#8

UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS
UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. 20231
www.uspto.gov

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
09/524,095	03/13/2000	Christine Halverson	SRI1P037

24277
Kevin J. Zilka
PO Box 721030
San Jose, CA 95172

OC000000005610555

Date Mailed: 12/11/2000

NOTICE REGARDING POWER OF ATTORNEY

This is in response to the Power of Attorney filed 12/04/2000.

- The Power of Attorney to you in this application has been revoked by the applicant. Future correspondence will be mailed to the new address of record(37 CFR 1.33).

Lavinia Johnson
Customer Service Center
Initial Patent Examination Division (703) 308-1202

OFFICE COPY

#9

United States Patent and Trademark Office

COMMISSIONER FOR PATENTS
UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. 20231
www.uspto.gov

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
09/524,095	03/13/2000	Christine Halverson	SRI1P037

OC000000005610560
OC000000005610560

I. KEITH STEPHENS
CARLTON, FIELDS, WARD, EMMANUEL, SMITH & CUTLER
P.O. BOX 3239
TAMPA, FL 33601-3239

Date Mailed: 12/11/2000

NOTICE REGARDING POWER OF ATTORNEY

This is in response to the Power of Attorney filed 12/04/2000.

The Power of Attorney in this application is accepted. Correspondence in this application will be mailed to the above address as provided by 37 CFR 1.33.

Laurina Johnson
Customer Service Center
Initial Patent Examination Division (703) 308-1202

OFFICE COPY

UNITED ST. ;S DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
09/524,095	03/13/00	HALVERSON C	SRI1P037

TM02/0424

L. KEITH STEPHENS
CARLTON, FIELDS, WARD, EMMANUEL, SMITH &
P.O. BOX 3239
TAMPA FL 33601-3239

EXAMINER	
BACKER, F	
ART UNIT	PAPER NUMBER
2155	10

DATE MAILED: 04/24/01

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

Commissioner of Patents and Trademarks

PTO-90C (Rev.11/00)

1- File Copy

Office Action Summary	Application No. 09/524,095	Applicant(s) HALVERSON ET AL.	
	Examiner Firmin Backer	Art Unit 2155	

-- The MAILING DATE of this communication appears on the cover sheet with the 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 *13 March 2000* .

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) *56-126* 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) *56-126* is/are rejected.

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

8)☐ Claims _____ 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 objected to by the Examiner.

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. § 119

13)☐ 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.

14)☐ Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

Attachment(s)

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

16) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

17) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .

18) ☐ Interview Summary (PTO-413) Paper No(s). _____ .

19) ☐ Notice of Informal Patent Application (PTO-152)

20) ☐ Other: .

DETAILED ACTION

This is in response to a letter for patent filed on June 30th, 2000 in which claims 56-126 are presented for examination. Claims 56-126 are pending in the letter.

Claim Rejections - 35 USC § 102

1. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371(c) of this title before the invention thereof by the applicant for patent.

2. Claims 56-126 are rejected under 35 U.S.C. 102(e) as being anticipated by Levin et al. (U.S. Patent No. 6,173,279).

3. As per claim 56, Levin et al teach a method for speech-based navigation (information server, 110) of an electronic data source located at one or more network servers located remotely from a user, (see abstract, fig 1, column 3 lines 5-35), comprising receiving a spoken request (*receive a natural language query*) for desired information from the user (user); rendering an interpretation (*creating a semantic representation*) of the spoken request, constructing a navigation (*generating search)* query based upon the interpretation; soliciting additional input from the user (*one or more questions are generated*...), including user interaction in a modality different that the original request and, refining the navigation query, based upon the additional

input (see column 6 lines 20-59), using the 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 primarily stationary, display device located locally with the user. (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22)

4. As per claim 57, Levin et al teach a method of rendering the interpretation includes deriving linguistic information by using a speech recognition and a linguistic parser (see abstract, fig 1, column 3 lines 37-5 lines 40).

5. As per claim 58-62, Levin et al teach a method of constructing a navigation query in the form of a database query on a computing device located on a network including extracting an input template for an online scripted interface to the data source to be used for the construction of the navigation query and dynamically scraping the online scripted interface (see abstract, fig. 1-3, column 3 line 36-9 line 5)

6. As per claim 63-68, Levin et al teach a method of soliciting additional input is performed in response deficiency including unresolved word encountered after the first navigation of the data source, required element of the navigational query, data recorded within the data source, failure to identify data record responsive to navigational query (see column 6 lines 20-59).

7. As per claim 69, Levin et al teach a method wherein the additional input is solicited upon receiving a user-input statement…(see column 6 lines 20-59).

8. As per claim 70-73, Levin et al teach a method of soliciting additional input from the user, including presenting: a menu, a textual or an audible request, a list of portions of data source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

9. As per claim 74-75, Levin et al teach a method wherein additional input received from the user is speech based, of no spoken input source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

10. As per claim 76, Levin et al teach a method wherein steps (d)-(e) are repeated until the navigational query if deemed adequate source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

11. As per claim 77, 78, Levin et al teach a method wherein the input modality includes selecting (by speaking) from a displayed option menu (see abstract, fig. 1-3, column 3 line 36-9 line 5).

12. As per claim 79, Levin et al teach a method performed with respect to a plurality of user and corresponding client devices (see abstract, fig. 1-3, column 3 line 36-9 line 5).

13. As per claim 80-81, Levin et al teach a method of selecting data source from plurality of electronic data source storing multimedia content including audio and video content (see abstract, fig. 1-3, column 3 line 36-9 line 5)

14. As per claim 82, Levin et al teach a system for speech-based navigation (*information server, 110*) of an electronic data source located at one or more network servers located remotely from a user, (see abstract, fig 1, column 3 lines 5-35), comprising a portable microphone (*microphone, 105*) receiving a spoken request (*receive a natural language query*) for desired information from the user (user) a language processing logic (*natural language server, 114*) rendering an interpretation (*creating a semantic representation*) of the spoken request, (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22) a query construction logic (*service host, 112*) constructing a navigation (*generating search)* query based upon the interpretation; a query interaction logic (*service host, 112)* soliciting additional input from the user (*one or more questions are generated...*), including user interaction in a modality different that the original request and, (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22), a query refining logic (*service host, 112)* refining the navigation query, based upon the additional input (see column 6 lines 20-59), a navigation logic (*service host, 112)* using the navigation query to select a portion of the electronic data source; electronic infrastructure (*network, 108*) transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user. (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22).

15. As per claim 83, Levin et al teach a system of rendering the interpretation includes deriving linguistic information by using a speech recognition and a linguistic parser (see abstract, fig 1, column 3 lines 37-5 lines 40).

16. As per claim 84-86, Levin et al teach a system of constructing a navigation query in the form of a database query on a computing device located on a network including extracting an input template for an online scripted interface to the data source to be used for the construction of the navigation query and dynamically scraping the online scripted interface (see abstract, fig. 1-3, column 3 line 36-9 line 5).

17. As per claim 87, 88, 100, Levin et al teach a system wherein at least a portion of the language processing if hosted on a computing device coupled with a microphone located locally with a user and a network computing device located remotely and data in a two-way communication infrastructure (coaxial, DSL, satellite, wireless/cellular, fiber-optic) (see abstract, fig. 1-3, column 3 line 36-9 line 5).

18. As per claim 89-94, Levin et al teach a system of soliciting additional input is performed in response deficiency including unresolved word encountered after the first navigation of the data source, required element of the navigational query, data recorded within the data source, failure to identify data record responsive to navigational query (see column 6 lines 20-59).

19. As per claim 95, 96, Levin et al teach a system wherein the input modality includes selecting (by speaking) from a displayed option menu (see abstract, fig. 1-3, column 3 line 36-9 line 5).

20. As per claim 97-98, Levin et al teach a system of selecting data source from plurality of electronic data source storing multimedia content including audio and video content (see abstract, fig. 1-3, column 3 line 36-9 line 5).

