searches in accordance with queries **270** over corpus **41**" such that "The execution of the initial and any additional queries causes a set of documents **240** to be retrieved from corpus **41**" (see, for example, Ex. 1013 at 11:53-60, and see also, for example, ¶¶ 106-112 above).

829. Similarly, in reference to FIG. 11, Kupiec describes that "In operation, *transducer **220** accepts an input question **301*** and converts it into a signal **320**" wherein "*input question **301** can be a spoken utterance, in which case transducer **220** comprises audio signal processing equipment* that converts the spoken utterance to signal **320**" (emphasis added, see, for example, Ex. 1013 at 24:22-41).

830. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

831. Kupiec describes that "*Processor **10** is* a computer processing unit (CPU)" that typically is "*part of a* mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

832. Kupiec further discloses that the system "has been demonstrated on a Sun SparcStation 10 workstation" and that "*Discrete-word speech can be input using a Sennheiser HMD414 headset microphone* and a Rane MS-1 preamplifier, *with signal processing performed in software by the SparcStation*" so that such "Input speech is transcribed into a phone sequence using hidden Markov model methods" as exemplified in a prior art 1989 IEEE paper (emphasis added, see, for example, Ex. 1013 at 29:45-46, 30:48-56).

833. See also, for example, ¶¶ 94, 106, 114, 115, 125 and 129 above with respect to Kupiec and this claim element.

834. Thus, Kupiec discloses a "mobile information appliance" (such as a personal computer that can be a portable personal digital assistant) that is "operable to receive a spoken

request" (at least when the microphone/transducer connected to such computer accepts a spoken input question) that is "for desired information" (such as the material to be searched and retrieved from the IR subsystem) and that is "from the user" (at least when the input question is a user's spoken utterance), and wherein "said mobile information appliance comprises a portable remote control device or a set-top box for a television" (at least because in a speech recognition controlled system a portable headset microphone constitutes a portable remote control device).

835. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

836. In the section entitled "A Multimodal Map Application", Cheyer describes "a prototype map-based application for a travel planning domain" wherein "the *system permits the user to* simultaneously combine direct manipulation, gestural drawings, handwritten, typed and *spoken natural language*" and this system uses "Existing commercial or research natural language and *speech recognition systems*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

837. In reference to Fig. 1, Cheyer discloses that "the *user*" is "*presented with a* pen sensitive map *display*" and can provide "*spoken input*" and the "*user may ask the map to perform various actions*" such as "*information retrieval*" (emphasis added, see, for example, Ex. 1019 at p. 5).

838. Cheyer further notes that "The application also makes use of *multimodal (multimedia) output* as well as input: *video, text, sound and voice can all be combined when presenting an answer to a query*" (emphasis added, see, for example, Ex. 1019 at p. 5).

839. Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either *a microphone or a telephone for*

voice input" (emphasis added, see, for example, Ex. 1019 at p. 5). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

840. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

841. Thus, Cheyer discloses a "mobile information appliance" (such as a handheld PDA) that is "operable to receive a spoken request" (at least when the microphone or telephone connected to such handheld PDA accepts spoken or voice input) that is "for desired information" (such as for information retrieval) and that is "from the user" (at least when the spoken input is provided by a user), and wherein "said mobile information appliance comprises a portable remote control device or a set-top box for a television" (at least because the handheld PDA is itself a portable remote control device for controlling access to the information retrieval system).

842. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

843. To the extent that Kupiec's disclosures are considered to not include the recited "mobile information appliance" of this claim (or any of its elements below, or its dependent claims herein), then in my opinion, a POSITA would have modified the system of Kupiec in view of the system of Cheyer specifically to arrive at a combination that meets the limitations of this claim element (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

844. First, see ¶ 232 above.

845. Second, see ¶ 233 above.

846. Third, see ¶ 234 above.

847. Fourth, see ¶ 235 above.

848. Fifth, see ¶ 236 above.

849. Sixth, see ¶ 256 above.

850. Therefore, in my opinion, Kupiec in view of Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

851. Kimura "relates to a *remote control system for remotely controlling various electronic devices*, and *more particularly to a remote control system for remotely controlling devices such as AV (audio visual) devices by way of voice commands*" including such "AV devices" as "*television receivers*" (emphasis added, see, for example, Ex. 1015 at 1:8-14). Additionally, Kimura is directed to a "*voice-operated remote control system which can vary a speech recognition process depending* on the degree of importance of a control command" in view of the fact that "the magnitudes of *effects caused by erroneous recognition*, may not necessarily be the same" (emphasis added, see, for example, Ex. 1015 at 1:41-42, 1:54-57).

852. Kimura discloses a "general *remote control system*" that "comprises a *transmitter **101** for transmitting a remote control signal from a position remote from a controlled device **103** such as an AV device*, and a receiver **102** for receiving the transmitted remote control signal, decoding the remote control signal, and sending the decoded information to the controlled device **103**" (emphasis added, see, for example, Ex. 1015 at 3:10-15).

853. More specifically, Kimura discloses that "FIG. 3 is a block diagram of the transmitter of a general voice-operated remote control system", wherein the "*transmitter **101** has a microphone **M** for converting a voice command* into an electric signal" that "is *applied to a speech recognition circuit **15*** in the form of a speech recognition LSI circuit or the like *which includes a microprocessor*" and "*produces command data corresponding to the recognized*

contents", and further wherein "The transmitter **101** also has a controller **16** comprising a microprocessor" (emphasis added, see, for example, Ex. 1015 at 3:27-36).

854. Kimura describes in reference to FIG. 4 that "*transmitter **10A** of the voice-operated remote control system* has a unitary casing **11** which *allows the operator to carry the transmitter freely around*" (emphasis added, see, for example, Ex. 1015 at 4:9-12). Kimura further describes that "casing **11** supports a microphone **M** on an upper panel thereof" wherein "microphone **M** converts a voice command given by the operator into an electric signal" and on "one side of the casing **11**, there is disposed a *voice input switch* (hereinafter referred to as a "talk switch") **12** which is closed when pressed and can automatically be released or opened when released" so that when a "voice command is to be entered, the talk switch **12** is closed to operate the transmitter **10A**" or "Otherwise, the talk switch **12** is open keeping the transmitter **10A** out of operation" (emphasis added, see, for example, Ex. 1015 at 4:12-28).

855. Kimura describes in reference to FIG. 7 that "the *speech recognition circuit **15A** comprises* an analog processor **21** for processing an analog voice command signal which is received through the microphone **M** and outputting the processed analog voice command signal as a time-division digital data **20**, *a speech recognition processor **22** for recognizing the voice command* based on the time-division digital data **20** from the analog processor **21**, a memory **23A** for storing standard pattern data for speech recognition, and an interface **24** for transmitting signals to and receiving signals from the controller **16A**" (emphasis added, see, for example, Ex. 1015 at 5:3-18).

856. Additionally, Kimura describes in reference to FIG. 8 that "the *speech recognition processor **22** comprises a system controller **40*** for analyzing and processing control commands from the controller **16** and also for controlling the entire operation of the speech recognition

processor **22**, and a digital processor **41** for effecting distance calculations and controlling the memory **23A**" wherein the "*system controller **40** comprises a CPU (Central Processing Unit) **42** for controlling the overall operation of the transmitter* **1**" (emphasis added, see, for example, Ex. 1015 at 6:21-30).

857. See also, for example, ¶¶ 178, 180, 184, 187 and 189 above with respect to Kimura and this claim element.

858. Thus, Kimura discloses "a portable remote control device" that is "for a television" (such as the portable transmitter with a CPU that provides a remote control signal to a television receiver) and is part of a system for "receiving a spoken request" (at least because the transmitter includes a microphone and a speech recognition processor).

859. Although Kupiec in view of Cheyer renders obvious the limitations of this claim element per my analysis herein under the proper interpretation, to the extent that the limitation of "said mobile information appliance comprises a portable remote control device or a set-top box for a television" were construed to mean "said mobile information appliance comprises a *portable remote control device* or a set-top box *for a television*", as opposed to a "*portable remote control device*" that can be for purposes that may not necessarily be "*for a television*", then in my opinion, a POSITA would have modified the system of Kupiec in view Cheyer further in view of the system of Kimura specifically to arrive at a combination that meets the limitations of this claim element under this alternative claim construction (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

860. First, see ¶ 267 above.

861. Second, see ¶ 268 above.

862. Third, see ¶ 269 above.

863. Fourth, see ¶ 270 above.

864. Fifth, see ¶ 271 above.

865. Therefore, in my opinion, Kupiec in view of Cheyer further in view of Kimura discloses the limitations of this claim element for the alternative construction of this claim element as described herein.

19(b) spoken language processing logic, operable to render an interpretation of the spoken request;

866. In reference to FIG. 2, Kupiec describes that "*signal* ***220*** *produced by transducer* ***20*** *is fed to transcriber* ***50****, where it is converted into a phonetic transcription* ***250***" using "any of a variety of transcription techniques" that were "well-known among those of skill in the art" (emphasis added, see, for example, Ex. 1013 at 9:18-23). Kupiec explains that "*phonetic transcription* ***250*** *is an ordered sequence* of phones, that is, *of component sounds that can be used to form words*" (emphasis added, see, for example, Ex. 1013 at 9:29-31).

867. Also, in reference to FIG. 3, Kupiec describes that "*First the system accepts a user utterance* as input (Step A)" and that "This *utterance is converted to a signal (Step B) that is transcribed into a sequence of phones* (Step C)" (emphasis added, see, for example, Ex. 1013 at 12:48-57).

868. Similarly, in reference to FIG. 11, Kupiec describes that "Transducer **220** provides signal **320** to transcriber **250**" such that "*Transcriber* ***250*** *converts signal* ***320*** *to a string* ***350*** *that represents a transcription of the input question* ***301***" (emphasis added, see, for example, Ex. 1013 at 24:66-25:4).

869. Kupiec discloses in reference to FIG. 11 that "*processor* ***200*** *executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "*Transcriber* ***250****,* hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator

280 are *typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21).

870. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

871. Thus, Kupiec discloses "spoken language processing logic" (at least the transcriber that can be implemented as a software module executing on a processor) that is "operable to render an interpretation" (at least when the transcriber produces a phonetic transcription of component sounds that can be used to form words) that is "of the spoken request" (at least because the transcriber operates upon the user's spoken utterance).

872. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

873. However, to the extent that alleged non-disclosure of the antecedent "the spoken request" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the spoken request" of this claim element (see, for example, ¶¶ 843-850 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the spoken request" (see, for example, ¶¶ 866-872 above).

874. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; access to existing data sources including the World Wide Web; and a mobile handheld interface" that is implemented using "a

hierarchical distributed network of heterogeneous software agents" (emphasis added, see, for example, Ex. 1019 at p. 2).

875. In the section entitled "A Multimodal Map Application", Cheyer describes "a prototype map-based application for a travel planning domain" wherein "the *system permits the user to* simultaneously combine direct manipulation, gestural drawings, handwritten, typed and *spoken natural language*" and this system uses "*Existing commercial* or research natural language and *speech recognition systems*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

876. Cheyer also specifically describes, from a 1990 prior art article, a "*Speech Recognition (SR) Agent*" that "provides a mapping from the Interagent Communication Language to the API *for the Decipher (Corona) speech recognition system*", which Cheyer describes as "*a continuous speech speaker independent recognizer based on Hidden Markov Model technology*" (emphasis added, see, for example, Ex. 1019 at p. 8). Cheyer further specifically describes, with reference to a 1994 prior art article, a "*Natural Language (NL) Parser Agent*" that "*translates English expressions into the Interagent Communication Language (ICL)*" (emphasis added, see, for example, Ex. 1019 at p. 8).

877. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

878. Thus, Cheyer discloses "spoken language processing logic" (at least the speech recognition and/or natural language parser software agents in combination with the Decipher/Corona speech recognition system) that is "operable to render an interpretation" (at least when speech recognition system and parser creates natural language English expressions)

that is "of the spoken request" (at least because the speech recognition system and parser operates upon the user's spoken natural language).

879. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

880. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

19(c) query construction logic, operable to construct a navigation query based upon the interpretation;

881. See ¶ 83 above regarding claim construction for this claim element.

882. In reference to FIG. 2, Kupiec describes that "The *phonetic transcription **250** is provided to hypothesis generator **60*** where it is matched using phonetic index **62** *to generate a set of hypotheses **260***" (emphasis added, see, for example, Ex. 1013 at 9:38-61) and further that "Once the set of *hypotheses **260*** has been generated, it is *provided to query constructor **70***" such that "*Query constructor **70*** uses the hypotheses **260** *to construct one or more queries **270** that will be sent to IR subsystem **40** for execution*" (emphasis added, see, for example, Ex. 1013 at 11:10-13).

883. Additionally, Kupiec specifically discloses that "Queries **270** are Boolean queries with proximity and order constraints" and provides a specific example wherein the "user speaks two words" that lead to a "set of hypotheses" such "an initial *query*" is constructed that "*seeks occurrences of* at least one of the words (*search terms*)" and is "*sent to the IR subsystem **40*** where it is executed" (emphasis added, see, for example, Ex. 1013 at 11:13-41).

884. Also, in reference to FIG. 3, Kupiec describes that "First the system accepts a user utterance" that is "transcribed" and then "*<u>used to generate hypotheses</u>* (Step D)" such that "*<u>Boolean queries with proximity and order constraints are constructed based on the hypotheses and are executed to retrieve documents</u>* (Step E)" (emphasis added, see, for example, Ex. 1013 at 12:48-57).

885. Similarly, in reference to FIG. 11, Kupiec describes that "Transducer **220** provides signal **320** to transcriber **250**" that "converts signal **320** to a string **350** that represents a transcription of the input question **301**" such that "Hypothesis generator **260** converts string **350** and any alternatives to a set of hypotheses **360**" provided to "*<u>Query/IR mechanism</u>* **270**" that "*<u>converts the hypotheses **360** to one or more information retrieval queries</u>* **370**" that are "in a format *<u>that can be searched</u>* by processor **200** (or a separate IR processor that communicates with processor **200**) using corpus **241**" (emphasis added, see, for example, Ex. 1013 at 24:66-25:61).

886. Kupiec discloses in reference to FIG. 11 that "*<u>processor **200** executes software</u>* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, *<u>hypothesis generator **260**, query/IR mechanism **270**,</u>* and analyzer/evaluator **280** *<u>are typically implemented as software modules that are part of software</u>* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*<u>source code</u>*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, *<u>for query construction</u>*, and for scoring" and the "second file includes source code *<u>for hypothesis generation</u>*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

887. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

888. Thus, Kupiec discloses "query construction logic" (at least the hypothesis generator and the query constructor that can be implemented as software modules executing on a processor) that is "operable to construct a navigation query" (at least when the query constructor constructs one or more queries that can be used to search the IR subsystem) that is "based upon the interpretation" (at least because the query constructor operates upon the hypotheses formed from the transcription of the user's spoken utterance).

889. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

890. However, to the extent that alleged non-disclosure of the antecedent "the interpretation" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the interpretation" of this claim element (see, for example, ¶ 873 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the interpretation" (see, for example, ¶¶ 881-889 above).

891. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

892. In reference to Figure 1, Cheyer discloses that "the *user*" can provide "*spoken input*" and the "*user may ask the map to perform various actions*" such as "*information retrieval*" (emphasis added, see, for example, Ex. 1019 at p. 5).

893. Cheyer further describes that the system can "capture signals emitted during a user's interaction" and also "integrates a set of modality agents, each responsible for a very specialized kind of signal" wherein the "modality agents are connected to an '*interpret agent*' which is *responsible for combining the inputs across all modalities to form a valid command* for the application" (emphasis added, see, for example, Ex. 1019 at p. 7).

894. Cheyer discloses "Database Agents" wherein exemplary "*databases*" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning *HTML pages from the World Wide Web (WWW)*" as well as "information" that is "extracted by an HTML reading database agent" (emphasis added, see, for example, Ex. 1019 at p. 8).

895. Cheyer also discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*" wherein this "reference resolution agent (RR)" in an exemplary embodiment "sends database requests asking for the coordinates of the items in question" (emphasis added, see, for example, Ex. 1019 at pp. 8-10).

896. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

897. Thus, Cheyer discloses "query construction logic" (at least the reference resolution software agent and/or user interface, database and modality software agents) that is "operable to construct a navigation query" (at least when the reference resolution agent sends database requests for data on the World Wide Web) that is "based upon the interpretation" (at

least because the reference resolution agent creates database requests from the output of agents that respond to spoken input).

898. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

899. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

19(d) navigation logic, operable to select a portion of the electronic data source using the navigation query,

900. See ¶ 83 above regarding claim construction for this claim element.

901. Kupiec describes that "Depending on the results obtained from execution of the initial query, *additional queries can be constructed and executed, in a process called query reformulation*" in order to "*send the query thus modified back to IR subsystem* ***40*** *to be executed again*" (emphasis added, see, for example, Ex. 1013 at 11:42-48).

902. Kupiec also states that "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "*The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query*" (emphasis added, see, for example, Ex. 1013 at 15:1-12).

903. Kupiec discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up*

search based on the retrieved results" (emphasis added, see, for example, Ex. 1013 at 19:32-36). See also ¶ 121 above.

904. Kupiec further discloses that "*IR subsystem **40*** incorporates a processor that *can process queries to search for documents in corpus **41***" (emphasis added, see, for example, Ex. 1013 at 6:22-25). According to Kupiec, "a series of queries **270** is constructed by query constructor **70** and provided to IR subsystem **40**, which executes them by *conducting searches in accordance with queries* **270** over corpus **41**" such that "The execution of the initial and any additional *queries causes a set of documents **240** to be retrieved from corpus **41***" (emphasis added, see, for example, Ex. 1013 at 11:53-60).

905. Kupiec discloses in reference to FIG. 9 that "A relevant subset of the hypotheses and retrieved documents is presented to the user (Step KK). If documents have previously been retrieved, then user relevance feedback commands and search terms can be routed to the hypothesis generator, to instruct the hypothesis generator to use retrieved document titles as the basis for confirming hypotheses (Step LL), or to cease doing this upon a "new search" or similar command. The *system then can perform operations such as a vector space search or the selection of one among several preferred hypotheses* (Step MM). Results of these operations are presented to the user (Step KK)" (emphasis added, see, for example, Ex. 1013 at 20:21-32).

906. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, *hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp"

wherein the "first file includes source code for reading a phonetic index file, *for query construction, and for scoring*" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

907. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

908. Thus, Kupiec discloses "navigation logic" (at least the analyzer/evaluator and/or query constructor that can be implemented as software modules executing on a processor in combination with the information retrieval subsystem that incorporates a processor to conduct searches) that is "operable to select a portion of the electronic data source using the navigation query" (at least by searching the information retrieval subsystem or a subset thereof with the relevance feedback modified query wherein the documents from the information retrieval subsystem search are selected via a preferred hypothesis based on the user's relevance feedback commands).

909. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

910. However, to the extent that alleged non-disclosure of the antecedent "the navigation query" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the navigation query" of this claim element (see, for example, ¶ 890 above) and Kupiec discloses the subsequent additional limitations recited by this

claim element in view of the antecedent "the navigation query" (see, for example, ¶¶ 900-909 above).

911. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

912. Cheyer discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*" (emphasis added, see, for example, Ex. 1019 at pp. 8-9).

913. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

914. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

915. Thus, Cheyer discloses "navigation logic" (at least the database and/or reference resolution software agents) that is "operable to select a portion of the electronic data source using the navigation query" (at least by sending database requests after user resolution of an

ambiguous reference wherein such refined database requests return the selected information about the items in question).

916. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

917. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

19(e) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to the mobile information appliance of the user.

918. See ¶ 84 above regarding claim construction for this claim element.

919. Kupiec describes that "*Processor* ***10*** *is* a computer processing unit (CPU)" that typically is "*part of a* mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

920. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor* ***10*** *via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

921. Kupiec discloses that "*Display* ***30*** *provides visual output to the user*" such as for "*documents retrieved from corpus* ***41***" and typically comprises "*a computer screen* or monitor", and further that "*Corpus* ***41*** *comprises a database of documents* that can be searched by IR subsystem **40**" wherein such documents comprise "for example, *books, articles from newspapers and periodicals, encyclopedia articles, abstracts, office documents*, etc." (emphasis added, see,

for example, Ex. 1013 at 6:12-15, 29-33). Similarly, Kupiec additionally discloses that "*Output channel* ***230*** *can send interpretation* ***400*** *to be displayed using a visual display* ***231***" in order "to facilitate the understanding of the inputs that the *user provides as relevance feedback*" (emphasis added, see, for example, Ex. 1013 at 27:7-18).

922. In reference to FIG. 3 of Kupiec, "A *relevant subset of the* hypotheses and *retrieved documents is presented to the user* (Step G)" (emphasis added, see, for example, Ex. 1013 at 12:48-57). Similarly, in reference to FIG. 9 of Kupiec, "The *system* then *can perform operations such as a vector space search or the selection of one among several preferred hypotheses* (Step MM)" such that "*Results of these operations are presented to the user* (Step KK)" (emphasis added, see, for example, Ex. 1013 at 20:21-32).

923. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

924. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

925. Thus, Kupiec discloses "electronic communications infrastructure" (at least the suitable communication network that connects the information retrieval subsystem at a remote site to the personal computer or personal digital assistant of the user) for "transmitting the selected portion of the electronic data source from the network server to the mobile information appliance of the user" (at least by retrieving documents from an information retrieval subsystem at a remote site over a communications network and displaying the selected documents on a computer screen such as that of a personal digital assistant).

926. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

927. However, to the extent that alleged non-disclosure of the antecedent "the selected portion" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element or alleged non-disclosure of the antecedent "the mobile information appliance" were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the selected portion" and "the mobile information appliance" of this claim element (see, for example, ¶¶ 843-850 above and ¶ 910 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the selected portion" and "the mobile information appliance" (see, for example, ¶¶ 918-926 above).

928. Cheyer is "distinguished by a synergistic combination of handwriting, gesture and speech modalities; *access to existing data sources including the World Wide Web*; and a *mobile handheld interface*" (emphasis added, see, for example, Ex. 1019 at p. 2). Cheyer further describes the system as enabling "Through the multimodal interface" that "a user" can "*transparently access a wide variety of data sources, including information stored in HTML form on the World Wide Web*" and Cheyer describes the system as also having a "user interface" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" (emphasis added, see, for example, Ex. 1019 at p. 5). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

929. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a*

gesture indicating "the hotel", issuing prompts if necessary" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" and then subsequently "*requests the user interface to produce output displaying the result* of the calculation" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

930. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

931. Thus, Cheyer discloses "electronic communications infrastructure" (at least the modem or ethernet that connects data sources such as the World Wide Web to the user's computer with monitor or handheld PDA) for "transmitting the selected portion of the electronic data source from the network server to the mobile information appliance of the user" (at least by retrieving information in databases on the World Wide Web that gets displayed on the user interface of mobile pen-equipped personal computers such as the Dauphin handheld PDA).

932. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

933. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

'718 Patent: Claim 20

> 20. The system of claim 19, wherein the spoken language processing logic renders the interpretation of the spoken request at the one or more network servers.

20. The system of claim 19, wherein the spoken language processing logic renders the interpretation of the spoken request at the one or more network servers.

934. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 19 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 809-933 above.

935. According to Kupiec, "*System **1** comprises a processor **10*** coupled to an input audio transducer **20**, an output visual display **30**, an optional output speech synthesizer **31**, and an information retrieval (IR) subsystem **40** which accesses documents from corpus **41** using a word index **42**" as well as "*a phonetic transcriber **50**, a hypothesis generator **60***, a phonetic index **62**, a query constructor **70**, and a scoring mechanism **80**" (emphasis added, see, for example, Ex. 1013 at 5:43-51).

936. Kupiec describes that "*Processor **10** is a computer processing unit (CPU)*" that typically is "*part of a mainframe*, workstation, or personal *computer*" but can comprise "*multiple processing elements in some embodiments*" (emphasis added, see, for example, Ex. 1013 at 5:52-55). In my opinion, a POSITA at the time of alleged invention would consider an operation performed in a "mainframe computer" to be an example of at least performing such an operation "at the one or more network servers".

937. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "*Transcriber **250**, hypothesis generator **260**,* query/IR mechanism **270**, and analyzer/evaluator **280** *are typically implemented as software modules that are part of software* **205** and are *executed by processor* **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, for query

construction, and for scoring" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

938. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

939. Thus, Kupiec discloses the "system of claim 19" (as described herein), wherein the "spoken language processing logic" (at least the transcriber that can be implemented as a software module executing on a processor) "renders the interpretation of the spoken request at the one or more network servers" (at least when the phonetic transcriber and hypothesis generator are implemented on a mainframe computer that would normally be located remotely from the user such as at one or more network servers).

940. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

941. However, to the extent that alleged non-disclosure of the antecedent "system of claim 19" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element or alleged non-disclosure of the antecedent "the interpretation" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "system of claim 19" and "the interpretation" of this claim element (see, for example, ¶¶ 809-933 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "system of claim 19" and "the interpretation" (see, for example, ¶¶ 934-940 above).

942. Cheyer describes the system as also having a "*user interface*" that "runs *on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" wherein "The interface is *connected either by modem or ethernet to a server machine* which will *manage database access, natural language processing and speech recognition* for the application" (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

943. Cheyer describes its system as implemented with an "*Open Agent Architecture (OAA)*" that "*provides a framework for coordinating a society of agents which interact to solve problems for the user*" and "*provides distributed access to* commercial applications, such as mail systems, calendar programs, *databases*, etc." (emphasis added, see, for example, Ex. 1019 at p. 7).

944. Cheyer also describes that "In the *Open Agent Architecture, agents are distributed entities that can run on different machines*, and communicate together to solve a task for the user" (see, for example, Ex. 1019 at p. 8). More specifically Cheyer discloses "Macro Agents", which "contain some knowledge and ability to reason about a domain, *and can answer or make queries to other macro agents using the Interagent Communication Language*", and "Micro Agents", which "are responsible for handling a single input or output data stream, either filtering the signal to or from a hierarchically superior 'interpret' agent" (emphasis added, see, for example, Ex. 1019 at p. 8).

945. According to Cheyer, the "network architecture" used was "hierarchical at two resolutions" wherein "micro agents are connected to a superior macro agent" and "macro agents are connected in turn to a facilitator agent" but "In both cases, *a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

946. Cheyer also specifically describes a "*Speech Recognition (SR) Agent*" that "provides a mapping from the Interagent Communication Language to the API for the Decipher (Corona) speech recognition system", which Cheyer describes as "a continuous speech speaker independent recognizer based on Hidden Markov Model technology" (emphasis added, see, for example, Ex. 1019 at p. 8). Cheyer further specifically describes a "Natural Language (NL) Parser Agent" that "translates English expressions into the Interagent Communication Language (ICL)" (emphasis added, see, for example, Ex. 1019 at p. 8).

947. Cheyer also specifically discloses the "*Speech Recognition (SR) Agent*" and the "*Reference Resolution (RR) Agent*" as examples of "macro agents" (see, for example, Ex. 1019 at p. 9, Figure 3).

948. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

949. Thus, Cheyer discloses the "system of claim 19" (as described herein), wherein the "spoken language processing logic" (at least the speech recognition and/or natural language parser software agents in combination with the Decipher/Corona speech recognition system) "renders the interpretation of the spoken request at the one or more network servers" (at least when the speech recognition, language parser and database agents are implemented on a server or different machines that are networked together via a modem or ethernet).

950. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

951. Additionally, see ¶ 643 above.

952. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, because the additional limitations of this claim element were well known to a

POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

<u>'718 Patent: Claim 21</u>

> 21. The system of claim 19, wherein the spoken language processing logic renders the interpretation of the spoken request at the mobile information appliance.

21. The system of claim 19, wherein the spoken language processing logic renders the interpretation of the spoken request at the mobile information appliance.

953. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 19 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 809-933 above.

954. Kupiec specifically discloses that "<u>*IR subsystem **40** can be located at the same site as processor **10***</u> or can be located at a remote site and connected to processor **10** via a suitable communication network" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

955. According to Kupiec, "<u>*System **1** comprises a processor **10***</u> coupled to an input audio transducer **20**, an output visual display **30**, an optional output speech synthesizer **31**, and an information retrieval (IR) subsystem **40** which accesses documents from corpus **41** using a word index **42**" as well as "<u>*a phonetic transcriber **50**, a hypothesis generator **60***</u>, a phonetic index **62**, a <u>*query constructor **70***</u>, and a scoring mechanism **80**" (emphasis added, see, for example, Ex. 1013 at 5:43-51).

956. Kupiec describes that "<u>*Processor **10** is a computer processing unit (CPU)*</u>" that typically is "<u>*part of a*</u> mainframe<u>*, workstation, or personal computer*</u>" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

957. Kupiec notes that "*The general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to handwriting recognition in *pen-based computers and personal digital assistants*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

958. Kupiec further discloses that the system "has been *demonstrated on a Sun SparcStation 10 workstation*" and that "Discrete-word speech can be input using a Sennheiser HMD414 headset microphone and a Rane MS-1 preamplifier, *with signal processing performed in software by the SparcStation*" so that such "*Input speech is transcribed* into a phone sequence using hidden Markov model methods" as exemplified in a prior art 1989 IEEE paper (emphasis added, see, for example, Ex. 1013 at 29:45-46, 30:48-56). In my opinion, a POSITA would understand such a SparcStation 10 workstation with a headset microphone discrete-word speech input at the time of the alleged invention to be an example of a computing device that can be located locally with the user, and therefore would inherently disclose the limitation that such rendering be performed at "the mobile information appliance" to the extent that "the mobile information appliance" is the computing device, such as a personal computer or a personal digital assistant, that is located locally with the user as in the case of Kupiec or Kupiec in view of Cheyer as discussed herein.

959. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

960. Thus, Kupiec discloses the "system of claim 19" (as described herein), wherein the "spoken language processing logic" (at least the transcriber that can be implemented as a software module executing on a processor) "renders the interpretation of the spoken request at the mobile information appliance" (at least when the phonetic transcriber is implemented on a

workstation or a portable personal computer or personal digital assistant with a headset microphone input that would normally be located locally with the user).

961. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

962. However, to the extent that alleged non-disclosure of the antecedent "system of claim 19" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element or alleged non-disclosure of the antecedent "the mobile information appliance" were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "system of claim 19" and "the mobile information appliance" of this claim element (see, for example, ¶¶ 809-933 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "system of claim 19" and "the mobile information appliance" (see, for example, ¶¶ 953-961 above).

963. Cheyer describes its system as implemented with an "*Open Agent Architecture (OAA)*" that "*provides a framework for coordinating a society of agents which interact to solve problems for the user*" and "*provides distributed access to* commercial applications, such as mail systems, calendar programs, *databases*, etc." (emphasis added, see, for example, Ex. 1019 at p. 6).

964. Cheyer also describes that "In the *Open Agent Architecture, agents are distributed entities that can run on different machines*, and communicate together to solve a task for the user" (see, for example, Ex. 1019 at p. 8). More specifically Cheyer discloses "Macro Agents", which "contain some knowledge and ability to reason about a domain, *and can answer or make*

queries to other macro agents using the Interagent Communication Language", and "Micro Agents", which "are responsible for handling a single input or output data stream, either filtering the signal to or from a hierarchically superior 'interpret' agent" (emphasis added, see, for example, Ex. 1019 at p. 8).

965. According to Cheyer, the "network architecture" used was "hierarchical at two resolutions" wherein "micro agents are connected to a superior macro agent" and "macro agents are connected in turn to a facilitator agent" but "In both cases, *a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

966. Cheyer also specifically describes a "*Speech Recognition (SR) Agent*" that "provides a mapping from the Interagent Communication Language to the API for the Decipher (Corona) speech recognition system", which Cheyer describes as "a continuous speech speaker independent recognizer based on Hidden Markov Model technology" (emphasis added, see, for example, Ex. 1019 at p. 8). Cheyer further specifically describes a "Natural Language (NL) Parser Agent" that "translates English expressions into the Interagent Communication Language (ICL)" (emphasis added, see, for example, Ex. 1019 at p. 8).

967. Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either *a microphone or a telephone for voice input*" (emphasis added, see, for example, Ex. 1019 at p. 5). See also ¶ 156 above for a depiction of a "Dauphin handheld PDA".

968. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

969. Thus, Cheyer discloses the "system of claim 19" (as described herein), wherein the "spoken language processing logic" (at least the speech recognition and/or natural language

parser software agents in combination with the Decipher/Corona speech recognition system) "renders the interpretation of the spoken request at a **computing device**" (at least when the speech recognition, parser and database agents are implemented on a server or different machines) but not specifically where such rendering is performed by a "**computing device**" that is "*the mobile information appliance*". However, Cheyer does not teach away from or preclude such implementation on a "**computing device**" that is "*the mobile information appliance*" as disclosed by Cheyer, and thereby does not provide any motivation not to combine Cheyer with Kupiec with respect to this claim element, or otherwise.

970. Additionally, see ¶ 351 above.

971. At least because Kupiec discloses the additional limitations of this claim element, and Cheyer discloses some of the additional limitations of this claim element, because the additional limitations of this claim element were well known to a POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

972. Kimura "relates to a *remote control system for remotely controlling various electronic devices*, and *more particularly to a remote control system for remotely controlling devices such as AV (audio visual) devices by way of voice commands*" including such "AV devices" as "*television receivers*" (emphasis added, see, for example, Ex. 1015 at 1:8-14).

973. Kimura discloses a "general *remote control system*" that "comprises a *transmitter **101** for transmitting a remote control signal from a position remote from a controlled device **103** such as an AV device*, and a receiver **102** for receiving the transmitted remote control signal,

decoding the remote control signal, and sending the decoded information to the controlled device **103**" (emphasis added, see, for example, Ex. 1015 at 3:10-15).

974. More specifically, Kimura discloses that "FIG. 3 is a block diagram of the transmitter of a general voice-operated remote control system", wherein the "*transmitter **101** has a microphone **M** for converting a voice command* into an electric signal" that "is *applied to a speech recognition circuit **15** in the form of a speech recognition LSI circuit* or the like which includes a microprocessor" and "*produces command data corresponding to the recognized contents*", and further wherein "The transmitter **101** also has a controller **16** comprising a microprocessor" (emphasis added, see, for example, Ex. 1015 at 3:27-36).

975. Kimura describes in reference to FIG. 4 that "*transmitter **10A** of the voice-operated remote control system* has a unitary casing **11** which *allows the operator to carry the transmitter freely around*" (emphasis added, see, for example, Ex. 1015 at 4:9-12).

976. Kimura describes in reference to FIG. 7 that "the *speech recognition circuit **15A** comprises* an analog processor **21** for processing an analog voice command signal which is received through the microphone **M** and outputting the processed analog voice command signal as a time-division digital data **20**, *a speech recognition processor **22** for recognizing the voice command* based on the time-division digital data **20** from the analog processor **21**, a memory **23A** for storing standard pattern data for speech recognition, and an interface **24** for transmitting signals to and receiving signals from the controller **16A**" (emphasis added, see, for example, Ex. 1015 at 5:3-18).

977. See also, for example, ¶¶ 178, 180, 184, 187 and 189 above with respect to Kimura and this claim element.

978. Thus, Kimura discloses "spoken language processing logic" (at least the speech recognition circuit and/or processor within the handheld portable transmitter) that "renders the interpretation of the spoken request at the mobile information appliance" (at least when the handheld portable transmitter with microphone uses a speech recognition processor within such transmitter to render an interpretation of a spoken request).

979. Although Kupiec in view of Cheyer renders obvious the limitations of this claim element per my analysis herein under the proper interpretation, to the extent that the disclosures of Kupiec in view of Cheyer regarding this claim element were considered to not include the recited step of this claim element, then in my opinion, a POSITA would have modified the system of Kupiec in view Cheyer further in view of the system of Kimura specifically to arrive at a combination that meets the limitations of this claim element under the proper interpretation (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

980. First, see ¶ 267 above.

981. Second, see ¶ 362 above.

982. Third, see ¶ 269 above.

983. Fourth, see ¶ 270 above.

984. Fifth, see ¶ 365 above.

985. Therefore, in my opinion, Kupiec in view of Cheyer further in view of Kimura discloses the limitations of this claim element for the alternative construction of this claim element as described herein.

<u>'718 Patent: Claim 22</u>

> 22. The system of claim 19, further comprising user interaction logic operable to solicit additional input from the user, including user interaction in a modality different than the original request; and query refining logic operable to refine the navigation query based upon the additional input; wherein the navigation logic users the refined navigation query to select a portion of the electronic data source.

22. The system of claim 19, further comprising

986. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 19 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 809-933 above.

22(a) user interaction logic operable to solicit additional input from the user, including user interaction in a modality different than the original request;

987. With respect to FIG. 2, Kupiec discloses that "*transcriber **50** is error-prone and produces a phonetic transcription **250** that is imperfect*" (emphasis added, see, for example, Ex. 1013 at 9:35-37) and that "Because the transcriber **50** is known to be error-prone, *hypothesis generator **60** develops alternative possible transcriptions for each word spoken by the user*, in addition to the original phone sequences provided by transcriber **50**" (emphasis added, see, for example, Ex. 1013 at 9:38-61), thereby occasionally causing "*hypothesis generator **60***" to "halt processing of the user's question and *prompt the user to repeat the question*" (emphasis added, see, for example, Ex. 1013 at 11:1-9).

988. Kupiec also discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search based on the retrieved results*" (emphasis added, see, for example, Ex. 1013 at 19:32-36). Thus, Kupiec notes that "the *best matching documents that correspond at any time* to the words that the user has spoken so far *can be displayed to the user on a screen*" such that "Upon seeing the titles (or other descriptive content) the *user can speak additional words to direct the search to particular documents or cause them to be excluded* by invoking the NOT operation" (emphasis added, see, for example, Ex. 1013 at 19:64-20:2). See also ¶ 121 above.

989. Kupiec discloses that "*Section 8* concerns an embodiment in which the *input can take forms besides speech*" (emphasis added, see, for example, Ex. 1013 at 20:42-44), and

describes "Section 8" in reference to FIG. 11 that "illustrates a specific embodiment of the invention that is *adaptable to a range of input sources*, transcription techniques, hypothesis generation techniques, information retrieval techniques, and analysis techniques" (emphasis added, see, for example, Ex. 1013 at 23:19-25).

990. Kupiec also discloses in reference to FIG. 11 that "In operation, *transducer **220** accepts an input question **301*** and converts it into a signal **320**" wherein "*input question **301** can be a spoken utterance*, in which case *transducer **220** comprises audio signal processing equipment* that converts the spoken utterance to signal **320**" as well as other input modalities such as "*handwritten*" or "*typewritten*" wherein "*transducer **220** comprises a digitizing tablet* or input-sensitive display screen as is typical of pen-based computers" or "*transducer **220** comprises a conventional computer keyboard*" (emphasis added, see, for example, Ex. 1013 at 24:22-41).

991. With respect to FIG. 11, Kupiec additionally discloses that "*Analyzer/evaluator **280** provides the hypothesis or hypotheses most likely to correctly interpret question **301*** and, if appropriate, query results relevant to this hypothesis or hypotheses, *as an interpretation **400*** that is output via output channel **230**" wherein such "hypotheses can be represented, for example, as ASCII text", thereby enabling that "*Output channel **230** can send interpretation **400** to be displayed using a visual display* **231**" such that "If the appropriate command keywords are supported, the *user can provide relevance feedback based on displayed or speech-synthesized output*" in order "to facilitate the understanding of the inputs that the user provides as relevance feedback" (see, for example, Ex. 1013 at 26:66-27:18).

992. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein

"Transcriber **250**, *hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, *for query construction*, and for scoring" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

993. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

994. Thus, Kupiec discloses "user interaction logic" (at least the analyzer/evaluator and/or the hypothesis generator and query constructor that can be implemented as software modules executing on a processor) that is "operable to solicit additional input from the user" (at least when the system prompts a user to repeat a question or when the system accepts additional words provided as relevance feedback to direct a search) that is "including user interaction in a **non-spoken** modality" (at least because the system accepts user inputs in the form of handwritten or typewritten modalities).

995. Kupiec does not explicitly disclose the totality of this claim element because Kupiec's disclosed functionality for "soliciting additional input from the user, including user interaction in a **non-spoken** modality" does not necessarily require that such "**non-spoken** modality" be "*different* than the original request" as recited by this claim limitation. However, Kupiec does not teach away or exclude the case wherein Kupiec's disclosed functionality for "soliciting additional input from the user, including user interaction in a **non-spoken** modality" would be "*different* than the original request". Instead, Kupiec is silent with respect to this

particular limitation within this claim element, and thereby does not provide any motivation not to combine Cheyer with Kupiec with respect to this claim element, or otherwise.

996. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" that is implemented using "a *hierarchical distributed network of heterogeneous software agents*" (emphasis added, see, for example, Ex. 1019 at p. 2).

997. Cheyer states that "*direct manipulation and natural language* seem to be very *complementary modalities*" and further that "It is therefore not surprising that a number of multimodal systems combine the two" including Cheyer's description that "A number of systems have focused on *combining the speed of speech with* the reference provided by *direct manipulation of a mouse pointer*" (emphasis added, see, for example, Ex. 1019 at pp. 3-4).

998. Cheyer describes a "*system*" that "permits the user to *simultaneously combine direct manipulation*, gestural drawings, handwritten, typed *and spoken natural language*" (emphasis added, see, for example, Ex. 1019 at pp. 4-5).

999. In reference to Fig. 1, Cheyer discloses that "the user is presented with a *pen sensitive map display on which drawn gestures* and written natural language statements *may be combined with spoken input*" such that "content presented by the map change, according to the requests of the user" and the "*user may ask the map to perform various actions*" (emphasis added, see, for example, Ex. 1019 at p. 5).

1000. According to Cheyer, this "system" has "*modality agents*" that are "*connected to an 'interpret agent'* which is responsible for *combining the inputs across all modalities* to form a valid command for the application" wherein this "interpret agent receives filtered results from the modality agents, sorts the information into the correct fields, performs type-checking on the arguments, *and prompts the user for any missing information*" (emphasis added, see, for example, Ex. 1019 at p. 7).

1001. Cheyer further discloses an "*Interface Agent*" as "responsible for managing what is currently being displayed to the user, and *for accepting the user's multimodal input*" wherein this "Interface Agent also coordinates client modality agents and resolves ambiguities among them" such that "*handwriting and gestures are* interpreted locally by micro agents and *combined with results from the speech recognition agent*, running on a remote speech server" (emphasis added, see, for example, Ex. 1019 at p. 9).

1002. Additionally, Cheyer notes that "An important task for the *interface agent* is to record which objects of each type are currently salient, in order *to resolve contextual references* such as "the hotel" or "where I was before"" wherein such "Deictic references are resolved *by gestural or direct manipulation commands*" and further wherein "*If no such indication is currently specified, the user interface agent* waits long enough to give the user an opportunity to supply the value, and then *prompts the user for it*" (emphasis added, see, for example, Ex. 1019 at p. 9).

1003. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

1004. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

1005. Thus, Cheyer discloses "user interaction logic" (at least the user interface software agent and/or the reference resolution and modality software agents) that is "operable to solicit additional input from the user" (at least when the system prompts a user to provide missing information) that is "including user interaction in modality different than the original request" (at least because the system prompts the user that had provided spoken input for additional information to be returned by non-spoken modalities such as handwriting, gestures or direct manipulation by mouse pointer or typing).

1006. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

1007. In my opinion, a POSITA would have modified the system of Kupiec in view of the system of Cheyer specifically to arrive at a combination that meets the limitations of this claim element for at least the following reasons.

1008. First, see ¶ 232 above.

1009. Second, see ¶ 233 above.

1010. Third, see ¶ 390 above.

1011. Fourth, see ¶ 391 above.

1012. Fifth, see ¶ 392 above.

1013. Sixth, see ¶ 393 above.

1014. Seventh, see ¶ 394 above.

1015. Therefore, in my opinion, Kupiec in view of Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

22(b) query refining logic operable to refine the navigation query based upon the additional input;

1016. See ¶ 83 above regarding claim construction for this claim element.

1017. Kupiec describes that "Depending on the results obtained from execution of the initial query, *additional queries can be constructed and executed, in a process called query reformulation*" in order to "send the query thus modified back to IR subsystem **40** to be executed again" (emphasis added, see, for example, Ex. 1013 at 11:42-48).

1018. Kupiec also states that "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "*The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query*" (emphasis added, see, for example, Ex. 1013 at 15:1-12).

1019. Kupiec discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search* based on the retrieved results" (emphasis added, see, for example, Ex. 1013 at 19:32-36). See also ¶ 121 above.

1020. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, *hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, *for query* *construction*, and for scoring" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

1021. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

1022. Thus, Kupiec discloses "query refining logic" (at least the analyzer/evaluator and/or the hypothesis generator and query constructor that can be implemented as software modules executing on a processor) that is "operable to refine the navigation query" (at least by the query reformulation process that is used for searching the information retrieval subsystem) wherein such step is "based upon the additional input" (at least when the user provides relevance feedback commands that cause the system to perform a follow-up search).

1023. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

1024. However, to the extent that non-disclosure of the specifically-limited antecedent "the additional input" of this claim element as discussed above for Kupiec were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the specifically-limited antecedent "the additional input" of this claim element (see, for example, ¶¶ 1007-1015 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the additional input" (see, for example, ¶¶ 1016-1023 above).

1025. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and *speech modalities*; *access to existing data sources including the World Wide Web*; and a mobile handheld interface" that is implemented using "a

hierarchical distributed network of heterogeneous software agents" (emphasis added, see, for example, Ex. 1019 at p. 2).

1026. Cheyer discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*" (emphasis added, see, for example, Ex. 1019 at pp. 8-9).

1027. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

1028. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

1029. Thus, Cheyer discloses a "query refining logic" (at least the reference resolution software agent and/or user interface, database and modality software agents) that is "operable to refine the navigation query" (at least by the interactions of the reference resolution, user interface, database and modality agents as described herein) wherein such step is "based upon the additional input" (at least when the user provides additional input such as gesture following an original spoken request).

1030. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

1031. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

22(c) wherein the navigation logic users the refined navigation query to select a portion of the electronic data source.

1032. See ¶ 83 above regarding claim construction for this claim element.

1033. Kupiec describes that "Depending on the results obtained from execution of the initial query, *additional queries can be constructed and executed, in a process called query reformulation*" in order to "*send the query thus modified back to IR subsystem* ***40*** *to be executed again*" (emphasis added, see, for example, Ex. 1013 at 11:42-48).

1034. Kupiec also states that "Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**" such that "*The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query*" (emphasis added, see, for example, Ex. 1013 at 15:1-12).

1035. Kupiec discloses the use of "*Relevance feedback commands*" wherein "After the user's question has been processed, so that documents have been retrieved and presented in response to the question, *the user has the option of directing the invention to perform a follow-up search* based on the retrieved results" (emphasis added, see, for example, Ex. 1013 at 19:32-36). See also ¶ 121 above.

1036. Kupiec further discloses that "*IR subsystem* ***40*** incorporates a processor that *can process queries to search for documents in corpus* ***41***" (emphasis added, see, for example, Ex.

1013 at 6:22-25). According to Kupiec, "a series of queries **270** is constructed by query constructor **70** and provided to IR subsystem **40**, which executes them by *conducting searches in accordance with queries* **270** over corpus **41**" such that "The execution of the initial and any additional *queries causes a set of documents **240** to be retrieved from corpus **41***" (emphasis added, see, for example, Ex. 1013 at 11:53-60).

1037. Kupiec discloses in reference to FIG. 9 that "A relevant subset of the hypotheses and retrieved documents is presented to the user (Step KK). If documents have previously been retrieved, then user relevance feedback commands and search terms can be routed to the hypothesis generator, to instruct the hypothesis generator to use retrieved document titles as the basis for confirming hypotheses (Step LL), or to cease doing this upon a "new search" or similar command. The *system then can perform operations such as a vector space search or the selection of one among several preferred hypotheses* (Step MM). Results of these operations are presented to the user (Step KK)" (emphasis added, see, for example, Ex. 1013 at 20:21-32).

1038. Kupiec discloses in reference to FIG. 11 that "*processor **200** executes software* **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**" wherein "Transcriber **250**, *hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software* **205** and are executed by processor **200**" (emphasis added, see, for example, Ex. 1013 at 24:15-21). Kupiec also provides an "Appendix" that "includes two files" of "*source code*" written in "Lisp" wherein the "first file includes source code for reading a phonetic index file, *for query construction, and for scoring*" and the "second file includes source code *for hypothesis generation*" (emphasis added, see, for example, Ex. 1013 at 29:39-54).

1039. See also, for example, ¶¶ 94, 106, 114, 117, 125 and 129 above with respect to Kupiec and this claim element.

1040. Thus, Kupiec discloses "navigation logic" (at least the analyzer/evaluator and/or query constructor that can be implemented as software modules executing on a processor in combination with the information retrieval subsystem software that conducts searches) that "users the refined navigation query to select a portion of the electronic data source" (at least by searching the information retrieval subsystem or a subset thereof with the relevance feedback modified query wherein the documents from the information retrieval subsystem search are selected via a preferred hypothesis based on the user's relevance feedback commands).

1041. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

1042. However, to the extent that alleged non-disclosure of the antecedent "the refined navigation query" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "the refined navigation query" of this claim element (see, for example, ¶ 1024 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "the refined navigation query" (see, for example, ¶¶ 1032-1041 above).

1043. As Cheyer summarizes in its "Abstract" section, Cheyer presents "a synergistic combination of handwriting, gesture and <u>*speech modalities*</u>; <u>*access to existing data sources including the World Wide Web*</u>; and a mobile handheld interface" that is implemented using "a

hierarchical distributed network of heterogeneous software agents" (emphasis added, see, for example, Ex. 1019 at p. 2).

1044. According to Cheyer, "*Database Agents*" can "*reside at local or remote locations and can be grouped hierarchically according to content*" wherein such "databases" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the World Wide Web (WWW)" as well as "information" that is "extracted by an HTML reading database agent" such as a "list of current movie times and reviews" (emphasis added, see, for example, Ex. 1019 at p. 8).

1045. Cheyer discloses a "*Reference Resolution Agent*" that is "responsible for merging requests arriving in parallel from different modalities, and *for controlling interactions between the user interface agent, database agents and modality agents*" (emphasis added, see, for example, Ex. 1019 at pp. 8-9).

1046. Cheyer also provides a specific example of operation of the system wherein "A *user speaks*: "How far is the restaurant from this hotel?"", and subsequently "The *interface agent* uses contextual structures to find what "the restaurant" refers to, and *waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary*" and then the "*reference resolution agent (RR)*" in an exemplary embodiment "*sends database requests asking for the coordinates of the items in question*" (emphasis added, see, for example, Ex. 1019 at pp. 9-10).

1047. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

1048. Thus, Cheyer discloses "navigation logic" (at least the database and/or reference resolution software agents) that "users the refined navigation query to select a portion of the electronic data source" (at least by sending database requests after user resolution of an ambiguous reference wherein such refined database requests return the selected information about the items in question).

1049. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

1050. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

‘718 Patent: Claim 24

24. The system of claim 19, wherein the system operates with respect to multiple users.

24. The system of claim 19, wherein the system operates with respect to multiple users.

1051. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 19 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 809-933 above.

1052. As Kupiec summarizes in its “Background of the Invention” section, the Kupiec patent “relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer” including “*transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems*” (emphasis added, see, for example, Ex. 1013 at 1:36-45).

1053. Kupiec notes that “The *general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to *handwriting recognition in pen-based computers and personal digital assistants*” (emphasis added, see, for example, Ex. 1013 at 1:56-67).

1054. Kupiec describes that "Processor **10** is a computer processing unit (CPU)" that typically is "part of a mainframe, workstation, or *personal computer*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

1055. Kupiec discloses that "*Display **30** provides visual output to the user*", which may be of the form of "alphanumeric display of the texts or titles", such as for "documents retrieved from corpus **41**" and typically comprises "a *computer screen or monitor*" (emphasis added, see, for example, Ex. 1013 at 6:12-15).

1056. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

1057. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

1058. Thus, Kupiec discloses the "system of claim 19" (as described herein), wherein the "system operates with respect to multiple users" (at least when the system operates with multiple computerized information-retrieval (IR) systems and multiple computers and personal digital assistants over a suitable communications network).

1059. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

1060. However, to the extent that alleged non-disclosure of the antecedent "system of claim 19" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent “system of claim 19” of this claim element (see, for example, ¶¶ 809-933 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent “system of claim 19” (see, for example, ¶¶ 1051-1059 above).

1061. Cheyer describes “Direct manipulation interface technologies” as comprising “the use of menus and a graphical user interface” such that “*users* are presented with sets of discrete actions and the objects on which to perform them” (see, for example, Ex. 1019 at p. 2) and further notes that “Gestures allow *users* to communicate a surprisingly wide range of meaningful requests with a few simple strokes” (emphasis added, see, for example, Ex. 1019 at p. 3).

1062. Cheyer describes the system design criteria as including a “*user interface*” that is “light and fast enough to *run on a handheld PDA while able to access applications and data that may require a more powerful machine*” (emphasis added, see, for example, Ex. 1019 at p. 5). Similarly, Cheyer describes the system as also having a “*user interface*” that “*runs on pen-equipped PC's or a Dauphin handheld PDA*” using “either a microphone or a telephone for voice input” (emphasis added, see, for example, Ex. 1019 at p. 5).

1063. Cheyer also discloses that “The *interface is connected either by modem or ethernet to a server machine* which will manage database access, natural language processing and speech recognition for the application” such that “The *result is a mobile system that provides a synergistic pen/voice interface to remote databases*” (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

1064. Cheyer describes that “In the Open Agent Architecture, *agents are distributed entities that can run on different machines, and communicate together to solve a task for the* *user*" and "*a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

1065. According to Cheyer, "*Database Agents*" can "*reside at local or remote locations* and can be grouped hierarchically according to content" wherein such "*databases*" can include "Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the *World Wide Web (WWW)*" as well as a "*Reference Resolution Agent*" that is "*responsible for merging requests arriving in parallel*" (emphasis added, see, for example, Ex. 1019 at p. 8).

1066. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

1067. Thus, Cheyer discloses the "system of claim 19" (as described herein), wherein the "system operates with respect to multiple users" (at least because the described mobile system supports multiple users operating multiple handheld computers connected via modem or Ethernet to remote databases managed by agents distributed on multiple different machines to handle multiple requests arriving in parallel).

1068. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

1069. Additionally, see ¶ 444 above.

1070. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, because the additional limitations of this claim element were well known to a POSITA at the time of the alleged invention, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

1071. Freeman describes that "This invention is a new model and *system for managing personal electronic information*" wherein "*streams and filters provide a unified framework* that subsumes many separate desktop applications to accomplish and handle personal communication, scheduling, and *search and retrieval tasks*" (emphasis added, see, for example, Ex. 1014 at 3:62-4:2). Freeman states that "in accordance with the present invention *users can access their personal document streams from any available platform such as* a UNIX machine, a Macintosh or IBM-compatible *personal computer*, a *personal digital assistant (PDA)*, or a *set-top box via cable*" (emphasis added, see, for example, Ex. 1014 at 2:56-61).

1072. Freeman also describes an "embodiment of the present invention" that "is implemented in *a client/server architecture running over the Internet*" as well as embodiments that implement "a *client viewport using graphically based X Windows*", "a *client viewport solely with text* in standard ASCII", and "a *client viewport for the NEWTON personal digital assistant (PDA)*" (emphasis added, see, for example, Ex. 1014 at 6:8-9, 6:17-23).

1073. Freeman specifically discloses that "*A stream according to the present invention can be controlled by a voice-interface* as well as a computer and thereby be *accessed via a conventional phone*" wherein this "*voice interface would allow: (1) the stream to be searched and manipulated*; (2) new objects to be installed; (3) objects to be transferred; and (4) other capability" (emphasis added, see, for example, Ex. 1014 at 11:38-43).

1074. Freeman observes that "A *stream is a data structure that can be* examined and to the extent possible *manipulated by many processes simultaneously*" wherein "A *stream must support simultaneous access* because: (1) a *user creates many software agents which may need* *to examine the stream concurrently*; and (2) a user may have granted other users limited access to the user's stream, and the user will want access to this stream even while the other users access the stream" (emphasis added, see, for example, Ex. 1014 at 13:50-52, 13:59-64).

1075. Freeman further discloses that "One embodiment of the *present invention is configured such that each server may support three to four simultaneous users with stream sizes on the order of 100,000 documents* (perhaps a year or two of documents for the average user)" (emphasis added, see, for example, Ex. 1014 at 13:65-14:4). Additionally, Freeman discloses "embodiments of the present invention utilize a *multi-server and multi-threaded approach which provides a more scalable architecture*" (emphasis added, see, for example, Ex. 1014 at 14:19-21).

1076. Thus, Freeman discloses a voice-driven navigation system for information retrieval that is relevant to the "system of claim 19", and wherein the "system operates with respect to multiple users" (at least because the described system operates with multiple simultaneous users, each with a corresponding client device such as a personal computer, PDA, or set-top box via cable).

1077. In my opinion, a POSITA would have modified the system of Kupiec in view of Cheyer further in view of the system of Freeman specifically to arrive at a combination that meets the limitations of this claim element (or any other claims or claim elements of the '718 Patent) for at least the following reasons.

1078. First, see ¶ 453 above.

1079. Second, see ¶ 454 above.

1080. Third, see ¶ 455 above.

1081. Fourth, see ¶ 456 above.

1082. Therefore, in my opinion, Kupiec in view of Cheyer further in view of Freeman discloses the limitations of this claim element under the proper interpretation proposed herein.

<u>'718 Patent: Claim 26</u>

> 26. The system of claim 19, wherein the mobile information appliance is a portable computing device.

26. The system of claim 19, wherein the mobile information appliance is a portable computing device.

1083. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 19 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 809-933 above.

1084. As Kupiec summarizes in its "Background of the Invention" section, the Kupiec patent "relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer" including "*<u>transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems</u>*" (emphasis added, see, for example, Ex. 1013 at 1:36-45).

1085. Kupiec notes that "The *<u>general problem</u>* of disambiguating the words contained in an error-prone transcription of user input *<u>arises in a number of contexts</u>* beyond speech recognition, *<u>including</u>* but not limited to *<u>handwriting recognition in pen-based computers and personal digital assistants</u>*" (emphasis added, see, for example, Ex. 1013 at 1:56-67).

1086. Kupiec describes that "Processor **10** is a computer processing unit (CPU)" that typically is "part of a mainframe, workstation, or *<u>personal computer</u>*" but can comprise "multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

1087. Kupiec discloses that "*<u>Display **30** provides visual output to the user</u>*", which may be of the form of "alphanumeric display of the texts or titles", such as for "documents retrieved from corpus **41**" and typically comprises "a *computer screen or monitor*" (emphasis added, see, for example, Ex. 1013 at 6:12-15).

1088. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

1089. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

1090. Thus, Kupiec discloses the "system of claim 19" (as described herein), wherein "the mobile information appliance is a portable computing device" (at least when the system operates with personal computers and personal digital assistants).

1091. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

1092. However, to the extent that alleged non-disclosure of the antecedent "system of claim 19" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "system of claim 19" of this claim element (see, for example, ¶¶ 809-933 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "system of claim 19" (see, for example, ¶¶ 1083-1091 above).

1093. Cheyer describes the system design criteria as including a "*user interface*" that is "light and fast enough to *run on a handheld PDA while able to access applications and data that may require a more powerful machine*" (emphasis added, see, for example, Ex. 1019 at p. 5).

Similarly, Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" (emphasis added, see, for example, Ex. 1019 at p. 5). In my opinion, a POSITA at the time of the alleged invention of the '718 Patent would understand Cheyer's disclosure of "PDA" to mean "personal digital assistant".

1094. Cheyer also discloses that "The *interface is connected either by modem or ethernet to a server machine* which will manage database access, natural language processing and speech recognition for the application" such that "The *result is a mobile system that provides a synergistic pen/voice interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

1095. Cheyer describes that "In the Open Agent Architecture, *agents are distributed entities that can run on different machines, and communicate together to solve a task for the user*" and "*a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

1096. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

1097. Thus, Cheyer discloses the "system of claim 19" (as described herein), wherein "the mobile information appliance is a portable computing device" (at least because the described mobile system supports users operating handheld computers or personal digital assistants).

1098. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

1099. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

‘718 Patent: Claim 27

> 27. The system of claim 26, wherein the portable computing device is a personal digital assistant.

27. The system of claim 26, wherein the portable computing device is a personal digital assistant.

1100. Each of Kupiec in view of Cheyer and Kupiec in view of Cheyer further in view of Kimura renders obvious the recited Claim 26 of this claim element under the proper interpretation for at least the reasons summarized in ¶¶ 1083-1099 above.

1101. As Kupiec summarizes in its “Background of the Invention” section, the Kupiec patent “relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer” including “*transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems*” (emphasis added, see, for example, Ex. 1013 at 1:36-45).

1102. Kupiec notes that “The *general problem* of disambiguating the words contained in an error-prone transcription of user input *arises in a number of contexts* beyond speech recognition, *including* but not limited to *handwriting recognition in pen-based computers and personal digital assistants*” (emphasis added, see, for example, Ex. 1013 at 1:56-67).

1103. Kupiec describes that “Processor **10** is a computer processing unit (CPU)” that typically is “part of a mainframe, workstation, or *personal computer*” but can comprise “multiple processing elements in some embodiments" (emphasis added, see, for example, Ex. 1013 at 5:52-55).

1104. Kupiec discloses that "*Display **30** provides visual output to the user*", which may be of the form of "alphanumeric display of the texts or titles", such as for "documents retrieved from corpus **41**" and typically comprises "a *computer screen or monitor*" (emphasis added, see, for example, Ex. 1013 at 6:12-15).

1105. Kupiec specifically discloses that "*IR subsystem* **40** can be located at the same site as processor **10** or *can be located at a remote site and connected to processor **10** via a suitable communication network*" (emphasis added, see, for example, Ex. 1013 at 6:25-28).

1106. See also, for example, ¶¶ 94, 106, 114, 125 and 129 above with respect to Kupiec and this claim element.

1107. Thus, Kupiec discloses the "system of claim 26" (as described herein), wherein "the portable computing device is a personal digital assistant" (at least when the system operates with personal digital assistants).

1108. Therefore, in my opinion, Kupiec discloses the limitations of this claim element under the proper interpretation proposed herein.

1109. However, to the extent that alleged non-disclosure of the antecedent "system of claim 26" of this claim element for Kupiec alone in view of another antecedent limitation in a previous claim element were considered to constitute non-disclosure of this claim element by Kupiec, then Kupiec in view of Cheyer explicitly discloses the limitations of this claim element under the proper interpretation proposed herein at least because such a combination discloses the antecedent "system of claim 26" of this claim element (see, for example, ¶¶ 1083-1099 above) and Kupiec discloses the subsequent additional limitations recited by this claim element in view of the antecedent "system of claim 26" (see, for example, ¶¶ 1100-1108 above).

1110. Cheyer describes the system design criteria as including a "*user interface*" that is "light and fast enough to *run on a handheld PDA while able to access applications and data that may require a more powerful machine*" (emphasis added, see, for example, Ex. 1019 at p. 5). Similarly, Cheyer describes the system as also having a "*user interface*" that "*runs on pen-equipped PC's or a Dauphin handheld PDA*" using "either a microphone or a telephone for voice input" (emphasis added, see, for example, Ex. 1019 at p. 5). In my opinion, a POSITA at the time of the alleged invention of the '718 Patent would understand Cheyer's disclosure of "PDA" to mean "personal digital assistant".

1111. Cheyer also discloses that "The *interface is connected either by modem or ethernet to a server machine* which will manage database access, natural language processing and speech recognition for the application" such that "The *result is a mobile system that provides a synergistic pen/voice interface to remote databases*" (emphasis added, see, for example, Ex. 1019 at pp. 5-6).

1112. Cheyer describes that "In the Open Agent Architecture, *agents are distributed entities that can run on different machines, and communicate together to solve a task for the user*" and "*a server is responsible for the supervision of its client sub-agents*" (emphasis added, see, for example, Ex. 1019 at p. 8).

1113. See also, for example, ¶¶ 142, 145, 147, 148, 150, 153, 165, 171 and 173 above with respect to Cheyer and this claim element.

1114. Thus, Cheyer discloses the "system of claim 26" (as described herein), wherein "the portable computing device is a personal digital assistant" (at least because the described mobile system supports users operating personal digital assistants).

1115. Therefore, in my opinion, Cheyer discloses the limitations of this claim element under the proper interpretation proposed herein.

1116. At least because each of Kupiec and Cheyer discloses the additional limitations of this claim element, and because Kupiec in view of Cheyer discloses the antecedent limitations of this claim element, then Kupiec in view of Cheyer also renders obvious the limitations of this claim element under the proper interpretation proposed herein. Similarly, Kupiec in view of Cheyer further in view of Kimura also discloses the limitations of this claim element.

IX. CONCLUSION

1117. In my opinion, the claims of the ‘718 Patent are invalid for at least the reasons stated above.

1118. I reserve the right to supplement my opinions in the future to respond to any arguments raised by the owner of the ‘718 Patent and to take into account new information that becomes available to me.

1119. I declare under penalty of perjury that all statements made herein are of my own knowledge and are true and correct.

Respectfully submitted,

K. J. N

Date: December 20, 2017

Kevin J. Negus

ATTACHMENT A

Dr. Kevin J. Negus

Contact Information: kevin@tctwest.net, 650-472-1548
Updated: Aug. 9, 2017 522 Moose Lake Road, Philipsburg, MT, 59858

2015 – Present: Montana Tech University
Current Position: Professor, Department of Electrical Engineering
Responsibilities: Telecommunications Courses and 5G/WiFi Mobile Broadband Research

2003 – Present: Technology Consultant
Example Clients: Cisco, Nokia, Motorola, Apple, Dish, HP, Dell, Verizon, AT&T, Sprint
Responsibilities: Primarily expert witness consulting for patent litigations

2010 – 2016: Fastback Networks
Last Position Held: Co-Founder, Chairman and Chief Technology Officer
Responsibilities: System Architecture, Technology Roadmap, IP and Team Development

2004 – 2016: Camp Ventures
Last Position Held: General Partner
Responsibilities: Early Stage Investments, Product Development, Team Mentoring

2003 –2007: WiDeFi, Inc (acquired by Qualcomm in 2007).
Last Position Held: Executive Chairman
Responsibilities: Corporate Management, RF/baseband ASIC Development

1998 – 2003: Proxim Corporation
Last Position Held: Chief Technology Officer
Responsibilities: ASIC Development, Standards, M&A Deals, OEM Deals, Patent Licensing

1988 – 1998: Hewlett-Packard Company (acquired Avantek, Inc. in 1991)
Last Position Held: Principal Systems Architect
Responsibilities: Management, RFIC Design, RF Systems, Core Technology Development

1977 –1988: Student Employment
Organizations: Fairchild Semiconductor, Waterloo Engineering Software, University of Waterloo, Wabush Mines, Chalk River Nuclear Labs, McDonald's, Canadian Armed Forces

Past Positions: Member of the FCC Technological Advisory Council (2000-2002)
Member of the Wyoming State Telecommunications Council (2001-2003)

Education: Ph.D., 1988, University of Waterloo (UW), Joint ME/EE Departments
M.A.Sc., 1985, University of Waterloo, Department of Mech. Engineering
B.A.Sc., 1984, University of Waterloo, Department of Mech. Engineering

Awards: 1985 UW Gold Medal, 1989 IEEE Best Paper, 2010 UW Alumni Award,
2016 IEEE Senior Member Recognition for Wireless Technology Innovation

Publications: Over 40 published articles and approximately 55 issued US patents

Start-up Company Engagements (1999 – Present)

Company	Products	Status	Engagement Type
Atheros	WiFi Chips	Sold to Qualcomm	Investor
Resonext	WiFi Chips	Sold to RFMD	Advisor
Athena	WiFi & Mobile TV Chips	Sold to Broadcom	Advisor
WinNet	Outdoor wireless systems	Sold to Alvarion	Investor
Cayman	DSL Modems	Sold to Motorola	Investor
Simple Devices	WLAN appliances	Sold to Motorola	Investor
MobileStar	WiFi Public Access	Sold to T-Mobile	Investor
Cymil	WiMax Chips	Liquidated	Advisor
Clarus	IP Telephony Tools	Liquidated	Investor
Mirra	Network Storage Devices	Sold to Seagate	Investor
WiDeFi	WiFi Chips	Sold to Qualcomm	Executive Chairman
Quorum	Cellular Terminal Chips	Sold to Spreadtrum	Investor, Advisor
Larian	IP Telephony Software	Sold to SS8	Investor, Chairman
TXE	Internet Software	Liquidated	Investor, Board
SiTime	MEMS-based Chips	Sold to MegaChips	Investor, Advisor
Picaboo	Digital Photo Books	Ongoing	Investor
MetroFi	WiFi Public Access	Liquidated	Investor
AirTight	Wireless Security Devices	Ongoing	Advisor
Seabridge	Internet Sports Marketing	Liquidated	Investor
Zing	Portable Music Appliances	Sold to Dell	Investor
Mojix	RFID Readers	Ongoing	Advisor
Tribal Shout	Telephony Internet Access	Liquidated	Investor, Chairman
Quantance	Cellular Terminal Chips	Sold to Skyworks	Investor, Advisor, Board
Lemon	Mobile Payment System	Sold to LifeLock	Investor
GainSpan	WiFi Chips and Modules	Sold to ASIC	Board, Advisor, Investor
Tasting Room	Internet Commerce	Liquidated	Investor
Work Simple	Internet Software	Liquidated	Investor
Cloud	IP Telephony Appliances	Liquidated	Investor
Qik	Mobile Video Sharing	Sold to Skype	Investor
Small Demons	Online books	Liquidated	Investor
All Trails	Mobile application	Ongoing	Investor
Nimble Heart	Wireless ECG monitor	Ongoing	Investor, Advisor
Guerrilla RF	Infrastructure RF Chips	Ongoing	Investor, Advisor
Fastback	4G/5G Network Equipment	Liquidated	Investor, Founder, Board, Employee

Detailed Past Employment Experience:

Jun. 2010 – Dec. 2016: CBF Networks, Inc. (dba Fastback Networks)
Location: San Jose, CA
Position Held: Co-Founder, Chairman and Chief Technology Officer
Responsibilities: System Architecture, Technology Roadmap, IP and Team Development

System Architecture: - Developed a novel architecture for 4G/5G cellular network backhaul in non-line-of-sight (NLOS) conditions based on re-use of LTE standards-based silicon chips
- Developed a novel architecture for 5G cellular network backhaul in line-of-sight (LOS) conditions in millimeter wave bands including 57-115 GHz using array antenna technology and conventional silicon modem chips
- Developed a novel approach to deployment challenges for backhaul of high density 4G and 5G, as well as high performance Wi-Fi (802.11ac or 802.11ax), networks that dramatically reduced deployment time and expense in both NLOS and LOS propagation environments
- Developed a novel approach to interference mitigation across space, time and frequency dimensions for unlicensed spectrum backhaul

Technology Roadmap: - Worked with key RF and baseband silicon vendors to adapt high performance infrastructure 4G/5G cellular network chips and/or 802.11 chips for high performance backhaul applications
- Provided specific input to software defined radio (SDR) chip architectures adopted by silicon vendors such as Qualcomm, Coherent Logix and Lime Microsystems

Intellectual Property: - Responsible for managing the IP portfolio of over 50 US patent filings
- Named inventor on over 50 US patent filings

Team Development: - Led the process of hiring over 40 engineers spanning disciplines such as embedded firmware development, network software development, baseband algorithm design, digital hardware design, RF circuit design, antenna array design, and overall mechanical system design
- Mentored over 50 engineers and managers in development teams distributed amongst San Jose, CA, Vancouver, BC and Newton Abbey, UK

Jun. 2003 – Oct. 2007: WiDeFi, Inc.
Location: Melbourne, FL
Position Held: Executive Chairman
Earlier Position: Management and Technology Advisor (prior to June 2003)
Responsibilities: Corporate Management, RF/baseband ASIC Development

Management: - Led Board of Directors as Independent representative of both common and preferred shareholders
- Responsible for performance evaluation of the CEO and executive staff
- Conducted search for and hired new CEO while retaining early stage CEO as a critical technology contributor (CTO)
- Participated in all financing rounds and the eventual sale of the company

ASIC Development:

- Provided key technology and management interface to outsourced ASIC design partner Atmel in Colorado Springs
- Co-inventor of core technology architecture
- Conducted detailed technical reviews of ASIC development at both circuit and system design levels
- Assisted in debugging technical problems encountered with prototype devices

<u>Oct. 1998 – Nov. 2002: Proxim Corporation</u>

Location: Sunnyvale, CA
Last Position Held: Chief Technology Officer
Earlier Positions: VP Corporate Development, VP Business Development
Responsibilities: ASIC Development, Standards, M&A Deals, OEM Deals, Patent Licensing

ASIC Development:

- Recruited and managed a 20 person ASIC development team including systems architects, modem designers, ASIC designers, verification engineers, and firmware engineers
- Defined product requirements for Phoenix - a 130 nm 4M gate ASIC based on software defined radio for 802.11/16 WLAN/WMAN (project was cancelled in Nov 2002 about 3 months prior to tapeout)
- Phoenix contained full MAC and PHY for 802.11a/b/g, draft 802.11n and 802.16a/d/e with Proxim-proprietary MAC and PHY extensions and additional modes for point to point communication up to 200 Mb/s
- Core of Phoenix was an SDR fabric that extended a MIPS 4KE processor core to implement blocks such as an iterative soft-input/output Viterbi decoder, IFFT/FFT, FEC encoders, interleavers, mappers, etc
- MAC in Phoenix was 95%+ firmware based on a second MIPS core
- Security features in Phoenix included support for 802.11i (AES), TKIP, 802.1x, Radius, WEP, and Proxim proprietary modes
- I/O’s included Ethernet, PCI, and USB 2.0
- Analog I/F’s included dual 12 bit ADCs and DACs at 80 Ms/s
- Also drove development of the PX82475 – a 0.18 um 1M gate ARM-based ASIC for HomeRF 2.0, 1.2 and OpenAir standards that included world’s first MLSE-based equalizer operating at 10 Mb/s for 4-level GMSK in a 3.5 MHz channel bandwidth (project started May 1999, taped out Nov 2000, volume production Jun 2001)

Standards:

- Directed Proxim’s involvement with 802.11 and 802.16 standards groups
- Voting member of the 802.11 standards committee – most active with 802.11g, 802.11h and the WNG-SC process that launched 802.11n
- Directed Proxim’s involvement with the HomeRF Working Group
- Former Chairman of both the Technical Subcommittee and the Board of Directors of HomeRF
- Successfully led a coalition of over 50 companies to convince the FCC to significantly modify the Part 15.247 2.4 GHz band rules in 1999 and 2000 against a powerful and organized opposition with much greater funding
- Accepted nomination to the FCC’s Technological Advisory Committee and served alongside CTOs of major companies such as Motorola, Intel,

Disney, Panasonic, Siemens and many others to advise the FCC on wireless broadband strategies to benefit all US residents
- Nominated by the Governor and elected by Senate confirmation to the Wyoming State Telecommunications Council to advise the Governor on pending state legislation regarding telecom matters

M&A Deals:
- Completed eight separate M&A transactions including Wavespan, Farallon, Micrilor, Siemens US Cordless R&D, Card Access, nBand, Orinoco, and Western Multiplex
- Last deal was an ~$600M sale of Proxim, Inc. (Nasdaq: PROX) to Western Multiplex Corp (Nasdaq: WMUX) in Mar. 2002
- After the sale of Proxim, Inc., WMUX changed its company name to Proxim Corporation, changed its stock symbol to PROX, filed for bankruptcy in 2005, sold assets to Terabeam, Inc. (Nasdaq: TRBM) and the WMUX business unit was renamed Proxim Wireless Corporation
- Responsibilities for M&A deals included all technical diligence including patents, retaining key employees, negotiating purchase terms, and in two cases assuming direct line reporting for the purchased companies
- Led 5 venture investments including Atheros, WinNet, Cayman, Simple Devices and MobileStar

OEM Deals:
- Developed and negotiated 3 largest OEM deals in the company’s history with Intel, Motorola and Siemens
- Each OEM partner made $10M equity investments in Proxim, Inc. (these investments each represented ~3-4% of the market capitalization of Proxim at the time they were made)

Patent Licensing:
- Corporate representative for licensing program including patent litigation
- Provided numerous 30(b)6 depositions for technical and business issues
- Testified at trial as fact witness on technical issues
- Filed 5 US patent applications for WLAN PHY & MAC layer inventions

Feb. 1988 – Oct. 1998: Hewlett-Packard Company (acquired Avantek, Inc. Nov 1991)

Location: Palo Alto, CA
Last Position Held: Principle Systems Architect
Earlier Positions: Director of RFIC Chipset Development, Manager of Silicon RFIC Design, Member of the Technical Staff
Responsibilities: Management, RFIC Development, RF Systems, Core Technology

Management:

- At departure, the RF Components division had over $100M per year in revenue from products developed under my leadership
- Negotiated strategic supply agreements for wafer fabrication
- Managed a team of about 20 engineers reporting via 3 1st level managers for RFIC development in multiple technology specialties
- Ran complex chipset development programs, such as the world's first 802.11 RF chipset, or such as a complete IS-95 transmit and receive chain with resources spread across Europe, North America and Asia
- Organized and led a joint venture program with a former East German microelectronics company

RFIC Development:

- Personally designed over 20 RFIC products
- Designed world's first highly integrated digital cell phone transmit RFIC
- Designed world's first highly integrated receive RFIC for GPS
- Designed world's first 4 Gb/s 4:1 MUX/DEMUX ICs in silicon bipolar
- Designed world's first spec-compliant, fully monolithic silicon VCO for wireless communications standards
- Designed RFICs specifically for GSM, DECT, IS-95, 802.11, HomeRF, CT-2, DBS and other wireless standards
- Designed in silicon bipolar, gallium arsenide MESFET and BiCMOS
- Designed such RF blocks as mixers, synthesizers, low noise amplifiers, power amplifiers, switches, variable gain amplifiers, phase shifters, limiters, discriminators, voltage-controlled oscillators, modulators and demodulators

RF Systems:

- Defined complete chipsets including performance characteristics and system architecture for HomeRF, 802.11, IS-95B and GSM
- Partnered with baseband suppliers such as TI, VLSI, AMD, and others on reference designs for various wireless devices
- Partnered with reference design consultancies including Symbionics, TTPCom, Wavecom, RTX and others

Core Technology:

- Developed a proprietary silicon device simulation model used by HP/Avantek to enhance first pass design success
- Primary standards monitor for HP RF Components on such efforts as 802.11, HomeRF, IS-54, IS-95, GSM, DECT, HiperLAN, and 3GPP
- Lead liaison with HP Labs on wireless research
- Lead liaison with HP Product Divisions for WLAN products and FCC policy
- Co-author of HP Company Strategic Plan for Wireless Technology across all of HP's Measurement, Components, Computing and Printing businesses
- Filed several patent applications for RFIC designs

May 1986 – Nov. 1987: Fairchild Semiconductor
Location: Palo Alto, CA
Position Held: Research Engineer (consultant-basis only Sep. 1986 – Nov. 1987)
Projects: - Design of bipolar circuits for high speed ECL and telecom applications
- Development of packages for high speed circuits (patent granted)
- Optimization of clock chip for Clipper (world's first RISC processor)

Sep. 1986 – Dec. 1987: Waterloo Engineering Software
Location: Waterloo, Canada
Position Held: Sales Engineer
Projects: - Venture-funded startup with 10 employees sold in late 1987
- Sold silicon device simulation software worldwide

May. 1981 – Dec. 1987: University of Waterloo
Location: Waterloo, Canada
Last Position Held: Research Engineer
Earlier Positions: Teaching Assistant, Research Associate, Undergrad Research Assistant
Projects: - Senior researcher for multi-disciplinary research lab on microelectronics device modeling and thermal analysis
- Consulted to or performed research on behalf of companies such as IBM, DEC, Nortel, Thomson CSF, GEC, and Westinghouse on bipolar transistor modeling and cooling of high power bipolar and CMOS transistors and circuits
- Tutored for numerous undergraduate courses

May. 1980 – Apr. 1981: Wabush Mines
Location: Sept-Iles, Canada
Position Held: Engineer
Projects: - Designed numerous facilities and machinery "fixes" in an iron ore mining operation located in Labrador and Northern Quebec

Summer 1979: Chalk River Nuclear Laboratories

Location: Chalk River, Canada
Position Held: Decontamination Technician
Tasks: - Decontaminated radioactive waste, trained for reactor meltdown

Summer/Fall 1978: McDonald's Restaurant

Location: Pembroke, Canada
Position Held: Associate
Tasks: - Flipped burgers, made fries, took orders, cleaned everything

1977 – 1979: Canadian Armed Forces Army Cadet Program

Location: CFB Petawawa, Canada
Position Held: Infantry Sergeant
Tasks: - completed military basic training, received training on infantry small unit tactics to counter Soviet ground invasion forces

Detailed University Education Background:

Ph.D., Feb 1988, University of Waterloo, Waterloo, Canada

Department: Joint Electrical and Mechanical Engineering
Thesis Title: “Thermal and Electrical Modeling of Bipolar Transistors”
Supervisors: Prof. David J. Roulston (EE) and Prof. M. Michael Yovanovich (ME)

Research Topic:

- Developed novel analytical techniques for predicting the performance of bipolar semiconductor devices in multiple applications such as power, RF or high speed data communications
- Key advantage was computational efficiency to enable unprecedented analysis of combined thermal and electrical effects to optimize performance of leading-edge bipolar transistors and circuits
- Foundations for research came from novel application of classical mathematic techniques dating back as far as Euler combined with the application of numerical advances made for fluid mechanics to the drift-diffusion equations governing semiconductor devices

Coursework: Advanced Topics in Semiconductor Device Physics and Circuits
Computational Fluid Mechanics and Convective Heat Transfer
Advanced Topics in Heat Conduction
Graduate Level Applied Mathematics

M.A.Sc., May 1985, University of Waterloo, Waterloo, Canada

Department: Mechanical Engineering
Thesis Title: “Temperature Distributions in Contacting Electrical Conductors”
Supervisor: Prof. M. Michael Yovanovich (ME)

Research Topic:

- Solved the classic coupled problem of determining the temperatures of rough surfaces that conduct electricity with self-heating due electrical constriction resistance by developing novel approximate analytical numerical techniques based on images
- Practical applications for determining contact pressures in any metal to metal electrical contact

Coursework: Semiconductor Device Physics, Fabrication and Circuits
Electromagnetics, RF Propagation and Field Theory
Fluid Mechanics, Conductive, Convective and Radiative Heat Transfer
Advanced Topics in Numerical Analysis
Theory of Models

B.A.Sc., May 1984, University of Waterloo, Department of Mechanical Engineering.

- 5-year undergraduate program that alternates 4-month coursework semesters with 4-month “co-op” work terms in industry with engineering project work requirements.
- Studied all basic ME subjects including heat transfer, fluid mechanics, machine design, stress analysis, automation, manufacturing techniques, and basic electrical circuit design

Academic Achievements:

1989 IEEE “Best Paper” Award for an IEEE Journal publication, this paper was based upon my Ph.D. thesis work.

1988 University of Waterloo, Faculty of Engineering Award for Outstanding Ph.D. work and Faculty sole nominee for University-wide Gold Medal Award.

1985 University of Waterloo, Gold Medal Award for Outstanding Master’s Degree work on a University-wide basis.

1984 University of Waterloo, Dept. of Mechanical Engineering, Graduated 3rd out of 200.

1981 University of Waterloo, Faculty of Engineering, Award for Outstanding Co-Op Work Term Report.

1979 Valedictorian and graduated 1st out of 200 for High School in Pembroke, Ontario.

1979 Descartes High School Math contest winner for Eastern Ontario Region.

Selected Personal Highlights:

2010 Recipient of the University of Waterloo, Faculty of Engineering Alumni Achievement Award for technical innovations in and contributions to the development of wireless Internet and cellular communications technology products over the past 25 years

US Citizen since Feb 2006, US Permanent Resident since 1989, US H1 Visa 1986-1989.

Born in Fredericton, New Brunswick, Canada on Dec. 30, 1961.

Senior Member of the IEEE. Member of the IEEE since 1988. Co-chair of the RFIC Subcommittee for the IEEE BCTM Conference from 1996 to 1998.

Director, Adult Recreation Programs, Philipsburg Ice Association, Granite County, Montana.

Past Owner and operator with wife Eva of a working 200-cow cattle ranch in rural Wyoming.

Past Chairman (2003-2014), Hyattville Community Center Association.

Past Board Member, Hyattville Water Company.

Past Director, Youth Ice Hockey Program, Big Horn County in Wyoming.

Former provincial (“State”) high school champion in the pole vault.

Avid outdoorsman, water skier, hockey player and snow skier.

Selected Publications:

Wiles, E., Negus, K., et al., "Measurement and Analysis of Spectrum Occupancy from 140 to 1000 MHz in Rural Western Montana", European Conference on Antennas and Propagation, Davos, Switzerland, Apr. 10-15, 2016.

Lea, A., Negus, K., et al., "Spectrum Options for Wireless Backhaul of Small Cells", European Conference on Antennas and Propagation, The Hague, Netherlands, Apr. 6-11, 2014.

Negus, K.J., "Spectrum Options for Wireless Backhaul of Small Cells", Small Cell Forum, Dallas, TX, December 4, 2013.

Negus, K.J. and Petrick, A., "History of Wireless Local Area Networks (WLANs) in the Unlicensed Bands", George Mason University Law School Conference, Information Economy Project, Arlington, VA., April 4, 2008.

Primary co-author of the HomeRF 2.01 Technical Specification (526 pages), July 2002, published by the HomeRF Working Group.

Negus, K.J. and Swan, B., "HomeRF: Design-in Module Practices", Intel Developer Forum, San Jose, CA, Feb. 2001.

Negus, K.J., "Designing with HomeRF Technology", Intel Developer Forum, San Jose, CA, Aug. 2000.

Negus, K.J., Stephens, A., and Lansford, J., "HomeRF: Wireless Networking for the Connected Home", IEEE Journal of Personal Communications, Vol. 7, No. 1, Feb. 2000, pp. 20-27.

Negus, K.J., Waters, J., et. al., "HomeRF and SWAP: Wireless Networking for the Connected Home", ACM Mobile Computing and Comms Review, Vol. 2, No. 4, Oct. 1998, pp. 28-37.

Morkner, H., Frank, M. and Negus, K., "A Novel Integrated Microwave Bias Network for Low-Cost Multistage Amplifiers", IEEE MTT-S Symposium Digest, Vol. 1, Jun. 1997, pp. 9-12.

Jansen, B., Negus, K., and Lee, D., "Silicon Bipolar VCO Family for 1.1 to 2.2 GHz with Fully-Integrated Tank and Tuning Circuits", 44th IEEE ISSCC Digest of Technical Papers, Feb. 1997, pp. 392-393.

Hutchinson, C., Frank, M., and Negus, K., "Silicon Bipolar 12 GHz Downconverter for Satellite Receivers", Proc. 1995 IEEE Bipolar Circuits and Technology Meeting, Oct. 1995, pp. 198-201.

Negus, K., et. al., "Highly-Integrated Transmitter RFIC with Monolithic Narrowband Tuning for Digital Cellular Handsets", 41st IEEE ISSCC Digest of Technical Papers, Feb. 1994, pp. 38-39.

Negus, K. and Millicker, D., "RFICs for Reduced Size, Cost and Power Consumption in Handheld Wireless Transceivers", Proceedings of the IEEE 2nd International Conference on Universal Personal Communications, Oct. 1993, pp. 919-925.

Negus, K.J., et. al., "3.3V GPS Receiver MMIC Implemented on a Mixed-Signal, Silicon Bipolar Array", IEEE MTT-S Symposium Digest, Vol. 2, Jun. 1992, pp. 1071-1074.

Negus, K., et. al., “Silicon Bipolar Mixed-Signal Parameterized-Cell Array for Wireless Applications to 4 GHz”, 39th IEEE ISSCC Digest of Technical Papers, Feb. 1992, pp. 230-231.

Negus, K. J., “Multi-Gbits/s Silicon Bipolar Multiplexer and Demultiplexer with Interleaved Architectures”, Proc. 1991 IEEE Bipolar Circuits and Technology Meeting, Oct. 1991, pp. 35-38.

Negus, K.J. and Wholey, J.N., “Multifunction Silicon MMICs for Frequency Conversion Applications”, IEEE Transactions on Microwave Theory and Techniques, Vol. 38, No. 9, Sep. 1990, pp. 1191-1198.

Negus, K.J. and Wholey, J.N., “Implementation of RF/Microwave Receiver Components on a Semi-Custom Silicon Bipolar Array”, IEEE MTT-S Symposium Digest, Jun. 1990, pp. 67-72.

Negus, K.J., Franklin, R.W. and Yovanovich, M.M., “Thermal Modeling and Experimental Techniques for Microwave Bipolar Devices”, IEEE Transactions on Components, Hybrids and Manufacturing Technology, Vol. 12, No. 4, Dec. 1989, pp. 680-689.
- this paper won the IEEE award for Best Journal Paper in 1989

Negus, K.J. and Roulston, D.J., “Simplified Modeling of Delays in the Emitter-Base Junction”, Solid State Electronics, Vol. 31, No. 9, Sep. 1988, pp. 1464-1466.

Negus, K.J. and Yovanovich, M.M., “Correlation of the Gap Conductance Integral for Conforming Rough Surfaces”, Journal of Thermophysics and Heat Transfer, Vol. 2, No. 3, July 1988, pp. 279-281.

Negus, K.J., Yovanovich, M.M. and Thompson, J.C., “Constriction Resistance of Circular Contacts on Coated Surfaces: Effect of Boundary Conditions”, Journal of Thermophysics and Heat Transfer, Vol. 2, No. 2, Apr. 1988, pp. 158-164.

Negus, K.J., Yovanovich, M.M. and Roulston, D.J., “An Introduction to Thermal Electrical Coupling in Bipolar Transistors”, Proc of ASME Thermal Engineering Conference, Vol. 3, July 1987, pp. 395-401.

Negus, K.J. and Yovanovich, M.M., “Simple Separability for Steady Heat Conduction with Spatially-Varying Thermal Conductivity”, Int. Journal of Heat and Mass Transfer, Vol. 30, No. 7, July 1987, pp. 1552-1555.

Negus, K.J. and Yovanovich, M.M., “Thermal Analysis and Optimization of Convectively-Cooled Microelectronic Circuit Boards”, Proc of ASME Thermophysics and Heat Transfer Conference, Vol. 57, June 1986, pp. 167-176.

Negus, K.J., Yovanovich, M.M. and DeVaal, J.W., “Development of Thermal Constriction Resistance for Anisotropic Rough Surfaces by the Method of Images”, 23rd ASME National Heat Transfer Conference, Denver, CO, July 1985.

Thompson, J.C. and Negus, K.J., “Developments in a Least Squares Asymptotic Analysis of Isochromatic Data from Stress Concentration Regions in Plane Problems”, Strain, Vol. 20, No. 3, 1984, pp.133-134.

Selected Patents:

Negus, K.J. and Proctor, J.A., Assigned to Fastback Networks, US 8,422,540, “Intelligent Backhaul Radio with Zero Division Duplexing”, filed Sep. 10, 2012.

Lea, D.A., Negus, K.J., *et al*, Assigned to Fastback Networks, US 8,467,363, “Intelligent Backhaul Radio and Antenna System”, filed Jun. 28, 2012.

Negus, K.J. and Proctor, J.A., Assigned to Fastback Networks, US 8,385,305, “Hybrid Band Intelligent Backhaul Radio”, filed Apr. 16, 2012.

Negus, K.J. and Proctor, J.A., Assigned to Fastback Networks, US 8,502,733, “Transmit Co-Channel Spectrum Sharing”, filed Feb. 10, 2012.

Negus, K.J. and Duffy, K.J., Assigned to Fastback Networks, US 8,300,590, “Intelligent Backhaul System”, filed Oct. 11, 2011.

Negus, K.J., Assigned to Fastback Networks, US 8,238,318, “Intelligent Backhaul Radio”, filed Aug. 17, 2011.

Gainey, K.M., Negus, K.J., *et al*, Assigned to WiDeFi, Inc., US 7,187,904, “Frequency translating repeater with low cost high performance local oscillator architecture”, filed Jun. 3, 2005.

Negus, K., Assigned to Proxim, Inc., US 7,035,283, “Asymmetric data traffic throughput in CSMA/CA networks”, filed Apr. 6, 2001.

Negus, K., Assigned to Proxim, Inc., US 7,085,284, “Prioritization scheme for CSMA/CA”, filed Nov. 3, 2000.

Romans, C., Gaoit, L., Negus, K.J., et. al., Assigned to Hewlett Packard, US 6,587453, “Method of communicating first and second data types”, filed Dec. 17, 1998.

Nguyen, N.M. and Negus, K.J., Assigned to Hewlett Packard, US 5,532,655, “Method and apparatus for AC/DC signal multiplexing”, filed Feb. 24, 1995.

Wholey, J. and Negus, K., Assigned to Hewlett Packard, US 5,436,595, “Low voltage bipolar amplifier”, filed Aug. 1, 1994.

Negus, K.J., Assigned to Hewlett Packard, US 5,150,364, “Interleaved time-division demultiplexor”, filed Aug. 24, 1990.

Negus, K.J., Assigned to Avantek, US 5,111,455, “Interleaved time-division multiplexor with phase-compensated frequency doublers”, filed Aug. 24, 1990.

Phy, W.S., Early, J.M. and Negus, K.J., Assigned to Fairchild Semiconductor, US 4,839,717, “Ceramic package for high frequency semiconductor devices”, filed Dec. 19, 1986.

US005500920A

United States Patent [19]

Kupiec

[11] **Patent Number: 5,500,920**

[45] **Date of Patent: Mar. 19, 1996**

[54] **SEMANTIC CO-OCCURRENCE FILTERING FOR SPEECH RECOGNITION AND SIGNAL TRANSCRIPTION APPLICATIONS**

[75] Inventor: **Julian M. Kupiec**, Cupertino, Calif.

[73] Assignee: **Xerox Corporation**, Stamford, Conn.

[21] Appl. No.: **316,619**

[22] Filed: **Sep. 30, 1994**

Related U.S. Application Data

[63] Continuation of Ser. No. 126,170, Sep. 23, 1993, abandoned.

[51] **Int. Cl.**^{6} **G10L 9/00**

[52] **U.S. Cl.** **395/2.79**; 395/2.84; 395/2.86; 364/419.07

[58] **Field of Search** 395/2.44, 2.69, 395/2.79, 2.84, 2.86; 381/43, 44, 52; 364/419.03, 419.08, 419.07, 419.13

[56] **References Cited**

U.S. PATENT DOCUMENTS

2,921,133	1/1960	Kalfaian	178/31
3,158,685	11/1964	Gerstman et al.	395/2.69
3,996,569	12/1976	Saunders	365/189.07
4,270,182	5/1981	Asija	364/419.2
4,674,066	6/1987	Kucera	395/600
4,823,306	4/1989	Barbic et al.	395/600
4,931,935	6/1990	Ohira et al.	364/419.08
4,994,967	2/1991	Asakawa	364/419.08
5,062,074	10/1991	Kleinberger	395/600
5,063,508	11/1991	Yamada et al.	364/419.03
5,278,918	1/1994	Bernzott et al.	364/419.08
5,278,980	1/1994	Pedersen et al.	395/600
5,390,281	2/1995	Luciw et al.	364/419.08
5,406,480	4/1995	Kanno	364/419.08

FOREIGN PATENT DOCUMENTS

0157539	10/1985	European Pat. Off.	G06F 15/40
0304191	2/1989	European Pat. Off.	G06F 15/40
0425291A2	2/1991	European Pat. Off. .	

OTHER PUBLICATIONS

Webster's II New Riverside University Dictionary, 1988, p. 1226.

"Text Search and Retrieval Reference Manual for the Automated Patent System," U.S. Patent & Trademark Office, Oct. 21, 1992.

Communication: European Search Report for D/93184 dated 11 Jan. 1995.

Cutting et al., "An Object-Oriented Architecture for Text Retrieval," in Intelligent Text and Image Handling, Proceedings of a Conference on Intelligent Text and Image Handling 'RIAO 91, ' Barcelona, Spain Apr. 2–5, 1991 (A. Lichnerowicz, ed.), pp. 285–298.

Glavitsch et al., "A System for Retrieving Speech Documents," 15th Ann. Int'l. SIGIR '92/Denmark–Jun. 1992, pp. 168–176.

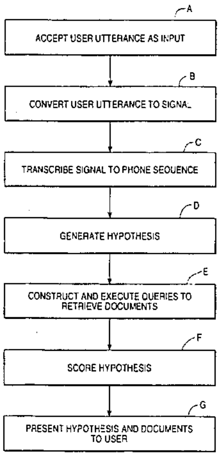
(List continued on next page.)

Primary Examiner—David D. Knepper
Assistant Examiner—Michael A. Sartori
Attorney, Agent, or Firm—Alexander E. Silverman

[57] **ABSTRACT**

A system and method for automatically transcribing an input question from a form convenient for user input into a form suitable for use by a computer. The question is a sequence of words represented in a form convenient for the user, such as a spoken utterance or a handwritten phrase. The question is transduced into a signal that is converted into a sequence of symbols. A set of hypotheses is generated from the sequence of symbols. The hypotheses are sequences of words represented in a form suitable for use by the computer, such as text. One or more information retrieval queries are constructed and executed to retrieve documents from a corpus (database). Retrieved documents are analyzed to produce an evaluation of the hypotheses of the set and to select one or more preferred hypotheses from the set. The preferred hypotheses are output to a display, speech synthesizer, or applications program. Additionally, retrieved documents relevant to the preferred hypotheses can be selected and output.

22 Claims, 11 Drawing Sheets


A
ACCEPT USER UTTERANCE AS INPUT
B
CONVERT USER UTTERANCE TO SIGNAL
C
TRANSCRIBE SIGNAL TO PHONE SEQUENCE
D
GENERATE HYPOTHESIS
E
CONSTRUCT AND EXECUTE QUERIES TO RETRIEVE DOCUMENTS
F
SCORE HYPOTHESIS
G
PRESENT HYPOTHESIS AND DOCUMENTS TO USER

OTHER PUBLICATIONS

Harris, Mary Dee Harris "Introduction to Natural Language Processing," pp. 102–114 (Reston Publishing Company, Inc., Reston, VA), 1985.

Hopcroft et al., "Introduction to Automata Theory, Languages, and Computation," pp. 13–76 (Copyright © 1979 by Addison–Wesley Publishing Company, Reading, MA).

Lamel et al., "Speech Database Development: Design and Analysis of the Acoustic–Phonetic Corpus," pp. 100–109 (Proceedings of the Speech Recognition Workshop held in Palo Alto, CA, Feb. 19–20, 1986).

Lucchesi et al., "Applications of Finite Automata Representing Large Vocabularies," *Software–Practice and Experience,* 23(1):15–30, Jan. 1993 (see especially pp. 26–27).

Niimi et al., "An Information Retrieval System With a Speech Interface," 1992 Int'l. Conf. on Spoken Language Processing (eds. John J. Ohala, Terrance Nearey, Bruce Denning, Megan Hodge, Grace Wiebe), held in Banff, Alberta, Canada, Oct. 12–16, 1992 (pp. 1407–1410).

Rabiner, Lawrence R., "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE,* 77(2):257–285, Feb. 1989.

Salton et al., "Extended Boolean Information Retrieval," Communications of the ACM, 26(12):1022–1036 (Dec. 1983).

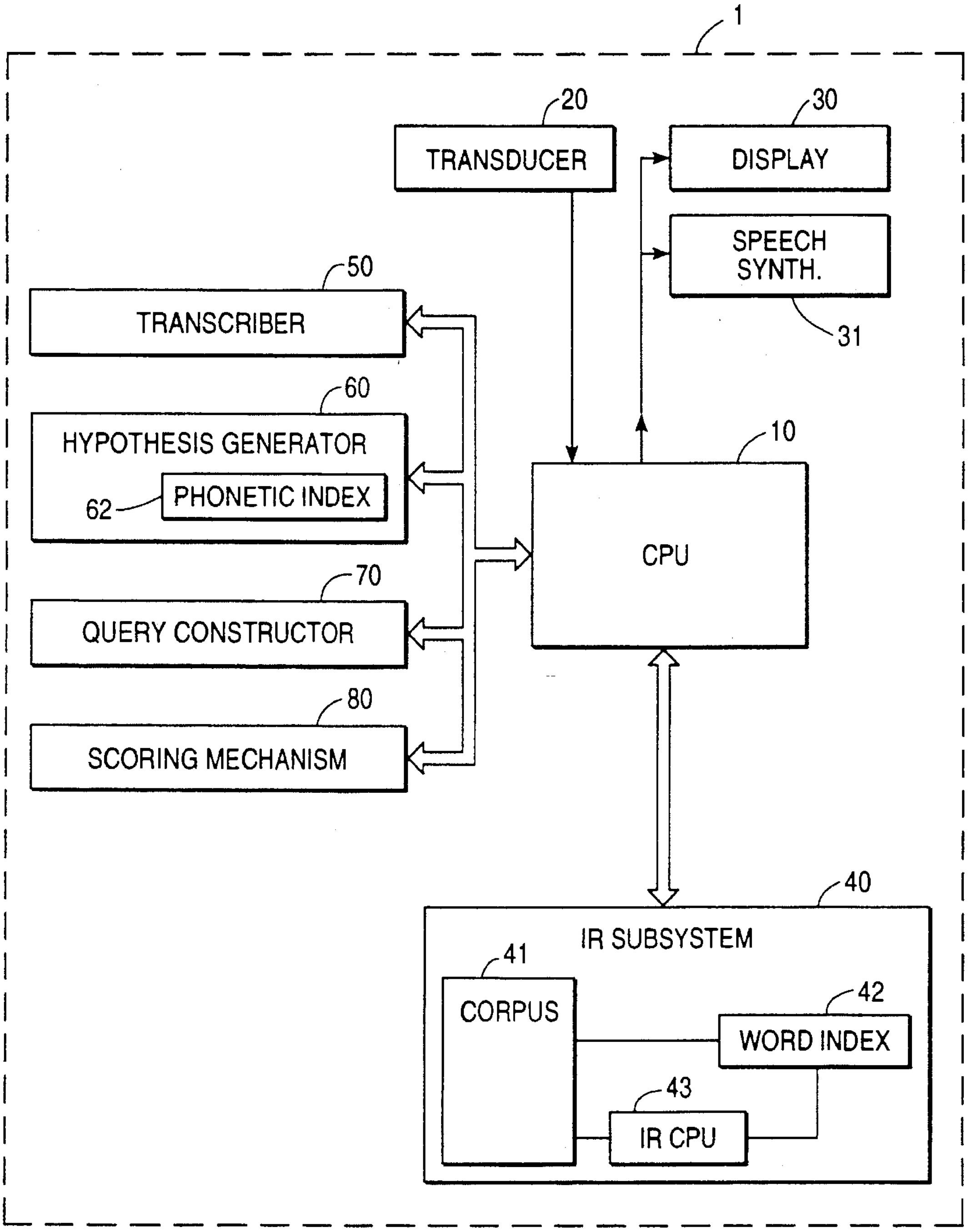

1
20
TRANSDUCER
30
DISPLAY
SPEECH SYNTH.
31
50
TRANSCRIBER
60
HYPOTHESIS GENERATOR
62
PHONETIC INDEX
70
QUERY CONSTRUCTOR
80
SCORING MECHANISM
10
CPU
40
IR SUBSYSTEM
41
CORPUS
42
WORD INDEX
43
IR CPU

FIG. 1

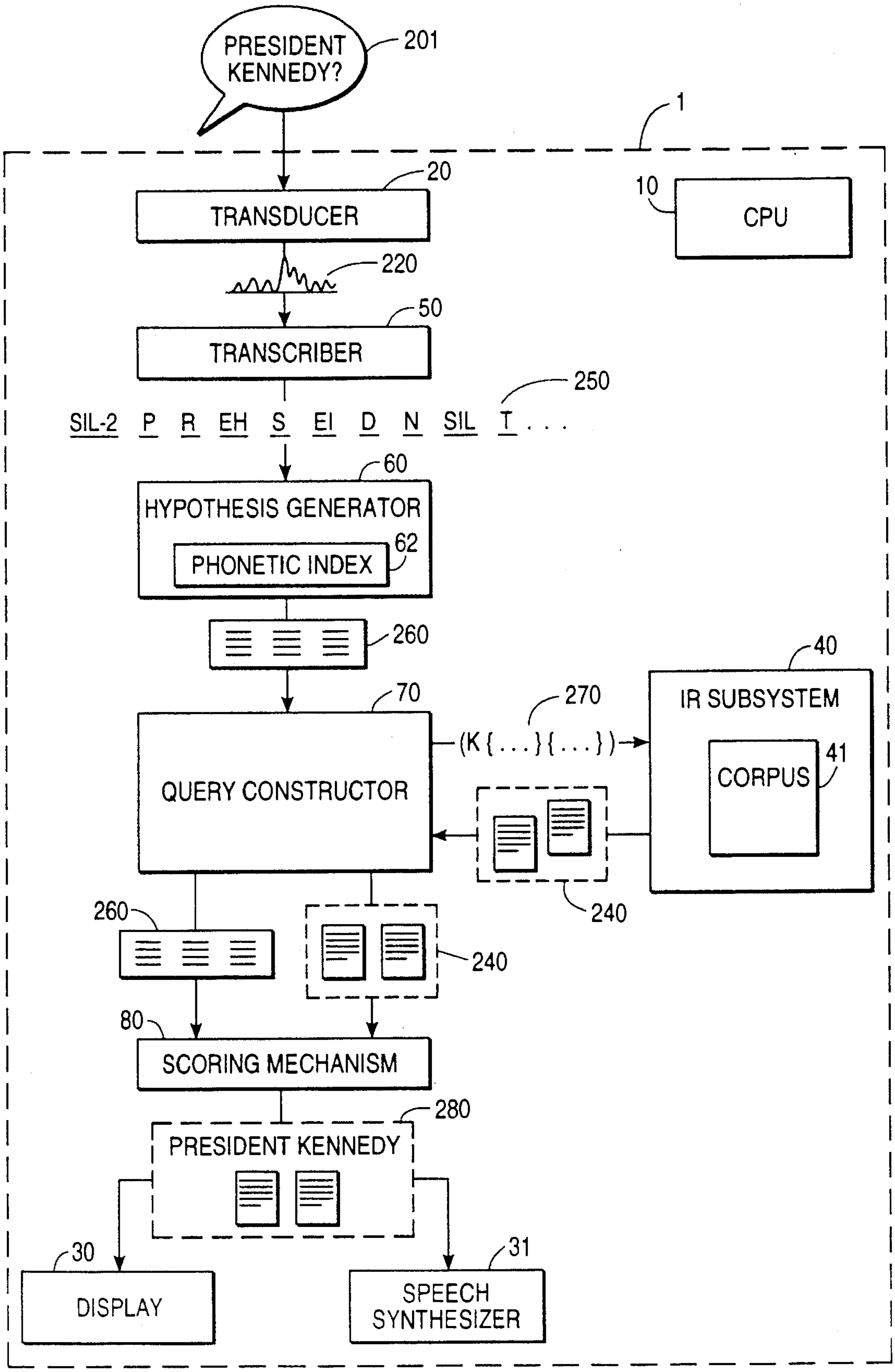

PRESIDENT KENNEDY?
201
1
20
TRANSDUCER
10
CPU
220
50
TRANSCRIBER
250
SIL-2 P R EH S EI D N SIL T . . .
60
HYPOTHESIS GENERATOR
PHONETIC INDEX
62
260
40
70
270
IR SUBSYSTEM
(K{. . .}{. . .})
QUERY CONSTRUCTOR
CORPUS
41
240
260
240
80
SCORING MECHANISM
280
PRESIDENT KENNEDY
30
31
DISPLAY
SPEECH SYNTHESIZER

FIG. 2

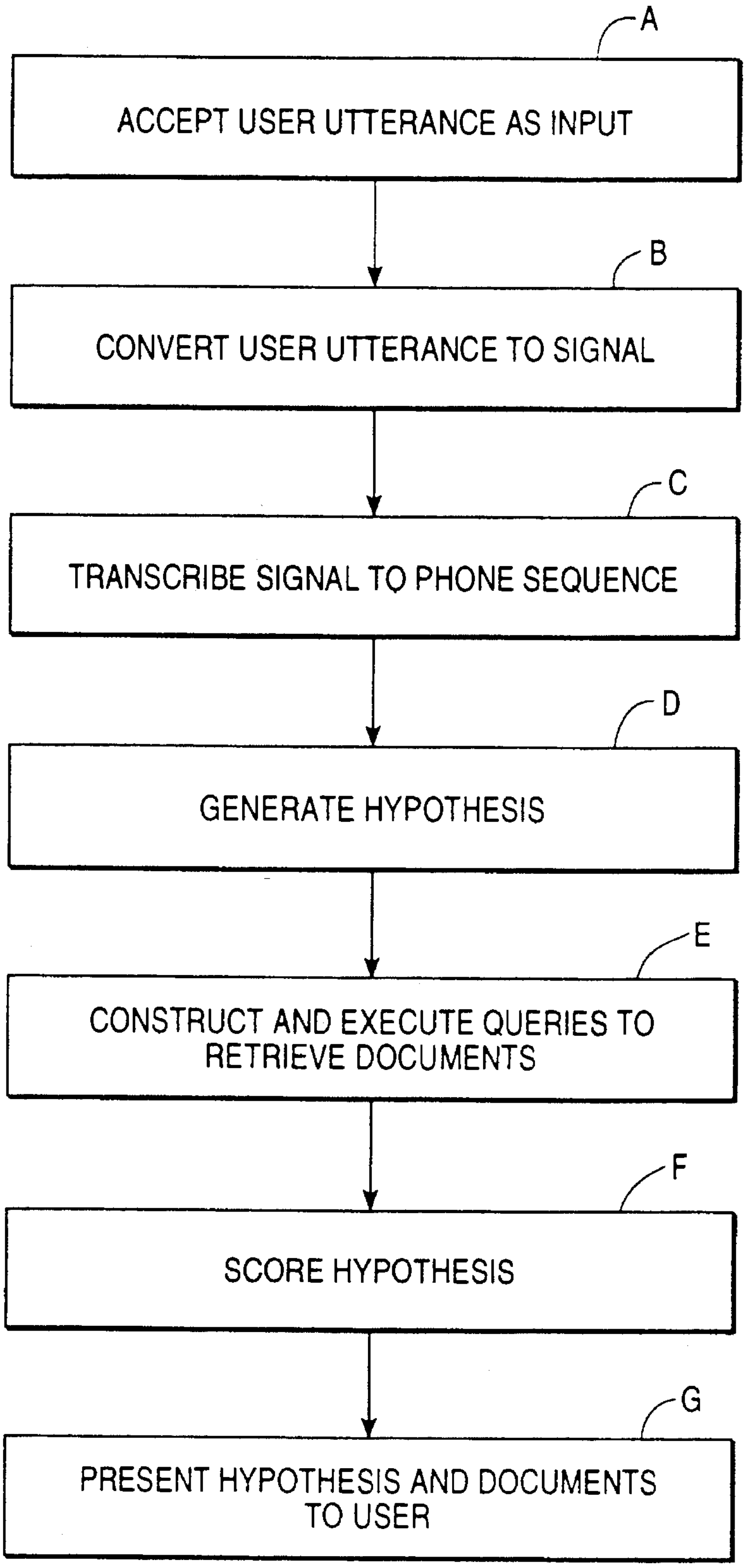
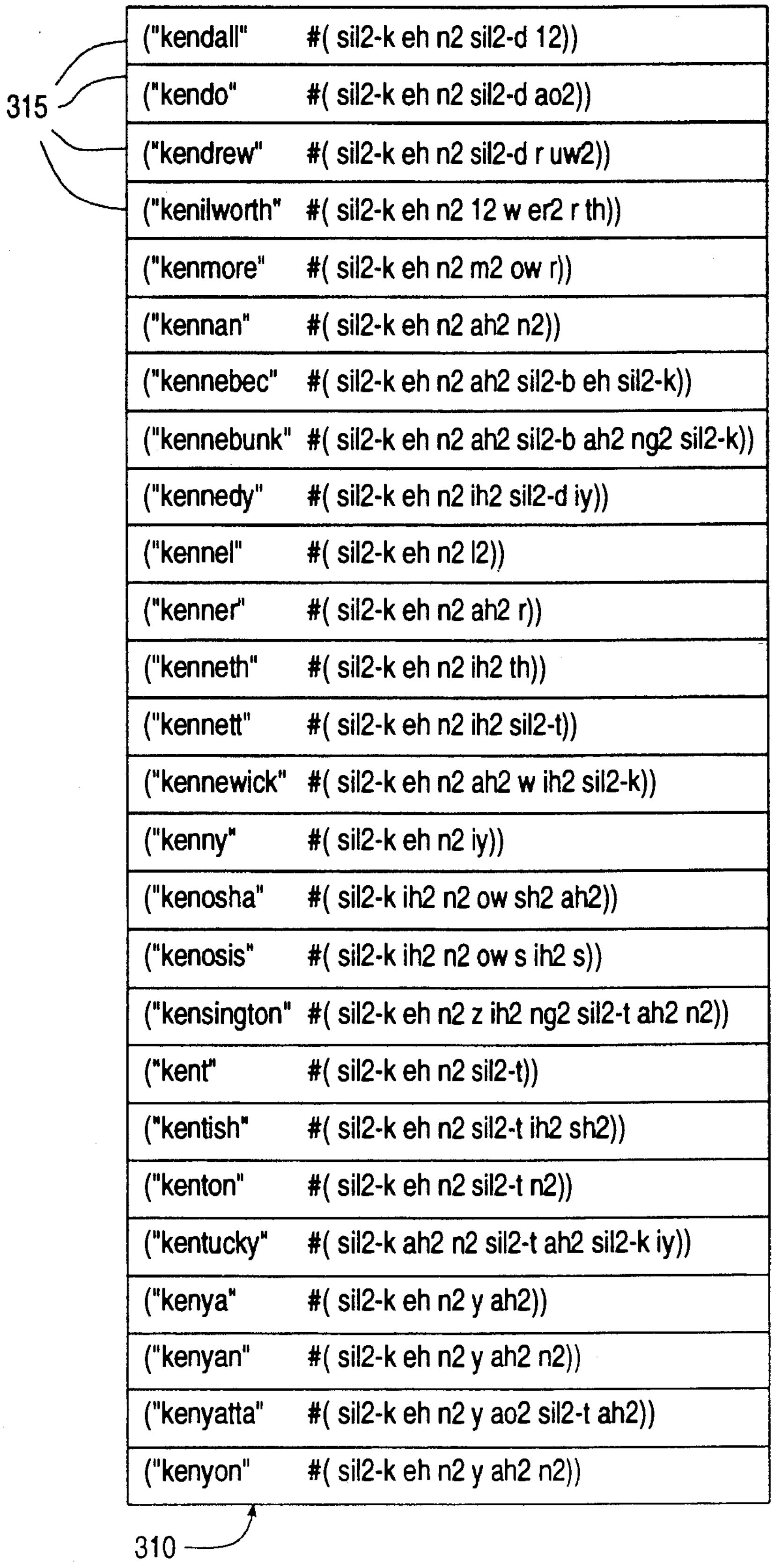

A
ACCEPT USER UTTERANCE AS INPUT
B
CONVERT USER UTTERANCE TO SIGNAL
C
TRANSCRIBE SIGNAL TO PHONE SEQUENCE
D
GENERATE HYPOTHESIS
E
CONSTRUCT AND EXECUTE QUERIES TO RETRIEVE DOCUMENTS
F
SCORE HYPOTHESIS
G
PRESENT HYPOTHESIS AND DOCUMENTS TO USER

FIG. 3



("kendall"	#(sil2-k eh n2 sil2-d 12))
("kendo"	#(sil2-k eh n2 sil2-d ao2))
("kendrew"	#(sil2-k eh n2 sil2-d r uw2))
("kenilworth"	#(sil2-k eh n2 12 w er2 r th))
("kenmore"	#(sil2-k eh n2 m2 ow r))
("kennan"	#(sil2-k eh n2 ah2 n2))
("kennebec"	#(sil2-k eh n2 ah2 sil2-b eh sil2-k))
("kennebunk"	#(sil2-k eh n2 ah2 sil2-b ah2 ng2 sil2-k))
("kennedy"	#(sil2-k eh n2 ih2 sil2-d iy))
("kennel"	#(sil2-k eh n2 l2))
("kenner"	#(sil2-k eh n2 ah2 r))
("kenneth"	#(sil2-k eh n2 ih2 th))
("kennett"	#(sil2-k eh n2 ih2 sil2-t))
("kennewick"	#(sil2-k eh n2 ah2 w ih2 sil2-k))
("kenny"	#(sil2-k eh n2 iy))
("kenosha"	#(sil2-k ih2 n2 ow sh2 ah2))
("kenosis"	#(sil2-k ih2 n2 ow s ih2 s))
("kensington"	#(sil2-k eh n2 z ih2 ng2 sil2-t ah2 n2))
("kent"	#(sil2-k eh n2 sil2-t))
("kentish"	#(sil2-k eh n2 sil2-t ih2 sh2))
("kenton"	#(sil2-k eh n2 sil2-t n2))
("kentucky"	#(sil2-k ah2 n2 sil2-t ah2 sil2-k iy))
("kenya"	#(sil2-k eh n2 y ah2))
("kenyan"	#(sil2-k eh n2 y ah2 n2))
("kenyatta"	#(sil2-k eh n2 y ao2 sil2-t ah2))
("kenyon"	#(sil2-k eh n2 y ah2 n2))

FIG. 4

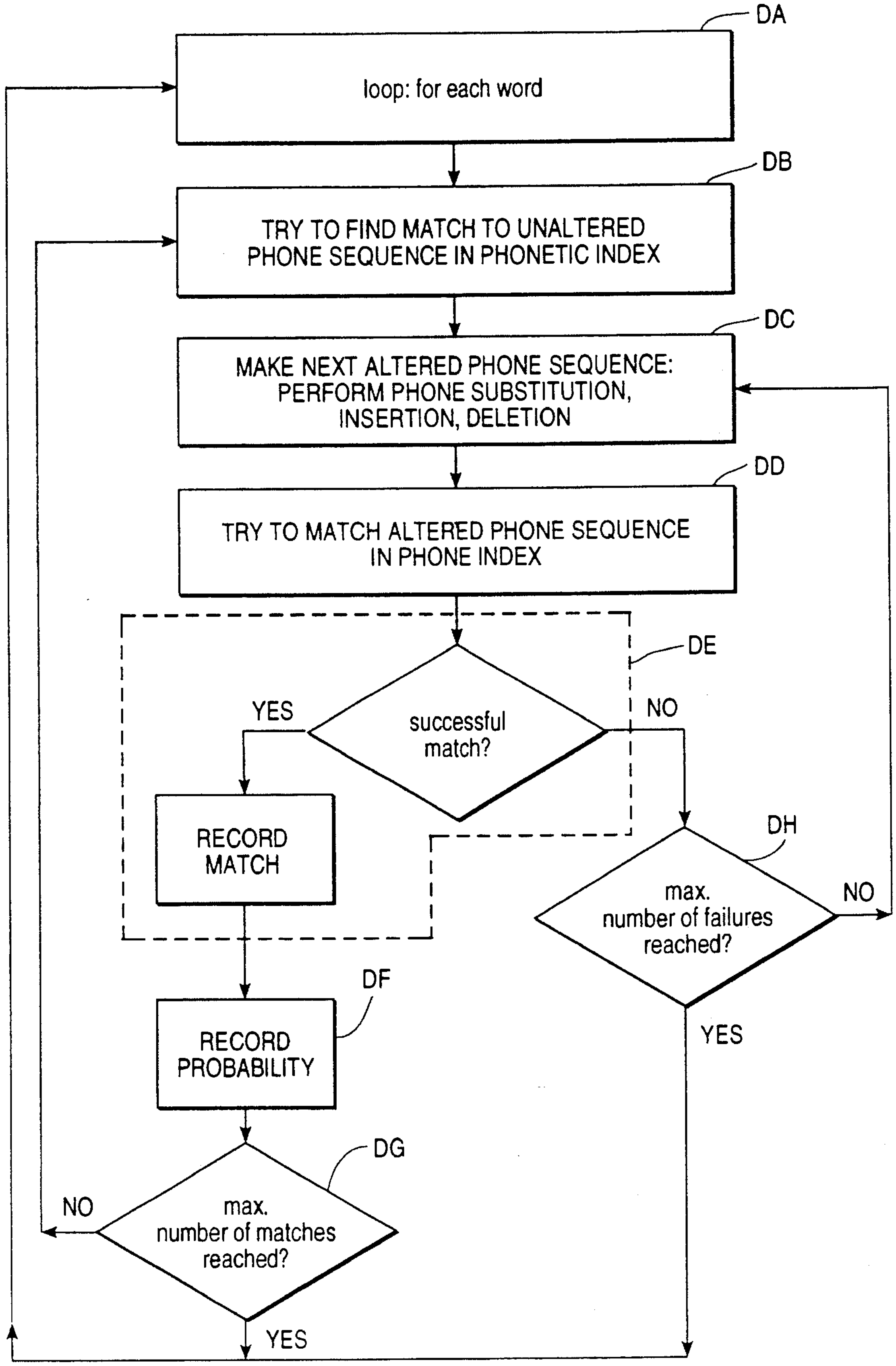

DA
loop: for each word
DB
TRY TO FIND MATCH TO UNALTERED PHONE SEQUENCE IN PHONETIC INDEX
DC
MAKE NEXT ALTERED PHONE SEQUENCE: PERFORM PHONE SUBSTITUTION, INSERTION, DELETION
DD
TRY TO MATCH ALTERED PHONE SEQUENCE IN PHONE INDEX
DE
YES
successful match?
NO
RECORD MATCH
DH
max. number of failures reached?
NO
DF
RECORD PROBABILITY
YES
DG
NO
max. number of matches reached?
YES

FIG. 5

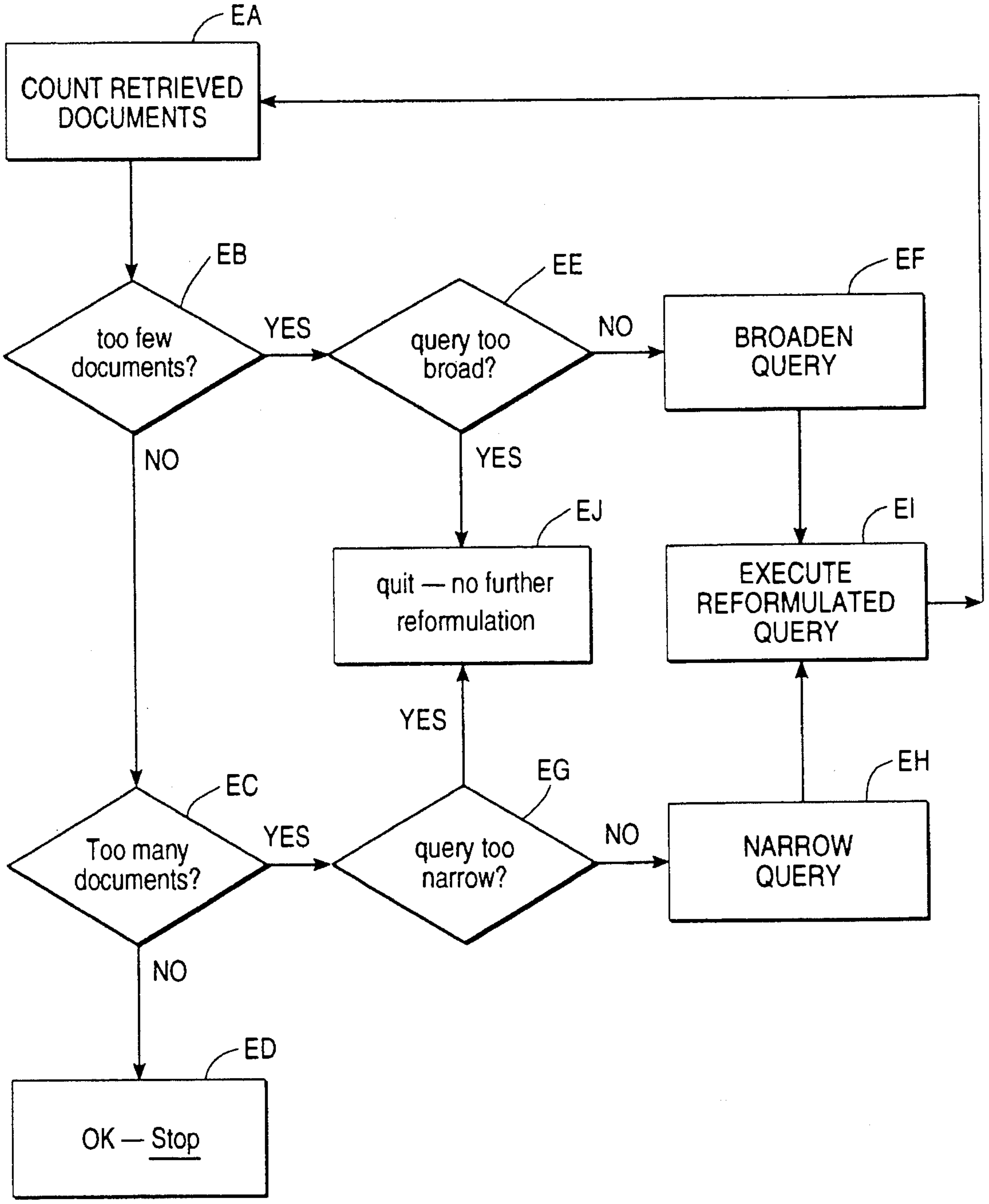

EA
COUNT RETRIEVED DOCUMENTS
EB
too few documents?
YES
NO
EE
query too broad?
NO
YES
EF
BROADEN QUERY
EJ
quit — no further reformulation
EI
EXECUTE REFORMULATED QUERY
EC
Too many documents?
YES
NO
EG
query too narrow?
YES
NO
EH
NARROW QUERY
ED
OK — Stop

FIG. 6

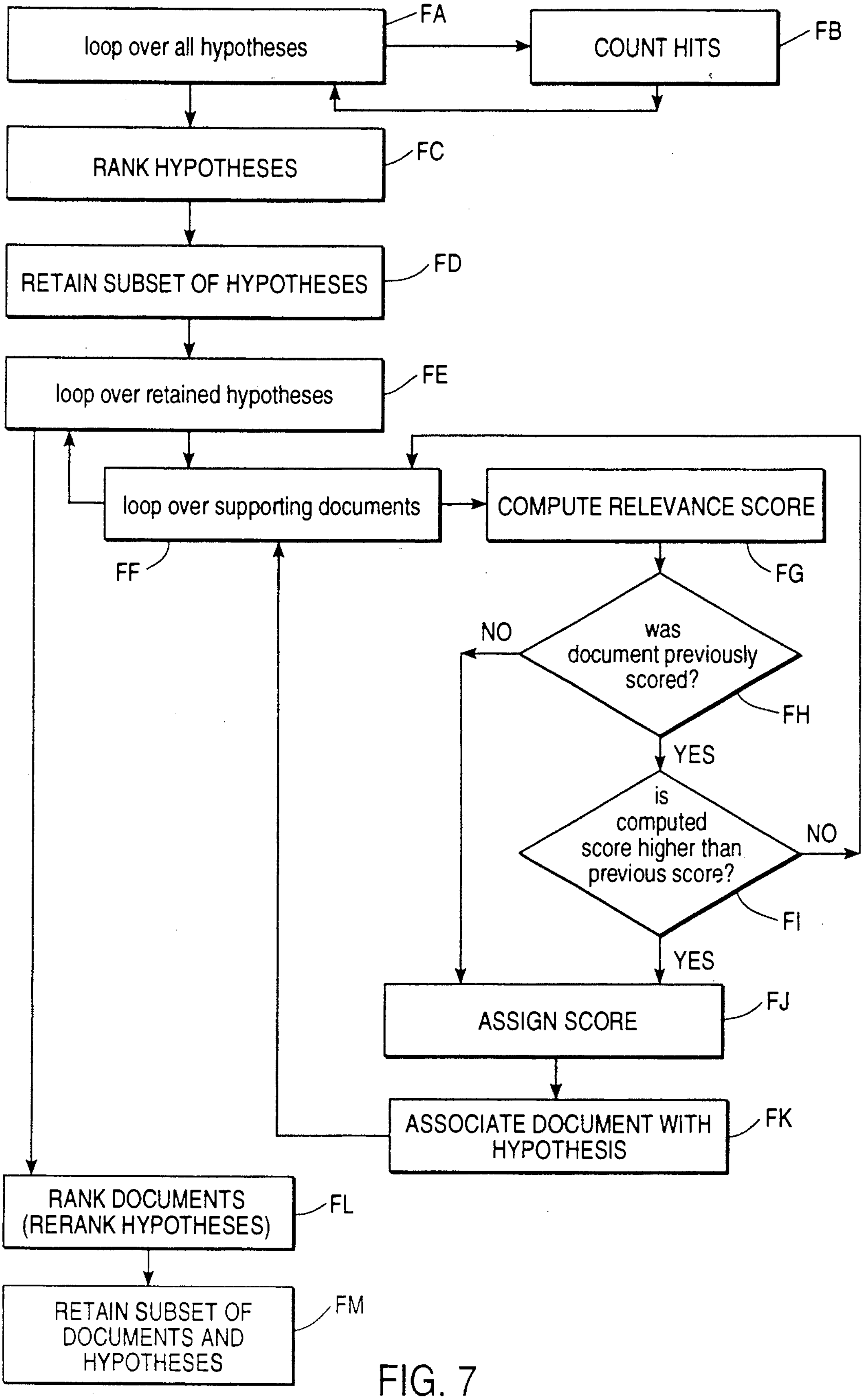

loop over all hypotheses
FA
COUNT HITS
FB
RANK HYPOTHESES
FC
RETAIN SUBSET OF HYPOTHESES
FD
loop over retained hypotheses
FE
loop over supporting documents
FF
COMPUTE RELEVANCE SCORE
FG
NO
was document previously scored?
FH
YES
is computed score higher than previous score?
NO
FI
YES
ASSIGN SCORE
FJ
ASSOCIATE DOCUMENT WITH HYPOTHESIS
FK
RANK DOCUMENTS (RERANK HYPOTHESES)
FL
RETAIN SUBSET OF DOCUMENTS AND HYPOTHESES
FM

FIG. 7

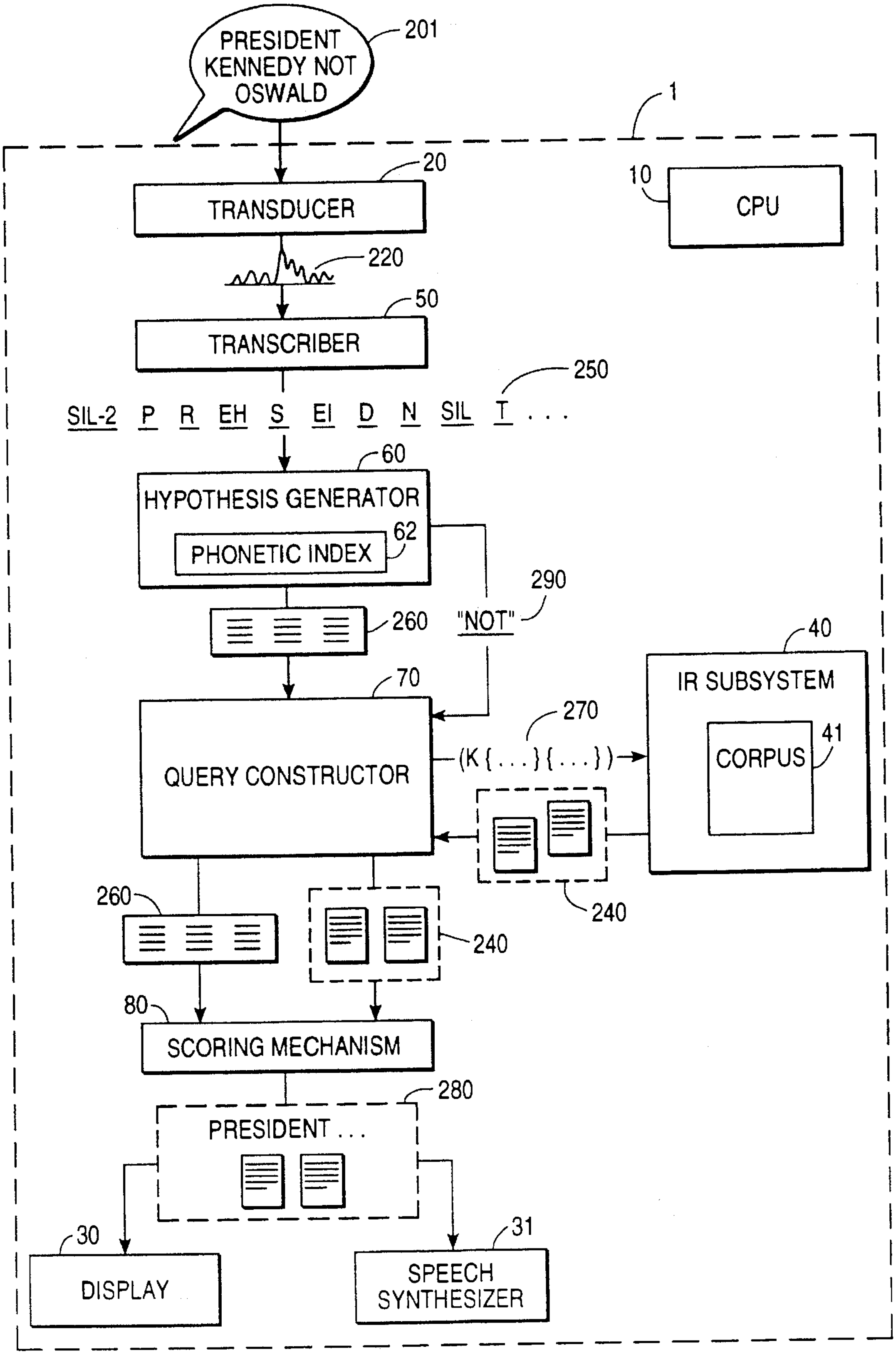

PRESIDENT KENNEDY NOT OSWALD
201
1
20
TRANSDUCER
10
CPU
220
50
TRANSCRIBER
250
SIL-2 P R EH S EI D N SIL T . . .
60
HYPOTHESIS GENERATOR
PHONETIC INDEX
62
290
260
"NOT"
40
70
IR SUBSYSTEM
270
QUERY CONSTRUCTOR
(K{...}{...})
CORPUS
41
240
260
240
80
SCORING MECHANISM
280
PRESIDENT . . .
30
31
DISPLAY
SPEECH SYNTHESIZER

FIG. 8

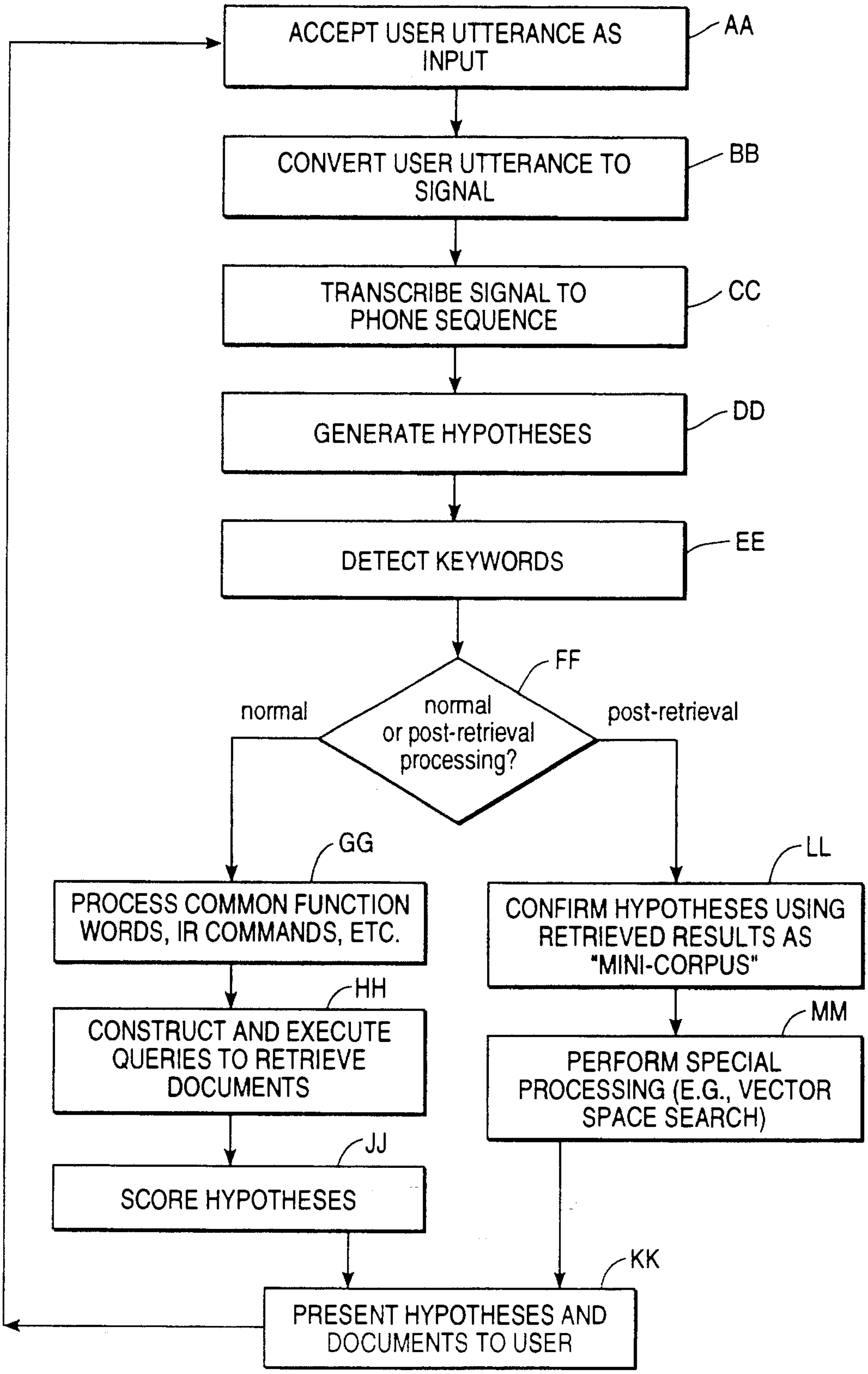

ACCEPT USER UTTERANCE AS INPUT
AA
CONVERT USER UTTERANCE TO SIGNAL
BB
TRANSCRIBE SIGNAL TO PHONE SEQUENCE
CC
GENERATE HYPOTHESES
DD
DETECT KEYWORDS
EE
FF
normal
normal or post-retrieval processing?
post-retrieval
GG
PROCESS COMMON FUNCTION WORDS, IR COMMANDS, ETC.
HH
CONSTRUCT AND EXECUTE QUERIES TO RETRIEVE DOCUMENTS
JJ
SCORE HYPOTHESES
LL
CONFIRM HYPOTHESES USING RETRIEVED RESULTS AS "MINI-CORPUS"
MM
PERFORM SPECIAL PROCESSING (E.G., VECTOR SPACE SEARCH)
KK
PRESENT HYPOTHESES AND DOCUMENTS TO USER

FIG. 9

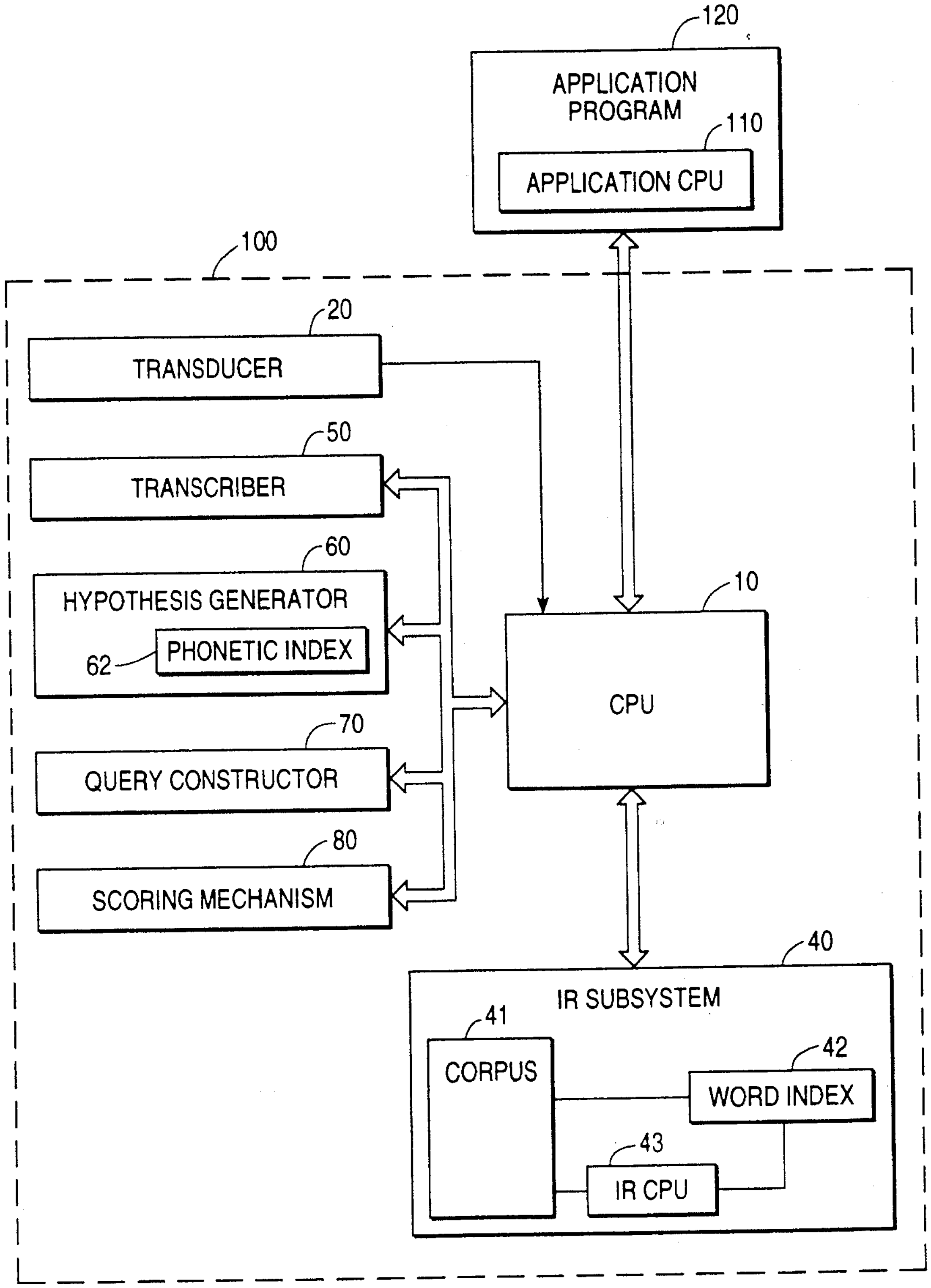

120
APPLICATION PROGRAM
110
APPLICATION CPU
100
20
TRANSDUCER
50
TRANSCRIBER
60
HYPOTHESIS GENERATOR
62
PHONETIC INDEX
70
QUERY CONSTRUCTOR
80
SCORING MECHANISM
10
CPU
40
IR SUBSYSTEM
41
CORPUS
42
WORD INDEX
43
IR CPU

FIG. 10

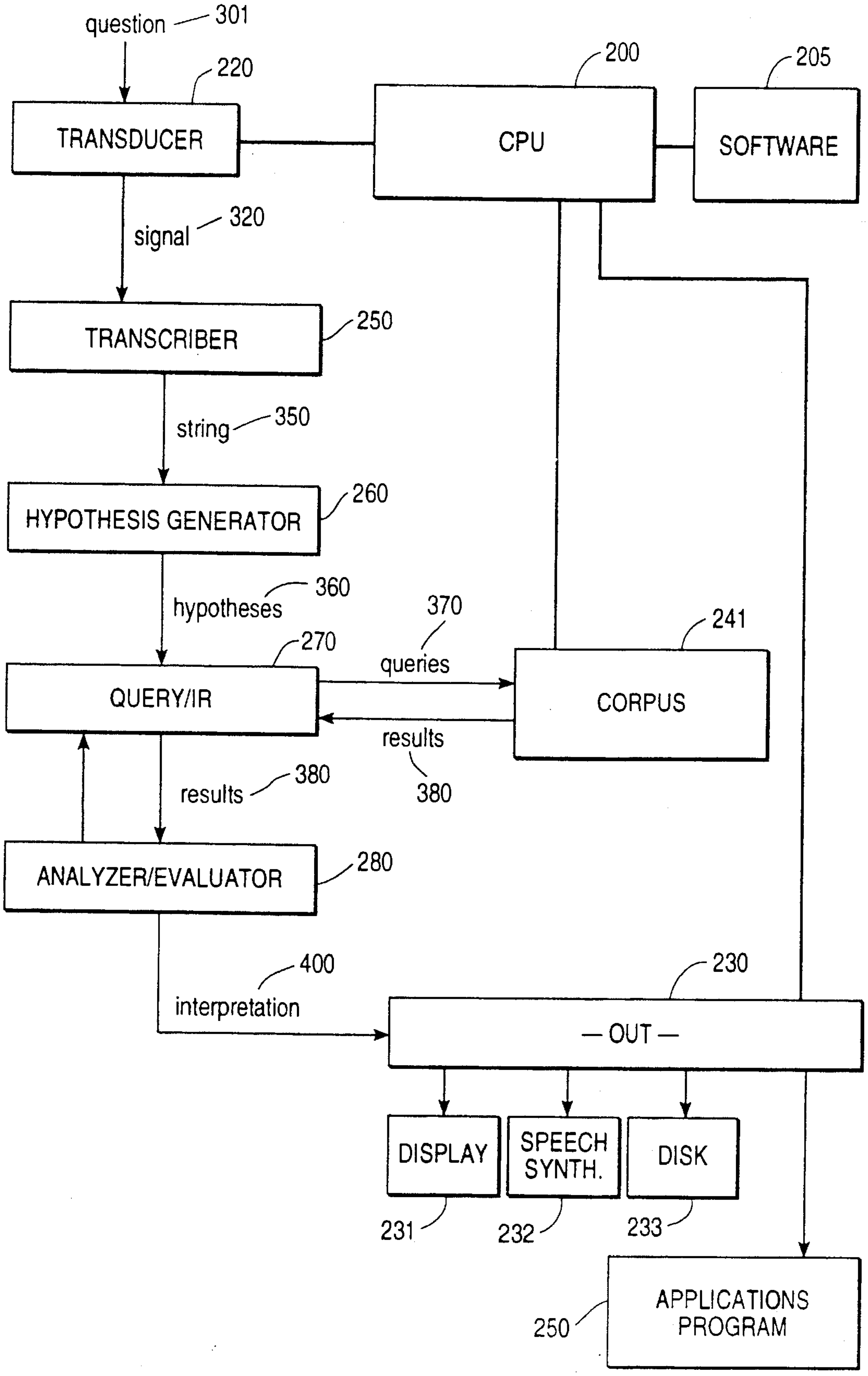

question
301
220
TRANSDUCER
200
CPU
205
SOFTWARE
signal
320
TRANSCRIBER
250
string
350
HYPOTHESIS GENERATOR
260
hypotheses
360
370
queries
241
270
QUERY/IR
CORPUS
results
380
results
380
ANALYZER/EVALUATOR
280
400
interpretation
230
— OUT —
DISPLAY
SPEECH SYNTH.
DISK
231
232
233
250
APPLICATIONS PROGRAM

FIG. 11

SEMANTIC CO-OCCURRENCE FILTERING FOR SPEECH RECOGNITION AND SIGNAL TRANSCRIPTION APPLICATIONS

This is a continuation of application Ser. No. 08/126,170, filed Sep. 23, 1993, now abandoned.

COPYRIGHT NOTIFICATION

SOFTWARE APPENDIX

An appendix comprising 71 pages is included as part of this application. The appendix provides two (2) files of a source code software program for implementation of an embodiment of the method of the invention on a digital computer.

The files reproduced in the appendix represent unpublished work that is Copyright ©1993 Xerox Corporation. All rights reserved. Copyright protection claimed includes all forms and matters of copyrightable material and information now allowed by statutory or judicial law or hereafter granted, including without limitation, material generated from the software programs which are displayed on the screen such as icons, screen display looks, etc.

BACKGROUND OF THE INVENTION

The present invention relates to systems and methods for transcribing words from a form convenient for input by a human user, e.g., spoken or handwritten words, into a form easily understood by an applications program executed by a computer, e.g., text. In particular, it relates to transcription systems and methods appropriate for use in conjunction with computerized information-retrieval (IR) systems and methods, and more particularly to speech-recognition systems and methods appropriate for use in conjunction with computerized information-retrieval systems and methods used with textual databases.

In prior art IR systems, the user typically enters input—either natural-language questions, or search terms connected by specialized database commands—by typing at a keyboard. Few IR systems permit the user to use speech input, that is, to speak questions or search strings into a microphone or other audio transducer. Systems that do accept speech input do not directly use the information in a database of free-text natural-language documents to facilitate recognition of the user's input speech.

The general problem of disambiguating the words contained in an error-prone transcription of user input arises in a number of contexts beyond speech recognition, including but not limited to handwriting recognition in pen-based computers and personal digital assistants (e.g., the Apple Newton) and optical character recognition. Transcription of user input from a form convenient to the user into a form convenient for use by the computer has any number of applications, including but not limited to word processing programs, document analysis programs, and, as already stated, information retrieval programs. Unfortunately, computerized transcription tends to be error-prone.

SUMMARY OF THE INVENTION

The present invention provides a technique for using information retrieved from a text corpus to automatically disambiguate an error-prone transcription, and more particularly provides a technique for using co-occurrence information in the corpus to disambiguate such input. According to the invention, a processor accepts an input question. The processor is used to generate a hypothesis, typically as to a first word and a second word in the input question, and then is used to gather confirming evidence for the hypothesis by seeking a co-occurrence of the first word and the second word in a corpus.

In one aspect, the present invention provides a system and method for automatically transcribing an input question from a form convenient for user input into a form suitable for use by a computer. The question is a sequence of words represented in a form convenient for the user, such as a spoken utterance or a handwritten phrase. The question is transduced into a signal that is converted into a sequence of symbols. A set of hypotheses is generated from the sequence of symbols. The hypotheses are sequences of words represented in a form suitable for use by the computer, such as text. One or more information retrieval queries are constructed and executed to retrieve documents from a corpus (database). Retrieved documents are analyzed to produce an evaluation of the hypotheses of the set and to select one or more preferred hypotheses from the set. The preferred hypotheses are output to a display, speech synthesizer, or applications program. Additionally, retrieved documents relevant to the preferred hypotheses can be selected and output.

In another aspect, the invention provides a system and method for retrieving information from a corpus of natural-language text in response to a question or utterance spoken by a user. The invention uses information retrieved from the corpus to help it properly interpret the user's question, as well as to respond to the question.

The invention takes advantage of the observation that the intended words in a user's question usually are semantically related to each other and thus are likely to co-occur in a corpus within relatively close proximity of each other. By contrast, words in the corpus that spuriously match incorrect phonetic transcriptions are much less likely to be semantically related to each other and thus less likely to co-occur within close proximity of each other. The invention retrieves from the corpus those segments of text or documents that are most relevant to the user's question by hypothesizing what words the user has spoken based on a somewhat unreliable, error-prone phonetic transcription of the user's spoken utterance, and then searching for co-occurrences of these hypothesized words in documents of the corpus by executing Boolean queries with proximity and order constraints. Hypotheses that are confirmed by query matching are considered to be preferred interpretations of the words of the user's question, and the documents in which they are found are considered to be of probable relevance to the user's question.

A further understanding of the nature and advantages of the invention will become apparent by reference to the remaining portions of the specification and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates a system that embodies the invention;

FIG. 2 schematically depicts information flow in a system according to a first specific embodiment of the invention;

FIG. **3** is a flowchart of method steps carried out according to a first specific embodiment of the invention;

FIG. **4** illustrates a conceptual model of a portion of a phonetic index;

FIG. **5** is a flowchart of steps for phonetic index matching;

FIG. **6** is a flowchart of steps for query reformulation;

FIG. **7** is a flowchart of steps for scoring;

FIG. **8** schematically depicts an example of information flow in a system according to a second specific embodiment of the invention;

FIG. **9** is a flowchart of method steps carried out according to a second specific embodiment of the invention;

FIG. **10** illustrates a system in which the invention is used as a "front end" speech-recognizer component module in the context of a non-information-retrieval application; and

FIG. **11** is a specific embodiment that is adaptable to a range of input sources, hypothesis generation mechanisms, query construction mechanisms, and analysis techniques.

DESCRIPTION OF SPECIFIC EMBODIMENTS

The disclosures in this application of all articles and references, including patent documents, are incorporated herein by reference.

1. Introduction

The invention will be described in sections 1 through 6 with respect to embodiments that accept user input in the form of spoken words and that are used in information retrieval (IR) contexts. In these embodiments, the invention enables a person to use spoken input to access information in a corpus of natural-language text, such as contained in a typical IR system. The user is presented with information (e.g., document titles, position in the corpus, words in documents) relevant to the input question. Some of these embodiments can incorporate relevance feedback.

The invention uses information, particularly co-occurrence information, present in the corpus to help it recognize what the user has said. The invention provides robust performance in that it can retrieve relevant information from the corpus even if it does not recognize every word of the user's utterance or is uncertain about some or all of the words.

A simple example illustrates these ideas. Suppose that the corpus comprises a database of general-knowledge articles, such as the articles of an encyclopedia, and that the user is interested in learning about President Kennedy. The user speaks the utterance, "President Kennedy," which is input into the invention. The invention needs to recognize what was said and to retrieve appropriate documents, that is, documents having to do with President Kennedy. Suppose further that it is unclear whether the user has said "president" or "present" and also whether the user has said "Kennedy" or "Canada." The invention performs one or more searches in the corpus to try to confirm each of the following hypotheses, and at the same time, to try to gather documents that are relevant to each hypothesis:

president	kennedy
present	kennedy
president	canada
present	canada

The corpus is likely to include numerous articles that contain phrases such as "President Kennedy," "President John F. Kennedy," and the like. Perhaps it also includes an article on "present-day Canada," and an article that contains the phrase "Kennedy was present at the talks . . . ." It does not include any article that contains the phrase "president of Canada" (because Canada has a prime minister, not a president).

The invention assumes that semantically related words in the speaker's utterance will tend to appear together (co-occur) more frequently in the corpus. Put another way, the invention assumes that the user has spoken sense rather than nonsense, and that the sense of the user's words is reflected in the words of articles of the corpus. Thus the fact that "President Kennedy" and related phrases appear in the corpus much more frequently than phrases based on any of the other three hypotheses suggests that "President Kennedy" is the best interpretation of the user's utterance and that the articles that will most interest the user are those that contain this phrase and related phrases. Accordingly, the invention assigns a high score to the articles about President Kennedy and assigns lower scores to the article about present-day Canada and the article about Kennedy's presence at the talks. The highest-scoring articles can be presented to the user as a visual display on a computer screen, as phrases spoken by a speech synthesizer, or both. Optionally, the user can make additional utterances directing the invention to retrieve additional documents, narrow the scope of the displayed documents, and so forth, for example, "Tell me more about President Kennedy and the Warren Commission."

The present invention finds application in information retrieval systems with databases comprising free (unpreprocessed) natural-language text. It can be used both in systems that recognize discrete spoken words and in systems that recognize continuous speech. It can be used in systems that accommodate natural-language utterances, Boolean/proximity queries, special commands, or any combination of these.

More generally, the invention finds application in speech-recognition systems regardless of what they are connected to. A speech recognizer that embodies or incorporates the method of the invention with an appropriate corpus or corpora can be used as a "front end" to any application program where speech recognition is desired, such as, for example, a word-processing program. In this context, the invention helps the application program "make more sense" of what the user is saying and therefore make fewer speech-recognition mistakes than it would otherwise. This is discussed further in section 7 below.

Still more generally, the invention finds application beyond speech-recognition in handwriting recognition, optical character recognition, and other systems in which a user wishes to input words into a computer program in a form that is convenient for the user but easily misinterpreted by the computer. This is discussed further in Section 8 below. The technique of the present invention, in which a sequence of words supplied by a user and transcribed by machine in an error-prone fashion is disambiguated and/or verified by automatically formulating alternative hypotheses about the correct or best interpretation, gathering confirming evidence for these hypotheses by searching a text corpus for occurrences and co-occurrences of hypothesized words, and analyzing the search results to evaluate which hypothesis or hypotheses best represents the user's intended meaning, is referred to as semantic co-occurrence filtering.

2. Glossary

The following terms are intended to have the following general meanings:

Corpus: A body of natural language text to be searched, used by the invention. Plural: corpora.

Document match: The situation where a document satisfies a query.

FSM, finite-state recognizers: A device that receives a string of symbols as input, computes for a finite number of steps, and halts in some configuration signifying that the input has been accepted or else that it has been rejected.

Hypothesis: A guess at the correct interpretation of the words of a user's question, produced by the invention.

Inflected form: A form of a word that has been changed from the root form to mark such distinctions as case, gender, number, tense, person, mood, or voice.

Information retrieval, IR: The accessing and retrieval of stored information, typically from a computer database.

Keyword: A word that received special treatment when input to the invention; for example, a common function word or a command word.

Match sentences: Sentences in a document that cause or help cause the document to be retrieved in response to a query. Match sentences contain phrases that conform to the search terms and constraints specified in the query.

Orthographic: Pertaining to the letters in a word's spelling.

Phone: A member of a collection of symbols that are used to describe the sounds uttered when a person pronounces a word.

Phonetic transcription: The process of transcribing a spoken word or utterance into a sequence of constituent phones.

Query: An expression that is used by an information retrieval system to search a corpus and return text that matches the expression.

Question: A user's information need, presented to the invention as input.

Root form: The uninflected form of a word; typically, the form that appears in a dictionary citation.

Utterance: Synonym for question in embodiments of the invention that accept spoken input.

Word index: A data structure that associates words found in a corpus with all the different places such words exist in the corpus.

3. System Components

Certain system components that are common to the specific embodiments of the invention described in sections 4, 5, and 6 will now be described.

FIG. **1** illustrates a system **1** that embodies the present invention. System **1** comprises a processor **10** coupled to an input audio transducer **20**, an output visual display **30**, an optional output speech synthesizer **31**, and an information retrieval (IR) subsystem **40** which accesses documents from corpus **41** using a word index **42**. Also in system **1** are a phonetic transcriber **50**, a hypothesis generator **60**, a phonetic index **62**, a query constructor **70**, and a scoring mechanism **80**. Certain elements of system **1** will now be described in more detail.

Processor **10** is a computer processing unit (CPU). Typically it is part of a mainframe, workstation, or personal computer. It can comprise multiple processing elements in some embodiments.

Transducer **20** converts a user's spoken utterance into a signal that can be processed by processor **10**. Transducer **20** can comprise a microphone coupled to an analog-to-digital converter, so that the user's speech is converted by transducer **20** into a digital signal. Transducer **20** can further comprise signal-conditioning equipment including components such as a preamplifier, a pre-emphasis filter, a noise reduction unit, a device for analyzing speech spectra (e.g., by Fast Fourier Transform), or other audio signal processing devices in some embodiments. Such signal-conditioning equipment can help to eliminate or minimize spurious or unwanted components from the signal that is output by transducer **20**, or provide another representation (e.g., spectral) of the signal.

Display **30** provides visual output to the user, for example, alphanumeric display of the texts or titles of documents retrieved from corpus **41**. Typically, display **30** comprises a computer screen or monitor.

Speech synthesizer **31** optionally can be included in system **1** to provide audio output, for example, to read aloud portions of retrieved documents to the user. Speech synthesizer **31** can comprise speech synthesis hardware, support software executed by CPU **10**, an audio amplifier, and a speaker.

IR subsystem **40** incorporates a processor that can process queries to search for documents in corpus **41**. It can use processor **10** or, as shown in FIG. **1**, can have its own processor **43**. IR subsystem **40** can be located at the same site as processor **10** or can be located at a remote site and connected to processor **10** via a suitable communication network.

Corpus **41** comprises a database of documents that can be searched by IR subsystem **40**. The documents comprise natural-language texts, for example, books, articles from newspapers and periodicals, encyclopedia articles, abstracts, office documents, etc.

It is assumed that corpus **41** has been indexed to create word index **42**, and that corpus **41** can be searched by IR subsystem **40** using queries that comprise words (search terms) of word index **42** with Boolean operators and supplemental proximity and order constraints expressible between the words. This functionality is provided by many known IR systems. Words in word index **42** can correspond directly to their spellings in corpus **41**, or as is often the case in IR systems, can be represented by their root (uninflected) forms.

Transcriber **50**, hypothesis generator **60**, phonetic index **62**, query constructor **70**, and scoring mechanism **80** are typically implemented as software modules executed by processor **10**. The operation and function of these modules is described more fully below for specific embodiments, in particular with reference to the embodiments of FIGS. **2** and **8**. It will be observed that corresponding elements in FIGS. **1**, **2**, **8**, and **10** are similarly numbered.

3.1 Query Syntax

It is assumed that IR subsystem **40** can perform certain IR query operations. IR queries are formulated in a query language that expresses Boolean, proximity, and ordering or sequence relationships between search terms in a form understandable by IR subsystem **40**. For purposes of discussion the query language is represented as follows:

term	represents the single search term term. A term can be an individual word or in some cases another query.
<p term1 term2 . . . >	represents strict ordering of terms. The IR subsystem determines that a document matches this query if and only if all the terms enclosed in the angle brackets appear in the

-continued

	document within a sequence containing a maximum of p intervening words between the first and last words of the sequence (that is, a sequence of at most p+2 words) and in thc exact order in which they appear in the brackets. The query <0 phrase> matches only the exact wording of phrase phrase. Strict ordering queries can be nested as in, for example, the query <5 <0 Abraham Lincoln> president>
(p term1 term2 . . .)	represents terms within a proximity of p words from one another, with no strict ordering imposed. The IR subsystem determines that a document matches this query if and only if all the terms enclosed in parentheses appear in the document within a sequence containing a maximum of p intervening words. For example, the query (3 big white ball) matches a document containing a sentence that begins "Taking up the white ball, with the big bat in its hands, the gorilla began to play baseball . . ." because the sequence that begins with the first term matched ("white") and ends with the last term matched ("big") has no more than 3 intervening words between the first and last words of the sequence. The order of the terms within the sequence is not considered for this query. Proximity queries can be nested and can also contain strict ordering queries as in, for example, the query (20 <0 Abraham Lincoln> (10 slavery freedom))
[term1 term2 . . .]	represents a Boolean logical AND of terms. The IR subsystem determines that a document matches this query if and only if each of the terms within the square brackets occurs at least once in the document. AND queries can include proximity queries or strict ordering queries, as in, for example, the query [<0 Abraham Lincoln> (10 slavery freedom)]
{term1 term2 . . . }	represents a Boolean logical OR of terms. The IR subsystem determines that a document matches this query if any of the terms within the curly brackets occurs at least once in the document. OR queries are commonly used as terms in AND, proximity, or strict ordering queries, as in, for example, the query (10 {president leader} {Lincoln Washington Kennedy})
/term1 term2 . . . \	represents a Boolean logical NOT of terms. The IR subsystem determines that a document does NOT match this query if any of the terms between the slash and backslash occurs at least once in the document. NOT queries are commonly used as a limitation on other kinds of query, as in, for example, the query (10 president Washington) /<2 District Columbia>\

3.2 Discrete-Word and Continuous Speech

In some embodiments, the invention accepts discrete-word speech input. Typically, the user is expected to pause between each spoken word, so that the system can readily determine where one spoken word ends and the next begins. The system is thus freed of the task of determining word boundaries within the user's speech. In other embodiments, the invention accepts continuous speech input and attempts to determine the word boundary positions for itself.

Restricting the user to discrete-word speech increases computational efficiency, because the invention has fewer possible interpretations of the user's utterance to consider. An example illustrates why this is so. Suppose that the user speaks the phrase "how to recognize speech." In a continuous-speech embodiment, the invention must determine the word boundaries in this utterance; for example, it must decide whether the user has said "how to recognize speech" or "how to wreck a nice beach" or even "powdered egg and iced peach." The number of possible interpretations of a given utterance increases greatly when continuous speech is permitted.

There are at least two ways for the user to make word boundaries explicit:

a) By speaking the words with sufficient pauses between them to enable a word end-point detector to delineate the words.

b) By recognizing a set of specific words (called keywords) and providing a phonetic transcription for the speech not recognized as keywords. Keywords can include common function words (e.g., a, the, of, it, etc.) that can be ignored when they occur in the spoken input, and also command words that can signify special IR operations. Such operations can include, for example:

1. Boolean operations such as AND, OR and NOT. The NOT operation specifies that words formed from a subsequent phonetic transcription are not to be present

in documents when matching is performed. Related operations include parenthesizing words that are to be grouped together in a Boolean operation, and explicitly specifying the proximity constraints to be used.

2. An operation that specifies that words formed from a subsequent phonetic transcription are to be treated as a strict sequence when matching is performed. This provides extra constraint when words are known to be likely to occur in a specific sequence, e.g., "atomic bomb" is more likely to occur as a phrase than "bomb atomic".

4. A First Specific Embodiment

The invention will now be described with reference to a first specific embodiment. This embodiment accepts discrete-word rather than continuous speech input. Keywords are not supported in this embodiment. The system of this embodiment of the invention is the system of FIG. **1**.

FIG. **2** illustrates the information flow in the first specific embodiment. The user inputs a question **201** into system **1** by speaking into audio transducer **20**. The signal **220** produced by transducer **20** is fed to transcriber **50**, where it is converted into a phonetic transcription **250**.

Transcriber **50** can be implemented using any of a variety of transcription techniques. One such technique, well-known among those of skill in the art, involves the use of statistical models called hidden Markov models. See. e.g., Lawrence R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE,* vol. 77, no. 2, Feb. 1989, pp. 257–285.

The phonetic transcription **250** is an ordered sequence of phones, that is, of component sounds that can be used to form words. Because the input speech is discrete-word speech in this embodiment, the phonetic transcription **250** comprises several smaller ordered sequences of phones, each such smaller sequence being a phonetic transcription of a single word of the user's utterance. Typically transcriber **50** is error-prone and produces a phonetic transcription **250** that is imperfect.

The phonetic transcription **250** is provided to hypothesis generator **60** where it is matched using phonetic index **62** to generate a set of hypotheses **260**. In broad outline, hypothesis generation proceeds as follows: Within phonetic transcription **50** are one or more smaller sequences of phones, each of which corresponds to a single word spoken by the user. Each such smaller sequence of phones is analyzed, and sets of alternative phone sequences are developed. The phone sequences and their alternatives are compared against word pronunciations stored in phonetic index **62** to determine candidate words, that is, words that could be the words spoken by the user. Candidate words are concatenated to form hypotheses that represent possible interpretations of the user's entire utterance.

More particularly, hypothesis generation proceeds as follows: Because the transcriber **50** is known to be error-prone, hypothesis generator **60** develops alternative possible transcriptions for each word spoken by the user, in addition to the original phone sequences provided by transcriber **50**. For example, hypothesis generator **60** can attempt to correct mistakes commonly made by transcriber **50** by adding, deleting, or substituting one or more phones into the sequence of phones that represents the word as originally transcribed. Such "correction" can be based, for example, on a statistical model of the performance of transcriber **50**. Hypothesis generator **60** thus systematically generates different possible "corrected" versions of the word's transcription. Probabilities can optionally be associated with each alternate "corrected" transcription to express its relative likelihood based on the statistical model.

An example illustrates how "corrected" versions of a transcription can be developed. If the user speaks the word "president," it can be phonetically transcribed into a phone sequence (an erroneous phone sequence) such as

<SIL P R EH S IH D R N T SIL>

where SIL represents a silence. If hypothesis generator **60** has information about the mistakes commonly made by transcriber **50** that includes the fact that transcriber **50** commonly outputs the phone "R" where the user intended no phone or the phone "EH", and commonly outputs the phone "D" where the user intended no phone, then it can develop the following alternative versions:

<SIL P R EH S IH D EH N T SIL>	("president")
<SIL P R EH S IH D N T SIL>	("president")
<SIL P R EH S IH R N T SIL>	(a nonsense word)
<SIL P R EH S IH EH N T SIL>	("prescient")
<SIL P R EH S IH N T SIL>	("present")

The hypothesis generator matches the original and "corrected" versions of the word transcription against phonetic index **62** to determine whether any of them match any words in phonetic index **62**. This matching process is described in more detail below with reference to FIGS. **4** and **5**. Hypothesis generator **60** considers each word that is matched in phonetic index **62** to be a candidate—a possibly valid interpretation of the user's intended word.

Hypothesis generator **60** repeats the matching process for each word of the user's utterance until it has candidates for all the words of the utterance. It then concatenates all possible combinations of candidates according to the sequence of words in the utterance to generate the set of hypotheses **260**. If the user's utterance is a sequence of N words W_i, then the hypotheses are of the form

$<$candidate(W_1) candidate(W_2) . . . candidate(W_i) . . . candidate(W_N)$>$

For example, if the user speaks two words, and the candidates for the first word are "precedent," "president," "resident," "prescient," and "present," and the candidates for the second word are "kennedy," "kenny," "canada," and "tenant," then these hypotheses are generated:

precedent kennedy
president kennedy
resident kennedy
prescient kennedy
present kennedy
precedent kenny
president kenny
resident kenny
prescient kenny
present kenny
precedent canada
president canada
resident canada
prescient canada
present canada
precedent tenant
president tenant
resident tenant
prescient tenant
present tenant

If there are n_i phonetic index matches (that is, n_i candidates) for the ith word of the sequence, then the number of hypotheses is

$$n_1xn_2xn_3x. . . xn_ix. . . xn_W.$$

In the above example, there are 5 candidates for the first word and 4 for the second, for a total of 20 hypotheses. (It will be appreciated that in some implementations, the hypotheses can be represented more compactly as a sequence of the candidate sets, so that each individual hypothesis need not be explicitly represented as in this example.)

Occasionally, no candidates will be found for one or more words of the user's utterance. This can happen, for example, if part of the utterance is garbled. In this case, hypothesis generator **60** can omit the unrecognized word from the generated hypotheses. Alternatively, hypothesis generator **60** can halt processing of the user's question and prompt the user to repeat the question. This course of action can be adopted, for example, if none of the user's words is recognized.

Once the set of hypotheses **260** has been generated, it is provided to query constructor **70**. Query constructor **70** uses the hypotheses **260** to construct one or more queries **270** that will be sent to IR subsystem **40** for execution. Queries **270** are Boolean queries with proximity and order constraints. In this embodiment, an initial query that is constructed is of the form:

```
(k      {all candidates for word 1}
        {all candidates for word 2}
        . . .
        {all candidates for word i}
        . . . )
```

Here, k is a proximity constraint value, e.g., 5, 10, or 20. For example, suppose that the user speaks two words, and the set of hypotheses **260** is:

president	kennedy
present	kennedy
president	canada
present	canada

Then if k=10, the initial query that query constructor **70** constructs is:

(10 {present president} {kennedy canada})

This query seeks occurrences of at least one of the words (search terms) "present" or "president" within a proximity of 10 words of at least one of the words "kennedy" or "canada." The initial query is sent to the IR subsystem **40** where it is executed.

Depending on the results obtained from execution of the initial query, additional queries can be constructed and executed, in a process called query reformulation. For example, if no matches are found for the initial query, query constructor **70** can increase the proximity value k, for example to 20, and send the query thus modified back to IR subsystem **40** to be executed again. Alternatively or in addition, query constructor **70** can drop one or more words from the query. This can be helpful, for example, if one of the user's intended words is not present in phonetic index **62**, so that none of the candidates for this word is correct. Query reformulation is described in further detail with reference to FIG. **6** below. In general, a series of queries **270** is constructed by query constructor **70** and provided to IR subsystem **40**, which executes them by conducting searches in accordance with queries **270** over corpus **41**.

The execution of the initial and any additional queries causes a set of documents **240** to be retrieved from corpus **41**. Each of the retrieved documents **240** matches one or more of the queries **270**, that is, contains search terms of one or more of the queries **270** with the specified proximity and order relationships. For example, a document that contains the phrase "President John F. Kennedy" matches the query (10 {present president} {kennedy canada}), because it contains the search terms "president" and "kennedy" within 10 words of each other.

The retrieved documents **240** and the query matches that they contain are provided along with hypotheses **260** to scoring mechanism **80**. Scoring mechanism **80** assigns scores to the various hypotheses **260** according to probable relevance to the user's input question **201** and ranks the hypotheses **260** according to the scores thus assigned. This provides the invention with the ability to determine which hypothesis or hypotheses best match the user's intended utterance. Scoring mechanism **80** outputs a set of results **280** that comprises the top-ranked hypotheses, and can in association with these hypotheses further comprise the retrieved documents that support the hypotheses, the queries used to retrieve those documents, and the matched search terms within the retrieved documents.

A hypothesis receives a score based on the number of query matches it generates. For example, if the hypothesis "president kennedy" is being scored, it receives a point for each of its occurrences in the corpus, that is, for each instance in which the words "president" and "kennedy" were found in the corpus within the desired proximity of one another. Additional or different scoring criteria can be used in other embodiments. Such criteria can include, for example, the number of distinct documents in which a hypothesis occurs; the total number of occurrences of the hypothesis or its constituent words in any one document; the number of words of the hypothesis that appear in a document title; and the probability scores associated with the "corrected" transcriptions that gave rise to the hypothesis in the first place. Documents can be scored along with the hypotheses to determine, for any given hypothesis, which documents are most likely to be relevant to that hypothesis. Scoring is described in further detail with reference to FIG. **7** below.

When scoring is finished, the results **280** can be presented to the user using processor **10** in conjunction with visual display **30**. Typically, the user's question as interpreted according to the best of the hypotheses **260** is displayed in conjunction with the titles of a reasonable number (e.g., between 1 and 30) of the highest-ranked retrieved documents. Excerpts of the documents showing the occurrence of the matched search terms therein are also typically displayed. Additional output can be made using processor **10** in conjunction with optional speech synthesizer **31** if such synthesizer is included in system **1**. The speech output can be, for example, a synthetic reading of document titles or selected text portions from retrieved documents.

The flowchart of FIG. **3** summarizes the method or processing steps performed by the system of FIG. **2**. First the system accepts a user utterance as input (Step A). This utterance is converted to a signal (Step B) that is transcribed into a sequence of phones (Step C). The phone sequence is used to generate hypotheses (Step D). Boolean queries with proximity and order constraints are constructed based on the hypotheses and are executed to retrieve documents (Step E). Hypotheses are scored in order of relevance (Step F). A relevant subset of the hypotheses and retrieved documents is presented to the user (Step G).

4.1 Phonetic Index Matching

The process of matching in the phonetic index, which is part of hypothesis generation, will now be examined in more detail with reference to FIGS. **4** and **5**.

Phonetic index **62** is a data structure that stores word spellings (orthographic forms) in association with word pronunciations (spoken forms). Typically, each word in phonetic index **62** is associated with its most common pronunciation(s). Pronunciations are represented as phone sequences in a form that can readily be compared with the phone sequences produced by transcriber **50** and "corrected" versions of these sequences produced by hypothesis generator **60**.

FIG. **4** depicts a conceptual model of a portion of phonetic index **62**. In this model, phonetic index **62** is represented as a table **310** that comprises entries **315**. Each of the entries **315** comprises a spelling and a pronunciation for a single word. Some entries can include two or more spellings associated with a single pronunciation, to indicate that two or more words sound alike (homonyms), for example, "canon" and "cannon." In cases where a single word is subject to more than one pronunciation, there can be (but need not be) multiple entries corresponding to the same word. Alternatively, a single entry can include multiple pronunciations. It will be observed that because hypothesis generator **60** develops alternative pronunciations, in many instances a user's variant pronunciation of a word can be converted by hypothesis generator **60** to the pronunciation of the word as found in phonetic index **62**.

Although table **310** represents conceptually the association between pronunciations and orthographic spellings in phonetic index **62**, it will be appreciated by those of skill in the art that a linear search through table **310** is computationally inefficient. Accordingly, in practice a more compact and computationally efficient representation of phonetic index **62** is preferable. In one such representation, the orthographic spellings are stored in a vector and the phone sequences are represented by a finite-state network that provides an index (or indices) into the vector. This technique is substantially similar to the minimal perfect hashing technique described in Lucchesi, Claudio L. and Tomasz Kowaltowski, "Applications of Finite Automata Representing Large Vocabularies," *Software—Practice and Experience,* vol. 23(1), pp. 15–30, January 1993 (see especially pp. 26–27). Other representations that can be used include word-trie representations.

The words included in phonetic index **62** are words that the user is likely to speak as input. Typically, they include some or all of the words of word index **42**, as well as additional keywords, such as command words to be treated specially on input and common function words to be ignored on input. Because the words of phonetic index **62** are taken from word index **42**, they are guaranteed to be present in corpus **41**. This means that hypotheses **260** generated using phonetic index **62** contain words from corpus **41** and so are suitable for use in query construction by query constructor **70**.

It will be appreciated that the vocabulary of phonetic index **62** need not be exhaustive: Although all words in phonetic index **62** appear in word index **42** and thus in corpus **41**, there can be words in word index **42** or in corpus **41** that do not appear in phonetic index **62**. If transcriber **50** produces a phone sequence for which phonetic index **62** contains no word, hypothesis generator **60** can nevertheless generate a hypothesis or hypotheses. Most often, hypothesis generator **60**, through its attempts to "correct" the pronunciation of a word, can generate a spurious match, that is, a match to a word that is present in phonetic index **62** but that the user did not intend. The query construction process is sufficiently robust to accommodate the absence of correct candidates for one or more words of a hypothesis; in particular, query constructor **70** can drop search terms from a query during query reformulation if few matches are found. Less commonly, if no matches whatsoever are found in phonetic index **62**, hypothesis generator **60** can simply omit the unmatched word and generate a hypothesis based on the remaining words in the user's utterance.

FIG. **5** flowcharts the steps involved in matching phonetic transcriptions. These steps are carried out by hypothesis generator **60**. FIG. **5** is an expansion of a portion of Step D of the flowchart of FIG. **3**.

A loop is executed for each word of the user's utterance (Step DA). First an attempt is made to match the sequence of phones corresponding to the word to a word or words in phonetic index **62** (Step DB). The matching involves a phone-by-phone comparison between the phone sequence and entries in the index. An exact match is required; that is, the phone sequence as transcribed must be exactly the same as the phone sequence of a stored word pronunciation in order for a match to occur.

Thereafter, the phones of the phone sequence are examined for possible substitutions, additions, or deletions according to a statistical performance model of transcriber **50**. Substitutions, additions, and deletions are tried systematically, alone and in combinations, to generate a series of alternative, "corrected" pronunciations. Various search strategies can be used to determine what order to apply the substitutions, additions, and deletions; in general, the most likely changes are tried sooner and the less likely changes are tried later.

The statistical performance model of transcriber **50** can be implemented through a set of tables. For each phone, the substitution table contains the various probabilities that the correct transcription for the phone is in fact a different phone (e.g., the probability that the phone "b" in the output of transcriber **50** corresponds to another phone that was actually spoken, such as "p"). The substitution table is ordered in terms of decreasing substitution probabilities. In like manner, there are also insertion and deletion tables, similarly ordered, for insertion and deletion errors respectively. An insertion error means that an extra phone was inserted by transcriber **50** where none was intended by the speaker, and a deletion error means that transcriber **50** omitted a phone that was in fact intended by the speaker. The probabilities of substitution, insertion, and deletion errors are estimated with the aid of an alignment program, which is one of several methods available for estimating the best correspondence of a known transcription of a speech signal with the phonetic output from transcriber **50**. Probabilities can be calculated independently of the phonetic context in which they occur, or can be based on context to provide more accurate probabilities (for example, the likelihood that the transcriber will register the phone "k" in the word "keep" can be different from the likelihood that it will register the phone "k" for the somewhat different k-sound in the word "claw").

As each new alternative pronunciation is generated (Step DC), an attempt is made to find a corresponding word or words in phonetic index **62** (Step DD). This matching proceeds in the much same manner as for the original phone sequence: through a phone-by-phone comparison between the "corrected" alternative and the stored pronunciations. Once again, an exact match is required.

As matches are found, they are recorded (Step DE). Also recorded in association with each match is a probability that represents the relative likelihood, based on the statistical model, that the particular transcription errors that give rise to this "corrected" alternative would in fact have occurred (Step DF).

The development of alternative pronunciations and the search for matches continues until either a certain number of matches (e.g., 30) is found (Step DG), or until a certain number of consecutive unsuccessful match attempts (e.g., 15,000) occurs (Step DH).

4.2 Query Reformulation

FIG. **6** flowcharts the steps involved in query reformulation. These steps are carried out by query constructor **70** in conjunction with information retrieval subsystem **40**. FIG. **6** is an expansion of a portion of Step E of the flowchart of FIG. **3**.

Query reformulation is the process of modifying the initial query constructed by query constructor **70** and executing the query thus modified using IR subsystem **40**. The initial query can be modified and re-run once, many times, or not at all, depending on the results obtained at each from executing each intermediate query.

Upon execution of the initial query, the IR subsystem returns a set of documents that match the search terms and constraints of the query. To determine whether any additional queries are necessary, query constructor **70** performs a preliminary analysis of the returned documents (Step EA). In this embodiment, query constructor **70** counts the number of documents returned. The number of documents is then tested against a predefined minimum value such as 15 (Step EB) and a predefined maximum value such as 50 (Step EC). If the number of documents is reasonable, that is, greater than or equal to the minimum value and less than the maximum value, then no additional queries are deemed necessary (Step ED).

If there are too few retrieved documents, then an attempt is made to broaden the query. To broaden a query is to modify it in such a way that the number of documents likely to be retrieved upon its execution increases. First, a check is made to see whether the query can helpfully be broadened further (Step EE). This check is performed to ensure that queries are not broadened indefinitely, and to prevent an infinite loop of broadening and narrowing operations. If the check succeeds, then the query is broadened (Step EF).

Broadening can be carried out in several ways. The proximity value k can be increased over a range of values, e.g., 10, 20, 40, and the proximity constraint can be then dropped altogether so that a simple AND query over the scope of the entire document is performed. To further broaden the query, one or more individual words can be dropped from the query. For example, if the initial query is

(10 {present president} {kennedy canada})

then increasing the proximity constraint to 20 and then to 40 yields the broadened queries

(20 {present president} {kennedy canada})

(40 {present president} {kennedy canada})

Dropping the proximity constraint in favor of an AND query produces

[{present president} {kennedy canada}]

Dropping the first word of the user's utterance from the query gives

[{kennedy canada}]

and dropping the second word gives

[{present president}]

Words can be selected to be dropped according to the probabilities associated with them during "correction" of transcription; that is, if all the candidates for a particular word are of low probability, this suggests that the word was garbled or otherwise not well-transcribed and should be dropped. Still other broadening techniques can be devised and used, for example, consulting a thesaurus to generate synonyms of search terms.

Broadening by dropping words is particularly important because it can happen that some words of the user's question, even when correctly phonetically transcribed, do not co-occur in relevant documents with other words of the user's question. Dropping words makes it possible for subsets of the words in the transcribed question to be matched against documents. A series of IR queries can be made using different combinations of words from the transcribed question.

If there are too many retrieved documents, then an attempt is made to narrow the query. To narrow a query is to modify it in such a way that the number of documents likely to be retrieved upon its execution decreases. First, a check is made to see whether the query can helpfully be narrowed further (Step EG). This check is performed to ensure that queries are not narrowed indefinitely, and to prevent an infinite loop of broadening and narrowing operations. If this check succeeds, then the query is narrowed (Step EH).

Narrowing can be carried out in several ways. The proximity constraint can be narrowed, either overall or between pairs of words in the utterance. The latter technique is helpful for words that tend to appear together, such as "united" and "states" in the phrase "United States." Other narrowing techniques include imposing an order constraint on some or all of the user's words, so that in order for a match to occur the words are required to appear in the text in the same order they occurred in the user's utterance. Yet another way to narrow a query is to duplicate individual words in the query, using the same or a broader proximity constraint. This tends to locate documents in which words recur, suggesting that these words are closely related to the topic of the document and thus tending to retain only the most relevant documents.

Once the query has been broadened or narrowed, the query thus modified is sent to IR subsystem **40** where it is executed (Step EI). Results from the broadened or narrowed query are once again analyzed (Step EA) and the count of returned documents is once again checked to see whether too few (Step EB) or too many (Step EC) documents have been returned. If the number of documents is now reasonable, query reformulation stops (Step ED); otherwise, further broadening or narrowing occurs (Steps EE through EI). The loop of broadening or narrowing proceeds until either a reasonable number of documents is found (Step ED) or no further broadening or narrowing is deemed helpful (Step EJ).

If query reformulation terminates successfully (in Step ED) with a reasonable number of documents retrieved, these documents are passed on to scoring mechanism **80** for scoring. If query reformulation terminates unsuccessfully (in Step EJ) with too few or too many documents retrieved, either such documents as have been retrieved can be passed on for scoring, or, alternatively, no documents can be passed on and an error message can be displayed to the user on visual display **30**.

4.3 Scoring

FIG. **7** flowcharts the steps involved in scoring hypotheses. These steps are carried out by scoring mechanism **80**, typically in conjunction with IR subsystem **40**. FIG. **7** is an expansion of a portion of Step F of the flowchart of FIG. **3**.

In overview, scoring is a two-stage process. First, hypotheses are ranked to determine which of the hypotheses are most likely to represent a correct interpretation of the user's utterance—that is, which of the hypotheses probably say what the user intended to say. Second, for each preferred hypothesis, retrieved documents are ranked to determine which retrieved documents are most likely to be relevant to the particular hypothesis.

In more detail, scoring proceeds as follows: A loop is performed over all hypotheses (Step FA). For each hypothesis, the number of matches ("hits") is computed (Step FB). This is the number of places in the corpus where the query whose results were passed on to scoring mechanism **80**—that is, the query considered to be the most successful after any and all broadening and narrowing operations were completed-was satisfied by the hypothesis in question. Typically, scoring mechanism **80** computes the number of hits for a particular hypothesis by scanning the retrieved documents for matched search terms that correspond to the hypothesis. Alternatively, scoring mechanism **80** can compute the number of hits by a conceptually equivalent procedure in which it constructs an auxiliary query that is specific to the particular hypothesis, executes this hypothesis-specific query using IR subsystem **40**, and counts the number of hits returned. The auxiliary query is a specialization for the particular hypothesis of the most successful query executed during query,reformulation, wherein each search term in the query is a word in the particular hypothesis rather than a Boolean OR of all possible candidate words from all hypotheses. For example, if the hypothesis is "president kennedy" and the most successful query executed during query reformulation was

(5 {present president} {kennedy canada})

then the number of "hits" for the hypothesis "president kennedy" is the number of matches to the query

(5 president kennedy)

in the retrieved documents.

After the number of hits for each hypothesis has been determined, the hypotheses are sorted in order from highest to lowest rank (step FC). A subset of the highest-ranked hypotheses is then retained for further processing (Step FD); for example, a predetermined number (e.g., 1, 10, or 50) of the highest-ranking hypotheses can be retained, or hypotheses for which more than a predetermined number of hits (e.g., 5) were retrieved can be retained.

Next, the retrieved documents are scored. A loop over the retained hypotheses is performed (Step FE). For each hypothesis, a loop over the documents which support that hypothesis is performed (Step FF). For each supporting document, a relevance score is computed (Step FG). This score expresses the relative likelihood that the document in question is relevant to the hypothesis in question. If the document has not yet been assigned a score (Step FH), the relevance score is assigned to the document (Step FJ), and the document is associated with the hypothesis (Step FK). Otherwise, if the document has previously received a score (in association with another hypothesis), the document's present score is compared to the relevance score (Step FI). If the relevance score is higher than the document's present score, the relevance score is assigned to the document, replacing its present score (Step FJ), and the document is associated with the hypothesis (Step FK). Thus, upon completion of the loop over hypotheses, each document is assigned to the hypothesis to which it is most likely relevant, and has a score based on the hypothesis to which it is assigned.

Relevance scores are computed in heuristic fashion. They can be based on one or more criteria, including, for example:

(1) the number of times the hypothesis or its constituent words appears within a document;

(2) the occurrence of one or more words of the hypothesis in a document's title;

(3) the probability that the words of the hypothesis represent correct transcriptions of the user's speech, as indicated by the probabilities that were optionally associated with each alternate "corrected" word transcription to express its relative likelihood based on the statistical model of the performance of transcriber **50**.

Once all documents have been scored for all retained hypotheses, the documents are ranked in order from highest to lowest score (Step FL). It will be appreciated that because the documents are associated with particular hypotheses, this ranking of the documents effectively re-ranks the hypotheses as well. A subset of the highest-ranked documents is retained for further processing (Step FM); for example, a predetermined number (e.g., between 1 and 30) of the highest-ranking documents can be retained. The retained documents and the hypotheses with which they are associated are the hypotheses and documents that are subsequently presented to the user on output (Step G of FIG. **3**).

It will be appreciated that alternative methods for scoring and ranking hypotheses and documents can be used in other embodiments. It will be further appreciated that although in the specific embodiment described, scoring is performed separately from query reformulation, these steps can be more tightly integrated in other embodiments. For example, an assessment of document rankings based on a number of simple criteria can be used as a basis for determining whether to construct and execute additional queries.

5. A Second Specific Embodiment

The system and method of the invention will now be described with respect to a second specific embodiment. This second embodiment incorporates user relevance feedback and supports keywords. Like the first specific embodiment described above, it accepts discrete-word rather than continuous speech input. Once again, the system of this embodiment of the invention is the system of FIG. **1**.

FIG. **8** illustrates an example of information flow in the second specific embodiment. Insofar as the information flow is similar in many respects to that shown in FIG. **2** for the first specific embodiment, only the differences between the two embodiments will be pointed out here. The user question **201** is transduced into a signal which is converted into a phonetic transcription **250** by transcriber **50** as in the first embodiment. However, after phonetic transcription **250** is passed on to hypothesis generator **60**, keywords are trapped by hypothesis generator **60** and processed separately from other words. Keywords are assumed to be correctly transcribed, and no checking of alternative or "corrected" pronunciations is performed for keywords. Keywords can be single words or longer phrases.