21. As per claim 99, Levin et al teach a system wherein the display device receives data from the electronic device on the network via a communication box (see abstract, fig. 1-3, column 3 line 36-9 line 5).

22. As per claim 101, Levin et al teach a computer program for speech-based navigation (information server, 110) of an electronic data source located at one or more network servers located remotely from a user, (see abstract, fig 1, column 3 lines 5-35), comprising code segment receiving a spoken request (*receive a natural language query*) for desired information from the user (user); code segment rendering an interpretation (*creating a semantic representation*) of the spoken request, code segment constructing a navigation (*generating search)* query based upon the interpretation; soliciting additional input from the user (*one or more questions are generated...*), including user interaction in a modality different that the original request and, code segment refining the navigation query, based upon the additional input (see column 6 lines 20-59), code segment using the navigation query to select a portion of the electronic data source; and code segment transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22).

23. As per claim 102, Levin et al teach a code segment deriving linguistic information by using a speech recognition and a linguistic parser (see abstract, fig 1, column 3 lines 37-5 lines 40).

24. As per claim 103-105, Levin et al teach a code segment of constructing a navigation query in the form of a database query on a computing device located on a network including extracting an input template for an online scripted interface to the data source to be used for the construction of the navigation query and dynamically scraping the online scripted interface (see abstract, fig. 1-3, column 3 line 36-9 line 5).

25. As per claim 106-107, Levin et al teach a computer program wherein rendering of the interpretation and the construction of the navigation query are performed on a computing device located locally with or remotely from the user (see abstract, fig. 1-3, column 3 line 36-9 line 5).

26. As per claim 108-114, Levin et al teach a code segment that solicits additional input display on option menu is performed by speaking in response deficiency including unresolved word encountered after the first navigation of the data source, required element of the navigational query, data recorded within the data source, failure to identify data record responsive to navigational query (see column 6 lines 20-59).

27. As per claim 115, Levin et al teach a computer program the act of selecting from the display is performed by speaking (see column 6 lines 20-59)

28. As per claim 116, Levin et al teach a code segment of the computer program operate with respect to a plurality of simultaneous user and corresponding client devices (see abstract, fig. 1-3, column 3 line 36-9 line 5).

29. As per claim 117, Levin et al teach a code segment that select data source form a plurality of electronic data source .... content (see abstract, fig. 1-3, column 3 line 36-9 line 5).

30. As per claim 118, Levin et al teach a computer program of selecting data source from plurality of electronic data source storing multimedia content including audio and video content (see abstract, fig. 1-3, column 3 line 36-9 line 5).

31. As per claim 119, Levin et al teach a computer program wherein the additional input is solicited upon receiving a user-input statement...(see column 6 lines 20-59).

32. As per claim 120-123, Levin et al teach a code segment of soliciting additional input from the user, including presenting: a menu, a textual or an audible request, a list of portions of data source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

33. As per claim 124-125, Levin et al teach a computer program wherein additional input received from the user is speech based, of no spoken input source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

As per claim 126, Levin et al teach a code segment wherein steps (d)-(e) are repeated until the navigational query if deemed adequate source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

Conclusion

34. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. (6,192,338).

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Firmin Backer whose telephone number is 703-305-0624. The examiner can normally be reached on Mon-Thu 8:30-6:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Sheikh Ayaz can be reached on 703-305-9648. The fax phone numbers for the organization where this application or proceeding is assigned are 703-305-3718 for regular communications and 703-305-5352 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 703-305-3900.

Firmin Backer
Firmin Backer
April 9, 2001

FORM PTO-892	U.S. DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE	SERIAL NO.	GROUP ART UNIT	ATTACHMENT TO PAPER NO.
NOTICE OF REFERENCES CITED		09/524,095	2781	10
		APPLICANT(S) HALVERSON ET AL.		

U.S. PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	NAME	CLASS	SUB-CLASS	FILING DATE
	A	6,192,338	2/2001	Zasto et al	704	257	
	B	6,173,279	1/2001	Levin et al.	707	5	
	C						
	D						
	E						
	F						
	G						
	H						
	I						
	J						
	K						

FOREIGN PATENT DOCUMENTS

*		DOCUMENT NO.	DATE	COUNTRY	NAME	CLASS	SUB-CLASS
	L						
	M						
	N						
	O						
	P						
	Q						

OTHER REFERENCES (Including Author, Title, Date, Pertinent Pages, Etc.)

	R	
	S	
	T	
	U	

EXAMINER	DATE	
Firmin Backer	April 9, 2001	Form892ccs2106b

* A copy of this reference is not being furnished with this office action.
(See Manual of Patent Examining Procedure, section 707.05(a).)

2155

CARLTON FIELDS LLP

97 SOUTH SECOND STREET
SUITE 100
SAN JOSE, CALIFORNIA 95113

MAILING ADDRESS:
P.O. BOX 721030, SAN JOSE, CA 95172-1030
TEL (408) 271-2300 FAX (408) 275-9579

#11
LDS
4-27-01

OIPE JC154
APR 16 2001
PATENT & TRADEMARK OFFICE

Writer's Phone Number: (408) 271-2300

April 11, 2001

RECEIVED
APR 19 2001
Technology Center 2100

Assistant Commissioner for Patents
Washington, DC 20231

Re:	Patent Application Serial No.:	09/524,095
	Inventor:	Christine Halverson, et al.
	Title:	Navigating Network-Based Electronic Information Using Spoken Natural Language Input with Multimodal Error Feedback
	Filed:	March 13, 2000
	Our File No.:	44454/02742/SRI1P037/(US4116-2)

Dear Sir:

Please enter the enclosed Revocation and Power of Attorney into the file of the referenced application.

Very truly yours,

Kevin J. Zilka, Reg. No. 41,429

KJZ:ELm
Enclosure
cc: Edward E. Davis, Asst. Secretary (w/ encl.)

CERTIFICATE OF MAILING

I do hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail, postage prepaid, in an envelope addressed to Assistant Commissioner for Patents, Washington, DC 20231, on the date set forth below.

Erica L. Mann — 4/11/2001
Erica L. Mann — Date

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

SRI1P044/44454/02740 (US4015-2)

APPLICATION SERIAL NO.:	09/398,233
INVENTOR:	Douglas E. Appelt, et al.
ASSIGNEE:	SRI International
TITLE:	Information Retrieval by Natural Language Querying
FILING DATE:	September 17, 1999

Attorney Docket No.: SRI1P038/44454/02743 (US4116-4)

APPLICATION SERIAL NO.:	09/524,056
INVENTOR:	Luc Julia et al.
ASSIGNEE:	SRI International
TITLE:	System Method and Article of Manufacture for Navigating Network-Based Electronic Multimedia Content Using Spoken Natural Language Input
FILING DATE:	March 13, 2000

Attorney Docket No.: SRI1P037/44454/02742 (US4116-3)

APPLICATION SERIAL NO.:	09/524,095
INVENTOR:	Christine Halverson
ASSIGNEE:	SRI International
TITLE:	Navigating Network-Based Electronic Information Using Spoken Natural Language Input With Multimodal Error Feedback
FILING DATE:	March 13, 2000

Attorney Docket No.: SRI1P039/44454/02744 (US4116-5)

APPLICATION SERIAL NO.:	09/524,868
INVENTOR:	Luc Julia, et al.
ASSIGNEE:	SRI International
TITLE:	Accessing Network-Based Electronic Information Through Scripted Online Interfaces Using Spoken Natural Language Input
FILING DATE:	March 14, 2000

Attorney Docket No.: SRI1P040/44454/02745 (US4015-3)

APPLICATION SERIAL NO.:	09/613,237
INVENTOR:	James Arnold, et al.
ASSIGNEE:	SRI International
TITLE:	System and Method for Incorporating Concept-Based Retrieval Within Boolean Search Engines
FILING DATE:	July 10, 2000

Attorney Docket No.: SRI1P041/44454/02746 (US4015-4)