Three kinds of keywords are supported in this embodiment:

(1) Common function words. These include words such as "a," "an," "the," "of," "any," etc. Words of the input utterance that match a common function word in the phonetic index can be ignored for purposes of hypothesis generation and query construction. For example, if the user's utterance is "climb a mountain," possible hypotheses can be, e.g.,

climb mountain
climber mountain
climb mounting
climber mounting

The word "a" is recognized as a function word and is eliminated from the hypotheses. If the word "a" is misrecognized as "bay," however, it is included in the hypothesis, e.g.,

climb bay mountain

climb bay mounting

and so forth.

(2) Command words. The user can supply keywords that signify special IR operations to be carried out during query construction. This includes the Boolean keywords "and," "or," and "not," and also keywords to indicate that terms should be queried in strict order or within a certain proximity of one another. The special IR operations can be communicated to query constructor **70** via one or more commands **290**, as shown in FIG. **8**.

For example, if the user's question is "president kennedy NOT oswald," query constructor **70** generates queries such as

(10 {president present prescient) {kennedy canada}) \{oswald osgood asphalt}/

Other kinds of command keywords can also be available. For example, a command can indicate that the words that follow it, which ordinarily would be ignored as common function words, are to be included in the hypotheses. This allows the user to cause documents to be retrieved in response to utterances such as "To be or not to be." Other commands can be used to select among competing hypotheses. For example, if the invention determines that the two best interpretations of the user's utterance, as indicated by the highest-ranked hypotheses, are "president kennedy" and "present-day canada," the user can indicate that "president kennedy" is the better choice.

(3) Relevance feedback commands. After the user's question has been processed, so that documents have been retrieved and presented in response to the question, the user has the option of directing the invention to perform a follow-up search based on the retrieved results. For example, suppose the user's initial question is "president kennedy" and that the best hypothesis, "president kennedy," is displayed in conjunction with the titles of several relevant documents, including a document entitled "Warren Commission" (which describes the Warren Commission's investigation of President Kennedy's assassination). At this point, the user has the option of saying, "Tell me more about the Warren Commission." The keyword phrase "Tell me more about" signals the invention that the question that follows is to be treated differently from a normal question and, in particular, that its words can be found among the displayed titles. This means that there is no need for a search in the corpus at large to determine the best interpretation of the user's question. Instead, the displayed document titles are used as a "mini-corpus" to determine the best transcription of the user's question. The correct document title is identified, and additional documents that contain words similar to the words contained in the identified article are retrieved (e.g., using vector space search techniques). These documents can be displayed to the user along with the documents originally retrieved. The user can supply further commands using the titles of the "president kennedy" and "warren commission" documents to cause still further documents to be retrieved. For example, the user can say, "Tell me more about Lee Harvey Oswald but NOT Jack Ruby." To discard or save results and proceed with a new search, the user can say other keywords, e.g., "save," "quit," "new search," etc.

In general, the best matching documents that correspond at any time to the words that the user has spoken so far can be displayed to the user on a screen. Upon seeing the titles (or other descriptive content) the user can speak additional words to direct the search to particular documents or cause them to be excluded by invoking the NOT operation.

The flowchart of FIG. **9** summarizes the processing steps performed according to the method of the second embodiment. The system accepts a user utterance as input (Step AA). This utterance is converted to a signal (Step BB) that is transcribed into a sequence of phones (Step CC). The phone sequence is used to generate hypotheses (Step DD); keywords in the phone sequence are trapped for special processing (Step EE). A test is made to determine whether documents have previously been retrieved since the last "new search" or similar command (step FF). If no documents have been retrieved, then a search for documents is made. Prior to query construction, keywords are processed (Step GG); in particular, common function words can be filtered out of the hypotheses, and IR command words are routed to the query constructor, where they can be interpreted and incorporated into queries. Thereafter, Boolean queries with proximity and order constraints are constructed based on the hypotheses and the keywords, and these queries are executed to retrieve documents (Step HH). Hypotheses are scored in order of relevance (Step JJ). A relevant subset of the hypotheses and retrieved documents is presented to the user (Step KK). If documents have previously been retrieved, then user relevance feedback commands and search terms can be routed to the hypothesis generator, to instruct the hypothesis generator to use retrieved document titles as the basis for confirming hypotheses (Step LL), or to cease doing this upon a "new search" or similar command. The system then can perform operations such as a vector space search or the selection of one among several preferred hypotheses (Step MM). Results of these operations are presented to the user (Step KK).

6. Variations and Extensions

Beyond the specific embodiments described above, additional variations and extensions of the present invention will be apparent to those of skill in the art. Some of these will now be described. This section (section 6) concerns certain alternatives that can be used in implementing embodiments such as those previously described, i.e., embodiments that accept speech input and are used primarily for information retrieval, and can be used in certain other embodiments as well. Section 7 concerns an embodiment that is not limited to IR tasks, and Section 8 concerns an embodiment in which the input can take forms besides speech.

6.1 Alternative Implementations of the Phonetic Index

Phonetic index **62** is subject to a number of alternative implementations in different embodiments of the invention.

If words in word index **42** are represented by their root forms, they can also be represented by their root forms in phonetic index **62**. In such embodiments, for best performance the user is restricted to using root forms in the input question. In other embodiments, in which inflected forms are included in word index **42** and phonetic index **62**, the user can use inflected forms as well as root forms, and thus can speak more naturally when posing questions to the invention. In still other embodiments, although words in word index **42** are represented by their root forms, the spoken (though not the spelled) words in phonetic index **62** are represented by both root and inflected forms. Thus the user's inflected speech is mapped into its root forms as it is transcribed. For example, if the root form "rock" appears in word index **42**, the pronunciations of the root form "rock" and the inflected forms "rocks," "rocked," "rocking," and so forth can all be stored in association with the spelling ROCK in phonetic index **62**. Yet another alternative is to apply phonetic rules to transform inflected spoken words to one or more possible uninflected forms.

More generally, arbitrary spoken pronunciations can be associated with arbitrary written words in phonetic index **62**. This means that the spoken language of the invention need not be the same as the written language of the corpus. Phonetic index **62** can, for example, be constructed from translation(s) of word index **42** into foreign languages, to permit multi-lingual access to corpus **41**.

The construction of phonetic index **62** from word index **42** can be automated. The pronunciations stored in phonetic index **62** can be created automatically from the orthographic spellings of the words in word index **42** by techniques such as those used for text-to-speech systems. Thus the pronunciations for phonetic index **62** can be constructed automatically for any corpus of text.

In some embodiments, the corpus **41** can be composed of phonetically transcribed text, in which case word index **42** and phonetic index **62** can be identical.

6.2 Smoothing

In some embodiments, if the number of matches found in the phonetic index is low (e.g., significantly less than 30), the hypothesis generator can vary the estimates used in the statistical model of transcriber errors in order to obtain improved results. It can do this, for example, by substituting higher values for substitution/insertion/deletion probabilities in the model by "smoothing" with a uniform distribution. Thereafter, the hypothesis generator re-runs phonetic index matching using the new probabilities.

6.3 Further Query Reformulation Strategies

Additional or alternative strategies for query reformulation can be used in some embodiments. For example, in some embodiments, queries can be reformulated on the basis of the probabilities associated with "corrected" transcriptions. In particular, if low "corrected" transcription probabilities are associated with all candidates for a particular word of the user's utterance, which suggests that the word was garbled or poorly transcribed, the word can be dropped from the query.

6.4 Further Scoring Criteria

Additional or alternative criteria for computing relevance scores for documents can be used in some embodiments. Such criteria can include, for example:

(1) probabilities associated with particular pronunciations in the phonetic index according to speaker-dependent criteria;

(2) probabilities, traditionally computed in IR systems, that take into account the frequency with which the hypothesized words occur in particular documents or in the corpus as a whole;

(3) the number of co-occurrences of two or more words of a hypothesis across several documents, which tends to indicate that these words are semantically related. If in any single document, a particular combination of two or more words of the given hypothesis appear with great frequency, this tends to indicate a coherent set of words that characterize a topic. This word-pair or word-set can be expected to appear in many documents. For example, in documents about President Kennedy, there are likely to be many occurrences of the word "Kennedy" and a smaller number of occurrences of the word "President," and these words will tend to occur in proximity to one another.

6.5 Continuous Speech

In some embodiments the invention accepts continuous speech instead of discrete-word speech. This frees the user to speak more naturally. However, it increases the computational load on the invention, because the invention must maintain multiple hypotheses about word boundary locations. For example, if the user says, "climb a mountain," the invention must determine whether the user has said "climb a mountain" or "climber mountain" or "climb amount ten." It does so by maintaining all these hypotheses in an appropriate data structure such as a word lattice, and then searching for confirming documents in corpus **41** using Boolean/proximity queries that incorporate search terms from the hypotheses. Put another way, whereas in discrete-word embodiments the invention processes a series of relatively short phone sequences, each corresponding to a single word, in continuous-speech embodiments the invention processes all at once the entire phone sequence that is the transcription of the user's utterance, hypothesizing different alternatives for words and word boundary locations and carrying out queries to try to confirm its various hypotheses. To the extent that some word boundaries can be determined without recourse to queries, for example, by noticing pauses in the user's speech or by detecting keywords, this can be incorporated in continuous-speech embodiments to reduce the computation required to disambiguate the user's speech.

7. Embodiment in a Speech Transcription System

The invention can be used in contexts other than information retrieval. FIG. **10** illustrates the invention used in a general-purpose speech recognizer **100** that serves as a "front end" speech-to-text converter for an application program **120** that accepts text input, such as a word processor. It will be appreciated that the method of the invention can be used as the sole speech-recognition technique of the speech recognizer **100**, or as one of a suite of techniques used in combination by speech recognizer **100**. Whichever approach is adopted, speech recognizer **100** benefits from a real-world knowledge base in the form of a text corpus or corpora, such as corpus **41**. When used in a non-IR context, the documents retrieved by the method of the invention are considered intermediate results that need not be displayed to the user.

The components of speech recognizer **100** are similar to, and similarly numbered to, the components of system **1** of FIG. **1**, except that display **30** or speech synthesizer **31** are not provided. Instead, output is fed to application program **120**. Application program **120** is executed by its own separate processor **110** in this embodiment; in other embodiments, it can be executed by the same processor **10** that is used in the invention.

In operation, the user's spoken input is transduced into a signal by transducer **20** and then converted into a phone sequence or a set of phone sequences by transcriber **50**. If the invention is used as the sole speech recognition technique employed by speech recognizer **100**, processing proceeds in a manner similar to that described earlier in the IR context. Specifically, hypothesis generator **60** generates hypotheses that are used as the basis for queries constructed by query constructor **70** and executed using IR subsystem **40** which includes corpus **41**. Scoring mechanism **80** determines the best hypothesis (or possibly a set of several best alternative hypotheses) and provides this as output to application program **120**. If the invention is used in conjunction with other speech recognition techniques in speech recognizer **100**, processing can proceed in the same manner; it is up to application program **120** to combine the interpretations of the user's input utterance provided by the invention and by other techniques. Alternatively, other techniques can be applied first, by routing the signal produced by transducer **20** to other modules (not shown) that incorporate other techniques; portions of the utterance that are not successfully recognized by these other modules are then fed to transcriber **50** and further processed according to the method of the invention.

8. General Applicability of Semantic Co-Occurrence Filtering

Semantic co-occurrence filtering refers to the technique, used in the present invention, in which a sequence of words supplied by a user and transcribed by machine in an error-prone fashion is disambiguated and/or verified by automatically formulating alternative hypotheses about the correct or best interpretation, gathering confirming evidence for these hypotheses by searching a text corpus for occurrences and co-occurrences of hypothesized words, and analyzing the search results to evaluate which hypothesis or hypotheses best represents the user's intended meaning. The documents retrieved in the text corpus search can also be analyzed and evaluated for relevance to the preferred hypothesis or hypotheses in applications where this is appropriate. Semantic co-occurrence filtering is a technique of broad general applicability, as will become-apparent from the embodiment of the invention that will next be described.

FIG. **11** illustrates a specific embodiment of the invention that is adaptable to a range of input sources, transcription techniques, hypothesis generation techniques, information retrieval techniques, and analysis techniques. The embodiment comprises a processor running appropriate software and coupled to a text corpus, an input transducer, and an output facility such as an output channel, stream, or device.

In brief, the user supplies as input a question, such as a spoken utterance, handwritten phrase, optically scanned document excerpt, or other sequence of words. A transducer converts the input question into a signal suitable for processing by computer. A processor (computer) transcribes the signal into at least one string that comprises a sequence of symbols such as letters, phones, orthographically represented words, or phonetically represented words. Typically the processor's transcription is error-prone, so that the string contains some symbols that do not necessarily correctly represent the user's intended words. Thus there is a need to disambiguate the transcription of the user's question—that is, to determine, insofar as possible, what words the user really intended as input.

The processor generates a set of alternative hypotheses that represent different possible interpretations of what the user meant to say in the question. The hypotheses typically comprise sequences of orthographically represented words; if the text corpus is represented phonetically instead of orthographically, the hypotheses can comprise sequences of phonetically represented words. Alternative hypotheses can be generated in a variety of ways, such as modifying portions of the transcription based on a statistical model of common transcriber errors, consulting a thesaurus, consulting a table of related words, and (if the input question is not delineated into discrete words) positing alternative choices for word boundaries. The processor uses the hypotheses to construct one or more information retrieval queries that are executed over a text corpus. The queries seek hypothesized words in the corpus, and in particular seek co-occurrences of two or more words of a given hypothesis. When two or more words of a hypothesis appear in proximity to one another in a number of documents of the corpus, this is taken as an indication that the words are semantically related and therefore are more likely to represent a correct interpretation of the user's question than semantically unrelated words would. The processor analyzes the query results, which comprise sets of documents returned from the corpus and matched search terms within those documents, to evaluate which hypothesis or hypotheses are most likely to represent a correct interpretation of the user's question, and, in some cases, which documents are most relevant to the preferred hypotheses. Feedback is provided from the analysis component to the query component, so that based on its analysis of the results of a query or queries, the processor can construct reformulated (modified) queries, cause these to be executed, and analyze their results, construct further queries, and so forth until the processor is satisfied with the results or otherwise determines that further queries are not worthwhile. The processor can then output the preferred hypothesis or hypotheses and, if appropriate, the relevant documents for each. Output can be made to a visual display, a speech synthesizer, a storage device, etc., and can also be made to an applications program such as a word processor, desktop publishing program, document analysis program, or other program that accepts textual input.

Referring now more specifically to the elements of FIG. **11**, processor **200** executes software **205** and is coupled to input transducer **220**, output channel **230**, and corpus **241**. Transcriber **250**, hypothesis generator **260**, query/IR mechanism **270**, and analyzer/evaluator **280** are typically implemented as software modules that are part of software **205** and are executed by processor **200**.

In operation, transducer **220** accepts an input question **301** and converts it into a signal **320**. Input question **301** comprises a sequence of words, typically represented in a form convenient for the user but not easily understood by computer. For example, input question **301** can be a spoken utterance, in which case transducer **220** comprises audio signal processing equipment that converts the spoken utterance to signal **320**. Input question **301** can be a phrase handwritten with a pen or stylus, in which case transducer **220** comprises a digitizing tablet or input-sensitive display screen as is typical of pen-based computers. Input question **301** can be a document or document excerpt to be optically scanned, in which case transducer **220** comprises an optical scanner with optical character recognition capability. Input question **301** can even be a typewritten character sequence, as when the user is a poor typist, in which case transducer **220** comprises a conventional computer keyboard. Other possible forms of input question **301** and transducer **220** (and indeed, of all elements of the system of FIG. **11**) will be apparent to those of skill in the art.

Input question **301** can comprise a sequence of discrete words or a single entity that represents a continuous stream of words. Thus, for example, if input question **301** is handwritten, the user can leave distinct spaces between words to indicate word boundaries, or the handwritten input can be treated as a unit, with the invention determining where word boundaries lie. Input question **301** can in some cases consist of a single word, but preferably comprises two or more words, so that a co-occurrence search can be performed. (It is to be understood that although the user-supplied input **301** is here termed a question, it can be in a form other than a question; for example, it can be a declarative statement, a phrase that is a sentence fragment, or even an entire paragraph or page of text.)

Some or all of the words of input question **301** can be keywords, such as command words or common function words. Keywords are not used in hypotheses or queries, but instead control the actions of the subsequent modules. More particularly, common function words can be ignored on input; they are not incorporated into hypotheses or queries. Command words are also not incorporated into hypotheses or queries, but they can affect the functioning of transcriber **250**, hypothesis generator **260**, query/IR mechanism **270**, or analyzer/evaluator **280**.

Transducer **220** provides signal **320** to transcriber **250**. Transcriber **250** converts signal **320** to a string **350** that represents a transcription of the input question **301**. Typically, transcriber **250** is error-prone, and string **350** does not necessarily represent a correct transcription of what the user intended to say in question **301**.

String **350** is a sequence of symbols. If input question **301** comprises discrete words, the symbols of string **350** can be, for example, orthographic (textual) representations of words or phonetic representations of words. If input question **301** comprises a continuous stream of words, the symbols of string **350** can be, for example, letters or phones. Transcriber **250** can optionally provide alternative transcriptions, represented as additional strings in the same format as string **350**.

Transcriber **250** provides string **350** and any additional strings representing alternative transcriptions to hypothesis generator **260**. Hypothesis generator **260** converts string **350** and any alternatives to a set of hypotheses **360**. Typically there is more than one hypothesis in this set. Each hypothesis comprises a sequence of words, typically represented by their orthographic spellings. (It is also possible for the words to be represented phonetically; this will be the case if corpus **241** comprises documents in which words are represented phonetically rather than as written text.) Each hypothesis comprises a sequence of candidates, which are possible interpretations of each of the user's words. If input question **301** is presented as discrete words, then word boundaries are the same for all hypotheses, so that different hypotheses can be generated simply by interchanging candidates. If input question **301** is presented as a continuous stream, then word boundaries can differ for different hypotheses.

In producing the hypotheses **360**, hypothesis generator **260** can make use of any alternative transcriptions produced by transcriber **250** and can also generate additional possible interpretations of the words of the user's question **301**. Conversion of string **350** and any alternatives to hypotheses **360** can be done according to a wide variety of techniques, used alone or in combination. For example, if string **350** is a sequence of phones or a sequence of phonetically represented words, phonetic substitutions, insertions, and deletions according to a model of transcriber performance and errors, can be used in conjunction with phonetic index matching as described earlier with reference to FIGS. **2**, **4**, and **5**. Additionally, a table of homonyms can be incorporated. If string **350** is a sequence of orthographic words, a spelling dictionary can be used. Additionally, a thesaurus, or a table of related words (e.g., for spoken input, a table of words with related sounds), can be used to generate more search terms. Although such related words do not appear explicitly in question **301**, their presence in corpus **241** in conjunction with other words of question **301** and related words will tend to confirm particular hypothesized interpretations of question **301**.

Hypothesis generator **260** provides hypotheses **360** to query/IR mechanism **270**. Query/IR mechanism **270** converts the hypotheses **360** to one or more information retrieval queries **370**. Any of a number of information retrieval query techniques can be used, such as Boolean queries with proximity and order constraints or extended Boolean queries. Whatever technique is used, the queries **370** are in a format that can be searched by processor **200** (or a separate IR processor that communicates with processor **200**) using corpus **241**. Typically, a query searches for the co-occurrence of two or more candidate words from a hypothesis or hypotheses. If a hypothesis consists of a single word, then a query can search for that word alone.

If input question **301** is provided in discrete-word form, so that the candidates for any given word of input question **301** all share the same word boundaries, then many hypotheses can be combined to construct a single query. For example, if question **301** is the handwritten phrase "president kennedy," and it is known that this phrase contains two words (as indicated by the blank space between them), then if Boolean/proximity queries are used, a query such as

(k {president present prescient . . . }
{kennedy canada . . . })

can be constructed. This query seeks the co-occurrence within k words of any candidate for the first word of question **301** and any candidate for the second word of question **301**. Each search term of the query is a Boolean OR of all candidates for a particular word of the query.

In contrast, if input question **301** is provided as a continuous stream of words, so that hypothesis generator **260** can entertain multiple possibilities for word boundaries, then it may not be possible to combine hypotheses in a single query. For example, if question **301** is the handwritten phrase "presidentkennedy," with no blank spaces, then if Boolean/proximity queries are used, queries such as

(k {preside precede} {other either} {unity remedy})

and

(k {preside precede} {other either}
{unity remedy})
and
(k {president present prescient . . . }
{kennedy canada . . . })

can be executed independently of one another. Alternatively, several of these queries can be represented together, for example as regular expressions.

Queries **370** are executed by query/IR mechanism **370** over corpus **241**. The actual search of corpus **241** can be done by processor **200** or by a separate subsystem comprising one or more separate IR processors. Results **380** are returned as a consequence of executing queries **370**.

Query/IR mechanism **370** provides results **380** to analyzer/evaluator **280**. Analyzer/evaluator **280** analyzes results **380** to determine which of the hypothesis or hypotheses **360** is most likely to represent a correct interpretation of question **301**. Additionally, if appropriate to the particular application of the invention, analyzer/evaluator **280** determines which retrieved documents are most relevant to the preferred hypothesis or hypotheses. Analyzer/evaluator **280** can provide feedback to query/IR mechanism **270**, to request that reformulated queries be executed, for example to obtain a broader or narrower range of results.

Analyzer/evaluator **280** can employ a variety of analysis strategies to implement semantic co-occurrence filtering. Some examples are the heuristic strategies described earlier with reference to FIGS. **2** and **7**. Strategies can also, for example, make use of synonyms generated by a thesaurus incorporated into hypothesis generator **260**. Whatever strategies are used, they have in common that the co-occurrence of two or more candidate words of a given hypothesis—that is, the appearance of two candidate words of a hypothesis in proximity to one another in one or more documents of corpus **241**—is taken as evidence tending to confirm both the candidate words and the hypothesis, and can further be taken as evidence that the document in which the candidate words appear is relevant to input question **301**.

Analyzer/evaluator **280** provides the hypothesis or hypotheses most likely to correctly interpret question **301** and, if appropriate, query results relevant to this hypothesis or hypotheses, as an interpretation **400** that is output via output channel **230**. The hypotheses can be represented, for example, as ASCII text. The query results can include, for example, documents in excerpt or full-text form, document titles, and matched search terms.

Output channel **230** can send interpretation **400** to be displayed using a visual display **231**, spoken to the user by a speech synthesizer **232**, or stored using a storage device such as a hard disk **233**. Additionally or instead, output channel **320** can send interpretation **400** to an applications program **250** to be used as input thereby.

If the appropriate command keywords are supported, the user can provide relevance feedback based on displayed or speech-synthesized output. In this case, interpretation **400** can be used as a "mini-corpus" to facilitate the understanding of the inputs that the user provides as relevance feedback.

9. Operational Example

The following example illustrates an output trace from a testbed embodiment of the invention that simulates a speech-recognizing IR application. An on-line version of Grolier's Academic American Encyclopedia (*The Academic American Encyclopedia,* Danbury, Conn.: Grolier Electronic Publishing, 1990) serves as the text corpus for this embodiment. The on-line encyclopedia contains approximately 27,000 articles, which are accessed via the Text Database (D. R. Cutting, J. Pedersen, and P. -K. Halvorsen, "An object-oriented architecture for text retrieval," in *Conference Proceedings of RIAO '91, Intelligent Text and Image Handling,* Barcelona, Spain, April 1991, pages 285–298), which is a flexible platform for the development of retrieval system prototypes and is structured so that additional functional components can be easily integrated. The phonetic index is constructed by merging the contents of a 23,000 word phonetic dictionary with Grolier's encyclopedia, to produce a phonetic index that covers 16,000 of the total 100,000 words in the Grolier word index.

The hardware for the testbed embodiment comprises a Sun SparcStation 10 workstation. The testbed embodiment of the invention has no transducer or transcriber components, and the user does not actually speak the words of the input question. Instead, the embodiment accepts input in the form of a simulated errorful phonetic transcription of the user's words, which is created manually from a confusion matrix. The simulated transcription assumes discrete-word speech.

The testbed embodiment of the invention does not do query reformulation. The initial query is the only query; no additional or modified queries are constructed or executed in this embodiment.

For the example below, the user's utterance consists of the spoken words "first", "atomic" and "bomb." For each spoken word, the words in the phonetic index that match a simulated errorful transcription are shown. The form of an IR query is then presented, followed by a list of matching documents, in alphabetical title order. The entries marked with triple asterisks '***' indicate extra documents that are not present if an IR query is made from a perfect phonetic transcription: (5 first atomic bomb).

Phonetic index entries that match a noisy version of the word "first": (total of 19 entries)

fez:	(F EH Z)	fierce:	(F IH2 R S)
fife:	(F AY F)	first:	(F ER2 S SIL2-T)
firth:	(F ER2 TH)	fish:	(F IH2 SH2)
fist:	(F IH2 S SIL2-T)	fizz:	(F IH2 Z)
frizz:	(F R IH2 Z)	frost:	(F R AO2 S SIL2-T)
froth:	(F R AO2 TH)	furze:	(F ER2 Z)
fuss:	(F AH2 S)	fuzz:	(F AH2 Z)
phiz:	(F IH2 Z)	theft:	(TH EH F SIL2-T)
thirst:	(TH ER2 S SIL2-T)	thrice:	(TH RAY S)
thrift:	(TH R IH2 F SIL2-T)		

Phonetic index entries that match a noisy version of the word "atomic": (total of 41 entries)

acid:	(AE S IH2 SIL2-D)	adage:	(AE SIL2-D IH2 JH)
adduce:	(AH2 SIL2-D UW2 S)	adjudge:	(AH2 JH AH2 JH)
admit:	(AH2 SIL2-D M2 IH2 SIL2-T)	apace:	(AH2 SIL2-P EY S)
apart:	(AH2 SIL2-P AO2 R SIL2-T)	apeak:	(AH2 SIL2-P IY SIL2-K)
apiece:	(AH2 SIL2-P IY S)	aport:	(AH2 SIL2-P OW R SIL2-T)
appanage:	(AE SIL2-P AH2 N2 IH2 JH)	apse:	(AE SIL2-P S)
apt:	(AE SIL2-P SIL2-T)	aside:	(AH2 S AY SIL2-D)
ask:	(AE S SIL2-K)	asp:	(AE S SIL2-P)
assert:	(AH2 S ER2 SIL2-T)	assess:	(AH2 S EH S)
asset:	(AE S EH SIL2-T)	assort:	(AH2 S AO2 R SIL2-T)
assuage:	(AH2 S W EY JH)	atomic:	(AH2 SIL2-T AO2 M2 IH2 SIL2-K)
atop:	(AH2 SIL2-T AO2 SIL2-P)	attack:	(AH2 SIL2-T AE SIL2-K)
attic:	(AE SIL2-T IH2 SIL2-K)	audit:	(AO2 SIL2-D IH2 SIL2-T)
automate:	(AO2 SIL2-T AH2 M2 EY SIL2-T)	edit:	(EH SIL2-D IH2 SIL2-T)
educe:	(IH2 SIL2-D UW2 S)	epic:	(EH SIL2-P IH2 SIL2-K)

-continued

epoch:	(EH SIL2-P AH2 SIL2-K)	erst:	(ER2 S SIL2-T)
essence:	(EH S N2 S)	isthmus:	(IH2 S M2 AH2 S)
its	on stop list--ignored	opaque:	(OW SIL2-P EY SIL2-K)
opt:	(AO2 SIL2-P SIL2-T)	opus:	(OW SIL2-P AH2 S)
oust:	(AW S SIL2-T)	outward:	(AW SIL2-T W ER2 SIL2-D)
outwork:	(AW SIL2-T W ER2 SIL2-K)	upward:	(AH2 SIL2-P W ER2 SIL2-D)

Phonetic index entries that match a noisy version of the word "bomb":
(total of 44 entries)

a:	on stop list--ignored		
awe:	(AO2)	awn:	(AO2 N2)
ay:	(AY)	ayah:	(AY AH2)
aye:	(AY)	balm:	(SIL2-B AO2 M2)
beau:	(SIL2-B OW)	boa:	(SIL2-B OW AH2)
bob:	(SIL2-B AO2 SIL2-B)	bomb:	(SIL2-B AO2 M2)
bone:	(SIL2-B OW N2)	bough:	(SIL2-B AW)
bow:	(SIL2-B OW)	bum:	(SIL2-B AH2 M2)
bump:	(SIL2-B AH2 M2 SIL2-P)	bun:	(SIL2-BB AH2 N2)
bung:	(SIL2-B AH2 NG2)	buy:	(SIL2-B AY)
by:	on stop list--ignored	bye:	(SIL2-B AY)
eye:	(AY)	i:	(AY)
o:	(OW)	ohm:	(OW M2)
on:	on stop list--ignored	ope:	(OW SIL2-P)
owe:	(OW)	own:	(OW N2)
pa:	(SIL2-P AO2)	palm:	(SIL2-P AO2 M2)
pas:	(SIL2-P AO2)	paw:	(SIL2-P AO2)
pawn:	(SIL2-P AO2 N2)	pi:	(SIL2-P AY)
pie:	(SIL2-P AY)	pine:	(SIL2-P AY N2)
pipe:	(SIL2-P AY SIL2-P)	pomp:	(SIL2-P AO2 M2 SIL2-P)
pone:	(SIL2-P OW N2)	pop:	(SIL2-P AO2 SIL2-P)
pope:	(SIL2-P OW SIL2-P)	pub:	(SIL2-P AH2 SIL2-B)
pump:	(SIL2-P AH2 M2 SIL2-P)	pun:	(SIL2-P AH2 N2)
pup:	(SIL2-P AH2 SIL2-P)	up:	(AH2 SIL2-P)

IR Query: (5 {fez fierce . . . first . . . thrice thrift}
{acid adage . . . atomic . . . outwork upward}
{awe awn ay . . . bomb . . . pun pup up})

DISCUSSION OF THE SOFTWARE

A current embodiment of the invention is implemented in software on a digital computer. The appendix (unpublished work, Copyright ©1993 Xerox Corporation) provides source code for a software program for implementation of this embodiment. The source code is written in the Lisp language (Franz Allegro Common Lisp version 4.1), well known to those of skill in the art. The software program has been demonstrated on a Sun SparcStation 10 workstation, although it will be apparent to those of skill in the art that a wide variety of programming languages and hardware configurations can readily be used based on this disclosure without departing from the scope of the invention.

The Appendix includes two files. The first file includes source code for reading a phonetic index file, for query construction, and for scoring. The second file includes source code for hypothesis generation.

The two source code files provided in the Appendix are designed to be used in conjunction with certain additional software modules. An on-line version of Grolier's Academic American Encyclopedia (*The Academic American Encyclopedia,* Danbury, Conn.: Grolier Electronic Publishing, 1990), serves as the text corpus. Encyclopedia articles are accessed via the Text Database (D. R. Cutting, J. Pedersen, and P. -K. Halvorsen, "An object-oriented architecture for text retrieval," in *Conference Proceedings of RIAO* '91, *Intelligent Text and Image Handling,* Barcelona, Spain, April 1991, pages 285–298), a flexible platform for the development of retrieval system prototypes. The phonetic index is constructed by merging the contents of a phonetic dictionary (the Moby Pronunciator II, distributed by Grady Ward, Arcata, Calif.) with Grolier's encyclopedia. In the phonetic index, orthographic spellings are stored in a vector, and phone sequences are represented by a finite-state network that provides an index into the vector. This technique is substantially similar to the minimal perfect hashing technique described in Lucchesi, Claudio L. and Tomasz Kowaltowski, "Applications of Finite Automata Representing Large Vocabularies," *Software—Practice and Experience,* vol. 23(1), pp. 15–30, January 1993 (see especially pp. 26–27). Discrete-word speech can be input using a Sennheiser HMD414 headset microphone and a Rane MS-1 preamplifier, with signal processing performed in software by the SparcStation. Input speech is transcribed into a phone sequence using hidden Markov model methods, a discussion of which can be found in Lawrence R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE,* vol. 77, no. 2, Feb. 1989, pp. 257–285. An alignment program is used to compare the phonetic transcription with phone sequences of the phonetic index. Alternatively, a simulated errorful phonetic transcription of the user's words can be created manually from a confusion matrix and provided in lieu of actual speech input, transcription, and alignment.

CONCLUSION

The present invention provides a system and method for disambiguating words in the output of a transcriber based on the co-occurrence of words in a text corpus. The invention uses the likelihood of co-occurrence of words in the corpus to filter or constrain the candidate hypotheses. The invention takes advantage of the observations that the user's input questions most often comprise semantically related words, that documents of the corpus do likewise, that words that co-occur in a user question are likely to co-occur with some regularity in the documents of the corpus, and that words that are the product of transcription errors tend not to be semantically related and therefore tend not to co-occur in documents of the corpus. Boolean IR queries are used to disambiguate the words of the user's question and, in embodiments where information retrieval is the primary application, simultaneously to retrieve relevant documents from the corpus.

The invention has been explained with reference to specific embodiments. Other embodiments are contemplated without departing from the spirit and scope of the invention. For example, in some embodiments, syntactic or semantic analysis (e.g., noun-phrase recognition) can be used in analyzing documents returned in response to queries (e.g., IR query matches to words of a given hypothesis can be assigned relatively higher scores if the matched words appear in the retrieved document in a noun phrase structure that corresponds to a noun phrase structure in the hypothesis). As another example, in other embodiments hypotheses can be ranked initially according to relatively inaccurate but computationally efficient ("coarse") criteria, and then a subset of these ranked further according to relatively accurate but computationally inefficient ("fine") criteria. It is therefore not intended that this invention be limited, except as indicated by the appended claims.

What is claimed is:

1. An automated transcription disambiguation method comprising the steps of:

providing an input question having first and second words to a processor in a form subject to misinterpretation by the processor;

generating a plurality of hypotheses with the processor, the hypotheses including alternative interpretations of at least one of the first and second words due to possible misinterpretations of the input question by the processor;

producing with the processor an initial evaluation of the hypotheses;

gathering confirming evidence for the hypotheses by searching with the processor in a text corpus for co-occurrences of hypothesized first and second words for the hypotheses;

automatically and explicitly selecting with the processor from among the plurality of hypotheses a preferred hypothesis as to both of the first and second words based at least in part on the initial evaluation and at least in part on the gathered confirming evidence; and

outputting a transcription result from the processor, the transcription result representing the selected preferred hypothesis.

2. In the operation of a system comprising a processor, an input transducer, an output facility, and a corpus comprising at least one document comprising words represented in a first form, a method for transcribing an input question by transforming the input question from a sequence of words represented in a second form, subject to misinterpretation by the processor, into a sequence of words represented in the first form, the method comprising the steps of:

accepting the input question into the system, the question comprising a sequence of words represented in the second form;

converting the input question into a signal with the input transducer;

converting the signal into a sequence of symbols with the processor;

generating a set of hypotheses from the sequence of symbols with the processor, the hypotheses of the set comprising sequences of words represented in the first form, the set of hypotheses including alternative interpretations of at least one of the words to account for possible misinterpretation of the input question;

producing with the processor an initial evaluation of the hypotheses;

automatically constructing a query from hypotheses of the set with the processor;

executing the constructed query by searching with the processor in the corpus for co-occurrences of hypothesized words for the hypotheses;

analyzing the co-occurrences and the initial evaluation with the processor to produce a revised evaluation of the hypotheses of the set;

automatically and explicitly selecting a preferred hypothesis from the set with the processor responsively to the revised evaluation, the preferred hypothesis comprising a preferred sequence of words in the first form and thus a preferred transcription of the sequence of words of the input question; and

outputting the preferred hypothesis with the output facility.

3. The method of claim 2 wherein:

the corpus includes a plurality of documents;

the step of executing the constructed query includes retrieving documents containing the co-occurrences;

the step of automatically and explicitly selecting the preferred hypothesis further comprises selecting with the processor a preferred set of documents, the preferred set of documents comprising a subset of the retrieved documents that are relevant to the preferred hypothesis, and

the step of outputting the preferred hypothesis further comprises outputting with the output facility at least a portion of a document belonging to the preferred set of documents.

4. The method of claim 3 further comprising the steps, performed after the step of outputting at least a portion of a document belonging to the preferred set of documents, of:

accepting a relevance feedback input into the system, the relevance feedback input comprising a sequence of words represented in the second form, the sequence of words including a relevance feedback keyword and a word that occurs in the outputted document;

converting the relevance feedback input into an additional query with the processor; and

executing the additional query with the processor to retrieve an additional document from the corpus.

5. The method of claim 2 wherein:

the step of automatically and explicitly selecting the preferred hypothesis further comprises selecting a plurality of preferred hypotheses with the processor; and

the step of outputting the preferred hypothesis further comprises outputting the selected plurality of preferred hypotheses with the output facility.

6. The method of claim 2 wherein:

the step of accepting an input question further comprises accepting information into the system, the information concerning the locations of word boundaries between words of the question; and

the step of converting the signal into a sequence of symbols further comprises specifying subsequences of the sequence of symbols with the processor according to the locations of word boundaries thus accepted.

7. The method of claim **2** wherein the step of generating a set of hypotheses from the sequence of symbols further comprises generating hypothesized locations of word boundaries with the processor.

8. The method of claim **2** wherein the step of converting the input question into a signal comprises converting spoken input into an audio signal with an audio transducer.

9. The method of claim **2** wherein the step of constructing a query from hypotheses of the set comprises constructing a Boolean query with a proximity constraint.

10. The method of claim **2** wherein the step of generating a set of hypotheses from the sequence of symbols comprises detecting a keyword with the processor to prevent inclusion of the keyword in hypotheses of the set.

11. The method of claim **10** wherein the step of constructing a query from hypotheses of the set comprises constructing a query from hypotheses of the set with the processor, the query being responsive to the detected keyword.

12. The method of claim **2** wherein the step of constructing a query from hypotheses of the set comprises constructing an initial query with the processor and prior to the outputting step automatically constructing a reformulated query with the processor, the reformulated query comprising a reformulation of the initial query.

13. The method of claim **2** wherein the step of outputting the preferred hypothesis comprises visually displaying the preferred hypothesis.

14. The method of claim **2** wherein the step of outputting the preferred hypothesis comprises synthesizing a spoken form of the preferred hypothesis.

15. The method of claim **2** wherein the step of outputting the preferred hypothesis comprises providing the preferred hypothesis to an applications program.

16. The method of claim **15** further comprising the step of accepting the preferred hypothesis into the applications program as textual input to the applications program.

17. The method of claim **2** wherein the step of producing an initial evaluation comprises determining an initial evaluation measurement for each hypothesis.

18. In a system comprising a processor, a method for processing an input utterance comprising speech, the method comprising the steps of:

accepting the input utterance into the system;

producing a phonetic transcription of the input utterance with the processor;

responsively to the phonetic transcription, generating with the processor a set of hypotheses, the hypotheses of the set being hypotheses as to a first word contained in the input utterance and further as to a second word contained in the input utterance, the set of hypotheses including alternative interpretations of at least one of the words to account for the error-prone nature of speech analysis;

determining with the processor an initial evaluation measurement for each hypothesis;

automatically constructing an information retrieval query with the processor, the query comprising the set of hypotheses and a proximity constraint;

executing the constructed query in conjunction with an information retrieval subsystem comprising a text corpus; and

responsively to the results of the executed query with respect to each hypothesis of the set of hypotheses, and taking into consideration the initial evaluation measurements of the hypotheses, automatically and explicitly selecting with the processor from among the hypotheses of the set a preferred hypothesis, the preferred hypothesis including the first and second words.

19. The method of claim **18** wherein the step of generating a set of hypotheses comprises matching portions of the phonetic transcription against a phonetic index with the processor.

20. In a system comprising a processor, an error-prone input facility, and an information retrieval subsystem, said information retrieval subsystem comprising a natural-language text corpus, a method for accessing documents of the corpus, the method comprising the steps of:

transcribing a question with the error-prone input facility and the processor, the question comprising a sequence of words;

selecting a subset of words of the sequence with the processor;

forming with the processor a plurality of hypotheses about the selected subset of words, the hypotheses of the plurality representing possible alternative transcriptions of the question to account for the error-prone nature of the input facility;

producing with the processor an initial evaluation of the hypotheses;

automatically constructing a co-occurrence query with the processor, the co-occurrence query being based on hypotheses of the plurality;

executing the co-occurrence query in conjunction with the information retrieval subsystem to retrieve a set of documents;

analyzing the initial evaluation and documents of the retrieved set with the processor to produce a revised evaluation of the hypotheses;

responsively to the revised evaluation, automatically and explicitly selecting with the processor a preferred hypothesis representing a preferred transcription of the sequence of words of the question;

evaluating documents of the retrieved set with the processor with respect to the selected hypothesis to determine a relevant document; and

outputting from the system the relevant document thus determined.

21. An automated system for producing a preferred transcription of a question presented in a form prone to erroneous transcription, comprising:

a processor;

an input transducer, coupled to the processor, for accepting an input question and producing a signal therefrom;

converter means, coupled to the input transducer, for converting the signal to a string comprising a sequence of symbols;

hypothesis generation means, coupled to the converter means, for developing a set of hypotheses from the string, each hypothesis of the set comprising a sequence of word representations, the set of hypotheses representing a set of possible alternative transcriptions of the input question to account for the likelihood of erroneous transcription;

initial scoring means, coupled to the hypothesis generation means, for determining an initial score for each hypothesis;

query construction means, coupled to the hypothesis generation means, for automatically constructing at least one information retrieval query using hypotheses of the set;

a corpus comprising documents, each document comprising word representations;

query execution means, coupled to the query construction means and to the corpus, for retrieving from the corpus documents responsive to said at least one query;

analysis means, coupled to the query execution means, for generating an analysis of the retrieved documents and evaluating the hypotheses of the set based on the initial scores and the analysis to determine a preferred hypothesis from among the hypotheses of the set, the preferred hypothesis representing a preferred transcription of the sequence of words of the input question; and

output means, coupled to the analysis means, for outputting the preferred hypothesis.

22. A speech processing apparatus comprising:

input means for transducing a spoken utterance into an audio signal;

means for converting the audio signal into a sequence of phones;

means for analyzing the sequence of phones to generate a plurality of hypotheses comprising sequences of words, the hypotheses representing possible alternative transcriptions of the spoken utterance to account for the error-prone nature of speech analysis;

means for determining an initial evaluation measurement for each hypothesis;

means for automatically constructing a query using the hypotheses of the plurality;

information retrieval means, coupled to a corpus of documents and to the constructing means, for retrieving documents of the corpus relevant to the constructed query;

means for automatically and explicitly ranking the hypotheses of the plurality according to confirming evidence found in the retrieved documents and further according to the initial evaluation measurements previously determined; and

means for outputting a subset of the hypotheses thus ranked, each hypothesis of the subset comprising a sequence of words representing a possible transcription of the spoken utterance.

* * * * *

US006006227A

United States Patent [19]

Freeman et al.

[11] **Patent Number: 6,006,227**

[45] **Date of Patent: *Dec. 21, 1999**

[54] **DOCUMENT STREAM OPERATING SYSTEM**

[75] Inventors: **Eric Freeman**, Branford; **David H. Gelernter**, Woodbridge, both of Conn.

[73] Assignee: **Yale University**, New Haven, Conn.

[*] Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

[21] Appl. No.: **08/673,255**

[22] Filed: **Jun. 28, 1996**

[51] **Int. Cl.**^{6} **G06F 17/30**

[52] **U.S. Cl.** **707/7**; 707/2; 707/102

[58] **Field of Search** 395/611, 616; 707/100, 102, 200, 2, 7

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,063,495	11/1991	MacPhail	395/761
5,140,676	8/1992	Langelaan	395/777
5,159,669	10/1992	Trigg et al.	395/357
5,402,526	3/1995	Bauman et al.	706/49
5,448,729	9/1995	Murdock	395/615
5,530,859	6/1996	Tobias, II et al.	395/551
5,535,063	7/1996	Lamming	360/4
5,625,818	4/1997	Zarmer et al.	395/615
5,649,182	7/1997	Reitz	707/7
5,835,129	11/1998	Kumar	348/15
5,890,177	3/1999	Moody et al.	707/511

OTHER PUBLICATIONS

Cowart, Mastering Windows 3.1, 1992, chapter 12, pp. 396–417, Jan. 1, 1992.

Gelernter, The Cyber–Road Not Taken, The Washington Post, Apr. 3, 1994, 9 pages.

Freeman, Lifestreams Project Home Page, 3 pages, 1994–1996, Jan. 1, 1994.

Freeman, Lifestreams for the Newton, Steve Mann's Developers Corner, Oct. 31, 1995, 5 pages.

Steinberg, Lifestreams, Wired 5.02, Feb. 28, 1997, 11 pages.

Freeman et al, Lifestreams: Organizing your Electronic Life, AAAI Fall Symposium, AI Applications in Knowledge Navigation and Retrieval, Nov. 30, 1995, Cambridge MA, 6 pages.

Getting Results with Microsoft Outlook, copyright Microsoft Corporation 1995–1996, pp. 28–29, Jan. 1, 1995.

Nelson, Theodor H., The Right Way to Think About Software Design, In The Art of Human–Computer Interface Design, Brenda Laurel, (Ed.), (1990): pp. 235–243.

Landsdale, M., The Psychology of Personal Information Management, Applied Ergonomics, (Mar. 1988): pp. 55–66.

Malone, Thomas W., How Do People Organize Their Desks? Implications For The Design Of Office Information Systems., ACM Transactions On Office Systems, vol. 1, No. 1 (Jan. 1983): pp. 99–112.

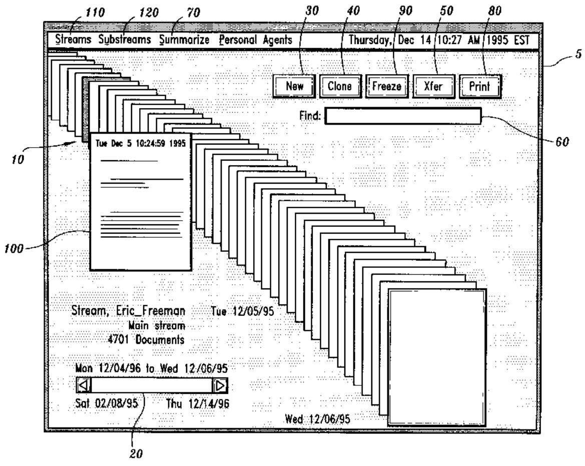
Primary Examiner—Wayne Amsbury

Attorney, Agent, or Firm—Cooper & Dunham LLP

[57] **ABSTRACT**

A document stream operating system and method is disclosed in which: (1) documents are stored in one or more chronologically ordered streams; (2) the location and nature of file storage is transparent to the user; (3) information is organized as needed instead of at the time the document is created; (4) sophisticated logic is provided for summarizing a large group of related documents at the time a user wants a concise overview; and (5) archiving is automatic. The documents can include text, pictures, animations, software programs or any other type of data.

33 Claims, 6 Drawing Sheets



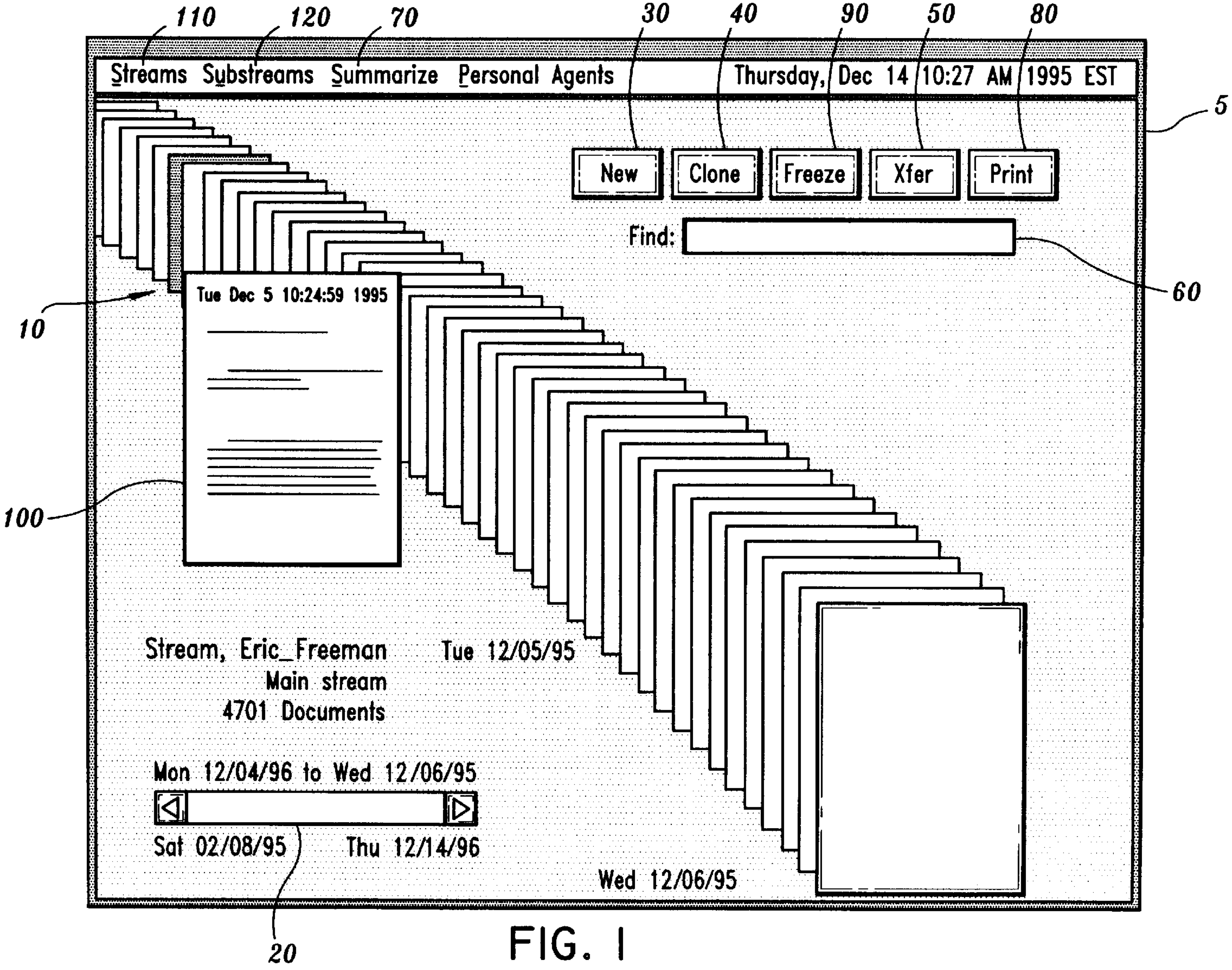

110
120
70
30
40
90
50
80
Streams
Substreams
Summarize
Personal Agents
Thursday, Dec 14 10:27 AM 1995 EST
5
New
Clone
Freeze
Xfer
Print
Find:
60
10
Tue Dec 5 10:24:59 1995
100
Stream, Eric_Freeman
Main stream
4701 Documents
Tue 12/05/95
Mon 12/04/96 to Wed 12/06/95
Sat 02/08/95
Thu 12/14/96
Wed 12/06/95
20

FIG. 1

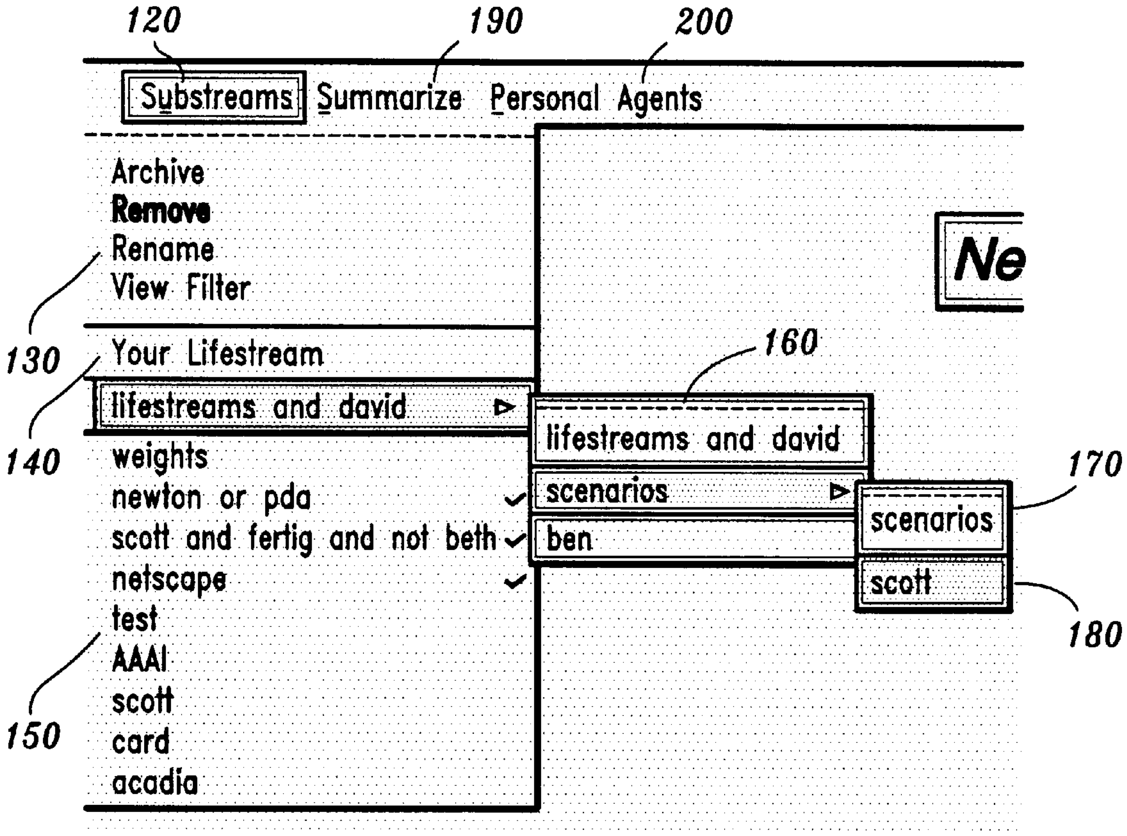

120
190
200
Substreams
Summarize
Personal Agents
Archive
Remove
Rename
View Filter
Ne
130
Your Lifestream
140
lifestreams and david ▹
weights
newton or pda ✓
scott and fertig and not beth ✓
netscape ✓
test
AAAI
scott
card
acadia
150
160
lifestreams and david
scenarios ▹
ben
170
scenarios
scott
180

FIG. 2

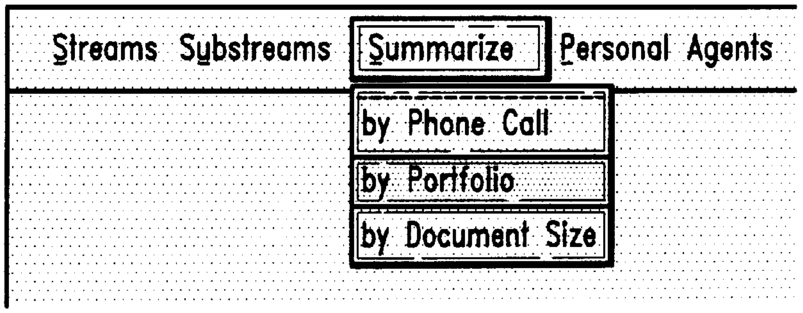

Streams
Substreams
Summarize
Personal Agents
by Phone Call
by Portfolio
by Document Size

FIG. 3

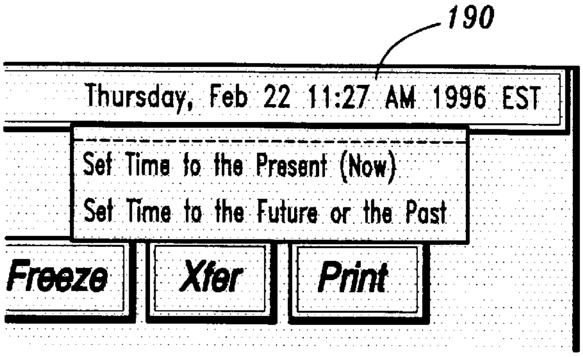
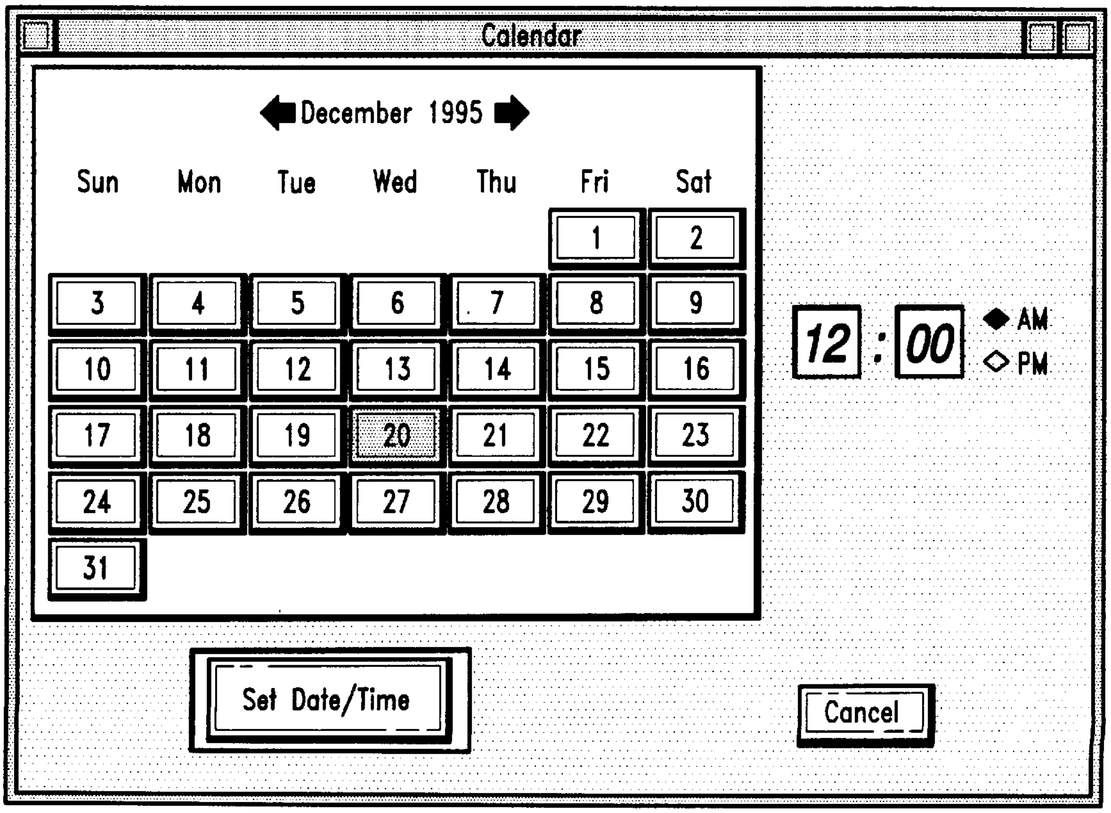

190
Thursday, Feb 22 11:27 AM 1996 EST
Set Time to the Present (Now)
Set Time to the Future or the Past
Freeze
Xfer
Print

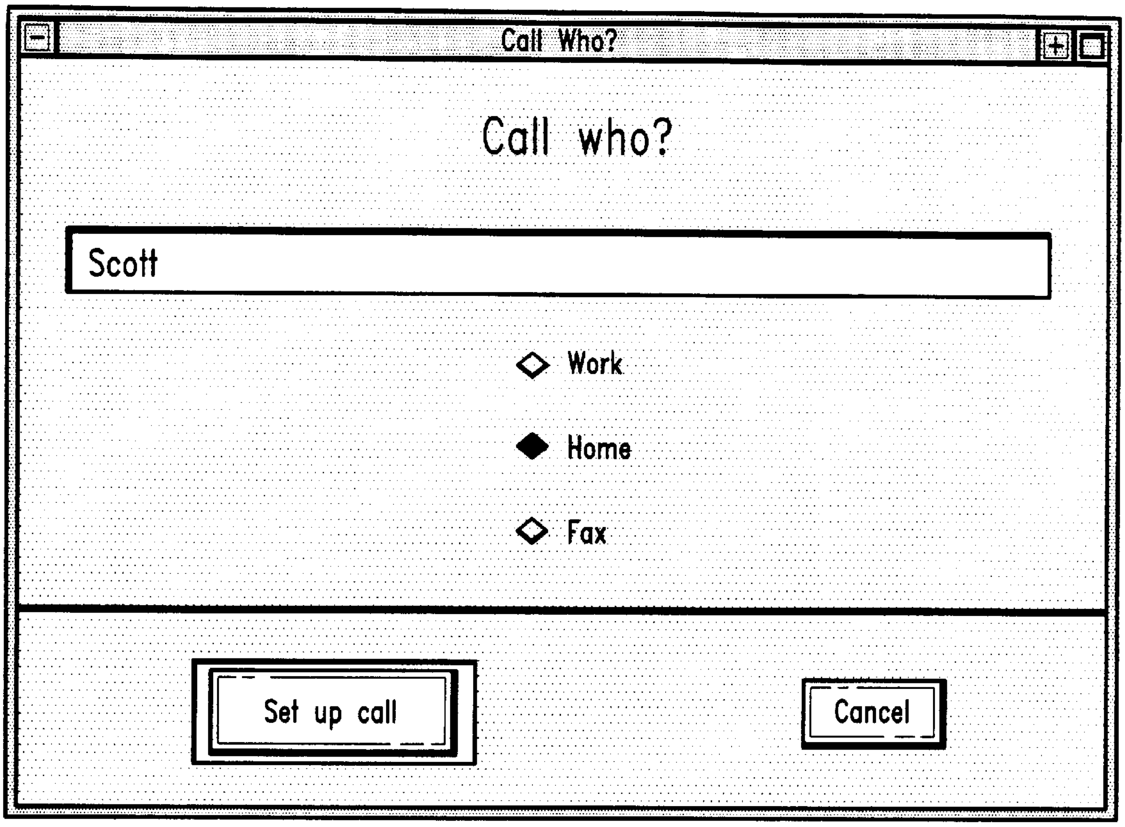
FIG. 4



Sun	Mon	Tue	Wed	Thu	Fri	Sat
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

12 : 00 ◆ AM ◇ PM

FIG. 5



Call who?

Scott

◇ Work

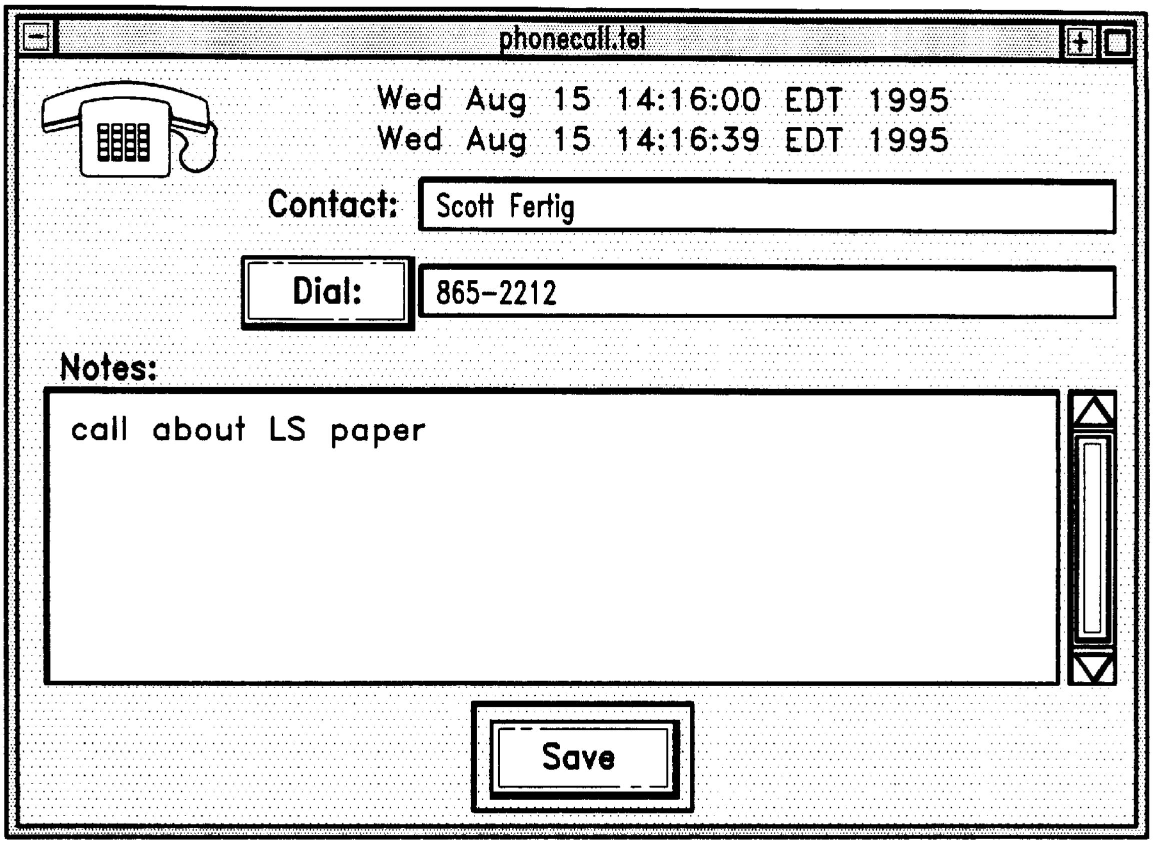
◆ Home

◇ Fax

FIG. 6A

```
WHO           ON                                   AT             ABOUT
-----------------------------------------------------------------------------
Scott Fertig  Tue  Aug  1  12:05  EDT 1995  432-6433       Port to PPC
Ward Mullins  Tue  Aug  1  11:57  EDT 1995  415 224-1912   Tcl/Java discussion
Beth Freeman  Tue  Aug  1  10:22  EDT 1995  432-1287       insurance
```

FIG. 6B



phonecall.tel

Wed Aug 15 14:16:00 EDT 1995
Wed Aug 15 14:16:39 EDT 1995

Contact: Scott Fertig

Dial: 865-2212

Notes:

call about LS paper

FIG. 7

Quote-O-Matic Stock Service for 5/16/95

GVIL	14.00
LMASX	20.84
ODWA	18.50
SPLS	27.12
TSA	19.25
lavtx	21.41

FIG. 8A

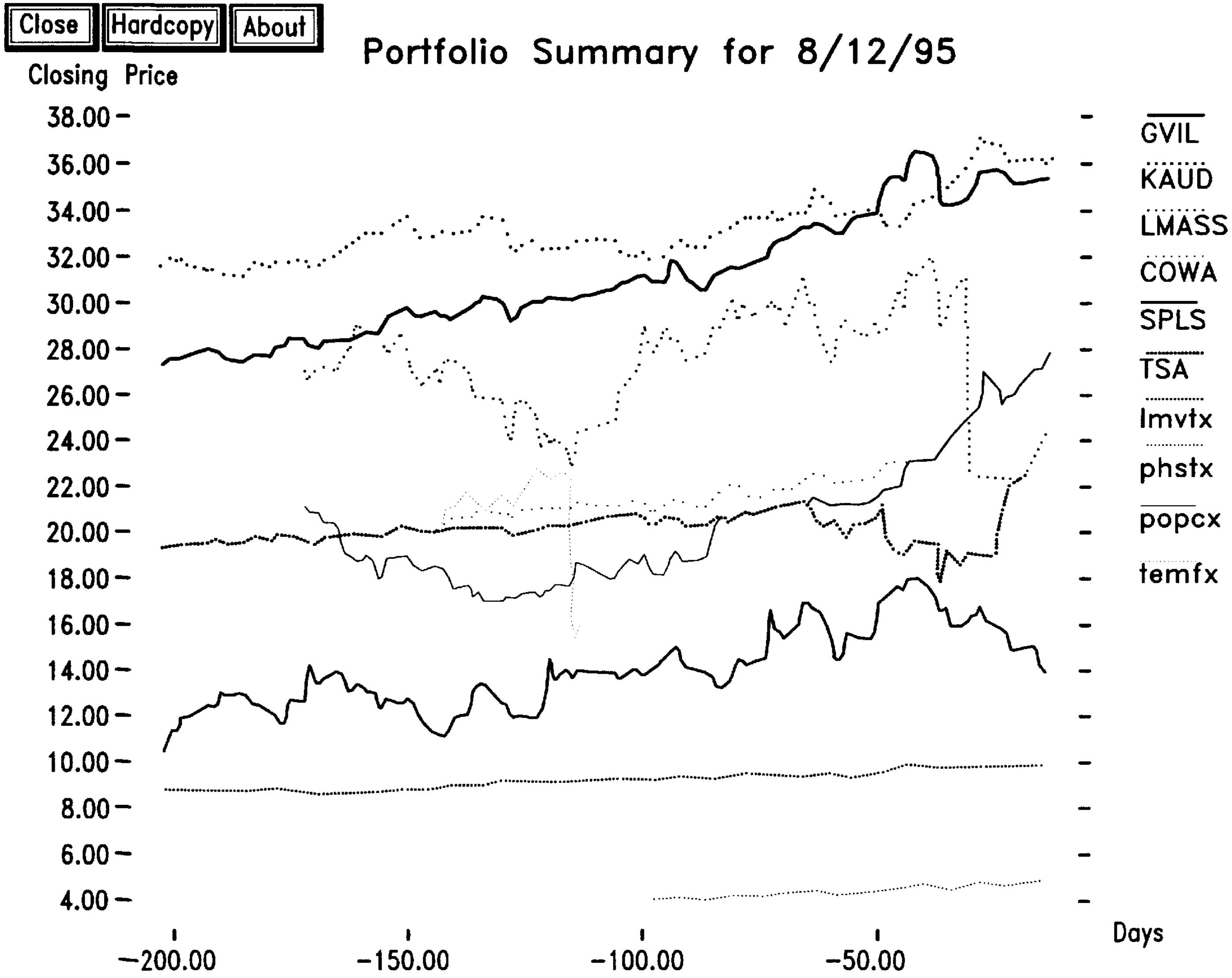

Close
Hardcopy
About
Portfolio Summary for 8/12/95
Closing Price
38.00
36.00
34.00
32.00
30.00
28.00
26.00
24.00
22.00
20.00
18.00
16.00
14.00
12.00
10.00
8.00
6.00
4.00
GVIL
KAUD
LMASS
COWA
SPLS
TSA
lmvtx
phstx
popcx
temfx
Days
−200.00
−150.00
−100.00
−50.00

FIG. 8B

DOCUMENT STREAM OPERATING SYSTEM

FIELD OF THE INVENTION

The present invention relates to an operating system in which documents are stored in a chronologically ordered "stream". In other words, that is, as each document is presented to the operating system, the document is placed according to a time indicator in the sequence of documents already stored relative to the time indicators of the stored documents.

Within this application several publications are referenced by arabic numerals within parentheses. Full citations for these and other references may be found at the end of the specification immediately preceding the claims. The disclosures of all of these publications in their entireties are hereby incorporated by reference into this application in order to more fully describe the state of the art to which this invention pertains.

BACKGROUND OF THE INVENTION

Conventional operating systems frequently confuse inexperienced users because conventional operating systems are not well suited to the needs of most users. For example, conventional operating systems utilize separate applications which require file and format translations. In addition, conventional operating systems require the user to invent pointless names for files and to construct organizational hierarchies that quickly become obsolete. Named files are an invention of the 1950's and the hierarchical directories are an invention of 1960's.

Some conventional operating systems employ a "desktop metaphor" which attempts to simplify common file operations by presenting the operations in the familiar language of the paper-based world, that is, paper documents as files, folders as directories, a trashcan for deletion, etc. Also, the paper-based model is a rather poor basis for organizing information where the state of the art is still a messy desktop and where one's choices in creating new information paradigms is constrained [1].

Thus, conventional operating systems suffer from at least the following disadvantages: (1) a file must be "named" when created and often a location in which to store the file must be indicated resulting in unneeded overhead; (2) users are required to store new information in fixed categories, that is directories or subdirectories, which are often an inadequate organizing device; (3) archiving is not automatic; (4) little support for "reminding" functions are provided; (5) accessibility and compatibility across data platforms is not provided and (6) the historical context of a document is lost because no tracking of where, why and how a document evolves is performed. "Naming" a file when created and choosing a location in which to place the file is unneeded overhead: when a person grabs a piece of paper and starts writing, no one demands that a name be bestowed on the sheet or that a storage location be found. Online, many filenames are not only pointless but useless for retrieval purposes. Storage locations are effective only as long as the user remembers them.

Data archiving is an area where conventional electronic systems perform poorly compared to paper-based systems. Paper-based systems are first and foremost archiving systems, yet data archiving is difficult in conventional desktop systems. Often, users throw out old data rather than undertaking the task of archiving and remembering how to get the data back. If archiving and retrieval of documents is convenient, old information could be reused more often.

Reminding is a critical function of computer-based systems [2] [3], yet current systems supply little or no support for this function. Users are forced either to use location on their graphical desktops as reminding cues or to use add-on applications such as calendar managers.

A solution to these disadvantages is to use a document stream operating system. One such system is outlined in a 1994 article [4]. However, this article fails to address many of the disadvantages of conventional operating systems.

SUMMARY OF THE INVENTION

One object of the present invention is to provide a document stream operating system and method which solves many, if not all, of the disadvantages of conventional operating systems.

Another object of the present invention is to provide a document stream operating system in which documents are stored in one or more chronologically ordered streams.

An additional object of the present invention is to provide an operating system in which the location and nature of file storage is transparent to the user, for example, the storage of the files is handled automatically and file names are only used if a user chooses to invent such names.

A further object of the present invention is to provide an operating system which takes advantage of the nature of electronic documents. For example, a conventional paper document can only be accessed in one place, but electronic documents can be accessed from multiple locations.

Another object of the present invention is to organize information as needed instead of at the time the document is created. For example, streams may be created on demand and documents may belong to as many streams as seems reasonable or to none.

An additional object of the present invention is to provide an operating system in which archiving is automatic.

A further object of the present invention is to provide an operating system with sophisticated logic for summarizing or compressing a large group of related documents when the user wants a concise overview. In addition, this summarizing can include pictures, sounds and/or animations. Also, no matter how many documents fall into a given category, the operating system is capable of presenting an overview in a form so that all the documents are accessible from a single screen.

Also, an object of the present operating system is to make "reminding" convenient.

Another object of the present invention is to provide an operating system in which personal data is widely accessible anywhere and compatibility across platforms is automatic. Accordingly, this invention provides that computers using the operating system of the present invention need not be independent data storage devices, but also act as "viewpoints" to data stored and maintained on external systems such as the INTERNET. Thus, in accordance with the present invention users can access their personal document streams from any available platform such as a UNIX machine, a Macintosh or IBM-compatible personal computer, a personal digital assistant (PDA), or a set-top box via cable.

According to one embodiment of the invention a computer program for organizing one or more data units is provided. The computer program includes: (1) means for receiving one or more of the data units, each of which is associated with one or more chronological indicators; and (2) means for linking each of the data units according to the chronological indicators to generate one or more streams of data units. Other embodiments of the invention also provide: (1) chronological indicators including past, present, and future times; and (2) means for displaying the streams, wherein respective indicia representing the data units are displayed and each data unit includes textual data, video data, audio data and/or multimedia data. The means for displaying the streams may further include displaying selected segments of the streams corresponding to selected intervals of time. The means for receiving may further include means for receiving data units from the Word Wide Web or from a client computer.

According to another embodiment of the invention, a method of organizing one or more data units is provided including the steps of: (1) receiving one or more data units, each of which is associated with one or more chronological indicators; and (2) linking each of the data units according to the chronological indicators to generate one or more streams of data units. In other embodiments, the chronological indicators may include past, present, and future times. The method may further include the steps of: (1) displaying the streams, wherein respective indicia represent each data unit and each of the data units may be textual data, video data, audio data and/or multimedia data. The step of displaying the streams may further include the steps of: (1) receiving from a user one or more values indicative of one or more selected segments of the streams corresponding to selected intervals of time; and (2) displaying the segments of the streams corresponding to the selected intervals of time.

These and other advantages of the present invention will become apparent from the detailed description accompanying the claims and the attached figures.

DESCRIPTION OF THE DRAWINGS

FIG. **1** shows a viewport in one embodiment of the present invention;

FIG. **2** shows a substream menu in one embodiment of the present invention;

FIG. **3** shows a list of summary types for the substream chosen in

FIG. **2** of the present invention;

FIG. **4** shows the time display in one embodiment of the present invention;

FIG. **5** shows a calendar-based dialog box in one embodiment of the present invention;

FIG. **6***a* shows a dialog box in connection with a phone call in one embodiment of the present invention;

FIG. **6***b* shows a summary of phone calls in one embodiment of the present invention;

FIG. **7** shows a phone call record dialog box in one embodiment of the present invention;

FIG. **8***a* shows text data used by one embodiment of the present invention; and

FIG. **8***b* shows the result of a summarize operation in one embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

This invention is a new model and system for managing personal electronic information which uses a time-ordered stream as a storage model and stream filters to organize, locate, summarize and monitor incoming information. Together, streams and filters provide a unified framework that subsumes many separate desktop applications to accomplish and handle personal communication, scheduling, and search and retrieval tasks. One embodiment of the present invention utilizes a machine-independent, client/server open architecture so that users can continue to use the same conventional document types, viewers and editors.

A "stream" according to the present invention is a time-ordered sequence of documents that functions as a diary of a person or an entity's electronic life. Every document created and every document send to a person or entity is stored in a main stream. The tail of a stream contains documents from the past, for example starting with an electronic birth certificate or articles of incorporation. Moving away from the tail and toward the present and future, that is, toward head of the stream more recent documents are found including papers in progress or new electronic mail. A document can contain any type of data including but not limited to pictures, correspondence, bills, movies, voice mail and software programs. Moving beyond the present and into the future, the stream contains documents allotted to future times and events, such as, reminders, calendar items and to-do lists. Time-based ordering is a natural guide to experience. Time is the attribute that comes closest to a universal skeleton-key for stored experience. Accordingly, streams add historical context to a document collection with all documents eventually becoming read-only, analogously as history becomes "set in stone". The stream preserves the order and method of document creation. Also, like a diary, a stream records evolving work, correspondence and transactions because historical context can be crucial in an organizational setting.

One embodiment of this invention allows for basic operations to be perform on a stream: new, clone, transfer, find and summarize.

Users create documents by means of the new and clone operations. New creates a new, empty document and adds the document to the main stream. Clone duplicates an existing document and adds the duplicate to the main stream at a new time point. Documents can also be created indirectly through the transfer operation. The transfer operation copies a document from one stream to another stream. Creation of a document if "transparent" because documents, by default, are added to the at the present time point. Internally, the document is identified by a time indication so no name is required from the user for the document. Nevertheless, a user can optionally name a document is desired.

Some streams can be organized on the fly with the find operation. Find prompts for a search query, such as "all E-mail I haven't responded to," or "all faxes I've sent to Schwartz" and creates a substream. Substreams, unlike conventional, virtual or fixed directories which only list filenames, present the user with a stream "view" of a document collection. This view, according to the present invention contains all documents that are relevant to the search query. Also, unlike searches of conventional fixed directories, the substream is generated by default from all the documents in the main stream. Accordingly, individual substreams may overlap, that is, contain some documents that are the same and can be created and destroyed on the fly without affecting the main stream or other substreams.

The find operation creates a substream of the main stream or of another substream based on, for example, a boolean attribute-and-keyword expression or a 'chronological expression', for example, "my last letter to Schwartz". Also, substreams may point to the future, for example, "my next appointment".

Once created, substreams operate dynamically, that is, if a user allows a substream to persist, the substream will collect new documents that match the search criteria as documents arrive from outside the operating system or as the user creates the document. This dynamic operation provides automatic monitoring of information because the substream not only organizes the documents as received but also filters for incoming information. For example, a substream created with the query "find all documents created by other people" would subsume a user's mailbox and automatically collect all arriving mail from other users. A substream remains in existence until destroyed by the user and acts as a filter by examining each new document that enters the main stream.

Although a document may belong to any number of substreams, the document also enters and remains on the main stream. A substream, in other words, is a "subset" of the main stream document collection. In other words, a way of looking at the main stream so as to exclude certain documents temporarily.

The summarize operation "compresses" or "squish" a stream to generate one or more overview documents. The content of an overview document depends on the type of documents in the stream. For instance, if the stream contains the daily closing prices of all the stocks and mutual funds in a user's investment portfolio, the overview documents may contain a chart displaying the historical performance of particular securities and the user's net worth. As another example, if the substream contains a list of tasks a user needs to complete, the overview document might display a prioritized "to-do" list. Thus, the summarize operation collapses a stream into a summary document. This summary document is a "live" document which is updated as additional documents are added to the main stream.

The type of summary depends on the type of documents in the substream. In one embodiment of the present invention at least one general "squish" function is provided no matter which stream is to be squished. Typically, however, the user will have a number of different squishers to choose from, for example, one squisher might produce a summary in words, while another squisher might produce a graph. Also "custom squishers" may be supplied by third parties or created by the user.

Another aspect of the invention is that applications execute "inside" a stream document leaving any output in that document. Thus, running an application is a variant of the new operation. For example, to run a spreadsheet application such as Lotus 1-2-3, a user creates a new document at the head of the main stream, specifically, a "live" spreadsheet document. The application itself is stored on the main stream, or located by means of a calling card that points to another stream containing the application.

A stream has three main portions: past, present, and future. The "present" portion of the stream holds "working documents", which also includes the timepoint in the stream where new documents are created and where incoming documents are placed. As documents age and newer documents are added, older documents pass from the user's view and enter the "past portion" where the documents are eventually "archived". By disappearing from view old information is automatically cleared away so the old information will not clutter up the workspace. At some future point if documents in the past portion are needed, such documents can be located with the find operation even if the past document has already been archived.

The "future" portion of the stream allows documents to be created in the future. Future creation is a natural method of posting reminders, for example, meeting dates and scheduling information. The system allows users to dial to the future by selecting a future timepoint for a document. The present invention keeps the document until that future time occurs. When the time of documents timepoint arrives the reminder document is brought into view and the document enters the present portion of the stream.

One embodiment of the present invention is implemented in a client/server architecture running over the Internet. The server is the workhorse of this embodiment handling one or more streams by storing all main stream and substream documents. Each view of a stream is implemented as a client of the server and provides the user with a "viewpoint" interface to document collections, that is, streams. The "look and feel" of the viewport may be different for different computing platforms but each viewport should support the basic operations.

One embodiment of the present invention implements a client viewport using graphically based X Windows, another embodiment implement a client viewport solely with text in standard ASCII (American Standard Characters for Information Interchange) and yet another embodiment implements a client viewport for the NEWTON personal digital assistant (PDA). The X Windows viewport provides the full range of functionalities including picture and movie display. In contrast the text-only viewport has a mail-like interface although all the basic operations are available. Also, because the NEWTON PDA lacks substantial internal memory and relies on slow external communications, that is low bandwidth, a minimal stream-access method is provided.

The X Windows viewport embodiment is shown in FIG. **1**. The interface is based on a visual representation of the stream metaphor **5**. Users can slide the mouse pointer **10** over the document representations to "glance" at each document, or use the scroll bar **20** in the lower left-hand corner to move through time, either into the past or into the future portion of the stream.

Color and animation indicate important document features. A red border in one embodiment means "unseen" and a blue border means "writable". Open documents may be offset to the side to indicate when the document is being edited. In this embodiment incoming documents slide in from the left side and newly created documents pop down from the top and push the steam backwards by one document into the past.

External applications are used to view and edit documents which the user can select by clicking on the documents graphical representation. The external applications speed the learning process significantly because new users can continue to use familiar applications for example, conventional UNIX application such as emacs, xv, and ghostview, to create and view documents while using streams to organize and communicate the documents.

The X Windows interface prominently displays the basic operations, that is, New **30**, Clone **40**, Xfer **50** (that is, transfer), Find **60**, and Summarize **70** as buttons and/or menus. As discussed previously the New button creates a new document and adds the document to the stream at the "present" timepoint. The Clone button duplicates an existing document and places the copy in the stream. The Xfer button first prompts the user for one or more mail addresses and then forwards the selected document. The find operation is supported through a text entry box **60** that allows the user to enter a boolean search query which results in a new substream being created and displayed. The summarize menu **70** generates a new document which displays information from documents in a stream in a desired format, for example, a graph.

The X Windows interface of this document also provides additional buttons. The Print button **80** copies a selected document to a printer where documents may be either printed conventionally or moved to a printer stream. A software agent which can be associated with the stream forwards each new document to an appropriate printer. The Freeze button **90** makes a document read-only.

Pulldown menus are used to select documents from streams or existing substreams, create summaries, initiate personal agents and change the clock.

The Streams menu **110** allows the user to select from a list of locally available streams.

FIG. **2** shows the Substreams menu **120** of one embodiment of the present invention. This menu is divided into three sections. The first section **130** contains a list of operations that can be performed on substreams, for example, remove. The second section **140** contains one menu entry labeled "Your Lifestream", and causes the viewport to display a user's main stream. The third section **150** lists all of the user's substreams. As indicated by this third section, substreams can be created in an incremental fashion, that is, one substream generated from another resulting in a nested set of menus. In this example the nested menus were created by first creating a substream "lifestreams and david" **160** from the main stream and then creating two substreams from this substream, "scenarios" and "ben" **170**. Substream "scott" **180** was created from the "scenarios" substream. Semantically this incremental substreaming amounts to a boolean 'and' of each new query with the previous substream's query.

FIG. **3** shows the summarize menu **190** which lists the possible summary types. Choosing any of these menu options creates a substream summary and a new document containing the summary is placed on the stream.

The Personal Agents menu **200** lists a number of available software agent types. Personal software agents can be added to the user interface in order to automate common tasks.

The embodiment illustrated in FIG. **4** always displays the time in the upper right hand corner of the viewport interface. This time display also acts as a "time" pull-down menu **190** that allows the user to set the viewport time to the future or past via a calendar-based dialog box as illustrated in FIG. **5**. Setting the viewport time causes the cursor to point to that timepoint position in the stream such that all documents forward of that timepoint, that is, towards the head of the stream have a future timestamp and all documents behind that timepoint, that is, towards the tail, have a past timestamp. As time progresses, this cursor moves forward towards the head of the stream. When the cursor slips in front of the present timepoint "future" documents are added to the visible part of the stream in the viewpoint, just like new mail arrives.

The effect of setting the time to the future or past is to reset the time-cursor temporarily to a fixed position designated by the user. Normally the user interface displays all documents from the past up to the time-cursor. Setting the time-cursor to the future allows the user to see documents in the future part of the stream. Creating a document in the future results in a document with a future timestamp. Once the user is finished time-tripping, the user can reset to the present time by selecting the "Set time to present" menu option in the time menu.

In one embodiment of the present invention "browse cards" **100** are employed so that when the user touches a document in the stream-display with the cursor, a browse card appears. The purpose of the browse card is to help the user identify a document by providing the user some idea of the document's contents in a small window. The content of browse cards is an abbreviated version of a document which as been compressed into an micro-document like an index card. In one embodiment, the browse card creation operation does header stripping so that the browse card displays the first non-trivial words in a document. In another embodiment, complex analysis is performed on the document contents so that 'most important' words, pictures and/or sounds are presented.

Another embodiment of the present invention provides "calling cards" which represent or point to a stream or substream Every stream has a calling card and the only way to reference a stream is via this calling card. In this embodiment the find operation performs as follows: (1) the user provides a search query; (2) an appropriate substream is generated; (3) the substream's calling card is generated; and (4) the new calling card is deposited as a new document at the head of the main stream. Every duplicated calling card bears on the face text, an icon or both. In the case of the find operation, the new calling card is marked with the argument supplied by the user for the search query, for example "from: Schwartz and Lifestreams" or "last letter from Piffel". As a default in this embodiment, the interface will automatically display the new substream.

Another embodiment of the present invention allows documents to be grouped explicitly into a substream. With this feature the user marks, that is, selects all documents to be included in the substream and groups the selected documents into a substream by creating a new calling card. The new calling card comes equipped with a system-created icon which is marked on all documents that are part of the new stream and the user may add any other notation to the face of the new calling card, for example, "these should be merged together to produce the Zeppelin report."

In the embodiments with calling cards the "transfer" operation takes two arguments: a document and a calling card so that the document is copied onto the stream designated by the calling card. The document may itself be a calling card and depending on instructions from the user, either the calling card itself or the stream designated by the calling card is copied onto the new stream.

Each main stream in this embodiment has a calling card which allows 'inter-main stream' communication. To communicate a user includes on the face of the calling card for the user's main stream whatever information the user is willing to make public. Other users wanting to send that electronic mail will need a copy of that user's calling card, which might be, for example, "Rock Q. Public, Blimp Mechanic, Passaic N.J."

To give only limited access to user's stream, a user provides a copy of the calling card customized to provide the desired access. Minimal access gives other users append-only privileges, that is, user B can send user A mail, but cannot view anything on user A's stream. Access restrictions beyond "minimal" are stated in terms of substreams. In other words, a calling card gives access to all documents contained in the specified substream unless that document is also contained on one of specified excluded substreams.

The present invention allows a stream document to contain another stream, that is a 'stream envelope'. A stream envelope is equivalent to a 'value' calling card versus the 'reference' calling cards discussed above. In other words, rather than point to another stream with a calling card, the stream envelope contains a copy all the documents from the other stream. For example, user A transfers to user B a substream consisting of all Zeppelin-manual correspondence which contains many documents. However, a single new document appears on user B's stream: a stream envelope. The stream envelope may be opened yielding the many documents of the forwarded stream.

According to the present invention, a text-editor designed specifically for stream A can treat a document as a stream of bytes so that software agents designed to 'ride' streams could ride documents as well. Also, the stream find operation can scan the streamed document and synchronization based on stream properties can be applied to the streamed document.

Streams can be copied and combined into new streams, that is, streams can be merged. For example, if a user acquires stream segments from ten electronic newspapers and magazines all covering the same one-month period, the segments can be merged in a sorted order into a single combined stream.

Another feature of the present invention is a card gallery which consists of some reasonable number of microdocuments, for example twelve, arranged in such a way that each is always fully visible on the viewport for example, in two columns of 6 each at the right of the display. Each micro-document is a calling card or a micro-browse-card (MBC) to a "regular" document on the stream or in a squish. The micro browse card in the card gallery represent documents a user has been working on. Whenever a user opens a document or creates a substream or squish, the corresponding micro browse card is added to the card gallery. A user can re-open the document, squish or have the viewport display the substream, by clicking on, or otherwise selecting, the corresponding micro browse card.

The micro browse card is administered as a least-recently-used cache, that is, new cards are dealt on top of the least-recently-used existing card, however, users can override this mechanism and place or lock a card in the gallery. For example, a live squish can act as weather-station, appointment calendar, stock ticker or other current-status reporter if the user locks the micro browse card for the squish in the gallery.

Because a users' card gallery includes by default the calling cards of streams the user has recently opened, the card gallery acts, to the extent streams are used as Web sites, as a World Wide Web "hot list."

In one embodiment of the present invention at least some part of the stream is in the form of a receding stack of upright rectangles, framed in such a way that only the top line of each document is visible. A foreshortened viewing angle yields a view that is approximately a right triangle, the bottom edge aligned with the bottom of the display and the left edge aligned with the display's left border.

In another embodiment of the inventive operating system a 'slide rule' bar display is provided which is labelled with the endpoints of the stream, that is, the dates of the farthest-past and farthest-future documents. The document density can be illustrated, for example, by the amount of color saturation of the bar at any point. This type of display aids the user because some days, weeks, months or other time period have more associated documents, some have fewer. The slide rule has a magnifier that the user can slide via a mouse, for example up and down the bar. The magnifier obscures the portion of the sliderule that lies beneath, but the obscured segment is replaced by an enlarged view of the small part of the stream starting at the point touched by the upper edge of the magnifier or, some similar protocol for defining the starting point of the magnified segment. By sliding the magnifier, you change the part of the stream currently displayed in the main perspective view.

In another embodiment of the present invention at least some part of the stream is in the form of a conventional calendar month display. With this display, the stream-segment associated with day n appears as a list of document headers in n's calendar box.

To contemplate the future instead of the past, according to one embodiment of the present invention, a user can reverse the stream so that the head of the stream now appears in front, with the nearest-future document immediately behind that document, next-nearest behind that and so forth. The user then looks from the present into the future so that furthest-away document in the display equals furthest-away in future time.

All documents older than some date d may be moved by the server from immediately-accessible storage to cheaper, long-term storage. When a document is archived in this way, however, the browse card of that document may remain available in immediately-accessible storage, so that the archived document appears in the regular way in the viewport. When a user opens an archived document, the user may incur some delay as the server locates and reloads the body of the document.

Automatic archiving is a feature of the standalone embodiment and user-managed web site embodiment. In either embodiment, the streams operating system monitors remaining disk space and when available space is low, the operating system asks the user to pop in some diskettes or other storage media. Similarly when an archived document needs to be reloaded, the operating system tells the user which diskettes or other storage media to insert. In another embodiment of the present invention a chat feature is provided. If two users want to chat online in UNIX TALK style; the user creates a new stream and each user focusses the viewport on that new stream. To make a comment, a user pops a new document on the stream head with the comment contained as text inside the document. The stream synchronization properties allow many users to manipulate a stream concurrently, and allow a user to block at the end waiting for the arrival of a new document which would mean in this case awaiting the next comment.

A chat stream by its nature provides: (1) permanent record and (2) support for multiple parties to a conversation. A chat stream is in this sense is a real-time bulletin board. In this regard a network bulletin board may be stored in a stream providing: (1) archived comments that can be searched and retrieved using the standard streams operations; (2) synchronization characteristics like a chat stream; and (3) a bulletin board that can be located via the find operation.

In another embodiment of the present invention any software agent with the necessary access can ride your stream. Therefore streams can be the basis of groupware systems implemented for example as a flock of agents. For example, when user A's wants to schedule a meeting, a software agent departs from user A's stream to visit the streams of each of the other intended participants. Each user's stream lists the current appointments in the stream's future portion, and each user also includes a document giving the user's general availability in pre-arranged terms so that the meeting-maker software agent can understand. When the software agent finds an appropriate meeting time, the software agent posts a document to each stream's future and creates a new stream for the meeting itself. The software agent forwards the calling card of the meeting stream to each participant. This new stream serves as a chat stream on which the participants can discuss the meeting beforehand, accumulate any material developed during the meeting itself and persists when the meeting is over as a record and a vehicle for post-meeting discussions.

In the following embodiments a stream naturally provides a structure for storing technical an electronic versions of a newspaper or magazine.

In addition, a mail-order firm might store its catalog in a stream with each document describing one item. A top page can embed calling cards as hyperlinks so that the streams pointed-to are updated automatically by the persistent substream mechanism. Each user can also reformat the catalog to taste by creating a substream containing descriptions of whatever sort of object interests the user.

In another embodiment of the present invention a phone conversation is stored as a time-ordered sequence of spoken sounds or as electronic representations. When two users want to have a phone conversation, the users can use software such as a software agent, that creates a new stream and hands each user the calling card. Each user's 'phone agent' tosses digitized representations of speech frames onto the stream and grabs each new frame that appears, turning each speech frame into sound. In this scheme, phone and voicemail are integrated in the all-purpose stream context and can be manipulated using the standard stream operations.

Additionally, in another embodiment of the present invention a television source can be stored as a time-ordered sequence of sound-and-image frames. Such television information is an archive as well as a realtime source and can be searched and substreamed. A television set is merely a viewport. Also, scheduling information can be stored in the television stream's future and tuning into a television station only requires double-clicking on the appropriate calling card. Similar embodiments can provide for radio stations, music sources, etc.