APPLICATION SERIAL NO.:	09/613,236
INVENTOR:	James Arnold
ASSIGNEE:	SRI International
TITLE:	System, Method and Article of Manufacture for Interactive Question-Answering and Automated Information Routing
FILING DATE:	July 10, 2000

Attorney Docket No.: SRI1P042/44454/02748 (US4015-5)

APPLICATION SERIAL NO.:	09/613,235
INVENTOR:	James Arnold, et al.
ASSIGNEE:	SRI International
TITLE:	System, Method and Article of Manufacture for Concept Based Information Searching
FILING DATE:	July 10, 2000

Attorney Docket No.: SRI1P043+ (US4148-2P)

APPLICATION SERIAL NO.:	60/228,804
INVENTOR:	Stephen Pullman, et al.
ASSIGNEE:	SRI International
TITLE:	Arbitrary Querying for Information Extraction
FILING DATE:	May 5, 2000

OIPE JC
APR 1 6 2001
PATENT & TRADEMARK OFFICE

REVOCATION AND POWER OF ATTORNEY

Assistant Commissioner for Patents
Washington, DC 20231

The undersigned assignee of the above-referenced patent applications hereby revokes all prior powers of attorney and appoints as his attorney, with full powers of substitution and revocation, to transact all business in the Patent and Trademark Office connected with these applications and any patents resulting therefrom, the following:

Kevin J. Zilka, Reg. No. 41,429
Dominic M. Kotab, Reg. No. 42,762
C. Douglas McDonald, Reg. No. 26,659
John C. Clark, Reg. No. 43,552

RECEIVED
APR 1 9 2001
Technology Center 2100

Please direct all future communications and telephone calls to:

Kevin J. Zilka
CARLTON FIELDS, P.A.
P.O. Box 721030
San Jose, CA 95172-1030
(408)-271-2300

SRI INTERNATIONAL

Date: 09 April 2001

By: Edward E. Davis
Edward E. Davis, Assistant Secretary

#11

UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS
UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. 20231
www.uspto.gov

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
09/524,095	03/13/2000	Christine Halverson	SRI1P037

CONFIRMATION NO. 6294

OC000000006017790

I. KEITH STEPHENS
CARLTON, FIELDS, WARD, EMMANUEL, SMITH & CUTLER
P.O. BOX 3239
TAMPA, FL 33601-3239

Date Mailed: 04/27/2001

NOTICE REGARDING POWER OF ATTORNEY

This is in response to the Power of Attorney filed 04/16/2001.

- The Power of Attorney to you in this application has been revoked by the assignee who has intervened as provided by 37 CFR 3.71. Future correspondence will be mailed to the new address of record(37 CFR 1.33).

Lavinia Johnson

Customer Service Center
Initial Patent Examination Division (703) 308-1202

OFFICE COPY

#12

UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS
UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. 20231
www.uspto.gov

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
09/524,095	03/13/2000	Christine Halverson	SRI1P037

CONFIRMATION NO. 6294

OC000000006017814

KEVIN J. ZILKA
CARLTON FIELDS, P.A.
P.O. BOX 721030
SAN JOSE, CA 95172-1030

Date Mailed: 04/27/2001

NOTICE REGARDING POWER OF ATTORNEY

This is in response to the Power of Attorney filed 04/16/2001.

The Power of Attorney in this application is accepted. Correspondence in this application will be mailed to the above address as provided by 37 CFR 1.33.

Lavinia Johnson

Customer Service Center
Initial Patent Examination Division (703) 308-1202

OFFICE COPY

2758 #13
2154

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

OIPE JC18 APR 3 0 2001 PATENT & TRADEMARK OFFICE

In re the application of:	)	Group Art Unit: 2758
Halverson et al.	)	
Application No. 09/524,095	)	Atty. Docket No. SRI1P037 44454/02742
Filed: 03/13/2000	)	
For: NAVIGATING NETWORK-BASED ELECTRONIC INFORMAITON USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK	)	Date: April 27, 2001

RECEIVED
MAY 4 - 2001
Technology Center 2100

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: Assistant Commissioner for Patents, Washington, DC 20231 on April 27, 2001.

Signed: Erica L. Mann
Erica L. Mann

SUPPLEMENTAL INFORMATION DISCLOSURE STATEMENT
UNDER 37 CFR §§ 1.56 AND 1.97(c)

Assistant Commissioner for Patents
Washington, DC 20231

Dear Sir:

The references listed in the attached PTO Form 1449, copies of which are attached, may be material to examination of the above-identified patent application. Applicants submit these references in compliance with their duty of disclosure pursuant to 37 CFR §§ 1.56 and 1.97. The Examiner is requested to make these references of official record in this application.

This Information Disclosure Statement is not to be construed as a representation that a search has been made, that additional information material to the examination of this application does not exist, or that these references indeed constitute prior art.

This Information Disclosure Statement is believed to be filed before the mailing date of a first Office Action on the merits. Accordingly, it is believed that no fees are due in connection with the filing of this Information Disclosure Statement. However, if it is determined that any fees are due, the Commissioner is hereby authorized to charge such fees to Deposit Account 03-0683 (Order No. 44454/02742/SRI1P037).

OIPE JC139
APR 3 0 2001
PATENT & TRADEMARK OFFICE

Respectfully submitted,
CARLTON FIELDS

Dominic M. Kotab
Reg. No. 42,762

RECEIVED
MAY 4 - 2001
Technology Center 2100

P.O. Box 721030
San Jose, CA 95172-1030
Telephone: (408) 271-2300

Form 1449 (Modified) Information Disclosure Statement By Applicant (Use Several Sheets if Necessary)	Atty. Docket No. SRI1P037 Applicant: Halverson et al. Filing Date: 03/13/2000	Application No.: 09/524,095 Group Art Unit: 2758

OIPE JC169 APR 3 0 2001 PATENT & TRADEMARK OFFICE

U.S. Patent Documents

Examiner Initial	No.	Patent No.	Date	Patentee	Class	Sub-class	Filing Date
	A						
	B						
	C						
	D						
	E						
	F						
	G						
	H						
	I						
	J						
	K						

Foreign Patent or Published Foreign Patent Application

Examiner Initial	No.	Document No.	Publication Date	Country or Patent Office	Class	Sub-class	Translation Yes	Translation No
	L							
	M							
	N							
	O							
	P							

Other Documents

Examiner Initial	No.	Author, Title, Date, Place (e.g. Journal) of Publication
F.D	R	Stent, Amanda et al., "The CommandTalk Spoken Dialogue System", SRI International
	S	Moore, Robert et al., "CommandTalk: A Spoken-Language Interface for Battlefield Simulations", October 23, 1997, SRI International
FD	T	Dowding, John et al., "Interpreting Language in Context in CommandTalk", February 5, 1999, SRI International
Examiner		Date Considered 11/21/02

Examiner: Initial citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

Form 1449 (Modified) **Information Disclosure Statement By Applicant** (Use Several Sheets if Necessary)	Atty. Docket No. SRI1P037 Applicant: Halverson et al. Filing Date: 03/13/2000	Application No.: 09/524,095 Group Art Unit: 2758

OIPE JC18 APR 3 0 2001 PATENT & TRADEMARK OFFICE

U.S. Patent Documents

Examiner Initial	No.	Patent No.	Date	Patentee	Class	Sub-class	Filing Date
	A						
	B						
	C						
	D						
	E						
	F						
	G						
	H						
	I						
	J						
	K						

RECEIVED
MAY 4 - 2001
Technology Center 2100

Foreign Patent or Published Foreign Patent Application

Examiner Initial	No.	Document No.	Publication Date	Country or Patent Office	Class	Sub-class	Translation Yes	Translation No
	L							
	M							
	N							
	O							
	P							

Other Documents

Examiner Initial	No.	Author, Title, Date, Place (e.g. Journal) of Publication
F.B	R	http://www.ai.sri.com/~oaa/infowiz.html, "InfoWiz: An Animated Voice Interactive Information System, May 8, 2000
	S	Dowding, John, "Interleaving Syntax and Semantics in an Efficient Bottom-up Parser", SRI International
F.B	T	Moore, Robert et al., "Combining Linguistic and Statistical Knowledge Sources in Natural-Language Processing for ATIS", SRI International