A stream according to the present invention can be controlled by a voice-interface as well as a computer and thereby be accessed via a conventional phone. The voice interface would allow: (1) the stream to be searched and manipulated; (2) new objects to be installed; (3) objects to be transferred; and (4) other capability.

The following embodiment discusses how the present invention is used for electronic mail. To send a message, the user creates a new document, for example by clicking on the New button and composes the message using a favorite editor. After composition, the message document is sent with a push of the Xfer button. Similarly, existing documents are easily forwarded to other users, or can be cloned and replied to. While all mail messages, both incoming and outgoing, are intermixed with other documents in the stream, the user can create a mailbox by substreaming on documents created by other users. A user can also create substreams that contain a subset of the mailbox substream, such as "all mail from Bob", or "all mail I haven't responded to".

With the present invention, a reminder can be generated as future electronic mail, that is a user can send mail that will arrive in the future. If the user dials to the future before writing a message document, when the message document is transferred the message document will not appear on recipient's stream until either that time arrives or the recipient happens to dial the recipient's viewport to the set creation date. In the present, the document will be in the stream data structure but the viewport will not show the document. By appearing just-in-time and not requiring the user to switch to yet another application, these reminders are more effective

than those included in a separate calendar or scheduling utility program.

One embodiment of the present invention supports an electronic business cards document type as well as a 'phone call record' document for noting the date and time of phone contacts. In addition, the task of creating a phone call record is automated through a personal agent. The personal software agent is automatically attached to the personal agent menu so that anytime a user wants to make a call the use chooses "Make Phonecall" from the personal agent menu. The agent is spawned and the dialog box in FIG. **6***a* appears. The user types in the name of the callee and the agent searches the current stream for a business card with that name. If the name is found, the software agent creates and fills in the appropriate entries of the phone call record as seen in FIG. **7**. This functionality is similar to the use of the personal assistant on the Newton personal digital assistant. The user can later use the streams summarize operation to summarize the phone calls made. This results in a report as shown in FIG. **6***b*.

In another embodiment of the present invention this functionality is extended to include the functions of a time manager. Time managers generally track the billable hours a professional spends on one or more projects. In streams this is easily accomplished by creating a timecard that marks the starting and ending time of each task. These timecards are just thrown onto the stream as used. Then, before each billing period, the stream is summarized by the timecards, resulting in a detailed billing statement for each contract.

Another embodiment of the present invention organizes a user's personal finance. Large number of users already track their checking accounts, savings, investments, and budgets with applications such as QUICKEN. The types of records and documents used in these applications such as electronic checks, deposits, securities transactions, reports are conveniently stored and generated by streams.

For example, a stock quote service may forward the daily closing prices of a given portfolio to a user's stream at the end of every business day. These documents are as shown in FIG. **8***a*. Such documents can list each stock and mutual fund along with its closing price, giving the user a method of calculating the value of the user's assets on a specific day. But if the user wants higher-level view of the portfolio over time the summarize operation can be used. For example, the user first selects a substream containing the stock quote documents and selects the "summarize by portfolio" menu item. This operation compresses the data into a single chart of historical data which summarize the portfolio documents in the substream. This result is illustrated in FIG. **8***b*.

Another embodiment of present invention provides a stream-based checking account. Each check written creates a record on the users stream. Some of these checks are electronic checks sent to companies with an online presence; other checks are transcribed from written checks. The user, in this embodiment, employs a personal software agent to help balance his checkbook. At year's end the user runs a tax summary which squishes the financial information in the users stream onto income tax forms which can be sent electronically to the Internal Revenue Service.

Streams can also be used for budgeting, tracking expenditures, etc. Streams contain everything a user deals with in the user's electronic life in a convenient and searchable location.

As discussed previously, every user can send out custom calling cards that grant access to a user's stream. Thus, the particular user's stream can function as a personal World

Wide Web site such that the web site is merely a subset of the user's main stream or a substream. For the convenience of external users, a user can generate a "guide to this stream" document that functions as a top page. In the context of the present invention, a hyperlink, or a bookmark is just a calling card. By double clicking, or some comparable mechanism, on a calling card the viewport displays the specified stream. Embedding a link from one document to another document means to embed calling cards.

The present invention's personal web site provides more features than a conventional worldwide user side because: (1) the web site and personal information site are unified and maintained simultaneously with the same toolset; (2) visitors to the site use the same interface as for the visitor's own stream, that is, the visitor can browse, create substreams and squish; (3) visitors can be given customized access levels so that friendly visitors get to see more; and (4) the personal web site can filter incoming documents.

Streams of the present invention are designed to work with conventional World Wide Web browsers, thus opening a document of type web bookmark causes the appropriate browser to fire-up as an application the way a text editor fires up when the user opens a text document. However, streams also provide an indigenous web-browsing model. Key features such as calling cards and find provide this functionality so that the viewport itself functions as the browser.

Streams may also be quite useful for managing information outside of the system. For example, keeping track of web bookmarks is difficult and bookmarks are inconvenient to pass to other users. Conventional systems accomplish those transactions by copying a Web address from a web browser to an electronic mail message which the recipient then copies from electronic mail back to recipient's browser and adds this web address as a bookmark. Streams solve both of these problems.

In one embodiment, an agent watches each user's bookmark file for each time a new bookmark is added and then adds the same bookmark to a stream as a new Web address document. The effect of opening a Web address document in a stream is that the web browser comes to the foreground and attempts to connect to the Web address. In this way streams create a bookmark substream while at the same time making the data in the bookmarks readily available to any other search a user may make.

Passing Web addresses around is trivial, the user merely copies the Web address document to another user's stream (a one-step process) and the Web address is automatically included in the recipient bookmark substream.

A stream is a data structure that can be examined and to the extent possible manipulated by many processes simultaneously. Also a process may block the end of a stream, that is, suspend the stream operation, until awakened when a new document appears on the stream head. Streams need to support the block-at-the-end operation so that a software agent or what amounts to the same thing, that is, a substream or a live squish document can examine each new document arriving at the stream.

A stream must support simultaneous access because: (1) a user creates many software agents which may need to examine the stream concurrently; and (2) a user may have granted other users limited access to the user's stream, and the user will want access to this stream even while the other users access the stream.

One embodiment of the present invention is configured such that each server may support three to four simultaneous users with stream sizes on the order of 100,000 documents (perhaps a year or two of documents for the average user). In another embodiment, the operating system is configured such that lifestreams may have millions of documents or more. The substreaming aspect of one embodiment of the present invention is efficiently implemented using an inverse index of the document collection maintained by the server. No real performance problems with respect to retrieval have occurred. Given the very large indices that are being used on the Internet the retrieval scheme is expected to scale to large document collections.

Since a user is unlikely look at 10,000 documents at once and discern any usable information, the present invention does not provide the user with an entire document collection at once. Instead "cursors" are used to allow the user to view segments of the document collection and to load in more segments as needed.

One embodiment of the invention provides a single-threaded server which allows a single point of access. Other embodiments of the present invention utilize a multi-server and multi-threaded approach which provides a more scalable architecture.

Regarding the term "agent" used in this application, it is noted that this term refers one of three kinds of embedded computations: personal agents, document agents, and stream agents. Personal agents are typically attached to the user interface and can automate tasks or can learn from the user's interactions with streams. Document agents live on documents and are spawned by various events, for example, the first time that a document is accessed. Stream agents are attached to streams and execute whenever the stream changes in some way, for example, a new document appears on the stream.

Further, regarding the term "document", it is noted that this term includes traditional text based files, electronic mail files, binary files, audio data, video data, and multimedia data.

Additionally, this document stream operating system can be implemented as an independent operating system with all required subsystems such as: a storage subsystems in software and/or hardware for writing documents to disc drive, tape drives and the like; interrupt handling subsystems; and input/output subsystems. However, the present invention also encompasses implementations which utilize subsystems from other operating systems such as the Disk Operating System (DOS), WINDOWS, and OPERATING SYSTEM 7. In such implementations, the graphic user interface (GUI) of the other operating system can be replaced by the present invention viewports. Alternatively, the present invention can operate as a document stream utility for the other operating system.

It must be noted that although the present invention is described by reference to particular embodiments thereof, many changes and modifications of the invention may become apparent to those skilled in the art without departing from the spirit and scope of the invention, which is only limited by the appended claims. For documents may have associated attributes used to locate the document during a search, for example, a special code word selected by the user.

References

[1] Theodor Nelson. The right way to think about software design. In The Art of Human-Computer Interface Sesign (Ed.) Brenda Laurel, 1990.

[2] Thomas W. Malone. How do people organize their desks?Implications for the design of office information

systems. ACM Transactions on Office Systems, 1(1):99–112, January 1983.

[3] M. Lansdale. The psychology of personal information management. Applied Ergonomics, March 1988.

[4] David Gelernter. The cyber-road not taken. The Washington Post, April 1994.

What is claimed is:

1. A computer system which organizes each data unit received by or generated by the computer system, comprising:

means for generating a main stream of data units and at least one substream, the main stream for receiving each data unit received by or generated by the computer system, and each substream for containing data units only from the main stream;

means for receiving data units from other computer systems;

means for generating data units by the computer system;

means for selecting a timestamp to identify each data unit;

means for associating each data unit with at least one chronological indicator having the respective timestamp;

means for including each data unit according to the timestamp in the respective chronological indicator in the main stream; and

means for maintaining the main stream and the substreams as persistent streams.

2. The computer system of claim **1**, wherein each timestamp is selected from the group consisting of: past, present, and future times.

3. The computer system of claim **1**, wherein each data unit includes textual data, video data, audio data and/or multimedia data.

4. The computer system of claim **1**, wherein the means for receiving further comprises means for receiving data units from the World Wide Web.

5. The computer system of claim **1**, wherein said means for receiving further comprises means for receiving data units from a client computer.

6. The computer system according to claim **1**, further comprising:

means for displaying alternative versions of the content of the data units.

7. A computer system according to claim **1** further comprising:

means for summarizing the contents of data units in one of the streams to generate one or more overview data units and for including the overview data unit in one of the streams.

8. A computer system according to claim **7**, wherein the means for summarizing further comprises means for continuously updating the overview data units to include changes in the contents of data units in the stream being summarized.

9. A computer system according to claim **1** further comprising:

means for archiving a data unit associated with a timestamp older than a specified time point while retaining the respective chronological indicator and/or a data unit having a respective alternative version of the content of the archived data unit.

10. The computer system of claim **1**, wherein the computer program further comprises:

means for operating on any of the streams using a set of operations selected by a user.

11. The computer system of claim **1** further comprising: means to generate substreams from existing substreams.

12. A computer system as in claim **1**, further comprising:

means for generating a data unit comprising an alternative version of the content of another data unit; and

means for associating the alternative version data unit with the chronological indicator of the another data unit.

13. A method which organizes each data unit received by or generated by a computer system, comprising the steps of:

generating a main stream of data units and at least one substream, the main stream for receiving each data unit received by or generated by the computer system, and each substream for containing data units only from the main stream;

receiving data units from other computer systems;

generating data units in the computer system;

selecting a timestamp to identify each data unit;

associating each data unit with at least one chronological indicator having the respective timestamp;

including each data unit according to the timestamp in the respective chronological indicator in at least the main stream; and

maintaining at least the main stream and the substreams as persistent streams.

14. The method of claim **13**, wherein each timestamp is selected from the group consisting of: past, present, and future times.

15. The method of claim **13**, further comprising the step of displaying the streams on a display device as visual streams.

16. The method of claim **15**, wherein the step of displaying the streams further comprises the steps of:

a) receiving from a user one or more indications of one or more selected segments of the streams corresponding to one or more selected intervals of time, and

b) displaying the selected segments.

17. The method of claim **13**, wherein each data unit includes textual data, video data, audio data and/or multimedia data.

18. The method of claim **13**, further comprising the step of:

providing access to a first stream from a second stream by generating a data unit indicating the first stream.

19. The method of claim **13**, further comprising the steps of:

selecting access privileges to provide to a first stream from a second stream; and

providing access to the first stream from the second stream according to the access privileges.

20. The method of claim **13**, further comprising the step of:

displaying data from one of the data units in abbreviated form.

21. The method of claim **13**, further comprising the step of:

summarizing the contents of data units in a stream to generate one or more overview data units and including the overview data unit in one of the streams.

22. The method of claim **13**, further comprising the step of:

archiving data units having timestamps older than a specified time point.

23. A computer system for organizing each data unit received by or generated by the computer system, comprising:

means for generating a main stream of data units and at least one substream, the main stream for receiving each data unit received by or generated by the computer system, and each substream for containing data units only from the main stream; means for associating each data unit with at least one chronological indicator having a respective timestamp which identifies the data unit; means for including each data unit according to the timestamp in a respective chronological indicator in the main stream; means for maintaining the main stream and substreams as persistent streams;

means for generating a data unit having indicia to allow access to a first stream from a second stream;

means for including the data unit having the indicia in the second stream; and

means for providing access to the first stream from the second stream in accordance with the indicia.

24. A computer system according to claim **23** further comprising:

means for providing limited access to the first stream from the second stream by generating a data unit indicating access privileges to the first stream.

25. A computer system for organizing each data unit received by or generated by the computer system, comprising:

means for generating a main stream of data units and at least one substream, the main stream for receiving each data unit received by or generated by the computer system, and each substream for containing data units only from the main stream; means for associating each data unit with at least one chronological indicator having a respective timestamp which identifies the data unit; means for including each data unit according to the timestamp in a respective chronological indicator in the main stream; means for maintaining the main stream and the substreams as a persistent streams;

means for representing one or more data units of a selected stream on a display device as document representations, each document representation including the timestamp of the respective data unit and the order of appearance of each data representation on the display device determined by the timestamp of the respective data unit;

means for selecting which data units are represented on the display device by selecting one of the document representations and displaying document representations corresponding to data units having timestamps within a range of a timepoint; and

means for selecting one or more of the document representations with a pointing device so that the data units represented by the selected document representations are further displayed with a second document representation comprising an alternative version of the content of the respective data unit.

26. A computer system as in claim **25**, wherein the document representations form a visual stream having a three-dimensional effect.

27. A computer system as in claim **26**, wherein the three-dimensional effect further comprises a perspective view.

28. A computer system as in claim **25**, wherein each document representation comprises a polygon and the polygons overlap to form a visual stream of polygons.

29. A computer system as in claim **25**, wherein the alternative version is an abbreviated version.

30. A computer system as in claim **25**, wherein the alternative version is a caption version.

31. A computer system as in claim **25**, wherein the alternative version is an expanded version.

32. A computer system as in claim **25**, further comprising:

means for selecting one or more alternative versions of the content of a respective data unit to display another alternative version of the content of the data unit.

33. A computer system as in claim **25**, further comprising:

means for updating the display device to provide a document representation for data units associated with chronological indicators having timestamps which become the present time.

* * * * *

US005247580A

United States Patent [19]

Kimura et al.

[11] **Patent Number: 5,247,580**

[45] **Date of Patent: Sep. 21, 1993**

[54] **VOICE-OPERATED REMOTE CONTROL SYSTEM**

[75] Inventors: **Toshiyuki Kimura; Kazuo Yabe,** both of Kawagoe, Japan

[73] Assignee: **Pioneer Electronic Corporation,** Tokyo, Japan

[21] Appl. No.: **917,672**

[22] Filed: **Jul. 22, 1992**

Related U.S. Application Data

[63] Continuation of Ser. No. 578,914, Sep. 7, 1990, abandoned.

[30] **Foreign Application Priority Data**

Dec. 29, 1989 [JP] Japan 1-341630

[51] **Int. Cl.**^{5} **G10L 5/06;** G10L 3/00; H03G 3/70; H04M 11/00

[52] U.S. Cl. **381/43;** 381/41; 381/46; 381/105; 381/110; 379/102

[58] **Field of Search** 381/41-43, 381/45-48, 104, 105, 110; 379/102

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,286,115	8/1981	Sakoe	381/43
4,654,881	3/1987	Dolikian et al.	379/102
4,864,623	9/1989	Van Nes et al.	381/110
5,003,602	3/1991	Koyama	381/43
5,020,107	5/1991	Rohani et al.	381/43

FOREIGN PATENT DOCUMENTS

0052916	5/1981	Japan	381/110

Primary Examiner—Dale M. Shaw
Assistant Examiner—Kee M. Tung
Attorney, Agent, or Firm—Sughrue, Mion, Zinn, Macpeak & Seas

[57] **ABSTRACT**

A voice-operated remote control system which transmits a remote control signal in response to a voice command has a degree-of-importance determining unit for determining the degree of importance of the voice command that is applied to the remote control system. The degree-of-importance determining unit sends a degree-of-importance signal corresponding to the degree of importance of the voice command to a recognition accuracy determining unit. Depending on the degree of importance of the input voice command as indicated by the degree-of-importance signal, the recognition accuracy determining unit determines whether the accuracy of the recognition result is high or low, and delivers only the recognition result of higher recognition accuracy to a transmitting circuit.

5 Claims, 11 Drawing Sheets

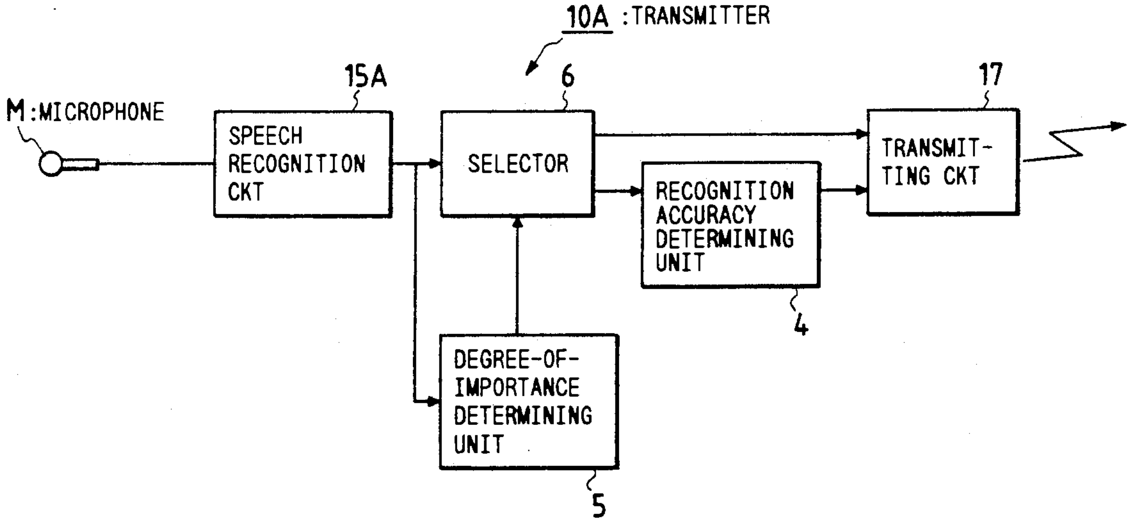


FIG. 1

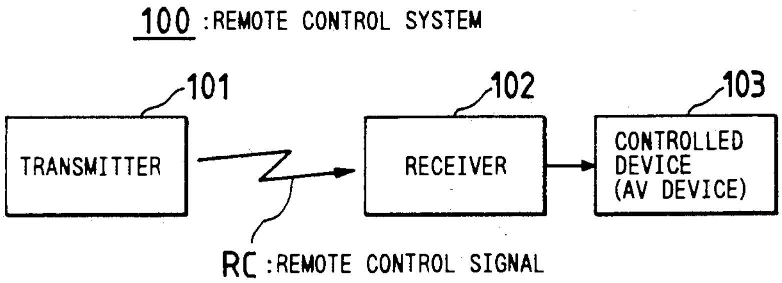

100 :REMOTE CONTROL SYSTEM
101
102
103
TRANSMITTER
RECEIVER
CONTROLLED DEVICE (AV DEVICE)
RC :REMOTE CONTROL SIGNAL

FIG. 2

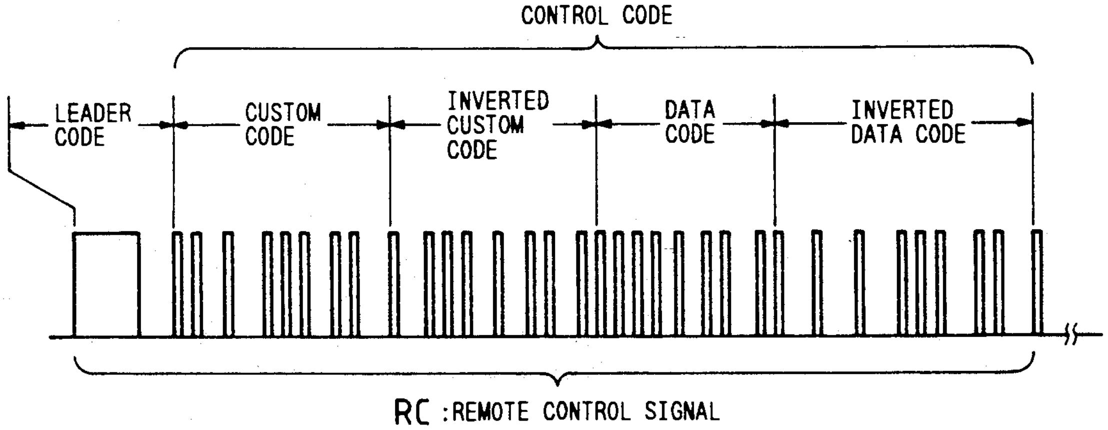

CONTROL CODE
LEADER CODE
CUSTOM CODE
INVERTED CUSTOM CODE
DATA CODE
INVERTED DATA CODE
RC :REMOTE CONTROL SIGNAL

FIG. 3

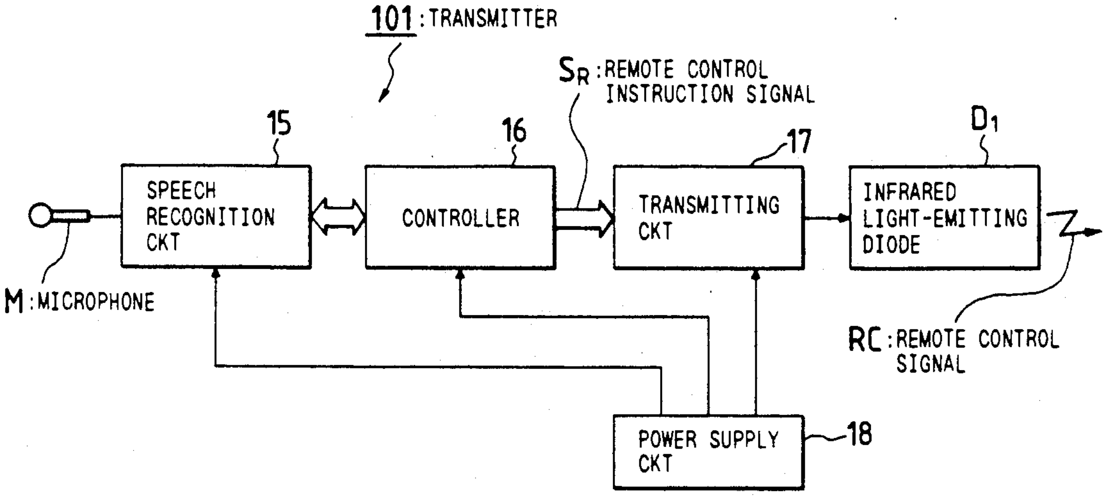

101 : TRANSMITTER
SR : REMOTE CONTROL INSTRUCTION SIGNAL
15
16
17
D1
SPEECH RECOGNITION CKT
CONTROLLER
TRANSMITTING CKT
INFRARED LIGHT-EMITTING DIODE
M : MICROPHONE
RC : REMOTE CONTROL SIGNAL
POWER SUPPLY CKT
18

FIG. 4

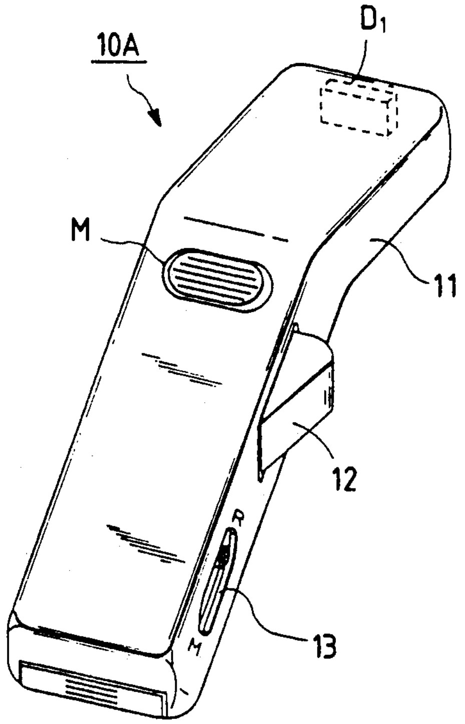

D1
10A
M
11
12
R
M
13

FIG. 5

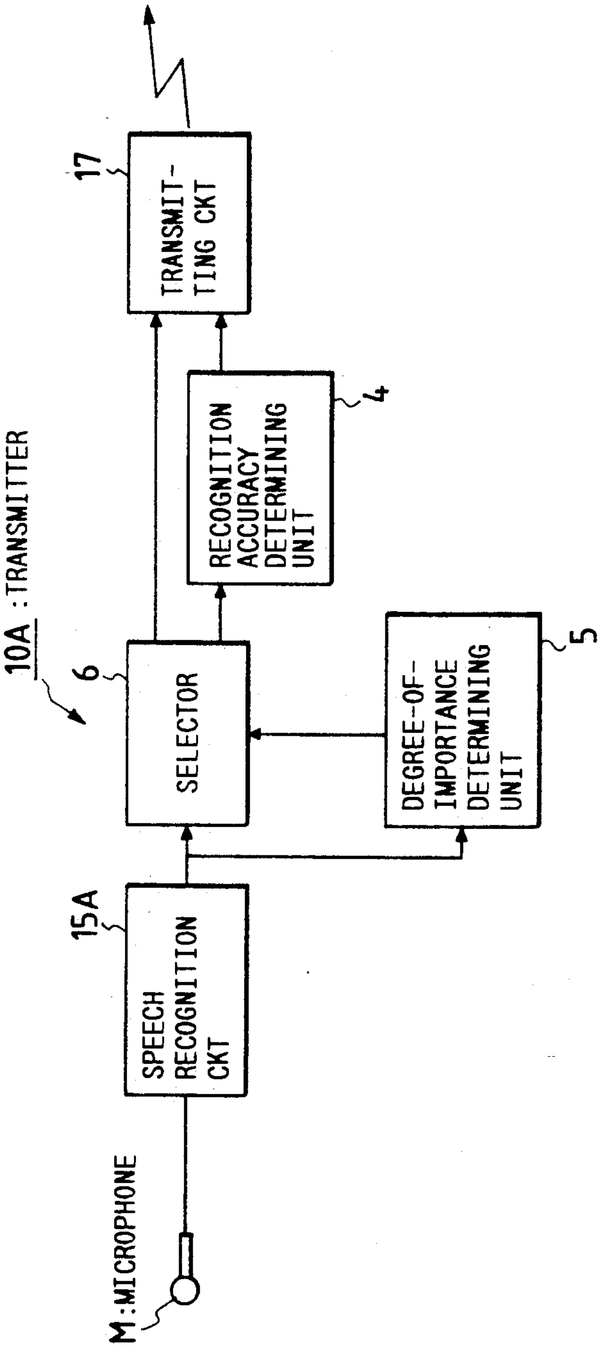

10A :TRANSMITTER
M:MICROPHONE
15A
SPEECH RECOGNITION CKT
6
SELECTOR
5
DEGREE-OF-IMPORTANCE DETERMINING UNIT
4
RECOGNITION ACCURACY DETERMINING UNIT
17
TRANSMIT-TING CKT

FIG. 6

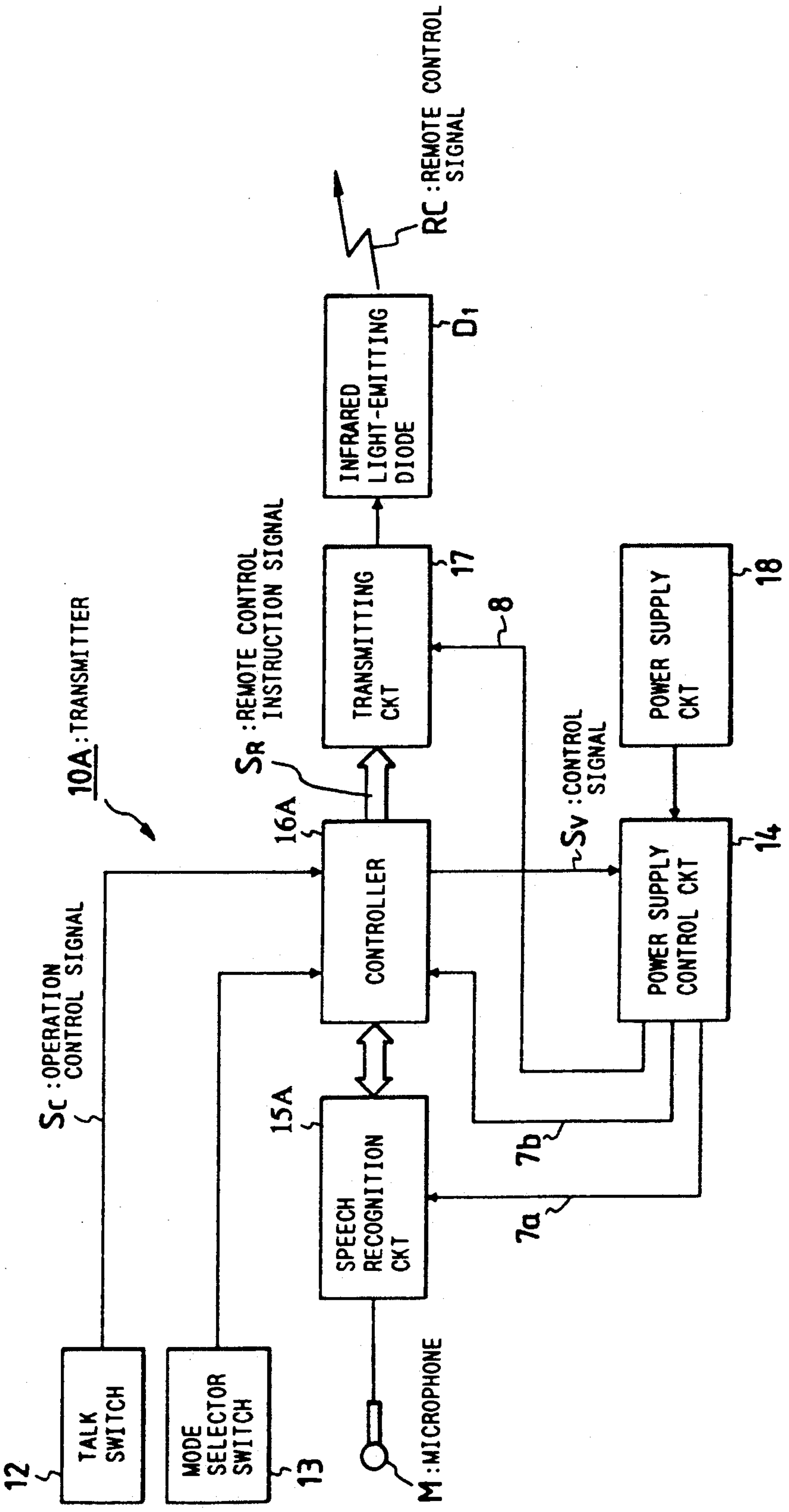

10A:TRANSMITTER
12
TALK SWITCH
13
MODE SELECTOR SWITCH
M:MICROPHONE
Sc:OPERATION CONTROL SIGNAL
15A
SPEECH RECOGNITION CKT
16A
CONTROLLER
SR:REMOTE CONTROL INSTRUCTION SIGNAL
17
TRANSMITTING CKT
INFRARED LIGHT-EMITTING DIODE
D1
RC:REMOTE CONTROL SIGNAL
8
Sv:CONTROL SIGNAL
14
POWER SUPPLY CONTROL CKT
18
POWER SUPPLY CKT
7a
7b

FIG. 7

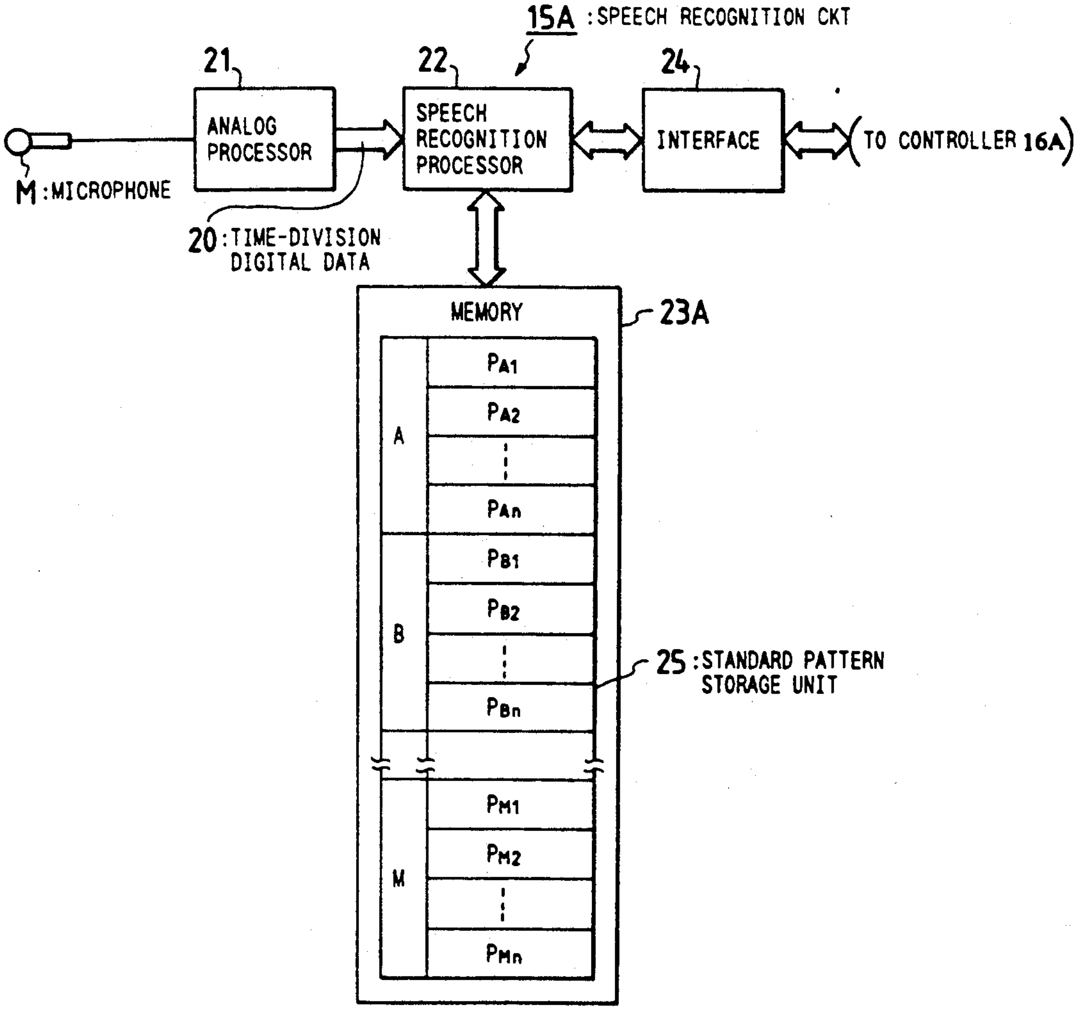

15A :SPEECH RECOGNITION CKT
21
ANALOG PROCESSOR
22
SPEECH RECOGNITION PROCESSOR
24
INTERFACE
(TO CONTROLLER 16A)
M :MICROPHONE
20 :TIME-DIVISION DIGITAL DATA
MEMORY
23A
A
P_{A1}
P_{A2}
P_{An}
B
P_{B1}
P_{B2}
P_{Bn}
25 :STANDARD PATTERN STORAGE UNIT
M
P_{M1}
P_{M2}
P_{Mn}

FIG. 8

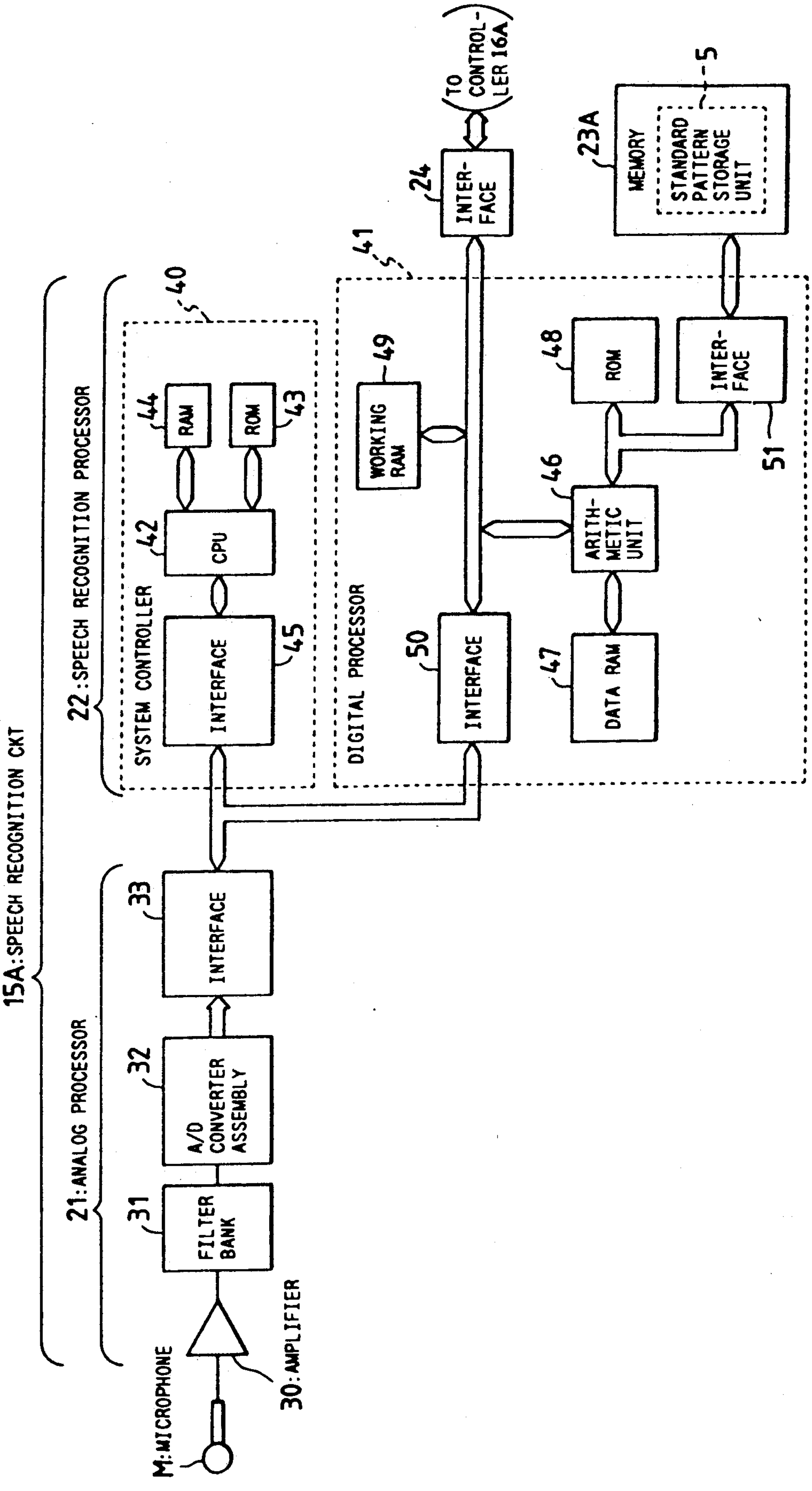

15A:SPEECH RECOGNITION CKT
21:ANALOG PROCESSOR
22:SPEECH RECOGNITION PROCESSOR
M:MICROPHONE
30:AMPLIFIER
31
FILTER BANK
32
A/D CONVERTER ASSEMBLY
33
INTERFACE
40
SYSTEM CONTROLLER
42
CPU
44
RAM
43
ROM
45
INTERFACE
41
DIGITAL PROCESSOR
49
WORKING RAM
50
INTERFACE
46
ARITH-METIC UNIT
47
DATA RAM
48
ROM
51
INTER-FACE
24
INTER-FACE
TO CONTROL-LER 16A
23A
MEMORY
5
STANDARD PATTERN STORAGE UNIT

FIG. 9(a)

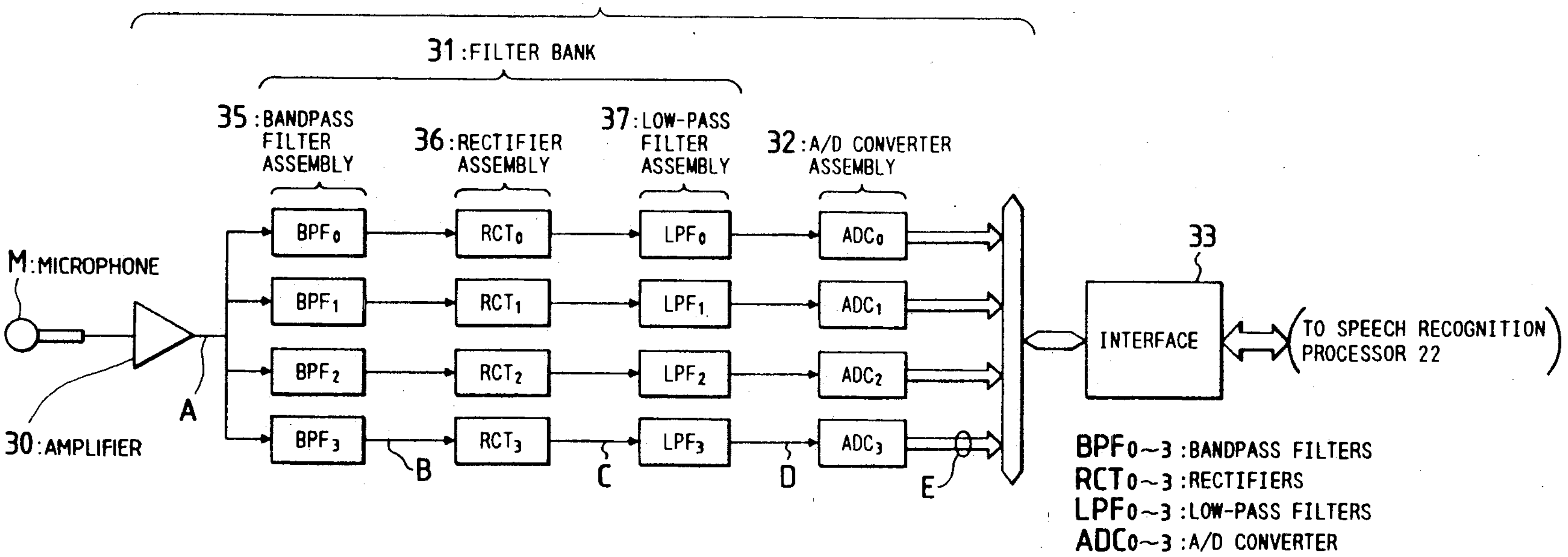


FIG. 9(b)

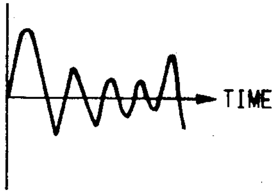


FIG. 9(c)

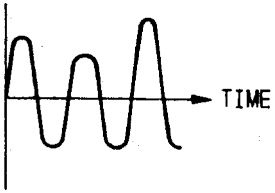


FIG. 9(d)

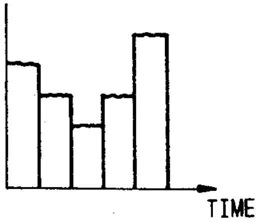


FIG. 9(e)

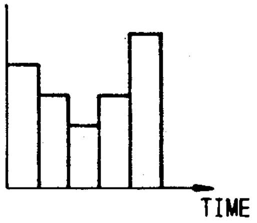


FIG. 9(f)

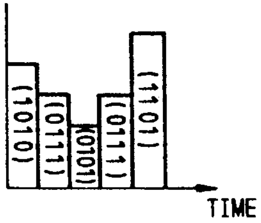


FIG. 10

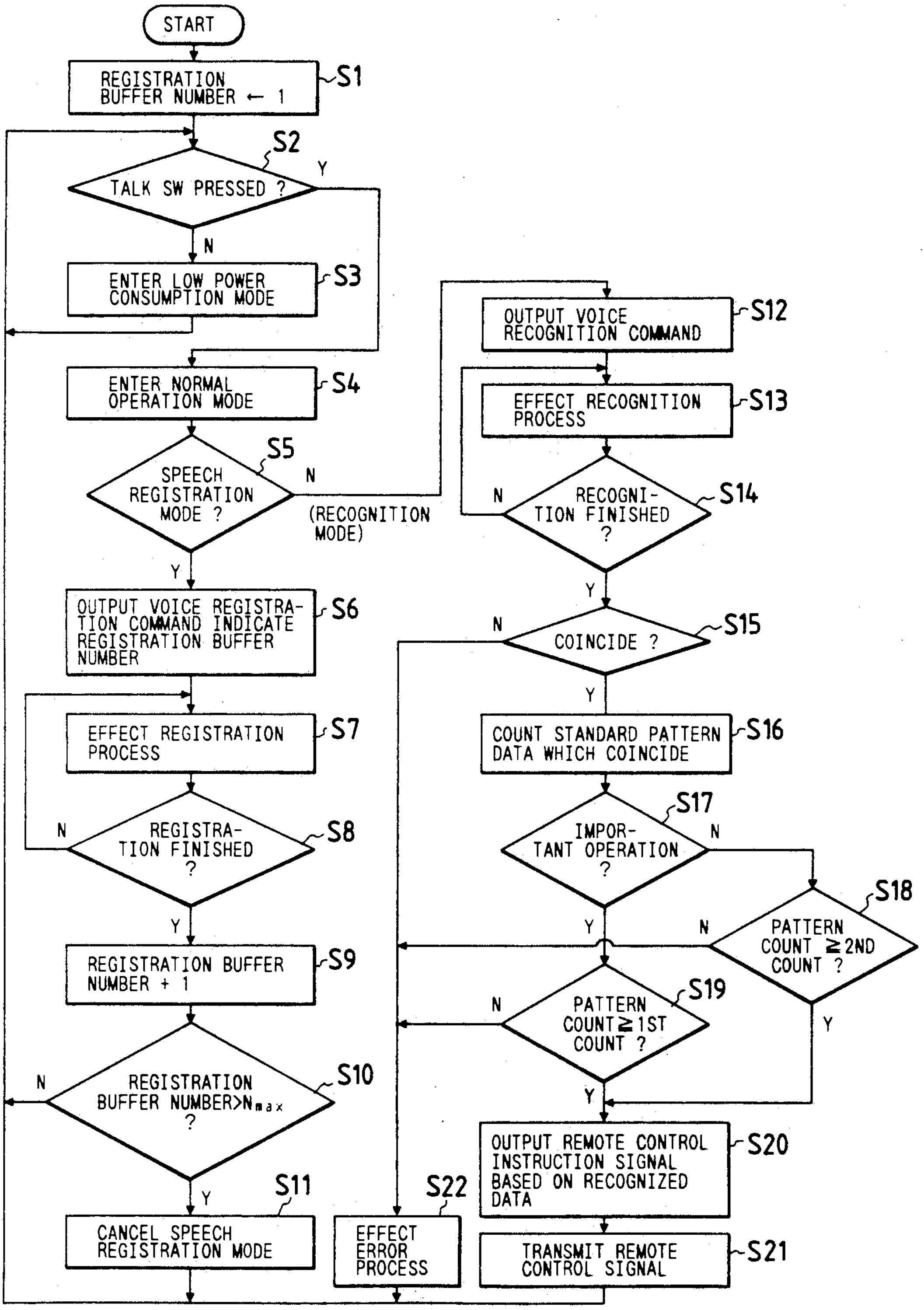

START
REGISTRATION BUFFER NUMBER ← 1
S1
TALK SW PRESSED ?
S2
Y
N
ENTER LOW POWER CONSUMPTION MODE
S3
ENTER NORMAL OPERATION MODE
S4
SPEECH REGISTRATION MODE ?
S5
N
(RECOGNITION MODE)
Y
OUTPUT VOICE REGISTRATION COMMAND INDICATE REGISTRATION BUFFER NUMBER
S6
EFFECT REGISTRATION PROCESS
S7
REGISTRATION FINISHED ?
S8
N
Y
REGISTRATION BUFFER NUMBER + 1
S9
REGISTRATION BUFFER NUMBER>N_max ?
S10
N
Y
CANCEL SPEECH REGISTRATION MODE
S11
OUTPUT VOICE RECOGNITION COMMAND
S12
EFFECT RECOGNITION PROCESS
S13
RECOGNITION FINISHED ?
S14
N
Y
COINCIDE ?
S15
N
Y
COUNT STANDARD PATTERN DATA WHICH COINCIDE
S16
IMPORTANT OPERATION ?
S17
N
Y
PATTERN COUNT ≥ 2ND COUNT ?
S18
N
Y
PATTERN COUNT ≥ 1ST COUNT ?
S19
N
Y
OUTPUT REMOTE CONTROL INSTRUCTION SIGNAL BASED ON RECOGNIZED DATA
S20
TRANSMIT REMOTE CONTROL SIGNAL
S21
EFFECT ERROR PROCESS
S22

FIG. 11

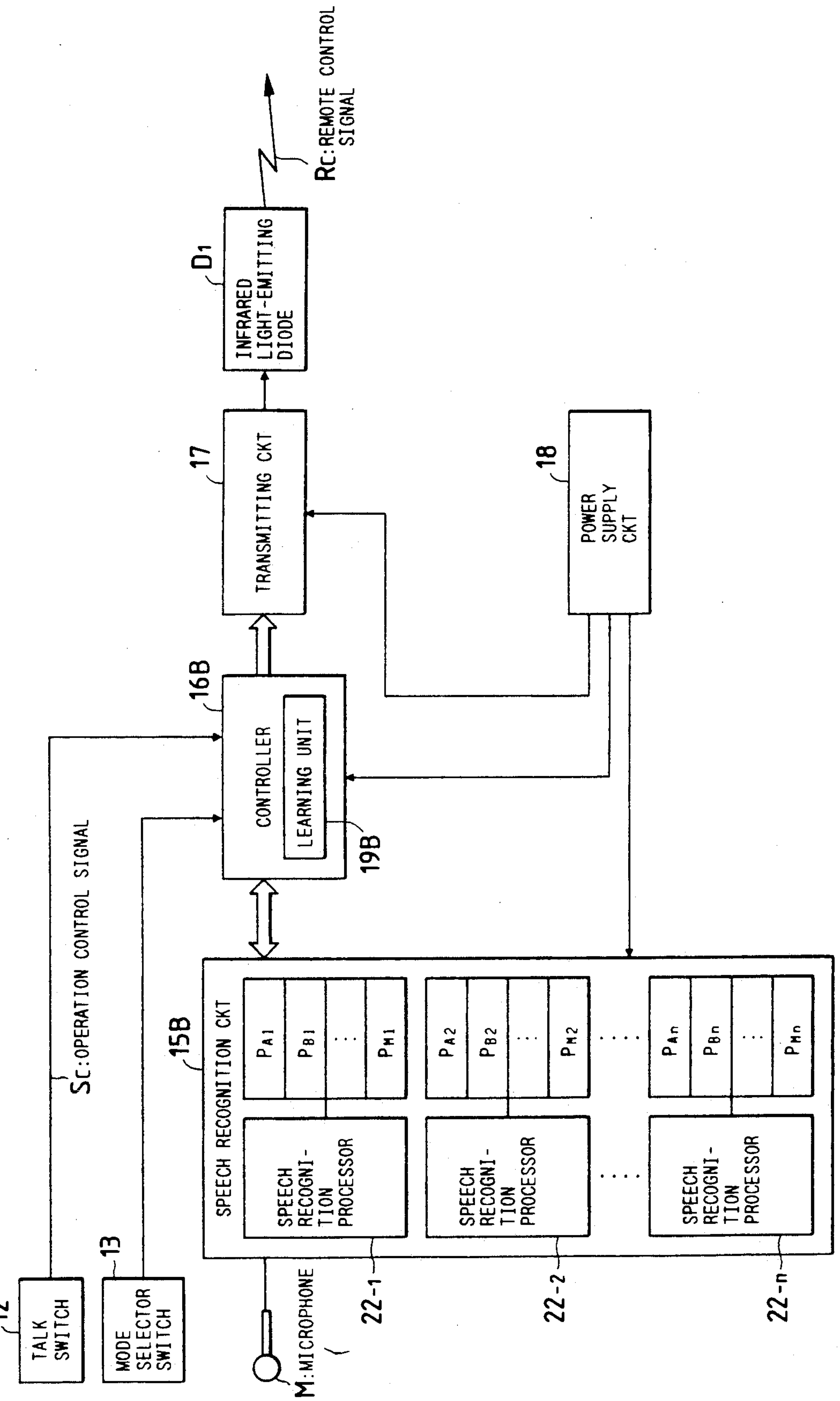

Rc: REMOTE CONTROL SIGNAL
D1
INFRARED LIGHT-EMITTING DIODE
17
TRANSMITTING CKT
18
POWER SUPPLY CKT
16B
CONTROLLER
LEARNING UNIT
19B
Sc: OPERATION CONTROL SIGNAL
15B
SPEECH RECOGNITION CKT
PA1
PB1
PM1
PA2
PB2
PM2
PAn
PBn
PMn
SPEECH RECOGNI-TION PROCESSOR
SPEECH RECOGNI-TION PROCESSOR
SPEECH RECOGNI-TION PROCESSOR
12
TALK SWITCH
13
MODE SELECTOR SWITCH
M: MICROPHONE
22-1
22-2
22-n

FIG. 12

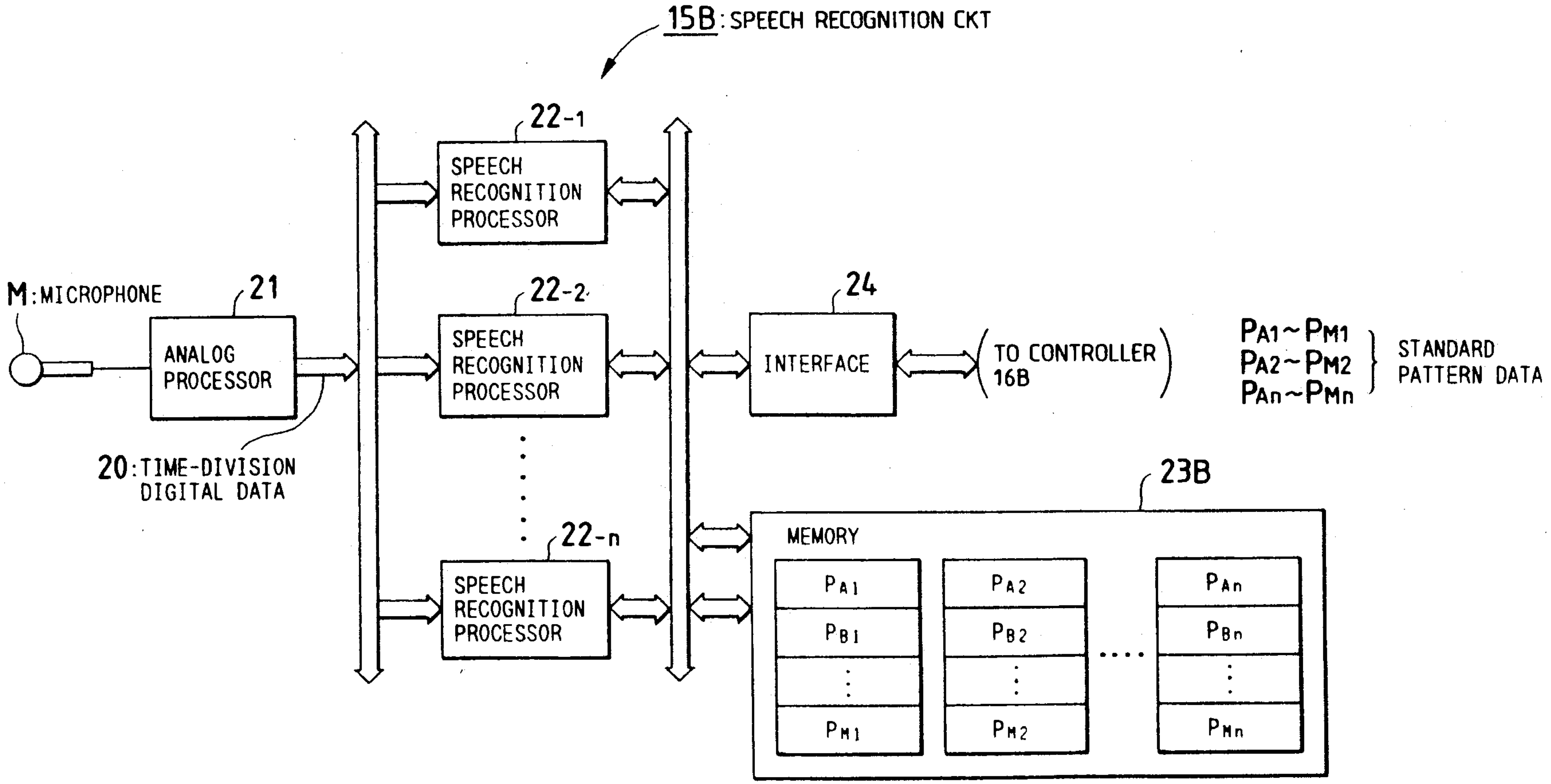
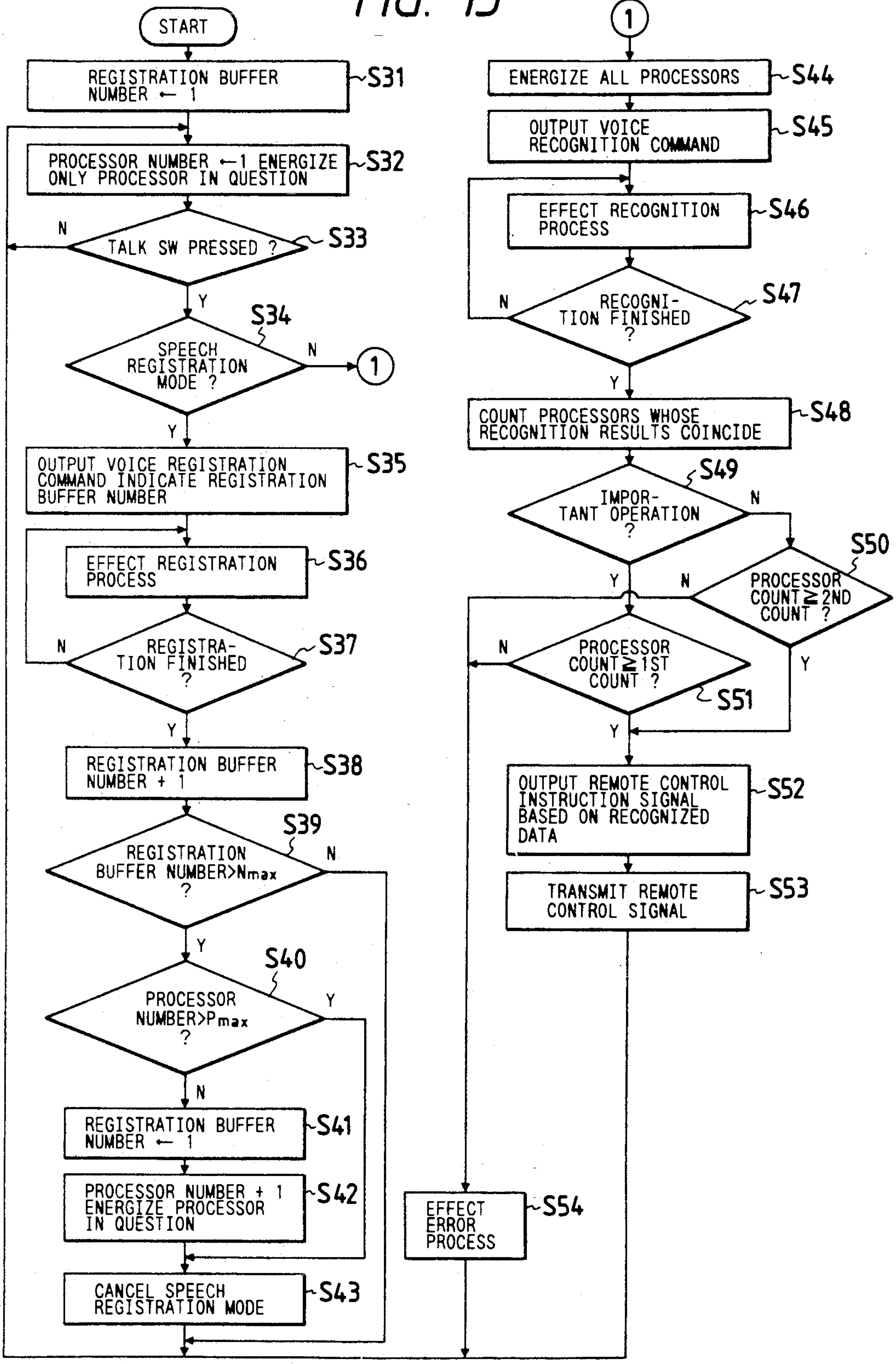

15B: SPEECH RECOGNITION CKT
M: MICROPHONE
21
ANALOG PROCESSOR
20: TIME-DIVISION DIGITAL DATA
22-1
SPEECH RECOGNITION PROCESSOR
22-2
SPEECH RECOGNITION PROCESSOR
22-n
SPEECH RECOGNITION PROCESSOR
24
INTERFACE
TO CONTROLLER 16B
PA1~PM1
PA2~PM2
PAn~PMn
STANDARD PATTERN DATA
23B
MEMORY
PA1
PB1
PM1
PA2
PB2
PM2
PAn
PBn
PMn

FIG. 13


START
REGISTRATION BUFFER NUMBER ← 1
S31
PROCESSOR NUMBER ←1 ENERGIZE ONLY PROCESSOR IN QUESTION
S32
TALK SW PRESSED ?
S33
N
Y
SPEECH REGISTRATION MODE ?
S34
N
1
Y
OUTPUT VOICE REGISTRATION COMMAND INDICATE REGISTRATION BUFFER NUMBER
S35
EFFECT REGISTRATION PROCESS
S36
REGISTRATION FINISHED ?
S37
N
Y
REGISTRATION BUFFER NUMBER + 1
S38
REGISTRATION BUFFER NUMBER>Nmax ?
S39
N
Y
PROCESSOR NUMBER>Pmax ?
S40
Y
N
REGISTRATION BUFFER NUMBER ← 1
S41
PROCESSOR NUMBER + 1 ENERGIZE PROCESSOR IN QUESTION
S42
CANCEL SPEECH REGISTRATION MODE
S43
1
ENERGIZE ALL PROCESSORS
S44
OUTPUT VOICE RECOGNITION COMMAND
S45
EFFECT RECOGNITION PROCESS
S46
RECOGNITION FINISHED ?
S47
N
Y
COUNT PROCESSORS WHOSE RECOGNITION RESULTS COINCIDE
S48
IMPORTANT OPERATION ?
S49
N
Y
PROCESSOR COUNT≥2ND COUNT ?
S50
N
Y
PROCESSOR COUNT≥1ST COUNT ?
S51
N
Y
OUTPUT REMOTE CONTROL INSTRUCTION SIGNAL BASED ON RECOGNIZED DATA
S52
TRANSMIT REMOTE CONTROL SIGNAL
S53
EFFECT ERROR PROCESS
S54

VOICE-OPERATED REMOTE CONTROL SYSTEM

This is a continuation of application Ser. No. 07/578,914 filed Sep. 7, 1990 now abandoned.

BACKGROUND OF THE INVENTION

The present invention relates to a remote control system for remotely controlling various electronic devices, and more particularly to a remote control system for remotely controlling devices such as AV (audio visual) devices by way of voice commands.

In recent years, various AV devices such as stereo sets, television receivers, cassette tape decks, video tape decks, compact disk players, laser vision disk players, or the like are equipped with remote control systems.

A remote control system has a transmitter which is usually positioned remotely from a controlled AV device. The transmitter, when operated, transmits a remote control signal, such as an infrared remote control signal, which is received by a receiver in the controlled AV device. The received remote control signal is decoded to control the AV device as intended by the remote control signal.

There has recently been developed a voice-operated remote control commands entered through keys. The voice-operated remote control system has a microphone mounted on a transmitter for converting a voice command into an electric voice signal, and a speech recognition LSI (Large Scale Integration) circuit for generating a remote control signal which corresponds to a voice pattern represented by the voice signal. The remote control signal thus generated is transmitted to a receiver in a controlled AV device.

In the conventional voice-operated remote control system, a remote control signal corresponding to the recognition result of a speech recognition process is transmitted as it is.

The degrees of importance of operations of a controlled device which correspond to respective remote control signals, i.e., the magnitudes of effects caused by erroneous recognition, may not necessarily be the same. For example, any trouble caused by erroneous recognition of a voice command is greater with respect to a recording operation than a reproducing operation. If control commands are handled with suitable different degrees of importance depending on the type of the control commands rather than with the same degree of importance, then the speech recognition can be processed highly efficiently, and more reliability can be achieved for more important control commands.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a voice-operated remote control system which can vary a speech recognition process depending on the degree of importance of a control command.

According to the present invention, there is provided a voice-operated remote control system comprising a microphone for entering a voice command, speech recognition means for comparing a pattern of the entered voice command with a predetermined standard pattern to recognize the contents of the voice command, transmitting means for generating and transmitting a remote control signal corresponding to the command data based on the result of recognition, degree-of-importance determining means for determining the degree of importance of the contents of the voice command to produce a degree-of-importance signal, and recognition accuracy determining means for determining the accuracy of the result of recognition of the voice command depending on the degree of importance thereof as indicated by the degree-of-importance signal, and for sending only a result of recognition which has a relatively high accuracy of recognition to the transmitting means.

The speech recognition means effects a speech recognition process through comparison between a pattern of a voice command entered through the microphone and a standard pattern. The degree-of-importance determining means determines the degree of importance of the voice command based on the result of recognition. The degree of importance thus determined is given to the recognition accuracy determining means, which determines the accuracy of the recognition result depending on the determined degree of importance of the voice command. For example, a level for determining the accuracy of recognition for a voice command of higher importance is higher, and a level for determining the accuracy of recognition for a voice command of lower importance is lower. The recognition accuracy determining means delivers only a recognition result of a higher recognition accuracy to the transmitting means. Therefore, only the recognition result which is of higher importance and higher recognition accuracy is converted into a remote control signal by the transmitting means for transmission to a remotely controlled device. Accordingly, an erroneous operation of the remotely controlled device due to erroneous recognition is prevented from happening.

The above and other objects, features and advantages of the present invention will become more apparent from the following description when taken in conjunction with the accompanying drawings in which preferred embodiments of the present invention are shown by way of illustrative example.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of a general remote control system;

FIG. **2** is a diagram showing a remote control signal by way of example;

FIG. **3** is a block diagram of the transmitter of a general voice-operated remote control system;

FIG. **4** is a perspective view of the transmitter of a voice-operated remote control system according to a first embodiment of the present invention;

FIG. **5** is a block diagram of the transmitter of the voice-operated remote control system according to the first embodiment;

FIG. **6** is a detailed block diagram of the transmitter according to the first embodiment;

FIG. **7** is a block diagram of a speech recognition circuit according to the first embodiment;

FIG. **8** is a detailed block diagram of the speech recognition circuit according to the first embodiment;

FIG. **9**(*a*) is a diagram showing an analog processor;

FIGS. **9**(*b*) through **9**(*f*) are diagrams showing the waveforms of signals in the analog processor shown in FIG. **9**(*a*);

FIG. **10** is a flowchart of an operation sequence of the transmitter according to the first embodiment;

FIG. **11** is a block diagram of the transmitter of a voice-operated remote control system according to a second embodiment of the present invention;

FIG. **12** is a block diagram of a speech recognition circuit according to the second embodiment; and

FIG. 13 is a flowchart of an operation sequence of the transmitter according to the second embodiment.

DETAILED DESCRIPTION OF THE INVENTION

General Remote Control System

For a better understanding of the present invention, a general remote control system and a voice remote control system will first be described below.

As shown in FIG. 1, a remote control system **100** comprises a transmitter **101** for transmitting a remote control signal from a position remote from a controlled device **103** such as an AV device, and a receiver **102** for receiving the transmitted remote control signal, decoding the remote control signal, and sending the decoded information to the controlled device **103**.

FIG. 2 shows a general remote control signal. The remote control signal is composed of a leader code which indicates the transmission of data to a receiver, a custom code and an inverted custom code which indicate a controlled device, a data code and an inverted data code which indicate a control command for the controlled device. The inverted custom code and the inverted data code are used to detect any error in the custom code and the data code, respectively.

FIG. 3 schematically shows the transmitter **101** of the voice-operated remote control system **100**. The transmitter **101** has a microphone M for converting a voice command into an electric signal. The converted electric signal is applied to a speech recognition circuit **15** in the form of a speech recognition LSI circuit or the like which includes a microprocessor. The speech recognition circuit **15** recognizes the contents of the applied electric signal, and produces command data corresponding to the recognized contents. The transmitter **101** also has a controller **16** comprising a microprocessor. Based on the command data from the speech recognition circuit **15**, the controller **16** produces and applies a remote control instruction signal SR to a transmitting circuit **17**, which then energizes an infrared light-emitting diode **D1** to transmit a remote control signal RC. The above components of the transmitter **101** are supplied with electric energy from a power supply circuit **18**.

When a voice command is received through the microphone M, the speech recognition circuit **15** converts the voice command into pattern data. The speech recognition circuit **15** compares the voice command pattern data with a plurality of standard pattern data which are stored therein, and determines the distance between the voice command data and the standard pattern data, and outputs command data corresponding to the standard pattern data, the distance of which from the voice command pattern data is smallest. There may also be employed another speech recognition process in which the similarity of the compared pattern data is determined according to a known simple similarity method and command data corresponding to the standard pattern data which has the highest similarity are outputted. The command data thus produced are applied to the controller **16**.

The controller **16** sends a remote control signal SR corresponding to the applied command data to the transmitting circuit **17**. In response to the supplied remote control instruction signal SR, the transmitting circuit **17** drives the infrared light-emitting diode **D1** to transmit a remote control signal RC. The controlled device **103** is therefore remotely controlled by the remote control signal RC.

First Embodiment

A voice-operated remote control system according to a first embodiment of the present invention will now be described below with reference to FIGS. **4** through **10**.

[External Structure]

As shown in FIG. **4**, a transmitter **10**A of the voice-operated remote control system has a unitary casing **11** which allows the operator to carry the transmitter freely around. The casing **11** supports a microphone M on an upper panel thereof. The microphone M converts a voice command given by the operator into an electric signal. An infrared light-emitting diode **D1**, for example, is mounted in one end of the casing **11**. The infrared light-emitting diode **D1** is used to transmit a remote control signal to the receiver of a remotely controlled device (not shown). On one side of the casing **11**, there is disposed a voice input switch (hereinafter referred to as a "talk switch") **12** which is closed when pressed and can automatically be released or opened when released. The talk switch **12** may be an automatic-return pushbuttom switch or a slide-type switch. When a voice command is to be entered, the talk switch **12** is closed to operate the transmitter **10**A. Otherwise, the talk switch **12** is open keeping the transmitter **10**A out of operation. The casing **11** also supports, on its side, a mode selector switch **13** in the form of a slide-type switch, for example. The mode selector switch **13** serves to select one of modes at a time. The modes include a speech registration mode in which a voice command is registered in the transmitter **10**A and a speech recognition mode in which a voice command is recognized, as described later on. The casing **11** accommodates therein an electronic circuit of the voice-operated remote control system according to the present invention.

Electronic Circuit Structure

FIG. **5** shows in block form the electronic circuit of the transmitter **10**A of the voice-operated remote control system according to the present invention. The transmitter **10**A has a speech recognition circuit **15**A for recognizing a voice command entered from a microphone M according to the pattern recognition process, a degree-of-importance determining unit **5** for determining the degree of importance of a recognized voice command, a selector **6** for selecting circuits or destinations to which an output signal from the speech recognition unit **15**A is to be sent, depending on the determined degree of importance, a recognition accuracy determining unit **4** for determining the accuracy of recognition of a voice command which has a high degree of importance, and a transmitting circuit **17** for converting recognized voice command data into a remote control signal RC and transmitting the remote control signal RC.

As shown in FIG. **6**, the transmitter **10**A has a controller **16**A to which the talk switch **12** and the mode selector switch **13** are connected. The controller **16**A applies a remote control instruction signal SR to the transmitting circuit **17** which energizes the infrared light-emitting diode **D1** to transmit a remote control signal RC to the receiver of a remotely controlled device. The speech recognition circuit **15**A, the controller **16**A, and the transmitting circuit **17** are supplied with electric energy from a power supply circuit **18** through a power supply control circuit **14** and power supply wires.

As shown in FIG. **7**, the speech recognition circuit **15A** comprises an analog processor **21** for processing an analog voice command signal which is received through the microphone M and outputting the processed analog voice command signal as a time-division digital data **20**, a speech recognition processor **22** for recognizing the voice command based on the time-division digital data **20** from the analog processor **21**, a memory **23A** for storing standard pattern data for speech recognition, and an interface **24** for transmitting signals to and receiving signals from the controller **16A**.

The memory **23A** includes a standard pattern data storage unit **25** which stores a plurality of different standard pattern data **PA1** through **PAn**, **PB1** through **PBn**, . . . , **PM1** through **PMn** with respect to respective voice commands.

As shown in FIG. **8**, the analog processor **21** generally comprises an amplifier **30** for amplifying a voice command signal transmitted from the microphone M to a suitable level, a filter bank **31** for dividing an amplifier output signal into signals in different frequency bands and rectifying and outputting the signals in these different frequency bands, an analog-to-digital converter assembly (hereinafter referred to as an "A/D converter assembly") **32** for converting the output signals in the different frequency bands from the filter bank **31** into digital signals, and an interface **33** for transmitting signals to and receiving signals from the speech recognition processor **22**.

As shown in FIG. **9**(*a*), the filter bank **31** comprises a bandpass filter assembly **35** for dividing the input voice signal into signals in a plurality of frequency bands (four frequency bands in FIG. **9**(*a*)), a rectifier assembly **36** for rectifying output signals from the bandpass filter assembly **35**, and a low-pass filter assembly **37** for removing ripples from output signals from the rectifier assembly **36**.

The bandpass filter assembly **35** comprises a plurality of (four in FIG. **9**(*a*)) bandpass filters **PBFO** through **BPF3** which have respective central frequencies **f0**, **f1**, **f2**, **f3** (**f0**<**f1**< **f2**<**f3**) corresponding to the frequency bands.

The rectifier assembly **36** comprises four rectifiers **RCTO** through **RCT3** connected in series with the bandpass filters **BPF0** through **BPF3** of the bandpass filter assembly **35**, respectively. The rectifiers **RCT0** through **RCT3** rectify the output signals from the band pass filters **BPF0** through **BPF3** in the respective frequency bands.

The low-pass filter assembly comprises four low-pass filters **LPF0** through **LPF3** connected in series with the rectifiers **RCT0** through **RCT3** of the rectifier **36**, respectively. The low-pass filters **LPF0** through **LPF3** remove ripples from the rectified signals in the respective frequency bands.

The A/D converter assembly **32** comprise four A/D converters **ADC0** through **ADC3** connected in series with the low-pass filters **LPF0** through **LPF3** of the low-pass filter assembly **37**, respectively. The A/D converters **ADC0** through **ADC3** convert the analog output signals from the low-pass filters **LPF0** through **LPF3** into digital signals.

Operation of the analog processor **21** will be described below. For the sake of brevity, only signal processing in one frequency band (e.g., through the band pass filter **PBF3**) will be described. However, similar signal processing is carried out in the other frequency bands.

When a voice command is applied to the microphone M, the output electric signal from the microphone M is amplified to a suitable signal level by the amplifier **30**, which outputs an amplified signal A (see FIG. **9**(*b*)). The amplified signal A is applied to the band passes filter **PBF3**, which then passes only a signal B in its passband. The signal B is then applied to the rectifier **RCT3** (see FIG. **9**(*c*)). The signal B is rectified by the rectifier **RCT3**, and a rectified output signal C (FIG. **9**(*d*)) from the rectifier **RCT3** is transmitted to the low-pass filter **LPF3**. The low-pass filter **LPF3** removes ripples which may be contained in the signal C, and produces a ripple-free output signal D (see FIG. **9**(*e*)) which is then inputted to the A/D converter **ACD3**. The A/D converter **ADC3** then converts the supplied input signal D into a signal E composed of 4-bit time-division digital data (**1010**), (**0111**), (**0101**), (**0111**), (**1101**), . . . , as shown in FIG. **9**(*f*).

As illustrated in FIG. **8**, the speech recognition processor **22** comprises a system controller **40** for analyzing and processing control commands from the controller **16** and also for controlling the entire operation of the speech recognition processor **22**, and a digital processor **41** for effecting distance calculations and controlling the memory **23A**.

The system controller **40** comprises a CPU (Central Processing Unit) **42** for controlling the overall operation of the transmitter **1**, a ROM (Read-Only Memory) **43** for storing a control program to be executed by the CPU **42** for the overall operation of the transmitter **101**, a RAM (Random-Access Memory) **44** for temporarily storing data, and an interface **45** for transmitting data to and receiving data from the analog processor **21** and the digital processor **41**.

The digital processor **41** comprises an arithmetic unit **46** for effecting distance calculations and identifying input voice commands based on the results of the distance calculations, a data RAM **47** for storing data necessary for distance calculations, a ROM **48** for storing a program for distance calculations, a working RAM **49** for temporarily storing processed data, an interface **50** for transmitting data to and receiving data from the analog processor **21** and the system controller **40**, and an interface **51** for transmitting data to and receiving data from the memory **23A**.

The speech recognition processor **22** operates as follows: When a control command is applied from the controller **16** through the interface **24** to the speech recognition processor **22**, the system controller **40** receives the control command through the interfaces **50**, **45** and analyzes the received control command. If the result of analysis indicates a speech recognition process, the system controller **40** sends an instruction for speech recognition to the digital processor **41** through the interfaces **45**, **50**.

When instructed by the system controller **40**, the digital processor **41** introduces time-division digital data (input voice command signal) **20** from the analog processor **21** through the interface **50** into the data RAM **47**. The arithmetic unit **46** reads the standard pattern data from the first address in the memory **23A** which stores the different standard pattern data **PA1** through **PAn**, . . . , **PM1** through **PMn**, through the interface **51**. Then, the arithmetic unit **46** determines the logarithm of the first time-division digital data of a plurality of time-division digital data which constitute one of the read standard pattern data and also the logarithm of the first time-division digital data of the input voice command signal, and then determines the differences between the logarithms. The arithmetic unit 46 further squares the differences, and adds the squares to determine a distance D. Therefore, the distance D is given by:

$$D = \sum_{t=0}^{x} (\log(f(t)) - \log(fs(t)))^2$$

where

x: the number of time divisions;

f(t): the input voice command data (time-division digital data); and

fs(t): the standard pattern data (time-division digital data).

Likewise, the distances D are calculated in the same manner for all the standard pattern data. The smaller the calculated distances, the higher the probability that the standard pattern data are similar to the voice command. The recognition results thus obtained are collected for each of the voice commands. Then, command data corresponding to the voice command to which the standard pattern data are most similar as a whole are outputted as command data from the speech recognition circuit **15A** through the interface **24** to the controller **16A**.

Referring back to FIG. **6**, the controller **16A** is in the form of a microprocessor, for example. The microprocessor of the controller **16A** comprises a CPU, a ROM, a RAM, and an interface. The CPU executes arithmetic operations while referring to data stored in the RAM, which serves as a working memory, according to the algorithm (see FIG. **10**) of a control program stored in the ROM, for thereby effecting the overall operation of the transmitter **10A**. The controller **16A** also receives signals from the talk switch **12** and the mode selector switch **13** as interrupt signals, and effects control functions according to commands indicated by these interrupt signals. Operation of the transmitter **10A** under the control of the controller **16A** will be described below.

The controller **16A** implements the recognition accuracy determining unit **4**, and the selector **6** according to a program. Specifically, the degree-of-importance determining unit **5** and the selector **6** are implemented by a step **S17** in FIG. **10**, and the recognition accuracy determining unit **4** is implemented by steps **S16**, **S18**, **S19** in FIG. **10**.

Overall Operation

The transmitter **10A** operates depending on whether the talk switch **12** is pressed or released (i.e., turned on or off). If the talk switch **12** is pressed, the transmitter **10A** is capable of transmitting remote control signals, and if the talk switch **12** is released, the transmitter **10A** is kept in the low power consumption mode, waiting for voice commands to be applied. There are two input modes for entering voice commands. In one input mode, voice commands of the operator are registered, and in the other input mode, voice commands of the operator are recognized. In the voice registration mode, a command word such as for "reproduction" is recorded in the transmitter **10A**.

Now, operation of the transmitter **10A** will be described below with reference to the flowchart of FIG. **10**. It is assumed that the talk switch **12** is not pressed and the transmitter **10A** is in a standby condition in the low power consumption mode.

First, the controller **16A** initializes a registration buffer number to **1** in a step **S1**.

Then, the controller **16A** detects whether the talk switch **12** is pressed or not in a step **S2** by detecting whether there is produced an operation control signal Sc from the talk switch **12** or not. If the talk switch **12** is pressed, it produces an operation control signal Sc, and the controller **16A** sends a control signal Sv to the power supply control circuit **14**. The power supply control circuit **14** supplies electric energy in a normal mode, enabling the transmitter **10A** in a step **S4**.

If the talk switch **12** is not pressed, the transmitter **1** is left in the low power consumption mode, and the steps **S2** and **S3** are repeated.

Thereafter, the controller **16A** reads the condition of the mode selector switch **13** to determine whether or not it indicates the speech registration mode for voice commands to generate standard pattern data in a step **S5**.

If the speech registration mode is indicated, control then goes to a step **S6** in which the controller **16A** outputs a command to instruct the speech recognition circuit **15A** to carry out a speech registration process. At the same time, the controller **16A** sends a registration buffer number to the speech recognition circuit **15A** in the step **S6**.

The speech recognition circuit **15A** then stored speech recognition standard pattern data in a corresponding registration buffer in the memory **23A**, i.e., a registration buffer having the registration buffer number =1, in a step **S7**.

The controller **16A** reads a status register (not shown) in the speech recognition circuit **15A** to determine whether the registration of a voice command is finished or not in a step **S8**. If the registration is not yet finished, then the steps **S7** and **S8** are repeated until the registration is finished. If the registration is finished, the registration buffer number is incremented by 1 in a step **S9**.

Then, the controller **16A** determines whether the current registration buffer number has exceeded a maximum number Nmax that can be registered or not in a step **S10**. If not, then control returns to the step **S2**. If exceeded, the controller **16A** sends a command to cancel the speech registration mode to the speech recognition circuit **15A**, thereby canceling the speech registration mode in a step **S11**. Then, control goes back to the step **S2**.

If the speech registration mode is not indicated by the mode selector switch **13** in the step **S5**, i.e., if the speech recognition mode is indicated by the mode selector switch **13** in the step **S5**, then the controller **16A** outputs a speech recognition command to the speech recognition circuit **15A** in a step **S12**. The speech recognition circuit **15A** now effects a speech recognition process as described above in a step **S13**.

The controller **16A** reads a status register (not shown) in the speech recognition circuit **15A** to determine whether the speech recognition is finished or not in a step **S14**. If the speech recognition is not yet finished, then the steps **S13** and **S14** are repeated until the speech recognition is finished. If the speech recognition is finished, then the controller **16A** determines whether the input voice command data and the standard pattern data coincide with each other, i.e., the distance D falls within a predetermined distance, or not in a step **S15**. If the input voice command data and the standard pattern data coincide with each other, then the controller **16A** counts the number of standard pattern data which agree with the recognition result, among all the standard pattern data corresponding to the recognition result, in a step **S16**. The count is referred to as a pattern count which is used as a level to determine the accuracy of recognition.

Then, the controller **16A** determines whether the recognition result indicates an important operation or not in a step **S17**. If not, the controller **16A** determines whether or not the pattern count is equal to or greater than a predetermined second count in a step **S18**. The second count corresponds to a condition in which the same recognition result is obtained using more than half standard pattern data, and represents a relatively low accuracy of recognition. If the pattern count is lower than the second count, then since the input voice command cannot be recognized due to the low recognition accuracy, control goes to a step **S22** for an error process. If the pattern count is equal to or greater than the second count, then since the recognition accuracy is relatively high and the recognition process is reliable, the controller **16A** produces a remote control instruction signal SR based on the recognized voice command data and sends the remote control instruction signal SR to the transmitting circuit **17** in a step **S20**.

If the recognition result indicates an important operation in the step **S17**, then the controller **16A** determines whether or not the pattern count is equal to or greater than a predetermined first count in a step **S19**. The first count represents a relatively high degree of recognition and corresponds to a condition in which the same recognition result is obtained using 90 percent or more of the standard pattern data. For control commands of high importance, in order to maintain control reliability, no remote control signal RC is generated unless the condition of the step **S19** is met.

The controller **16A** produces a remote control instruction signal SR based on the recognized voice command data of high recognition accuracy and sends the remote control instruction signal SR to the transmitting circuit **17** in the step **S20**. In response to the remote control instruction signal ST, the transmitting circuit **17** transmits a corresponding remote control signal RC in a step **S21**. If the input voice command data and the standard pattern data do not coincide with each other in the step **S15**, or the pattern count is smaller than the second count in the step **S18**, or smaller than the first count in the step **S19**, then the controller **16A** effects an error process such as the generation of a buzzer sound in the step **S22**, and control goes back to the step **S2**.

In the first embodiment of the present invention, as described above one voice command is recognized using a plurality of different standard pattern data PA **1** through PAn, PB1 through PBn, . . . PM1 through PMn for respective voice commands, and if the recognition result indicates an important operation, then the recognition accuracy is confirmed to lower the erroneous recognition rate.

Second Embodiment

A voice-operated remote control system according to a second embodiment of the present invention will be described below with reference to FIGS. **11** through **13**.

External Structure

The voice-operated remote control system according to the second embodiment has a transmitter which has the same external structure as that of the transmitter **10A** shown in FIG. **4**.

Electronic Circuit Structure

The transmitter, generally indicated at **10B**, of the voice-operated remote control system according to the second embodiment is shown in FIGS. **11** and **12**. Those parts shown in FIG. **11** which are identical to those shown in FIG. **6** are denoted by identical reference numerals, and will not be described in detail.

The transmitter **10B** has a speech recognition circuit **15B** which, as shown in FIG. **12**, comprises an analog processor **21** for processing an analog voice command signal which is received through the microphone M and outputting the processed analog voice command signal as a time-division digital data **20**, a plurality of parallel speech recognition processors **22-1** through **22**-n for independently recognizing the voice command based on the time-division digital data **20** from the analog processor **21**, a memory **23B** for storing standard pattern data for speech recognition by the speech recognition processors **22-1** through **22**-n, respectively, and an interface **24** for transmitting signals to and receiving signals from a controller **16B**.

The speech recognition processors **22-1** through **22**-n use respective standard pattern data with respect to each voice command, and effect independent speech recognition processes. For example, the speech recognition processor **22-1** uses standard pattern data PA1, PB1, . . . PM1, and the speech recognition processor **22-2** uses standard pattern data PA2, PB2, . . . PM2.

The memory **23B** has address areas allotted to the respective speech recognition processors **22-1** through **22**-n, and stores standard pattern data PA1 through PM1, PA2 through PM2, . . . , PAn through PMn.

These standard pattern data may be stored in different areas in the memory **23B** as described above, or may be stored in respective memories associated with the respective speech recognition processors.

The controller **16B** has the functions of the recognition accuracy determining unit **4**, the degree-of-importance determining unit **5**, and the selector **6** according to the first embodiment. More specifically, if the control command indicated by the recognition result giving by the speech recognition circuit **15B** is determined to be of high importance, then the controller **16B** collects the recognition result of the speech recognition processors **22-1** through **22**-n, and outputs only the recognition result of high recognition accuracy as command data. If any recognition result which meets the degree of importance of the control command is not obtained, then the controller **16B** not output any command data. The recognition accuracy determining unit **4** is implemented by steps **S50**, **S51** in FIG. **13**, and the degree-of-importance determining unit **5** and the selector **6** are implemented by a step **S49** in FIG. **13**.

The analog processor **21**, the speech recognition processors **22-1** through **22**-n, and the interface **24** shown in FIG. **12** and the other structural details shown in FIG. **11** are identical to those according to the first embodiment.

Overall Operation

The transmitter **10B** according to the second embodiment operates as follows: It is assumed that the talk switch **12** is not pressed and the transmitter **10B** is in a standby condition.

First, the controller **16B** initializes a registration buffer number to 1 in a step **S31** (FIG. **13**). The processor number is set initialized to 1, and the speech recognition processor corresponding to that processor number, e.g., the speech recognition processor **22-1**, is enabled in a step S32.

Then, the controller **16B** detects whether the talk switch **12** is pressed or not in a step **S33** by detecting whether there is produced an operation control signal Sc from the talk switch **12** or not. If the talk switch **12** is pressed, then an operation control signal Sc is produced, and the controller **16B** enables the transmitter **10B** to receive a voice command.

Thereafter, the controller **16B** reads the condition of the mode selector switch **13** to determine whether it indicates the speech registration mode for voice commands to generate standard pattern data or not in a step S34.

If the speech registration mode is indicated, control then goes to a step **S35** in which the controller **16B** outputs a command to instruct the speech recognition circuit **15B** to carry out a speech registration process. At the same time, the controller **16B** sends a registration buffer number to the speech recognition circuit **15B** in the step S35.

The speech recognition circuit **15B** then stores speech recognition standard pattern data in a corresponding registration buffer for the speech recognition processor in question in the memory **23B**, i.e., a registration buffer having the registration buffer number = 1, in a step S36.

The controller **16B** reads a status register (not shown) in the speech recognition circuit **15B** to determine whether the registration of a voice command is finished or not in a step S37. If the registration is not yet finished, then the steps S36 and S37 are repeated until the registration is finished. If the registration is finished, the registration buffer number is incremented by 1 in a step S38.

Then, the controller **16B** determines whether or not the current registration buffer number has exceeded a maximum number Nmax that can be registered for the speech recognition processor in question, in a step S39. If not, then control returns to the step S32. If exceeded, the controller **16B** determines whether the processor number has exceeded a maximum speech recognition processor number Pmax or not in a step **S40**. If not exceeded, then control proceeds to a step **S41**.

In the step **S41**, the controller **16B** sets the registration buffer number to 1 again. Thereafter, the processor number is incremented by 1 and the speech recognition processor corresponding to the processor number, e.g., the speech recognition processor **22-2**, is enabled in a step **S42**.

If the processor number has exceeded the maximum speech recognition processor number Pmax in the step **S40**, the controller **16B** sends a command to cancel the speech registration mode to the speech recognition circuit **15B**, thereby canceling the speech registration mode in a step **S43**. Then, control goes back to the step S32.

If the speech recognition mode is indicated by the mode selector switch **13** in the step **S34**, then the controller **16B** enables all the speech recognition processors in a step **S44**. Then, the controller **16B** outputs a speech recognition command to the speech recognition circuit **15B** in a step **S45**. The speech recognition processors **22-1** through **22**-n in the speech recognition circuit **15B** now effect a speech recognition process as described above in a step **S46**.

The controller **16B** reads a status register (not shown) in the speech recognition circuit **15B** to determine whether the speech recognition by the speech recognition processors **22-1** through **22**-n is finished or not in a step **S47**. If the speech recognition is not yet finished, then the steps **S46** and **S47** are repeated until the speech recognition is finished. If the speech recognition is finished, then the controller **16** counts the number of speech recognition processors whose recognition results coincide in a step **S48**. The count is referred to as a processor count which is used as a level to determine the accuracy of recognition.

Then, the controller **16B** determines whether the recognition result indicates an important operation or not in a step **S49**. If not, the controller **16B** determines the accuracy of recognition, i.e., whether or not the processor count is equal to or greater than a predetermined second count in a step **S50**. The second count corresponds to a condition in which the same recognition result is obtained by more than half speech recognition processors, and represents a relatively low accuracy of recognition. If the processor count is lower than the second count, then since the input voice command cannot be recognized due to the low recognition accuracy, control goes to a step **S54** for an error process. If the processor count is equal to or greater than the second count, then since the recognition accuracy is relatively high and the recognition process is reliable, the controller **16B** produces a remote control instruction signal SR based on the recognized voice command data and sends the remote control instruction signal SR to the transmitting circuit **17** in a step **S52**.

If the recognition result indicates an important operation in the step **S49**, then the controller **16B** determines whether or not the processor count is equal to or greater than a predetermined first count in order to determine the accuracy of recognition in a step **S51**. The first count represents a relatively high degree of recognition and corresponds to a condition in which the same recognition result is obtained using 90 percent or more of the speech recognition processors. For control commands of high importance, in order to maintain control reliability, no remote control signal RC is generated unless the condition of the step **S51** is met.

The controller **16B** produces a remote control instruction signal SR based on the recognized voice command data of high recognition accuracy and sends the remote control instruction signal SR to the transmitting circuit **17** in the step **S52**. In response to the remote control instruction signal SR, the transmitting circuit **17** transmits a corresponding remote control signal RC in a step **S53**.

If the processor count is lower than the second count in the step **S50**, or lower than the first count in the step **S51**, then the controller **16B** effects an error process such as the generation of a buzzer sound in the step **S54**, and control goes back to the step **S32**.

In the second embodiment, an input voice command is recognized by the plural speech recognition processors **22-1** through **22**-n based on their respective standard pattern data **PA1** through **PM1**, **PA2** through PM **2**, . . . PAn through PMn. If the recognition result indicates an important operation, the accuracy of recognition is determined to lower the erroneous recognition rate.

Since the accuracy of recognition of a voice command which is of higher importance is confirmed, the recognition rate of the voice-operated remote control system is increased, and any trouble due to erroneous recognition is minimized.

The invention may be embodied in other specific forms without departing from the spirit or essential characteristics thereof. The present embodiments are therefore to be considered in all respects as illustrative and not restrictive, the scope of the invention being indicated by the appended claims rather than by the foregoing description and all changes which come within the meaning and range of equivalency of the claims are therefore intended to be embraced therein.

What is claimed is:

1. A voice-operated remote control system comprising:

a microphone for entering a voice command;

speech recognition means for comparing a pattern of the entered voice command with a predetermined standard pattern to recognize contents of the voice command and for producing command data which corresponds thereto, said command data being representative of an operation to be carried out;

transmitting means for generating and transmitting a remote control signal corresponding to the command data;

degree-of-importance determining means for determining a degree of importance of said command data, including command data being representative of at least important operations and not important operations, and for producing a corresponding degree-of-importance signal; and

recognition accuracy determining means for determining an accuracy of recognition of the command data depending on the corresponding degree of importance thereof as indicated by said degree-of-importance signal, and for sending to said transmitting means only command data that has been determined as being representative of important operation and having an accuracy of recognition exceeding a first predetermined threshold and for sending command data that has been determined as being representative of not important operations and having an accuracy of recognition exceeding a second predetermined threshold.

2. A voice-operated remote control system according to claim **1**, wherein said recognition accuracy determining means has a predetermined threshold for determining the accuracy of recognition for each degree of importance determination of the operation corresponding to the command data.

3. A voice-operated remote control system according to claim **1**, wherein said speech recognition means employs a plurality of standard patterns to be compared with one voice command, and said recognition accuracy determining means has a level for determining the accuracy of recognition, said level being represented by a number of standard patterns which substantially coincide with a pattern of a voice command entered through said microphone.

4. A voice-operated remote control system according to claim **3**, wherein said speech recognition means comprises a memory for storing the standard patterns in respective storage areas thereof, and a speech recognition processor for reading the standard patterns corresponding to the entered voice command from said memory and recognizing the voice command based on the standard patterns read from said memory.

5. A voice-operated remote control system according to claim **3**, wherein said speech recognition means comprises a plurality of memories for storing the standard patterns respectively, and a plurality of speech recognition processors, connected respectively to said memories, for reading the standard patterns corresponding to the entered voice command from said memory and recognizing the voice command based on the standard patterns read from said memory.

* * * * *

IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

IPA TECHNOLOGIES INC., Plaintiff, v. DISH NETWORK CORP., and DISH NETWORK LLC, Defendants.	C.A. No. ________________ **JURY TRIAL DEMANDED**

COMPLAINT FOR PATENT INFRINGEMENT

Plaintiff IPA Technologies Inc. ("IPA") as and for its complaint against DISH Network Corp. and DISH Network LLC (collectively, "Defendants") alleges as follows:

PARTIES

1. IPA is a Delaware corporation with a principal place of business at 600 Anton Blvd., Suite 1350, Costa Mesa, California 92626.

2. On information and belief, Defendant DISH Network Corp. is a Nevada corporation with a principal place of business at 9601 South Meridien Blvd., Englewood, Colorado 80112. DISH Network Corp. can be served with process pursuant to the Delaware Long Arm Statute, 10 *Del. C.* § 3104.

3. On information and belief, Defendant DISH Network LLC is a Colorado limited liability company with a principal place of business at 9601 South Meridien Blvd., Englewood, Colorado 80112. DISH Network LLC can be served with process pursuant to the Delaware Long Arm Statute, 10 *Del. C.* § 3104.

JURISDICTION AND VENUE

4. This action arises under the patent laws of the United States, Title 35 of the United States Code. Accordingly, this Court has subject matter jurisdiction under 28 U.S.C. §§ 1331 and 1338(a).

5. This Court has specific and general personal jurisdiction over Defendants pursuant to due process and/or the Delaware Long Arm Statute, due to Defendants' substantial business in this forum, including: (i) at least a portion of the infringement alleged herein; and (ii) regularly doing or soliciting business, engaging in other persistent courses of conduct, and/or deriving substantial revenue from goods and services provided to individuals in Delaware and in this Judicial District.

6. Venue is proper in this District under 28 U.S.C. §§ 1391 (b)-(c) and 1400(b) because Defendants are subject to personal jurisdiction in this District.

BACKGROUND

7. SRI International, Inc. ("SRI"), the original owner of the patents-in-suit, is an independent, not-for-profit research institute that conducts client-supported research and development for government agencies, commercial businesses, foundations, and other organizations.

8. Among its many areas of research, SRI has engaged in fundamental research and development related to personal digital assistants and speech-based navigation of electronic data sources.

9. SRI's innovative work on personal digital assistants was a key area of development in one of the world's largest artificial intelligence projects, the Cognitive Assistant that Learns and Organizes ("CALO"). The vision for the SRI-led CALO

project, which was funded by the U.S. Defense Advanced Research Projects Agency ("DARPA"), was to create groundbreaking software that could revolutionize how computers support decision-makers.

10. SRI's work on personal digital assistants and speech-based navigation of electronic data sources, which started before the launch of the CALO project, developed further as part of the project. SRI's engineers were awarded numerous patents on their groundbreaking personal digital assistant and speech-based navigation inventions.

11. To bring the personal digital assistant and speech-based navigation technology to the marketplace, SRI formed the spin-off company Siri, Inc. in 2007, and granted it a non-exclusive license to the patent portfolio. The technology was demonstrated as an iPhone app at technology conferences and later released as an iPhone 3GS app in February 2010. In April 2010, Apple Inc. acquired Siri, Inc. In 2011, the Siri personal digital assistant was released as an integrated feature of the iPhone 4S.

12. Speech-based navigation of electronic data sources has continued to be implemented as an effective and user-friendly solution for interacting with electronic devices.

13. On May 6, 2016, IPA acquired the SRI speech-based navigation patent portfolio. IPA is a wholly-owned subsidiary of WiLAN, a leading technology innovation and licensing business actively engaged in research, development, and licensing of new technologies.

ASSERTED PATENTS

14. IPA is the owner by assignment of U.S. Patent No. 6,742,021 (the "'021 Patent"). The '021 Patent is entitled "Navigating Network-Based Electronic Information

Using Spoken Input With Multimodal Error Feedback." The '021 Patent issued on May 25, 2004. A true and correct copy of the '021 Patent is attached hereto as Exhibit A.

15. IPA is the owner by assignment of U.S. Patent No. 6,523,061 (the "'061 Patent"). The '061 Patent is entitled "System, Method, and Article of Manufacture For Agent-Based Navigation in a Speech-Based Data Navigation System." The '061 Patent issued on February 18, 2003. A true and correct copy of the '061 Patent is attached hereto as Exhibit B.

16. IPA is the owner by assignment of U.S. Patent No. 6,757,718 (the "'718 patent"). The '718 Patent is entitled "Mobile Navigation of Network-Based Electronic Information Using Spoken Input." The '718 Patent issued on June 29, 2004. A true and correct copy of the '718 Patent is attached hereto as Exhibit C.

COUNT I
(Infringement of U.S. Patent No. 6,742,021)

17. Plaintiff re-alleges and incorporates by reference the allegations in the foregoing paragraphs as if fully set forth herein.

18. Plaintiff is informed and believes, and on that basis alleges, that Defendants have infringed and are currently infringing one or more claims (*e.g.*, claim 1) of the '021 Patent, in violation of 35 U.S.C. § 271.

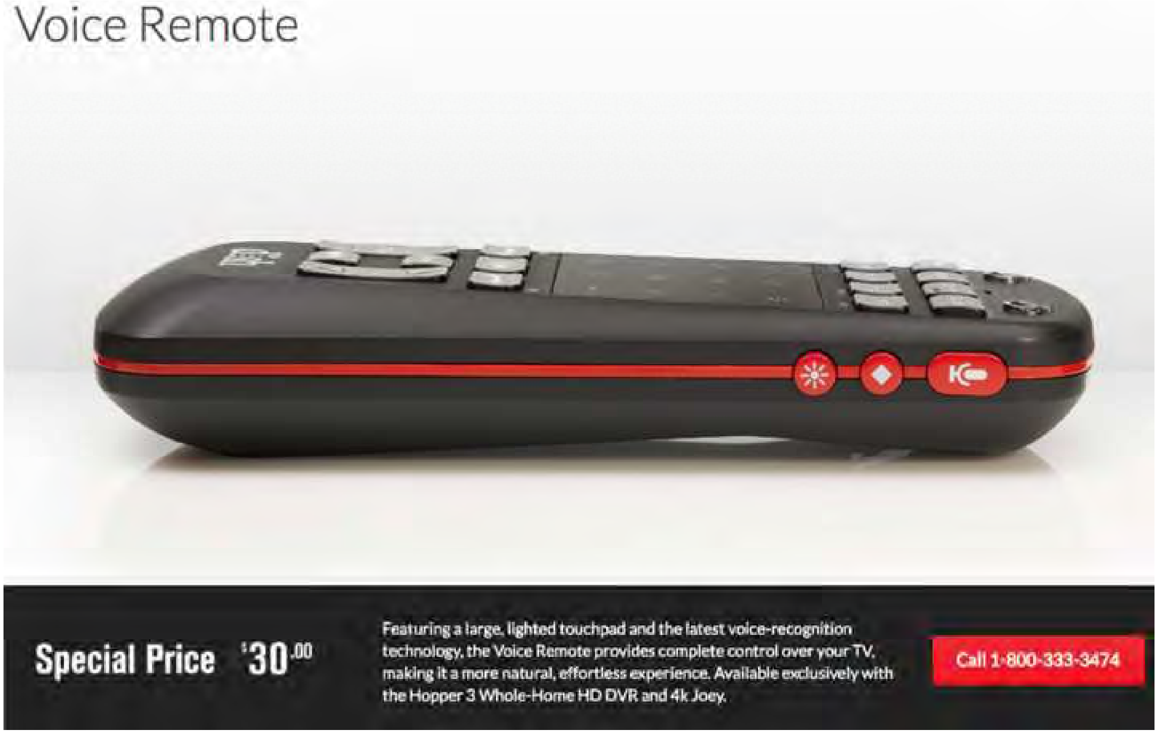
19. Defendants have infringed and are currently infringing literally and/or under the doctrine of equivalents, by, among other things, making, using, offering for sale, selling, and/or importing within this judicial district and elsewhere in the United States, without license or authority, infringing products, including but not limited to Voice Remote with Hopper 3 / 4k Joey set-top box products, and related products and/or

processes falling within the scope of one or more claims of the ’021 Patent, including claims 1 and 27. Exemplary claim 1 is reproduced below:

> A method for speech-based navigation of an electronic data source, the electronic data source being located at one or more network servers located remotely from a user, comprising the steps of:
>
> (a) receiving a spoken request for desired information from the user;
>
> (b) rendering an interpretation of the spoken request;
>
> (c) constructing at least part of a navigation query based upon the interpretation;
>
> (d) soliciting additional input from the user, including user interaction in a non-spoken modality different than the original request without requiring the user to request said non-spoken modality;
>
> (e) refining the navigation query, based upon the additional input;
>
> (f) using the refined navigation query to select a portion of the electronic data source; and
>
> (g) transmitting the selected portion of the electronic data source from the network server to a client device of the user.

20. Defendants’ acts of making, using, offering for sale, selling, and/or importing infringing products, including but not limited to Voice Remote with Hopper 3 / 4k Joey set-top box products, and related products and/or processes satisfy, literally or under the doctrine of equivalents, each and every claim limitation, including but not limited to limitations of claims 1 and 27. For example, Defendants’ Voice Remote with Hopper 3 / 4k Joey set-top box products use speech-based navigation of an electronic data source. The Voice Remote with Hopper 3 / 4k Joey set-top box products receive a spoken request for desired information from the user (such as a spoken request for particular television programming), render an interpretation of the spoken request, construct at least part of a navigation query based on the spoken request, solicit additional

input from the user, including user interaction in a non-spoken modality different than the original request without requiring the user to request the non-spoken modality, and transmit the selected portion from a network server to the Voice Remote with Hopper 3 / 4k Joey set-top box products, as described on the Voice Remote product page at http://dish.com:[1]

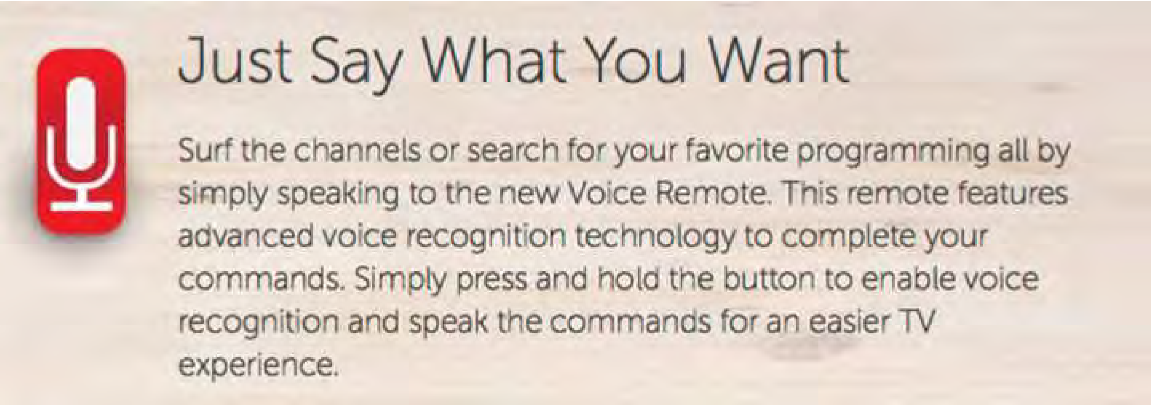


Voice Remote

Special Price $30.00

Featuring a large, lighted touchpad and the latest voice-recognition technology, the Voice Remote provides complete control over your TV, making it a more natural, effortless experience. Available exclusively with the Hopper 3 Whole-Home HD DVR and 4k Joey.

[1] Plaintiff reserves the right to identify additional asserted claims as this litigation proceeds. For example, Plaintiff expressly reserves the right to identify additional asserted claims in their infringement contentions to be served during the discovery process.



Just Say What You Want

Surf the channels or search for your favorite programming all by simply speaking to the new Voice Remote. This remote features advanced voice recognition technology to complete your commands. Simply press and hold the button to enable voice recognition and speak the commands for an easier TV experience.

21. Defendants have also infringed indirectly and continue to infringe indirectly the ’021 Patent by active inducement under 35 U.S.C. § 271(b).

22. On information and belief, Defendants gained knowledge of the ’021 Patent no later than the filing of this complaint or shortly thereafter.

23. On information and belief, Defendants have intended, and continue to intend, to induce patent infringement by its users and have had knowledge that the inducing acts would cause infringement or have been willfully blind to the possibility that their inducing acts would cause infringement. For example, Defendants encourage end users to perform speech-based navigation of an electronic data source using a system as claimed in claim 27 of the ’021 Patent through the very nature of the products. As a further example, Defendants instruct users on how to use the infringing products to perform speech-based navigation of an electronic data source using a system as claimed in claim 27 of the ’021 Patent (*e.g.*, “Voice Remote … Features,” *available at* https://www.mydish.com/voice-remote). By using the infringing products to perform speech-based navigation of an electronic data source, users directly infringe at least claim 27 of the ’021 Patent. By continuing to provide instructions to users on how to use the infringing products to perform speech-based navigation of an electronic data source using

a system as claimed in claim 27 of the ’021 Patent, and by continuing to encourage such use, Defendants have and continue to specifically intend to induce infringement of the ’021 Patent.

24. To the extent that facts learned in discovery show that Defendants’ infringement of the ’021 Patent is or has been willful, Plaintiff reserves the right to request such a finding at the time of trial.

25. To the extent applicable, the requirements of 35 U.S.C. § 287(a) have been met with respect to the ’021 Patent.

26. As a result of Defendants’ infringement of the ’021 Patent, Plaintiff has suffered monetary damages in an amount adequate to compensate for Defendants’ infringement, but in no event less than a reasonable royalty for the use made of the invention by Defendants, together with interest and costs as fixed by the Court, and Plaintiff will continue to suffer damages in the future unless Defendants’ infringing activities are enjoined by this Court.

27. Unless a permanent injunction is issued enjoining Defendants and their agents, servants, employees, representatives, affiliates, and all others acting or in active concert therewith from infringing the ’021 Patent, Plaintiff will be greatly and irreparably harmed.

COUNT II
(Infringement of U.S. Patent No. 6,523,061)

28. Plaintiff re-alleges and incorporates by reference the allegations in the foregoing paragraphs as if fully set forth herein.

29. Plaintiff is informed and believes, and on that basis alleges, that Defendants have infringed and are currently infringing one or more claims (*e.g.*, claim 1) of the ’061 Patent, in violation of 35 U.S.C. § 271.

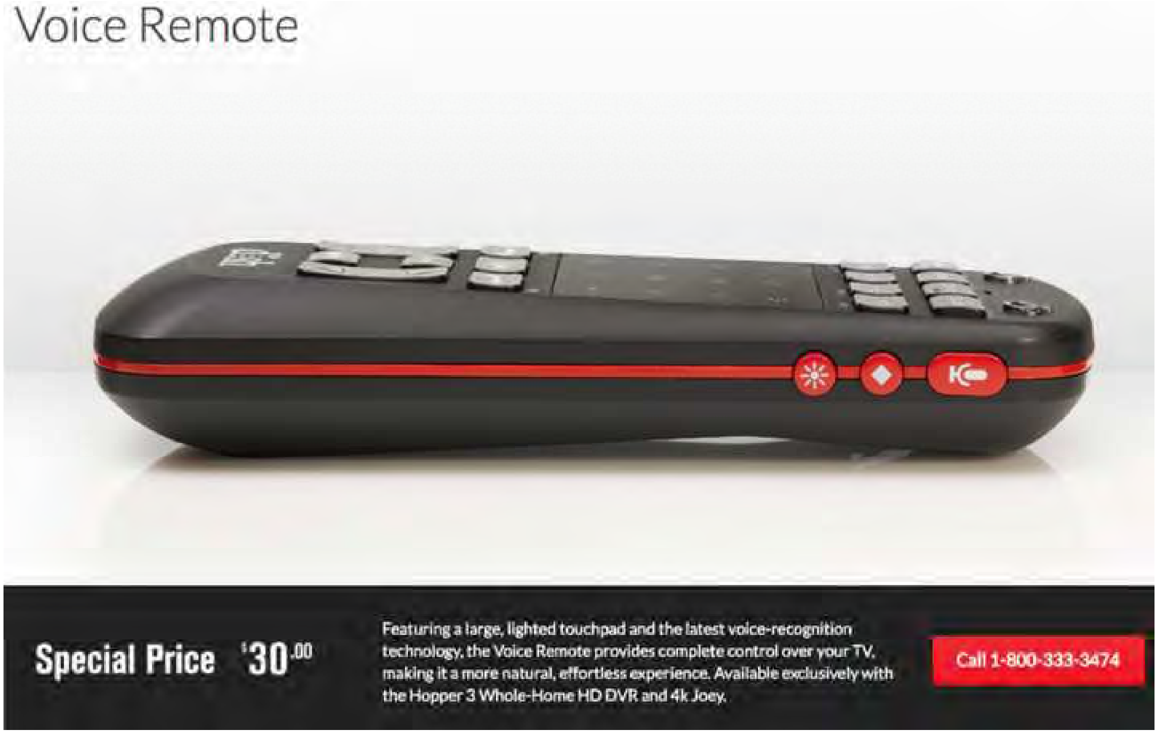
30. Defendants have infringed and are currently infringing literally and/or under the doctrine of equivalents, by, among other things, making, using, offering for sale, selling, and/or importing within this judicial district and elsewhere in the United States, without license or authority, infringing products, including but not limited to Voice Remote with Hopper 3 / 4k Joey set-top box products, and related products and/or processes falling within the scope of one or more claims of the ’061 Patent, including claims 1 and 13. Exemplary claim 1 reproduced below:

> A method for utilizing agents for speech-based navigation of an electronic data source, comprising the steps of:
>
> (a) receiving a spoken request for desired information from a user;
>
> (b) rendering an interpretation of the spoken request;
>
> (c) constructing a navigation query based upon the interpretation;
>
> (d) routing the navigation query to at least one agent, wherein the at least one agent utilizes the navigation query to select a portion of the electronic data source; and
>
> (e) invoking a user interface agent for outputting the selected portion of the electronic data source to the user, wherein a facilitator manages data flow among multiple agents and maintains a registration of each of said agents' capabilities.

31. Defendants’ acts of making, using, offering for sale, selling, and/or importing infringing products, including but not limited to Voice Remote with Hopper 3 / 4k Joey set-top box products, and related products and/or processes satisfy, literally or under the doctrine of equivalents, each and every claim limitation, including but not limited to limitations of claims 1 and 13. For example, Defendants’ Voice Remote with

Hopper 3 / 4k Joey set-top box products use speech-based navigation of an electronic data source. The Voice Remote with Hopper 3 / 4k Joey set-top box products receive a spoken request for desired information from the user (such as a spoken request for particular television programming), render an interpretation of the spoken request, constructs a navigation query based on the interpretation, route the navigation query to at least one agent that utilizes the navigation query to select a portion of the electronic data source, and invoke a user interface agent for outputting the selected portion of the electronic data source wherein a facilitator manages data flow among multiple agents and maintains a registration of each of the agents' capabilities, as described on the Voice Remote product page at http://dish.com:[2]

[2] Plaintiff reserves the right to identify additional asserted claims as this litigation proceeds. For example, Plaintiff expressly reserves the right to identify additional asserted claims in its infringement contentions to be served during the discovery process.

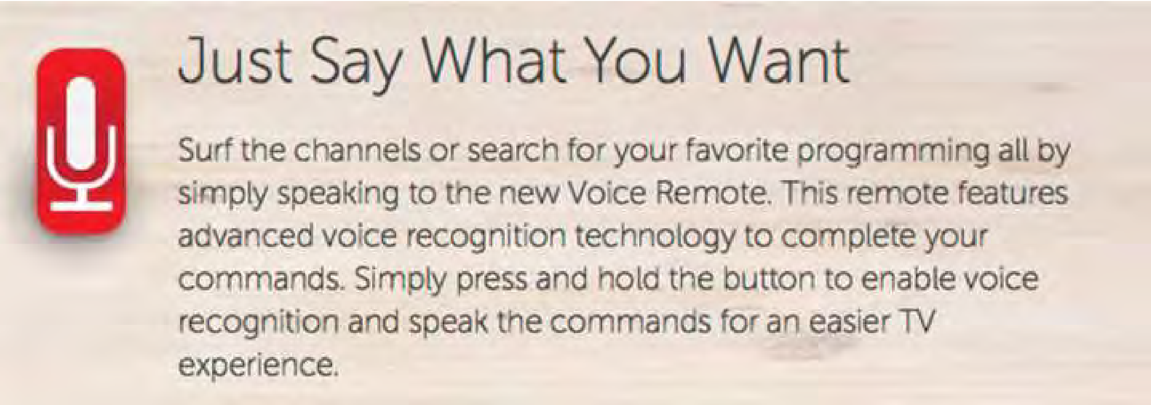


Voice Remote

Special Price $30.00

Featuring a large, lighted touchpad and the latest voice-recognition technology, the Voice Remote provides complete control over your TV, making it a more natural, effortless experience. Available exclusively with the Hopper 3 Whole-Home HD DVR and 4k Joey.

Call 1-800-333-3474



Just Say What You Want

Surf the channels or search for your favorite programming all by simply speaking to the new Voice Remote. This remote features advanced voice recognition technology to complete your commands. Simply press and hold the button to enable voice recognition and speak the commands for an easier TV experience.

32. Defendants have also infringed indirectly and continue to infringe indirectly the ’061 Patent by active inducement under 35 U.S.C. § 271(b).

33. On information and belief, Defendants gained knowledge of the ’061 Patent no later than the filing of this complaint or shortly thereafter.

34. On information and belief, Defendants have intended, and continue to intend, to induce patent infringement by its users and have had knowledge that the inducing acts would cause infringement or have been willfully blind to the possibility that

their inducing acts would cause infringement. For example, Defendants encourage end users to perform speech-based navigation of an electronic data source using a system as claimed in claim 13 of the ’061 Patent through the very nature of the products. As a further example, Defendants instruct users on how to use the infringing products to perform speech-based navigation of an electronic data source using a system as claimed in claim 13 of the ’061 Patent (*e.g.*, “Voice Remote … Features,” *available at* https://www.mydish.com/voice-remote). By using the infringing products to perform speech-based navigation of an electronic data source, users directly infringe at least claim 13 of the ’061 Patent. By continuing to provide instructions to users on how to use the infringing products to perform speech-based navigation of an electronic data source using a system as claimed in claim 13 of the ’061 Patent, and by continuing to encourage such use, Defendants have and continue to specifically intend to induce infringement of the ’061 Patent.

35. To the extent that facts learned in discovery show that Defendants’ infringement of the ’061 Patent is or has been willful, Plaintiff reserves the right to request such a finding at the time of trial.

36. To the extent applicable, the requirements of 35 U.S.C. § 287(a) have been met with respect to the ’061 Patent.

37. As a result of Defendants’ infringement of the ’061 Patent, Plaintiff has suffered monetary damages in an amount adequate to compensate for Defendants’ infringement, but in no event less than a reasonable royalty for the use made of the invention by Defendants, together with interest and costs as fixed by the Court, and

Plaintiff will continue to suffer damages in the future unless Defendants' infringing activities are enjoined by this Court.

38. Unless a permanent injunction is issued enjoining Defendants and their agents, servants, employees, representatives, affiliates, and all others acting or in active concert therewith from infringing the '061 Patent, Plaintiff will be greatly and irreparably harmed.

COUNT III
(Infringement of U.S. Patent No. 6,757,718)

39. Plaintiff re-alleges and incorporates by reference the allegations in the foregoing paragraphs as if fully set forth herein.

40. Plaintiff is informed and believes, and on that basis alleges, that Defendants have infringed and are currently infringing one or more claims (*e.g.*, claim 1) of the '718 Patent, in violation of 35 U.S.C. § 271.

41. Defendants have infringed and are currently infringing literally and/or under the doctrine of equivalents, by, among other things, making, using, offering for sale, selling, and/or importing within this judicial district and elsewhere in the United States, without license or authority, infringing products, including but not limited to Voice Remote with Hopper 3 / 4k Joey set-top box products, and related products and/or processes falling within the scope of one or more claims of the '718 Patent, including claims 1 and 19. Exemplary claim 1 is reproduced below:

> A method for speech-based navigation of an electronic data source located at one or more network servers located remotely from a user, wherein a data link is established between a mobile information appliance of the user and the one or more network servers, comprising the steps of:
>
> (a) receiving a spoken request for desired information from the user utilizing the mobile information appliance of the user, wherein said mobile information

appliance comprises a portable remote control device or a set-top box for a television;

(b) rendering an interpretation of the spoken request;

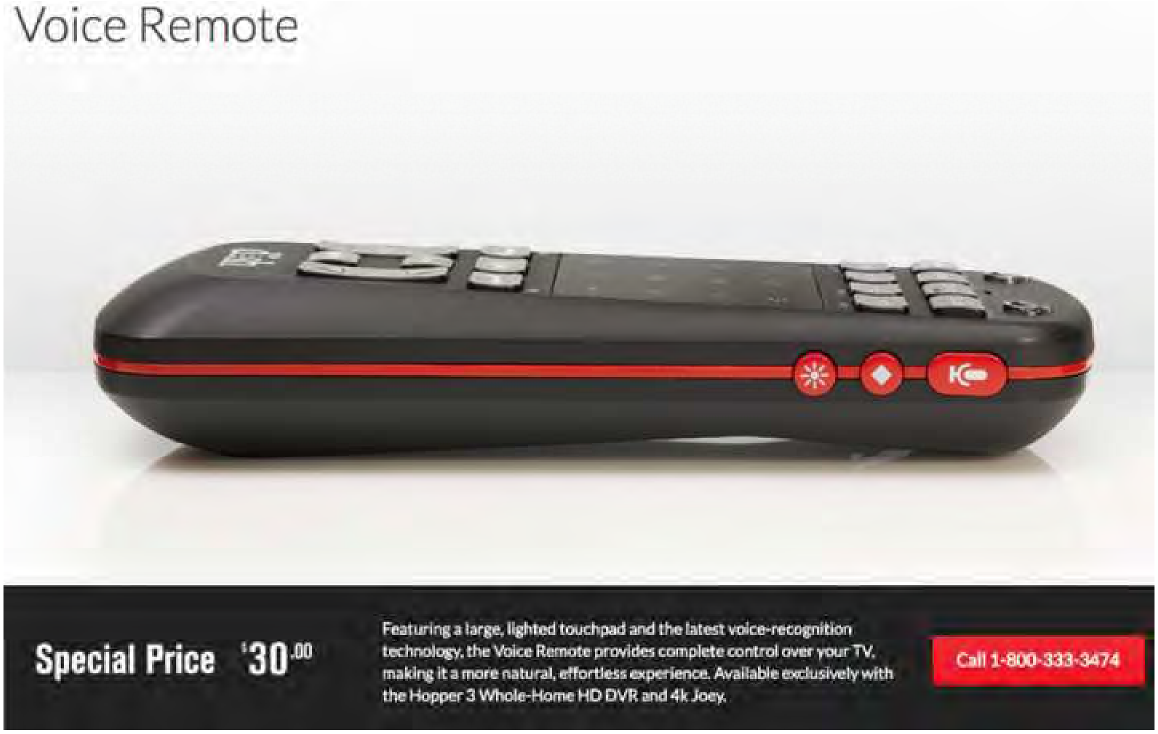
(c) constructing a navigation query based upon the interpretation;

(d) utilizing the navigation query to select a portion of the electronic data source; and

(e) transmitting the selected portion of the electronic data source from the network server to the mobile information appliance of the user.

42. Defendants' acts of making, using, offering for sale, selling, and/or importing infringing products, including but not limited to Voice Remote with Hopper 3 / 4k Joey set-top box products, and related products and/or processes satisfy, literally or under the doctrine of equivalents, each and every claim limitation, including but not limited to limitations of claims 1 and 19. For example, Defendants' Voice Remote with Hopper 3 / 4k Joey set-top box products use speech-based navigation and include a set-top box for a television and a portable remote control device. The Voice Remote with Hopper 3 / 4k Joey set-top box products receive a spoken request for desired information from the user (such as a spoken request for particular television programming), render an interpretation of the spoken request, constructs a navigation query, utilize the navigation query to select a portion of an electronic data source, and transmit the selected portion from a network server to the Voice Remote with Hopper 3 / 4k Joey set-top box products, as described on the Voice Remote product page at http://dish.com:[3]

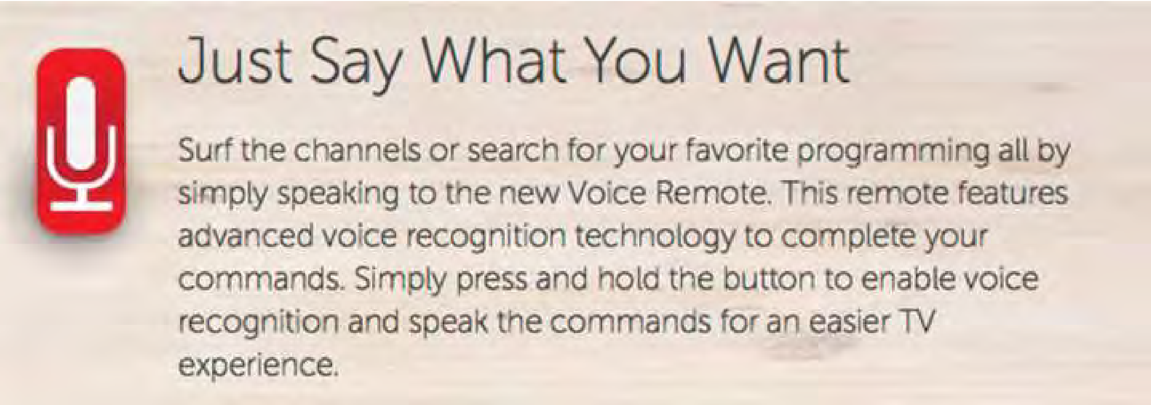
[3] Plaintiff reserves the right to identify additional asserted claims as this litigation proceeds. For example, Plaintiff expressly reserves the right to identify additional asserted claims in its infringement contentions to be served during the discovery process.



Voice Remote

Special Price $30.00

Featuring a large, lighted touchpad and the latest voice-recognition technology, the Voice Remote provides complete control over your TV, making it a more natural, effortless experience. Available exclusively with the Hopper 3 Whole-Home HD DVR and 4k Joey.



Just Say What You Want

Surf the channels or search for your favorite programming all by simply speaking to the new Voice Remote. This remote features advanced voice recognition technology to complete your commands. Simply press and hold the button to enable voice recognition and speak the commands for an easier TV experience.

43. Defendants have also infringed indirectly and continue to infringe indirectly the ’718 Patent by active inducement under 35 U.S.C. § 271(b).

44. On information and belief, Defendants gained knowledge of the ’718 Patent no later than the filing of this complaint or shortly thereafter.

45. On information and belief, Defendants have intended, and continue to intend, to induce patent infringement by its users and have had knowledge that the inducing acts would cause infringement or have been willfully blind to the possibility that

their inducing acts would cause infringement. For example, Defendants encourage end users to perform speech-based navigation of an electronic data source using a system as claimed in claim 19 of the ’718 Patent through the very nature of the products. As a further example, Defendants instruct users on how to use the infringing products to perform speech-based navigation of an electronic data source using a system as claimed in claim 19 of the ’718 Patent (*e.g.*, “Voice Remote … Features,” *available at* https://www.mydish.com/voice-remote). By using the infringing products to perform speech-based navigation of an electronic data source, users directly infringe at least claim 19 of the ’718 Patent. By continuing to provide instructions to users on how to use the infringing products to perform speech-based navigation of an electronic data source using a system as claimed in claim 19 of the ’718 Patent, and by continuing to encourage such use, Defendants have and continue to specifically intend to induce infringement of the ’718 Patent.

46. To the extent that facts learned in discovery show that Defendants’ infringement of the ’718 Patent is or has been willful, Plaintiff reserves the right to request such a finding at the time of trial.

47. To the extent applicable, the requirements of 35 U.S.C. § 287(a) have been met with respect to the ’718 Patent.

48. As a result of Defendants’ infringement of the ’718 Patent, Plaintiff has suffered monetary damages in an amount adequate to compensate for Defendants’ infringement, but in no event less than a reasonable royalty for the use made of the invention by Defendants, together with interest and costs as fixed by the Court, and

Plaintiff will continue to suffer damages in the future unless Defendants' infringing activities are enjoined by this Court.

49. Unless a permanent injunction is issued enjoining Defendants and their agents, servants, employees, representatives, affiliates, and all others acting or in active concert therewith from infringing the '718 Patent, Plaintiff will be greatly and irreparably harmed.

PRAYER FOR RELIEF

Plaintiff prays for the following relief:

1. A judgment that Defendants have infringed one or more claims of the '021, '061, and '718 Patents;

2. A permanent injunction enjoining Defendants and their officers, directors, agents, servants, affiliates, employees, divisions, branches, subsidiaries, parents, and all others acting in active concert or participation with Defendants, from infringing the '021, '061, and '718 Patents;

3. An award of damages resulting from Defendants' acts of infringement in accordance with 35 U.S.C. § 284;

4. A judgment and order finding that this is an exceptional case within the meaning of 35 U.S.C. § 285 and awarding to Plaintiff its reasonable attorneys' fees against Defendants.

5. A judgment and order requiring Defendants to provide accountings and to pay supplemental damages to Plaintiff, including, without limitation, prejudgment and post-judgment interest; and

6. Any and all other relief to which Plaintiff may show itself to be entitled.

JURY TRIAL DEMANDED

Plaintiff hereby demands a trial by jury of all issues so triable.

Dated: December 9, 2016

OF COUNSEL:

Marc A. Fenster
Brian Ledahl
Adam Hoffman
Amir Naini
Russ, August & Kabat
12424 Wilshire Boulevard, 12th Floor
Los Angeles, CA 90025-1031
(310) 826-7474
mfenster@raklaw.com
bledahl@raklaw.com
ahoffman@raklaw.com
anaini@raklaw.com

BAYARD, P.A.

/s/ Stephen B. Brauerman
Stephen B. Brauerman (No. 4952)
Sara E. Bussiere (No. 5725)
222 Delaware Avenue, Suite 900
P.O. Box 25130
Wilmington, Delaware 19801
(302) 655-5000
sbrauerman@bayardlaw.com
sbussiere@bayardlaw.com

Attorneys for Plaintiff IPA Technologies, Inc.

BAYARD

222 Delaware Avenue • Suite 900
P.O. Box 25130 • Wilmington, DE • 19899
Zip Code For Deliveries 19801

STEPHEN B. BRAUERMAN
Direct Dial: (302) 429-4232
Email: sbrauerman@bayardlaw.com

December 14, 2016

VIA CERTIFIED MAIL
RETURN RECEIPT REQUESTED

DISH Network LLC
9601 South Meridien Blvd.
Englewood, Colorado 80012

Re: *IPA Technologies Inc. v. DISH Network Corp., et al., C.A. No. 16-1170-RGA*

To Whom It May Concern:

In accordance with the provisions of 10 *Del. C.* § 3104, we enclose copies of the summons and the complaint which were filed in the above captioned action. Under the provisions of 10 *Del. C.* § 3104, service of the summons and the complaint via certified mail is as effectual to all intents and purposes as if it had been made upon DISH Network LLC personally within the State of Delaware.

We suggest you deliver these papers immediately to your attorney.

Sincerely,

Stephen B. Brauerman

SBB:jl
Enclosures
File No.: 38206-1

AO 440 (Rev. 12/09) Summons in a Civil Action

IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

IPA TECHNOLOGIES INC., Plaintiff, v. DISH NETWORK CORP., and DISH NETWORK LLC, Defendants.	C.A. No. 16-1170

Summons in a Civil Action

To: DISH Network LLC
9601 South Meridien Blvd.
Englewood, Colorado 80112

A lawsuit has been filed against you.

Within 21 days after service of this summons on you (not counting the day you received it) — or 60 days if you are the United States or a United States agency, or an officer or employee of the United States described in Fed. R. Civ. P. 12 (a)(2) or (3) — you must serve on the plaintiff an answer to the attached complaint or a motion under Rule 12 of the Federal Rules of Civil Procedure. The answer or motion must be served on the plaintiff or plaintiff's attorney, whose name and address are:

Stephen B. Brauerman
Bayard, P.A.
222 Delaware Avenue, Suite 900
Wilmington, Delaware 19801
302-655-5000 (phone)
302-658-6395 (fax)

If you fail to respond, judgment by default will be entered against you for the relief demanded in the complaint. You also must file your answer or motion with the court.

DEC 12 2016

DATE

DEPUTY CLERK'S SIGNATURE

Civil Action No.

PROOF OF SERVICE

(This section should not be filed with the court unless required by Fed. R. Civ. P. 4(l))

This summons for *(name of individual and title, if any)* ______________________________

was received by me on *(date)* ________________.

□ I personally served the summons on the individual at *(place)* ____________________

____ on *(date)* __________________; or

□ I left the summons at the individual's residence or usual place of abode with *(name)* __

____, a person of suitable age and discretion who resides there, on *(date)* ________________, and mailed a copy to the individual's last known address; or

□ I served the summons on *(name of individual)* ____________________________, who

is

designated by law to accept service of process on behalf of *(name of organization)*

____________________________________ on *(date)* ____________________; or

□ I returned the summons unexecuted because ______________________________; or

□ Other *(specify)*:

My fees are $ ________ for travel and $ ________ for services, for a total of $ ________.

I declare under penalty of perjury that this information is true.

Date: ____________________

____
Server's signature

____
Printed name and title

____
Server's address

Additional information regarding attempted service, etc:

Tracking Number: 70150640000594269121

Status

Your item was delivered to an individual at the address at 11:53 am on December 20, 2016 in ENGLEWOOD, CO 80112.

✓ Delivered

December 20, 2016 at 11:53 am
DELIVERED, LEFT WITH INDIVIDUAL
ENGLEWOOD, CO 80112

Delivered

FIRST • CLASS

CERTIFIED MAIL

7015 0640 0005 9426 9121

First

Rec'd
12/20/16

BAYARD

222 Delaware Avenue • Suite 900
P.O. Box 25130 • Wilmington, DE • 19899
Zip Code for Deliveries 19801

DISH Network LLC
9601 South Meridien Blvd.
Englewood, CO 80112

IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

IPA TECHNOLOGIES INC.,

Plaintiff,

v.

DISH NETWORK CORP., and DISH NETWORK LLC,

Defendants.

C.A. No. 16-1170

Summons in a Civil Action

To: DISH Network Corp.
9601 South Meridien Blvd.
Englewood, Colorado 80112

A lawsuit has been filed against you.

Within 21 days after service of this summons on you (not counting the day you received it) — or 60 days if you are the United States or a United States agency, or an officer or employee of the United States described in Fed. R. Civ. P. 12 (a)(2) or (3) — you must serve on the plaintiff an answer to the attached complaint or a motion under Rule 12 of the Federal Rules of Civil Procedure. The answer or motion must be served on the plaintiff or plaintiff's attorney, whose name and address are:

Stephen B. Brauerman
Bayard, P.A.
222 Delaware Avenue, Suite 900
Wilmington, Delaware 19801
302-655-5000 (phone)
302-658-6395 (fax)

If you fail to respond, judgment by default will be entered against you for the relief demanded in the complaint. You also must file your answer or motion with the court.

DEC 12 2016

DATE

DEPUTY CLERK'S SIGNATURE

Civil Action No.

PROOF OF SERVICE

(This section should not be filed with the court unless required by Fed. R. Civ. P. 4(l))

This summons for *(name of individual and title, if any)* ________________________________

was received by me on *(date)* ________________.

□ I personally served the summons on the individual at *(place)* ____________________

____ on *(date)* __________________; or

□ I left the summons at the individual's residence or usual place of abode with *(name)* __

____, a person of suitable age and discretion who resides there, on *(date)* ________________, and mailed a copy to the individual's last known address; or

□ I served the summons on *(name of individual)* ____________________________, who

is

designated by law to accept service of process on behalf of *(name of organization)*

____________________________________ on *(date)* ____________________; or

□ I returned the summons unexecuted because ______________________________; or

□ Other *(specify)*:

My fees are $ ________ for travel and $ ________ for services, for a total of $ ________.

I declare under penalty of perjury that this information is true.

Date: ____________________

____
Server's signature

____
Printed name and title

____
Server's address

Additional information regarding attempted service, etc:

Tracking Number: 70150640000594269145

Status

✓ **Delivered**

December 20, 2016 at 11:53 am
DELIVERED, LEFT WITH INDIVIDUAL
ENGLEWOOD, CO 80112

Your item was delivered to an individual at the address at 11:53 am on December 20, 2016 in ENGLEWOOD, CO 80112.

Delivered

FIRST • CLASS

7015 0640 0005 9426 9145

UNITED STATES POSTAGE
PITNEY BOWES
02 1P $ 008.62⁰
0000882371 DEC 14 2016
MAILED FROM ZIP CODE 19801

First

U.S. POSTAGE » PITNEY BOWES
ZIP 19801 $ 000.00⁰
02 1W
0001378651 DEC 16 2016

Rec'd 12/20/16

BAYARD

222 Delaware Avenue • Suite 900
P.O. Box 25130 • Wilmington, DE • 19899
Zip Code for Deliveries 19801

DISH Network Corp.
9601 South Meridien Blvd.
Englewood, CO 80112

08/316619

ISSUE CLASSIFICATION	
Class	Sl
395	2.-

UTILITY SERIAL NUMBER 08/316619	PATENT DATE MAR 19 1996	PATENT NUMBER 5500920

SERIAL NUMBER	FILING DATE	CLASS	SUBCLASS	GROUP ART
08/316,619	09/30/94	395	2.79	2308

5500920

APPLICANTS

JULIAN M. KUPIEC, CUPERTINO, CA.

CONTINUING DATA*******************
VERIFIED THIS APPLN IS A CON OF 08/126,170 09/23/93 ABN

12/2/94

FOREIGN/PCT APPLICATIONS**********
VERIFIED

12/2/94 none

FOREIGN FILING LICENSE GRANTED 11/04/94

Foreign priority claimed □ yes ☒ no 35 USC 119 conditions met □ yes ☒ no Verified and Acknowledged 12/2/94 Examiner's Initials	AS FILED →	STATE OR COUNTRY	SHEETS DRWGS.	TOTAL CLAIMS	INDEP. CLAIMS	FILING FEE RECEIVED	ATTORNEY'S DOCKET NO.
		CA	11	21	6	$954.00	13188711XERO

ADDRESS

~~TOWNSEND AND TOWNSEND KHOURIE AND CREW~~
~~STEUART STREET TOWER~~
~~ONE MARKET PLAZA~~
~~SAN FRANCISCO CA 94105~~

Xerox Corporation
Pat. Dept.
Xerox Sq. - 020
100 Clinton Ave. South
Rochester, NY 14644

TITLE

SEMANTIC CO-OCCURRENCE FILTERING FOR SPEECH RECOGNITION AND SIGNAL TRANSCRIPTION APPLICATIONS

U.S. DEPT. of COMM.-Pat. & TM Office — PTO-436L (rev. 10-78)

PARTS OF APPLICATION FILED SEPARATELY

M Jones 9.14.95
Applications Examiner

NOTICE OF ALLOWANCE MAILED		CLAIMS ALLOWED	
9/14/95	DR. MICHAEL A. SARTORI Assistant Examiner	Total Claims 22	Print Claim 2

ISSUE FEE		DAVID D. KNEPPER PRIMARY EXAMINER GROUP 2300 Primary Examiner	DRAWING	
Amount Due 1210.00	Date Paid 12-14-95		Sheets Drwg. 11	Figs. Drwg. 11	Print Fig. 3

ISSUE BATCH NUMBER P1M

Label Area

PREPARED FOR ISSUE

WARNING: The information disclosed herein may be restricted. Unauthorized disclosure may be prohibited by the United States Code Title 35, Sections 122, 181 and 368. Possession outside the U.S. Patent & Trademark Office is restricted to authorized employees and contractors only.

Form PTO-436A
(Rev. 8/92)

(FACE)

	ISSUE CLASSIFICATION
Subclass	
Class	

UTILITY SERIAL NUMBER 08/126170	PATENT DATE	PATENT NUMBER

SERIAL NUMBER	FILING DATE	CLASS	SUBCLASS	GROUP ART UNIT	EXAMINER
08/126,170	09/23/93	395	2.79	~~2301~~ 2308	Sartori

APPLICANTS

JULIAN M. KUPIEC, CUPERTINO, CA.

CONTINUING DATA*******************
VERIFIED

4/22/94 MAJ none

FOREIGN/PCT APPLICATIONS**********
VERIFIED

MAJ 4/22/94 none

FOREIGN FILING LICENSE GRANTED 12/30/93

		STATE OR COUNTRY	SHEETS DRWGS.	TOTAL CLAIMS	INDEP. CLAIMS	FILING FEE RECEIVED	ATTORNEY'S DOCKET NO.
Foreign priority claimed □ yes ☒ no 35 USC 119 conditions met □ yes ☒ no Verified and Acknowledged MAJ 4/22/94 Examiner's Initials	AS FILED →	CA	11	22	6	$1,106.00	1318871

ADDRESS

DAVID N. SLONE
TOWNSEND AND TOWNSEND KHOURIE AND CREW
STEUART STREET TOWER
ONE MARKET PLAZA, 20TH FLOOR
SAN FRANCISCO, CA 94105

TITLE

SEMANTIC CO-OCCURRENCE FILTERING FOR SPEECH RECOGNITION AND SIGNAL TRANSCRIPTION APPLICATIONS

U.S. DEPT. of COMM.-Pat. & TM Office — PTO-436L (rev. 10-78)

PARTS OF APPLICATION FILED SEPARATELY	Applications Examiner

NOTICE OF ALLOWANCE MAILED	Assistant Examiner	CLAIMS ALLOWED Total Claims / Print Claim
ISSUE FEE Amount Due / Date Paid	Primary Examiner	DRAWING Sheets Drwg. / Figs. Drwg. / Print Fig.
Label Area	PREPARED FOR ISSUE	ISSUE BATCH NUMBER

WARNING: The information disclosed herein may be restricted. Unauthorized disclosure may be prohibited by the United States Code Title 35, Sections 122, 181 and 368. Possession outside the U.S. Patent & Trademark Office is restricted to authorized employees and contractors only.

Form PTO-436A
(Rev. 8/92)

(FACE)

08/126170

Date Entered or Counted

	1.	Application ______ papers.	
	2.	no fee notes	10/6/93
	3.	Fee, Clm. Fee(s), Surch. & Decl.	10-18-93
	4.	prior art	11-5-93
4/29	5.	Rej. (3 mos.)	5/3/94 c/m
	6.	Amdt. A	5-23-94 / 5-18
6/7	7.	Final Rej. (3 mos.)	6/8/94
	8.	AMDT. B (N.E.)	9-12-94 / c/m 9-8
9/22	9.	Advisory Action	9-23-94
	10.	EXT OF TIME (1 month)	9-30-94
	11.		
	12.		
	13.		
	14.		
	15.		
	16.		
	17.		
	18.		
	19.		
	20.		
	21.		
	22.		
	23.		
	24.		
	25.		
	26.		
	27.		
	28.		
	29.		
	30.		
	31.		
	32.		

(FRONT)

08/316619

PATENT APPLICATION

08316619

APPROVED FOR LICENSE □

INITIALS ____________

CONTENTS

Date Entered or Counted		Date Received or Mailed
	1. Application ✓ papers.	
11	2. PRE AMDT. C	9/30/94
12	3. PRE AMDT. D	9/30/94
12-5 13	4. Rej (3) mons	12-6-94
14	5. Info Disclosure State	Mar 20, 1995
15	6. Ext. of time (1 month)	3-17-95
16	7. AMDT. E	3-17-95
5/10 17	8. Final rej (3)	5-5-95
18	9. Interview Summary	7/26/95
19	10. AMDT. F (N.E.)	7-31-95
8/10 20	11. Advisory Action	8/10/95
21	12. Exr Interview Summary	8-24-95
22	13. Ext of time (1 month)	8-28-95
23	14. Amdt G (N.E.)	8-28-95
9/13 24	15. Ntc of Allow.	9/14/95
25	16. Change of Address	8/28/95
26	17. Associate Power	10/27/95
11/3/95 27	18. Formal Drawings (11 shts) set 1	10/4/95
28	19. PTO GRANT MAR 19 1996	
29	20.	
30	21.	
31	22.	
32	23.	
33	24.	
34	25.	
35	26.	
36	27.	
37	28.	
38	29.	
39	30.	
40	31.	
41	32.	

(FRONT)

Attorney Docket No. 13188-71
Xerox Docket No. D/93288

APPENDIX TO PATENT APPLICATION FOR:

SEMANTIC CO-OCCURRENCE FILTERING
FOR SPEECH RECOGNITION AND SIGNAL TRANSCRIPTION APPLICATIONS

Inventor:	Julian Kupiec 10079 Craft Drive Cupertino, California 95014 British citizen
Assignee:	XEROX CORPORATION 800 Long Ridge Road Stamford, Connecticut 06904 New York corporation
Entity:	Large

DO NOT PRINT APPENDIX
MAP 1/30/96

TOWNSEND and TOWNSEND KHOURIE and CREW
Steuart Street Tower, 20th Floor
One Market Plaza
San Francisco, California 94105
(415) 543-9600

Attorney Docket No. 13188-71
Xerox Docket No. D/93288

SEMANTIC CO-OCCURRENCE FILTERING SOFTWARE PROGRAM

SOURCE CODE FILE #1

THIS FILE INCLUDES CODE FOR:

- READING IN PHONETIC INDEX FILES
- QUERY CONSTRUCTION
- SCORING

```
;;;-*- Package: USER; Syntax: Common-Lisp; Base: 10 -*-


;;; File converted on  6-Sep-93 17:03:56 from source phonetic-corpus
;;;. Original source {dsk}<project>markov>phonetic>phonetic-corpus.;168 created  6-Sep-93
00:47:15

;;;. Copyright (c) 1989, 1992, 1993 by Xerox Corporation


(provide "PHONETIC-CORPUS")

(in-package "USER")

;;; Shadow, Export, Require, Use-package, and Import forms should follow here


(defvar *buffer-is-a-word* nil)

(defvar *interval-wd-vec* nil)

(defvar *moby-hashtable* nil)

(defvar *moby-to-reduced* '(("_" "wd-sep")
                            ("7-/" "skip")
                            ("/&/" "ae")
                            ("/(@)/" "eh")
                            ("/[@]/" "er2")
                            ("/A/" "ao2")
                            ("/eI/" "ey")
                            ("/@/" "ah2")
                            ("b" "sil2-b")
                            ("/tS/" "ch")
                            ("d" "sil2-d")
                            ("/E/" "eh")
                            ("/i/" "iy")
                            ("f" "f")
                            ("g" "sil2-g")
                            ("h" "hh2")
                            ("/hw/" "w")
                            ("/I/" "ih2")
                            ("/aI/" "ay")
                            ("/dZ/" "jh")
                            ("k" "sil2-k")
                            ("l" "l2")
                            ("m" "m2")
                            ("/N/" "ng2")
                            ("n" "n2")
                            ("/Oi/" "oy")
                            ("/A/" "ao2")
                            ("/AU/" "aw")
                            ("/O/" "ao2")
                            ("/oU/" "ow")
                            ("/u/" "uw2")
                            ("/U/" "uh")
                            ("p" "sil2-p")
                            ("r" "r")
                            ("/S/" "sh2")
                            ("s" "s")
                            ("/T/" "th")
                            ("/D/" "dh")
                            ("t" "sil2-t")
                            ("/@r/" "er2")
                            ("v" "v")
                            ("w" "w")
                            ("/j/" "y")
                            ("/Z/" "sh2")
                            ("/z/" "z")
                            ("z" "z")
                            ("A" "foreign")
                            ("N" "foreign")
                            ("R" "foreign")
                            ("/x/" "foreign")
                            ("/y/" "foreign")
                            ("Y" "foreign")
                            ("W" "w")
                                                          ; fixable common error
                            ("Z" "z")
                                                          ; fixable common error
                            ("V" "v")
```

```
                                                ; fixable common error
                          ("e" "eh")
                                                ; fixable common error
                          ("a" "ae")
                                                ; /&/ fixable common error
                          ("S" "s")
                                                ; mostly common error
                          ))

(defvar *pocket-fsm* nil)

(defvar *pocket-phone-fsm* nil)

(defvar *pocket-phone-vector* nil)

(defvar *pocket-vector* nil)

(defvar *proximity* 10)

(defvar *query-results* nil)

(defvar *stop-hashtable* nil)

(defvar *wds-buffer* (make-array 100 :element-type 'string-char :fill-pointer 0 :adjustable t))

(defvar *any-phone* nil)

(defvar *code-stream* nil)

(defvar *coded-pocket-vector* nil)

(defvar *confusion-class-fsms* nil)

(defvar *confusion-matrix-map*
       '((:iy (:iy :ih2 :ey :y))
         (:ih2 (:iy :ih2 :eh :ah2 :ae :uw2 :uh :er2 :ey :r :y :n2))
         (:eh (:ih2 :eh :ae :ah2 :er2 :ey :ay))
         (:ae (:eh :ae :ay :aw))
         (:ah2 (:ih2 :eh :ae :ah2 :uw2 :uh :ao2 :er2 :ay :oy :aw :ow :l2 :n2))
         (:uw2 (:iy :ih2 :uw2 :y))
         (:uh (:ih2 :uh :ah2))
         (:ao2 (:ah2 :ao2 :ay :aw :ow :w))
         (:er2 (:ih2 :eh :ah2 :er2 :r))
         (:ey (:iy :ih2 :eh :ey))
         (:ay (:ao2 :ay :oy))
         (:oy (:oy))
         (:aw (:aw))
         (:ow (:ah2 :uw2 :ao2 :aw :ow :l2))
         (:l2 (:ah2 :uw2 :ao2 :ow :l2 :w :n2))
         (:r (:ih2 :uw2 :ao2 :er2 :ay :r :w))
         (:y (:iy :y))
         (:w (:ao2 :l2 :w))
         (:m2 (:ah2 :m2 :n2 :ng2 :sil2))
         (:n2 (:ih2 :ah2 :l2 :n2 :m2 :ng2 :sil2-b :dx :sil2))
         (:ng2 (:n2 :ng2))
         (:ch (:ch))
         (:jh (:ch :jh :sh))
         (:s (:s :z :sh2 :f :th))
         (:z (:s :z :sh2 :th))
         (:sh2 (:ch :sh2))
         (:f (:f :th))
         (:v (:n2 :f :v :th :sil2))
         (:th (:f :th :sil2))
         (:dh (:n2 :v :dh :sil2-b :sil2-d :dx :sil2))
         (:hh2 (:hh2))
         (:sil2-b (:sil2-b :sil2-p))
         (:sil2-d (:sil2-b :sil2-d :sil2-t :sil2))
         (:sil2-g (:sil2-b :sil2-g))
         (:sil2-p (:hh2 :sil2-b :sil2-d :sil2-p :sil2-k))
         (:sil2-t (:ch :jh :s :sil2-d :sil2-p :sil2-t :sil2))
         (:sil2-k (:jh :s :hh2 :sil2-d :sil2-g :sil2-p :sil2-t :sil2-k :sil2))
         (:dx (:n2 :dx))
         (:sil2 (:iy :l2 :y :w :m2 :n2 :ng2 :s :z :v :th :dh :hh2 :dx :sil2))))

(defvar *consonant-hashtable* nil)

(defvar *consonant-letters*
       '(#\b #\c #\d #\f #\g #\h #\j #\k #\l #\m #\n #\p #\q #\s #\t #\v #\x #\z))

(defvar *debug1* nil)
```

```
(defvar *file-hashtable* nil)

(defvar *foot-pocket* nil)

(defvar *foot-pocket-code-file* "foot-pocket.codes")

(defvar *foot-pocket-to-reduced* '(("p" :sil2-p)
                                   ("t" :sil2-t)
                                   ("k" :sil2-k)
                                   ("b" :sil2-b)
                                   ("d" :sil2-d)
                                   ("g" :sil2-g)
                                   ("C" :ch)
                                   ("J" :jh)
                                   ("s" :s)
                                   ("S" :sh2)
                                   ("z" :z)
                                   ("Z" :sh2)
                                   ("f" :f)
                                   ("T" :th)
                                   ("v" :v)
                                   ("D" :dh)
                                   ("h" :hh2)
                                   ("n" :n2)
                                   ("m" :m2)
                                   ("G" :ng2)
                                   ("N" :n2)
                                   ("M" :m2)
                                   ("L" :l2)
                                   ("l" :l2)
                                   ("r" :r)
                                   ("w" :w)
                                   ("y" :y)
                                   ("i" :iy)
                                   ("I" :ih2)
                                   ("E" :eh)
                                   ("e" :ey)
                                   ("@" :ae)
                                   ("a" :ao2)
                                   ("W" :aw)
                                   ("Y" :ay)
                                   ("^" :ah2)
                                   ("c" :ao2)
                                   ("O" :oy)
                                   ("o" :ow)
                                   ("U" :uh)
                                   ("u" :uw2)
                                   ("R" :er2)
                                   ("x" :ah2)
                                   ("|" :ih2)
                                   ("X" :er2)))

(defvar *foot-pocket-to-reduced-hashtable* nil)

(defvar *fricative-fsm* nil)

(defvar *grolier-fsm-mode* t)

(defvar *group-vector* nil)

(defvar *hmm-state-hashtable* nil)

(defvar *hmm-state-list*
       '(:iy :ih2 :eh :ae :ah2 :uw2 :uh :ao2 :er2 :ey :ay :oy :aw :ow :l2 :r :y :w :m2 :n2 :ng2
:ch
            :jh :s :z :sh2 :f :v :th :dh :hh2 :b :d :g :p :t :k :dx :sil2 :sil2-b :sil2-d
:sil2-g
            :sil2-p :sil2-t :sil2-k :wd-sep))

(defvar *hypo-hashtable* nil)

(defvar *letter-codes-file* "letter.codes")

(defvar *letter-count-file* "letter-count-vector.lisp")

(defvar *letter-counts-for-states* nil)

(defvar *nasal-fsm* nil)
```

```
(defvar *nr-foreign-words* nil)

(defvar *nr-format-errors* nil)

(defvar *nr-grolier-wds* nil)

(defvar *nr-hyphenations-ignored* nil)

(defvar *nr-outputs* (+ 2 (- (char-code #\z)
                             (char-code #\a))))

(defvar *nr-states-per-phone* 2)

(defvar *phone-hashtable* nil)

(defvar *reduced-phones* '(("b" :b)
                           ("d" :d)
                           ("g" :g)
                           ("p" :p)
                           ("t" :t)
                           ("k" :k)
                           ("bcl" :sil2)
                           ("dcl" :sil2)
                           ("gcl" :sil2)
                           ("pcl" :sil2)
                           ("tcl" :sil2)
                           ("kcl" :sil2)
                           ("dx" :dx)
                           ("q" :sil2)
                           ("jh" :jh)
                           ("ch" :ch)
                           ("s" :s)
                           ("sh" :sh2)
                           ("z" :z)
                           ("zh" :sh2)
                           ("f" :f)
                           ("th" :th)
                           ("v" :v)
                           ("dh" :dh)
                           ("m" :m2)
                           ("n" :n2)
                           ("ng" :ng2)
                           ("em" :m2)
                           ("en" :n2)
                           ("eng" :ng2)
                           ("nx" :n2)
                           ("l" :l2)
                           ("r" :r)
                           ("w" :w)
                           ("y" :y)
                           ("hh" :hh2)
                           ("hv" :hh2)
                           ("el" :l2)
                           ("iy" :iy)
                           ("ih" :ih2)
                           ("eh" :eh)
                           ("ey" :ey)
                           ("ae" :ae)
                           ("aa" :ao2)
                           ("aw" :aw)
                           ("ay" :ay)
                           ("ah" :ah2)
                           ("ao" :ao2)
                           ("oy" :oy)
                           ("ow" :ow)
                           ("uh" :uh)
                           ("uw" :uw2)
                           ("ux" :uw2)
                           ("er" :er2)
                           ("ax" :ah2)
                           ("ix" :ih2)
                           ("axr" :er2)
                           ("ax-h" :ah2)
                           ("epi" :sil2)
                           ("pau" :sil2)
                           ("h#" :sil2)))

(defvar *sonorant-fsm* nil)

(defvar *stop-fsm* nil)
```

```
(defvar *stop-list*
      '("a" "about" "above" "after" "also" "all" "although" "an" "and" "any" "are" "as" "at" "b"
             "be" "because" "before" "between" "but" "by" "c" "can" "could" "d" "e" "each"
"either"
             "f" "for" "from" "further" "g" "h" "have" "he" "her" "here" "his" "him" "however"
"i"
             "if" "in" "into" "is" "it" "its" "j" "k" "l" "m" "may" "me" "might" "my" "n" "now"
"o"
             "of" "on" "one" "or" "other" "our" "out" "p" "q" "r" "s" "she" "should" "since"
"some"
             "such" "t" "than" "that" "the" "then" "there" "therefore" "these" "this" "those"
             "through" "thus" "to" "u" "until" "up" "upon" "v" "w" "was" "were" "what" "when"
"where"
             "whether" "which" "while" "with" "who" "will" "within" "without" "would" "x" "y"
"you"
             "your" "z"))

(defvar *tim-phone-list*
      '("b" "d" "g" "p" "t" "k" "bcl" "dcl" "gcl" "pcl" "tcl" "kcl" "dx" "q" "jh" "ch" "s" "sh"
"z"
             "zh" "f" "th" "v" "dh" "m" "n" "ng" "em" "en" "eng" "nx" "l" "r" "w" "y" "hh" "hv"
"el"
             "iy" "ih" "eh" "ey" "ae" "aa" "aw" "ay" "ah" "ao" "oy" "ow" "uh" "uw" "ux" "er" "ax"
             "ix" "axr" "ax-h" "epi" "pau" "h#"))

(defvar *timit-to-reduced-phone-hashtable* nil)

(defvar *use-confusion-classes* nil)

(defvar *use-nl-buf* t)

(defvar *vowel-fsm* nil)

(defvar *vowel-hashtable* nil)

(defvar *vowel-phones*
      '(:iy :ih2 :eh :ae :ah2 :uw2 :uh :ao2 :er2 :ey :ay :oy :aw :ow :r))

(defun add-pocket-counts nil (let ((add-list '((:b (#\b))
                                                (:d (#\d))
                                                (:g (#\g))
                                                (:p (#\p))
                                                (:t (#\t))
                                                (:k (#\k))
                                                (:dx (#\t))
                                                (:sil2 (#\b #\d #\g #\p #\t #\k)))))

                                  ;; some reduced phones just aren' covered
                                  (dolist (el add-list)
                                      (dolist (letter (second el))
                                          (dotimes (chain-index *nr-states-per-phone*)
                                              (incf (aref *letter-counts-for-states* (char-to-op
                                                                                      letter)
                                                          chain-index
                                                          (gethash (first el)
                                                                 *hmm-state-hashtable*))
                                                    1.0))))))

(defun any-single-phone-fsm (hashtable)
       (let ((fsm nil))
            (maphash #'(lambda (dummy phone-fsm)
                                  (let nil (setq fsm (if fsm
                                                        (fsm:union-fsm fsm phone-fsm)
                                                       phone-fsm))))
                   hashtable)
            fsm))

(defun apply-expr (expr &key (show nil))
       (let* ((count 0)
               (fsm (p-expr-to-fsm expr))
               (fsm-vec (if (arrayp fsm)
                            fsm
                            (vector fsm))))
             (maphash #'(lambda (key entry)
                                   (let ((found nil)
                                         phones-for-words)
                                        (setq phones-for-words (third entry))
                                        (when show
                                            (format t "~a~%" (first entry)))
```