Examiner	Date Considered 11/21/02

Examiner: Initial citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

Form 1449 (Modified) **Information Disclosure Statement By Applicant** (Use Several Sheets if Necessary)	Atty. Docket No. SRI1P037 Applicant: Halverson et al. Filing Date: 03/13/2000	Application No.: 09/524,095 Group Art Unit: 2758

OIPE JC189 APR 30 2001 PATENT & TRADEMARK OFFICE

U.S. Patent Documents

Examiner Initial	No.	Patent No.	Date	Patentee	Class	Sub-class	Filing Date
	A						
	B						
	C						
	D						
	E						
	F						
	G						
	H						
	I						
	J						
	K						

RECEIVED MAY 4 - 2001 Technology Center 2100

Foreign Patent or Published Foreign Patent Application

Examiner Initial	No.	Document No.	Publication Date	Country or Patent Office	Class	Sub-class	Translation Yes	Translation No
	L							
	M							
	N							
	O							
	P							

Other Documents

Examiner Initial	No.	Author, Title, Date, Place (e.g. Journal) of Publication
F.B	R	Dowding, John et al., "Gemini: A Natural Language System For Spoken-Language Understanding", SRI International
	S	
	T	

Examiner [signature]	Date Considered 1/27/02

Examiner: Initial citation considered. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

OIPE JCIS8B
JUN 1 1 2001
PATENT & TRADEMARK OFFICE

2155 2100

#14
6-20-01

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION NO.: 09/524,095
INVENTOR: Halversen, Christine
TITLE: NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN INPUT WITH MULTIMODAL ERROR FEEDBACK

FILING DATE: 3/13/00
ATTORNEY DOCKET NO. SRI1P037

NOTICE OF CHANGE OF CORRESPONDENCE ADDRESS

RECEIVED
JUN 1 9 2001
Technology Center 2100

Assistant Commissioner for Patents
Washington, DC 20231

Sir:

Please change the correspondence address relating to the above-identified application as follows:

C. Douglas McDonald , Esq.
Carlton Fields, et al.
P.O. Box 3239
Tampa, FL 33601-3239

Respectfully submitted,

C. Douglas McDonald

Date: May 10, 2001

C. Douglas McDonald
Reg. No. 26,659
CARLTON FIELDS, P.A.
P.O. Box 3239
Tampa, FL 33601-3239
(813) 223-7000
Attorney of Record

Under the Paperwork Reduction Act o… U.S. Patent and Trademark Offic… …. DEPARTMENT OF COMMERCE
no persons are required to respond to a collection of information unless it displays a valid OMB control number.

#15
LDJ
9-26-01

PETITION FOR EXTENSION OF TIME UNDER 37 CFR 1.136(a)	Docket Number (Optional) SRI 1P037

OIPE
SEP 2 1 2001
JC95
PATENT & TRADEMARK OFFICE

In re Application of HALVERSON, et al	
Application Number 09/524,095	Filed March 13, 2000
For Navigating Network-Based Electronic Information Using Spoken Input With Multimodal Error Feedback	
Group Art Unit 2155	Examiner F. Backer

This is a request under the provisions of 37 CFR 1.136(a) to extend the period for filing a response in the above identified application.

The requested extension and appropriate non-small-entity fee are as follows (check time period desired):

- ☐ One month (37 CFR 1.17(a)(1)) $
- ☒ Two months (37 CFR 1.17(a)(2)) $390.00
- ☐ Three months (37 CFR 1.17(a)(3)) $
- ☐ Four months (37 CFR 1.17(a)(4)) $
- ☐ Five months (37 CFR 1.17(a)(5)) $

☒ Applicant claims small entity status. See 37 CFR 1.27. Therefore, the fee amount shown above is reduced by one-half, and the resulting fee is: $ 195.00 .

☒ A check in the amount of the fee is enclosed.

☐ Payment by credit card. Form PTO-2038 is attached.

☐ The Commissioner has already been authorized to charge fees in this application to a Deposit Account.

☒ The Commissioner is hereby authorized to charge any fees which may be required, or credit any overpayment, to Deposit Account Number 20-0782 .
I have enclosed a duplicate copy of this sheet.

RECEIVED
SEP 2 5 2001
Technology Center 2100

I am the ☐ applicant/inventor.

☐ assignee of record of the entire interest. See 37 CFR 3.71
Statement under 37 CFR 3.73(b) is enclosed. (Form PTO/SB/96).

☒ attorney or agent of record.

☐ attorney or agent under 37 CFR 1.34(a).
Registration number if acting under 37 CFR 1.34(a). ______ .

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

September 19, 2001
Date

Signature

KIN-WAH TONG, Reg. No. 39,400
Typed or printed name

09/25/2001 MWOLDER1 00000026 09524095
01 FC:216 195.00 OP

NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below*.

☐ *Total of ____ forms are submitted.

OIPE
SEP 2 1 2001
JC95
PATENT & TRADEMARK OFFICE

SRI/4116-3

#16
LDS
9-26-01

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

PATENT APPLICATION

Applicant(s):	**HALVERSON, et al**	Atty. Docket No.	**SRI 1P037**
Serial No.:	**09/524,095**	Group Art Unit:	**2155**
Filed:	**March 13, 2000**	Examiner:	**F. BACKER**
Title:	**NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN INPUT WITH MULTIMODAL ERROR FEEDBACK**		

RECEIVED
SEP 2 5 2001
Technology Center 2100

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

REVOCATION OF PREVIOUS POWER OF ATTORNEY AND NEW APPOINTMENT

The undersigned assignee of the above-identified application hereby revokes all previous Powers of Attorney and appoints the following attorneys with full power to prosecute the application, to make alterations and amendments therein, and to transact all business in the United States Patent and Trademark Office connected therewith and with full power of substitution and revocation:

Raymond R. Moser, Jr.; Reg. No. 34,682; Kin-Wah Tong, Reg. No. 39,400; Robert Brush, Reg. No. 45,710; Steven Weiner, Reg. No. 38,360; and Edward E. Davis, Reg. No. 35,112.

CHANGE OF CORRESPONDENCE ADDRESS

Please change the correspondence address for the above-identified application to:

Thomason, Moser & Patterson, LLP
595 Shrewsbury Avenue – Suite 100
Shrewsbury, New Jersey 07702

Please direct all telephone calls to: Kin-Wah Tong, telephone # (732) 530-9404

OIPE
SEP 2 1 2001
PATENT & TRADEMARK OFFICE

SRI/4116-3

RECEIVED
SEP 25 2001
Technology Center 2100

CERTIFICATE UNDER 37 C.F.R. § 3.73(B)

SRI International, a corporation of the State of California, certifies that it is the assignee of the entire right, title and interest in the patent application identified above by virtue of:

An Assignment from the inventor(s) of the patent application identified above. The Assignment was recorded in the United States Patent and Trademark Office, for which a copy thereof is attached.

The undersigned (whose title is supplied below) is empowered to act on behalf of the assignee.

Respectfully submitted,

Date: 9/11/01

~~EDWARD E. DAVIS, Assistant Secretary~~
STEVEN LOEINER, VICE-PRESIDENT

SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
Telephone No.: 650-859-3115

ASSIGNMENT OF PATENT APPLICATION

(Not Accompanying Application)

Whereas I/we the undersigned inventor(s) have invented certain new and useful improvements as set forth in the patent application entitled:

NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK

for which I/we have executed an application for a United States Letters Patent which was filed in the U.S. Patent and Trademark Office on March 13, 2000, and which bears the Application No. 09/524,095.

For good and valuable consideration, the receipt and sufficiency of which is hereby acknowledged, I/we the undersigned inventor(s) hereby:

1) Sell(s), assign(s) and transfer(s) to **SRI International**, a California non-profit corporation having a place of business at 333 Ravenswood Avenue, Menlo Park, California 94025, (hereinafter referred to as "ASSIGNEE"), the entire right title and interest in any and all improvements and inventions disclosed in, application(s) based upon, and Patent(s) (including foreign patents) granted upon the information which is disclosed in the above referenced application.

2) Authorize and request the Commissioner of Patents to issue any and all Letters Patents resulting from said application or any division(s), continuation(s), substitutes(s) or reissue(s) thereof to the ASSIGNEE.

3) Agree to execute all papers and documents and, entirely at the ASSIGNEE's expense, perform any acts which are reasonably necessary in connection with the prosecution of said application, as well as any derivative and applications thereof, foreign applications based thereon, and/or the enforcement of patents resulting from such applications.

4) Agree that the terms, covenants and conditions of this assignment shall inure to the benefit of the Assignee, its successors, assigns and other legal representative, and shall be binding upon the inventor(s), as well as the inventor's heirs, legal representatives and assigns.

5) Warrant and represent that I/we have not entered, and will not enter into any assignment, contract, or understanding that conflicts with this assignment.

Signed on the date(s) indicated beside my (our) signature(s).

1) Signature: Christine Halverson Date: 6-16-00.
Typed Name: Christine Halverson

2) Signature: ____________ Date: ________
Typed Name: Luc Julia

3) Signature: Dimitris Voutsas Date: 6/16/00
Typed Name: Dimitris Voutsas

4) Signature: Adam J. Cheyer Date: 6/22/00
Typed Name: Adam Cheyer

ASSIGI IENT OF PATENT APPLICA ,ON

(Not Accompanying Application)

Whereas I/we the undersigned inventor(s) have invented certain new and useful improvements as set forth in the patent application entitled:

NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK

for which I/we have executed an application for a United States Letters Patent which was filed in the U.S. Patent and Trademark Office on March 13, 2000, and which bears the Application No. 09/524,095.

For good and valuable consideration, the receipt and sufficiency of which is hereby acknowledged, I/we the undersigned inventor(s) hereby:

1) Sell(s), assign(s) and transfer(s) to **SRI International**, a California non-profit corporation having a place of business at 333 Ravenswood Avenue, Menlo Park, California 94025, (hereinafter referred to as "ASSIGNEE"), the entire right title and interest in any and all improvements and inventions disclosed in, application(s) based upon, and Patent(s) (including foreign patents) granted upon the information which is disclosed in the above referenced application.

2) Authorize and request the Commissioner of Patents to issue any and all Letters Patents resulting from said application or any division(s), continuation(s), substitutes(s) or reissue(s) thereof to the ASSIGNEE.

3) Agree to execute all papers and documents and, entirely at the ASSIGNEE's expense, perform any acts which are reasonably necessary in connection with the prosecution of said application, as well as any derivative and applications thereof, foreign applications based thereon, and/or the enforcement of patents resulting from such applications.

4) Agree that the terms, covenants and conditions of this assignment shall inure to the benefit of the Assignee, its successors, assigns and other legal representative, and shall be binding upon the inventor(s), as well as the inventor's heirs, legal representatives and assigns.

5) Warrant and represent that I/we have not entered, and will not enter into any assignment, contract, or understanding that conflicts with this assignment.

Signed on the date(s) indicated beside my (our) signature(s).

1) Signature: Christine Halverson Date: 6-16-00.
Typed Name: Christine Halverson

2) Signature: Date: 6·20.00
Typed Name: Luc Julia

3) Signature: Date: 6/16/00
Typed Name: Dimitris Voutsas

4) Signature: Date:
Typed Name: Adam Cheyer

ASSIGNMENT OF PATENT APPLICATION

(Not Accompanying Application)

Whereas I/we the undersigned inventor(s) have invented certain new and useful improvements as set forth in the patent application entitled:

NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK

for which I/we have executed an application for a United States Letters Patent which was filed in the U.S. Patent and Trademark Office on March 13, 2000, and which bears the Application No. 09/524,095.

For good and valuable consideration, the receipt and sufficiency of which is hereby acknowledged, I/we the undersigned inventor(s) hereby:

1) Sell(s), assign(s) and transfer(s) to **SRI International**, a California non-profit corporation having a place of business at 333 Ravenswood Avenue, Menlo Park, California 94025, (hereinafter referred to as "ASSIGNEE"), the entire right title and interest in any and all improvements and inventions disclosed in, application(s) based upon, and Patent(s) (including foreign patents) granted upon the information which is disclosed in the above referenced application.

2) Authorize and request the Commissioner of Patents to issue any and all Letters Patents resulting from said application or any division(s), continuation(s), substitutes(s) or reissue(s) thereof to the ASSIGNEE.

3) Agree to execute all papers and documents and, entirely at the ASSIGNEE's expense, perform any acts which are reasonably necessary in connection with the prosecution of said application, as well as any derivative and applications thereof, foreign applications based thereon, and/or the enforcement of patents resulting from such applications.

4) Agree that the terms, covenants and conditions of this assignment shall inure to the benefit of the Assignee, its successors, assigns and other legal representative, and shall be binding upon the inventor(s), as well as the inventor's heirs, legal representatives and assigns.

5) Warrant and represent that I/we have not entered, and will not enter into any assignment, contract, or understanding that conflicts with this assignment.

Signed on the date(s) indicated beside my (our) signature(s).

1) Signature: Christine Halverson Date: 6-16-00.
Typed Name: Christine Halverson

2) Signature: ________ Date: ________
Typed Name: Luc Julia

3) Signature: [signature] Date: 6/16/00
Typed Name: Dimitris Voutsas

4) Signature: ________ Date: ________
Typed Name: Adam Cheyer

#16

UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS
UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. 20231
www.uspto.gov

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
09/524,095	03/13/2000	Christine Halverson	SRI1P037

CONFIRMATION NO. 6294

OC000000006797094

C. Douglas McDonald, ESQ.
CARLTON FIELDS, et al.
P.O. Box 3239
Tampa, FL 33601-3239

Date Mailed: 09/26/2001

NOTICE REGARDING POWER OF ATTORNEY

This is in response to the Power of Attorney filed 09/21/2001.

- The Power of Attorney to you in this application has been revoked by the assignee who has intervened as provided by 37 CFR 3.71. Future correspondence will be mailed to the new address of record(37 CFR 1.33).

Lavinia Johnson

LAVINIA D JOHNSON
2100 7033085229

OFFICE COPY

#17

UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS
UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. 20231
www.uspto.gov

APPLICATION NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
09/524,095	03/13/2000	Christine Halverson	SRI1P037

CONFIRMATION NO. 6294

OC000000006797149

THOMASON, MOSER & PATTERSON, LLP
595 SHREWSBURY AVENUE
SUITE 100
SHREWSBURY, NJ 07702

Date Mailed: 09/26/2001

NOTICE REGARDING POWER OF ATTORNEY

This is in response to the Power of Attorney filed 09/21/2001.

The Power of Attorney in this application is accepted. Correspondence in this application will be mailed to the above address as provided by 37 CFR 1.33.

Lavinia Johnson

LAVINIA D JOHNSON
2100 7033085229

OFFICE COPY

09/524,095

OIPE SEP 2 1 2001 JC95 PATENT & TRADEMARK OFFICE

#18
LDO
9-26-01

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

RECEIVED
SEP 25 2001
Technology Center 2100

PATENT APPLICATION

Applicant: **Halverson et al.**

Case: **SRI1P037**

Serial No.: **09/524,095** Filed: **March 13, 2000**

Group Art Unit: **2155**

Examiner: **Firmin Backer**

Title: **NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK**

ASSISTANT COMMISSIONER FOR PATENTS
Box Non-Fee Amendment
Washington, D. C. 20231

S I R:

RESPONSE UNDER 37 C.F.R. § 1.111

This response addresses the Office Action dated April 24, 2001 (Paper No. 10).