000008

```
                                     (do ((index 0 (1+ index)))
                                         ((or (= index (length phones-for-words))
                                              found))
                                       (setq found (next-word-match phones-for-words index fsm-vec
0
                                                          key entry)))
                                     (when found (incf count))))
                    *file-hashtable*)
             (format t "Nr sent hits: ~a~%" count)))

(defun ask-grolier (expr-list &key (show nil)
                           (show-wds t)
                           (print-limit 20)
                           (heap-limit 100)
                           (max-postings 5000)
                           (prox 10)
                           (search nil))
       (let* ((rval nil)
              query query-str heap conjunct-ids item doc-id nr-hits title-str title-hits)
             (setq heap (pq::make-priority-queue #'(lambda (x y)
                                                           (< (svref x 1)
                                                              (svref y 1)))))
             (when (or (null *hmm-state-hashtable*)
                       (null *grolier-fsm-mode*))
                   (setup-grolier-mode))
             (when (null *stop-hashtable*)
                 (setq *stop-hashtable* (make-hash-table :test #'equal))
                 (mapcar #'(lambda (el)
                                   (setf (gethash el *stop-hashtable*)
                                        t))
                        *stop-list*))
             (multiple-value-setq (query-str conjunct-ids rval)
                    (construct-query expr-list prox show show-wds))
             (when search
                 (setq query (concordance::parse-boolean-query query-str concordance::*tdb*))
                 (format t "Searching...")
                 (concordance::phonetic-search query heap heap-limit max-postings)
                 (format t "Done~%")
                 (setq *the-heap* heap)
                 (score-hits heap)
                 (dotimes (index (pq::pq-length heap))
                     (setq item (pq:pq-pop heap))
                     (setq doc-id (svref item 0))
                     (setq nr-hits (svref item 1))
                     (setq title-str (tdb::doc-title doc-id concordance::*tdb*))
                     (setq title-hits (get-title-hits title-str))
                     (format t "~&~5D: ~A  Score: ~A Title wds: ~A~%" doc-id title-str nr-hits
                          title-hits))
                 (format t "Scoring...~%"))
             rval))

(defun char-to-op (char &optional (print-error t))
       (let ((op nil)
             (charcode (char-code char))
             (a-charcode (char-code #\a))
             (space-charcode (char-code #\Space)))
            (if (null (characterp char))
                (error "ERROR: ~a not a char~%" char)
                (if (and (>= charcode a-charcode)
                         (<= charcode (char-code #\z)))
                    (setq op (1+ (- charcode a-charcode)))
                    (if (= charcode space-charcode)
                        (setq op 0)
                        (when print-error (error "ERROR: ~a is invalid char~%" char)))))
            op))

(defun check-coverage nil (let ((grolier-count 0)
                                (pocket-count 0))
                               (setup-grolier-mode)
                               (format t "Mapping grolier terms..~%")
                               (tdb:map-terms #'(lambda (word)
                                                        (let nil (incf grolier-count)
                                                             (when (= (rem grolier-count 1000)
                                                                      0)
                                                                   (format t "done ~a entries~%"
                                                                          grolier-count))
                                                             (when (fsm:word-to-index word
                                                                          *pocket-fsm*)
                                                                   (incf pocket-count))))
                                      concordance::*tdb*)
                                (format t "~%Nr grolier terms: ~a   Nr coverec ~a~%" grolier-count
                                       pocket-count)))

(defun code-fp-word (word phones-for-word stream)
       (let nil (format stream "#(#(~a ~a)" (char-to-op #\Space)
                        (gethash :wd-sep *hmm-state-hashtable*))
            (phone-to-char-matches word phones-for-word stream)
            (format stream "#(~a ~a))~%" (char-to-op #\Space)
                    (gethash :wd-sep *hmm-state-hashtable*))))

(defun code-sentence (word-vec reduced-phones-for-sent stream)
       (let (word phones-for-word)


;;; Each word is a different obs seq

            (dotimes (wd-index (length word-vec))
                (format stream "#(#(~a ~a)" (char-to-op #\Space)
                        (gethash :wd-sep *hmm-state-hashtable*))
                (setq word (filter-word (aref word-vec wd-index)))
                (setq phones-for-word (aref reduced-phones-for-sent wd-index))
                (phone-to-char-matches word phones-for-word stream)
                (when nil
                    (dotimes (char-index (length word))
                        (format stream "#(~a" (char-to-op (char word char-index)))
                        (dotimes (phone-index (length phones-for-word))
                            (format stream " ~a" (gethash (aref phones-for-word phone-index)
                                                         *hmm-state-hashtable*)))
                        (format stream ")~%" (char-to-op #\Space)))
                    (format stream "#(~a ~a))~%" (char-to-op #\Space)
                            (gethash :wd-sep *hmm-state-hashtable*))))))

(defun code-timit nil (let  ((state-id 0))
                            (setq *hmm-state-hashtable* (make-hash-table :test #'equal))
                            (mapcar #'(lambda (el)
                                             (setf (gethash el *hmm-state-hashtable*)
                                                  ,state-id)
                                             (incf state-id))
                                  *hmm-state-list*)
                            (setq *timit-to-reduced-phone-hashtable* (make-hash-table :test
                                                                             #'equal))
                            (dolist (el *reduced-phones*)
                                (setf (gethash (first el)
                                             *timit-to-reduced-phone-hashtable*)
                                      (second el)))
                            (setq *consonant-hashtable* (make-hash-table :test #'equal))
                            (dolist (el *consonant-letters*)
                                (setf (gethash el *consonant-hashtable*)
                                     el))
                            (setq *vowel-hashtable* (make-hash-table :test #'equal))
                            (dolist (el *vowel-phones*)
                                (setf (gethash el *vowel-hashtable*)
                                     el))
                            (setq *letter-counts-for-states* (make-array (list *nr-outputs*

*nr-states-per-phone*
                                                                                (length

*hmm-state-list*
                                                                                       ))
                                                                    :initial-element 0))
                            (with-open-file (*code-stream* *letter-codes-file* :direction :output
                                                   :if-exists :new-version)
                                   (maphash #'code-timit-sentence *file-hashtable*))
                            (normalize-count-vector)))

(defun code-timit-sentence (key entry)
       (let ((word-vec (first entry))
             (phone-vec (second entry))
             (reduced-phones-for-sent (make-array 0 :adjustable t :fill-pointer 0)))
            (dotimes (index (length phone-vec))
                (vector-push-extend (convert-timit-phones (aref phone-vec index))
                       reduced-phones-for-sent))
            (when *debug* (format t "Reduced phones: for ~a ~%~%~a~%" phone-vec
                                 reduced-phones-for-sent))
            (when (> (length reduced-phones-for-sent)
                     0)


;;; make sure that phones do exist e.g. none for the word "a" in /makr0/sil982
```

```
                (fill-count-vector reduced-phones-for-sent word-vec)

                ;; write char-to-ops to op-stream
                (code-sentence word-vec reduced-phones-for-sent *code-stream*))))

(defun construct-query (expr-list proximity show show-wds)
       (let ((expr-index 0)
             (conjunct-strs (make-array (length expr-list)
                                   :initial-element nil))
             (conjunct-ids nil)
             (rval nil)
             curr-score pair word-hit score query-str stemmed-term word-present term-str count)
            (when nil
                (setq conjunct-ids (make-array (length expr-list)
                                          :initial-element
                                          (make-array 50 :adjustable t :fill-pointer 0))))
            (setq query-str " )")
            (if (hash-table-p *hypo-hashtable*)
                (clrhash *hypo-hashtable*)
                (setq *hypo-hashtable* (make-hash-table :test #'equal)))
            (dolist (expr expr-list)
                (when show (format t "~%expr: ~a~%" expr))
                (setq term-str ") ")
                (setq count 0)
                (setq word-present nil)
                (dolist (pair expr)
                    (setq word-hit (car pair))
                    (setq score (cdr pair))
                    (setq stemmed-term (twol-stemmer::lookup-stem (copy-seq word-hit)))
                    (if (or (gethash word-hit *stop-hashtable*)
                            (gethash stemmed-term *stop-hashtable*))
                        (when show (format t "~a on stop list -- ignored~%" word-hit))
                        (progn (setq word-present t)
                               (push word-hit (aref conjunct-strs expr-index))
                               (when nil
                                   (vector-push-extend (vector nil 0)
                                                             ; was wd-index
                                          (aref conjunct-ids expr-index)))
                               (setq term-str (concatenate 'string word-hit " " term-str))


;;; Shouldn't put words not recognized by  groliers in here !!!! --Must not insert nils into
;;; *HYPO-HASHTABLE*

                               (when (null stemmed-term)
                                   (setq stemmed-term (copy-seq word-hit)))


;;; Not very good way of doing things, because word may appear in different list and get
different
;;; scores ----for now keep max

                               (setq curr-score (gethash stemmed-term *hypo-hashtable*))
                               (if curr-score
                                   (when (> score curr-score)
                                       (setf (gethash stemmed-term *hypo-hashtable*)
                                             score))
                                   (setf (gethash stemmed-term *hypo-hashtable*)
                                         score))
                               (when (or show show-wds)
                                   (format t "~D: ~12,3T (~D)~%" word-hit (tdb::term-freq
word-hit

concordance::*tdb*)))
                               (incf count))))
                (when (and word-present (eql 1 (length expr-list)))
                    (setq rval t))
                (when (> (length expr-list)
                         1)
                    (setq rval (nconc rval (list (if word-present
                                                     t
                                                     nil)))))
                (setq term-str (concatenate 'string "( " term-str))
                (setq query-str (concatenate 'string term-str query-str))
                (when (or show show-wds)
                      (format t "Nr hits : ~a~%" count))
                (incf expr-index))
            (setq query-str (concatenate 'string "(" (format nil "~d " proximity)
                                   query-str))
            (when show (format t "Query : ~a~%" query-str))
            (values query-str conjunct-ids rval)))

(defun convert-foot-pocket nil (let ((state-id 0)
                                     (file-name "foot-pocket-no-markers.lisp")
                                     reduced-phones word)
                                    (format t "Converting Pocket Dictionary~%")
                                    (when (null *foot-pocket*)
                                          (load file-name))
                                    (setq *foot-pocket-to-reduced-hashtable* (make-hash-table
                                                                              :test
                                                                              #'equal))
                                    (dolist (el *foot-pocket-to-reduced*)
                                        (setf (gethash (char (first el)
                                                             0)
                                                     *foot-pocket-to-reduced-hashtable*)
                                              (second el)))
                                    (setq *hmm-state-hashtable* (make-hash-table :test #'equal))
                                    (mapcar #'(lambda (el)
                                                     (setf (gethash el *hmm-state-hashtable*)
                                                           state-id)
                                                     (incf state-id))
                                           *hmm-state-list*)
                                    (setq *timit-to-reduced-phone-hashtable* (make-hash-table
                                                                              :test
                                                                              #'equal))
                                    (dolist (el *reduced-phones*)
                                        (setf (gethash (first el)
                                                     *timit-to-reduced-phone-hashtable*)
                                              (second el)))
                                    (setq *consonant-hashtable* (make-hash-table :test #'equal))
                                    (dolist (el *consonant-letters*)
                                        (setf (gethash el *consonant-hashtable*)
                                              el))
                                    (setq *vowel-hashtable* (make-hash-table :test #'equal))
                                    (dolist (el *vowel-phones*)
                                        (setf (gethash el *vowel-hasntable*)
                                              el))
                                    (setq *letter-counts-for-states* (make-array (list
*nr-outputs*

*nr-states-per-phone*
                                                                                       (length

*hmm-state-list*
                                                                                             ))
                                                                            :initial-element 0))
                                    (with-open-file (stream *foot-pocket-code-file* :direction
                                                           :output :if-exists :new-version)
                                           (dolist (el *foot-pocket*)
                                               (setq word (filter-word (string-downcase (first
el))))

                                               (setq reduced-phones (convert-fp-word (second
el)))

                                               (code-fp-word word reduced-phones stream)
                                               (fill-count-vector (vector reduced-phones)
                                                     (vector word))))
                                    (add-pocket-counts)
                                    (normalize-count-vector)))

(defun convert-fp-word (fp-phones-for-word)
       (let (reduced-phone (reduced-phones-for-word (make-array 0 :adjustable t :fill-pointer
0)))
            (dotimes (index (length fp-phones-for-word))
                (setq reduced-phone (gethash (char fp-phones-for-word index)
                                           *foot-pocket-to-reduced-hashtable*))
                (when (null reduced-phone)
                    (error "ERROR: word ~a char ~a not in hashtable~%" fp-phones-for-word
                           (char fp-phones-for-word index)))
                (vector-push-extend reduced-phone reduced-phones-for-word))
            reduced-phones-for-word))

(defun convert-moby-pronunciator (&key (errorstream t)
                                       (grolier-overlap t))
       (let ((in-file "/project/corpora/moby-products/pronunciator/Moby_Pronunciator_II.txt")
             (full-outfile
"/project/corpora/moby-products/pronunciator/mapped-pronunciations.txt")
             (grolier-outfile "/project/markov/phonetic/moby-grolier-overlap.txt")
             (error-file "/project/markov/phonetic/moby-errors.txt")
             (char-buf (make-array 100 :element-type 'string-char :fill-pointer 0 :adjustable t))
             (output-buf (make-array 1000 :element-type 'string-char :fill-pointer 0 :adjustable
t))
             (line-number 0)
             outfile)
            (setq outfile (if grolier-overlap
                              grolier-outfile
                              full-outfile))
            (fill-moby-hashtable)
            (setq *nr-format-errors* 0)
            (setq *nr-foreign-words* 0)
            (setq *nr-grolier-wds* 0)
            (setq *nr-hyphenations-ignored* 0)
            (setq *char-buf* char-buf)
            (with-open-file (in-stream in-file :direction :input)
                   (with-open-file (outstream outfile :direction :output :if-exists :new-version)
                          (do ((line (read-line in-stream nil nil)
                                     (read-line in-stream nil nil)))
                              ((null line))
                            (read-moby-form grolier-overlap line line-number output-buf char-buf
                                   outstream errorstream)
                            (when (= 0 (rem line-number 1000))
                                  (format t "~a words done. Current line: ~a~%" line-number
line))
                            (incf line-number))))
            (format t "Nr total lines: ~A~%" line-number)
            (format t "Nr word format errors: ~A~%" *nr-format-errors*)
            (format t "Nr grolier words: ~A~%" *nr-grolier-wds*)
            (format t "Nr hyphenations ignored: ~A~%" *nr-hyphenations-ignored*)
            (format t "Nr Foreign words ignored: ~A~%" *nr-foreign-words*)))

(defun convert-timit-pnones (phones-for-word)
       (let ((skipping-index 0)
             (reduced-phones-for-word (make-array 0 :adjustable t :fill-pointer 0))
             clos-posn burst-posn timit-phone reduced-phone)

            ;; Need to convert Kcl followed by k to kcl-k state, and what about wd boundaries and
            ;; phone that definitely don't map to a letter?
            (when nil (vector-push-extend :wd-sep reduced-phones-for-word))
            (do nil
                ((= skipping-index (length phones-for-word)))
              (setq timit-phone (aref phones-for-word skipping-index))
              (setq reduced-phone (gethash timit-phone *timit-to-reduced-phone-hashtable*))
              (when (null reduced-phone)
                    (format t "ERROR: No reduced phone for ~a~%" timit-phone))
              (when (and (> (length phones-for-word)
                             (1+ skipping-index))
                         (setq clos-posn (position timit-phone '("bcl" "dcl" "gcl" "pcl" "tcl"
"kcl")
                                                         :test
                                                         #'string=))
                         (setq burst-posn (position (aref phones-for-word (1+ skipping-index))
                                                         '("b" "d" "g" "p" "t" "k")
                                                         :test
                                                         #'string=))
                         (= clos-posn burst-posn))
                  (setq reduced-phone (nth clos-posn '(:sil2-b :sil2-d :sil2-g :sil2-p :sil2-t
                                                             :sil2-k)))
                  (incf skipping-index))
              (incf skipping-index)
              (vector-push-extend reduced-phone reduced-phones-for-word))
            (when nil (vector-push-extend :wd-sep reduced-phones-for-word))
            reduced-phones-for-word))

(defun decode (arg &key (name "phon"))
       (let (obs-vec (obs-name "CURRENT-SENTENCE"))
            (if (null (setq obs-vec (rhmm:get-training-data arg)))
                (progn (setq obs-vec (make-array (+ 2 (length arg))))
                       (setf (aref obs-vec 0)
                             (vector 0))
                       (dotimes (index (length arg))
                           (setf (aref obs-vec (1+ index))
                                 (vector (char-to-op (char arg index)))))
                       (setf (aref obs-vec (1+ (length arg)))
                             (vector 0))))
            (rhmm:set-training-data (list obs-name obs-vec)
                   :overwrite t)
            (rhmm:viterbi-search name obs-name)
            (dotimes (index (length obs-vec))
                (format t "~a  ~a~%" (op-to-char (aref (aref obs-vec index)
                                                       0))
                       (nth (truncate (aref rhmm:*viterbi-states* index)
                                   *nr-states-per-phone*)
                            *hmm-state-list*)))))

(defun fill-count-vector (reduced-phones-for-sent word-vec)
       (let* ((attenuation 0.5)
              (center-phone-state (truncate (* 0.5 *nr-states-per-phone*)))
              (letter-spread *nr-states-per-phone*)
              (left-spread (truncate (* 0.5 letter-spread)))
              (right-spread (- letter-spread left-spread 1))
              weight sub-phone-index word wd-len nr-phones this-phone this-phone-index
              center-wd-index low-wd-index hi-wd-index phones-for-word)


;;;


;;; Right now, LETTER-SPREAD must be equal to *NR-STATES-PER-PHONE*


;;; otherwise there should be a map


;;;

             (dotimes (wd-index (length reduced-phones-for-sent))
                 (setq phones-for-word (aref reduced-phones-for-sent wd-index))
                 (setq word (filter-word (aref word-vec wd-index)))
                 (setq wd-len (length word))
                 (setq nr-phones (length phones-for-word))
                 (dotimes (phone-index nr-phones)
                     (setq this-phone (aref phones-for-word phone-index))
                     (setq this-phone-index (gethash this-phone *hmm-state-hashtable*))
                     (setq center-wd-index (if (= nr-phones 1)
                                               (round (* 0.5 (float (1- wd-len))))
                                               (round (/ (* (float phone-index)
                                                            (float (1- wd-len)))
                                                         (1- nr-phones)))))
                     (setq low-wd-index (- center-wd-index left-spread))
                     (setq hi-wd-index (+ center-wd-index right-spread))
                     (when *debug* (format t "~a: " this-phone))
                     (do ((index low-wd-index (1+ index)))
                         ((> Index hi-wd-index))
                       (when (and (>= index 0)
                                  (< index wd-len)
                                  (char-to-op (char word index)
                                         nil))
                           (setq weight 1.0)
                           (unless (= index center-wd-index)
                               (setq weight (* attenuation weight)))
                           (setq sub-phone-index (+ center-phone-state (- index
center-wd-index)))
                           (when (and (>= sub-phone-index 0)
                                      (< sub-phone-index *nr-states-per-phone*))
                               (incf (aref *letter-counts-for-states* (char-to-op (char word
index))
                                           sub-phone-index this-phone-index)
                                     weight)
                               (when *debug*
                                   (format t "~a: ~a " (char word index)
                                          weight)))))
                     (when *debug* (format t "~%"))))))

(defun fill-moby-hashtable nil (let nil (setq *moby-hashtable* (make-hash-table :test #'equal))
                                    (dolist (el *moby-to-reduced*)
                                        (setf (gethash (first el)
                                                     *moby-hashtable*)
                                              (second el)))))

(defun fill-phone-hmm-op-vec (model)
       (let ((nr-states (rhmm::hmm-defn-nr-states model))
             (nr-outputs (rhmm::hmm-defn-nr-outputs model))
             this-index prob)

            ;;

            ;; don't forget  :wd-sep = 1.0 op for Space
            ;;
            (dotimes (i-state (length *hmm-state-list*))
                (dotimes (chain-index *nr-states-per-phone*)
                    (setq this-index (+ chain-index (* i-state *nr-states-per-phone*)))
                    (dotimes (obs nr-outputs)
                        (setq prob (aref *letter-counts-for-states* obs chain-index i-state))
                        (when (> 1.0E-10 prob)
                              (setq prob 1.0E-10))
                        (setf (aref (aref (aref *output-matrix* 0)
                                                  this-index)
                                            obs)
                              (double-float prob))
                        (when (= i-state (position :wd-sep *hmm-state-list*))
                            (setf (aref (aref (aref *output-matrix* 0)
                                                      this-index)
                                                obs)
                                  (if (= obs (char-to-op #\Space))
                                      (double-float 1.0)
                                      (double-float 0.0)))))))
            (rhmm::hmm-defn-output-to-state-index model)
            (rhmm:make-output-to-state-index nr-states nr-outputs *output-matrix*)))

(defun fill-phonetable (phone-vec &key (show t))
       (let (this-phone phones-for-wd not-done this-fsm)
            (dotimes (wd-index (length phone-vec))
                (setq phones-for-wd (aref phone-vec wd-index))
                (dotimes (ph-index (length phones-for-wd))
                    (setq this-phone (aref phones-for-wd ph-index))
                    (when (null (gethash this-phone *phone-hashtable*))
                        (setq not-done t)
                        (when show (format t "NEW PHONE: ~a~%" this-phone))
                        (setq this-fsm (fsm:make-word-list-fsm (let ((word this-phone))
                                                                    (if not-done
                                                                        (progn (setq not-done
nil)
                                                                               word)
                                                                        nil))
                                                      :print-p nil))
                        (setf (gethash this-phone *phone-hashtable*)
                              this-fsm)))))

(defun filter-word (word)
       (let ((ok t)
             (failure nil)
             new-wd)
            (dotimes (index (length word))
                (when (null (char-to-op (char word index)
                                   nil))
                    (unless (char= #\' (char word index))
                           (setq failure t))
                    (setq ok nil)))
            (when failure (format t "FAILURE: ~a contains illegal chars~%" word))
            (when (null ok)
                (setq new-wd (delete #\' word)))
            (if ok
                word
                new-wd)))

(defun find-group-recurrences (howmany &key (show nil))
       (let ((group-hashtable (make-hash-table :test #'equal))
             key value count result group-vec start end doc-id interval-wds (last-end -1)
             (last-wds nil)
             (last-doc-id -1))
            (do* ((index 0 (1+ index)))
                 ((or (= index (length *interval-wd-vec*))
                      (null (svref *interval-wd-vec* index))
                      (>= index howmany)))
               (setq interval-wds (svref *interval-wd-vec* index))


;;; There may be multiple identical words in interval vec, causing longer redundant hypothesis
;;; combinations (we are assuming here that words aren't repeated in the spoken query .

               (when nil
                   (setq key (copy-seq interval-wds)))
               (setq key (remove-duplicates interval-wds :test #'string-equal))
               (when (null key)
                     (format t "ERROR: key is null for interval index ~A~%" index))
               (setq result (svref *query-results* index))
```

```
                (setq doc-id (svref result 0))
                (when (/= doc-id last-doc-id)
                     (setq last-end -1))
                (setq start (svref result 1))
                (setq end (svref result 2))

                ;; get non-overlapping intervals

;;; (or (null (subsetp key last-wds :test #'string-equal)) (> start last-end))


;;; Different word in the interval vec mean a different hit is to be considered, irrespective of
;;; start/end ??
                (when (or (/= doc-id last-doc-id)
                          (null (subsetp key last-wds :test #'string-equal))
                          (> start last-end))
                    (setq value (gethash key group-hashtable))
                    (if value
                        (progn (pushnew doc-id (svref value 0))
                               (incf (svref value 2)))
                        (progn (setq value (vector (list doc-id)
                                                   key 1))
                               (setf (gethash key group-hashtable)
                                     value))))
                (setq last-end end)
                (setq last-wds key)
                (setq last-doc-id doc-id))


;;; Turn into a vector so I can sort it.

            (setq group-vec (make-array (hash-table-count group-hashtable)))
            (setq count 0)
            (maphash #'(lambda (key1 val1)
                               (let nil (setf (svref group-vec count)
                                              val1)
                                    (incf count)))
                   group-hashtable)
            (setq group-vec (sort group-vec #'> :key #'(lambda (x)
                                                               (length (svref x 0)))))
            (when show
                (dotimes (index (length group-vec))
                    (format t "~A   ~D~%" (svref (svref group-vec index)
                                                  1)
                           (svref (svref group-vec index)
                                  2))))
            group-vec))

(defun find-profile (conjunct-ids)
       (let (disjunct-vec)
            (dotimes (index (length conjunct-ids))
                (setq disjunct-vec (aref conjunct-ids index)))))

(defun find-title-words (word-list doc-title)
       (let ((count 0)
             (title-len (length doc-title))
             start end)
            (dolist (word word-list)
                (setq start (search word doc-title :test #'char-equal))
                (when start
                    (setq end (+ start (length word)))
                    (when (and (or (= start 0)
                                   (null (alpha-char-p (char doc-title (1- start)))))
                               (or (= end title-len)
                                   (null (alpha-char-p (char doc-title end)))))
                          (incf count))))
            count))

(defun get-doc-freq (term doc-id)
       (let ((doc-freq 0))


;;; Would be quicker if I could return immediately on finding the doc

            (tdb::do-freqs (id freq term concordance::*tdb*)
                   (when (= id doc-id)
                         (setq doc-freq freq)))
            doc-freq))

(defun get-files (&key (base-dir "/net/llama/llama/speech2/timit/train/")
                       (dr-dirs '("dr1" "dr2" "dr3" "dr4" "dr5" "dr6" "dr7" "dr8"))
                       (verbose t)
                       (add-speaker-variants nil))
       (let (file-str diff-posn spkr-dirs word-vec phone-vec utt utt-dirs posn)
            (setq *file-hashtable* (make-hash-table :test #'equal))
            (setq *phone-hashtable* (make-hash-table :test #'equal))
            (fill-phonetable (vector (apply #'vector *tim-phone-list*))
                   :show nil)
            (setq *grolier-fsm-mode* nil)
            (setq *any-phone* (any-single-phone-fsm *phone-hashtable*))
            (dolist (el dr-dirs)
                (format t "On dir: ~a~%" el)
                (setq spkr-dirs (directory (concatenate 'string base-dir el "/")))
                (dolist (speaker spkr-dirs)
                    (setq utt-dirs (directory (concatenate 'string (namestring speaker)
                                                                "/")))
                    (dolist (file-pathname utt-dirs)
                        (setq file-str (namestring file-pathname))
                        (when (setq posn (or (search "/sx" file-str)
                                             (search "/si" file-str)))
                            (setq utt (subseq file-str posn))
                            (when add-speaker-variants
                                (setq diff-posn (mismatch file-str (concatenate 'string base-dir
el))
                                      )
                                (setq utt (subseq file-str diff-posn)))
                            (setq file-str (concatenate 'string file-str (subseq file-str posn)))
                            (if (gethash utt *file-hashtable*)
                                (when verbose (format t "~a already in hashtable~%" utt))
                                (progn (multiple-value-setq (word-vec phone-vec)
                                             (read-file file-str))
                                       (setq *word-vec* word-vec)
                                       (setq *phone-vec* phone-vec)
                                       (setf (gethash utt *file-hashtable*)
                                             (list word-vec phone-vec (make-phone-fsms phone-vec)
                                                   (make-word-fsm word-vec)
                                                   file-str))
                                       (fill-phonetable phone-vec)
                                       (when verbose (format t "Entered: ~a ~%" utt)))))))))
            (setq *any-phone* (any-single-phone-fsm *phone-hashtable*))))

(defun get-phones-for-word (phone-stream wd-start wd-end)
       (let (phone-start phone-end phone (phones-for-word (make-array 0 :adjustable t
:fill-pointer 0
                                                                ))))
            (loop (setq phone-start (read phone-stream nil nil))
                  (setq phone-end (read phone-stream nil nil))
                  (setq phone (read-line phone-stream nil nil))
                  (when (>= phone-start wd-start)
                      (vector-push-extend phone phones-for-word)
                      (when nil (format t "~a : ~a  ~a~%" phone-start phone-end phone))
                      (if (>= phone-end wd-end)
                          (return nil))))
            phones-for-word))

(defun get-title-hits (title-str)
       (let ((nr-hits 0)
             (start 0)
             (end -1))
            (do nil
                ((null end))
              (setq end (position #\Space title-str :start start))
              (if (null end)
                  (when (gethash (subseq title-str start (length title-str))
                                 *hypo-hashtable*)
                        (incf nr-hits))


;;; (gethash (get-stem (subseq title-str start end)))

                  (progn (when (gethash (subseq title-str start end)
                                        *hypo-hashtable*)
                                 (incf nr-hits))
                         (setq start (1+ end)))))
            nr-hits))

(defun gq (expr &optional (show t))
       (let* ((count 0)
              (fsm (p-expr-to-fsm expr))
              coded-word)
             (when (null *grolier-fsm-mode*)
                   (setup-grolier-mode))
             (dotimes (wd-index (length *coded-pocket-vector*))
                 (setq coded-word (aref *coded-pocket-vector* wd-index))
                 (when (eq :accepted (fsm:apply-fsm-to-array coded-word fsm))
                     (when show
                         (format t "~a: ~a~%" (fsm:index-to-word wd-index *pocket-fsm*)
                                 (aref *pocket-vector* wd-index)))
                     (incf count)))
             (format t "Nr hits : ~a~%" count)))

(defun grol (expr-list &key (show nil)
                   (show-wds t)
                   (print-limit 20)
                   (limit 4000)
                   (prox 10)
                   (search nil))
       (let* ((rval nil)
              query query-str conjunct-ids)
             (when (or (null *hmm-state-hashtable*)
                       (null *grolier-fsm-mode*))
                   (setup-grolier-mode))
             (when (null *stop-hashtable*)
                 (setq *stop-hashtable* (make-hash-table :test #'equal))
                 (mapcar #'(lambda (el)
                                  (setf (gethash el *stop-hashtable*)
                                        t))
                        *stop-list*))
             (multiple-value-setq (query-str conjunct-ids rval)
                    (construct-query expr-list prox show show-wds))
             (when search
                 (setq query (concordance::parse-boolean-query query-str concordance::*tdb*))
                 (format t "Searching...")
                 (concordance::my-offset-search query concordance::*tdb* nil)
                 (setq *query* query)
                 (format t "Done~%")
                 (show-doc-titles limit print-limit))
             rval))

(defun groll (expr-list &key (show nil)
                    (show-interval-hits nil)
                    (show-wds nil)
                    (pr 20)
                    (limit 3000)
                    (prox *proximity*)
                    (search t)
                    (max-doc-search 500)
                    (show-nr-docs 33)
                    (use-moby t))
       (let* ((rval nil)
              (doc-id-hashtable (make-hash-table :test #'equal))
              query the-vec nr-results query-str group-vec conjunct-ids)
             (setq *temp-hash-table* doc-id-hashtable)
             (when (or (null *hmm-state-hashtable*)
                       (null *grolier-fsm-mode*))
                 (if use-moby
                     (moby-setup)
                     (setup-grolier-mode)))
             (when (or (null *query-results*)
                       (< (length *query-results*)
                          limit))
                 (setq *query-results* (make-array limit))
                 (dotimes (index limit)
                     (setf (svref *query-results* index)
                           (make-array 3))))
             (when (or (null *interval-wd-vec*)
                       (< (length *interval-wd-vec*)
                          (length *query-results*)))
                 (setq *interval-wd-vec* (make-array (length *query-results*))))
             (dotimes (index (length *interval-wd-vec*))
                 (setf (svref *interval-wd-vec* index)
                     nil))
             (when nil
                 (when (or (null *group-vector*)
                           (< (length *group-vector*)
                              limit))
                     (setq *group-vector* (make-array limit :adjustable t :fill-pointer 0))
                     (dotimes (index limit)
                         (setf (svref *group-vector* index)
                               (make-array 3))))
                 (dotimes (index (length *group-vector*))
                     (setq the-vec (svref *group-vector* index))
                     (setf (svref the-vec 0)
                           nil)
                     (setf (svref the-vec 1)
                           nil)
                     (setf (svref the-vec 2)
                           0)))
             (when (null *stop-hashtable*)
                 (setq *stop-hashtable* (make-hash-table :test #'equal))
                 (mapcar #' (lambda (el)
                                  (setf (gethash el *stop-hashtable*)
                                        t))
                       *stop-list*))
             (multiple-value-setq (query-str conjunct-ids rval)
                  (construct-query expr-list prox show show-wds))
             (when search
                 (setq query (concordance::parse-boolean-query query-str concordance::*tdb*))
                 (setq *query* query)
                 (format t "Searching...")
                 (setq nr-results (concordance::my-vec-offset-search query *query-results*))
                 (format t "Done~%")
                 (if (= nr-results (length *query-results*))
                     (format t "** Search was truncated at ~D hits **~%" nr-results)
                     (format t "Search found ~D hits ~%" nr-results))
                 (offset-scoring *query-results* *interval-wd-vec* nr-results doc-id-hashtable)
                 (when show-interval-hits
                     (print-interval-hits (min pr nr-results)))
                 (setq group-vec (find-group-recurrences nr-results :show show-interval-hits))
                 (when *use-nl-buf*
                     (concordance::clear-nl-buf)
                     (concordance::print-in-nl-buf (format nil "Relevant Documents:~%~%")))


;;; Is the last argument useful  below in SCORE-GROUPS ??

                 (score-groups group-vec nr-results show-nr-docs max-doc-search))
             t))

(defun hmm-op (reduced-phone &key (name "phon"))
       (let* (char (state-id (gethash reduced-phone *hmm-state-hashtable*))
                   (model (rhmm:find-model name))
                   (op-vec (aref (rhmm::hmm-defn-outputs model)
                                 0)))
             (dotimes (index *nr-outputs*)
                 (setq char (if (= index 0)
                                #\Space
                                (code-char (+ (1- index)
                                              (char-code #\a)))))
                 (format t "~a: " char)
                 (dotimes (chain-index *nr-states-per-phone*)
                     (format t " ~10,7f" (aref (aref op-vec (+ chain-index (*
*nr-states-per-phone*
                                                                             state-id)))
                                              index)))
                 (format t "~%"))))

(defun is-a-word (phone-list-or-vec &optional (end (length phone-list-or-vec)))
       (let (the-vec)
            (setq the-vec (etypecase phone-list-or-vec
                              (array phone-list-or-vec)
                              (list (let ((nr-phones (length phone-list-or-vec))
                                          (buffer-len (length *buffer-is-a-word*))
                                          (phone-index 0))
                                         (when (> nr-phones buffer-len)
                                             (setq *buffer-is-a-word* (make-array nr-phones)))
                                         (dolist (el phone-list-or-vec)
                                             (setf (svref *buffer-is-a-word* phone-index)
                                                   el)
                                             (incf phone-index))
                                         *buffer-is-a-word*))))
            (fsm:apply-fsm-to-array the-vec *pocket-phone-fsm* 0 end)))

(defun make-moby-fsm nil (let ((in-file "/project/markov/phonetic/moby-grolier-overlap.txt")
                               (fsm-output-file "/project/markov/phonetic/moby-grolier.fsm")
                               fsm)
                              (with-open-file (in-stream in-file :direction :input)
                                       (setq fsm (fsm:make-word-list-fsm (let ((line (read
in-stream
                                                                                              nil
nil))
                                                                                  (word nil))
                                                                                 (setq word (car
line))
                                                                                word)
                                                         :partition-form
                                                         (format t
                                                              "Partition number: ~a States so far:
~a~%"
                                                                fsm::partition-count
                                                                fsm::total-states-created))))
                                (fsm:index-network fsm)
                                (fsm:network-to-file fsm fsm-output-file)
                                (when (null fsm)
                                      (format t "WARNING: No FSM was made~%"))
                                fsm))

(defun make-moby-phone-fsm (moby-vector)
       (let ((fsm-output-file "/project/markov/phonetic/moby-phones.fsm")
             (count 0)
             (vec-len (length moby-vector))
             fsm)

;;; (length *pocket-vector*) is NOT the same as (fsm::index-network fsm)...!!


;;;  Homophones or something ???

            (setq fsm (fsm:make-word-list-fsm
                        (let nil (if (< count vec-len)
                                      (prog1 (svref moby-vector count)
                                          (when nil
                                              (map 'vector #'(lambda (x)
                                                                      (intern (symbol-name x)
                                                                               "USER"))
                                                   (aref moby-vector count)))
                                          (incf count))
                                      nil))
                        :partition-form
                        (format t "Partition number: ~a States so far: ~a~%" fsm::partition-count
                               fsm::total-states-created)))
            (fsm:index-network fsm)
            (fsm:network-to-file fsm fsm-output-file)
            fsm))

(defun make-moby-phone-vector (moby-vector moby-phone-fsm moby-fsm)
       (let ((vec-len (length moby-vector))
             (outfile "/project/markov/phonetic/moby-phone-vector.lisp")
             phone-str phone-lookup-index phones entry)


;;; only needs to be length of *MOBY-PHONE-FSM*

            (setq *pocket-phone-vector* (make-array (fsm:state-name (fsm:network-start-state
                                                                                moby-phone-fsm))
                                                         :initial-element nil))
            (dotimes (index vec-len)
                (setq phones (aref moby-vector index))
                (setq phone-str "")
                (dotimes (str-index (length phones))
                    (setq phone-str (concatenate 'string phone-str (symbol-name (aref phones

str-index)))))
                (when (null (setq phone-lookup-index (fsm:word-to-index phone-str
moby-phone-fsm))
                      (error "shouldnt happen in MAKE-MOBY-PHONE-VECTOR"))
                (setq entry (aref *pocket-phone-vector* phone-lookup-index))
                (if entry
                    (progn (unless (listp entry)
                               (setf (aref *pocket-phone-vector* phone-lookup-index)
                                      (list entry)))
                           (setf (aref *pocket-phone-vector* phone-lookup-index)
                                 (push (fsm:index-to-word index moby-fsm)
                                       (aref *pocket-phone-vector* phone-lookup-index))))
                    (setf (aref *pocket-phone-vector* phone-lookup-index)
                          (fsm:index-to-word index moby-fsm))))
            (write-pocket-vector *pocket-phone-vector* outfile)))

(defun make-moby-vector (fsm)
       (let* ((in-file "/project/markov/phonetic/moby-grolier-overlap.txt")
              (moby-vector-file "/project/markov/phonetic/moby-vector.lisp")
              (nr-fsm-entries (fsm:state-name (fsm:network-start-state fsm)))
              (coded-vector (make-array (list nr-fsm-entries)
                                        :initial-element nil))
              word wd-index)
             (with-open-file (in-stream in-file :direction :input)
                    (do ((line (read in-stream nil nil)
                               (read in-stream nil nil)))
                        ((null line))
                      (when line
                          (setq word (car line))
                          (if word
                              (progn (setq wd-index (fsm:word-to-index word fsm)))
                              (error "Word not recognized in MAKE-MOBY-VECTOR, line: ~A~%" line))
                          (when (null wd-index)
                                (error "Word not recognized in MOBY-fsm,: ~A~%" word))
                          (setf (svref coded-vector wd-index)
                                (second line)))))
             (write-pocket-vector coded-vector moby-vector-file)
             coded-vector))

(defun make-number-list (n)
       (if (> n 0)
           (cons n (make-number-list (- n 1)))
           (list 0)))

(defun make-phone-fsm (phone-vec)
       (let (fsm this-fsm not-done)
            (dotimes (index (length phone-vec))
                (setq not-done t)
                (setq this-fsm (fsm:make-word-list-fsm (let ((word (aref phone-vec index)))
                                                            (if not-done
                                                                (progn (setq not-done nil)
                                                                       word)
                                                                nil))
                                      :print-p nil))
                (setq fsm (if (> index 0)
                              (fsm:concat-fsm fsm this-fsm)
                              this-fsm)))
            fsm))

(defun make-phone-fsms (phone-vec)
       (let ((fsm-vec (make-array 0 :adjustable t :fill-pointer 0)))
            (dotimes (index (length phone-vec))
                (vector-push-extend (make-phone-fsm (aref phone-vec index))
                       fsm-vec))
            fsm-vec))

(defun make-phone-hmm (nr-outputs &key (reset nil)
                             (name *hmm-new-model-name*))
       (let ((center-phone-state (if (= *nr-states-per-phone* 2)
                                     0
                                     (truncate (* 0.5 *nr-states-per-phone*))))
            (nr-states (* *nr-states-per-phone* (length *hmm-state-list*)))
            model saved-level final-states to-prob last-in-chain ith-chain)
            (setq *predict-unknown-words* nil)
            (setq final-states nil)
            (dotimes (chain-index *nr-states-per-phone*)
                (push (+ chain-index (* *nr-states-per-phone* (position :wd-sep
*hmm-state-list*)))
                     final-states))
            (when (or reset (null *transition-matrix*))
                (setq *transition-matrix* (rhmm:my-make-array (list nr-states nr-states)
                                                 :element-type
                                                 'double-float)))


;;; Output matrix is different from hmm, as has extra codebook dimension

            (when (or reset (null *output-matrix*))
                (setq *output-matrix* (rhmm:my-make-array (list 1 nr-states nr-outputs)
                                             :element-type
                                             'double-float :initial-element (double-float 0.0))))
            (when (or reset (null *initial-matrix*))
                (setq *initial-matrix* (rhmm:my-make-array (list nr-states)
                                              :element-type
                                              'double-float)))
            (dotimes (i-state (length *hmm-state-list*))
                (dotimes (chain-index *nr-states-per-phone*)
                    (setf (aref *initial-matrix* (+ chain-index (* *nr-states-per-phone*
i-state)))
                          (/ (double-float 1.0)
                             (double-float (* *nr-states-per-phone* (length
*hmm-state-list*)))))))
            (dotimes (i-state (length *hmm-state-list*))

;;; put self loop on last state for if there are more than 4 letters/phone


;;; self loop is unlikely

                (setq ith-chain (* *nr-states-per-phone* i-state))
                (setq last-in-chain (+ (1- *nr-states-per-phone*)
                                       ith-chain))
                (setf (aref (aref *transition-matrix* last-in-chain)
                            last-in-chain)
                      (double-float 1.0E-10))


;;; forward transitions within a phone

                (when (> *nr-states-per-phone* 1)
                    (dotimes (chain-index (1- *nr-states-per-phone*))
                        (setq to-prob (if (= chain-index (- *nr-states-per-phone* 2))
                                          (/ (double-float 1.0)
                                             (* 10 nr-states))
                                          (/ (double-float 1.0)
                                             nr-states)))
                        (setf (aref (aref *transition-matrix* (+ chain-index ith-chain))
                                    (1+ (+ chain-index ith-chain)))
                              to-prob)))
                (dotimes (j-state (length *hmm-state-list*))


;;; dont loop to same state..phones don't do that

                    (when (/= j-state i-state)
                        (do ((i-chain-index center-phone-state (1+ i-chain-index)))
                            (>= i-chain-index *nr-states-per-phone*))
                          (dotimes (j-chain-index *nr-states-per-phone*)


;;; make transitions before the centre state unlikely

                              (setq to-prob (if (< j-chain-index center-phone-state)
                                                (double-float 1.0E-10)
                                                (/ (double-float 1.0)
                                                   nr-states)))
                              (setf (aref (aref *transition-matrix* (+ i-chain-index ith-chain))
                                          (+ j-chain-index (* *nr-states-per-phone* j-state)))
                                    to-prob))))))
            (setq saved-level rhmm:*report-level*)
            (setq rhmm:*report-level* nil)
            (rhmm:set-model-parameters name nr-states nr-outputs :final-states final-states
                   :allow-incomplete-model t)
            (setq rhmm:*report-level* saved-level)
            (setq model (rhmm:find-model name))
            (setf (rhmm::hmm-defn-outputs model)
                  *output-matrix*)
            (setf (rhmm::hmm-defn-transitions model)
                  *transition-matrix*)
            (setf (rhmm::hmm-defn-initial model)
                  *initial-matrix*)
            (setf (rhmm::hmm-defn-output-to-state-index model)
                  (rhmm:make-output-to-state-index nr-states nr-outputs *output-matrix*))
            model))

(defun make-pocket-fsm nil (let ((file-name "foot-pocket-no-markers.lisp")
                                 (fsm-output-file "pocket.fsm")
                                 fsm)
                                (when (null *foot-pocket*)
                                      (load file-name))
                                (setq *foot-pocket-to-reduced-hashtable* (make-hash-table
                                                                           :test
                                                                           #'equal))
                                 (dolist (el *foot-pocket-to-reduced*)
                                     (setf (gethash (char (first el)
                                                          0)
                                                 *foot-pocket-to-reduced-hashtable*)
                                           (second el)))
                                 (setq *pocket-head* *foot-pocket*)
                                 (setq fsm (fsm:make-word-list-fsm (let ((word (first (first
*pocket-head*
                                                                                             ))))
                                                                        (setq *pocket-head*
                                                                              (cdr *pocket-head*))
                                                                        (if word
                                                                            (string-downcase word)
                                                                            nil))
                                                  :partition-form
                                                  (format t "Partition number: ~a States so far:
~a~%"
                                                         fsm::partition-count
                                                         fsm::total-states-created)))
                                (fsm:index-network fsm)
                                (fsm:network-to-file fsm fsm-output-file)
                                (when (null fsm)
                                       (format t "WARNING: No FSM was made~%"))
                                fsm))

(defun make-pocket-phone-fsm (pocket-vector)
       (let ((fsm-output-file "pocket-phones.fsm")
             (count 0)
             (vec-len (length pocket-vector))
             fsm)

;;; (length *pocket-vector*) is NOT the same as (fsm::index-network fsm)...!!


;;;  Homophones or something ???

            (setq fsm (fsm:make-word-list-fsm
                       (let nil (if (< count vec-len)
                                    (progl (map 'vector #'(lambda (x)
                                                              (intern (symbol-name x)
                                                                      "USER"))
                                                 (aref pocket-vector count))
                                           (incf count))
                                    nil))
                       :partition-form
                       (format t "Partition number: ~a States so far: ~a~%" fsm::partition-count
                              fsm::total-states-created)))
            (fsm:index-network fsm)
            (fsm:network-to-file fsm fsm-output-file)
            fsm))

(defun make-pocket-phone-vector nil (let ((vec-len (length *pocket-vector*))
                                          (outfile "pocket-phone-vector.lisp")
                                          phone-str phone-lookup-index phones entry)

;;; only needs to be length of *POCKET-PHONE-FSM*

                                         (setq *pocket-phone-vector* (make-array (fsm:state-name
                                                                                  (
fsm:network-start-state
*pocket-phone-fsm*
                                                                                   ))
                                                                           :initial-element
nil))
                                         (dotimes (index vec-len)
                                             (setq phones (aref *pocket-vector* index))
                                             (setq phone-str "")
                                             (dotimes (str-index (length phones))
                                                 (setq phone-str (concatenate 'string phone-str
                                                                        (symbol-name (aref phones
str-index)
```

```
                                                                                ))))
                                              (when (null (setq phone-lookup-index
(fsm:word-to-index
                                                                                    phone-str
*pocket-phone-fsm*
                                                                                    )))
                                                    (error
                                                        "shouldnt happen in
MAKE-POCKET-PHONE-VECTOR"
                                                          ))
                                              (setq entry (aref *pocket-phone-vector*
                                                                phone-lookup-index))
                                              (if entry
                                                  (progn (unless (listp entry)
                                                             (setf (aref *pocket-phone-vector*
                                                                        phone-lookup-index)
                                                                   (list entry)))
                                                         (setf (aref *pocket-phone-vector*
                                                                    phone-lookup-index)
                                                               (push (fsm:index-to-word index
                                                                             *pocket-fsm*)
                                                                     (aref *pocket-phone-vector*
                                                                          phone-lookup-index))))
                                                  (setf (aref *pocket-phone-vector*
phone-lookup-index
                                                             )
                                                        (fsm:index-to-word index *pocket-fsm*))))
                                          (write-pocket-vector *pocket-phone-vector* outfile)))

(defun make-pocket-vector (fsm)
       (let* ((pocket-vector-file "pocket-vector.lisp")
              (nr-fsm-entries (fsm:state-name (fsm:network-start-state fsm)))
              (coded-vector (make-array (list nr-fsm-entries)
                                   :initial-element nil))
              fsm-index word)
             (dolist (el *foot-pocket*)
                 (setq word (filter-word (string-downcase (first el))))
                 (setq fsm-index (fsm:word-to-index word fsm))
                 (setf (aref coded-vector fsm-index)
                       (convert-fp-word (second el))))
             (write-pocket-vector coded-vector pocket-vector-file)
             coded-vector))

(defun make-word-fsm (word-vec)
       (let (fsm (index 0))
            (setq fsm (fsm:make-word-list-fsm (let ((word (if (< index (length word-vec))
                                                              (aref word-vec index)
                                                              nil)))
                                                   (incf index)
                                                   word)
                             :print-p nil))
            (fsm:index-network fsm)
            fsm))

(defun moby-setup (&key (rebuild nil)
                        (reload nil))
       (let ((state-id 0)
             (remake-coded-pocket-vector nil))
            (format t "Initializing TDB and loading Emacs code...")
            (concordance::init-tdb-and-load-emacs-to-franz)
            (format t "Done~%")
            (setq *grolier-fsm-mode* t)
            (when (null *hmm-state-hashtable*)
                (setq *hmm-state-hashtable* (make-hash-table :test #'equal))
                (mapcar #'(lambda (el)
                                 (setf (gethash el *hmm-state-hashtable*)
                                       state-id)
                                 (incf state-id))
                       *hmm-state-list*))
            (setup-phone-class-fsms)
            (if rebuild
                (progn (format t "Making Moby-Grolier FSM...~%")
                       (setq *pocket-fsm* (make-moby-fsm)))
                (when (or reload (null *pocket-fsm*))
                    (format t "Reading Moby-Grolier FSM..")
                    (setq *pocket-fsm* (fsm:network-from-file
                                              "/project/markov/phonetic/moby-grolier.fsm"))
                    (fsm:index-network *pocket-fsm*)
                    (format t "done~%")))
            (if rebuild
                (progn (format t "~%Making Moby Vector..")
                       (setq *pocket-vector* (make-moby-vector *pocket-fsm*))
                       (format t "done~%")
                       (setq remake-coded-pocket-vector t))
                (when (or reload (null *pocket-vector*))
                    (format t "Reading Moby vector..")
                    (setq *pocket-vector* (read-pocket-vector
                                                "/project/markov/phonetic/moby-vector.lisp"))
                    (format t "done~%")
                    (setq remake-coded-pocket-vector t)))
            (if rebuild
                (progn (format t "Making Moby Phone FSM..~%")
                       (setq *pocket-phone-fsm* (make-moby-phone-fsm *pocket-vector*)))
                (when (or reload (null *pocket-phone-fsm*))
                    (format t "Reading Moby PHONE FSM..")
                    (setq *pocket-phone-fsm* (fsm:network-from-file
                                                   "/project/markov/phonetic/moby-phones.fsm"))
                    (format t "done~%")))
            (fsm:index-network *pocket-phone-fsm*)
            (if rebuild
                (progn (format t "~%Making Moby Phone Vector..")
                       (setq *pocket-phone-vector* (make-moby-phone-vector *pocket-vector*
                                                         *pocket-phone-fsm* *pocket-fsm*))
                       (format t "done~%"))
                (when (or reload (null *pocket-phone-vector*))
                    (format t "Reading Moby Phone Vector..")
                    (setq *pocket-phone-vector* (read-pocket-vector

"/project/markov/phonetic/moby-phone-vector.lisp"
                                                      ))
                    (format t "done~%")))))

(defun modified-train-tagger (&key (train-on *default-training-file*)
                                   (iterate 16)
                                   (obs-stem "tr")
                                   (read t)
                                   (from-hmm-file nil)
                                   (to-hmm-file nil)
                                   (after-every 500)
                                   (name *hmm-new-model-name*))
       (let (hmm-pathname (hmm-name name)
                   (model (rhmm:find-model name))
                   (count 0)
                   (eof-val (cons nil nil))
                   obs-name file-name ok delta)


;;; Delete the observation sequence used by the tagger

            (setq rhmm:*training-sets* (delete "CURRENT-SENTENCE" rhmm:*training-sets* :test
                                             #'equal :key #'rhmm::training-defn-name))
            (in-package "RHMM")
            (when (or from-hmm-file (null model))
                (setq hmm-pathname (if from-hmm-file
                                       (make-pathname :defaults from-hmm-file)
                                       (if (pathname-name *hmm-pathname*)
                                           *hmm-pathname*
                                           (merge-pathnames (make-pathname :name hmm-name :type
"hmm"
                                                                  )
                                                  *tagger-pathname*))))
                (format t "Reading RHMM: ~a~%" hmm-pathname)
                (setq model (rhmm:read-hmm hmm-pathname))
                (if model
                    (setq hmm-name (rhmm::hmm-defn-name model))
                    (format t "Could not read rhmm: ~a~%" hmm-pathname)))

            ;;

            ;;  Don't create matrix logs because it is going to get written out, and they take up
a
            ;; lot of space

            ;;
            (when (and model)
                (when read
                    (setq rhmm:*training-sets* nil)
                    (when (null (listp train-on))
                        (setq train-on (list train-on)))
                   (dolist (train-file train-on)
                       (format t "Reading ~a~%" train-file)
                       (with-open-file (stream train-file :direction :input)
                             (do ((obs-vec (read stream nil eof-val)
                                         (read stream nil eof-val)))
                                 ((eq eof-val obs-vec))
                               (setq obs-name (concatenate 'string obs-stem "-"
                                                    (format nil "~a" count)))


;;;


;;; These observation matrices must be changed to be for a single codebook  and each observation
is
;;; converted froma single value to an array of one value as appropriate for a codebook.


;;;
                               (dotimes (index (length obs-vec))


;;; THE DIFFERENCE IS HERE

                                   (when nil
                                       (setf (aref obs-vec index)
                                            (vector (aref obs-vec index)))))
                               (rhmm:set-training-data (list obs-name obs-vec)
                                     :overwrite nil)
                               (incf count)))))
               (in-package "USER")
               (setq rhmm:*epsilon* (double-float 1.0E-15))
               (multiple-value-setq (ok delta)
                      (rhmm:estimate-model hmm-name :all :iterate iterate
:preserve-matrix-structure
                            t :after-every after-every))
               (if (null ok)
                   (format t "Error occurred in training--model aborted~%")
                   (progn (format t "Training completed ok: final delta: ~a~%" delta)
                         (setq file-name (if to-hmm-file
                                            (make-pathname :defaults to-hmm-file)
                                            (merge-pathnames (make-pathname
                                                             :type "hmm" :name
                                                             (concatenate 'string hmm-name
"-"
                                                                    (format nil "~a" (length


rhmm:*training-sets*
                                                                                     ))
                                                                    "s-"
                                                                    (format nil "~a" iterate)
                                                                    "its"))
                                                   *local-pathname*)))
                         (format t "Writing RHMM model to file : ~a~%" file-name)
                         (rhmm:write-hmm file-name hmm-name))))
           t))

(defun next-moby-phone (phone-start line char-buf errorstream)
       (let ((line-len (length line))
             (char-index phone-start)
             (buf-index 0)
             (abort-line nil)
             (the-char (char line phone-start))
             reduced-phone key (ok nil)
             (corrected-oi nil))
           (when (or (char= the-char #\')
                     (char= the-char #\,))
               (incf char-index)
               (when (= char-index line-len)
                   (format errorstream "Premature end of line in :~A~%" line)
                   (setq abort-line t)))
           (when (null abort-line)
               (setq the-char (char line char-index))
               (if (char= the-char #\/)
                   (progn (setq ok nil)
                         (setf (aref char-buf buf-index)
                                  the-char)
                            (incf buf-index)
                            (incf char-index)
                            (loop (when (= char-index line-len)
                                        (return))
                                  (setq the-char (char line char-index))
                                  (setf (aref char-buf buf-index)
                                        the-char)
                                  (incf buf-index)
                                  (incf char-index)
                                  (when (char= the-char #\/)
                                      (setq ok t)
                                      (return)))
                            (when (null ok)
                                (format errorstream "Didn't find last / of phone in :~A~%" line)
                                (setq abort-line t)))
                     (progn (setf (aref char-buf buf-index)
                                  the-char)
                            (incf buf-index)
                            (incf char-index)))
                 (when (null abort-line)
                     (setq key (subseq char-buf 0 buf-index))
                     (when (null (setq reduced-phone (gethash key *moby-hashtable*)))


;;;


;;; Put in acceptor code for common systematic problem with //Oi//


;;;


;;; Acceptor code for correcting common error //Oi// instead of /Oi/

                         (when (string= key "//")
                             (when (and (<= (+ 4 char-index)
                                            line-len))
                                 (when (string= "Oi//" (subseq line char-index (+ 4 char-index)))
                                     (setq char-index (- 4 char-index))
                                     (setq reduced-phone (gethash "/Oi/" *moby-hashtable*))
                                     (setq corrected-oi t))))
                         (if corrected-oi
                             (progn (setq reduced-phone (gethash "/Oi/" *moby-hashtable*)))
                             (progn (format errorstream "~A not in hashtable :~A~%" key line)
                                    (setq abort-line t))))))
            (values reduced-phone char-index abort-line)))

(defun next-word-match (phones-for-words wd-index fsm-vec fsm-vec-index key entry)
       (let* ((fsm-index fsm-vec-index)
              (phone-fsm (aref phones-for-words wd-index))
              (wd-fsm (aref fsm-vec fsm-index))
              (matched (null (fsm:null-fsm-p (fsm:intersect-fsm phone-fsm wd-fsm))))
              (found nil))
             (when matched
                 (incf fsm-index)
                 (if (and (< (1+ wd-index)
                             (length phones-for-words))
                          (< fsm-index (length fsm-vec)))
                     (setq found (next-word-match phones-for-words (1+ wd-index)
                                         fsm-vec fsm-index key entry))
                     (when (= fsm-index (length fsm-vec))
                         (format t "FOUND in ~a ...~a~%" key (first entry))
                         (setq found t)))
                 (when nil (format t "~a ~a~%" wd-index fsm-vec-index)))
             found))

(defun normalize-count-vector nil
       (let (total (wd-sep-index (gethash :wd-sep *hmm-state-hashtable*)))
            (dotimes (state-index (array-dimension *letter-counts-for-states* 2))
                (if (= state-index wd-sep-index)
                    (dotimes (chain-index *nr-states-per-phone*)
                        (setf (aref *letter-counts-for-states* (char-to-op #\Space)
                                    chain-index state-index)
                              (/ 1.0 *nr-states-per-phone*)))
                    (progn (dotimes (chain-index *nr-states-per-phone*)
                                (setq total 0.0)
                                (dotimes (op-index (array-dimension *letter-counts-for-states* 0))
                                    (incf total (aref *letter-counts-for-states* op-index
chain-index
                                                      state-index)))
                                (dotimes (op-index (array-dimension *letter-counts-for-states* 0))
                                    (setf (aref *letter-counts-for-states* op-index chain-index
                                                state-index)
                                          (/ (aref *letter-counts-for-states* op-index chain-index
                                                   state-index)
                                             total))))))))

(defun
 offset-scoring
 (result-vector interval-wd-vec nr-results doc-id-hashtable)
 (let ((last-id nil)
       curr-id el this-vec wd-copy this-id)
      (dotimes (index nr-results)
          (setq el (svref result-vector index))
          (setq curr-id (svref el 0))


;;; Don't do needless insertions, and use the hash value as the first appearance (via list
pointer)
;;; of the doc-id in *IDS-AND-OFFSETS* which we know will be ordered with all instances of a
doc-id
;;; togther in a sequence

          (when (null (eq last-id curr-id))
              (setf (gethash curr-id doc-id-hashtable)
                    (position curr-id result-vector :key #'(lambda (x)
                                                                  (svref x 0)))))
          (setq last-id curr-id))
      (maphash #'(lambda (key value)
                        (tdb::do-offsets
                         (id-var offset key concordance::*tdb*)
                         (let ((start-intervals-for-id (gethash id-var doc-id-hashtable))
                               interval-wds)
                              (when start-intervals-for-id
                                  (do* ((interval-index start-intervals-for-id (1+
interval-index)))
                                       ((or (= interval-index nr-results)
                                            (/= id-var (svref (svref result-vector
interval-index)
                                                              0))))
                                     (setq this-vec (svref result-vector interval-index))
                                     (setq this-id (svref this-vec 0))
                                     (when (and (>= offset (svref this-vec 1))
                                                (< offset (svref this-vec 2)))
                                         (setq interval-wds (svref interval-wd-vec
interval-index))
                                         (if (null interval-wds)
                                             (setf (svref interval-wd-vec interval-index)
                                                   (list (copy-seq key)))
                                             (progn (setq wd-copy (copy-seq key))
                                                    (push wd-copy (svref interval-wd-vec
                                                                     interval-index)))))))))))
             *hypo-hashtable*)))

(defun offsets (word doc)
       (let nil (tdb::do-offsets (id-var offset word concordance::*tdb*)
                       (let nil (when (or (= 1 doc)
                                          (= id-var doc))
                                      (format t "Doc: ~A  Offset: ~A~%" id-var offset))))))

(defun old-construct-query (expr-list proximity show show-wds)
       (let ((expr-index 0)
             (conjunct-strs (make-array (length expr-list)
                                  :initial-element nil))
             (conjunct-ids (make-array (length expr-list)
                                  :initial-element
                                  (make-array 50 :adjustable t :fill-pointer 0)))
             (rval nil)
             query-str stemmed-term word-present term-str count coded-word word-hit fsm)
            (setq query-str " )")
            (if (hash-table-p *hypo-hashtable*)
                (clrhash *hypo-hashtable*)
                (setq *hypo-hashtable* (make-hash-table :test #'equal)))
            (dolist (expr expr-list)
                (when show (format t "~%expr: ~a~%" expr))
                (setq fsm (p-expr-to-fsm expr))
                (setq term-str ") ")
                (setq count 0)
                (setq word-present nil)
                (dotimes (wd-index (length *coded-pocket-vector*))
                    (setq coded-word (svref *coded-pocket-vector* wd-index))
                    (when (eq :accepted (fsm:apply-fsm-to-array coded-word fsm))
                        (setq word-hit (fsm:index-to-word wd-index *pocket-fsm*))
                        (setq stemmed-term (twol-stemmer::lookup-stem (copy-seq word-hit)))
                        (if (or (gethash word-hit *stop-hashtable*)
                                (gethash stemmed-term *stop-hashtable*))
                            (when show (format t "~a on stop list -- ignored~%" word-hit))
                            (progn (setq word-present t)
                                   (push word-hit (aref conjunct-strs expr-index))
                                   (vector-push-extend (vector wd-index 0)
                                          (aref conjunct-ids expr-index))
                                   (setq term-str (concatenate 'string word-hit " " term-str))


;;; Shouldn't put words not recognized by  groliers in here !!!! --Must not insert nils into
;;; *HYPO-HASHTABLE*

                                    (when (null stemmed-term)
                                        (setq stemmed-term (copy-seq word-hit)))
                                    (setf (gethash stemmed-term *hypo-hashtable*)
                                          0)
                                    (when (or show show-wds)
                                        (format t "~D: ~12,3T ~D (~D)~%" word-hit (aref

*pocket-vector*
                                                                                          wd-index)
                                               (tdb::term-freq word-hit concordance::*tdb*)))
                                    (incf count)))))
                (when (and word-present (eql 1 (length expr-list)))
                       (setq rval t))
                (when (> (length expr-list)
                         1)
                    (setq rval (nconc rval (list (if word-present
                                                     t
                                                     nil)))))
                (setq term-str (concatenate 'string "( " term-str))
                (setq query-str (concatenate 'string term-str query-str))
                (when (or show show-wds)
                       (format t "Nr hits : ~a~%" count))
                (incf expr-index))
            (setq query-str (concatenate 'string "(" (format nil "~d " proximity)
                                    query-str))
            (when show (format t "Query : ~a~%" query-str))
            (values query-str conjunct-ids rval)))

(defun op-to-char (op)
       (let ((the-char nil)
             (a-charcode (char-code #\a)))
            (if (null (numberp op))
                (error "ERROR: ~a not a number~%" the-char)
                (if (and (> op 0)
                         (<= op (1+ (- (char-code #\z)
                                       a-charcode))))
                    (setq the-char (code-char (+ (1- op)
                                                  a-charcode)))
                    (if (= op 0)
                        (setq the-char #\Space)
                        (error "ERROR: ~a is invalid output~%" op))))
            the-char))

(defun opc (reduced-phone)
       (let (char (state-id (gethash reduced-phone *hmm-state-hashtable*)))
            (dotimes (index *nr-outputs*)
                (setq char (if (= index 0)
                               #\Space
                               (code-char (+ (1- index)
                                              (char-code #\a)))))
                (format t "~a: " char)
                (dotimes (chain-index *nr-states-per-phone*)
                    (format t " ~14a" (aref *letter-counts-for-states* index chain-index
state-id)))
                (format t "~%"))))

(defun
 p-atomic-fsm
 (expr)
 (let
  ((returned-fsm nil)
   hash-fsm
   (sym-fsm (fsm:sigma-star-fsm)))
  (typecase expr
       (string (if *grolier-fsm-mode*
                   (if *use-confusion-classes*
                       (aref *confusion-class-fsms* (gethash (intern (symbol-name expr)
                                                                      "KEYWORD")
                                                             *hmm-state-hashtable*))
                       (setq returned-fsm (fsm:sigma-fsm (gethash (intern (symbol-name expr)
                                                                           "KEYWORD")
                                                                  *hmm-state-hashtable*))))
                   (setq returned-fsm (make-word-fsm (vector expr)))))
       (number (error "numbers not allowed in make-atomic-fsm"))
       (symbol (if (setq hash-fsm (if *grolier-fsm-mode*
                                       (if *use-confusion-classes*
                                           (aref *confusion-class-fsms* (gethash (intern
(symbol-name
                                                                                                expr)
                                                                                              "KEYWORD")

*hmm-state-hashtable*))
                                           (fsm:sigma-fsm (gethash (intern (symbol-name expr)
                                                                           "KEYWORD")
                                                                  *hmm-state-hashtable*)))
                                       (gethash (string-downcase (symbol-name expr))
                                             *phone-hashtable*)))
                   (setq returned-fsm hash-fsm)
                   (if (eq :any (intern (symbol-name expr)
                                       "KEYWORD"))
                       (setq returned-fsm sym-fsm)
                       (if (eq :one (intern (symbol-name expr)
                                           "KEYWORD"))
                           (setq returned-fsm *any-phone*)
                           (if (eq :vow (intern (symbol-name expr)
                                               "KEYWORD"))
                               (setq returned-fsm *vowel-fsm*)
                               (if (eq :son (intern (symbol-name expr)
                                                   "KEYWORD"))
                                   (setq returned-fsm *sonorant-fsm*)
                                   (if (eq :nas (intern (symbol-name expr)
                                                       "KEYWORD"))
                                       (setq returned-fsm *nasal-fsm*)
                                       (if (eq :fric (intern (symbol-name expr)
                                                            "KEYWORD"))
                                           (setq returned-fsm *fricative-fsm*)
                                           (if (eq :stop (intern (symbol-name expr)
                                                                "KEYWORD"))
                                               (setq returned-fsm *stop-fsm*)
                                               (error "~s not correct*~%" expr)))))))))))
  returned-fsm))

(defmacro p-binary-op (op expr)
       `(,op (p-expr-to-fsm (cadr ,expr))
              (p-expr-to-fsm (caddr ,expr))))

(defun p-convert-sequence (lst)
       (loop for expr in lst collect (p-expr-to-fsm expr)))

(defun p-expr-to-fsm (expr)


;;; Converts A Regular Expression To A Finite State Machine. Special Symbols: Iteration = *
Positive
;;; Iteration = + Complementation = ~ Intersection = & Union = / Concatenation = ! -- Can be
;;; omitted. Difference = -

       (let nil (fsm::make-connected (if (null (listp expr))
                                           (p-atomic-fsm expr)
                                           (case (car expr)
                                               ((? optional) (p-unary-op fsm:optional-fsm expr))
                                               ((* star zero-plus) (p-unary-op fsm:zero-plus expr))
                                               ((+ plus one-plus) (p-unary-op fsm:one-plus expr))
                                               ((~ not compl) (p-unary-op fsm:negate-fsm expr))
                                               ((- minus relcompl) (p-binary-op fsm:minus-fsm
expr))
                                               ((& and) (p-n-ary-op fsm:intersect-fsm expr))
                                               ((/ or) (p-n-ary-op fsm:union-fsm expr))
```

```
                                              ((! seq sequence) (p-n-ary-op fsm:concat-fsm expr))
                                              (otherwise (p-n-ary-op fsm:concat-fsm
                                                                (cons '! expr)))))))

(defmacro p-n-ary-op (op expr)
       '(apply #',op (p-convert-sequence (cdr ,expr))))

(defmacro p-unary-op (op expr)
       '(,op (p-expr-to-fsm (cadr ,expr))))

(defun phn (wd)
       (let (wd-index (phones nil))
            (when (setq wd-index (fsm:word-to-index wd *pocket-fsm*))
                (setq phones (aref *pocket-vector* wd-index)))
            phones))

(defun phone-to-char-matches (word phones-for-word stream)
       (let ((from-vec (make-array (length word)
                              :initial-element nil))
             (to-vec (make-array (length word)
                            :initial-element nil))
             (to-index 0)
             (nr-phones (length phones-for-word))
             (from-index 0)
             (last-consonant-index nil)
             min max start-index end-index)
            (dotimes (char-index (length word))

                ;; default is to cover the whole word if things break down subsequently
                (when (gethash (char word char-index)
                             *consonant-hashtable*)
                    (setq from-index (if last-consonant-index
                                         (aref from-vec last-consonant-index)
                                         0))
                    (setf (aref from-vec char-index)
                          from-index)
                    (setq to-index (if last-consonant-index
                                       (aref to-vec last-consonant-index)
                                       (1- nr-phones)))
                    (setf (aref to-vec char-index)
                          to-index)
                    (setq start-index (do* ((phone-index (if (and last-consonant-index
                                                                   (< last-consonant-index
                                                                      (1- char-index)))
                                                              (1+ to-index)
                                                              from-index)
                                                    (1- phone-index)))
                                            ((or (>= phone-index nr-phones)
                                                 (null (gethash (aref phones-for-word phone-index)
                                                              *vowel-hashtable*)))
                                            phone-index)))
                    (when (and (< start-index nr-phones)
                               (null (gethash (aref phones-for-word start-index)
                                            *vowel-hashtable*)))
                        (setq last-consonant-index char-index)
                        (setf (aref from-vec char-index)
                              start-index)
                        (setf (aref to-vec char-index)
                              start-index)
                        (when *debugl* (format t "~a Char-index ~a start-index ~a~%" word
char-index
                                              start-index))
                        (setq end-index (do ((phone-index start-index (1+ phone-index)))
                                             ((or (= phone-index nr-phones)
                                                  (gethash (aref phones-for-word phone-index)
                                                         *vowel-hashtable*))
                                             phone-index)))
                        (when (or (= end-index nr-phones)
                                  (and (< end-index nr-phones)
                                       (> end-index 0)
                                       (gethash (aref phones-for-word end-index)
                                               *vowel-hashtable*)))
                            (setf (aref to-vec char-index)
                                  (1- end-index))
                            (when *debugl* (format t "end-index ~a~%" end-index))))
                    (when *debugl* (format t "~a  ~a~%~a~%from-vec: ~a~%to-vec: ~a~%" word
char-index
                                          phones-for-word from-vec to-vec))))
            (setq min nil)
            (dotimes (char-index (length word))
;;;

;;; For other letters (vowels and vowel-like consonants) they extend over left consonants and
next
;;; set of rightmost consonants, thus match ing consonants vowels and right consonants

;;;
                (if (aref from-vec char-index)
                    (setq min (aref from-vec char-index))
                    (setf (aref from-vec char-index)
                          (or min 0))))
            (setq max nil)
            (do ((char-index (1- (length word))
                        (1- char-index)))
                ((< char-index 0))
              (if (aref to-vec char-index)
                  (setq max (aref to-vec char-index))
                  (setf (aref to-vec char-index)
                        (or max (max 0 (1- nr-phones))))))
            (when *debugl* (format t "~%FROM: ~a~%TO: ~a~%" from-vec to-vec))
            (dotimes (char-index (length word))
                (format stream "#(~a" (char-to-op (char word char-index)))
                (do ((phone-index (aref from-vec char-index)
                            (1+ phone-index)))
                    ((> pnone-index (aref to-vec char-index)))
                  (format stream " ~a" (gethash (aref phones-for-word phone-index)
                                                *hmm-state-hashtable*)))
                (format stream ")~%" (char-to-op #\Space)))))

(defun phones (key)
       (let (entry)
            (if key
                (when (setq entry (gethash key *file-nashtable*))
                    (format t "~a~%" (second entry)))
                (maphash #'(lambda (dummy entry)
                                   (format t "~a~%~%" (second entry)))
                       *file-hashtable*))))

(defun pq (query)
       (let ((fsm-vec (make-array 0 :adjustable t :fill-pointer 0)))
            (dolist (wd-match query)
                (format t "making fsm for : ~a~%" wd-match)
                (vector-push-extend (p-expr-to-fsm wd-match)
                       fsm-vec))
            (apply-phone-fsm fsm-vec)))

(defun print-interval-hits (howmany)
       (let nil (dotimes (index howmany)
                    (format t "Doc: ~6D  start: ~6D  end: ~6D  wds: ~A~%" (svref (svref
*query-results*
                                                                                          index)
                                                                                   0)
                            (svref (svref *query-results* index)
                                   1)
                            (svref (svref *query-results* index)
                                   2)
                            (svref *interval-wd-vec* index)))))

(defun read-either-file (pathname)
       (let (end token (return-vec (make-array 0 :adjustable t :fill-pointer 0)))
            (with-open-file (stream pathname :direction :input)
                   (do* ((start (read stream nil nil)
                                (read stream nil nil)))
                        ((null start))
                      (setq end (read stream nil nil))
                      (setq token (read-line stream nil nil))
                      (vector-push-extend (vector start end token)
                             return-vec)))
            return-vec))

(defun read-file (file-name)
       (let (phones-for-a-word wd-start wd-end triple phn-triple phone-start phone-end
                   (phone-vec (make-array 0 :adjustable t :fill-pointer 0))
                   (word-vec (make-array 0 :adjustable t :fill-pointer 0))
                   (phones-for-words (make-array 0 :adjustable t :fill-pointer 0)))
            (setq word-vec (read-either-file (make-pathname :defaults file-name :type "wrd")))
            (setq phone-vec (read-either-file (make-pathname :defaults file-name :type "phn")))
            (dotimes (wd-index (length word-vec))
                (setq triple (aref word-vec wd-index))
                (setq wd-start (aref triple 0))
                (setq wd-end (aref triple 1))
                (setq phones-for-a-word (make-array 0 :adjustable t :fill-pointer 0))
                (dotimes (phone-index (length phone-vec))
                    (setq phn-triple (aref phone-vec phone-index))
                    (setq phone-start (aref phn-triple 0))
                    (setq phone-end (aref phn-triple 1))
                    (when (and (> phone-end wd-start)
                               (< phone-start wd-end))
                        (vector-push-extend (aref phn-triple 2)
                               phones-for-a-word)))
                (vector-push-extend phones-for-a-word phones-for-words))
            (dotimes (index (length word-vec))
                (setf (aref word-vec index)
                      (aref (aref word-vec index)
                            2)))
            (values word-vec phones-for-words)))

(defun read-letter-counts (&key (file-name *letter-count-file*))
       (let nil (with-open-file (stream file-name :direction :input)
                       (setq *letter-counts-for-states* (read stream)))
            nil))

(defun read-moby-form (grolier-overlap line line-number output-buf char-buf outstream
errorstream)
       (let ((op-index 0)
             (line-len (length line))
             (the-char (aref line 0))
             (char-index 0)
             (end-of-wd 0)
             (phone t)
             (phone-end -1)
             (phone-list nil)
             (ignore nil)
             (abort-line nil)
             line-was-aborted grolier-freq downcase-wd)
            (do nil
                ((or (= char-index line-len)
                     (char= the-char #\Space)))
              (incf char-index)
              (setq the-char (aref line char-index)))
            (when (= char-index line-len)
                (format errorstream "Word not terminated on line ~D: ~A~%" line-number line)
                (setq abort-line t))
            (setq end-of-wd char-index)
            (when (null abort-line)
                (dotimes (index char-index)
                    (setf (aref output-buf index)
                          (aref line index)))
                (setf (aref output-buf char-index)
                     #\Space)
                (setq op-index (1+ char-index))
                (do* ((phone-start (1+ char-index)
                             phone-end))
                     ((or (null phone)
                          abort-line
                          (>= phone-start line-len)))
                   (multiple-value-setq (phone phone-end line-was-aborted)
                          (next-moby-phone phone-start line char-buf errorstream))
                   (when (or line-was-aborted (null phone))
                       (when nil (format errorstream "Phone not terminated on line ~D: ~A~%"
                                        line-number line))
                       (setq abort-line t))
                   (when (null abort-line)
                       (if (string-equal phone "wd-sep")
                           (progn (when grolier-overlap
                                        (setq ignore t)
                                        (incf *nr-hyphenations-ignored*))
                                  (dotimes (index (length " - "))
                                      (setf (aref output-buf (+ index op-index))
                                            (char " - " index)))
                                  (incf op-index (length " - ")))
                           (if (string-equal phone "foreign")
                               (progn (when nil (format t "~A has foreign phone... ignored~%"
line))
                                      (incf *nr-foreign-words*)
                                      (setq ignore t))
                               (if (null (string-equal phone "skip"))
                                   (progn (when grolier-overlap
                                              (push (intern phone "KEYWORD")
                                                    phone-list))
                                          (dotimes (index (length phone))
                                              (setf (aref output-buf (+ index op-index))
                                                    (char phone index)))
                                          (incf op-index (length phone))
                                          (setf (aref output-buf op-index)
                                                #\Space)
                                          (incf op-index))))))
                (if abort-line
                    (incf *nr-format-errors*)
                    (when (null ignore)
                        (if grolier-overlap
                            (progn (setq downcase-wd (nstring-downcase (subseq line 0
end-of-wd)))
                                   (setq grolier-freq (tdb::term-freq downcase-wd
concordance::*tdb*)
                                         )
                                   (when (> grolier-freq 0)
                                       (incf *nr-grolier-wds*)
                                       (format outstream "(~S #(" downcase-wd)
                                       (dolist (el (nreverse phone-list))
                                           (format outstream " ~A" el))
                                       (format outstream "))~%")))
                            (format outstream "~A~%" (subseq output-buf 0 op-index)))))))

(defun read-pocket-vector (file-name)
       (let ((vector nil)
             length)
            (when (probe-file file-name)
                (with-open-file (stream file-name :direction :input :if-does-not-exist nil)
                       (setq length (read stream))
                       (setq vector (make-array (list length)))
                       (dotimes (i length)
                           (setf (svref vector i)
                                 (read stream)))))
            vector))

(defun score-groups (group-vec nr-results show-max-docs max-doc-search)
       (let ((doc-id-hashtable (make-hash-table :test #'equal))
             phonetic-score score best-so-far group wd-list nr-title-words ranked-list
nr-total-hits
             doc-id doc-vec score doc-list doc-len doc-title)
            (setq *group-vec* group-vec)


;;;


;;; re-use DOC-ID-HASHTABLE for efficiency while scoring


;;; Maybe too many entries for us to want to deal with... remake it  as above


;;;


;;; Do scoring within a group.

            (dotimes (index (min nr-results (length group-vec)))
                (setq group (svref group-vec index))
                (setq doc-list (svref group 0))
                (dolist (doc-id doc-list)
                    (when (null (gethash doc-id doc-id-hashtable))
                        (setq doc-len (tdb::doc-length doc-id concordance::*tdb*))
                        (setq doc-title (tdb::doc-title doc-id concordance::*tdb*))


;;; The 3rd element (initialized to nil) will indicate whether this title has already been shown
```

```
to
;;; the user, in which case it is not repeated.


;;; The 4th element is to maintain best-so-far, used below

                        (setf (gethash doc-id doc-id-hashtable)
                             (vector doc-len doc-title nil nil)))))
            (setq ranked-list nil)
            (dotimes (index (min nr-results (length group-vec)))
                (setq group (svref group-vec index))
                (setq wd-list (svref group 1))
                (setq doc-list (svref group 0))
                (setq nr-total-hits (svref group 2))


;;; The number of total hits are not used currently, neither are the documents ranked by their
;;; scores...

                (when nil


;;; Is not good from a retrieval standpoint to rank primarily in terms of number of overall
;;; recurrences of a group


;;;

                    (format t "~A: ~%" wd-list))
                (do* ((tail doc-list (cdr tail))
                      (doc-id (car tail)
                             (car tail))
                      (display-count 0 (1+ display-count)))
                     ((or (null tail)
                          (= display-count max-doc-search)))
                   (setq doc-vec (gethash doc-id doc-id-hashtable))
                   (setq doc-len (svref doc-vec 0))
                   (setq doc-title (svref doc-vec 1))
                   (setq nr-title-words (find-title-words wd-list doc-title))
                   (setq score 0.0)
                   (setq phonetic-score 0.0)


;;; Should also base the score on the nr co-occurrences WITHIN the article

                   (dolist (word wd-list)
                       (incf phonetic-score (gethash word *hypo-hashtable*))
                       (incf score (+ (* 50.0 (/ (float (get-doc-freq word doc-id))
                                                            doc-len))
                                      (* 10.0 nr-title-words))))


;;; use NR-TOTAL-HITS also now

                   (incf score (/ nr-total-hits 10.0))


;;; Then multiply by phonetic score sum

                   (setq score (* score phonetic-score))
                   (when nil
                       (setq best-so-far (find doc-id ranked-list :key #'(lambda (x)
                                                                                (svref x 1)))))
                   (setq best-so-far (svref doc-vec 3))
                   (if (null best-so-far)
                       (progn (setq rank-vec (vector score doc-id wd-list))
                              (push rank-vec ranked-list)
                              (setf (svref doc-vec 3)
                                   rank-vec))
                       (when (> score (svref best-so-far 0))
                           (setf (svref best-so-far 0)
                                score)
                           (setf (svref best-so-far 2)
                                wd-list)))))
            (setq ranked-list (sort ranked-list #'> :key #'(lambda (x)
                                                                  (svref x 0))))
            (do* ((tail ranked-list (cdr tail))
                  (the-vec (car tail)
                         (car tail))
                  (display-count 0 (1+ display-count)))
                 ((or (null tail)
                      (= display-count show-max-docs)))
               (setq doc-id (svref the-vec 1))


;;; Don't redisplay any title that was displayed before

               (setq doc-vec (gethash doc-id doc-id-hashtable))
               (when (null (svref doc-vec 2))
                   (setf (svref doc-vec 2)
                         t)
                   (setq score (svref the-vec 0))
                   (setq doc-len (svref doc-vec 0))
                   (setq doc-title (svref doc-vec 1))
                   (if (null *use-nl-buf*)
                       (format t "~6D ~A ~6,2F ~A~%" doc-id doc-title score (svref the-vec 2))
                       (concordance::print-in-nl-buf (format nil "~A  ~22,1T~A~%" doc-title
                                                             (svref the-vec 2)))))))

(defun score-hits (heap)
       (let ((internal-heap (pq::pq-heap heap))
             (heap-len (pq::pq-length heap))
             (doc-id-hashtable (make-hash-table :test #'equal))
             item doc-id)
            (setq *temp-hash-table* doc-id-hashtable)
            (dotimes (index heap-len)
                (setq item (svref internal-heap index))
                (setq doc-id (svref item 0))
                (when nil (format t "inserting ~A in doc id hashtable~%" doc-id))
                (setf (gethash doc-id doc-id-hashtable)
                      0))
            (maphash #'(lambda (key value)
                              (tdb::do-freqs (id-var freq key concordance::*tdb*)
                                     (when (gethash id-var doc-id-hashtable)
                                         (when nil (format t "Doc: ~A term: ~A~%" id-var key))
                                         (incf (gethash key *hypo-hashtable*)))))
                   *hypo-hashtable*)))

(defun setup-grolier-mode (&optional (rebuild nil))
       (let ((state-id 0))
            (setq *grolier-fsm-mode* t)
            (when (null *hmm-state-hashtable*)
                (setq *hmm-state-hashtable* (make-hash-table :test #'equal))
                (mapcar #'(lambda (el)
                                 (setf (gethash el *hmm-state-hashtable*)
                                       state-id)
                                 (incf state-id))
                       *hmm-state-list*))
            (setup-phone-class-fsms)
            (if rebuild
                (setq *pocket-fsm* (make-pocket-fsm))
                (when (null *pocket-fsm*)
                    (format t "Reading Pocket FSM..")
                    (setq *pocket-fsm* (fsm:network-from-file
"/project/markov/phonetic/pocket.fsm"))
                    (fsm:index-network *pocket-fsm*)
                    (format t "done~%")))
            (if rebuild
                (setq *pocket-vector* (make-pocket-vector *pocket-fsm*))
                (when (null *pocket-vector*)
                    (format t "Reading Pocket vector..")
                    (setq *pocket-vector* (read-pocket-vector
                                                 "/project/markov/phonetic/pocket-vector.lisp"))
                    (format t "done~%")))
            (if rebuild
                (setq *pocket-phone-fsm* (make-pocket-phone-fsm *pocket-vector*))
                (when (null *pocket-phone-fsm*)
                    (format t "Reading Pocket PHONE FSM..")
                    (setq *pocket-phone-fsm* (fsm:network-from-file

"/project/markov/phonetic/pocket-phones.fsm"))
                    (format t "done~%")))
            (fsm:index-network *pocket-phone-fsm*)
            (if rebuild
                (setq *pocket-phone-vector* (make-pocket-phone-vector))
                (when (null *pocket-phone-vector*)
                    (format t "Reading Pocket Phone Vector..")
                    (setq *pocket-phone-vector* (read-pocket-vector
"/project/markov/phonetic/pocket-phone-vector.lisp"
                                                        ))
                    (format t "done~%")))
            (when (null *coded-pocket-vector*)
                (format t "Making Coded Pocket vector..")
                (setq *coded-pocket-vector* (make-array (length *pocket-vector*)))
                (dotimes (index (length *pocket-vector*))
                    (setf (aref *coded-pocket-vector* index)
                           (map 'vector #'(lambda (x)
                                                 (gethash x *hmm-state-hashtable*))
                              (aref *pocket-vector* index))))
                (format t "done~%"))))

(defun setup-phone-class-fsms nil
       (let nil (setq *any-phone* (fsm:sigma-fsm (make-number-list (length *hmm-state-list*))))
            (if *use-confusion-classes*
                (progn (setq *confusion-class-fsms* (make-array (length *hmm-state-list*)))
                        (dolist (class *confusion-matrix-map*)
                            (setf (aref *confusion-class-fsms* (gethash (car class)
                                                                      *hmm-state-hashtable*))
                                  (fsm:sigma-fsm (map 'list #'(lambda (x)
                                                                      (gethash x
*hmm-state-hashtable*)
                                                                      )
                                                      (second class))))))
                (setq *confusion-class-fsms* nil))
            (setq *vowel-fsm*
                  (fsm:sigma-fsm (map 'list #'(lambda (x)
                                                      (gethash x *hmm-state-hashtable*))
                                      '(:iy :ih2 :eh :ae :ah2 :uw2 :uh :ao2 :er2 :ey :ay :oy :aw
:ow)
                                      )))
            (setq *sonorant-fsm* (fsm:sigma-fsm (map 'list #'(lambda (x)
                                                                      (gethash x
*hmm-state-hashtable*)
                                                                      )
                                                      '(:l2 :r :y :w))))
            (setq *nasal-fsm* (fsm:sigma-fsm (map 'list #'(lambda (x)
                                                                   (gethash x
*hmm-state-hashtable*))
                                                   '(:m2 :n2 :ng2))))
            (setq *fricative-fsm*
                  (fsm:sigma-fsm (map 'list #'(lambda (x)
                                                      (gethash x *hmm-state-hashtable*))
                                       '(:f :v :th :dh :hh2 :ch :jh :s :z :sh2))))
            (setq *stop-fsm* (fsm:sigma-fsm (map 'list #'(lambda (x)
                                                                  (gethash x
*hmm-state-hashtable*))
                                                  '(:sil2-b :sil2-d :sil2-g :sil2-p :sil2-t
:sil2-k)))
                  )))

(defun show-doc-titles (limit print-limit)
       (let ((nr-docs (length concordance::*ids-and-offsets*))
             (sort-vec (make-array limit :adjustable t :fill-pointer 0))
             next-id curr-id title-str title-hits (hits 0))
            (format t "~A Document matches:~%" nr-docs)
            (do ((tail concordance::*ids-and-offsets* (cdr tail))
                  (index 0 (1+ index)))
                 ((or (null tail)
                      (>= index limit)))
               (setq curr-id (aref (car tail)
                                   0))
               (setq next-id (and (cdr tail)
                                  (aref (car (cdr tail))
                                        0)))
               (if (eql next-id curr-id)
                   (incf hits)
                   (progn (setq title-str (tdb::doc-title curr-id concordance::*tdb*))
                          (setq title-hits (get-title-hits title-str))
                          (vector-push-extend (cons (+ (* 5 title-hits)
                                                        (1+ hits))
                                                    (car tail))
                                 sort-vec)
                          (setq hits 0))))
            (sort sort-vec #'> :key #'car)
            (dotimes (index (min print-limit (length sort-vec)))
                (setq curr-id (aref (cdr (aref-sort-vec index))
                                     0))
                (setq title-str (tdb::doc-title curr-id concordance::*tdb*))
                (setq title-hits (get-title-hits title-str))
                (format t "~&~5D: ~A  Score: ~A Title wds: ~A~%" curr-id title-str
                       (car (aref sort-vec index))
                       title-hits))
            (when (> nr-docs limit)
                (format t "..... ~a more titles.~%" (- nr-docs limit)))))

(defun train-phone-tagger (&key (full-init t)
                                (dr-dirs '("dr1" "dr2" "dr3" "dr4" "dr5" "dr6" "dr7" "dr8"))
                                (verbose t)
                                (hmm-name "phon")
                                (train t)
                                (to-codes nil)
                                (build nil)
                                (read-timit nil)
                                (read-codes nil)
                                (add-speaker-variants t)
                                (train-on *letter-codes-file*)
                                (iterate 8)
                                (reset t)
                                (zap-vowels t)
                                (after-every 500)
                                (obs-stem "tr")
                                (from-hmm-file nil)
                                (to-hmm-file nil))
       (let ((model nil)
             (cfg (list (list hmm-name nil :top-level))))
            (setq *debug* nil)
            (when (or read-timit full-init)
                (format t "Reading TIMIT files~%")
                (get-files :dr-dirs dr-dirs :verbose verbose :add-speaker-variants
                       add-speaker-variants))
            (when (or build full-init)
                (format t "Making Phone HMM...")
                (setq model (make-phone-hmm *nr-outputs* :reset reset :name hmm-name))
                (format t "done~%")
                (when nil
                    (format t "Building CFG...")
                    (rhmm::build-cfg cfg)
                    (format t "done~%")))
            (when (or to-codes full-init)
                (format t "Coding TIMIT and building letter counts...")
                (code-timit)
                (format t "done~%"))
            (when (and model (or build full-init))
                (when (and (null full-init)
                           (null *letter-counts-for-states*))
                    (format t "Reading letter count vector...")
                    (read-letter-counts)
                    (format t "done~%"))
                (format t "Initializing HMM output vector...")
                (fill-phone-hmm-op-vec (or model (rhmm:find-model hmm-name)))
                (format t "done~%")
                (when zap-vowels
                    (format t "Zapping vowels in consonant states~%")
                    (zap-vowels-in-other-states hmm-name)
                    (format t "done~%"))
                (when (or to-codes full-init)
                    (format t "Writing letter count vector...")
                    (write-letter-counts)
                    (format t "done~%")))
            (when (or train full-init)
                (format t "Training...~%")
                (when t
                    (modified-train-tagger :name hmm-name :read (or full-init read-codes)
                           :train-on train-on :iterate iterate :obs-stem obs-stem :from-hmm-file
                           from-hmm-file :to-hmm-file to-hmm-file))
                (when nil
                    (train-grammar :cfg hmm-name :train-on train-on :iterate iterate
:re-estimate-ops
                           t :read (or full-init read-codes)
                           :from-cfg-file from-hmm-file :to-cfg-file-stem to-hmm-file :limit
100000
                           :after-every after-every :max-len 70 :min-len 1 :auto-exit nil)))))

(defun wds (phone-list-or-vec &optional (end (length phone-list-or-vec)))
       (let (sym-name)
            (setf (fill-pointer *wds-buffer*) -
                  0)
            (etypecase phone-list-or-vec
                (array (dotimes (index end)
                           (setq sym-name (symbol-name (aref phone-list-or-vec index)))
                           (dotimes (char-index (length sym-name))
                               (vector-push-extend (char sym-name char-index)
                                      *wds-buffer*))))
                (list (dolist (el phone-list-or-vec)
                          (setq sym-name (symbol-name el))
                          (dotimes (char-index (length sym-name))
                              (vector-push-extend (char sym-name char-index)
                                     *wds-buffer*)))))
            (aref *pocket-phone-vector* (fsm:word-to-index *wds-buffer* *pocket-phone-fsm*))))

(defun words (key)
       (let (entry)
            (if key
                (when (setq entry (gethash key *file-hashtable*))
                    (format t "~a~%" (first entry)))
                (maphash #'(lambda (dummy entry)
                                  (format t "~a~%" (first entry)))
                       *file-hashtable*))))

(defun write-letter-counts (&key (file-name *letter-count-file*))
       (let nil (with-open-file (stream file-name :direction :output :if-exists :new-version)
                      (print *letter-counts-for-states* stream))
            nil))

(defun write-pocket-vector (array file-name)
       (let nil (with-open-file (stream file-name :direction :output :if-exists :new-version)
                      (print (length array)
                             stream)
                      (dotimes (i (length array))
                          (print (aref array i)
                                 stream)))
            nil))

(defun zap-vowels-in-other-states (model-name)
       (let* ((model (rhmm:find-model model-name))
              (output-matrix (rhmm::hmm-defn-outputs model))
              (vowels '(#\a #\e #\i #\o #\u #\y))
              (consonant-states '(:l2 :r :m2 :n2 :ng2 :ch :jh :s :z :sh2 :f :v :th :dh :hh2 :b :d
:g
                                      :p :t :k :dx :sil2 :sil2-b :sil2-d :sil2-g :sil2-p :sil2-t
                                      :sil2-k))
              (state-id 0))
             (when (null *hmm-state-hashtable*)
                 (setq *hmm-state-hashtable* (make-hash-table :test #'equal))
                 (mapcar #'(lambda (el)
                                  (setf (gethash el *hmm-state-hashtable*)
                                        state-id)
                                  (incf state-id))
                        *hmm-state-list*))
             (dolist (consonant consonant-states)
                 (dolist (vowel vowels)
                     (dotimes (chain-index *nr-states-per-phone*)
                         (setf (aref (aref (aref output-matrix 0)
                                           (+ chain-index (* *nr-states-per-phone* (gethash
consonant

*hmm-state-hashtable*
                                                                                          ))))
                                     (char-to-op vowel))
                               (double-float 1.0E-10)))))))
```