REMARKS

In view of the following discussion, the Applicants submit that none of the claims now pending in the application are anticipated under the provisions of 35 U.S.C. § 102. Thus, the Applicants believe that all of these claims are now in allowable form.

I. REJECTION OF CLAIMS 56-126 UNDER 35 U.S.C. § 102

The Examiner has rejected claims 56-126 in Paragraphs 2-33 of the Office Action as being anticipated by the Levin et al. patent (US Patent 6,173,279 issued January 9, 2001, hereinafter referred to as Levin). The rejection is respectfully traversed.

Levin teaches "a method of using at least one natural language query to retrieve information from one or more data resources and further performing a requested action using the retrieved information is disclosed". (See Levin, Column 2, lines 15-18) Namely, Levin teaches a method for using natural language query to obtain information, where upon receipt of the requested information, a desired action is executed based upon the requested information. To illustrate, Levin provides the example, where a user employs natural language to request the telephone number of a restaurant. Upon receipt of the telephone number, the telephone number is actually dialed for the user. (See Levin, Column 3 line 62 to Column 4, line 1)

In contrast, Levin fails to teach or suggest the novel concept of speech-based navigation where the method solicits additional input from the user, including user interaction in a modality different than the original request. Specifically, Applicants' independent claims 56, 82 and 101 positively recite:

56. 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 modality different than the original request;
(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. (emphasis added)

82. A system 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, the system comprising:

(a) a portable microphone operable to receive a spoken request for desired information from the user;
(b) language processing logic, operable to render an interpretation of the spoken request;
(c) query construction logic, operable to construct a navigation query

in response to the interpretation of the spoken request;

(d) user interaction logic, operable to solicit additional input from the user, including user interaction in a modality different than the original request;

(e) query refining logic, operable to refine the navigation query, based upon the additional input;

(f) navigation logic, operable to select a portion of the electronic data source using the navigation query; and

(g) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user. (emphasis added)

101. A computer program embodied on a computer readable medium 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:

(a) a code segment that receives a spoken request for desired information from the user;

(b) a code segment that renders an interpretation of the spoken request;

(c) a code segment that constructs at least part of a navigation query based upon the interpretation;

(d)a code segment that solicits additional input from the user, including user interaction in a modality different than the original request;

(e) a code segment that refines the navigation query, based upon the additional input;

(f) a code segment that uses the refined navigation query to select a portion of the electronic data source; and

(g) a code segment that transmits the selected portions of the electronic data source from the network server to a primarily stationary, display device located locally with the user. (emphasis added)

Applicants' invention teaches a novel method and apparatus for speech-based navigation where the method solicits additional input from the user, including user interaction **in a modality different than the original request**. Specifically, Applicants address the criticality of errors and deficiencies via user interface modalities in addition to spoken natural language. It has been observed that users are often frustrated by ineffective or non optimal speech-based navigation that simply engages the user repeatedly in a long series of questions and answers, i.e., "single modal interaction", to perfect the navigation query. This single modal approach is often tedious and uninspiring for a user who must refine the navigation query repeatedly to achieve the desired result, thereby increasing the time the user must interact with a system. In fact, one goal of the speech-based navigation is to relieve this very tedium where the user must engage a system repeatedly, e.g., via a long sequence of menus to achieve the desired result.

To address this criticality, Applicants' navigation query can be refined via input from the user, where the user interaction is in a modality different than the original request. To illustrate, if a portion of the navigation query can be achieved, then the result can be presented to the user in a way that the user can provide additional input via interaction that is in a modality that is different than the original request. For example, if the "partial" navigation query produces three possible results, then the results can be presented to the user via a menu with the most likely result being highlighted. The user can then press a button on a remote unit to accept the highlighted result or simply scroll to one of the other three choices. Thus, the pressing of the button by the user is a user interaction that is in a different modality than the original request, e.g., a natural language request that originally started the navigation request. This is an important aspect of the invention because of the psychological and real effect where the user perceives that the navigation query is actually progressing closer to the achieved result.

In contrast, Levin teaches that "the service host 112 determines if there are any ambiguities with respect to the response (step 222) and, if so, forwards **additional queries** to the user to help to resolve the ambiguities (step 224)". (emphasis added) (See Levin, Column 6, lines 40-43). Additionally, Levin states that "[t]he service host 112 includes a dialog control program that manages interactions with users **over several turns (e.g., it decides when to ask a question, when to give an answer**, provides means for clarifying ambiguities, and provides error control and recovery during an interaction)". (emphasis added) (See Levin, Column 5, lines 15-20). Levin's single modal approach is contrary to Applicants' invention and is one of the criticalities that Applicants' invention is designed to address. To further support Applicants' position, Levin states that "[t]he invention is independent of the actual modality of call placement". (See Levin, Column 4, lines 29-31) This statement is another clear indication that Levin is totally unconcerned with the modality of the user interaction and is simply teaching a single modal approach via queries and answers.

Therefore, the Applicants respectfully submit that independent claims 56, 82 and 101 are not anticipated by the Levin reference. As such, claims 56, 82 and 101 fully satisfy the requirements of 35 U.S.C. §102 and are patentable thereunder.

Claims 57-81, 83-100 and 102-126 depend, either directly or indirectly, from claims 56, 82 and 101 and recite additional features therefor. Since Levin fails to anticipate Applicants' invention as recited in Applicants' independent claims 56, 82 and 101, dependent claims 57-81, 83-100 and 102-126 are also not anticipated under 35 U.S.C. § 102 and are allowable for the same reason noted above.

Conclusion

Thus, the Applicants submit that all of these claims now fully satisfy the requirements of 35 U.S.C. §102. Consequently, the Applicants believe that all these claims are presently in condition for allowance. Accordingly, both reconsideration of this application and its swift passage to issue are earnestly solicited.

If, however, the Examiner believes that there are any unresolved issues requiring the issuance of a final action in any of the claims now pending in the application, it is requested that the Examiner telephone Mr. Kin-Wah Tong, Esq. at (732) 530-9404 so that appropriate arrangements can be made for resolving such issues as expeditiously as possible.

Respectfully submitted,

9/19/01

Kin-Wah Tong, Attorney
Reg. No. 39,400
(732) 530-9404

Moser, Patterson & Sheridan, LLP
595 Shrewsbury Avenue
First Floor,
Shrewsbury, New Jersey 07702

GAU 2155

PTO/SB/21 (08-00)

Please type a plus sign (+) inside this box → [+]

Approved for use through 10/31/2002. OMB 0651-0031

U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

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.

OIPE SEP 21 2001 JC95 PATENT & TRADEMARK OFFICE

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	09/524,095
Filing Date	March 13, 2000
First Named Inventor	HALVERSON
Group Art Unit	2155
Examiner Name	F. BACKER
Attorney Docket Number	SRI 1 P 037
Total Number of Pages in This Submission	

ENCLOSURES *(check all that apply)*

- [x] Fee Transmittal Form
  - [x] Fee Attached
- [x] Amendment / Response
  - [ ] After Final
  - [ ] Affidavits/declaration(s)
- [x] Extension of Time Request
- [ ] Express Abandonment Request
- [ ] Information Disclosure Statement
- [ ] Certified Copy of Priority Document(s)
- [ ] Response to Missing Parts/ Incomplete Application
  - [ ] Response to Missing Parts under 37 CFR 1.52 or 1.53
- [ ] Assignment Papers *(for an Application)*
- [ ] Drawing(s)
- [ ] Licensing-related Papers
- [ ] Petition
- [ ] Petition to Convert to a Provisional Application
- [x] Power of Attorney, Revocation Change of Correspondence Address
- [ ] Terminal Disclaimer
- [ ] Request for Refund
- [ ] CD, Number of CD(s)
- [ ] After Allowance Communication to Group
- [ ] Appeal Communication to Board of Appeals and Interferences
- [ ] Appeal Communication to Group *(Appeal Notice, Brief, Reply Brief)*
- [ ] Proprietary Information
- [ ] Status Letter
- [ ] Other Enclosure(s) *(please identify below)*:

Remarks

RECEIVED SEP 25 2001 Technology Center 2100

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm or Individual name	KIN-WAH TONG
Signature	K-Wah Tong
Date	September 19, 2001

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: Assistant Commissioner for Patents, Washington, D.C. 20231 on this date: September 19, 2001

Typed or printed name	Linda DeNardi		
Signature	Linda DeNardi	Date	September 19, 2001

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be send to the Chief Information Officer, U.S. Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Washington, DC 20231.

FEE TRANSMITTAL for FY 2001

Patent fees are subject to annual revision.

DURING PENDENCY, PLEASE CHARGE DEPOSIT ACCOUNT 20-0782 FOR ANY 37 C.F.R. 1.16 AND/OR 37 C.F.R. 1.17 FEES DUE AND NOT OTHERWISE AUTHORIZED. PLEASE CREDIT DEPOSIT ACCOUNT 20-0782 FOR ANY OVERPAYMENTS

TOTAL AMOUNT OF PAYMENT ($) 195.00

Complete If Known	
Application Number	09/524,095
Filing Date	March 13, 2000
First Named Inventor	HALVERSON
Examiner Name	F. BACKER
Group / Art Unit	2155
Attorney Docket No.	SRI 1P037

METHOD OF PAYMENT (check one)

1. ☒ The Commissioner is hereby authorized to charge indicated fees and credit any over payments to:

Deposit Account Number: 20-0782

Deposit Account Name:

☒ Charge Any Additional Fee Required Under 37 CFR 1.16 and 1.17

☒ Applicant claims small entity status. See 37 CFR 1.27

2. ☒ Payment Enclosed:

☒ Check ☐ Credit card ☐ Money Order ☐ Other

FEE CALCULATION

1. BASIC FILING FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
101	710	201	355	Utility filing fee	
106	320	206	160	Design filing fee	
107	490	207	245	Plant filing fee	
108	710	208	355	Reissue filing fee	
114	150	214	75	Provisional filling fee	

SUBTOTAL (1) ($) 0

2. EXTRA CLAIM FEES

			Extra Claims		Fee from below		Fee Paid
Total Claims		-20** =	0	X		=	0
Independent Claims		-3** =	0	X		=	0
Multiple Dependent				X		=	0

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description
103	18	203	9	Claims in excess of 20
102	80	202	40	Independent claims in excess of 3
104	270	204	135	Multiple dependent claim, if not paid
109	80	209	40	** Reissue independent claims over original patent
110	18	210	9	** Reissue claims in excess of 20 and over original patent

SUBTOTAL (2) ($) 0

**or number previously paid, if greater; For Reissues, see above

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
105	130	205	65	Surcharge - late filing fee or oath	
127	50	227	25	Surcharge - late provisional filing fee or cover sheet.	
139	130	139	130	Non-English specification	
147	2,520	147	2,520	For filing a request for reexamination	
112	920*	112	920*	Requesting publication of SIR prior to Examiner action	
113	1,840*	113	1,840*	Requesting publication of SIR after Examiner action	
115	110	215	55	Extension for reply within first month	
116	390	216	195	Extension for reply within second month	195.00
117	890	217	445	Extension for reply within third month	
118	1,390	218	695	Extension for reply within fourth month	
128	1,890	228	945	Extension for reply within fifth month	
119	310	219	155	Notice of Appeal	
120	310	220	155	Filing a brief in support of an appeal	
121	270	221	135	Request for oral hearing	
138	1,510	138	1,510	Petition to institute a public use proceeding	
140	110	240	55	Petition to revive – unavoidable	
141	1,240	241	620	Petition to revive – unintentional	
142	1,240	242	620	Utility issue fee (or reissue)	
143	440	243	220	Design issue fee	
144	800	244	300	Plant issue fee	
122	130	122	130	Petitions to the Commissioner	
123	130	123	130	Petitions related to provisional applications	
126	180	126	180	Submission of Information Disclosure Stmt	
581	40	581	40	Recording each patent assignment per property (times number of properties)	
146	710	246	355	Filing a submission after final rejection (37 CFR § 1.129(a))	
149	710	249	355	For each additional invention to be examined (37 CFR § 1.129(b))	
179	710	279	355	Request for Continued Examination (RCE)	
169	900	169	900	Request for expedited examination of a design application	

Other fee (specify)

*Reduced by Basic Filing Fee Paid SUBTOTAL (3) ($) 195.00

SUBMITTED BY				Complete *(if applicable)*	
Name (Print/Type)	KIN-WAH TONG	*Registration No. Attorney/Agent)*	39,400	*Telephone*	(732) 530-9404
Signature	K. W. Tong			*Date*	SEPTEMBER 19, 2001

SRJ/4116-3

#19
LDJ
10-9-01

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

PATENT APPLICATION

Applicant(s):	**HALVERSON, et al**	Atty. Docket No.	**SRI 1P037**
Serial No.:	**09/524,095**	Group Art Unit:	**2155**
Filed:	**March 13, 2000**	Examiner:	**F. BACKER**
Title:	**NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN INPUT WITH MULTIMODAL ERROR FEEDBACK**		

REQUEST FOR CORRECTED FILING RECEIPT

RECEIVED
OCT 0 5 2001
Technology Center 2100

Assistant Commissioner for Patents
Office of Initial Patent Examination
Customer Service Center
Washington, D. C. 20231

SIR:

Please issue a corrected filing receipt reflecting the correct spelling of the first name of the fourth inventor as follows:

Adam J. Cheyer

Respectfully submitted

Date 9/28/01

KIN-WAH TONG, Attorney
Reg. No. 39,400

Moser, Patterson & Sheridan, LLP
595 Shrewsbury Avenue – Suite 100
Shrewsbury, New Jersey 07702
(732)530-9404

SRI/4116-3

CERTIFICATE OF FACSIMILE TRANSMISSION

Under 37 C.F.R.§ 1.8(a)

I certify that this correspondence is being transmitted by facsimile (703-308-7751) under 37 C.F.R. 1.8(a) on September 28, 2001 and is addressed to the Assistant Commissioner for Patents, Office of Initial Patent Examination, Customer Service Center, Washington, D.C. 20231.

Linda DeNardi
Type or print name of person signing certification

Linda DeNardi
Signature

UNITED STATES PATENT AND TRADEMARK OFFICE

COMMISSIONER FOR PATENTS
UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. 20231
www.uspto.gov

APPLICATION NUMBER	FILING DATE	GRP ART UNIT	FIL FEE REC'D	ATTY.DOCKET.NO	DRAWINGS	TOT CLAIMS	IND CLAIMS
09/524,095	03/13/2000	2758	1529	SRI1P037	7	55	3

FILING RECEIPT

OC000000005388294

Hickman Stephens Coleman & Hughes LLP
PO Box 52037
Palo Alto, CA 94303-0746

Date Mailed: 09/11/2000

Receipt is acknowledged of this nonprovisional Patent Application. It will be considered in its order and you will be notified as to the results of the examination. Be sure to provide the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION when inquiring about this application. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. **If an error is noted on this Filing Receipt, please write to the Office of Initial Patent Examination's Customer Service Center. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts" for this application, please submit any corrections to this Filing Receipt with your reply to the Notice. When the PTO processes the reply to the Notice, the PTO will generate another Filing Receipt incorporating the requested corrections (if appropriate).**

Applicant(s)

Christine Halverson, San Jose, CA ;
Luc Julia, Menlo Park, CA ;
Dimitris Voutsas, Thessaloniki, GREECE;
Aden J. Cheyer, Palo Alto, CA ;
ADAM

Continuing Data as Claimed by Applicant

THIS APPLICATION IS A CIP OF 09/225,198 01/05/1999
WHICH CLAIMS BENEFIT OF 60/124,718 03/17/1999
WHICH CLAIMS BENEFIT OF 60/124,719 03/17/1999
WHICH CLAIMS BENEFIT OF 60/124,720 03/17/1999

Foreign Applications

If Required, Foreign Filing License Granted 05/12/2000

**** SMALL ENTITY ****

RECD SEP 18 2000

Title

Navigating network-based electronic information using spoken natural language input with multimodal error feedback

Preliminary Class

709

b

UNITED STATES DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

b

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.
09/524,095	03/13/00	HALVERSON C	SRI1P037

TM02/1010

THOMASON, MOSER & PATTERSON, LLP
595 SHREWSBURY AVENUE
SUITE 100
SHREWSBURY NJ 07702

EXAMINER	
BACKER, F	
ART UNIT	PAPER NUMBER
2155	# 20

DATE MAILED: 10/10/01

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

Commissioner of Patents and Trademarks

PTO-90C (Rev. 2/95)

File Copy

Office Action Summary	Application No.	Applicant(s)
	09/524,095	HALVERSON ET AL.
	Examiner	Art Unit
	Firmin Backer	2155

-- The MAILING DATE of this communication appears on the cover sheet with the 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 *21 September 2001* .