Attorney Docket No. 13188-71
Xerox Docket No. D/93288

SEMANTIC CO-OCCURRENCE FILTERING SOFTWARE PROGRAM

SOURCE CODE FILE #2

THIS FILE INCLUDES CODE FOR:
HYPOTHESIS GENERATION

```
;;; Given the output of a phonetic recognizer, this program generates
;;; the possible strings which could have led to the recognizer output.
;;;  This is done based on the confusion, insertion and deletion data
;;; generated from tests done on TIMIT using the SPHINX alignment program.

;;; like genphones12.lisp
;;; - fixes the global storage in bs properly
;;; - uses structs for the nodes so that floats are dealt with properly
;;; - adds a lot of float declarations
;;; - contains a ton of new defparameters and defvars to get rid of
;;;   compiler warning
;;; - no longer sets use-confusion-classes
;;; - added a minimum confusion os vowels with vowels
;;; - allocates global storage for the pointerlists in the node data structure

;;; pulls in the heap code from Doug Cutting
(use-package :priority-queue)


;;; convenient utilities
(defun lf () (load "/project/ssp/vijay/aristotle/genphones16.lisp"))
(defun cc () (compile-file "/project/ssp/vijay/aristotle/genphones16.lisp"))
(defun ll () (load "/project/ssp/vijay/aristotle/genphones16.f41"))
(defun cl ()
  (cc)
  (ll))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; global declarations
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;; $$$@@@$$$
(defvar *phonelist* nil)
(defvar  *PhoneCountList* nil)
(defvar *total* nil)
(defvar *totdels* nil)
(defvar *delprob* nil)
(defvar *DelList* nil)
(defvar *DelCountList* nil)
(defvar *SubList* nil)
(defvar *TotSubCount* nil)
(defvar *LContDel* nil)
(defvar *RContDel* nil)
(defvar *LContIns* nil)
(defvar *RContIns* nil)
(defvar *LContSub* nil)
(defvar *RContSub* nil)
(defvar *PairCount* nil)
(defvar *totpairs* nil)

;; $$@@$$

(defstruct (nd)
  (node-id nil)
  (mutations nil)
  (prob 0.0 :type single-float)
  (ptrs nil)

  )
;;; this is to prevent Julian's code from further confusing my confused
;;; phones.... don't need to do this any more
;;; (defparameter *use-confusion-classes* NIL)

;;; set this to be true when groliers has been loaded so that the
;;; recog-eval loop will know that it can call grolier's without
;;; fear of crashing
(defparameter *groliers-loaded* T)

;;; load my versions of some some of the functions in Julian's code
;;; I've fixed them up to run a little differently as better suits
;;; the automatic recog-eval loop
;;; (load "/tilde/vijayb/Speech/aristotle/grolfixup.lisp")


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;           GLOBAL VARIABLES
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; read in the confusion/insertion/deletion data produced by the
;;; counts perl script in /tilde/vijayb/Speech/aristotle/scripts
;;; the file probsout defines the variables:
;;; *phonelist* = array of the phones in the order they appear in arrays
;;; *PhoneCountList* = number of times each phone was recognized
;;; *total* = total number of phones in the training set
;;; *totdels* = total number of deletions in the training set
;;; *delprob* = context independent probability of deletion
;;; *DelList* = array of context-independent probabilities of deletion of phones
;;; *DelCountList* =  number of times each phone was deleted
;;; *SubList* = array of context-independent probabilities of substitutions
;;; *TotSubCount* = total number of times each observed phone was a substitution
;;; *LContDel* = number of times each phone was deleted given the left context
;;; *RContDel* =number of times each phone was deleted given the right context
;;; *LContIns* =number of times each phone was inserted given the left context
;;; *RContIns* =number of times each phone was inserted given the right context
;;; *LContSub* =number of times each phone was subst'ed given the left context
;;; *RContSub* =number of times each phone was subst'ed given the right context
;;; *PairCount* = number of times each pair of phones occured in the data
;;; *totpairs* = total number of pairs in the data
;;;  The phones in the arrays  are listed in the order in *phonelist*
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; I'm currently reading in data in terms of the phoneset that's presented
;;; to the dictionary

(defparameter *basedir* "/tilde/vijayb/Speech/aristotle/")

(defun add-base (filename)
  (concatenate 'string *basedir* filename))


(defparameter *timit-propfile* "/tilde/vijayb/Speech/aristotle/probsout.mod.timit")
(defparameter *don-probfile* "/tilde/vijayb/Speech/aristotle/probsout.mod.don")
(defparameter *confusiondir* "/tilde/vijayb/Speech/aristotle/confusion/")
(defun addconfdir (file) (concatenate 'string *confusiondir* file))
(defparameter user-list
  (list
        (cons 'don (addconfdir  "probsout.mod.don"))
       (cons 'timit (addconfdir"probsout.mod.timit"))
       (cons 'vijay (addconfdir "probsout.mod.vijay"))
       (cons 'test  (addconfdir "probsout.test"))
       (cons 'julian (addconfdir "probsout.mod.julian"))
       ))
(defparameter *probfile* (cdr (assoc 'julian user-list)))
(defparameter *user* 'julian)
;;; the next linbe has been replaced by (load-probfile) below
;;; (load *probfile*)



;;; Global declarations
(defvar *storage-array* NIL)
(defvar *pointer-storage* NIL)



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; The probability arrays are coerced to make sure that they are
;;; floats
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defun coerce-probs ()
;;;  (declare (special *delprob*))
  (coerce-DelList)
  (coerce-SubList)
  (setf *delprob* (float *delprob*)))

(defun coerce-DelList ()
;;;  (declare (special *phonelist*)
;;;          (special *DelList*))
  (dotimes (count (length *phonelist*))
    (setf (aref *DelList* count) (float (aref *DelList* count)))))

(defun coerce-SubList ()
;;;  (declare (special *phonelist*)
;;;          (special *SubList*))
    (let ((length (length *phonelist*)))
      (dotimes (count1 length)
        (dotimes (count2 length)
          (setf (aref *SubList* count1 count2)
            (float (aref *SubList* count1 count2))))
        (setf (aref *SubList* count1 length)
          (float (aref *SubList* count1 length)))))




;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; Code to smooth the substitution probabilities by setting up
;;; a minimum conufusion probability for some phone classes
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defvar *vowels*
    '(IY IH2 EH AE AH2 UW2 UH AO2 ER2 EY AY OY
      AW OW))

(defparameter *minsub* 0.001)

;;; smooth the substitution probabilities by
;;; setting the confusion probabilities between
;;; vowels to be *minsub*
(defun smooth-vowels ()
  (dolist (vowel *vowels*)
    (dolist (v2 *vowels*)
      (let ((vowelindex (index vowel))
            (v2index (index v2)))
        (if (< (aref *SubList* vowelindex v2index)
              *minsub*)
            (progn (setf (aref *SubList* vowelindex v2index) *minsub*)
                  (setf (aref *SubList* vowelindex vowelindex)
                    (-  (aref *SubList* vowelindex vowelindex) *minsub*)))))))))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; code to load in the confusion data
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defun load-probfile (probfile)
  (load probfile)
  ;;; make sure thatt the probabilities are floats
  (coerce-probs)
  ;;; smooth the vowel confusion probabilities
  (smooth-vowels)
  )

;;; load in the statistics
(load-probfile *probfile*)




;;; *phonelist* is actually an array.... convert it into a list
(defun convert-to-list (thearray)
  (let ((l (length thearray))
       (result '()))
    (dotimes (i l)
      (setf result (cons (aref thearray i) result)))
    (reverse result)))

(defparameter *ListOfPhones*   (convert-to-list *phonelist*))


;;; the string of phones for which are trying to find likely antecedents
(defparameter *phonestring* '())
;;; the file where phoestring can be found
(defparameter *phonefile* "/tilde/vijayb/Speech/aristotle/phonein")
(defparameter *localpipefile* "/tilde/vijayb/Speech/aristotle/phonepipein")
(defparameter *remotepipefile* "/tmp/phonepipein")


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;  Code to generate context dependent deletion probabilities from
;;;  the counts read in from the probsout file
;;; .... remember to update initworld to call these functions also...
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;;; Find the context dependent deletion probabilities.
;;; Pr(b was deleted| left = a, right = c)
;;;




(defun deletion-prob (leftindex rightindex phoneindex
                      &key ((:level printlevel) 0))
;;;  (declare (special *DelList*)
;;;         (special *delprob*))
  (if (> printlevel 5)
      (progn (princ leftindex) (princ " ")
             (princ rightindex)))
  (* (aref *DelList* phoneindex) *delprob*))


(defun print-dels (leftindex rightindex)
;;;  (declare (special *phonelist*))
  (dotimes (count (length *phonelist*))
    (print (context-deletion-prob leftindex rightindex count))))


(defun print-idels (leftindex rightindex)
;;;  (declare (special *phonelist*))
  (dotimes (count (length *phonelist*))
    (print (deletion-prob leftindex rightindex count))))



(defun print-subs-to (phone)
  (declare (special *phonelist*))
  (dotimes (count (length *phonelist*))
    (princ (aref *phonelist* count))
    (princ "->")
    (princ phone) (princ ": ")
    (princ (aref *SubList* (index (aref *phonelist* count))
                 (index phone)))
    (fresh-line))
  'DONE)

(defun print-subs (phone)
  (declare (special *phonelist*))
  (dotimes (count (length *phonelist*))
    (princ phone)
    (princ "->")
    (princ (aref *phonelist* count)) (princ ": ")
    (princ (aref *SubList*  (index phone) (index (aref *phonelist* count))))
    (fresh-line))
  'DONE)



;;; I am not combining things properly because I do in principle know the
;;; pair probability of the contexts.   I have collected the counts
;;; necessary to find this and it is known at the same level of confidence
;;; as the left and right context dependent deletion probabilities.
;;; nonetheless, I press on
(defun cdp (x y z &key ((:level level) 0))
  (context-deletion-prob x y z :level level))

(defun context-deletion-prob (leftindex rightindex phoneindex
                     &key ((:phonelist phonelist) *phonelist*)
                          ((:PhoneCount PhoneCount) *PhoneCountList*)
                          ((:PairCount PairCount) *PairCount*)
                          ((:level printlevel) 0))

;;;  (declare (special *phonelist*)
;;;        (special *PhoneCountList*)
;;;        (special *PairCount*)
;;;        (special *LContDel*)
;;;        (special *RContDel*))

  ;;; if neither is NIL use the info theory formula to combine contexts
  ;;; if both are NIL....?
  ;;; temporarily return the context independent deletion prob
;;;  (* (aref *DelList* phoneindex) *delprob*)

  (if (= leftindex -1)
      (float (/ (aref *RContDel* rightindex phoneindex)     (aref PhoneCount rightindex)))
    (if (= rightindex (length phonelist))
        (float (/ (aref *LContDel* leftindex phoneindex)
                  (aref PhoneCount leftindex)))
      (let* ((leftcount (aref PhoneCount leftindex))
             (rightcount (aref PhoneCount rightindex))
             (lcprob (float (/ (max (deletion-prob leftindex rightindex phoneindex)
                                    (aref *LContDel* leftindex phoneindex)) leftcount)))
             (rcprob (float (/ (max (deletion-prob leftindex rightindex phoneindex)
                                    (aref *RContDel* rightindex phoneindex)) rightcount)))
             ;;; context-independent deletion probability of the phone
             (pprob (deletion-prob leftindex rightindex phoneindex))
             (lprob (theprob leftindex :phonecount PhoneCount))
             (rprob (theprob rightindex :phonecount PhoneCount))
             (mindel 0.00001))

        (if (> printlevel 2)
            (progn (princ "leftcount: ") (princ leftcount) (fresh-line)
                   (princ "rightcount: ") (princ rightcount)(fresh-line)
                   (princ "lcprob: ") (princ lcprob)(fresh-line)
                   (princ "rcprob: ") (princ rcprob)(fresh-line)
                   (princ "pprob: ") (princ pprob)(fresh-line)
                   (princ "rprob: ") (princ rprob)(fresh-line)
                   (princ "lprob: ") (princ lprob)(fresh-line)
                   (princ "indep: ") (princ  (deletion-prob leftindex rightindex
phoneindex))(fresh-line)
                   (princ "lcont: ") (princ (aref *LContDel* leftindex phoneindex))(fresh-line)
                   (princ "rcont: ") (princ (aref *RContDel* rightindex phoneindex))(fresh-line)))

        (if (= pprob 0)
            mindel
          (let ((factor1 (/ (* lcprob rcprob lprob rprob) pprob)))
            (if (= factor1 0)
                mindel
              (let ((factor2 (+ (/ (* (- 1 lcprob) (- 1 rcprob) lprob rprob)
                                   (- 1 pprob))
                                factor1)))
                (/ factor1 factor2))))))))



;;; context independent probability that a given phone occurs
(defun theprob (phoneindex &key ((:phonecount PhoneCount) *PhoneCountList*))
;;;  (declare (special *total*)
;;;           (special *PhoneCountList*))
  (float (/ (aref PhoneCount phoneindex)
            *total*)))




;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; Methods of selectively changing the insertion/substitution/
;;; deletion probabilities
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;; tells you the probability that the observed phone "name" is deleted
;;; the input phone is phone put out by the recognizer
(defun what-del (name)
;;;  (declare (special *phonelist*)
;;;           (special *delprob*)
;;;           (special *DelList*))
  (let ((index (position name *phonelist*)))
    (if (null index)
        (progn (princ "Phone ") (princ name) (princ " not found.")
               (fresh-line))
      (progn (princ "Dictionary phone name = ") (princ (find-phone index))
             (fresh-line)
             (princ "Context independent deletion probability = ")
             (princ *delprob*) (fresh-line)
             (princ "Relative deletion probability = ")
             (princ (aref *DelList* index)) (fresh-line)
             (aref *DelList* index)))))

;;; gives the probability that the recognized phone hypname is a
;;; substitution for the true phone truename.   hypname and truename
;;; must be phones from the phone set used by the recognizer.
(defun what-sub (hypname truename)
;;;  (declare (special *phonelist*)
;;;        (special *SubList*))
  (let ((hypindex (position hypname *phonelist*))
        (trueindex (if (eq truename '?)
                       (length *phonelist*)
                     (position truename *phonelist*))))
    (if (or (null hypindex) (null trueindex))
        (progn (princ "what-sub: Phone not found.") (fresh-line))
      (progn (let ((subprob (aref *SubList* hypindex trueindex)))
               (princ "Dictionary phone names:") (fresh-line)
               (princ "    ") (princ hypname) (princ " = ")
               (princ (find-phone hypindex)) (fresh-line)
               (princ "    ") (princ truename) (princ " = ")
               (if (not (eq truename '?))
                  (princ (find-phone trueindex))) (fresh-line)
               (princ "Probability of the substituion ")
               (princ hypname) (princ "->") (princ truename)
               (princ " = ") (princ subprob) (fresh-line)
               subprob)))))


;;; sets the probability that the recognized phone hypname is a
;;; substitution for the true phone truename.   Hypname and truename
;;; should be phonelabels from the phone set used by the recxognizer.
(defun set-sub (hypname truename value)
;;;  (declare (special *phonelist*)
;;;        (special *SubList*))
  (if (or (> value 1) (< value 0))
      (error "set-sub: Hey... value should be a probability."))
  (if (eq truename hypname)
      (error "set-sub: Can't set the self substitution probability yet."))
  (let ((hypindex (position hypname *phonelist*))
        (trueindex (if (eq truename '?)
                       (length *phonelist*)
                     (position truename *phonelist*))))
    (if (or (null hypindex) (null trueindex))
        (progn (princ "Phone not found.") (fresh-line))
      (let* ((old-val (aref *SubList* hypindex trueindex))
             (diff (- value old-val))
             (self (aref *SubList* hypindex hypindex))
             (newself (- self diff)))
        (if (or (> newself 1)
               (< newself 0))
            (error "set-sub: value is too large or small"))
        (setf (aref *SubList* hypindex trueindex) value)
        (setf (aref *SubList* hypindex hypindex) newself)
        (what-sub hypname truename)))))




(defun setdelprob (val)
;;;  (declare (special *delprob*))
  (setf *delprob* val))

(defun op-sub-prob (val op)
;;;  (declare (special *phonelist*)
;;;        (special *SubList*))
  (let ((result (make-array (list (length *phonelist*)
                                  (+ (length *phonelist*) 1)))))
    ;;; The (+ 1 ...) is necessary cause insertions are treated
    ;;; as substitutions of NULL
    (dotimes (i (length *phonelist*))
      (let ((sum 0))
        (dotimes (j (+ 1 (length *phonelist*)))
          (if (not (= i j))
              (let ((new (apply op (list val (aref *SubList* i j)))))
               (setf (aref result i j) new)
               (setf sum (+ sum new))))
          )
        (if (> sum 1)
            (progn
              (princ "Error on *SubList* row:  ")
```

```
               (princ i) (fresh-line)
               (error
                "Hey.... I thought these were supposed to be probabilities")))
         (setf (aref result i i) (- 1 sum))))
     result))


(defun add-sub-prob (val)
  (op-sub-prob val #'+))

(defun scale-sub-prob (val)
  (op-sub-prob val #'*))



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;  newphonemap = BUSH set with closures
;;;  modphonemap = BUSH set with closures mapped to corresponding
;;;                consonants
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; the file where the map from TIMIT to our phones is found
(defvar *phonemapdir* "/tilde/vijayb/Speech/aristotle/dictionaries/")
(defun addphonedir (file) (concatenate 'string *phonemapdir* file))
(defvar *newphonemap* (addphonedir "newphonemap"))
(defvar *modphonemap* (addphonedir "modphonemap"))
;;; (setf *phoneconvertmap* (addphonedir "modphonemap.2"))
(defvar *modmap* (with-open-file (infile *modphonemap* :direction :input)
               (read infile)))
;;; (setf *conversionmap* (with-open-file (infile *phoneconvertmap* :direction :input)
;;;            (read infile)))
(defvar *phonemapfile* "nonexistant")
(defvar *phonemap* '())


;;; these are the phones that Julian uses I think
;;; (defvar *FOOT-POCKET-TO-REDUCED*
;;;     (QUOTE (("p" :SIL2-P) ("t" :SIL2-T) ("k":SIL2-K) ("b" :SIL2-B)
;;;         ("d" :SIL2-D) ("g" :SIL2-G) ("C" :CH) ("J" :JH) ("s" :S)
;;;         ("S" :SH2) ("z" :Z) ("Z" :SH2) ("f" :F) ("T" :TH) ("v" :V)
;;;         ("D" :DH) ("h" :HH2) ("n" :N2) ("m" :M2) ("G" :NG2) ("N" :N2)
;;;         ("M" :M2) ("L" :L2) ("l" :L2) ("r" :R) ("w" :W) ("y" :Y)
;;;         ("i" :IY) ("I" :IH2) ("E" :EH) ("e" :EY) ("@" :AE) ("a" :AO2)
;;;         ("W" :AW) ("Y" :AY) ("^" :AH2) ("c" :AO2) ("O" :OY) ("o" :OW)
;;;         ("U" :UH) ("u" :UW2) ("R" :ER2) ("x" :AH2) ("|" :IH2) ("X" :ER2))))



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;        HOW TO GET THE PHONE STRING FOUND BY THE RECOGNIZER
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;; ways to get at the phones that the recognizer found
(defun get-phone (&optional filename)
;;;  (declare (special *phonefile*))
  (if (not filename)
      (setf filename *phonefile*))
  (with-open-file (infile filename
                  :if-does-not-exist :error)
    (read infile)))

;;; set phonestring to be the phone string in filename
(defun setup-string (&optional filename)
;;;  (declare (special *phonestring*)
;;;        (special *insmax*)
;;;        (special *delprob*))
  (setf *phonestring* (get-phone filename))
;;; Maximum number of allowed insertions in order to correct the
;;; deletions in Method 1
  (setf *insmax* (* (length *phonestring*) *delprob*)))


;;; set up the initial phonestring
 (setf *phonestring*
  (when (open *phonefile* :direction :probe
              :if-does-not-exist NIL)
     (print "Reading in the current phone...")
    (get-phone)))

;;; (setf *phonestring* '(SIL2-B IH2 R))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;   THE RECOGNIZER PUTS OUT TIMIT PHONES.   THE probs ANALYSIS
;;;   USES THE SPHINX ALIGNMENT PROGRAM TO PULL OUT CONFUSION
;;;   STATISTICS FOR THESE TIMIT PHONES.   THIS SECTION CONTAINS
;;;   THE CODE NECESSARY FOR MAPPING THE RECOGNIZER PHONES
;;;   TO THE PHONES USED IN THIS  PROJECT.   THE PHONEMAP
;;;   CAN BE FOUND IN THE FILE newphonemap
;;; Well.... I now do the translap of phonemaps offline in a perl
;;; script... so I wind up passing the null phonemap to find-phone
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;; get the phonemap
(defun get-phonemap (&optional filename)
;;;  (declare (special *phonemapfile*))
  (if (not filename)
      (setf filename *phonemapfile*))
  (with-open-file (infile filename
                   :if-does-not-exist :error)
    (read infile)))


;;; set up the phonemap
(defun setup-phonemap (&optional filename)
;;;  (declare (special *phonemap*))
  (setf *phonemap* (get-phonemap filename)))


;;; set up the initial phonemap
(setf *phonemap*
  (when (open *phonemapfile* :direction :probe
              :if-does-not-exist NIL)
    (print "Reading in the current phonemap...")
    (get-phonemap)))


;;; translate a scorefile phone into our phones
;;; if we pass this the null phonemap then dont translate
;;; the phone at all and return the scorefile phone
(defun translate (phone &key ((:phonemap phonemap) *phonemap*) )
;;;  (declare (special *phonemap*))
  (if phonemap
      (cadr (assoc phone phonemap))
    phone))

;;; find the phone label corresponding to the phone
;;; with index i in *phonelist*
(defun find-phone (index &key ((:phonemap phonemap) *phonemap*))
;;;  (declare (special *phonemap*)
;;;         (special *phonelist*);
  (translate (aref *phonelist* index)
             :phonemap phonemap))





;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; METHOD 1: correct substitutions, insertions and deletions
;;; that occur with probability greater than some threshold.
;;; Use that to generate a candidate phone regexp for Julian's
;;; finite state machines
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;; file to write result to
(defparameter *fileout* "phoneout")

;;; The threshold probability above which we will allow substitutions
(defparameter subthresh 0.04)

;;; The threshold probability above which we will repair deletion
(defparameter delthresh 0.03)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;        DEALING WITH DELETIONS
;;; We treat deletions as context independent and insert phones
;;; to optionally recover the antecedents of an oberved phone
;;; string
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defun dels (threshold &key ((:phonemap phonemap) *phonemap*))
;;;  (declare (special *phonemap*)
;;;        (special *phonelist*)
;;;        (special *DelList*))
  (let ((result '()))
    (dotimes (i (length *phonelist*))
      (if (> (aref *DelList* i) threshold)
          (setf result (cons (find-phone i :phonemap phonemap) result))))
    (if result
        (list '? (cons '/ result))
      result)))

(defparameter *deletions*
  (dels delthresh))

(defun setdels (threshold &key ((:phonemap phonemap) *phonemap*))
  (declare (special *phonemap*)
           (special *deletions*))
  (setf *deletions* (dels threshold
                          :phonemap phonemap)))

;;; Maximum number of allowed insertions in order to correct the
;;; deletions
(defparameter *insmax* (* (length *phonestring*) *delprob*))




;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; BUILD THE REGEXP FOR THE PHONESTRINGS FROM WHICH THE RECOGNIZER
;;; MAY HAVE DERIVED  THE OBSERVED PHONE STRING
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;; find the antecedent phone strings without accounting for
;;; the bound on the number of insertions allowed in order to
;;; correct deletions
;;; If delt > 0, deletions will be set to (dels delt)
(defun confuse (&key ((:string string) *phonestring*)
                    ((:deletions deletions) *deletions*)
                    ((:sub sub) subthresh)
                    ((:delt delt) -5)
                    ((:phonemap phonemap) *phonemap*)
               &allow-other-keys)
;;;  (declare (special *phonestring*)
;;;        (special *deletions*)
;;;        (special *phonelist*)
;;;        (special *phonemap*)
;;;        (special subthresh))
    (if (>= delt 0)
        (setf deletions (dels delt)))
    (dolist (i string)
      (if (not (position i *phonelist*))
          (error "The recognizer is using unkown phone labels.")))
    (let ((result (if deletions
                      (list deletions)
                    '())))
      (dolist (i (reverse string))
        (setf result (cons (extend-phone i :sub sub
                                           :phonemap phonemap) result)))
        (if deletions
            (setf result (cons deletions result))
          result))
      result))


;;; confuse the string without inserting anything to find
;;; the antecedents of erroneous deletions
;;; set delt to be so high that no deletions will be included
(defun vconfuse (&rest rest &key ((:delt delt) 500) &allow-other-keys)
  (setf rest (cons :delt (cons delt rest)))
  (apply 'confuse rest))
```

```
;;; extend each phone to its list of possible antecedents
(defun extend-phone (phone &key ((:sub sub) subthresh)
                              ((:phonemap phonemap) *phonemap*))
;;;  (declare (special *phonelist*)
;;;         (special subthresh)
;;;         (special *phonemap*)
;;;         (special *SubList*))
  (let ((index (position phone *phonelist*))
        (result '())
        (max (length *phonelist*)))
    (dotimes (i max)
      (if (> (aref *SubList* index i) sub)
          (setf result (cons (find-phone i :phonemap phonemap) result))))
    (setf result (cons '/ result))
    (if (> (aref *SubList* index max) sub)
        (setf result (list '? result)))
    result))


(defun confuseout (&rest rest &key ((:out out) *fileout*) &allow-other-keys)
;;;  (declare (special *fileout*))
  (with-open-file (outfile out :direction :output)
    (write (apply 'confuse rest) :stream outfile)))



;;; find the antecedent phone strings but bound the
;;; the number of insertions allowed in order to
;;; correct deletions
;;; If delt > 0, deletions will be set to (dels delt)

(defun bconfuse (&rest rest &key ((:string string) *phonestring*)
                      ((:deletions deletions) *deletions*)
                      ((:sub sub) subthresh)
                      ((:delt delt) -5)
                      ((:bound bound) (floor *insmax*))
                 &allow-other-keys)

  (let ((result (apply 'vconfuse rest)))
    (generate result deletions bound)))


;;; We want to put in up to bound insertions.  Since all the
;;; insertions are optional, putting in bound insertions will make sure
;;; that less than bound insertions are taken care of also.
;;; I have to
(defun generate (alist object bound)
  (cond ((= bound 0) alist)
       ((and (not alist) (> bound 0))
        (cons object (generate alist object (- bound 1))))
       (T  (let ((res1 (generate (cdr alist) object bound))
                 (res2 (generate alist object (- bound 1))))
             (list 'or
                  (if (eq (car res1) 'or)
                      (list (car alist) res1)
                    (cons (car alist) res1))
                  (if (eq (car res2) 'or)
                      (list object res2)
                    (cons object res2)))))))




;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;         Method 2  - n-best
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;  - separate substitution and deletion mutation lists that are kept
;;;    sorted by earlier action.   This wwil make building the final
;;;    string easier


;;;; (defun in-dictionary (word)
;;;   (grol word)
;;;  T
;;;  )



;;;; implementation of the phone string data structure
(defun insert-phone (phone string position)
  (if (> position (length string))
      (error "Position too large.")
    (progn
      (let ((front (subseq string 0 position))
            (back (nthcdr position string)))
        (concatenate 'list front (list phone) back)))))

(defun sub-phone (newphone string position)
  (if (or (= position 0)
          (> position (length string)))
      (error "Invalid position.")
    (progn
      (let ((front (subseq string 0 (- position 1)))
            (back (nthcdr position string)))
        (concatenate 'list front (list newphone) back)))))



;;; implementation of substitution/deletion list data structure
;;; A substitution or deletion list is a list of pairs.   The
;;; car of the pair is the probability of the substitution or the
;;; deletion.   The cdr of the pair is the actual substitution or
;;; deletion.   The substitution/deletion list must be sorted by
;;; probability.
(defun prob (actionpair)
  (or (car actionpair)
      0.0))


(defun action (actionpair)
  (cdr actionpair))

;;; substitution or deletion
(defun action-type (action)
  (car action))

;;; which phone to act on
(defun action-index (action)
  (cadr action))

;;; parameter required for the action.   E.g, what
;;; the substituted phone should be
(defun action-param (action)
  (caddr action))


;;; action acting on an earlier phone
(defun earlier-action (action1 action2)
  (< (action-index action1)
        (action-index action2))
      )

;;; substitutions are said to have precedence over deletions
(defun precedence (action1 action2)
  (if (and (eq (action-type action1) 'SUB)
           (eq (action-type action2) 'DEL))
      T
    NIL))


(defun most-likely-action (actionlist)
  (car actionlist))

(defun more-likely-pair (pair1 pair2)
   (> (the single-float (prob pair1)) (the single-float (prob pair2))))

(defun create-actionlist (alist)
  (sort alist #'more-likely-pair))

;;; create a list of possible substitutions of a phone
;;; leave out the self substitution
(defun create-sublist (phone index &key ((:phonelist phonelist) *phonelist*))
  (declare (special *SubList*)
          (special *phonelist*))
  (let ((phoneindex (position phone phonelist))
        (result NIL)
        (max (length phonelist)))
    (dotimes (i max)
      (if (not (= i phoneindex))
          (setf result (cons (cons (aref *SubList* phoneindex i)
                                 (list 'SUB index (find-phone i)))
                            result)))
      )

    (setf result (cons (cons (aref *SubList* phoneindex max)
                            (list 'SUB index NIL))
                      result))
    (setf result (create-actionlist result))
    result))


(defun create-dellist (leftphone rightphone index
                      &key ((:phonelist phonelist) *phonelist*)
                           ((:level printlevel) 0)
                           ((:indep indep) NIL))
  (let ((leftindex (position leftphone *phonelist*))
        (rightindex (position rightphone *phonelist*))
        (result NIL)
        (max (length phonelist)))
    (if (eq leftindex NIL)
        (setf leftindex -1))
    (if (eq rightindex NIL)
        (setf rightindex (length phonelist)))
    (dotimes (i max)
      (setf result (cons (cons
                         (if indep
                             (deletion-prob leftindex rightindex i :level printlevel)
                           (context-deletion-prob leftindex rightindex i :level
                                                   printlevel))
                              (list 'DEL index (find-phone i)))
                       result)))
    (setf result (create-actionlist result))
    result))



;;; implementation of the node data structure

(defun make-node (nodeid mutationlist prob pointers)
  (let ((result (vector nodeid mutationlist prob pointers)))
    result))

;;; $$@@$$

(defun make-nd-special (nodeid mutationlist prob pointers storage)
(declare (single-float prob))
  (let ()
    (setf (nd-node-id storage) nodeid)
    (setf (nd-mutations storage) mutationlist)
    (setf (nd-prob storage) prob)
    (setf (nd-ptrs storage) pointers)
    storage))

;;(defun make-node-special (nodeid mutationlist prob pointers storage)
;;  (if (= (length storage) 4)
;;      (progn (setf (svref storage 0) nodeid)
;;            (setf (svref storage 1) mutationlist)
;;            (setf (svref storage 2) prob)
;;            (setf (svref storage 3) pointers)
;;            storage)
;;    (error "make-node-special: storage should be an array of length 4.")))

(defun nodeid (node)
    (nd-node-id node))


(defun nodemutations (node)
    (nd-mutations node))
```

```
(defun nodeprob (node)
  (the single-float (nd-prob node)))


(defun pointers (node)
    (nd-ptrs node))

(defun setpointerpart (node newpointers)
  (setf (nd-ptrs node) newpointers))


;;; returns the probability of the most likely action pointed to in the
;;; pointer list
(defun top (node)
  (the single-float (likelihood (most-likely-pointer (pointers node)))))


;;; the most likely node for the next action is the one whose top
;;; action probability times the node probability is the greatest
(defun more-likely-node (node1 node2)
  (> (* (the single-float (top node1)) (the single-float (nodeprob node1)))
     (* (the single-float (top node2)) (the single-float (nodeprob node2)))))



;;; perform the best action on the node. Performs the action, destructively
;;; modifying the node's internal parameters
;;; returns a child node
(defun mutate (node storage pointer-storage)
  (let* ((prob 0.0)
        (pointers (copy-pointers pointer-storage (pointers node)))
        (best (most-likely-pointer pointers))
        (action (pointer-action best))
        (type (action-type action))
        (oldmutations (nodemutations node))
        (newmutations (cons action oldmutations))
        (newprob 0.0)
        (newpointers (if (eq type 'DEL)
                        pointers
                      (if (eq type 'SUB)
                           (null-best-pointer pointers))))
        (newnodespecial nil)

        )
    (declare (single-float newprob prob))

    (setq prob (the single-float (top node)))
;;;     (princ "Top: ") (princ (top node)) (fresh-line)
;;;     (princ "nodeprob: ") (princ (nodeprob node))
;;;     (fresh-line)
    (setq newprob (* prob
                    (the single-float (nodeprob node))))
    (if (position action oldmutations :test #'equal)
       (setq newprob (* newprob 0.1)))

    (setq newnodespecial (make-nd-special  (+ (nodeid node) 1)
                                         newmutations
                                         newprob
                                         newpointers
                                          storage))
    ;;; remember that step-best-pointer destructively modifies the
    ;;; thing it sorts
    (setpointerpart node (step-best-pointer (pointers node)))
    newnodespecial))



;;; code to fix the pointer structure of child nodes following deletion
;;; mutations to reflect the penalty for multiple deletions at the same
;;; position




;;; implementation of the pointers data structure
;;; pointers = (list of pointers into substiution and deletion list data
;;;     structures sorted by probability of first action pointed to)
(defun most-likely-pointer (pointers)
  (car pointers))

(defun rest-pointers (pointers)
  (cdr pointers))

(defun likelihood (pointer)
  (the single-float (prob (most-likely-action pointer))))

(defun pointer-action (pointer)
  (action (most-likely-action pointer)))

(defun more-likely-pointer (pointer1 pointer2)
  (>= (the single-float (likelihood pointer1))
     (the single-float (likelihood pointer2))))


(defun sort-pointer-list (alist)
  (sort alist #'more-likely-pointer))


(defun insert-pointer (pointer pointerlist)
  (if (more-likely-pointer pointer
                          (most-likely-pointer pointerlist))
      (cons pointer pointerlist)
    (progn (insert-pointer-aux pointer pointerlist 0)
          pointerlist)))

(defun insert-pointer-aux (pointer pointerlist count)
  (if (more-likely-pointer pointer (most-likely-pointer
                                   (cdr pointerlist)))
      (rplacd pointerlist (cons pointer (cdr pointerlist)))
    (insert-pointer-aux pointer (cdr pointerlist) (+ count 1))))



(defun increment-pointer (pointer)
  (cdr pointer))


;;; this should be using an efficient sort that exploits the dact that
;;; pointerlist is already sorted.
(defun add-pointer (pointer pointerlist)
;;;  (sort-pointer-list (cons pointer pointerlist))
  (insert-pointer pointer pointerlist))

;;; this should be using an efficient sort that exploits the
(defun step-best-pointer (pointers)
  (add-pointer (increment-pointer (most-likely-pointer pointers))
               (rest-pointers pointers)))


;;; set the best pointer to null... equivalently from the point of
;;; view of functionality, remove it from the list
(defun null-best-pointer (pointers)
  (rest-pointers pointers))


;;; copy pointers for one list into anotheer which has the storage already
;;; allocated
(defun copy-pointers (storage-list pointerlist)
  (let ((front storage-list))
    (dotimes (i (length storage-list))
      (rplaca storage-list (car pointerlist))
      (setf storage-list (cdr storage-list))
      (setf pointerlist (cdr pointerlist)))
    front))






;;; implement the nodelist abstraction
;;; (defun  most-likely-node (nodelist)
;;;   (car nodelist))
```

```
(defun  most-likely-node (nodelist)
  (pq:pq-top nodelist))


;;; this does not work anymore now that we have nodelist as a heap
;;; (defun rest-nodes (nodelist)
;;;   (cdr nodelist))

;;; sort-nodelist is not needed anymore cause be use a heap
;;; (defun sort-nodelist (nodelist)
;;;  (sort nodelist #'more-likely-node))


(defun insert-node (node nodelist)
  ;;; rather than using sort-nodelist this should use an efficient
  ;;; sort that exploits the fact that nodelist is already oredered
  ;;; (sort-nodelist (cons node nodelist))
  (pq:pq-insert node nodelist)
  )



;;; used after the best node has been mutated... this should be called
;;; to update the nodelist order... it currently iuses sort nodelist
;;; but should in fact use an efficient sort that uses the fact that
;;; nodelist is already ordered
;;; not needed anymore
;;; (defun update-nodelist (nodelist)
;;;   (sort-nodelist nodelist))


;;; setup for the n-best algorithm
(defun root-node (pointerlist)
  (vector 0 NIL 1.0 (sort-pointer-list pointerlist)
;;;                              ,   (make-array (+ length 1)
;;;                                            :initial-element 1)
;;;                                 ))
;;;  (list 0 NIL 1.0 (sort-pointer-list pointerlist)))

(defun special-root-nd (pointerlist storage)
  (let ()
    (setf (nd-node-id storage) 0)
    (setf (nd-mutations storage ) NIL)
    (setf (nd-prob storage) 1.0)
    (setf (nd-ptrs storage) (sort-pointer-list pointerlist))
    storage))

;;(defun special-root-node (pointerlist length storage)
;;  (if (= (length storage) 4)
;;      (progn (setf (svref storage 0) 0)
;;          (setf (svref storage 1) NIL)
;;          (setf (svref storage 2) 1.0)
;;          (setf (svref storage 3) (sort-pointer-list pointerlist))
;;          storage)
;;    (error "special-root-node: storage must be an array of length 4")))


;;; Find the n most likely phone sequences that could have been the
;;; antecedents from which the recognizer produced its candidate
;;; phonestring

(defun find-sublists (phonestring)
  (let ((result NIL)
       (count 0))
    (dolist (phone phonestring)
      (setf result (cons (create-sublist phone count)
                         result))
      (setf count (+ count 1)))
    result))


(defun find-dellists (phonestring
                    &key ((:level printlevel) 0)
                         ((:indep indep) NIL))
  (let ((result NIL)
       (prev NIL)
       (count -1))
    (dolist (phone phonestring)
      (setf result (cons (create-dellist prev phone count :level printlevel
                                      :indep indep)
```

```
                        result))
      (setf prev phone)
      (setf count (+ count 1)))
    (setf result (cons (create-dellist prev NIL count :level printlevel)
                      result))
    result))


;;; collext and return all the substitutions in a mutation list
(defun collect-subs (mutationlist)
  (let ((result NIL))
    (dolist (action mutationlist)
      (if (eq (action-type action) 'SUB)
          (setf result (cons action result))))
    (sort result #'earlier-action)))

;;; (defun collect-subs (mutationlist)
;;;   (let ((result NIL))
;;;     (dolist (action mutationlist)
;;;        (if (eq (action-type action) 'SUB)
;;;       (insert-thing action result #'compare-actions)))
;;;    result))

;;; assume that action1 is never nil
(defun compare-actions (action1 action2)
  (if action2
      (earlier-action action1 action2)
    T))

;;; (defun insert-thing (thing alist comparer)
;;;   (if (funcall comparer thing (Car alist))
;;;      (cons pointer pointerlist)
;;;     (progn (insert-thing-aux thing alist comparer 0)
;;;       alist)))

;;; (defun insert-thing-aux (thing alist comparer count)
;;;   (if (funcall comparer thing (cadr alist))
;;;       (rplacd alist (cons thing (cdr alist)))
;;;    (insert-thing-aux thing (cdr alist) comparer (- count 1))))




;;; Collect and return all the deletions in a mutation list
;;; This procedure returns a list of possible deletion sequences.
;;; Deletion sequences differ from each other in the order in which
;;; deletions between the same pair of phones are repaired.
(defun collect-dels (mutationlist)
  (let ((result NIL))
    (dolist (action mutationlist)
      (if (eq (action-type action) 'DEL)
          (setf result (cons action result))))

    (setf result (sort result #'earlier-action))
    (let ((ok (check result)))
      (if ok
          (generator ok)
        (list result)))))


;;; If there are no places with multiple deletions return NIL
;;; Else return a structure that is sent to the procedure
;;; generate below to generator the required dellist

(defun check (sortactionlist)
;;;  (princ "sortactionlist: ") (princ sortactionlist) (fresh-line)
  (let ((prev -1)
       (prevlist NIL)
       (indexcount 0)
       (result NIL)
       (flag T))
    (dolist (action sortactionlist)
      (let ((current (action-index action)))
       (if (= current prev)
           (progn (setf indexcount (+ indexcount 1))
                  (setf prevlist (cons action prevlist)))
         (progn (if (> indexcount 1)
                    (progn (setf flag NIL)
;;;                        (princ "prevlist: ") (princ prevlist) (fresh-line)
```

```
                           (setf result (cons (permutaticns prevlist)
                                             result)))
                  (if (> indexcount 0)
                      (setf result (cons (list prevlist) result))))
                (setf prev current)
                (setf prevlist (list action))
                (setf indexcount 1)))))
    (if (> indexcount 1)
        (progn (setf flag NIL)
;;;            (princ "prevlist: ") (princ prevlist) (fresh-line)
               (setf result (cons (permutations prevlist)
                                  result)))
      (setf result (cons (list prevlist) result)))
    (if flag
        NIL
      (reverse result))))


;;; Generate the list of deletions from the output of check
(defun generator (checkoutput)
  (if (null checkoutput)
      (list NIL)
    (let ((result NIL)
          (first (car checkoutput)))
      (dolist (el first)
        (let ((result1 (generator (cdr checkoutput))))
          (dolist (ell result1)
            (setf result (cons (append el ell) result)))))
      result)))


;;; takes a list and returns a list containing all distinct permutations
;;; of the elements of the list
(defun permutations (alist)
  (if (null alist)
      NIL
    (let ((result NIL)
          (begin NIL)
          (end alist))
      (do ((next end))
          ((null next))
        (let ((curr (car next))
              (rest (permutations (append begin (cdr next)))))
          (if (null rest)
              (setf result (list (list curr)))
            (dolist (res rest)
              (setf result (cons (cons curr res) result))))
          (setf begin (cons curr begin))
          (setf next (cdr next))))

      (let ((result1 NIL))
        (do ((rest result))
            ((null rest))
          (setf result1 (cons (car rest) result1))
          (setf rest (remove-if (pred (car rest))
                                (cdr rest))))
       result1))))


(defun pred (x)
  #'(lambda (y)
      (equal x y)))



;;; - Build the new phonestring by mutating the original phonestring
;;; - Also map SIL2 to NIL everywhere
;;; - Also if there are several corrected-deletions between a particular
;;;   pair of original phones, generate words resulting from all
;;;   permutations of these corrected-deletions
(defun build-string (mutationlist string &key ((:level printlevel) 0))
  (let ((subs (collect-subs mutationlist))
        (dels (collect-dels mutationlist))
        (result NIL)
        (result3 NIL)
        (done NIL)
        (count 0))
    (if (> printlevel 1)
        (progn (princ "Sub: ") (princ subs) (fresh-line)
               (princ "Del: ") (princ dels) (fresh-line)))
    (dolist (phone string)
      (if (eq (action-index (car subs)) count)
          (progn
            ;;; if action-param is NIL then we are correcting an insertion
            (let ((act (action-param (car subs))))
              (if (and act
                       (not (eq act 'SIL2)))
                  (setf result (cons act result))
                (setf result (cons '? result)))
              (setf subs (cdr subs)))))
        (if (not (eq phone 'SIL2))
            (setf result (cons phone result))))
      (setf count (+ count 1)))
    ;;; start count off at (- (length string) 1)
    ;;; (setf count (- count 1))


    (dolist (delseq dels)
      (let ((result2 NIL)
            ;;; start count at the end of the string since result is
            ;;; in reverse order
            (count (- (length string) 1)))
        ;;; since result is now in reverse order
        (setf delseq (reverse delseq))
        (dolist (phone result)
          (setf done NIL)
          (do ((thedel (car delseq) (car delseq)))
              (done)
            (if (eq (action-index thedel) count)
                (progn
                  (if (not (eq (action-param thedel) 'SIL2))
                      (setf result2 (cons (action-param thedel) result2)))
                  (setf delseq (cdr delseq)))
              (setf done T)))

          (setf result2 (cons phone result2))
          (setf count (- count 1)))

        (do ((thedel (car delseq) (car delseq)))
            ((null delseq))

          (if (not (eq (action-param thedel) 'SIL2))
              (setf result2 (cons (action-param thedel) result2)))
          (setf delseq (cdr delseq)))

        (setf result3 (cons (remove '? result2) result3))))

    (if (> printlevel 3)
        (progn (princ "build-string output: ")
               (princ result3)
               (fresh-line)))


    ;;; n-best expects back a list of phonestrings produced by
    ;;; mutating the original phonestring according to the
    ;;; mutations stored in the node.
    result3 ))



;;; new and improved build-string that does not use any sorts
(defun clean (array)
  (dotimes (i (length array))
    (setf (svref array i) NIL)))


(defun bs (mutationlist string subarray delarray
           bs-result &key ((:level printlevel) 0))
  (let* ((length (length string))
         (lengthsub 0)
         (lengthadd 0)
         (wordcount 1)
         (factor 1)
         (indexcount 0))
  (clean subarray)
  (clean delarray)

  ;;; collect the substitutions and deletions
  (dolist (action mutationlist)
    (if (> (action-index action)
           (- length 1))
        (error "The phone string isn't that long"))

  ;;; collect the substitutions
    (if (eq (action-type action) 'SUB)
        (setf (svref subarray (action-index action))
          (let ((act (action-param action)))
             (if (< (action-index action) 0)
               (error "There are no phones at that index..."))
            (if (and act
                     (not (eq act 'SIL2)))
               act
             (progn (setf lengthsub (+ lengthsub 1))
                    '?))))))
    ;;; collect the deletions
    (if (eq (action-type action) 'DEL)
        (setf (svref delarray (+ (action-index action) 1))
          (let ((act (action-param action)))
            (if (< (action-index action) -1)
               (error "You can't insert a phone there..."))
            (if (not (eq act 'SIL2))
               (progn
                 (setf lengthadd (+ lengthadd 1))
                 (cons act (svref delarray (+ (action-index action) 1))))))))))

  ;;; process the deletions to get the permutations
  (dotimes (count (+ length 1))
    (let* ((dels (svref delarray count))
           (dellength (length dels)))
     (cond ((= dellength 1)
             (setf (svref delarray count) (list dels)))
            ((> dellength 1)
             (let ((perms (permutations dels)))
              (setf factor (* factor (length perms)))
              (setf (svref delarray count) perms))))))
  (setf wordcount (* wordcount factor))


  (if (> printlevel 1)
     (progn (princ "Sub: ") (princ subarray) (fresh-line)
            (princ "Del: ") (princ delarray) (fresh-line)))



  ;;; build the string

  (if (> wordcount (length bs-result))
     (error "Too many strings produced by procedure bs."))
  (let* (
;;; (result-array (make-array wordcount))
        (poscount 0)
        (beforestring (svref delarray 0))
        (bstringlength (length beforestring))
        (beforelength (length (car beforestring))))

    ;;; intialize the result array
    (let ((thewordlength (- (+ length lengthadd) lengthsub)))
     (if (> thewordlength (length (svref bs-result 0)))
         (error "Words are too long in procedure bs.")))

     ;;; (dotimes (count wordcount)
     ;;; (setf (svref result-array count)
     ;;; (make-array thewordlength)))

    ;;; start by doing the deletions, if any, that are to be repaired
    ;;; before the first phone in string
    (if (> bstringlength 0)
        (progn
          (dotimes (count1 (/ wordcount bstringlength))
            (dotimes (count2 bstringlength)
             (let* ((wordindex (+ (* count1 bstringlength)
                                     count2))
                    (bs-word-array (svref bs-result wordindex))
;;;                    (word-array (svref result-array wordindex))
                       (phoneindex indexcount)
                       (current (nth count2 beforestring)))
                  (dolist (phone current)
                    ;;; (setf (svref word-array phoneindex) phone)
                    (setf (svref bs-word-array phoneindex) phone)
                    (setf phoneindex (+ phoneindex 1))))))
            (setf indexcount (+ indexcount beforelength))))
      ;;; now walk down the list of phones doing subtitutions/deletions/
      ;;; insertions  as necessary
      (dolist (phone string)
          ;;; do the substitutions
        (let* ((the-sub (svref subarray poscount))
               (the-phone (if the-sub
                             (if (eq the-sub '?)
                                NIL
                               the-sub)
                           phone)))
          (dotimes (count1 wordcount)
            (let (
                 ;;; (word-array (svref result-array count1))
                   (bs-word-array (svref bs-result count1)))
              (if the-phone
                  (progn
                    ;;; (setf (svref word-array indexcount) the-phone)
                        (setf (svref bs-word-array indexcount) the-phone)))
              ))

            ;;; if there is a phone at indexcount increment indexcount
            ;;; we don't overwrite
          (if the-phone
              (setf indexcount (+ indexcount 1))))
          ;;; increment position counter for 2 reasons. First of all,
          ;;; the deletion array needs to be indexed one ahead.   Secondly
          ;;; the next time around the loop we want poscount to be
          ;;; incremented by 1
        (setf poscount (+ poscount 1))

          ;;; do the deletions
        (let ((the-dels (svref delarray poscount)))
          (if the-dels      ;;; if there are any deletions
              (let ((perms (length the-dels))
                    (dellength (length (car the-dels))))
                (dotimes (count1 (/ wordcount perms))
                  (dotimes (count2 perms)
                    (let* ((wordindex (+ (* count1 perms)  count2))
                          ;;; (word-array (svref result-array wordindex))
                          (bs-word-array (svref bs-result wordindex))
                          (phoneindex indexcount)
                          (current (nth count2 the-dels)))
                      (dolist (phone current)
                       ;;; (setf (svref word-array phoneindex) phone)
                        (setf (svref bs-word-array phoneindex) phone)
                        (setf phoneindex (+ phoneindex 1))))))
                (setf indexcount (+ indexcount dellength))))))


;;;      (if (> printlevel 3)
;;;     (progn (princ "build-string output: ")
;;;            (princ result-array)
;;;            (fresh-line)))

      (if (> printlevel 3)
          (progn (princ "bs-array: ")
                 (princ bs-result)
                 (fresh-line)))

      (cons wordcount (- (+ length lengthadd) lengthsub))
;;;      result-array

      )))
```

```
;;; Find the n-best words phone string could have come from
;;; while mapping SIL2 to NIL
;;; This may sometimes return more than n strings if the number
;;; of words generated by the last call to build-string is
;;; greater than 1.  That could happen when more than one permutation
;;; of the corrected deletions at some point generates valid words


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; global storage for nodes
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; We can visit at most 20000 nodes
(defparameter *max-node* 20000)
;;; there can be no more than 30 phones in a word
(defparameter *max-phones* 20)

;;; storage for the node structures
(setf *storage-array*   (make-array *max-node* :initial-element NIL))
(dotimes (i (length *storage-array*))
  (setf (svref *storage-array* i)
    (make-nd)))

;;; storage for the  pointers stored in node
(setf *pointer-storage* (make-array *max-node* :initial-element NIL))
(dotimes (i (length *pointer-storage*))
  (setf (svref *pointer-storage* i)
     (let ((result NIL))
       (dotimes (j (+ (* 2 *max-phones*) 1))
         (setf result (cons NIL result)))
       result)))


;;;code to compare arrays
(defun equal-array (array1 array2 n)
  (if (> n (min (length array1) (length array2)))
       (error "equal-array: n too large")
     (let ((result T))
       (dotimes (i n)
         (if (not (equal (svref array1 i)
                         (svref array2 i)))
             (setf result NIL)))
       result)))



;;; pair1 turns out to actually be a triple for us.
;;; pair1 = (array array-length probability)
(defun compare-array-length-pairs (pair1 pair2)
  (if (not (= (cadr pair1) (cdr pair2)))
      NIL
    (equal-array (car pair1) (car pair2) (cadr pair1))))

(defvar *nb-debug* NIL)


;;; search n more than 15000 nodes
(defparameter *searchbound* 15000)

(defun new-n-best (phonestring n &key ((:level printlevel) 0)
                                      ((:searchbound searchbound) *searchbound*)
                                      ((:test test) NIL)
                                      ((:indep indep) NIL))
  (let*  ((sublists (find-sublists phonestring))
          (dellists (find-dellists phonestring :level printlevel :indep indep))
          (stringlength (length phonestring))
          (pointerlist (copy-pointers (svref *pointer-storage* 0)
                                      (concatenate 'list sublists dellists)))
          (specialroot (special-root-nd pointerlist
                                        (svref *storage-array* 0)))
          (nodelist (pq:make-priority-queue #'more-likely-node :size 1000))
          (stripstring (let ((temp (remove 'SIL2 phonestring)))
                         (make-array (length temp) :initial-contents temp)))
          (result NIL)
          (searchcount 0)
          (hitcounter 0)
          (testresult NIL)
          (subarray (make-array stringlength :initial-element NIL))
          (delarray (make-array (+ stringlength 1) :initial-element NIL))
          (newstring NIL)
          (current-node NIL)
          (allstrings NIL)
          (the-words NIL)
          (bs-result (make-array 40 :initial-element NIL))
          (new-node NIL))

    (if (> printlevel 0)
        (progn (princ "N-BEST RESULTS FOR: ") (princ phonestring)
               (fresh-line)
               (princ "--------------------") (fresh-line)))

    ;;; set up the result of build string
    (dotimes (counter (length bs-result))
      (setf (svref bs-result counter)
        (make-array 40 :initial-element NIL)))

;;;; $$$ changed to use specialroot instead of root
    (pq:pq-insert specialroot nodelist)
    (if (is-a-word stripstring (length stripstring) )
        (progn (setf n (- n 1))
               (if test
                   (progn
                     (setf the-words (wds stripstring (length stripstring)))
                     (if (if (listp the-words)
                              (member test the-words :test #'equal)
                            (equal test the-words))
                          (setf testresult (cons T 0)))))


               (setf result (cons (list stripstring (length stripstring) 1.0) result))

               (if *nb-debug*
                   (progn (princ "[0] ")
                          (princ "Probability: ")
                          (princ (nodeprob specialroot))
                          (princ "  WORD: ")
                          (princ (wds stripstring (length stripstring)))
                          (fresh-line)))

               (if (> printlevel 0)
                   (format t "~A ~10,1T " (wds stripstring (length stripstring))))


               ))

    (do ((counter 0))
        ((or (>= counter n)
             (>= searchcount searchbound)) (setf hitcounter counter))
      (setf searchcount (+ searchcount 1))
      (setq current-node (pq:pq-pop nodelist))
      (if (> printlevel 1)
          (progn (princ "**** Counter = ") (princ counter) (princ " ****")
                 (fresh-line)
                 (princ "Current Node [") (princ (nodeid current-node))
                 (princ "]") (princ " Probability = ")
                 (princ (nodeprob current-node)) (fresh-line)))

      ;;; setup the new node
      (setf new-node  (svref *storage-array* searchcount))
      ;;; note that the next line destructively modifies current-node
      ;;; and new-node
      (mutate current-node (svref *storage-array* searchcount)
                           (svref *pointer-storage* searchcount))

      (pq::pq-insert current-node nodelist)
      (if (> printlevel 1)
          (progn (princ "New Node [") (princ (nodeid new-node))
                 (princ "]") (princ " Probability = ")
                 (princ (nodeprob new-node)) (fresh-line)))

      (pq:pq-insert new-node nodelist)

      (setf allstrings (bs (nodemutations new-node)
                           phonestring subarray delarray bs-result
                           :level printlevel))
      (dotimes (stringcount (car allstrings))
        (setf newstring (svref bs-result stringcount))
        (if (> printlevel 3)
            (progn
              (princ "length: ") (princ (cdr allstrings))
              (princ "  newstring: ") (princ newstring) (fresh-line)
              (fresh-line)))

        (if (> (length newstring) 0)
            (if (and (is-a-word newstring (cdr allstrings) )
                    (not (member-if
                          #'(lambda (x) (compare-array-length-pairs
                                        x (cons newstring (cdr allstrings))))
                          result)))

               (progn
                 (setf counter (+ counter 1))
                      (if test
                          (progn
                            (setf the-words (wds newstring (cdr allstrings)))
                            (if (if (listp the-words)
                                    (member test the-words :test #'equal)
                                 (equal test the-words))
                               (setf testresult (cons T counter)))))

                      (if *nb-debug*
                          (progn (princ "[") (princ counter)
                                 (princ "] ")
                                 (princ "Probability: ")
                                 (princ (nodeprob new-node))
                                 (princ "  WORD: ")
                                 (princ (wds newstring (cdr allstrings)))
                                 (fresh-line)))

                      (if (> printlevel 0)
                          (format t "~A ~10,1T " (wds newstring (cdr allstrings))))
                      (if (= 0 (mod counter 3))
                          (fresh-line))

                      (setf result (cons (list (copy-seq newstring)
                                              (cdr allstrings)
                                              (nodeprob new-node)) result)))))
       ;;; ends the (dotimes (stringcount))
       )

      ;;; ends (do ((counter 0)))
            )

    (if (= searchcount searchbound)
        (progn (princ "Searched ") (princ searchcount) (princ " nodes.")
               (fresh-line)))


    (if (and test
             (not (car testresult)))
        (setf testresult (cons NIL hitcounter)))


    (if test
       testresult
    ;;; return the words in decreasing order of probability
      (convert-to-words result))

         ))

;;; - multiple paths return the same string because of the deletions
;;; - beacuse of the way I'm removing SIL2 from the phone set, multiple
;;;  paths exist that differ only by an extra deletion of SIL2... could
;;;  take these out explicitly?
;;; - Fix things to take the phonemap argument properly...


;;; convert a list of phonestrings into a list of the corresponding words.
;;; convert-to-words reverses the order of phonestrings
(defun convert-to-words (phonestrings)
  (let ((result NIL))
    (dolist (phonestringtriple phonestrings)
      (let ((the-words (wds (car phonestringtriple) (cadr phonestringtriple)))
            (prob (caddr phonestringtriple)))
        (if (listp the-words)
            (dolist (word the-words)
              (setf result (cons (cons word prob) result)))
          (setf result (cons (cons the-words prob) result)))))
    result))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;    CONTINUOUS RECOGNITION LOOP
;;; I'm reading from a named pipe *localpipefile*
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;; continuous ambiguation using method 1.  i.e., for every
;;; phone that is read in, ambiguate it and take appropriate
;;; action to produce a regexp for the TDB cod(grol '((SIL2-B IH2 R)))
;;; I can't use the get-phone call beca
(defun ambiguatel (phonestream &rest rest &key
                      ((:deletions deletions) *deletions*)
                      ((:sub sub) subthresh)
                      ((:delt delt) -5)
                      ((:bound bound) (floor *insmax*))
                      ((:show show) NIL)
                      ((:limit limit) 20)
                      ((:search search) T)
                      ((:prox prox) 10)
                      ((:show-wds show-wds) NIL)
                      ((:forcebound forcebound) NIL)
                      ((:printconf printconf) NIL)
                  &allow-other-keys)

  (declare (special *deletions*)
           (special subthresh)
           (special *insmax*)
           (special *delprob*))

  (do ((char (read-char-no-hang *standard-input* NIL NIL)
             (read-char-no-hang *standard-input* NIL NIL)))
      ((eq char #\q) (with-open-file (out "LispToCl.txt" :direction :output
                                          :if-exists :supersede)
                      (princ 'DONE out)
                      'DONE))

    (prin1 "*************") (fresh-line)
    (let ((utterance (read phonestream)))
      (if utterance
          (progn
            (let ((result NIL)
                  (conf NIL))
              (dolist (phonestring utterance result)
                (if forcebound
                    (setf bound forcebound)
                  (setf bound (floor (* *delprob* (length phonestring)))))
                (prin1 bound) (fresh-line)
                (prin1 phonestring) (fresh-line)
                (setf rest (cons :string (cons phonestring rest)))
                (setf rest (cons :bound (cons bound rest)))
                (setf conf (apply 'bconfuse rest))
                (setf result (append result (list conf)))
                (if printconf
                    (prin1 conf))
                (fresh-line)
;;;           (if *groliers-loaded*
;;;               (groll (list conf) :show show
;;;                      :show-wds show-wds
;;;                      :limit limit
;;;                      :search search
;;;                      :prox prox))
               )
;;;         (princ "$$$$$$$$$$$$") (princ result) (fresh-line)
              (groll result :show show :show-wds show-wds
                     :limit limit :search search :prox prox)))))))


(defun ambiguate (&rest rest &key ((:phonepipefile phonepipefile) *localpipefile*)
                 &allow-other-keys)
  (with-open-file (ostream phonepipefile :direction :input)
    (apply 'ambiguatel (cons ostream rest))))


;;;; continuous recognition loop using the n-best algorithm
(defun question (&rest rest &key
         ((:phonepipefile phonepipefile) *remotepipefile*)
                &allow-other-keys)
  (if (or (equal *user* 'don)
         (equal *user* 'vijay))
      (progn (princ "********* WARNING ********: Confusion data for ")
             (princ *user*)
             (princ " is out of date.")
             (fresh-line)))
  (with-open-file (ostream phonepipefile :direction :input)
    (apply 'query1 (cons ostream rest))))


(defun query1 (phonestream  &key
                           ((:doquery doquery) T)
                           ((:number number) 30)
                           ((:level printlevel) 1)
                           ((:searchbound searchbound) *searchbound*)
                            ;;; stuff to do with the groll query
                           ((:show show) NIL)
                           ((:search search) T)
                           ((:prox prox) 10)
                           ((:indep indep) NIL)
                           ((:show-wds show-wds) NIL)
              &allow-other-keys)


  (if (> printlevel 2)
      (progn (princ "In query1") (fresh-line)))



  (do ((char (read-char-no-hang *standard-input* NIL NIL)
             (read-char-no-hang *standard-input* NIL NIL)))
      ((eq char #\q) (with-open-file (out "LispToCl.txt" :direction :output
                                     :if-exists :supersede)
                      (princ 'DONE out)
                      'DONE))

    (prin1 "*************") (fresh-line)
    (let ((utterance (read phonestream)))
      (if utterance
          (progn
            (let ((result NIL)
                  (conf NIL))
              (if (> printlevel 0)
                  (progn (princ "Utterance: ") (princ (car utterance))
                         (dolist (aword (cdr utterance))
                           (fresh-line)
                           (princ "           ") (princ aword))
                         (fresh-line)))
              (dolist (phonestring utterance result)
                (if (> printlevel 1)
                    (progn (princ "Doing new-n-best for ")
                           (princ (prepare phonestring))
                           (fresh-line)))

                (setf conf (new-n-best (prepare phonestring) number
                                     :level printlevel
                                     :indep indep
                                     :searchbound searchbound))

                (if (> printlevel 1)
                    (progn (princ "n-best results: ")
                           (princ conf)
                           (fresh-line)))


                (setf result (cons conf result)))
              (if (> printlevel 1)
                  (progn (princ "Query String: ")
                         (print result)
                         (fresh-line)))
              (if doquery
                  (groll result :show show :show-wds show-wds
                      :search search :prox prox)))))
      )))



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; convenient methods of accessing grolier's using Julian's code
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;




;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;   Code to initialize the world by reading in new phonemaps,
;;;   new confusion matrices and the like
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defun initworld (&key ((:phonemapfile  phonemapfile) *phonemapfile*)
                      ((:probfile probfile) *probfile*)
                      ((:confusions confusion) NIL)
                      ((:groliers groliers) T)
                      ((:phonefile phonefile) *phonefile*)
                      ((:phonepipefile phonepipefile) *localpipefile*)
                      ((:subthresh subthresh) 0.05)
                      ((:delthresh delthresh) 0.03))

  (declare (special *phonemapfile*)
           (special *probfile*)
           (special *phonefile*)
           (special *localpipefile*)
           (special *phonestring*)
           (special *phonemap*)
           (special *deletions*)
           (special *fileout*)
           (special *insmax*)
           (special *delprob*))

  (setf *use-confusion-classes* confusion)
  (setf *groliers-loaded* groliers)
  (setf *phonestring* '())
  (setf *phonefile* phonefile)
  (setf *localpipefile* phonepipefile)
  (setf *phonemapfile* phonemapfile)
  (setf *probfile* probfile)
  (setf *phonemap* '())

  ;;; load in the confusion data
   (load-probfile probfile)

  ;;; set the current incoming phone string
  ;;; I think I just won't read in the new phonestring from *phonefile*
  ;;; when initializing because I'm not going to be doing single phonestrings
  ;;; any more
  ;;;  (setup-string)
  ;;; set the current phonemap
  ;;;  (setup-phonemap)
  ;;; no phonemap anymore
   (setf *phonemap* NIL)

  ;;; file to write result to

  (setf *fileout* "phoneout")

  (setq subthresh subthresh)
  (setq delthresh delthresh)

  (setf *deletions*  (dels delthresh))

  (setf *insmax* (* (length *phonestring*) *delprob*))

  (prinl "Initializations Completed.")
  T)

(defun set-user (name)
  (declare (special user-list)
          (special *user*))
  (setf *user* name)
  (let ((probfile (cdr (assoc name user-list))))
    (if (null probfile)
       (princ "No such user.")
      (initworld :probfile probfile)))
  'DONE)



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;  code to convert the TIMIT lexicon in timitdic.txt to BUSH
;;;  phones
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defun timit-convert (&key ((:outfile outfile)
                         "/tilde/vijayb/Speech/aristotle/dictionaries/timitmoby2.txt")
                        ((:infile infile)
                         "/tilde/vijayb/Speech/aristotle/dictionaries/timitdic2.txt"))
  (with-open-file (out outfile :direction :output :if-exists :supersede)
    (with-open-file (in infile :direction :input)
      (timit-convert-aux in out))))

(defun timit-convert-auxl (in out)
  (let ((line (read-line in)))
    (cond ((eq line NIL) 'DONE)
         ((eq (char line 0) #\;)
          (write-line line out)
        (timit-convert-auxl in out))
         (T (let* ((pos (position #\space line))
                  (word (subseq line 0 pos))
                  (phones (phn word)))
              (if (not (eq phones NIL))
                 (progn (princ word  out)
                        (princ "  /"  out)
                        (dotimes (i (length phones))
                          (princ (aref phones i)  out)
                          (if (< i (- (length phones) 1))
                             (princ " "  out)))
                        (princ "/"  out)
                        (fresh-line out)))
              (timit-convert-auxl in out)))))))

(defun timit-convert-aux (in out)
  (do ((line (read-line in NIL NIL) (read-line in NIL NIL)))
      ((null line))
    (cond ((eq line NIL) 'DONE)
         ((eq (char line 0) #\;)
          (write-line line out))
         (T (let* ((pos (position #\space line))
                  (word (subseq line 0 pos))
                  (phones (phn word)))
              (if (not (eq phones NIL))
                 (progn (princ word  out)
                        (princ "  /"  out)
                        (dotimes (i (length phones))
                          (princ (aref phones i)  out)
                          (if (< i (- (length phones) 1))
                             (princ " "  out)))
                        (princ "/"  out)
                        (fresh-line out))
               (progn (princ "Couldn't find phones for ")
                     (princ word) (fresh-line)
                     (princ word out)
                     (princ "  /" out)
                     (let ((rest (subseq line (+ pos 3))))
                       (print-phones rest out))
                     (princ "/" out)
                     (fresh-line out)))))))
  'DONE)


(defun print-phones (rest out)
  (let ((pos (position #\space rest)))
    (cond ((equal rest "/") 'DONE)
         (pos
          (let ((newphone (cadr (assoc (intern (get-the-phone rest pos))
                                   *modmap*))))
            (if (null newphone)
                (progn (princ "newphone: ") (princ newphone) (fresh-line)
                       (error "Phone not found.")))
             (princ newphone out)
             (princ " " out)
             (print-phones (subseq rest (+ pos 1)) out)))
          (T
           (let ((newphone (cadr (assoc (intern (get-the-phone rest (- (length rest) 1)))
                                    *modmap*))))
             (if (null newphone)
                (progn (princ "newphone: ") (princ newphone) (fresh-line)
                       (princ (get-the-phone rest (- (length rest) 1)))
                       (fresh-line)
                       (princ rest) (fresh-line)
                       (error "Phone not found.")))
             (princ newphone  out))
          'DONE))))


(defun get-the-phone (string spcpos)
  (let* ((phone (subseq string 0 spcpos))
         (length (length phone))
         (lastchar (char phone (- length 1))))
    (if (or (equal lastchar #\1)
            (equal lastchar #\2))
       (setf phone (subseq phone 0 (- length 1))))
    (string-upcase phone)))


(defun add-dir (dir filename)
  (concatenate 'string dir filename))

(defun add-dictdir (filename)
  (concatenate 'string "/project/ssp/vijay/aristotle/dictionaries/" filename))

;;; code to convert the entire dictionary into the TIMIT format
;;; for the C nbest to read

(defun convert-dictionary (&key ((:infile infile)
                               (add-dictdir "moby-grolier-overlap.txt"))
                              ((:outfile outfile)
                               (add-dictdir "moby-timit.txt")))
  (with-open-file (in infile :direction :input)
    (with-open-file (out outfile :direction :output
                    :if-does-not-exist :create
                    :if-exists :supersede)
      (let ((counter 0)
            (current-counter 0)
            (previous-phone-list NIL)
            (previous-word NIL))

        (princ "Reading dictionary from ") (princ infile) (fresh-line)
        (princ "Writing converted dictionary to ") (princ outfile) (fresh-line)

      (do ((line (read in NIL NIL) (read in NIL NIL)))
          ((null line))
        (cond ((null line) 'DONE)
              (T (let* ((word (car line))
                        (phone-array (cadr line))
                        (length (length phone-array)))
                  (if (not (equal word previous-word))
                      (progn (setf previous-word word)
                             (setf current-counter 1)
                             (setf counter (+ counter 1))
                             (setf previous-phone-list
                              (list phone-array))
                             (phone-print word phone-array length out))
                    (if (not (position phone-array previous-phone-list
                                     :test #'eq-array))
                        (progn (setf previous-phone-list
                                (cons phone-array previous-phone-list))
                              (setf current-counter (- current-counter 1))
                              (setf counter (+ counter 1))
                              (phone-print word phone-array length out))
                      ))))
             ))
      'DONE))))


(defun eq-array (array1 array2)
  (if (not (= (length array1) (length array2)))
      NIL
    (let ((result T))
      (dotimes (i (length array1))
        (if (not (eq (svref array1 i)
                     (svref array2 i)))
            (setf result NIL)))
      result)))



(defun phone-print (word phone-array length out)
  (princ word out) (princ "  /" out)
  (dotimes (i length)
    (princ (aref phone-array i) out)
    (if (< i (- length 1))
        (princ " " out)))
  (princ "/" out)
  (fresh-line out))



;;; write out the confusion data to a series of files for the HMM based
;;; n-best to read
(defun write-confusions (&key ((:user user) *user*))
  (let ((old-user *user*)
        (outdir (concatenate 'string
                  "/tilde/vijayb/Speech/aristotle/nbest-args/" user "/"))
        (length (length *phonelist*)))


    ;;; set the user to be the one whose data we want to output
    (set-user (intern user))

    ;;; Write out the  following variables
    ;;; *total*
    ;;; *totldels*
    ;;; *delprob*
    ;;; *PhoneCountList*
    ;;; *DelList*
    ;;; *DelCountList*
    ;;; *SubCountList*
    ;;; *LContDel*
    ;;; *RContDel*

    ;;; write out *total*
    (with-open-file (outfile (add-dir outdir "total") :direction :output
                    :if-exists :supersede :if-does-not-exist :create)
      (princ *total* outfile))

    ;;; write out *totdels*
    (with-open-file (outfile (add-dir outdir "totdels") :direction :output
                    :if-exists :supersede :if-does-not-exist :create)
      (princ *totdels* outfile))

    ;;; write out delprob
    (with-open-file (outfile (add-dir outdir "delprob") :direction :output
                    :if-exists :supersede :if-does-not-exist :create)
      (princ *delprob* outfile))


    ;;; write out *PhoneCountList*
    (with-open-file (outfile (add-dir outdir "PhoneCountList") :direction :output
                    :if-exists :supersede :if-does-not-exist :create)
      (dotimes (count length)
        (princ (aref *PhoneCountList* count) outfile)
        (princ " " outfile) ))


    ;;; write out *DelList*
    (with-open-file (outfile (add-dir outdir "DelList") :direction :output
                    :if-exists :supersede :if-does-not-exist :create)
      (dotimes (count length)
        (princ (aref *DelList* count) outfile)
        (princ " " outfile)))

    ;;; write out *DelCountList*
    (with-open-file (outfile (add-dir outdir "DelCountList") :direction :output
                  :if-exists :supersede :if-does-not-exist :create)
      (dotimes (count length)
        (princ (aref *DelCountList* count) outfile)
        (princ " " outfile)))


    ;;; write out *SubList*
    (with-open-file (outfile (add-dir outdir "SubList") :direction :output
                  :if-exists :supersede :if-does-not-exist :create)
      (dotimes (count length)
        (dotimes (count1 length)
          (princ (aref *SubList* count count1) outfile)
          (princ " " outfile))
        (fresh-line outfile)))

    ;;; write out *LContDel*
    (with-open-file (outfile (add-dir outdir "LContDel") :direction :output
                  :if-exists :supersede :if-does-not-exist :create)
      (dotimes (count length)
        (dotimes (count1 length)
          (princ (aref *LContDel* count count1) outfile)
          (princ " " outfile))
        (fresh-line outfile)))


    ;;; write out *RContDel*
    (with-open-file (outfile (add-dir outdir "RContDel") :direction :output
                  :if-exists :supersede :if-does-not-exist :create)
      (dotimes (count length)
        (dotimes (count1 length)
          (princ (aref *RContDel* count count1) outfile)
          (princ " " outfile))
        (fresh-line outfile)))


    ;;; return to the original user
    (set-user  old-user)

    ))



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; tuning of the substitution/insertion/deletion probabilities
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defun timit-tuneup ()
  (declare (special subthresh))
  (setf subthresh 0.05)
  (set-sub 'M '? 0.051)
  (set-sub 'N '? 0.051)
  (set-sub 'P '? 0.051)
  (set-sub 'K '? 0.051)
  (set-sub 'T '? 0.051)
  (set-sub 'OW 'UH 0.051)
  (set-sub 'Y 'IH 0.051))


(defun timit-tuneup-vijay ()
  (declare (special subthresh))
  (setf subthresh 0.05)
  (set-sub 'M '? 0.051)
  (set-sub 'N '? 0.051)
  (set-sub 'P '? 0.051)
  (set-sub 'K '? 0.051)
  (set-sub 'T '? 0.051)
  (set-sub 'T 'K 0.051)
  (set-sub 'OW 'UH 0.051)
  (set-sub 'Y 'IH 0.051))



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;;;  test suites
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defvar *test1* (add-base "tests/trials.txt"))


(defun convert (phonestring)
  (declare (special *modmap*))
  (let ((result nil))
    (dolist (phone phonestring)
      (setf result (cons (cadr (assoc phone *modmap* ))
                         result)))
    (reverse result)))


(defun index (name)
  (declare (special *phonelist*))
  (position name *phonelist*))


(defun prepare (phonestring)
  (remove 'SIL2 (convert phonestring)))


(defun read-tests (filename )
  (let ((result NIL))
    (with-open-file (infile filename :direction :input)
     (setf result (cons (read infile) result)))
    result))




(defun testtrials (&key ((:number number) 30)
                       ((:level level) 1)
                       ((:searchbound searchbound) 15000)
                       ((:infile infile)
                        "/tilde/vijayb/Speech/aristotle/tests/trials.txt"))
                       ;;; ((:outfile outfile)
                       ;;; "/tilde/vijayb/Speech/aristotle/tests/trial2.txt")

  (if (equal infile "/tilde/vijayb/Speech/aristotle/tests/trials.txt")
      (set-user 'don))
  (let ((summary (list NIL NIL)))
    (with-open-file (in infile :direction :input)
;;;     (with-open-file (out outfile :direction :output))
      (do ((line (read in NIL NIL) (read in NIL NIL)))
       ((null line))
       (let ((word (car line))
             (result NIL)
             (phonestring (prepare (cadr line))))
       (if (phn word)
           (progn (princ "$$$$ Working on: ") (princ word) (fresh-line)
                  (princ "$$$$ Recognizer output: ") (princ phonestring) (fresh-line)
                  (princ "$$$$ Dictionary phones: ") (princ (phn word)) (fresh-line)
                  (fresh-line)
                  (setf result (new-n-best phonestring number :level level
                                           :searchbound searchbound :test
                                           word))
                  (if (car result)
                      (rplaca summary (cons (cons word (cdr result)) (car summary)))
                    (rplaca (cdr summary) (cons
                                           (cons word (cdr result)) (cadr summary))))

                  )
         (progn (princ "%%%%%%  ") (princ word)
                (princ " not in dictionary.") (fresh-line)))))

    summary
    ))
```

DENK

SamenwerkingsOrgaan Brabantse Universiteiten

Proceedings of the

International Conference on Cooperative Multimodal Communication CMC/95

Part I

Multimodal Maps: An Agent-based Approach

Adam Cheyer and Luc Julia

SRI International
333 Ravenswood Ave
Menlo Park, CA 94025 - USA

Abstract

In this paper, we discuss how multiple input modalities may be combined to produce more natural user interfaces. To illustrate this technique, we present a prototype map-based application for a travel planning domain. The application is distinguished by a synergistic combination of handwriting, gesture and speech modalities; access to existing data sources including the World Wide Web; and a mobile handheld interface. To implement the described application, a hierarchical distributed network of heterogeneous software agents was augmented by appropriate functionality for developing synergistic multimodal applications.

Key words: Multimodal Interface, Agent Architecture, Distributed Artificial Intelligence.

1 Introduction

As computer systems become more powerful and complex, efforts to make computer interfaces more simple and natural become increasingly important. Natural interfaces should be designed to facilitate communication in ways people are already accustomed to using. Such interfaces allow users to concentrate on the tasks they are trying to accomplish, not worry about what they must do to control the interface.

In this paper, we begin by discussing what input modalities humans are comfortable using when interacting with computers, and how these modalities should best be combined in order to produce natural interfaces. In section three, we present a prototype map-based application for the travel planning domain which uses a synergistic combination of several input modalities. Section four describes the agent-based approach we used to implement the application and the work on which it is based. In section five, we summarize our conclusions and future directions.

2 Natural Input

2.1 Input Modalities

Direct manipulation interface technologies are currently the most widely used techniques for creating user interfaces. Through the use of menus and a graphical user interface, users are presented with sets of discrete actions and the objects on which to perform them. Pointing devices such as a mouse facilitate selection of an object or action, and drag and drop techniques allow items to be moved or combined with other entities or actions.

With the addition of electronic pen devices, gestural drawings add a new dimension direct manipulation interfaces. Gestures allow users to communicate a surprisingly wide range of meaningful requests with a few simple strokes. Research has shown that multiple gestures can be combined to form dialog, with rules of temporal grouping overriding temporal sequencing [22]. Gestural commands are particularly applicable to graphical or editing type tasks.

Direct manipulation interactions possess many desirable qualities: communication is generally fast and concise; input techniques are easy to learn and remember; the user has a good idea about what can be accomplished, as the visual presentation of the available actions is generally easily accessible. However, direct manipulation suffers from limitations when trying to access or describe entities which are not or can not be visualized by the user.

Limitations of direct manipulation style interfaces can be addressed by another interface technology, that of natural language interfaces. Natural language interfaces excel in describing entities that are not currently displayed on the monitor, in specifying temporal relations between entities or actions, and in identifying members of sets. These strengths are exactly the weaknesses of direct manipulation interfaces, and concurrently, the weaknesses of natural language interfaces (ambiguity, conceptual coverage, etc.) can be overcome by the strengths of direct manipulation.

Natural language content can be entered through different input modalities, including typing, handwriting, and speech. It is important to note that, while the same textual content can be provided by the three modalities, each modality has widely varying properties.

- Spoken language is the modality used first and foremost in human-human interactive problem solving [4]. Speech is an extremely fast medium, several times faster than typing or handwriting. In addition, speech input contains content that is not present in other forms of natural language input, such as prosidy, tone and characteristics of the speaker (age, sex, accent).

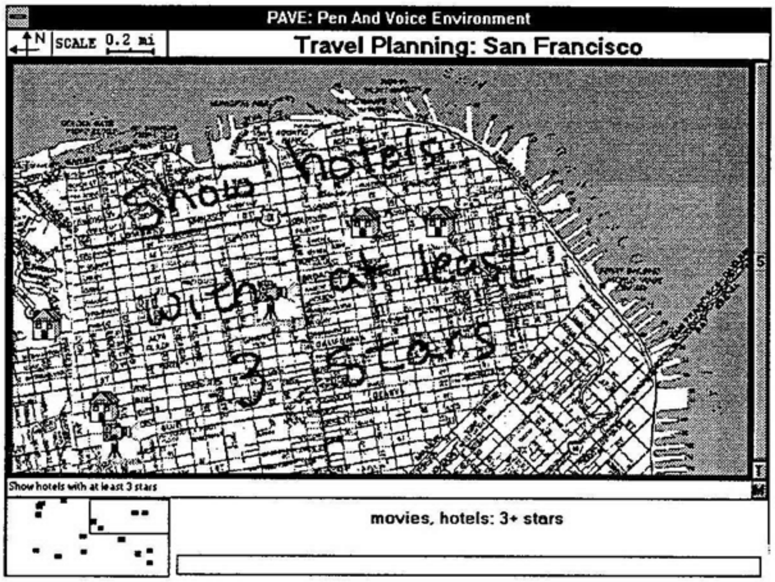
- Typing is the most common way of entering information into a computer, because it is reasonably fast, very accurate, and requires no computational resources.

- Handwriting has been shown to be useful for certain types of tasks, such as performing numerical calculations and manipulating names which are difficult to pronounce [18, 19]. Because of its relatively slow production rate, handwriting may induce users to produce different types of input than is generated by spoken language; abbreviations, symbols and non-grammatical patterns may be expected to be more prevalent amid written input.

2.2 Combination of Modalities

As noted in the previous section, direct manipulation and natural language seem to be very complementary modalities. It is therefore not surprising that a number of multimodal systems combine the two.

Notable among such systems is the Cohen's Shoptalk system [6], a prototype manufacturing and decision-support system that aids in tasks such as quality assurance monitoring, and production scheduling. The natural language module of Shoptalk is based on the Chat-85


PAVE: Pen And Voice Environment

Travel Planning: San Francisco

Show hotels with at least 3 stars

Show hotels with at least 3 stars

movies, hotels: 3+ stars

Figure 1: Multimodal Application for Travel Planning

natural language system [25] and is particularly good at handling time, tense, and temporal reasoning.

A number of systems have focused on combining the speed of speech with the reference provided by direct manipulation of a mouse pointer. Such systems include the XTRA system [1], CUBRICON [15], the PAC-Amodeus model [16], and TAPAGE [9].

XTRA and CUBRICON are both systems that combine complex spoken input with mouse clicks, using several knowledge sources for reference identification. CUBRICON's domain is a map-based task, making it similar to the application developed in this paper. However, the two are different in that CUBRICON can only use direct manipulation to indicate a specific item, whereas our system produces a richer mixing of modalities by adding both gestural and written language as input modalities.

The PAC-Amodeus systems such as VoicePaint and Notebook allow the user to synergistically combine vocal or mouse-click commands when interacting with notes or graphical objects. However, due to the selected domains, the natural language input is very simple, generally of the style "Insert a note here."

TAPAGE is another system that allows true synergistic combination of spoken input with direct manipulation. Like PAC-Amodeus, TAPAGE's domain provides only simple linguistic input. However, TAPAGE uses a pen-based interface instead of a mouse, allowing gestural commands. TAPAGE, selected as a building block for our map application, will be described more in detail in section 4.2.

Other interesting work regarding the simultaneous combination of handgestures and gaze can be found in [2, 13].

3 A Multimodal Map Application

In this section, we will describe a prototype map-based application for a travel planning domain. In order to provide the most natural user interface possible, the system permits the user to simultaneously combine direct manipulation, gestural drawings, handwritten, typed and spoken natural language When designing the system, other criteria were considered as well:

- The user interface must be light and fast enough to run on a handheld PDA while able to access applications and data that may require a more powerful machine.
- Existing commercial or research natural language and speech recognition systems should be used.
- Through the multimodal interface, a user must be able to transparently access a wide variety of data sources, including information stored in HTML form on the World Wide Web.

As illustrated in Figure 1, the user is presented with a pen sensitive map display on which drawn gestures and written natural language statements may be combined with spoken input. As opposed to a static paper map, the location, resolution, and content presented by the map change, according to the requests of the user. Objects of interest, such as restaurants, movie theaters, hotels, tourist sites, municipal buildings, etc. are displayed as icons. The user may ask the map to perform various actions. For example :

- *distance calculation* : e.g. "How far is the hotel from Fisherman's Wharf?"
- *object location* : e.g. "Where is the nearest post office?"
- *filtering* : e.g. "Display the French restaurants within 1 mile of this hotel."
- *information retrieval* : e.g. "Show me all available information about Alcatraz."

The application also makes use of multimodal (multimedia) output as well as input: video, text, sound and voice can all be combined when presenting an answer to a query.

During input, requests can be entered using gestures (see Figure 2 for sample gestures), handwriting, voice, or a combination of pen and voice. For instance, in order to calculate the distance between two points on the map, a command may be issued using the following:

- *gesture*, by simply drawing a line between the two points of interest.
- *voice*, by speaking "What is the distance from the post office to the hotel?".
- *handwriting*, by writing "dist p.o. to hotel?"
- *synergistic combination of pen and voice*, by speaking "What is the distance from here to this hotel?" while simultaneously indicating the specified locations by pointing or circling.

Notice that in our example of synergistic combination of pen and voice, the arguments to the verb "distance" can be specified before, at the same time, or shortly after the vocalization of the request to calculate the distance. If a user's request is ambiguous or underspecified, the system will wait several seconds and then issue a prompt requesting additional information.

The user interface runs on pen-equipped PC's or a Dauphin handheld PDA ([7]) using either a microphone or a telephone for voice input. The interface is connected either by

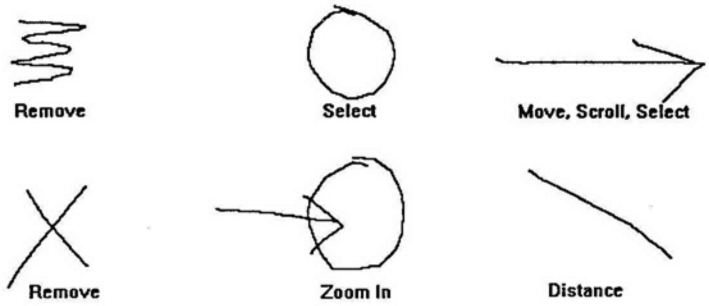


Figure 2: Sample gestures

modem or ethernet to a server machine which will manage database access, natural language processing and speech recognition for the application. The result is a mobile system that provides a synergistic pen/voice interface to remote databases.

In general, the speed of the system is quite acceptable. For gestural commands, which are handled locally on the user interface machine, a response is produced in less than one second. For handwritten commands, the time to recognize the handwriting, process the English query, access a database and begin to display the results on the user interface is less than three seconds (assuming an ethernet connection, and good network and database response). Solutions to verbal commands are displayed in three to five seconds after the end of speech has been detected; partial feedback indicating the current status of the speech recognition is provided earlier.

4 Approach

In order to implement the application described in the previous section, we chose to augment a proven agent- based architecture with functionalities developed for a synergistically multimodal application. The result is a flexible methodology for designing and implementing distributed multimodal applications.

4.1 Building Blocks

4.1.1 Open Agent Architecture

The Open Agent Architecture (OAA) [5] provides a framework for coordinating a society of agents which interact to solve problems for the user. Through the use of agents, the OAA provides distributed access to commercial applications, such as mail systems, calendar programs, databases, etc.

The Open Agent Architecture possesses several properties which make it a good candidate for our needs:

- An Interagent Communication Language (ICL) and Query Protocol have been developed, allowing agents to communicate among themselves. Agents can run on different platforms and be implemented in a variety of programming languages.

- Several natural language systems have been integrated into the OAA which convert English into the Interagent Communication Language. In addition, a speech recognition agent has been developed to provide transparent access to the Corona speech recognition system.

- The agent architecture has been used to provide natural language and agent access to various heterogeneous data and knowledge sources.

- Agent interaction is very fine-grained. The architecture was designed so that a number of agents can work together, when appropriate in parallel, to produce fast responses to queries.

The architecture for the OAA, based loosely on Schwartz's FLiPSiDE system[23], uses a hierarchical configuration where client agents connect to a "facilitator" server. Facilitators provide content-based message routing, global data management, and process coordination for their set of connected agents. Facilitators can, in turn, be connected as clients of other facilitators. Each facilitator records the published functionality of their sub-agents, and when queries arrive in Interagent Communication Language form, they are responsible for breaking apart any complex queries and for distributing goals to the appropriate agents. An agent solving a goal may require supporting information and the agent architecture provides numerous means of requesting data from other agents or from the user.

Among the assortment of agent architectures, the Open Agent Architecture can be most closely compared to work by the ARPA knowledge sharing community [10]. The OAA's query protocol, Interagent Communication Language and Facilitator mechanisms have similar instantiations in the SHADE project, in the form of KQML, KIF and various independent capability matchmakers. Other agent architectures, such as General Magic's Telescript [11], MASCOS [20], or the CORBA distributed object approach [17] do not provide as fully developed mechanisms for interagent communication and delegation.

The Open Agent Architecture provides capability for accessing distributed knowledge sources through natural language and voice, but it is lacking integration with a synergistic multimodal interface.

4.1.2 TAPAGE

TAPAGE (edition de Tableaux par la Parole et la Geste) is a synergistic pen/voice system for designing and correcting tables.

To capture signals emitted during a user's interaction, TAPAGE integrates a set of modality agents, each responsible for a very specialized kind of signal [9]. The modality agents are connected to an "interpret agent" which is responsible for combining the inputs across all modalities to form a valid command for the application. The interpret agent receives filtered results from the modality agents, sorts the information into the correct fields, performs type-checking on the arguments, and prompts the user for any missing information, according to the model of the interaction. The interpret agent is also responsible for merging the data streams sent by the modality agents, and for resolving ambiguities among them, based on its knowledge of the application's internal state. Another function of the interpret agent is to produce reflexes: reflexes are actions output at the interface level without involving the functional core of the application.

The TAPAGE system can accept multimodal input, but it is not a distributed system; its functional core is fixed. In TAPAGE, the set of linguistic input is limited to a *verb object argument* format.

4.2 Synthesis

In the Open Agent Architecture, agents are distributed entities that can run on different machines, and communicate together to solve a task for the user. In TAPAGE, agents are used to provide streams of input to a central interpret process, responsible for merging incoming data. A generalization of these two types of agents could be :

Macro Agents: contain some knowledge and ability to reason about a domain, and can answer or make queries to other macro agents using the Interagent Communication Language.

Micro Agents: are responsible for handling a single input or output data stream, either filtering the signal to or from a hierarchically superior "interpret" agent.

The network architecture that we used was hierarchical at two resolutions – micro agents are connected to a superior macro agent, and macro agents are connected in turn to a facilitator agent. In both cases, a server is responsible for the supervision of its client sub-agents.

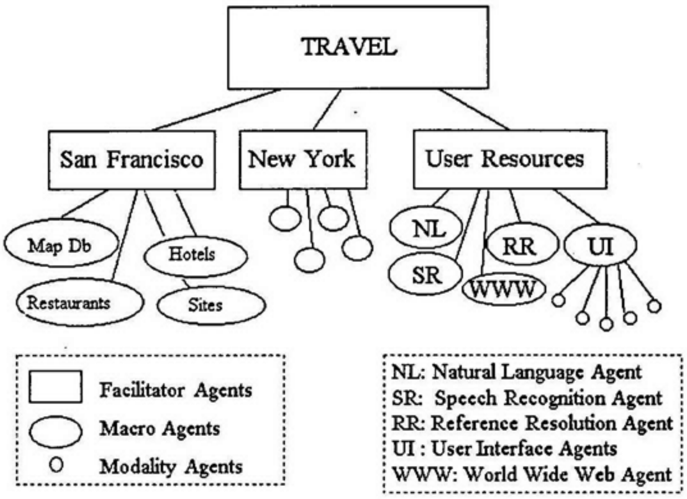
In order to describe our implementation, we will first give a description of each agent used in our application and then illustrate the flow of communication among agents produced by a user's request.

Speech Recognition (SR) Agent: The SR agent provides a mapping from the Interagent Communication Language to the API for the Decipher (Corona) speech recognition system [4], a continuous speech speaker independent recognizer based on Hidden Markov Model technology. This macro agent is also responsible for supervising a child micro agent whose task is to control the speech data stream. The SR agent can provide feedback to an interface agent about the current status and progress of the micro agent (e.g. "listening", "end of speech detected", etc.) This agent is written in C.

Natural Language (NL) Parser Agent: translates English expressions into the Interagent Communication Language (ICL). For a more complete description of the ICL, see [5]. The NL agent we selected for our application is the simplest of those integrated into the OAA. It is written in Prolog using Definite Clause Grammars, and supports a distributed vocabulary; each agent dynamically adds word definitions as it connects to the network. A current project is underway to integrate the Gemini natural language system [4], a robust bottom up parser and semantic interpreter specifically designed for use in Spoken Language Understanding projects.

Database Agents: Database agents can reside at local or remote locations and can be grouped hierarchically according to content. Micro agents can be connected to database agents to monitor relevant positions or events in real time. In our travel planning application, database agents provide maps for each city, as well as icons, vocabulary and information about available hotels, restaurants, movies, theaters, municipal buildings and tourist attractions. Three types of databases were used: Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the World Wide Web (WWW). In one instance, a local newspaper provides weekly updates to its Mosaic-accessible list of current movie times and reviews, as well as adding several new restaurant reviews to a growing collection; this information is extracted by an HTML reading database agent and made accessible to the agent architecture. Descriptions and addresses of new restaurants are presented to the user on request, and the user can choose to add them to the permanent database by specifying positional coordinates on the map (eg. "add this new restaurant here"), information lacking in the WWW database.

Reference Resolution Agent: This agent is responsible for merging requests arriving in parallel from different modalities, and for controlling interactions between the user interface



TRAVEL
San Francisco
New York
User Resources
Map Db
Hotels
Restaurants
Sites
NL
SR
RR
UI
WWW
Facilitator Agents
Macro Agents
Modality Agents
NL: Natural Language Agent
SR: Speech Recognition Agent
RR: Reference Resolution Agent
UI : User Interface Agents
WWW: World Wide Web Agent

Figure 3: Agent Architecture for Map Application

agent, database agents and modality agents. In this implementation, the reference resolution agent is domain specific: knowledge is encoded as to what actions must be performed to resolve each possible type of ICL request in its particular domain. For a given ICL logical form, the agent can verify argument types, supply default values, and resolve argument references. Some argument references are descriptive ("How far is it to the hotel on Emerson Street?"); in this case, a domain agent will try to resolve the definite reference by sending database agent requests. Other references, particularly when contextual or deictic, are resolved by the user interface agent ("What are the rates for this hotel?"). Once arguments to a query have been resolved, this agent agent coordinates the actions and calculations necessary to produce the result of the request.

Interface Agent: This macro agent is responsible for managing what is currently being displayed to the user, and for accepting the user's multimodal input. The Interface Agent also coordinates client modality agents and resolves ambiguities among them : handwriting and gestures are interpreted locally by micro agents and combined with results from the speech recognition agent, running on a remote speech server. The handwriting micro-agent interfaces with the Microsoft PenWindows API and accesses a handwriting recognizer by CIC Corporation. The gesture micro- agent accesses recognition algorithms developed for TAPAGE.

An important task for the interface agent is to record which objects of each type are currently salient, in order to resolve contextual references such as "the hotel" or "where I was before." Deictic references are resolved by gestural or direct manipulation commands. If no such indication is currently specified, the user interface agent waits long enough to give the user an opportunity to supply the value, and then prompts the user for it.

We shall now give an example of the distributed interaction of agents for a specific query. In the following example, all communication among agents passes transparently through a facilitator agent in an undirected fashion; this process is left out of the description for brevity.

1. A user speaks: "How far is the restaurant from this hotel?"

2. The speech recognition agent monitors the status and results from its micro agent, sending feedback received by the user interface agent. When the string is recognized, a translation is requested.

3. The English request is received by the NL agent and translated into ICL form.

4. The reference resolution agent (RR) receives the ICL distance request containing one definite and one deictic reference and asks for resolution of these references.

5. The interface agent uses contextual structures to find what "the restaurant" refers to, and waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary.

6. When the references have been resolved, the domain agent (RR) sends database requests asking for the coordinates of the items in question. It then calculates the distance according to the scale of the currently displayed map, and requests the user interface to produce output displaying the result of the calculation.

5 Conclusions

By augmenting an existing agent-based architecture with concepts necessary for synergistic multimodal input, we were able to rapidly develop a map-based application for a travel planning task. The resulting application has met our initial requirements: a mobile, synergistic pen/voice interface providing good natural language access to heterogeneous distributed knowledge sources. The approach used was general and should provide a for developing synergistic multimodal applications for other domains.

The system described here is one of the first that accepts commands made of synergistic combinations of spoken language, handwriting and gestural input. This fusion of modalities can produce more complex interactions than in many systems and the prototype application will serve as a testbed for acquiring a better understanding of multimodal input.

In the near future, we will continue to verify and extend our approach by building other multimodal applications. We are interested in generalizing the methodology even further; work has already begun on an agent-building tool which will simplify and automate many of the details of developing new agents and domains.

References

[1] Allegayer, J, Jansen-Winkeln, R., Reddig, C. and Reithinger, N. "Bidirectional use of knowledge in the multi-modal NL access system XTRA". In Proceedings of IJCAI-89, Detroit, pp. 1492-1497.

[2] Bolt, R. "Put that there: Voice and Gesture at the Graphic Interface". Computer Graphics, 14(3), 1980, pp. 262-270.

[3] Bellik, Y. and Teil, D. "Les types de multimodalites", In Proc. IIM'92 (Paris), pp. 22-28.

[4] Cohen, M., Murveit, H., Bernstein, J., Price, P., Weintraub, M., "The DECIPHER Speech Recognition System". 1990 IEEE ICASSP, pp. 77-80.

[5] Cohen, P.R., Cheyer, A., Wang, M. and Baeg, S.C. "An Open Agent Architecture". In Proc. AAAI'94 - SA (Stanford), pp. 1-8.

[6] Cohen, P. "The role of natural language in a multimodal interface". Proceedings of UIST'92, 143-149.

[7] Dauphin DTR-1 User's Manual, Dauphin Technology, Inc. 337 E. Butterfield Rd., Suite 900, Lombard, Ill 60148.

[8] Dowding, J., Gawron, J.M., Appelt, D., Bear, J., Cherny, L., Moore, B. and Moran D., "Gemini: A natural language system for spoken-language understanding", Technical Note 527, AI Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, April 1993.

[9] Faure, C. and Julia, L. "An Agent-Based Architecture for a Multimodal Interface". In Proc. AAAI'94 - IM4S (Stanford), pp. 82-86.

[10] Genesereth, M. and Singh, N.P. "A knowledge sharing approach to software interoperation". Computer Science Department, Stanford University, unpublished ms., 1994.

[11] General Magic, Inc., "Telescript Product Documentation", 1995.

[12] Julia, L. and Faure, C. "A Multimodal Interface for Incremental Graphic Document Design". HCI International '93, Orlando.

[13] Koons, D.B., Sparrell, C.J., and Thorisson, K.R. "Integrating Simultaneous Input from Speech, Gaze and Hand Gestures". In *Intelligent Multimedia Interfaces*, Edited by Mark Maybury, Menlo Park, CA, AAAI Press, 1993.

[14] Maybury, M.T. (ed.), *Intelligent Multimedia Interfaces*, AAAI Press/MIT Press: Menlo Park, Ca, 1993.

[15] Neal, J.G., and Shapiro, S.C. "Intelligent Multi-media Interface Technology". In *Intelligent User Interfaces*, Edited by J. Sullivan and S. Tyler, Addison-Wesley Pub. Co., Reading, MA, 1991.

[16] Nigay, L. and Coutaz, J. "A Design Space for Multimodal Systems: Concurrent Processing and Data Fusion". In Proc. InterCHI'93 (Amsterdam), ACM Press, pp. 172-178.

[17] Object Management Group, "The Common Object Request Broker: Architecture and Specification", OMG Document Number 91.12.1, December 1991.

[18] Oviatt, S. "Toward Empirically-Based Design of Multimodal Dialogue Systems". In Proc. AAAI'94 - IM4S (Stanford), pp. 30-36.

[19] Oviatt, S. and Olsen, E. "Integration Themes in Multimodal Human-Computer Interaction". Proceedings of ICSLP'94, Yokohama, pp. 551-554.

[20] Park, S.K., Choi J.M., Myeong-Wuk J., Lee G.L., and Lim Y.H. "MASCOS : A Multi-Agent System as the Computer Secretary". Submitted for publication.

[21] Pfaff, G. and Ten Hagen, P.J.W. *Seeheim workshop on User Interface Management Systems* (Berlin), Springer- Verlag.

[22] Rhyne J. "Dialogue Management for Gestural Interfaces". Computer Graphics, 21(2), 1987, pp. 137-142.

[23] Schwartz, D.G. "Cooperating heterogeneous systems: A blackboard-based meta approach". Technical Report 93-112, Center for Automation and Intelligent Systems Research, Case Western Reserve University, Cleveland Ohio, April 1993. Unpublished PhD. thesis.

[24] Sullivan, J. and Tyler, S. (eds.), *Intelligent User Interfaces*, Addison-Wesley Pub. Co., Reading, MA, 1991.

[25] Warren, D. and Pereira, F., "An Efficient Easily Adaptable System for Interpreting Natural Language Queries", in American Journal of Computational Linguistics, 8(3), 1982, pp. 110-123.

[26] Wauchope, K., "Eucalyptus: Integrating Natural Language with a Graphical User Interface." Naval Research Laboratory Technical Report NRL/FR/5510-94-9711, in press, 1994.

International Journal of Software Engineering and Knowledge Engineering
Vol. 7 No. 3 (1997) 351–369
© World Scientific Publishing Company

A FRAMEWORK FOR MULTI-AGENT SYSTEMS WITH MULTI-MODAL USER INTERFACES IN DISTRIBUTED COMPUTING ENVIRONMENTS

SANKYU PARK
AI Section, Electronics and Telecommunications Research Institute,
161 Kajong-Dong, Yusong-Gu, Taejon 305-350, Korea
e-mail: skpark@com.etri.re.kr

KEY-SUN CHOI
Computer Systems Lab., Computer Science Department,
Korea Advanced Institute of Science and Technology,
Yusong P.O. Box 111, Taejon 305-600, Korea
e-mail: kschoi@world.kaist.ac.kr

K. H. (KANE) KIM
Department of Electrical and Computer Engineering, University of California,
ET544E Irvine, CA 92697, USA
e-mail: kane@ece.uci.edu

Received 10 December 1996
Revised 1 March 1996
Accepted 10 March 1997

In current multi-agent systems, the user is typically interacting with a single agent at a time through relatively inflexible and modestly intelligent interfaces. As a consequence, these systems force the users to submit simplistic requests only and suffer from problems such as the low-level nature of the system services offered to users, the weak reusability of agents, and the weak extensibility of the systems. In this paper, a framework for multi-agent systems called the open agent architecture (OAA) which reduces such problems, is discussed. The OAA is designed to handle complex requests that involve multiple agents. In some cases of complex requests from users, the components of the requests do not directly correspond to the capabilities of various application agents, and therefore, the system is required to translate the user's model of the task into the system's model before apportioning subtasks to the agents. To maximize users' efficiency in generating this type of complex requests, the OAA offers an intelligent multi-modal user interface agent which supports a natural language interface with a mix of spoken language, handwriting, and gesture. The effectiveness of the OAA environment including the intelligent distributed multi-modal interface has been observed in our development of several practical multi-agent systems.

Keywords: Multi-agent systems, open agent architecture, multi-modal, user interface, natural language, distributed computing environments.

1. Introduction

In current multi-agent systems, the user is typically interacting with a single agent at a time through relatively inflexible and modestly intelligent interfaces. As a consequence, these systems force the users to submit simplistic requests only and suffer from problems such as the low-level nature of the system services offered to users, the weak reusability of agents, and the weak extensibility of the systems. Another problem in such systems is that dynamic extension of agents is costly and thus the application's domain tends to be fixed during the early phases of development and the wider reuse of the expertise embodied in an existing agent developed in an earlier project is inhibited. In order to provide general multi-agent facilities [8,9,12] which are not only distributed and easily extensible but also highly reusable, the Open Agent Architecture [OAA] [14] was formulated and a supporting intelligent multi-modal user interface agent [4, 15] has been designed and implemented. The OAA is also an effective framework for facilitating mobile computing.

1.1. *Goals of the OAA*

The OAA is a multi-agent system architecture that supports the creation of applications from the agents that were not designed to work together. It is intended to facilitate effective reuse of the expertise embodied in an agent. Key attributes of the OAA are the following.

- Open: The OAA supports agents written in multiple languages and hosted on multiple platforms. The current prototype implementations support languages such as C, Prolog, Lisp, Java, Microsoft's Visual Basic, and Borland's Delphi. They run on platforms such as PCs (Windows 3.1 and 95), Sun Workstations (Solaris 1.1 and 2.x), and SGI Workstations. This feature enables integrating a wide variety of agents developed in different projects.
- Distributed: The agents that compose an application can run cooperatively on multiple platforms.
- Extensible: Agents can be added to the system while it is running, and their capabilities will become immediately available to the pre-existing agents. Similarly, agents can be dynamically removed from the system.

A core of the OAA is the user interface, which can be viewed as a support for human agents to access the mechanized agents. We have developed agent-based multi- modal user interfaces which are structured in the same way the back-ends of these applications are structured. Key attributes of the multi-modal user interface agents are the following.

- Mobile: OAA-based applications can be utilized by the user equipped with a lightweight portable computer because only the user interface agents need to run on the portable computer. The interface agents enable the user to access a range of mechanized application agents running on other platforms.

- Collaborative: The user interface is implemented in the form of an agent, and thus the user appears to be just another agent to the mechanized application agents. This greatly simplifies the job of creating systems in which multiple humans and mechanized agents cooperate.
- Multiple Modalities: The user interface supports handwriting, gesture, and spoken language in addition to the traditional graphical user interface modality. Users can even enter commands with a mix of modalities, for example, a spoken command in which the object to be acted on is identified by a pen gesture (or other graphical pointing operation).

It is our viewpoint that a natural language interface is an appropriate component because it provides effective support to the users in stating complex queries. One of the challenges that are encountered in using natural languages in a multi-agent system is to combine the languages related to different agents. We have created an Agent Development Toolkit [2] to help the programmer create agents, and the toolkit allows the programmer to specify the vocabulary, the grammar, and the logical form predicates for a new agent.

1.2. *Comparison with other agent architectures*

Among the assortment of agent architectures, the OAA can be most closely compared to the work by the DARPA knowledge sharing community [6, 12, 16]. The OAA's query protocol, the inter-agent communication language (ICL), and the facilitator mechanisms (to be discussed in Sec. 2) have similarity with the mechanisms adopted in the SHADE project, e.g., KQML [16], KIF [12], and various independent capability matchmakers. But the multi-modal user interface agent in our OAA is unique. Other agent architectures, such as General Magic's Telescript [7] and the CORBA distributed object approach [12], do not provide as fully developed mechanisms for inter-agent communication and delegation as those in the OAA.

Genesereth and Singh's architecture [6] is more ambitious than ours in employing a full first-order logic as the agent communicaton language. As yet, there is no need to expand our language beyond Horn clauses with temporal constraints although an expansion step may become necessary in the future. However, in their architecture, there is also no consideration about adopting agent-structured user interfaces. Genesereth and Singh use KIF [12] as their basic language of predicates and as a knowledge integration strategy. Our user interface architecture requires capabilities for merging contributions by different agents of their natural language vocabularies, related pronunciations, and semantic mappings of those vocabulary items to underlying predicates.

ARCHON [9] introduces the agent layer on top of each domain-specific problem solver that is responsible for interactions with other agents. The agent layer maintains information about other agents as well as about itself. ARCHON emphasizes the independence of an agent, and interactions among agents are carried out via peer-to-peer communication among agents without employing a separate facilitator.

This strategy yields good reliability of the system because the failure of any one agent affects only a limited part of the system services. However, maintaining consistency among agents of their views about the agent community is difficult since each agent must update its capability knowledge base whenever a new agent joins the community and whenever an agent withdraws from the community.

Kautz [5] approaches the agent platform in a bottom-up fashion. They began by identifying possible, useful, and feasible tasks for a software item and then identifying the necessary properties of an effective agent platform. They first tackled the problem of handling the communication involved in scheduling a visit to their laboratory for a visitor. They offered an empirical agent system which had a task-specific agent, taskbot, and a user-specific agent, userbot. As we did in the OAA, they separated the user interaction part from the task handling part. However, their implementation is a single agent that consists of a taskbot and a userbot with direct communications between them. In contrast, our system integrates several agents to perform cooperative tasks with multiple interactions among them. As mentioned earlier, we also integrate all the interfaces of application agents into one representative multi-modal user interface agent by which user requests are analyzed, decomposed and transferred to application agents.

2. The Framework of Multi-agent Architecture

2.1. *Overview of the OAA*

Based loosely on Schwartz's FLiPSiDE which is a blackboard-based system [3], the OAA is a framework[17] allowing individual client software — *application agents* — to communicate by means of goals posted on a blackboard controlled by a server process — *facilitator* — agent.

Figure 1 presents the basic structure of the Open Agent Architecture (OAA). In this architecture, application agents are called *clients* because each acts (in some respects) as a client of the facilitator. The facilitator as a server is responsible for storing data that is global to the application agents, for identifying application agents that can achieve various goals, and for scheduling and maintaining the flow of communication during distributed computation. When initialized, an application agent makes a connection to the facilitator. Upon connection, an application agent informs its facilitator of the services it is capable of providing.

An application agent consists of an agent *interface layer* above a *domain knowledge* layer. The *domain knowledge* layer, in turn, may lie on top of existing standalone applications (e.g., mailers, calendar programs, databases). The *domain knowledge* layer can access the functionality of the underlying application through one or more of the following several means:

(1) manipulation of files (e.g., mail spool, calendar datafiles),
(2) calls to an application's application programmer interface (API) (e.g., MAPI in Microsoft Windows),

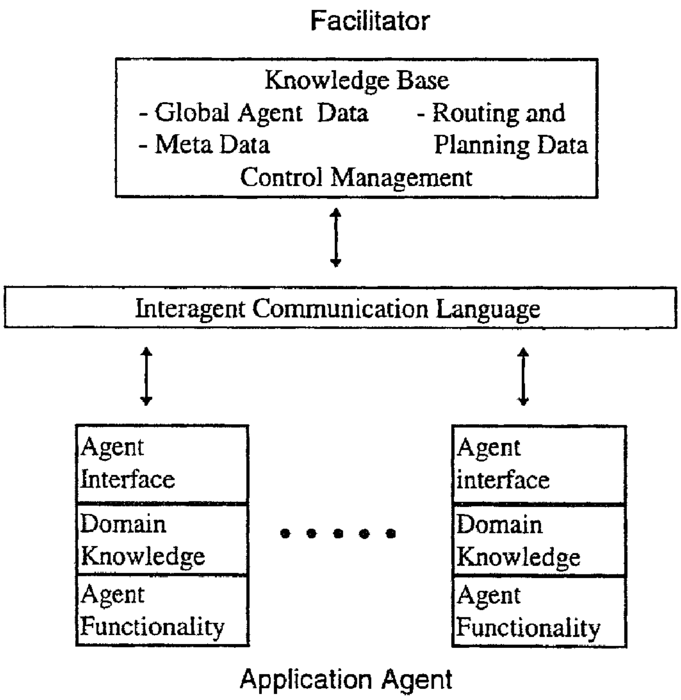


Fig. 1. OAA structure.

(3) a scripting language, or

(4) interpretation of an operating system's message events.

Each OAA communication between application agents passes through the facilitator. OAA agents communicate with each other in a high-level logical language called the Interagent Communication Language (ICL). An extension of Prolog has been chosen as ICL to take advantage of the unification and backtracking features when posting queries. ICL is similar in style and functionality to the core Knowledge Query and Manipulation Language (KQML) of the DARPA Knowledge Sharing Effort [6, 10]. The differences are a result of our focus on the user interface: ICL was designed to be compatible with the output of our natural language understanding systems, thereby simplifying the transformation of a user's query or command into one that can be handled by the mechanized agents.

2.2. *Multiple facilitator configuration of OAA in distributed environments*

In the description above, the OAA contains one blackboard server process called the facilitator, and many client application agents; client agents are permitted to execute on different host machines. Actually the OAA may contain a hierarchy of facilitators and a facilitator may itself be a client; if none of its application agents can solve a particular goal, this goal may be passed further up in the hierarchy. Figure 2 presents a hierarchy of facilitator agents, which could be distributed over a network. When a goal (G) is posted to a local facilitator (F1), and F1 determines that none of its application agents have the requisite knowledge to achieve the goal, it propagates the goal to a more senior facilitator (F4) in the hierarchy. (Application agents here refer to non-facilitator agents, which are not shown in the figure.) A senior facilitator agent maintains a knowledge base of the goals that its lower level facilitators can solve. When a senior facilitator agent receives such a request, it in turn propagates the request either down to one of its child facilitators which can solve the goal or up to its next higher senior facilitator. Figure 2 shows the flow of the goal when F9 can solve it. In the rest of this paper, the entire hierarchy is referred to as the facilitator.

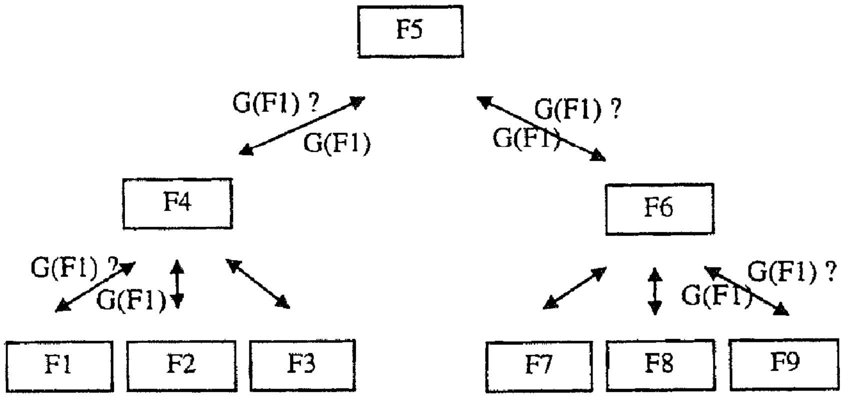


Fig. 2. Multiple facilitators arranged in a hierarchy.

2.3. *Infrastructure*

2.3.1. *Facilitator*

In the OAA framework, the facilitator plays a key role. When an agent is added to the application, it registers its capabilities with the facilitator. Part of this registration is the natural language vocabulary that can be used to talk about the tasks that the agent can perform. When an agent needs work done by other agents within the application, it sends a request to the facilitator, which then delegates it to an agent, or agents, that have registered that they can handle the needed tasks. The goal of the facilitator is to minimize the information and assumptions that the developer must embed in an agent, thereby making it easier to reuse agents in disparate applications.

The facilitator is responsible for managing three types of tasks for its set of application agents.

- Controller: The facilitator acts as a controller, deciding which enabled agents can consistently be executed and giving each of them a thread of control with which to proceed. Results from any given application agent are again posted on the facilitator for action by other agents. Thus, although agents are asynchronous processes, they can be coordinated by the controller at the time of invocation.

- Communication Router: The facilitator acts as a router, distributing messages and data among agents. Except in a few special cases, all agent communications pass through the facilitator. While this may add a small overhead to interagent communication, the advantage of flexibility and adaptability greatly outweighs the disadvantages.
- Knowledge Base Server: The facilitator serves as a repository for the facts about the user accumulated from both communication and observation, and the facts about the history of interactions, rules, and preferences. The knowledge base is needed for analyzing user inputs in order to support the agents' reasoning, and maintain the context of interaction. At the time of its creation, each agent registers its particular knowledge base with the knowledge base server, thereby allowing the agent to supply knowledge specific to its function and allowing other agents access to this information as needed.

We are interested in user interfaces that support interactions with a broad community of agents, and the facilitator is a key to handling complex queries. The facilitator (and supporting agents: e.g., ADT agents, UI agents, etc.) handles the translation of the user's model of the task into the system model (analogous to how natural language interfaces to databases handle transforming the user's model into the database's schemas). Also, the facilitator simplifies reusing agents in new applications. If a community of agents is assembled using agents acquired from other communities, those agents cannot be assumed to make atomic requests that can be handled by other agents: simple requests in one application may be implemented by a combination of agents in another application. The facilitator is responsible for decomposing complex requests and translating the terminology used.

In the OAA, the facilitator is a potential bottleneck if there is a high volume of communication among the agents. Our focus has been on supporting a natural user interface to a very large community of intelligent agents, and this environment produces a relatively low volume of communication through the facilitator.

2.3.2. *Interagent Communication Language* (*ICL*)

The ICL of OAA is the interface language shared by all agents, no matter what machine they are running on or what computer language they are programmed in. When using the ICL, an agent transmits messages composed in Prolog-like syntax, providing the familiar syntax and rich semantics of first order predicate logic, wrapped in an ICL.

An ICL message is conceptually a layered message consisting of the wrapper layer, the primitive layer, and the message body layer.

The wrapper layer encodes into the messages a set of features that describe the lower communication parameters such as the identity of the sender agent.

Agents communicating with each other require a well-known set of conventions. This set of conventions comprises a protocol that must be implemented at both ends of a connection. This protocol is specified in the primitive layer and determines the kinds of interactions one can have with another agent. This primitive signifies that the message is a query, a command, or any other mutually agreed-upon speech act [13].

Our current OAA environment supports seven basic primitives: *post-query, solve, solved, add-trigger, register-solvable-goals, read-bb, write-bb*. The *post-query* primitive is used to post a query to the facilitator. This primitive causes the facilitator to determine appropriate application agents that may handle this query, to send the agents a new ICL message containing the *solve* primitive, and also to add a trigger if necessary. The application agents receiving the ICL message with the *solve* primitive perform the tasks to achieve the relevant goal and return the results. All the results are conveyed by using an ICL message containing the *solved* primitive. This primitive denotes that the content of the message body layer is the solution to a query. The *add-trigger* primitive is used to let an agent do something *whenever* a condition is satisfied. The *register-solvable-goals* primitive allows agents to register their capabilities with the facilitator. Finally, the *read-bb* primitive is for reading data from the facilitator's blackboard and the *write-bb* primitive is for writing data to the blackboard.

The message body layer is the actual application-specific content of the ICL. This layer may contain a goal to solve or a command for controlling agents. It is necessary to specify the language of predicates that are used in the message body layer and are understood by all relevant agents. For example, if one agent needs to know the location of the user, it will post an expression, such as $post\text{-}query(location(user, U))$, which another agent knows how to evaluate. Here, an agreement among agents would be needed that the predicate name is "location", and its arguments are a person (input argument) and a location (result argument). Every agent participating in an OAA-based system defines and publishes a set of capability specifications, expressed in the ICL, describing the predicates that it supports. These establish a high-level interface to the agent, which is used by a facilitator in communicating with the agent, and, most importantly, in delegating service requests (or parts of requests) to the agent. Figure 3 shows an example of interactions among agents.

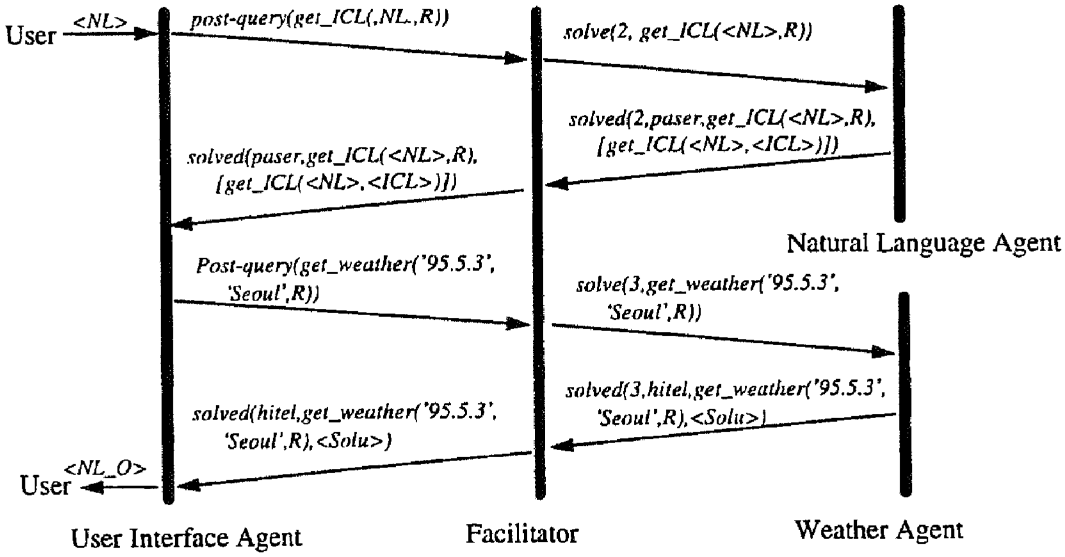
2.3.3. *Trigger*

When an asynchronous problem solving is needed, each agent can install triggers either locally for itself, or remotely, on either the facilitator or on another agent. There are currently four types of triggers.

- Event triggers: Any incoming or outgoing event (message) may be monitored.
- Data triggers: Global data triggers monitor the state of the global information written to the facilitator. These are always installed on the facilitator, as the

facilitator monitors all global data for processes. Individual database agents are expected to provide their own trigger mechanisms for monitoring the state of their local databases, patterned after the data trigger functionality provided by the facilitator.

- Test triggers: Test triggers are invoked after some front-end processing of each incoming event, and also whenever a time-out occurs in the event polling. Test triggers are useful for examining internal events that do not come through the facilitator. For example, a mail agent might watch for new incoming mail, or an airline database agent may monitor which flights will arrive later than scheduled.
- Time triggers: Time triggers monitor time conditions. These can keep track of a single fixed point in time (e.g., On December 23rd at 3 p.m.), or can handle periodic monitoring (e.g., Every three minutes from now until noon).


User
<NL>
post-query(get_ICL(,NL.,R))
solve(2, get_ICL(<NL>,R))
solved(2,paser,get_ICL(<NL>,R),
[get_ICL(<NL>,<ICL>)])
solved(paser,get_ICL(<NL>,R),
[get_ICL(<NL>,<ICL>)])
Natural Language Agent
Post-query(get_weather('95.5.3',
'Seoul',R))
solve(3,get_weather('95.5.3',
'Seoul',R))
solved(3,hitel,get_weather('95.5.3',
'Seoul',R),<Solu>)
solved(hitel,get_weather('95.5.3',
'Seoul',R),<Solu>)
<NL_O>
User
User Interface Agent
Facilitator
Weather Agent

<NL> = "get the weather in Seoul on May 3 1995"
<ICL> = get_weather('95/5/3', 'Seoul', SI)
<Solu> = get_weather('95/5/3', 'Seoul', fine)
<NL_O> = "The weather is fine"

Fig. 3. An example of interactions among agents.

3. Multi-modal User Interface

The design and development of the OAA environment have focused on providing access to agent-based applications through an intelligent, cooperative, distributed, and multi-modal agent-based user interface. The current multi-modal interface supports a mix of spoken language, handwriting and gesture, and is adaptable to the user's preferences, the available resources, and the environment. Figure 4 shows some agents related to the multi-modal user interface.

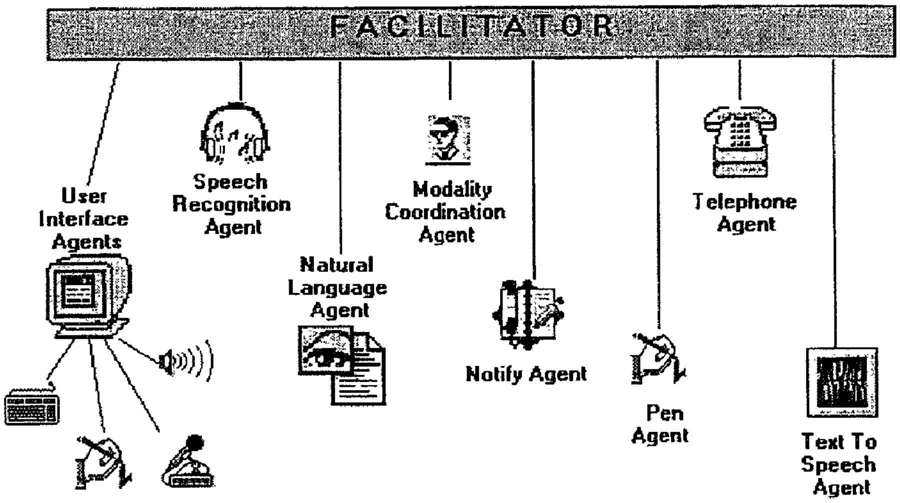


Fig. 4. Agents related with multi-modal user interface.

One of the major advantages of our agent-based user interface approach is that it enables a Personal Digital Assistants (PDA) or other low-powered computer (e.g., a portable PC) to have a user interface that incorporates substantial amounts of intelligence. Only the low-level user interface agents need to be running on the user's personal computer and all the other agents can run on remote computers. Thus, our current systems running on PCs or PDAs make use of speech recognizers, natural language systems, and other systems that require powerful data center servers.

The multi-modal user interface consists of a collection of sub-agents which are implemented within the OAA. These sub-agents can be easily replaced by new versions and adapted to a new application domain. This multi-modal user interface of the OAA is described in detail in [4].

3.1. *The user interface agent*

The user interface is implemented as a set of agents that have at their logical center an agent called the User Interface (UI) agent. The UI agent manages various modalities and applies additional interpretation to those inputs as needed. Our current system supports speech, handwriting, and pen-based gestures in addition to the conventional keyboard and mouse inputs. When speech input is detected, the UI agent sends a command to the Speech Recognition agent to process the audio input and to return the corresponding text. Three modes are supported for speech input: open microphone, push-to-talk, and click-to-start-talking. Spoken and handwritten inputs can either be treated as raw text or be interpreted by a natural language understanding agent.

There are two basic *styles* of user interface. The first style parallels the traditional graphical user interface (GUI) for an application offered by an agent: the user selects an application and is presented with a window that has been designed for the application and is composed of the familiar GUI-style items. In this style interface, each application is typically implemented as a primary agent, with which the user interacts, and a number of supporting agents that are used by the primary agent but hidden from the user. When text entry is needed, the user may use handwriting or speech instead of the keyboard, and the pen may be used as an alternative to the mouse. Because the UI agent handles all the modalities, the applications are isolated from the details of which modalities are being used. This simplifies the design of the applications and simplifies adding new modalities.

In the second basic style of user interface, not only is there no primary agent for each application but also individual application agents are largely invisible to the user unlike in the graphic window style case where the user is aware of primary application agents. The user's requests may involve cooperative actions of multiple agents. In our OAA environment implemented, this interface is based on natural language (for example, English), and the user's input is entered in either speech or handwriting. When the UI agent detects speech or pen-based input, it invokes a speech recognition agent or a handwriting recognition agent, and sends the text returned by that agent to a natural language understanding agent, which produces a *logical form* representation of the user's request. This logical form is then passed to a facilitator agent, which identifies the subtasks and delegates them to the appropriate application agents. For example, in our Map-based Tourist Information application for the city of San Francisco, the user can ask for the distance between a hotel and a sightseeing destination. The locations of the two places are in different databases, which are managed by different agents, and the distance calculation is performed by yet another agent.

These two basic styles of user interface can be combined in a single interface. In our Office Assistant application, the user is presented with a user interface based on the Rooms metaphor and is able to access conventional applications such as e-mail, calendar, and databases in the familiar manner. In addition, there is a subwindow for spoken or written natural language commands that can involve multiple agents.

3.2. *Multi-modal input*

A major focus in developing the UI agent was on handling multi-modal inputs, typically a mix of gesture/pointing with spoken or handwritten texts. The UI agent manages the interpretation of the individual modalities and passes the results to a Modality Coordination agent, which then produces the composite query and passes it to the facilitator agent for delegation to the appropriate application agents.

3.2.1. *Pen input*

Including a pen in the user interface has several significant advantages. First, the gestures that users employ with a pen-based system are substantially richer than those employed by other pointing and tracking devices (e.g., a mouse). Second, handwriting is an important adjunct to speech input. Speech recognizers (including humans) can have problems with unfamiliar words (e.g., new names). Users can use the pen to correct misspelled words, or may even anticipate the problem and switch from speaking to handwriting. Third, our experiences have shown that when a person who has been using a speech-and-gesture interface faces an environment where speech is inappropriate, switching to handwriting occurs naturally.

The gestures-recognition engine used in our system is fully described in [1] as the early recognition process. In most applications, this engine shares pen data with a handwriting recognizer. The UI agent in our OAA environment can work with any handwriting recognizer compatible with Microsoft's Pen Windows.

3.2.2. *Speech recognition*

We have used several speech recognition systems in order to meet different criteria. A research system — the DECIPHER speech recognition system [18] — developed by another laboratory in SRI International and by a commercial spin-off from that laboratory, has been used. We are currently evaluating other speech recognizers (Korean, Japanese and French speech recognition system), and plan to create agents wrapping their application programming interfaces (APIs).

3.2.3. *Natural language understanding*

In developing our OAA environment, we have used three different natural language (NL) systems: a simple one, based on Prolog DCG (Definite Clause Grammar), then an intermediate one based on CHAT [19], and finally, our most capable research system GEMINI [20]. The ability to easily substitute one natural language agent for another has been very useful in rapid prototyping of systems. The DCG-based agent is used during the early stages of development because grammars can be easily written and modified. Writing grammars for the more sophisticated NL agents requires bigger efforts, but it provides better coverage of the language that real users are likely to use. It is thus often cost-effective to delay upgrading to the more sophisticated NL agents until the application crosses certain thresholds of maturity and usage.

One of the well-known problems with systems that utilize spoken natural languages is in communicating to the user what can and cannot be said. A good solution to this is an open research problem. Our approach has been to use the design of the GUI to help illustrate what can be said: all simple operations can be invoked through traditional GUI items, such as menus, that cover much of the vocabulary. Each time these GUI actions are selected, corresponding natural language queries are displayed and the user can learn what kind of queries are appropriate in each situation.

3.3. *Linguistic Expertise Acquisition Program* (*LEAP*)

An agent system that provides an intelligent user interface — allowing users to express their requests by using spoken and hand-written natural languages in combination with other modalities — raises additional challenges regarding the establishment of a development environment. For example, one important question is how best to provide support for the agent developer, who is not likely to be a computational linguist, in tailoring the linguistic processing components of the system to handle the domain- specific expressions expected to appear in user's requests. An exploration of these issues has resulted in the creation of the Linguistic Expertise Acquisition Program (LEAP). LEAP is one part of the Agent Development Toolkit (ADT) and other parts will be described in Sec. 4.

LEAP facilitates the task of interfacing a new agent with existing linguistic support agents such as natural language parsers and speech recognition systems. This involves obtaining semantic information about the domain in which the agent operates, the services provided by the agent, and the English words that will be useful in composing requests for these services. To make these words useful to the system, LEAP obtains from the agent developer information about their linguistic attributes: it does so by asking the developer simple questions about how and when those words are used. Once the linguistic knowledge has been acquired, LEAP generates or updates the appropriate knowledge bases needed by the linguistic support agents. Details on LEAP are referred to [2].

3.4. *Modality coordination agent*

The Modality Coordination (MC) agent is responsible for combining the inputs expressed in different modalities to produce a single meaning that matches the user's intention. It is responsible for resolving references, for filling in missing information for an incoming request, and for resolving ambiguities by using contexts, equivalence, or redundancy.

Taking contexts into account implies establishing a hierarchy of rules among them. The importance of each context and the hierarchy may vary during a single session. In our OAA environment, missing information is extracted from the dialogue context.

When a user says "*Show me the photo of this hotel*" and simultaneously points with the pen to a hotel, the MC agent resolves the reference to the hotel based on that gesture. This is a scenario supported by the agent system to be described in Sec. 4.2. If no hotel is explicitly indicated, the MC agent searches the conversation context for an appropriate reference (for example, the hotel may have been selected by a gesture in the previous command). If there is no selected hotel in the current context, the MC agent will wait for a certain amount of time (currently 2 to 3 seconds) before asking the user to identify the hotel intended. This short delay is designed to accommodate various synchronization delays between speech and gesture: different users (or a single user in different circumstances) may point before, during, or just after speaking.

In another example, the user says "*Show me the distance from the hotel to here*" while pointing at a destination. The previous queries have resulted in a single hotel being focused upon, and the MC agent resolves "*the hotel*" from this context.

The gesture provides the MC agent with the reference of "*here*". Processing the resulting query may involve multiple agents, for example, the locations of hotels and sightseeing destinations may be in different databases, and these locations may be expressed in different formats, requiring another agent to resolve the differences and then compute the distance.

3.5. *Multi-modal output*

Some elementary planning and reasoning capabilities have been used in our OAA environment for selecting the appropriate output modality. For example, a user can request to be notified when a specified event occurs. The Notify agent has a set of rules for making the notification. If the user is not using his/her computer at that time, the agent will try to give him the message by telephone. This notification-by-telephone task is also specified by a set of rules. First, the agent checks the user's calendar (via the calendar agent) to attempt to determine where he currently is. If it finds an entry with an identifiable location, it queries the telephone book (another agent) to get the phone number. It then has the telephone agent make the call. We use a PIN (personal identification number) to determine that the intended person is the one answering the call, but an automatic speaker-verification technology may be used in future systems. If the attempt to notify by telephone fails (no one answers, or the user is not available), the Notify agent falls back to the next-best scheme given by the rules (e.g., call his pager, send e-mail, or call his secretary).

4. Experimental Prototype Implementations

Two applications, the *Office Assistant* and the *Map-based Tourist Information*, have been the primary experimental prototypes implemented so far. The agent architecture and the specific agents developed in these application systems have proven to be so useful that they are being used in an expanding set of other projects within our organizations.

Some of the projects adopting the OAA have been motivated by the availability of various agents, especially the user interface agents. Some projects went further before adopting the OAA and started using the OAA to integrate the major software components being developed in those projects.

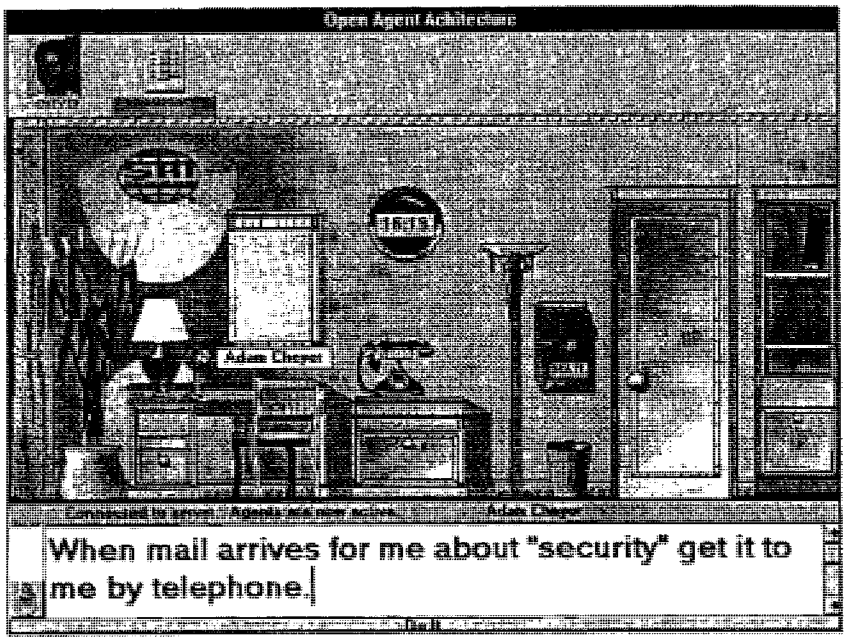
As an initial set of tools for assisting the creation of agents, the Agent Development Toolkit (ADT) has been established. These tools guide the developer through the agent creation process and automatically generate code templates from specifications (in the style of various commercial CASE tools). These tools are implemented as OAA agents, so they can interact with, and build upon, existing agents. The common agent support routines have been packaged as libraries, adapted to various languages supported.

These tools support both creation of entirely new agents and generation of agents from existing applications, including legacy systems. These latter agents are called *wrappers* (or transducers); they convert between ICL and the application's interface.

4.1. *Office assistant*

The OAA has been used as the framework for a number of applications spanning over several domain areas. In the first OAA-based system, the "office assistant", fourteen autonomous agents provide information retrieval and communication services for a group of co-workers in a networked computing environment. This system makes use of a multi- modal user interface running on a pen-enabled portable PC, and allows for the use of a telephone to give spoken commands to the system. Services are provided by agents running on UNIX workstations, many of which were created by providing agent wrappers for legacy applications. This system is described in detail in [14].

In a typical scenario, agents with expertise in e-mail processing, text-to-speech translation, notification planning, calendar and database access, and telephone control cooperate to find a user and to alert him or her of an important message. The office assistant system provides a demonstration of how new services can arise from the synergistic combination of the capabilities of components that were originally intended to operate in isolation. In addition, it demonstrates the combination of two basic styles of user interaction — one that directly involves a particular agent as the primary point of contact, and one that anonymously delegates requests across a collection of agents — in a way that allows the user to switch freely between the two.



Connected to server. Agents are now active. Adam Cheyer

When mail arrives for me about "security" get it to me by telephone.

Fig. 5. The interface window of Office Assistant.

In the interface for this system (Fig. 5), the initial screen portrays an office, in which familiar objects are associated with appropriate functionalities, as provided by some agents. For instance, clicking on a wall clock brings up a dialogue that allows one to interact with the calendar agent (that is, browsing and editing one's appointments). In this style of interaction, even though the calendar agent may call on other agents in response to some requests, it has the primary responsibility in that all requests through that dialogue are handled by it.

The alternative style of user interaction is one in which the user might speak "*When mail arrives for me about 'security', get it to me by telephone.*". In this case, the delegation of the request to appropriate agents — which is done by the User Interface agent in concert with a facilitator agent — reflects a style that is less direct and more anonymous.

4.2. *Map-based tourist information*

In a number of domains, access to information can very naturally be organized around a map-based interface. In creating such interfaces for several different systems, we have found again the agent-based approach to multi-modality to be highly useful. In these systems, all the components share a common interface — the map — and the fact that there are many agents is entirely invisible to the user.

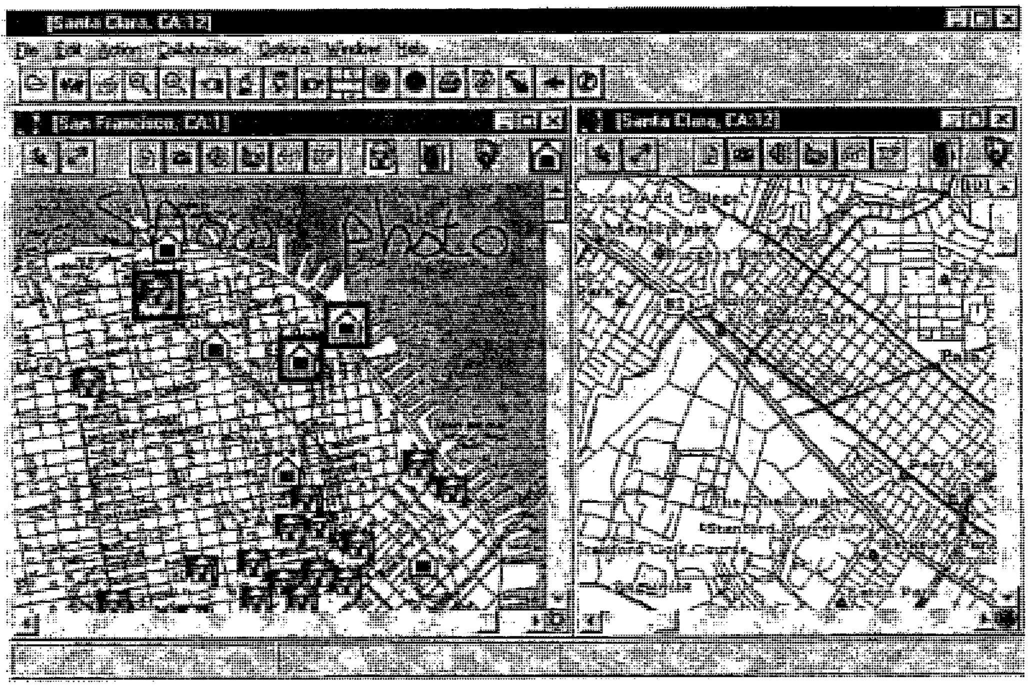


Fig. 6. The interface window of Map-based Tourist Information.

One example is a map-based system to provide tourist information about San Francisco. Requests expressed in a variety of modalities can be made to control the scrolling and zoom level of the map, to retrieve information about locations and distances, to display hotels or attractions meeting a user's preferences, or to present detailed information in a variety of media about particular hotels or attractions. Where appropriate, this information is derived and updated regularly from WWW sources. Figure 6 shows the interface of this system.

Map-based interfaces provide a rich setting in which to explore the coordination of gesture with speech and traditional GUI modalities. The tourist information system accommodates the use of a variety of familiar pen gestures, such as circling objects or regions, drawing arrows, X'ing positions or objects, and striking out objects. Depending on context and timing considerations, requests can be derived from single gestures, multiple gestures interpreted together, spoken or handwritten input, point-and-click, or some combination of these operations.

For example, an arrow drawn across a map from right to left (which itself is recognized from two or three pen strokes) is interpreted as a request to scroll the map. The same effect may be achieved by speaking "*scroll left*". Display of hotels can be obtained by writing or speaking "*Show hotels*", or, perhaps, "*Show hotels with a pool*". The distance between two objects or locations may be obtained by circling, X'ing, or clicking on each of them, and then drawing a straight line between them. Alternatively, one can speak "Show the distance from here to here", while selecting two locations, or one can write "*distance*" either before or after selecting two objects. Details on this system and the organization of the input recognition agents are referred to [1].

4.3. *Implementation in distributed environments*

4.3.1. *Multiple platforms*

The OAA applications that we have implemented run on a variety of platforms, and the exact locations of individual agents can be easily changed. We currently support PCs (Windows 3.1 and 95) and Sun and SGI workstations. Our primary user interface platform is the PC, partly because it currently offers about the strongest support for pen- based computing and partly because of our emphasis on providing user interfaces on lightweight computers (portable PCs and PDAs in near future). PCs also have the advantage of mass-market GUI-building packages such as Visual Basic and Delphi. A smaller-functionality version of the user interface has been implemented under X for UNIX workstations.

Even when the UI is on a PC, some of the agents in the UI package may be running elsewhere. Our preferred speech recognizer requires a UNIX workstation, and our natural language agents and Modality Coordination agent have been written for UNIX systems.

4.3.2. *Mobile computing*

We view mobile computing not only as computing by people moving around with portable computers using wireless communication, but also computing by people moving *between* computers. Today's user may have a workstation in his office, a personal computer at home, and a portable or PDA for meetings. In addition, when the user meets different people in different locations, e.g., managers, colleagues and customers, the accessible computers may be different platforms. From each of these machine environments, the user should be able to access his/her data and run his/her applications.

The OAA facilitates using multiple platforms because only the primary user interface agents need to be running on the local computer, thereby simplifying the problem of porting to new platforms and modality devices. Also, since only a minimal set of agents need to be run locally, lightweight computers (portable PCs, PDAs, and older systems) have the resources needed to utilize heavyweight, resource-hungry applications running in remote sites.

5. Conclusion

The OAA has proven to be useful in constructing sophisticated systems because it provides a sound and flexible facility for combining applications that were not originally envisioned as a package. The OAA approach differs from much of the other approaches for constructing distributed agents in its focus on providing multi-modal user interfaces to systems assembled from disparate agents.

Expanding our ICL to cover more complex tasks such as Contract-net protocols is considered to be a highly worthwhile topic for future research.

Acknowledgments

This paper is based on the work done in a joint-project of Electronics and Telecommunications Research Institute (Korea) and SRI International (USA). Our special appreciation goes to the members of this project, Douglas Moran who is the project leader of the SRI team, Adam Cheyer, David Martin, and Luc E. Julia in SRI International, and Gowang-Lo Lee and Myeong-Wuk Jang in ETRI for their contributions in development and testing of the OAA, the ADK, the multi-modal user interface system, and many application agent systems. The work reported here greatly benefited from the inspiring leadership of Phil Cohen who was the project leader until August 1994 (now at the Oregon Graduate Institute). The work by Kane Kim was supported in part by the US Defense Advanced Research Project Agency under Contract N66001-97-C-8516 monitored by NRaD and in part by the University of California's MICRO Program under Grant 96-169.

References

1. A. Cheyer and L. Julia, " Multi-modal maps: An agent-based approach", in *Proc. of the International Conference on Cooperative Multi-modal Communication*, Eindhoven, The Netherlands, May 1995, pp. 24–26.
2. D. L. Martin, A. Cheyer, and G.-L. Lee, "Agent development tools for OAA", in *Proc. of the First International Conference on the Practical Applications of Intelligent Agents and Multi-Agent Technology*, London, April 1996, pp. 387–404.
3. G. Schwartz, "Cooperating heterogeneous systems: A blackboard-based meta approach", *Technical Report 93-112*, Center for Automation and Intelligent Systems

Research, Case Western Reserve University, Cleveland, Ohio, April 1993. Unpublished PhD thesis.

4. D. B. Moran, A. Cheyer, L. Julia, and S. Park, "Multi-modal user interface in the open agent architecture", in *Proc. of Intelligent User Interface '97* (*accepted paper*), Orlando, Florida, Jan. 1997.
5. H. A. Kautz, B. Selman, and M. Coen, "Bottom-up design of software agents", *Communications of ACM* **37(7)**, (1994) 143–146.
6. M. R. Genesereth and N. P. Singh, "A knowledge sharing approach to software interoperation", Computer Science Department, Stanford University, unpublished ms, 1994.
7. General Magic, Inc., *Telescript Product Documentation*, 1995.
8. M.-W. Jang, J. Choi, S. Park, S. C. Baeg, G.-L. Lee, "Controlling agent interactions in a tightly coupled multi-agent framework, in *Proc. of International Conference on Computer Communication'95*, Seoul, Korea, Aug. 1995, pp. 41–46.
9. T. Wittig, N. R. Jennings and E. H. Mamdani, "ARCHON — A framework for intelligent co-operation", in *IEE–BCS Journal of Intelligent Systems Engineering — Special Issue on Real-time Intelligent Systems in ESPRIT* **3(3)** (1994) 168–179.
10. R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. Swartout, "Enabling technology for knowledge sharing", *AI Magazine* **12(3)** (1991) 36–56.
11. Object Management Group, "The common object request broker: Architecture and specification: Revision 2.0", *OMG Document*, July 1995.
12. Etzioni, N. Lesh, and R. Segal, "Building softbots for UNIX (preliminary report)", in *Working Notes of the AAAI Spring Symposium on Software Agents*, Menlo Park, CA, March 1994, pp. 9–16.
13. P. R. Cohen and H. J. Levesque, "Persistence, intention and commitment", *Intentions in Communication*, eds. P. R. Cohen, J. Morgan, and M. E. Pollack, MIT Press, Cambridge MA, 1990, pp. 33–69.
14. P. R. Cohen, A. Cheyer, M. Wang, and S. C. Baeg, "An open agent architecture", in *Working Nostes of the AAAI Spring Symposium on Software Agents*, Menlo Park, CA, March 1994, pp. 1–8.
15. S. Park, G.-L. Lee, J.-M. Choi, and M.-W. Jang, "An agent-based user interface system", in *Proc. of UNIEXPO'94*, Seoul, Korea, June 1994, pp. 23–27.
16. T. Finin and R. Fritzson, "KQML: A language and protocol for knowledge and information exchange", in *Proc. of the 13th International Distributed Artificial Intelligence Workshop*, 1994, pp. 127–136.
17. V. R. Lesser, R. C. Whitehair, D. D. Corkill, and J. A. Hernandez, "Goal relationships and their use in a blackboard architecture", in *Blackboard Architectures and Applications*, eds. V. Jagannathan, Rajendra Dodhiawala, and Lawrence S. Baum, Academic Press, San Diego, CA, 1989, pp. 9–26.
18. M. R. Cohen, H. Murveit, J. Bernstein, P. Price, and M. Weintraub, "The DECIPHER speech recognition system. In IEEE ICASSP, pages 77-80, 1990.
19. C. N. Pereira, *Logic for Natural Language Analysis*, PhD thesis, Univ. of Edinburgh, 1983.
20. J. Dowding, J. M. Gawron, D. Appelt, J. Bear, L. Cherny, R. Moore, and D. Moran, "GEMINI: A natural language system for spoken language understanding", in *Proc. of the 31st Annual Meeting of the Association for Computational Linguistics*, Ohio State University, Columbus, Ohio, Jun. 1993, pp. 54–61.

HUMAN-COMPUTER INTERACTION

Taylor & Francis

Human–Computer Interaction

ISSN: 0737-0024 (Print) 1532-7051 (Online) Journal homepage: http://www.tandfonline.com/loi/hhci20

A mechanism for Multimodal Presentation Planning Based on Agent Cooperation and Negotiation

Yi Han & Ingrid Zukerman

To cite this article: Yi Han & Ingrid Zukerman (1997) A mechanism for Multimodal Presentation Planning Based on Agent Cooperation and Negotiation, Human–Computer Interaction, 12:1-2, 187-226

To link to this article: https://doi.org/10.1080/07370024.1997.9667243

Published online: 22 Jun 2011.

HUMAN–COMPUTER INTERACTION, 1997, Volume 12, pp. 187–226

A Mechanism for Multimodal Presentation Planning Based on Agent Cooperation and Negotiation

Yi Han and **Ingrid Zukerman**
Monash University, Australia

ABSTRACT

A multimodal presentation planning mechanism must take into consideration the structure of the discourse and the constraints imposed by discourse relations. This requires that different processes that perform multimodal presentation planning be able to communicate with each other. In this article, we introduce a multiagent architecture based on the blackboard system that satisfies this requirement. In addition, we describe a constraint propagation mechanism that transfers plan constraints from one level of the presentation planning process to the next, and we discuss the cooperation and negotiation processes between modality-specific agents in a prototype system that implements the multiagent planning mechanism.

Yi Han is a systems analyst at Philips Public Telecommunication Systems, Philips Australia, with an interest in multimodal presentation planning; she was a PhD student at Monash University. **Ingrid Zukerman** is a computer scientist with an interest in discourse planning, multimodal presentation planning, and plan recognition; she is an Associate Professor in the Department of Computer Science at Monash University.

CONTENTS

1. INTRODUCTION

A multimodal presentation system selects automatically the modality used for different presentation components and generates a multimodal presentation that conveys the overall structure of the discourse (Arens, Hovy, & van Mulken, 1993). Different parts of a piece of discourse play different roles, such as supporting, contradicting, or contrasting with other parts. These discourse relations may pose requirements on the modalities used to present the different portions of the discourse. For instance, consider generating a multimodal presentation to convey that a sky diver controls his or her falling speed by changing body shapes. This goal can be achieved by comparing the falling speed corresponding to each body movement. To contrast these information items, the different kinds of body movements should be presented in the same modality (e.g., images) and their corresponding falling speeds in another (e.g., a chart). If a natural language is supported by the system, the reason for a sky diver being able to control the falling speed using his or her body can be presented in text or speech. This may require the system to generate a presentation in images, a chart, and spoken language simultaneously, so that the verbal explanation can be delivered at an appropriate time to link the images to the chart.

Discourse relations may restrict resources (e.g., time or space) to be used for presenting different discourse components. In the example just described, the different kinds of body movements (and corresponding falling speeds) are contrasted with each other; hence, they should be visible on the same screen. This requirement restricts the space consumption of the presentations generated for these items. Therefore, images may not be suitable due to insufficient screen space. For an online system, where the response time is often restricted, graphics may not be suitable for presenting the body movements if their generation takes more time than that available to the system, even if the space restrictions are satisfied. The time for generating presentations is also restricted when the presentations (e.g., a verbal explanation) must be delivered at a specific point in time. Thus, a previously selected modality may no longer be appropriate when the information regarding the resources required for each component becomes available at a later stage in the presentation planning process.

We represent the modality restrictions imposed on each presentation component by means of *modality constraints,* and the space restrictions imposed on each presentation component by means of *space constraints.* Constraints have two types of strength: *required* and *preferred,* representing constraints that must be satisfied and constraints that are preferably satisfied, respectively. For example, a constraint that stipulates that all the presentations of the different body shapes should be in the same modality (e.g., image) is required, whereas constraints that pertain to the layout format of the presentations of the body shapes and the falling speeds are preferred (e.g., constraints that specify the position of each body shape on the screen relative to its corresponding falling speed).

Our research is based on the tenet that multimodal presentation planning should be carried out by two different processes: the discourse planning process and the presentation planning process (Arens & Hovy, 1994), where the presentation planning process transfers the discourse structure generated by the discourse planning process into a presentation structure. In particular, we focus on four requirements of multimodal presentation planning:

1. *An independent process for each modality.* The processes used for modality-specific presentation planning are independent of each other in the sense that individual functions and algorithms are applied to generate modality-specific presentations. For example, the algorithm for calculating the scale on an axis in a chart is different from the algorithm for calculating the position of an entry in a table. Therefore, the functional modification of a modality-specific process should not affect other modality-specific processes. Maintaining an independent process for each modality also supports the satisfaction of modality constraints through the selection of modality-specific processes. In addition, it improves the extensibility of the system because new modality-specific processes can be easily incorporated into the system.
2. *Resource sharing.* The intended information is shared by alternative modality-specific processes (e.g., the text generator and the speech

generator in the preceding example). Further, if a multimodal presentation is cooperatively generated by several modality-specific processes, these processes share resources such as time and screen space.

3. *Communication between modality-specific processes.* Communication between these processes is necessary for a process to be able to modify its presentation in response to the actions of another process. For instance, if two discourse components need to be presented on the same screen, the presentation generator of one of the components must obtain the size of the presentation of the other component to be able to calculate its own size. This cooperation between modality-specific processes ensures the satisfaction of constraints placed on the presentation of each component. Negotiation between processes over space consumption takes place when alternative presentation formats may be used to generate a presentation. For example, after presenting visual components (e.g., images and a chart), the presentation planner may decide to reduce the space that was initially allocated to the text generator, thereby requiring the text generator to reformat the textual component.
4. *Simultaneous generation of different modality-specific components.* Modality-specific processes need to be executed simultaneously to present information cooperatively. Further, if cooperative input modalities, such as spoken language and hand gestures, are supported, concurrency is necessary to process the user's input via these modalities.

In this article, we present a multiagent mechanism based on a hierarchical blackboard architecture that satisfies these requirements. To satisfy the first requirement, a different agent is activated for the generation of each modality-specific component in a multimodal presentation. The constraints imposed by discourse relations and limited resources are satisfied through cooperation and negotiation between agents that share the screen space and time available for presenting the overall discourse. The hierarchical blackboard architecture enables agents to share information that is relevant to specific tasks and to communicate with each other with respect to the satisfaction of constraints pertaining to these tasks (see Section 2.1). Consequently, the second and third requirements are satisfied. A dynamic multiagent mechanism is required, rather than the static knowledge sources used in the blackboard system (Engelmore & Morgan, 1988), for the following reasons. First, the modality selection process in our mechanism needs the ability to (a) activate a modality-specific agent when a particular task is to be performed and (b) remove this agent when its job is finished or when an alternative agent has completed the job in a superior way. Second, the activation of multiple agents enables several modality-specific components to be generated simultaneously, particularly when several instances of the same component (e.g., icon) are required, which satisfies the last requirement.

Our mechanism has been implemented in a prototype system called MAGPIE (Multi-Agent Generation of Presentations In Physics Education) that generates multimodal presentations for conveying abstract concepts in high-school physics. MAGPIE uses Common Lisp Object System and Garnet (a Toolkit for graphical user interface design) in its implementation.

Section 2 of this article describes the multiagent architecture and the agent construction process in MAGPIE. Section 3 discusses blackboard events and explains how agents communicate with each other. Multimodal presentation plans and plan constraints are described in Section 4, and the propagation of these plan constraints is explained in Section 5. Section 6 describes the negotiation process between a group of agents in MAGPIE. Section 7 discusses related research comparing it with MAGPIE's approach. Finally, Section 8 presents concluding remarks and future research directions.

2. THE MULTIAGENT ARCHITECTURE

It is generally accepted that the structure of computer-generated discourse is determined by a discourse planning process. Therefore, we assume that MAGPIE receives as input a discourse plan generated by such a process. The presentation planning process of MAGPIE selects a modality to convey each discourse component, generates each modality-specific presentation, and finally displays the multimodal presentation in a window on the computer screen. Figure 1 illustrates a discourse plan received as input by MAGPIE. The rhetorical devices in this discourse plan are generated by a discourse planner such as that described by Zukerman and McConachy (1993) to convey information regarding the weight of objects. Assertions 1 and 2 state that the weight of an object is a force, and its magnitude depends on the mass of the object and the acceleration due to gravity. Comparison 1 states that the acceleration due to gravity is different on Earth and on the moon. Comparison 2 states that the weight of an object on Earth differs from its weight on the moon. Finally, Instantiations 1 and 2 illustrate Comparison 2 with respect to two objects: a box and a computer. MAGPIE may produce different multimodal presentations according to the screen space available in the display window and the time restrictions of the presentation. The multimodal presentation in Figure 2 contains a piece of text presenting the assertions and comparisons. The instantiations are presented in a chart that shows the difference in the weight of the two objects between Earth and the moon.[1] In Figure 3, the

[1] The dark bars in the charts in this article appear in red on the screen; the light bars appear in green.

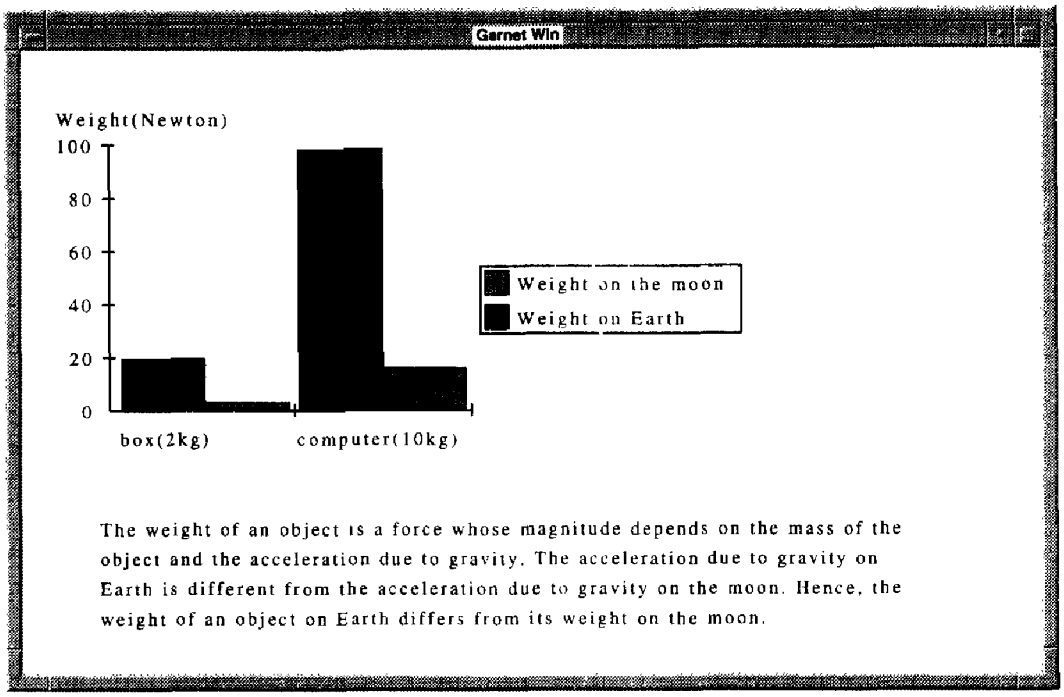
Figure 1. **Discourse structure that conveys the weight of objects.**

Assert [weight(object) isa force]
Assert [magnitude(weight(object)) $\propto$ mass(object) & gravity-acceleration]

Compare [gravity-acceleration(Earth) $\neq$ gravity-acceleration(moon)]
Compare [weight(object, Earth) $\neq$ weight(object, moon)]

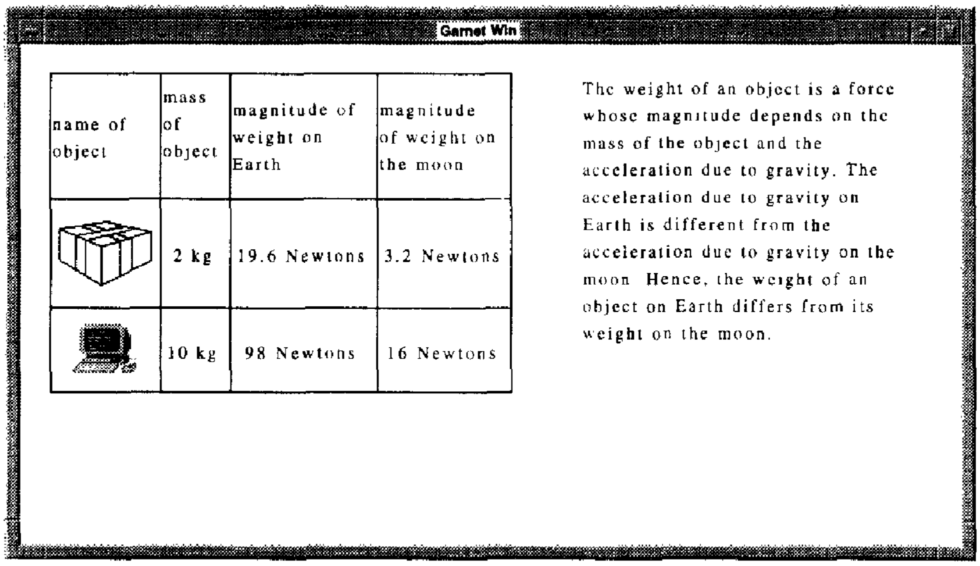
Instantiate
    Inst_1 [weight(box, location)] (Earth moon)
    Inst_2 [weight(computer, location)] (Earth moon)

Figure 2. **Sample multimodal presentation: chart and text.**



The weight of an object is a force whose magnitude depends on the mass of the object and the acceleration due to gravity. The acceleration due to gravity on Earth is different from the acceleration due to gravity on the moon. Hence, the weight of an object on Earth differs from its weight on the moon.

Figure 3. **Sample multimodal presentation: table and text.**



name of object	mass of object	magnitude of weight on Earth	magnitude of weight on the moon
	2 kg	19.6 Newtons	3.2 Newtons
	10 kg	98 Newtons	16 Newtons

The weight of an object is a force whose magnitude depends on the mass of the object and the acceleration due to gravity. The acceleration due to gravity on Earth is different from the acceleration due to gravity on the moon. Hence, the weight of an object on Earth differs from its weight on the moon.

intended information is conveyed by the same text and a table, in which icons are selected from the icon library in MAGPIE to present the objects, numbers are used to present the mass of the objects and their weight, and text strings are used to present table headings and units of mass and weight. MAGPIE prefers to present the text next to the table because our eyes naturally move from left to right (Holmes, 1984). The table in Figure 3 occupies less screen space than the chart in Figure 2 but takes a longer time to generate. If the system has restricted screen space for conveying the intended information, Figure 3 is displayed; if the system is restricted on time, Figure 2 is displayed.

Each component of these multimodal presentations is generated by a modality-specific process called an agent, and all the components are dynamically integrated by the presentation planning agent, which takes care of the overall discourse structure. For example, MAGPIE has an icon agent, a number agent, and a text agent, to produce icons, numbers, and text, respectively. These agents cooperate with the table agent to generate the display in Figure 3. In contrast, the table agent and the chart agent compete with each other for the presentation of the instantiations. In this section, we focus on the multiagent architecture of MAGPIE. The generation of presentation plans and the communication between agents are described in subsequent sections.

2.1. Basic Functions of Blackboards and Agents

Our multiagent architecture consists of two types of components: *agents* and *blackboards*. An agent is an independent process that can be created or removed dynamically on demand. Given intended information, an agent designs a presentation plan to convey this information. If the intended information is composite, the agent decomposes the intended information into several components and determines the information to be presented in each component. To ensure that these components can be integrated into a display format (e.g., a table or a chart), the agent must create constraints to restrict the possible modalities and space used by each component. If the intended information is atomic (e.g., a number), the agent determines the parameters for generating the presentation to convey this information (e.g., the font of the number), and ensures the satisfaction of constraints pertaining to the modality-specific presentation.

A blackboard is a set of facilities for agents to share information and to communicate with each other. It manages the registration of a group of agents and the presentation plans and blackboard events generated by these agents. It provides communication primitives for the group of agents to send or receive blackboard events via different protocols (see Section 3). In addition, it maintains propagation paths for constraints that are imposed on the components generated by these agents, and updates these paths dynamically each time a constraint is added or removed (see Section 5.1). These paths enable an agent to obtain the space requirements on its component from the space usage of the other agents by performing constraint propagation. The management of the propagation paths is necessarily done by an intermediate layer (i.e., the blackboard) because several agents may evaluate simultaneously the constraints that restrict their variables, and the local propagation of the values of these variables may yield unpredictable results (see Section 5.1).

Using blackboards supports communication and resource sharing between agents that cooperate on the presentation of composite information, and the selection process among agents that compete for the presentation of a piece of information:

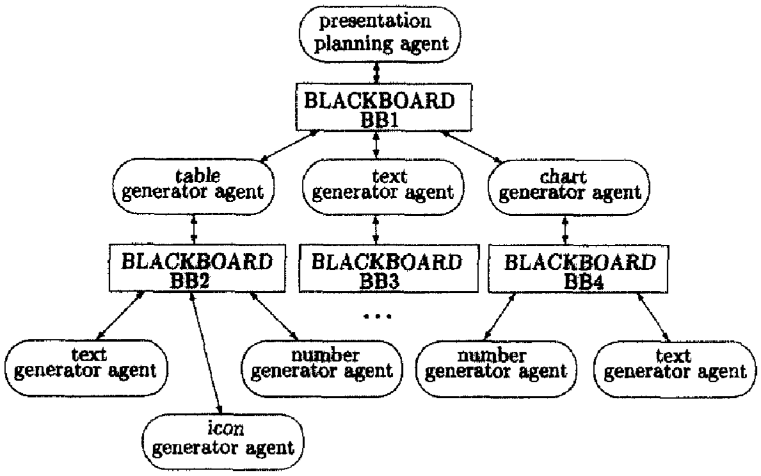
- *Supporting communication and resource sharing between cooperative agents.* This is done by combining blackboard events with constraint propagation. Blackboard events are used by agents to send a request or a response to a request. Constraint propagation is performed to enable agents to obtain requirements regarding the values of their variables, which in turn enables them to handle blackboard events during their planning process. For example, if two agents generate different components to be displayed on the screen, and one agent needs more space for its presentation than the space initially allocated to it, this agent may demand additional space. However, if this demand results in the reduction of the space allocated to the other agent's presentation, it is more efficient for this other agent to know how large its presentation can be—and then reduce its presentation accordingly—than to randomly select a size for reduction. Constraint propagation enables the second agent to obtain requirements on the size of its presentation from the size of the presentation generated by the first agent. This problem cannot be addressed by one agent simply telling another the value of its relevant variable (e.g., its size) because without constraint propagation this other agent cannot know whether or how the value of this variable affects its own variables.
- *Supporting the selection process among competing agents.* Each of the competing agents obtains space requirements on its presentation from constraints maintained by the blackboard and uses the blackboard to report on its ability to satisfy these constraints. This helps the modality selection process take into account the available space, which is a dynamic factor in our application domain.

A hierarchical blackboard architecture is used in MAGPIE, in which agents are dynamically organized into hierarchical groups during the presentation planning process on the basis of the task decomposition. That is, an agent may employ other autonomous agents to do the required subtasks. The agent that hires other agents is called the ***master agent,*** whereas agents that work for the master agent are called ***server agents.*** The master agent and its server agents form a group. A blackboard is bound to each group of agents to handle the information sharing and communication between them. This blackboard is called a ***local blackboard*** and it is owned by the master agent.

A hierarchical blackboard architecture was selected because it represents explicitly the discourse structure. In addition, a modality-specific presentation in MAGPIE may consist of several modality-specific components. For example, a table may contain icons as its entries. Thus, the modification of modality-specific components (e.g., the icons in a table) must be agreed on by the agent that integrates these components (e.g., the agent generating the table). In terms of the communication among agents, this means that an agent may receive requests to perform modifications to its presentation both from the agent that activated it and from the agents it activates. Such a mode of operation is catered for by a hierarchical architecture, in which an agent communicates with its master and its servers. As a result, the communication among agents is simplified. Finally, constraints in MAGPIE are imposed by each agent that is responsible for integrating components generated by other agents. The hierarchical blackboard architecture divides these constraints into groups, so that the effect of constraint propagation remains localized inside each group. The propagation of the constraints in each group is supported by a local blackboard, which stores and manages the propagation paths between variables controlled by the agents in the group. This supports the independent behavior of agents in the sense that they do not require information about other agents' variables. The addition or removal of agents (which may occur during the presentation planning process) may lead to the addition or deletion of constraints, which in turn may result in the definition of new variables or the elimination of existing variables. However, this does not affect an agent's ability to propagate the constraints pertaining to its variables because the propagation paths are dynamically updated by the local blackboard.

Figure 4 illustrates the agent construction process for generating the presentations in Figures 2 and 3. The presentation planning agent invokes a text agent to convey the assertions and comparisons in Figure 1, and a table agent and a chart agent to convey the instantiations (the group under Blackboard BB1 in Figure 4). When the presentation task is decomposed further, the table agent and the chart agent hire other agents to perform subtasks—for example, the table agent hires server agents to present the entries in the table (left branch of Figure 4). In this particular example, the agent group headed by the table agent includes the number agent (to present the mass of objects and their weight), the icon agent (to present the

Figure 4. **Agent construction in a multiagent planning mechanism.**



objects), and the text agent (to present the table headings and the units of mass and weight). An instance of an agent is created for each subtask. Thus, there are two instances of the icon agent, one for presenting a box and another for presenting a computer; and there are six instances of the number agent and the text agent (three instances for each object: one for presenting its mass, another for its weight on Earth, and a third for its weight on the moon). Four additional instances of the text agent are used for presenting the table headings. Thus, the icon agent, number agent, and text agent collaborate on the presentation of the entire table. Similarly, the table agent and the text agent under Blackboard BB1 in Figure 4 collaborate on the presentation of the entire discourse structure because the text agent presents the assertions and comparisons and the table agent presents the instantiations. At the same time, the table agent and the chart agent compete for the presentation of the instantiations.

2.2. Modality Selection

The focus of this research is on the system architecture, so at present MAGPIE uses very simple heuristics for modality selection. The modalities capable of presenting a particular information item in the input are currently hard-coded as a tag attached to this information item. For example, assertions and comparisons are tagged with text as the only modality capable of presenting them. However, MAGPIE was designed with a view to applying modality selection rules such as those described by Arens, Hovy, and Vossers (1993) to select suitable presentation modalities. Therefore, in addition to the structure of the discourse and the information to be presented, MAGPIE receives as input the following attributes, which are necessary for the application of these rules.

Figure 5. **Refinement of the sample instantiations.**

dimension: 2D
dimensional-focus: ((*object* name mass) (*weight* magnitude))
discourse-relation: (Compare *weight*(object, Earth) *weight*(object, moon))

Instantiation$_1$:	**Instantiation$_2$:**
object:	*object:*
name: box	*name:* computer
mass:	*mass:*
value: 2	*value:* 10
unit: kilogram	*unit:* kilogram
weight:	*weight:*
location: Earth	*location:* Earth
magnitude: 19.6	*magnitude:* 98
unit: Newton	*unit:* Newton
weight:	*weight:*
location: moon	*location:* moon
magnitude: 3.2	*magnitude:* 16
unit: Newton	*unit:* Newton
importance: t	*importance:* t

1. The dimension of the information set (e.g., one- or two-dimensional).
2. The dimensional focus of the information set, which indicates the elements of the information that should be presented along each dimensional axis.
3. The importance of the items in the information set (*t* or *nil*).
4. The discourse relations between the information items (e.g., contradiction or comparison).

These attributes can be generated by a discourse planner, but at present they are hand-coded. Figure 5 contains a refinement of Instantiations 1 and 2 in Figure 1, where these attributes have been filled in as follows:

1. The dimension of the information set is two-dimensional (object and weight).
2. The dimensional focus specifies that the name and the mass of objects are the focus of the object dimension, and that the magnitude of the weight is the focus of the weight dimension.
3. Both instantiations are considered important for the presentation (importance is *t*).
4. The discourse relation between the information items is comparison.

In addition to these attributes, information characteristics of the individual concepts to be presented, such as dimension, continuity (discrete or continuous), and information type (number, description, or name), are added to the information to be presented in order to be able to render these concepts (Arens, Hovy, & Vossers, 1993). These characteristics are obtained from a knowledge base where the individual concepts are stored.

In addition to the attributes provided in the input, a perceiver's preferences must be considered during modality selection. Currently MAGPIE has a very simple user profile that contains the perceiver's age and literacy level.[2] It uses heuristics to select a modality from a list of alternative modalities based on this profile. For example, MAGPIE uses nontextual modalities (e.g., icons) to present objects to perceivers with a low level of literacy. This is illustrated in Figure 3, in which icon rather than text is selected to present the objects in the table. The chart agent does not allow icons to be used as labels on the axes, so text is used in Figure 2 even though icons are preferred by the perceiver. If the user's preferences are not sufficient for selecting a modality, agents compete with each other for the presentation of the intended information. Considerations based on resource restrictions and on the resource consumption of the competing agents are then applied to select a presentation among those generated by these agents (see Section 5.2). For example, the instantiations in Figure 5 can be conveyed by a table, a chart, or text. MAGPIE determines from the user's profile that visual modalities, such as a table or a chart, are preferred by the user. However, the system cannot determine from this profile whether the user prefers a table or a chart; therefore, an agent is created for each of these modalities. The agent that generates a presentation suitable for the space available on the screen in the shortest time will be selected to present the instantiations.

As seen from this example, one of the features of our architecture is that it does not coerce the (possibly unmotivated) selection of a single modality for presenting a given piece of information. Rather, it allows several potentially suitable presentation agents to work in parallel on the presentation of the intended information, and it eventually selects a particular modality on the basis of its resource consumption and restrictions imposed by previous plans. In MAGPIE, restrictions on space consumption and restrictions imposed by previous plans are represented by means of constraints (see Sections 4 and 5). Currently, time restrictions are considered by the presentation planning agent through the application of a selection heuristic whereby the presentation generated by the agent that finishes first is displayed. In the future, we intend to represent time restrictions for presentation planning as constraints. A consequence of the propagation of time constraints is that the presentation planning agent is able to assign time sessions to the agents that cooperate with each other on the generation of a multimodal presentation (e.g., the presentation planning agent may schedule the presentation corresponding to the instantiations to be displayed 5 sec after the assertions and comparisons).

[2] Information regarding the age of a user supports the indirect inference of a literacy level, if direct evidence for the literacy level is not available.

In MAGPIE, the decision to hire an agent to present a component depends not only on the capability of this agent but also on the role played by the component generated by this agent in the whole presentation. The requirements placed by this role on the presentation are expressed by means of modality constraints (see Section 4.2). For instance, the table agent activates a text agent to generate a column heading because text is a modality capable of presenting the information in the heading, and it also satisfies the modality constraint pertaining to table headings, which stipulates that the column headings of a table must be textual. When new modalities are added to MAGPIE, more candidates may be available for presenting a piece of information. These candidates must be incorporated into the modality constraints. For example, a table entry may be presented by means of a vector, in addition to the currently available modalities (viz., icon, numbers, and text). Thus, adding new modalities to the system is relatively easy.

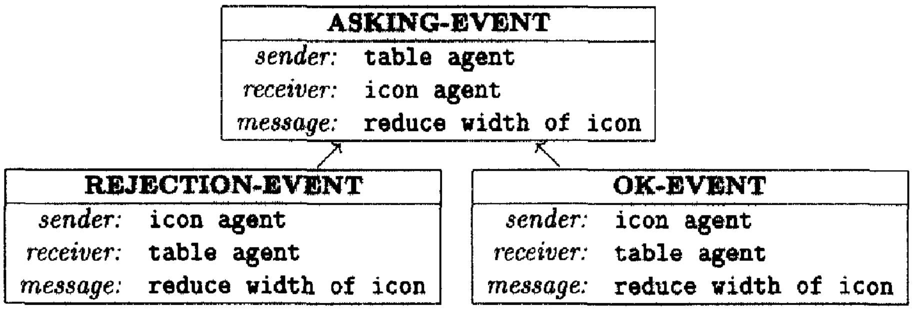
3. BLACKBOARD EVENTS

Agents share partial presentation plans on a blackboard and communicate with each other through blackboard events. Each agent has a set of event handlers that determine its reaction to different blackboard events. As stated previously, a local blackboard is bound to each agent group formed during the task decomposition process. Agents within a group read from the local blackboard plan requirements propagated from the previous level in the plan hierarchy and partial plans generated by other agents in the same group and then generate their own partial plans that satisfy these requirements.

Agents communicate with each other in different ways under different situations. For example, an agent may talk to one agent or a group of agents. In addition, if an agent has an urgent message for another agent, it may want to interrupt this other agent's process. Finin, Fritzson, McKay, and McEntire (1994) described an agent communication language containing a set of performatives for agents to use in the communication process. Each of these performatives identifies the protocol to be used to deliver a message and a speech act to be used by the sender to describe the content of a message. In contrast, MAGPIE organizes the communication between agents by defining several types of events, in which each type of event follows a particular protocol. Due to the hierarchical architecture used in MAGPIE, the communication needs between agents are restricted. Therefore, a powerful communication language such as that described by Finin and colleagues is not necessary in MAGPIE.

All blackboard events have one sender agent and either one receiver agent or a group of receiver agents. We identify three types of events: ***normal event, urgent event,*** and ***announcement*** (Han & Zukerman, 1995). Normal events are messages sent from one agent to another. They are

Figure 6. **Events related to a request.**



collected in the event queue maintained by the local blackboard. In contrast, urgent events are forwarded immediately (without staying in the event queue) to the receiver regardless of whether this agent is waiting for an event or working on a plan. Announcements are messages broadcast by a master agent to all the agents in its group. The broadcasting mechanism is the same as that used for forwarding urgent messages.

A message carried by a normal event may be either a request or a response to a request. A request from one agent demands a modification of the plan generated by another agent, and a response from this other agent indicates whether it is able to comply with the request. Because MAGPIE uses constraints to describe how the variables of one agent's plan are related to the variables of another, constraint propagation is used by each agent to obtain the requirements on its variables from the values of the variables of other agents.

An agent is able to send different requests to different agents and check their response with respect to each request. When an agent picks up a normal event from an event queue, its event handlers determine its reaction to the event (e.g., whether constraint propagation is necessary). For instance, if the table agent that generated the table in Figure 3 wishes to ask an icon agent to reduce the width of the icon it generated, the table agent generates an ***asking-event*** that contains this request (Figure 6). When the icon agent picks up this event, its event handler obtains the expectation on the icon's width by propagating the constraints pertaining to the width of the icon. The handler then tries to reduce the width of the icon to fit this expectation. If this modification fails due to the absence of smaller icons, the icon agent sends a ***rejection-event*** to the table agent on the same request. Otherwise, it sends an *OK-event*.

Announcements and urgent events carry messages that require an agent's immediate attention. They interrupt the process being carried out by the agent and force the agent to handle these events. An example of an announcement is ***time-up***, which indicates that a period of real time has elapsed. A time-up announcement is generated by an alarm process that is set up by the system for a particular amount of time at the beginning of the presentation planning process. When a time-up announcement is sent, all

Figure 7. **Message passing sequence for a request cancellation.**

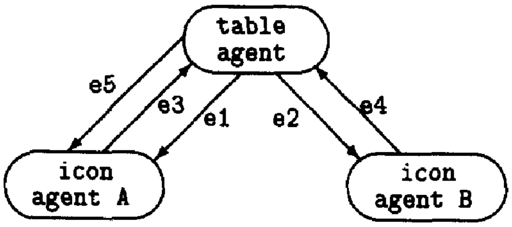


Figure 8. **Events causing a request cancellation.**

ASKING-EVENT e1	
sender:	table agent
receiver:	icon agent A
message:	enlarge Icon$_A$

↑

OK-EVENT e3	
sender:	icon agent A
receiver:	table agent
message:	enlarge Icon$_A$

↑

CANCEL-REQUEST-EVENT e5	
sender:	table agent
receiver:	icon agent A
message:	enlarge Icon$_A$

ASKING-EVENT e2	
sender:	table agent
receiver:	icon agent B
message:	enlarge Icon$_B$

↑

REJECTION-EVENT e4	
sender:	icon agent B
receiver:	table agent
message:	enlarge Icon$_B$

agents stop planning to handle this announcement, which requires them to display the best presentation plan generated so far.

MAGPIE has two types of urgent events: *cancel-request-event* and *remove-agent-event.*

A cancel-request-event is sent from a master agent to a server agent to withdraw a previous request and reinstate the previous result of the server's planning process. To illustrate the usage of a cancel-request-event, consider a situation in which the table agent wants to enlarge a table of two rows and two columns that contains an icon in each entry of the first column and text in the entries of the second column. To continue satisfying the preferred space constraints, the enlargement of the table requires that the icons in the first column and the font of the text in the second column be enlarged (see Section 6.2). The message passing sequence for the first requirement is illustrated in Figure 7, and the events appear in Figure 8. The table agent asks the two icon agents to enlarge their presentations (event e1 and event e2). This request is accepted by icon agent A (event e3), but rejected by icon agent B (event e4). Because of the rejection from icon agent B, this requirement is dropped even though icon agent A has no objection to it. The table agent then creates a cancel-request-event (e5), and proceeds to consider the second of these requirements. Event e5 interrupts icon agent A, and causes it to abort its plan and recover its previous presentation.

A remove-agent-event indicates that a master agent is no longer interested in the display being generated by a server agent, and that the server agent should abort its presentation planning process together with that of its own server agents (if any). One usage of this type of event is in situations in which several agents (e.g., the chart agent and the table agent) compete for the presentation of a piece of information, and one of the agents completes its presentation first in a satisfactory way. This obviates the need for the other agents, which are then sent a remove-agent-event.

4. PLAN REPRESENTATION AND CONSTRAINTS

A presentation structure generated by MAGPIE consists of ***segments*** and ***segment containers*** distributed hierarchically on local blackboards. A segment defines a modality-specific display that presents atomic information (i.e., information that is not decomposable). It determines parameters such as the font, color, and dimensions of the display. A segment container includes a list of elements that in turn can be either segments or segment containers. A segment container is generated by a master agent that hires other agents to generate the elements of this segment container. A segment container describes the display arrangement of these elements as required by the discourse relations between the discourse components that yield these elements. In this section, we first describe the generation of a segment container with particular reference to a segment container for the presentation of a table and then discuss the representation of constraints over presentation plans.

4.1. Table Segment Container

A table agent generates a table segment container to present a data set in the format of columns and rows. Figure 9 illustrates a segment container that stores the parameters defined by a table agent for presenting the table in Figure 3. These parameters are:

1. The width and height of the table (in pixels).
2. The number of columns and rows.
3. The width of the different columns and height of the different rows.
4. The type of column and row separator (defaulted to a solid line).
5. The alignment of each entry in each column (left, center, or right) and row (top, center, or bottom)
6. The headings for columns and rows.
7. The content to be conveyed in each entry (stored in **segment-list**).

In the input in Figure 5, the information to be conveyed in the instantiations has two dimensions (viz., object and weight). The instantiations

Figure 9. **Sample table container: weight of objects.**

modality:	table
position:	(0,0)
width:	316
height:	136
columns:	4
rows:	2
column-width:	(76 44 102 94)
row-height:	(76 60)
column-separator:	solid-line
row-separator:	solid-line
column-alignment:	(center center center center)
row-alignment:	(center center)
row-heading:	nil
column-heading:	("name of object" "mass of object" "magnitude of weight on Earth" "magnitude of weight on the moon")
segment-list:	(("box" "2 kg" "19.6 Newtons" "3.2 Newtons") ("computer" "10 kg" "98 Newtons" "16 Newtons"))

compare the weight of each object on Earth with its weight on the moon. This information can be conveyed either by (a) putting each instantiation in a row or (b) putting each instantiation in a column. The table agent in MAGPIE prefers to use Format (a), by which the attributes to be compared are displayed in columns, because our eyes naturally move from left to right (Holmes, 1984). However, if there are many attributes in focus for a few instantiations, Format (b) is used owing to the relatively small number of columns that can be displayed horizontally. A table with many columns and rows illustrates a situation in which the presentation violates the quality requirements of the discourse. Such a presentation is not acceptable due to the high density of the information being presented. In this case, the table agent may (a) remove rows that present nonessential information (i.e., they have a value of nil for importance), or (b) merge the information presented in several columns and select a modality capable of presenting the resulting composite information (e.g., a vector may be used to convey the magnitude and direction of a force). At present, the second option is not implemented. It will become available when additional graphical modalities are added to MAGPIE.

For the example in Figure 5, the table agent sets up two rows because there are two instantiations in the discourse, and four columns because four elements result from filtering the intended information with the dimensional focus for each dimension (viz., the name of an instantiated object, its mass, the magnitude of its weight on Earth, and the magnitude of its weight on the moon). The table agent fills in the column headings with the attributes in focus, and the entries in the columns with the values of these attributes. These entries are stored in segment-list.[3] The mechanism discussed in Sections 2.2 and 5.2 is then invoked to return suitable modalities for the presentation of each element in the table. In this example, the table agent asks 2 icon agents, 6 number agents, and 10 text agents to generate the entry segments and headings (Section 2.1).[4] After these segments are generated, the table agent propagates the size of the segments to fill in the values of column-width, row-height, width, and height (see Section 5.1). The table agent then calculates the display position of each entry segment from its size, the dimensions of its table entry, and the row and column alignment specifications. The display position of the table is calculated by the presentation planning agent from its size and from information that depends on other components being presented (see Section 6.1).

4.2. Plan Constraints

We define a variable as either a slot or an element of a slot in a segment or a segment container. Each agent in our system is responsible for the instantiation of a subset of the variables in a presentation plan. These are the variables associated with the slots in the agent's modality-specific plan. Relations between modality-specific plans are represented as constraints over variables that belong to different segments or segment containers. Thus, a multimodal presentation plan is found when all the variables are instantiated and all the constraints are satisfied. As stated in Section 1, the constraints in MAGPIE are either required or preferred. The required constraints must be satisfied by all segments and segment containers (see Sections 5 and 6.1). The preferred constraints may remain unsatisfied, but the system endeavors to satisfy as many preferred constraints as possible (see Section 6.2).

Modality constraints result from either (a) discourse relations (see Section 1) or (b) restrictions imposed by a specific modality (see Section 2.2). These constraints are required because they are imposed by a modality-specific generator according to previously generated plans. Modality constraints are specified in terms of a unification operator whose semantics are the same as the semantics of the unification operator in Unification Grammars (Allen, 1994).

[3] In Figure 9, the content of the column headings and the entries is given in plain English for the convenience of the readers. In practice, this content is stored as the attribute of each concept. For example, the word "box" in segment-list in Figure 9 represents the name of the object in `Instantiation1`, that is, (name (object `Instantiation1`)).

[4] The presentation of magnitudes of forces and units of measurement in the same entry in a table requires the definition of an additional segment container, called *multi-text,* which accepts a combination of segments that contain numbers and text.

Figure 10. **Space constraints for a column in a sample table.**

Name	Constraint	Strength
scn_1	$\mathsf{width}(column_1) = \mathsf{width}(entry_1)$	required
scn_2	$\mathsf{width}(column_1) = \mathsf{width}(entry_2)$	required
scn_3	$\mathsf{width}(entry_1) \geq \mathsf{width}(segment_1) + margin\_minimum$	required
scn_4	$\mathsf{width}(entry_2) \geq \mathsf{width}(segment_2) + margin\_minimum$	required
scn_5	$\mathsf{width}(entry_1) \leq \mathsf{width}(segment_1) + margin\_maximum$	preferred
scn_6	$\mathsf{width}(entry_2) \leq \mathsf{width}(segment_2) + margin\_maximum$	preferred

A table agent uses the following rules to impose modality constraints on each entry.

> ***Rule*$_1$** If Format (a) is selected to compare a list of attributes (i.e., putting each instantiation in a row), the same modality is used for all the entries in a column. For example,
> (unify modality(*entry*$_1$) modality(*entry*$_2$)),
> where *entry*$_1$ and *entry*$_2$ stand for the two entries in the first column of the table in Figure 3.
>
> ***Rule*$_2$** If Format (b) is selected to compare a list of attributes (i.e., putting each instantiation in a column), the same modality is used for all the entries in a row.
>
> ***Rule*$_3$** Column headings and row headings must be textual. For example,
> (unify modality(*column-heading*$_1$) (`text`)),
> where *column-heading*$_1$ stands for the heading of the first column in Figure 3.
>
> ***Rule*$_4$** Only `icon`, `number`, and `text` may be used (at present) for presenting the content of each entry. For example,
> (unify modality(*entry*$_1$) (`icon number text`)) and
> (unify modality(*entry*$_2$) (`icon number text`)).

The required space constraints generated by a table agent demand that entries in the same row have the same height, entries in the same column have the same width, and that an entry segment be displayed in a position that ensures at least a minimum margin from the borders of its entry. The preferred space constraints generated by a table agent require that the maximum width of the margins surrounding the presentation segments in table entries does not exceed a certain threshold, so that a segment is not too small for its entry. Figure 10 shows the space constraints that pertain to the width of the first column of the table in Figure 3. In Figure 10, *entry*$_1$ and *entry*$_2$ stand for the two entries in this column, and *segment*$_1$ and *segment*$_2$ stand for the segments in *entry*$_1$ and *entry*$_2$, respectively.

Constraints are created by a master agent when it generates its segment container. All the constraints are stored in the local blackboard, so that the information may be shared with its server agents. Therefore, when the master agent assigns values to its segment container, the server agents in its agent group know the requirements placed on their partial plans. A server agent can then add its own constraints if it is the master agent of another group of agents. Therefore, during the planning process, requirements of an existing plan are transferred to server agents by means of constraint propagation. These constraints ensure that each component segment satisfies the requirements of the overall discourse. For instance, if the presentation planning agent in Figure 4 wants the text to be displayed to the right of the table in the same window, it creates constraints on the width and height of the table and the text in relation to the window size (Han, 1996). The constraints pertaining to the table are propagated to the table agent and thus restrict the expansion of the table. In this case, the space left for the text must be wide enough for the longest word in the text and must exceed a certain proportion of the window. Otherwise, the text is displayed below the table, and a different set of constraints is created.

5. CONSTRAINT PROPAGATION

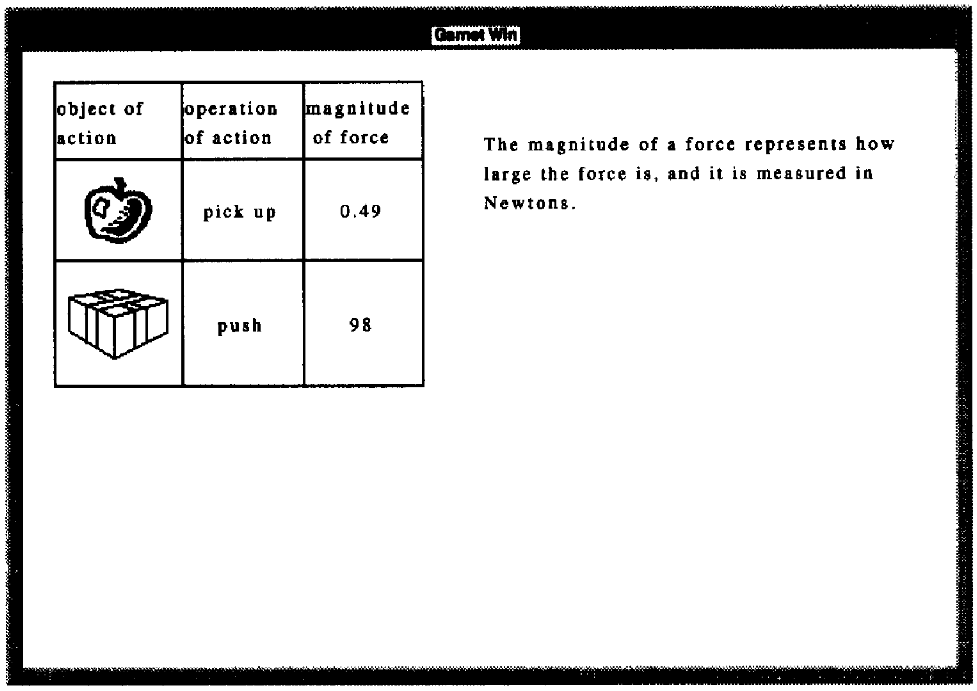
Because constraints are distributed in the plan hierarchy, the constraint satisfaction problem is considered a Distributed Constraint Satisfaction Problem, for which a solution is found through agent cooperation. MAGPIE uses local constraint propagation and unification algorithms to solve this problem. In this section, we first describe the propagation of space constraints, and then discuss the propagation of modality constraints.

5.1. Space Constraint Propagation

In MAGPIE, agents in a group must satisfy the constraints imposed on their plans because they are presenting a discourse structure cooperatively. To this effect, each agent in a group must (a) have a collection of all the constraints pertaining to its variables, (b) be able to determine the effect of the values of another agent's variables on its own variables, and (c) be able to inform another agent of its expectations (regarding the values that can be assigned to the variables of this other agent) and to inform this other agent whether it considers the values assigned to these variables satisfactory. In principle, these requirements can be satisfied by allowing one agent to simply ask another agent for the values of its variables. However, constraint propagation still has to be performed because the agents must ensure that they satisfy interagent constraints.

To illustrate these requirements, let us consider the presentation shown in Figure 11. It contains two Assertions presented in text: the magnitude of a force represents how large the force is, and it is measured in Newtons. In

Figure 11. **Sample multimodal presentation: simple table.**



object of action	operation of action	magnitude of force
🍎	pick up	0.49
📦	push	98

The magnitude of a force represents how large the force is, and it is measured in Newtons.

addition, the amount of force required to move some commonplace objects is illustrated in a table. The units of force are ignored in the table to simplify the example, so that fewer agents are involved in the generation of this presentation.

We adopt the constraints in Figure 10 for the first column of the table in Figure 11. These constraints are numerical and the values that satisfy them are nondeterministic. Further, let us assume that ***margin_maximum*** and ***margin_minimum*** are constant. Thus, when $\mathrm{width}(\mathit{segment}_i)$ is assigned a value (as a result of a presentation generated by the icon agent for $\mathit{segment}_i$), the following new constraints result from propagating the value of $\mathrm{width}(\mathit{segment}_i)$:

$$\mathrm{width}(\mathit{column}_1) \geq \mathrm{width}(\mathit{segment}_1) + \mathit{margin\_minimum}$$
$$\mathrm{width}(\mathit{column}_1) \leq \mathrm{width}(\mathit{segment}_1) + \mathit{margin\_maximum}$$

In this manner, the icon agent presenting $\mathit{segment}_1$ can convey to the table agent its expectation of $\mathrm{width}(\mathit{column}_1)$. The table agent in turn can determine whether its assignment of $\mathrm{width}(\mathit{column}_1)$ and the value of $\mathrm{width}(\mathit{segment}_1)$ assigned by the icon agent are satisfactory. Because the agents presenting $\mathit{segment}_1$ and $\mathit{segment}_2$ are independent processes, they generate constraints that affect simultaneously $\mathrm{width}(\mathit{column}_1)$. MAGPIE does not allow these agents to write directly to $\mathrm{width}(\mathit{column}_1)$ when they propagate the constraints resulting from the width of their segments in order to prevent this value from being overwritten in unpredictable ways.

MAGPIE satisfies the three requirements previously mentioned by providing *channels* that support the propagation of constraints from one level in a plan hierarchy to the next. A channel consists of a set of constraints (usually one or two). It specifies a sequence of evaluations that are performed to propagate the values of variables in an agent's plan to the variables of another agent's plan. Channels are maintained by a blackboard (e.g., the blackboard bound to the table agent and its server agents in this example). They are not the responsibility of individual agents because they involve interagent constraints. When a constraint is added or removed by the master agent in an agent group, a channel may need to be modified so that the newly added constraint can be included, or the constraint to be removed can be eliminated. To satisfy the first of the previously mentioned requirements, a list of the channels that affect each agent's variables is maintained by the blackboard. To satisfy the other two requirements, an agent invokes the local propagation process on a channel (either from another agent to itself or vice versa). Value binding is used when propagating space constraints through channels to avoid problems that arise when attempting to assign more than one value to a variable at the same time.

Among the constraints in Figure 10, scn_1 and scn_3 form a channel because the right-hand side of scn_1 contains the variable on the left-hand side of scn_3. Other channels can be formed from (a) scn_1 and scn_5, (b) scn_2 and scn_4, and (c) scn_2 and scn_6. In our example, four value ranges for width($column_1$) can be obtained from the propagation of constraints through these four channels. A value from the intersection of these four ranges satisfies scn_1 to scn_6 and therefore can be assigned to width($column_1$) by the table agent.

Channels enable the different agents in a group to keep track of the constraints placed on their variables and to modify their presentation plan accordingly. For example, the table agent can set width($column_1$) to a particular value, and send an asking-event to the icon agent requesting it to enlarge the width of $segment_1$ to fit in the column. The icon agent then calculates width($segment_1$) from the channel formed by scn_1 and scn_3 and the channel formed by scn_1 and scn_5, and selects an icon that matches this requirement (if such an icon is available). Normally, a wider icon is also taller. If the height of the selected icon does not satisfy the required height constraints, the icon agent sends an asking-event to the table agent requesting it to enlarge the height of its row. To this effect, the icon agent uses height-related channels pertaining to $segment_1$ that transmit the constraints for a new row height to fit $segment_1$. If the table agent approves the request and sets a new value for height(row_1), this value is transferred via other height-related channels to the agents that are presenting other segments in the same row. A consequence of this mechanism is that an agent can modify indirectly a partial plan of another agent.

5.2. Modality Constraint Propagation

Modality constraint propagation is invoked during task decomposition when a master agent determines the modalities to be used to perform a subtask. As discussed in Section 2.2, candidate modalities are currently obtained from the input, but in the future they will be determined based on the information characteristics of the information being conveyed and the capabilities of each modality. Modality constraints are nonnumerical, and an instantiation of variables in a modality constraint is a list of alternative modality names. Thus, a unification algorithm rather than value binding is used to propagate modality constraints. This unification algorithm is applied to a modality constraint and a substitution of its variables. It returns a new substitution of the variables where candidates that do not satisfy the modality constraint are eliminated. For instance, when the unification algorithm is applied to a constraint such as

(unify modality(*column-heading*$_1$) (`text`)),

generated from *Rule*$_3$ (see Section 4.2), `text` remains as the only possible substitution for modality(*column-heading*$_1$).

For each component, the propagation of modality constraints results in either a single substitution or several substitutions. If a single substitution is generated, an agent is created for this modality. Thus, a text agent is activated for presenting the heading of the first column of the table in Figure 11, even though there may be other modalities capable of presenting the information in this heading. At present, MAGPIE is restricted to using text to convey a description of movement (e.g., the operation of an action) and numbers to convey numerical information (e.g., the magnitude of a force). Thus, `text` and `number` remain as the only possible modalities for the second and third column of the table in Figure 11 respectively after unification with the modality constraints pertaining to the entries in these columns.

If several substitutions result from the propagation of a modality constraint, heuristics based on a perceiver's preferences are used to select modalities from these alternatives, and agents are created for the selected modalities. Resource restrictions due to discourse relations and the resource consumption of the candidate modalities are then taken into consideration to make a final modality selection. For example, consider the presentation of the object of an action in the first column of the table in Figure 11. The candidate modalities for presenting this object are `icon` and `text`. Both of these modalities remain as possible candidates because they unify with the constraints generated from *Rule*$_4$, namely

(unify modality(*entry*$_i$) (`icon number text`)) $i = 1, 2$.

From the two substitutions for modality(*entry*$_i$), MAGPIE selects `icon` first for presenting all the entries in the first column, as this modality is preferred by a user with a low level of literacy. It then activates two icon agents and waits for both of them to complete their task. If either of the icon agents fails, icon cannot be used to present the other object in this column because the constraint generated from *Rule*$_1$ requires that the same modality be used for *entry*$_1$ and *entry*$_2$. In this case the icon agent is removed, and two text agents are activated.

The presentation of the instantiations in Figure 5 illustrates a different usage of the these resource-related considerations. These instantiations can be conveyed by a table, a chart, or text, and no modality constraint is imposed on this component. Among these three modalities, MAGPIE determines from the user's profile that a table or a chart is preferred by the user (see Section 2.2). However, the system cannot determine from this profile whether the user prefers a table or a chart. Thus, an agent is created for each of these modalities. The agent that generates a presentation suitable for the space available on the screen in the shortest time is selected to present the instantiations. If none of these modalities can present the intended information, a text agent is activated (both tables and charts require more space than the equivalent text, so they cannot be used if there is not enough space on the screen).[5]

6. AGENT NEGOTIATION

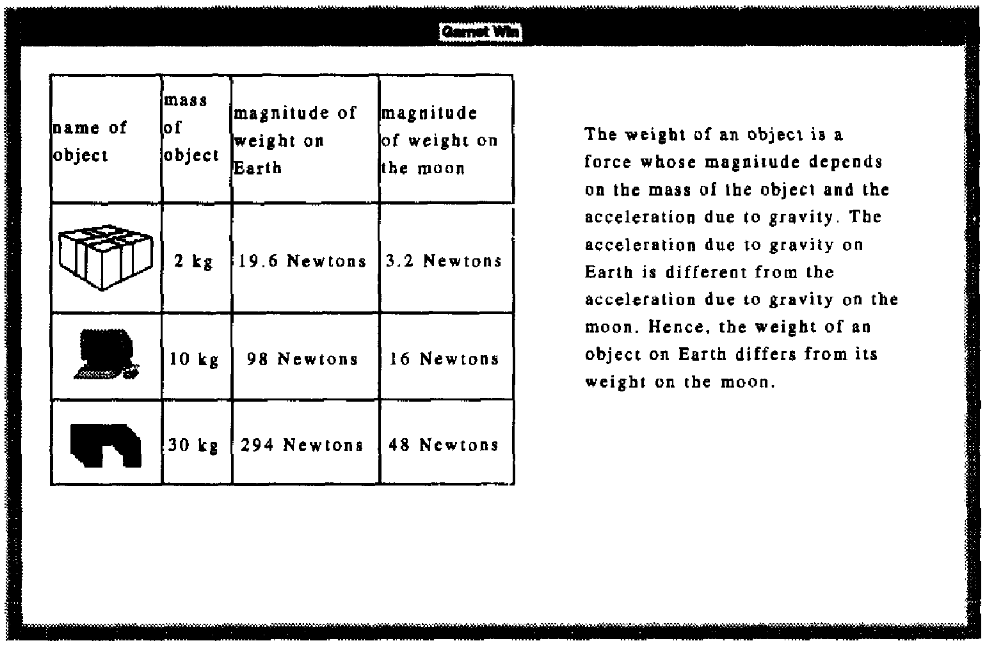
During presentation planning, agents start a negotiation process when a required constraint is violated or, if time permits, when a preferred constraint is violated. In this section, we describe policies implemented for the agent negotiation process in each situation.

6.1. Violation of a Required Constraint

When a group of agents works on a presentation plan, all the required constraints on the blackboard bound to these agents must be satisfied. If a required modality constraint is violated, the presentation task is relocated to another modality (see Section 5.2). However, if a required space constraint is violated, agents may enter a negotiation process. To illustrate this, consider a situation where an instantiation for a 30-kg desk is added to the discourse in Figure 1. A new row is then added to the table in Figure 3 to convey this new object, its mass, and its weight on Earth and on the moon, yielding the table in Figure 12.

[5] In principle, a system could create agents for all the modalities that are capable of presenting the intended information, so that a replacement modality can be readily selected if the preferred modality fails. However, MAGPIE is not implemented in this manner to limit the number of redundant agents.

Figure 12. **Sample multimodal presentation: table with three instantiations.**



name of object	mass of object	magnitude of weight on Earth	magnitude of weight on the moon
	2 kg	19.6 Newtons	3.2 Newtons
	10 kg	98 Newtons	16 Newtons
	30 kg	294 Newtons	48 Newtons

The weight of an object is a force whose magnitude depends on the mass of the object and the acceleration due to gravity. The acceleration due to gravity on Earth is different from the acceleration due to gravity on the moon. Hence, the weight of an object on Earth differs from its weight on the moon.

The addition of this instantiation to the chart requires the generation of two new bars (one to display the weight of the desk on Earth and another to display its weight on the moon). The chart agent has the following required constraint for the height of a chart to ensure that the smallest difference between bars is visible on the screen:

$$\mathrm{height}(chart) \geq diff \times \frac{v_n - v_1}{\min_{i=1}^{n-1}\left\{\left[v_{i+1} - v_i\right] \middle| v_{i+1} > v_i\right\}} + diff + \mathrm{height}(label_1) + \mathrm{height}(unit),$$

where v_1, v_2, ..., v_n are the values to be presented in the chart ordered in ascending order, ***label***$_1$ is the first label on the x-axis, ***unit*** is the string at the top of the chart, and ***diff*** is a constant that represents the smallest visible difference between bars. The height of the first label on the x-axis is added because the same font is used for all the labels; ***diff*** is added because we want to be able to see at least the top of the smallest bar. The minimum difference between two values of v is considered only when this difference is greater than zero, because it is possible that two bars in a chart are of the same height.

For the intended information in this example, the chart is too tall to fit into the display window. This violates a required constraint posted by the presentation planning agent in Figure 4 that restricts the height of a chart to be smaller than the height of the display window. The chart agent and the presentation planning agent then enter a negotiation process. During this process, the chart agent sends an asking-event to its master agent requesting the relaxation of the constraint on its height. The presentation planning agent refuses this demand through a rejection-event because the size of the display window cannot normally be changed. Upon receiving this event, the bar chart agent decides that it is unable to present the information and reports a failure to its master agent. Finally, the presentation planning agent sends a remove-agent-event to the chart agent, and the information is presented in a table (Figure 12). The bar chart is shown in Figure 13, where the window has been deliberately enlarged to satisfy the required constraints. Note the different position of the text in Figures 12 and 13. An initial presentation of the text is generated at the same time as the table and the chart in a format that fits the window size, and is assigned to a specific location by the presentation planning agent. Because MAGPIE prefers to present the text next to the chart or the table (Holmes, 1984), in Figure 12 the text is reshaped and placed next to the table on completion of the generation of the table. In contrast, the text in Figure 13 is displayed below the chart because there is no room next to the chart.[6]

6.2. Violation of a Preferred Constraint

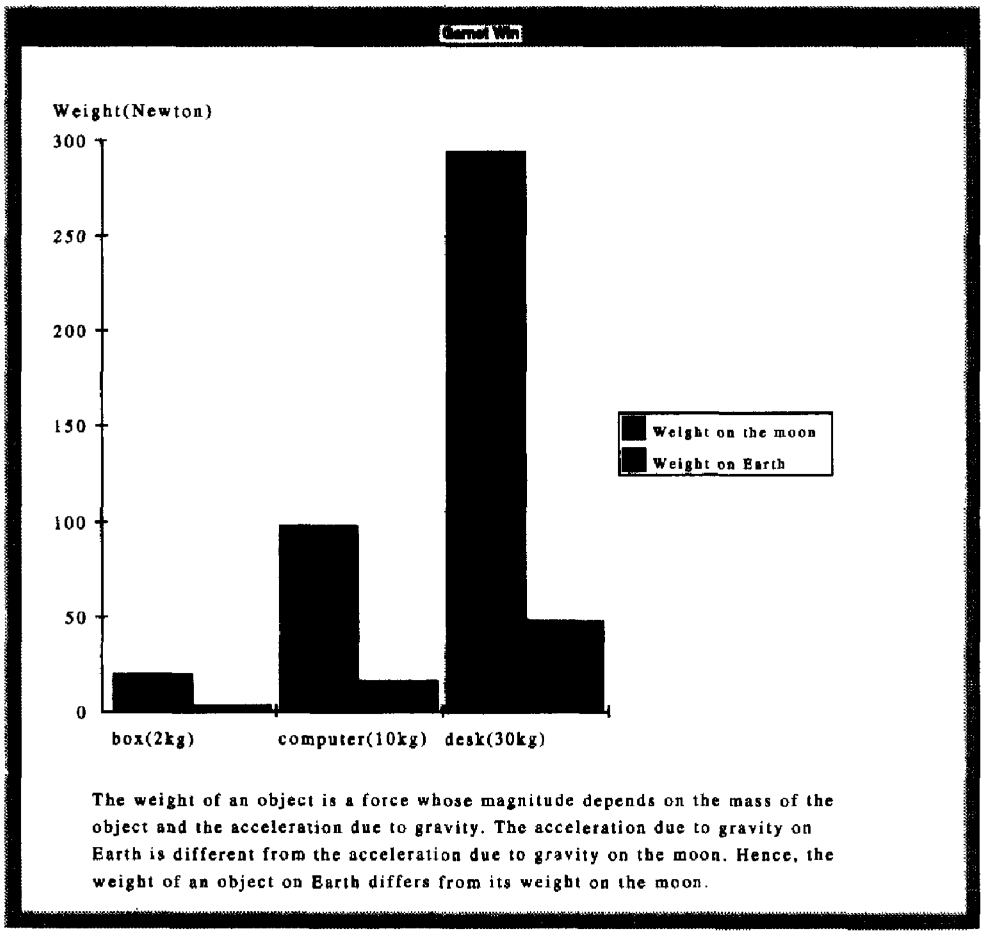
Agents initially select values for the variable slots of their partial plans to satisfy the required constraints. If time allows, agents look for plans that satisfy as many preferred constraints as possible. Each type of agent has its own algorithms for generating an initial plan and improving it. In this section, we discuss only the algorithms used by the table agent; the other algorithms are described by Han (1996).

The table agent starts from an initial plan in which all the entry segments are generated from default slot values (i.e., text and numbers are in a default font, and icons in a default size). At present, icons of different sizes are prestored in the icon library. However, in the future we propose to store only default-sized icons in the library and to generate icons of a particular size from the stored icons on demand (subject to the requirement that the quality of the generated icons is acceptable). The initial plan is the quickest presentation that an agent can generate. The initial plan of the table agent has values for the width of each column and the height of each row so that the biggest segment fits in the table. If there are m entries in $column_n$ and one segment in each entry, $\mathrm{width}(column_n)$ is calculated from the propagation of constraints such as those in Figure 10. That is:

$$\begin{aligned}\mathrm{width}(column_n) = \max\{&\mathrm{width}(heading_n),\\ &margin\_minimum + \mathrm{width}(segment_1), \ldots,\\ &margin\_minimum + \mathrm{width}(segment_m)\}\end{aligned}$$

[6] In the future, enhancements to the chart agent will enable it to move the legend to leave more space to the right of the chart so that the text can fit there. To this effect, the presentation planning and chart agents will have to engage in additional negotations.

Figure 13. **Sample multimodal presentation: chart with three instantiations.**

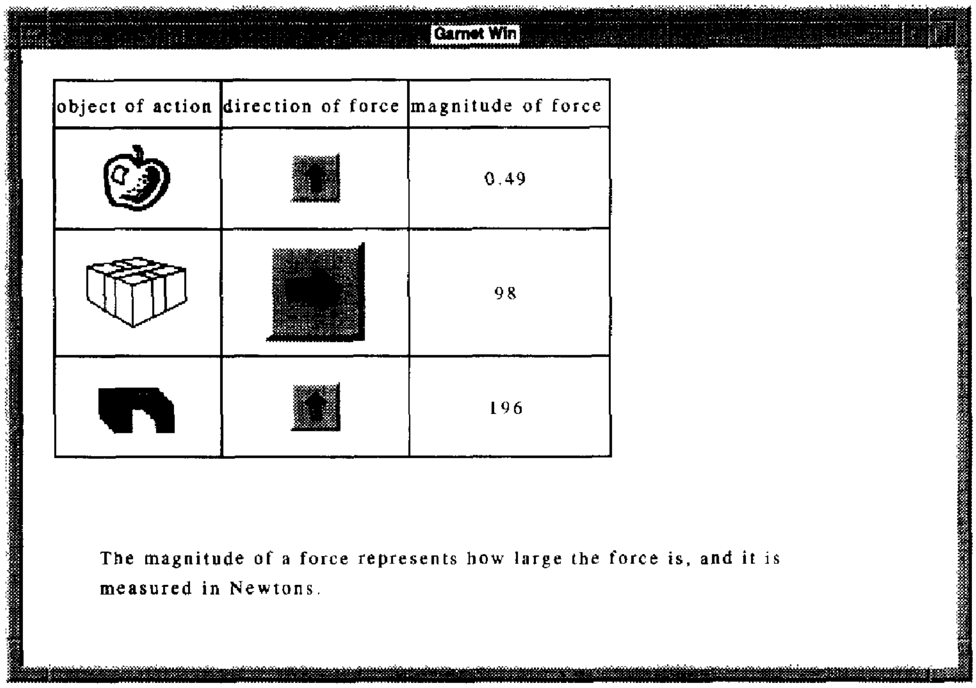


The weight of an object is a force whose magnitude depends on the mass of the object and the acceleration due to gravity. The acceleration due to gravity on Earth is different from the acceleration due to gravity on the moon. Hence, the weight of an object on Earth differs from its weight on the moon.

where width(*heading$_n$*) is the width of the column heading. This initial plan satisfies all the required constraints and can be displayed immediately if necessary. However, some preferred constraints, such as *scn$_5$* in Figure 10, may remain unsatisfied, leading to a presentation that some users would find unacceptable. Figure 14 presents such an example, in which the table contains one big icon and two small icons. This is similar to the table in Figure 11, but it has an additional instantiation, and its second column indicates the direction of the applied force rather than the operation of the action. A big right-arrow icon is presented in the second entry of this column because it is the smallest icon that is larger than the default size. A small up-arrow icon is displayed in the other entries of this column because only this size is available in the icon library.

The discrepancy in the size of the arrow icons yields a presentation that violates the preferred constraints that restrict the space around the first and third entry segments in the second column. To satisfy these constraints, the table agent can either reduce the width of this column (so that the entries fit most segments) or enlarge the width of the two small icons. The first

Figure 14. **Initial multimodal presentation: large icon.**



object of action	direction of force	magnitude of force
	↑	0.49
	→	98
	↑	196

The magnitude of a force represents how large the force is, and it is measured in Newtons.

option violates a required constraint with respect to the second entry segment, which does not fit the smaller width of the column. To satisfy this required constraint, MAGPIE propagates the new value for the column width to the server agents. The server agents that do not have presentations satisfying the required constraint (e.g., the server agent for the right-arrow icon in this example) return modified presentations if possible, thereby yielding a plan that satisfies all the required constraints and additional preferred constraints. The second option may cause the violation of a required constraint with respect to the height of a row because a wider icon is likely to be taller and may not fit in its row. In this case the agents engage in a negotiation process in which an icon agent presenting an up-arrow icon asks the table agent to increase the height of the row that contains the icon. If the icon agent receives an OK-event in regard to this request, it in turn creates an OK-event to respond to the request sent by the table agent for the enlargement of the width of the icon. Otherwise, the request of the table agent is rejected, leading to the failure of this option. An alternative method for improving an initial plan is to reduce the largest difference in the width of the segments in each column and in the height of the segments in each row. However, additional constraints must be created to propagate these differences.

The procedure for proposing possible modifications to satisfy additional preferred constraints is implemented as follows. We define a proposal as a set of actions. Each action is a pair of the form (*to-do variable*$_i$), where *to-do* can be either `enlarge` or `reduce`. It represents a modification request on a variable of *segment*$_i$ (e.g., width(*segment*$_i$)). Two proposals are the same if they contain the same actions (irrespective of their order). To find all the proposals that satisfy additional preferred constraints with respect to the width of a given column, we look at all the ranges that result from propagating the constraints pertaining to the width of this column. If these ranges unify (i.e., they have a nonempty intersection), all segments are of similar size and no action is proposed. Otherwise, an action is proposed to modify each segment that causes an outstanding range (i.e., a segment that is much larger or much smaller than the others). This outstanding range is then removed and we proceed to consider the rest of the ranges. A right-open range (e.g., [80, ∞)) is outstanding in a set of ranges if it starts from a point that is greater than the starting point of any other right-open range in the set. For instance, between [80, ∞) and [50, ∞), [80, ∞) is an outstanding range, that is, the segment yielding [80, ∞) is too large. A left-open range (e.g., (∞, 80]), is outstanding in a set of ranges if it finishes at a point that is smaller than the finishing point of any other left-open range in the set. For instance, between (∞, 80] and (∞, 112], (∞, 80] is an outstanding range, that is, the segment yielding (∞, 80] is too small. A `reduce` action is proposed for a segment leading to an outstanding right-open range, and an `enlarge` action for a segment leading to an outstanding left-open range.

Algorithm *Find-proposals* is called recursively to generate proposals that can satisfy additional preferred constraints with respect to the width of a column (Figure 15). It receives as input a list of ranges to be unified and a previously generated proposal, which is initially *nil*. The proposals generated by this algorithm for each column in a table are then exhaustively combined to produce a set of proposals for the entire table. Each of the proposals in this set is attempted in turn until a time-up announcement is received. During this process, if MAGPIE generates a plan that satisfies more preferred constraints than the plan currently stored in the blackboard, it replaces the previous plan with this new plan. As a result, when the time-up announcement is received, the best plan generated so far is displayed. Find-proposals does not consider the width of a column heading because a column heading cannot be modified independently from the headings of other columns (this is because all column headings must be in the same font), and a column heading can also be divided into multiple lines if necessary.

As an example, consider the three segments in the second column of the table in Figure 14, and assume that

$$\textit{margin\_minimum} = 12,\ \textit{margin\_maximum} = 42,\ \mathsf{width}(\textit{segment}_1) = 38,$$
$$\mathsf{width}(\textit{segment}_2) = 70, \text{ and } \mathsf{width}(\textit{segment}_3) = 38.$$

We adopt the constraints in Figure 10 for the first two entries and list the constraints for the third entry in Figure 16. The initial value for width(*col-*

Figure 15. **Algorithm Find-proposals.**

```
Procedure Find-proposals ({ranges},proposal)
Set {proposals} to $\emptyset$
Unify the ranges in {ranges}
If the intersection of the ranges is not empty then return proposal
Else
    Set {actions} to a set of actions which modify the segments that
        lead to an outstanding right-open range or an outstanding
        left-open range
    For each action $\in$ {actions}
        Set {newranges} to {ranges} minus the ranges that result
            from the propagation of a constraint on the segments
            affected by action
        Set {proposals} to
            {proposals} $\cup$
            Find-proposals ( {newranges}, proposal $\cup$ action )
        Return {proposals}
```

Figure 16. **Additional space constraints for a column in a sample table.**

Name	Constraint	Strength
scn_7	width($column_2$) = width($entry_3$)	required
scn_8	width($entry_3$) $\geq$ width($segment_3$) + $margin\_minimum$	required
scn_9	width($entry_3$) $\leq$ width($segment_3$) + $margin\_maximum$	preferred

Figure 17. **Ranges for the width of a column.**

Channel	Requirement for width($column_2$)	Result
$scn_1 \leftrightarrow scn_3$	$\geq$ width($segment_1$) + $margin\_minimum$	$\geq$ 50
$scn_2 \leftrightarrow scn_4$	$\geq$ width($segment_2$) + $margin\_minimum$	$\geq$ 82
$scn_7 \leftrightarrow scn_8$	$\geq$ width($segment_3$) + $margin\_minimum$	$\geq$ 50
$scn_1 \leftrightarrow scn_5$	$\leq$ width($segment_1$) + $margin\_maximum$	$\leq$ 80
$scn_2 \leftrightarrow scn_6$	$\leq$ width($segment_2$) + $margin\_maximum$	$\leq$ 112
$scn_7 \leftrightarrow scn_9$	$\leq$ width($segment_3$) + $margin\_maximum$	$\leq$ 80

umn_2) is 127 to fit *column-heading₂*. This value satisfies the required constraints scn_1 to scn_4, scn_7, and scn_8. Figure 17 shows the six ranges for width($column_2$) resulting from the constraints in Figures 10 and 16. The table agent then attempts to satisfy preferred constraints by calling Find-proposals with these ranges. Because the intersection between these ranges is empty (the column width should be less than 80 and greater than 82), Find-proposals initially yields two options: (a) (reduce width($segment_2$))—due to the outstanding right-open range obtained from $scn_2 \leftrightarrow scn_4$; and (b) (enlarge width($segment_1$))—due to the outstanding left-open range obtained from $scn_1 \leftrightarrow scn_5$. For the first option, after the ranges related to width($segment_2$) are deleted, the remaining ranges unify. In contrast, in the second option, after the ranges related to width($segment_1$) are deleted, the same left-open range remains because $segment_3$ is the same size