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) *56-126* 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) *56-126* is/are rejected.

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

8)☐ Claims _____ 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 objected to by the Examiner.

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. § 119

13)☐ 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.

14)☐ Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

Attachment(s)

15)☐ Notice of References Cited (PTO-892)

16)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

17)☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .

18)☐ Interview Summary (PTO-413) Paper No(s). _____ .

19)☐ Notice of Informal Patent Application (PTO-152)

20)☐ Other:

Response to Request for Reconsideration

This is in response to a request for reconsideration file on September 26th, 2001. Claims 56-126 are being reconsidered in this action.

Claim Rejections - 35 USC § 102

1. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371(c) of this title before the invention thereof by the applicant for patent.

2. Claims 56-126 are rejected under 35 U.S.C. 102(e) as being anticipated by Levin et al. (U.S. Patent No. 6,173,279).

3. As per claim 56, Levin et al teach a method for speech-based navigation (information server, 110) of an electronic data source located at one or more network servers located remotely from a user, (see abstract, fig 1, column 3 lines 5-35), comprising receiving a spoken request (*receive a natural language query*) for desired information from the user (user); rendering an interpretation (*creating a semantic representation*) of the spoken request, constructing a navigation (*generating search)* query based upon the interpretation; soliciting additional input from the user (*one or more questions are generated...*), including user interaction in a modality different that the original request and, refining the navigation query, based upon the additional input (see column 6 lines 20-59), using the 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 primarily stationary, display device located locally with the user. (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22)

4. As per claim 57, Levin et al teach a method of rendering the interpretation includes deriving linguistic information by using a speech recognition and a linguistic parser (see abstract, fig 1, column 3 lines 37-5 lines 40).

5. As per claim 58-62, Levin et al teach a method of constructing a navigation query in the form of a database query on a computing device located on a network including extracting an input template for an online scripted interface to the data source to be used for the construction of the navigation query and dynamically scraping the online scripted interface (see abstract, fig. 1-3, column 3 line 36-9 line 5)

6. As per claim 63-68, Levin et al teach a method of soliciting additional input is performed in response deficiency including unresolved word encountered after the first navigation of the data source, required element of the navigational query, data recorded within the data source, failure to identify data record responsive to navigational query (see column 6 lines 20-59).

7. As per claim 69, Levin et al teach a method wherein the additional input is solicited upon receiving a user-input statement…(see column 6 lines 20-59).

8. As per claim 70-73, Levin et al teach a method of soliciting additional input from the user, including presenting: a menu, a textual or an audible request, a list of portions of data source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

9. As per claim 74-75, Levin et al teach a method wherein additional input received from the user is speech based, of no spoken input source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

10. As per claim 76, Levin et al teach a method wherein steps (d)-(e) are repeated until the navigational query if deemed adequate source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

11. As per claim 77, 78, Levin et al teach a method wherein the input modality includes selecting (by speaking) from a displayed option menu (see abstract, fig. 1-3, column 3 line 36-9 line 5).

12. As per claim 79, Levin et al teach a method performed with respect to a plurality of user and corresponding client devices (see abstract, fig. 1-3, column 3 line 36-9 line 5).

13. As per claim 80-81, Levin et al teach a method of selecting data source from plurality of electronic data source storing multimedia content including audio and video content (see abstract, fig. 1-3, column 3 line 36-9 line 5)

14. As per claim 82, Levin et al teach a system for speech-based navigation (*information server, 110*) of an electronic data source located at one or more network servers located remotely from a user, (see abstract, fig 1, column 3 lines 5-35), comprising a portable microphone (*microphone, 105*) receiving a spoken request (*receive a natural language query*) for desired information from the user (user) a language processing logic (*natural language server, 114*) rendering an interpretation (*creating a semantic representation*) of the spoken request, (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22) a query construction logic (*service host, 112*) constructing a navigation (*generating search)* query based upon the interpretation; a query interaction logic (*service host, 112)* soliciting additional input from the user (*one or more questions are generated...*), including user interaction in a modality different that the original request and, (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22), a query refining logic (*service host, 112)* refining the navigation query, based upon the additional input (see column 6 lines 20-59), a navigation logic (*service host, 112)* using the navigation query to select a portion of the electronic data source; electronic infrastructure (*network, 108*) transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user. (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22).

15. As per claim 83, Levin et al teach a system of rendering the interpretation includes deriving linguistic information by using a speech recognition and a linguistic parser (see abstract, fig 1, column 3 lines 37-5 lines 40).

16. As per claim 84-86, Levin et al teach a system of constructing a navigation query in the form of a database query on a computing device located on a network including extracting an input template for an online scripted interface to the data source to be used for the construction of the navigation query and dynamically scraping the online scripted interface (see abstract, fig. 1-3, column 3 line 36-9 line 5).

17. As per claim 87, 88, 100, Levin et al teach a system wherein at least a portion of the language processing if hosted on a computing device coupled with a microphone located locally with a user and a network computing device located remotely and data in a two-way communication infrastructure (coaxial, DSL, satellite, wireless/cellular, fiber-optic) (see abstract, fig. 1-3, column 3 line 36-9 line 5).

18. As per claim 89-94, Levin et al teach a system of soliciting additional input is performed in response deficiency including unresolved word encountered after the first navigation of the data source, required element of the navigational query, data recorded within the data source, failure to identify data record responsive to navigational query (see column 6 lines 20-59).

19. As per claim 95, 96, Levin et al teach a system wherein the input modality includes selecting (by speaking) from a displayed option menu (see abstract, fig. 1-3, column 3 line 36-9 line 5).

20. As per claim 97-98, Levin et al teach a system of selecting data source from plurality of electronic data source storing multimedia content including audio and video content (see abstract, fig. 1-3, column 3 line 36-9 line 5).

21. As per claim 99, Levin et al teach a system wherein the display device receives data from the electronic device on the network via a communication box (see abstract, fig. 1-3, column 3 line 36-9 line 5).

22. As per claim 101, Levin et al teach a computer program for speech-based navigation (information server, 110) of an electronic data source located at one or more network servers located remotely from a user, (see abstract, fig 1, column 3 lines 5-35), comprising code segment receiving a spoken request (*receive a natural language query*) for desired information from the user (user); code segment rendering an interpretation (*creating a semantic representation*) of the spoken request, code segment constructing a navigation (*generating search)* query based upon the interpretation; soliciting additional input from the user (*one or more questions are generated...*), including user interaction in a modality different that the original request and, code segment refining the navigation query, based upon the additional input (see column 6 lines 20-59); code segment using the navigation query to select a portion of the electronic data source; and code segment transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user (see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22).

23. As per claim 102, Levin et al teach a code segment deriving linguistic information by using a speech recognition and a linguistic parser (see abstract, fig 1, column 3 lines 37-5 lines 40).

24. As per claim 103-105, Levin et al teach a code segment of constructing a navigation query in the form of a database query on a computing device located on a network including extracting an input template for an online scripted interface to the data source to be used for the construction of the navigation query and dynamically scraping the online scripted interface (see abstract, fig. 1-3, column 3 line 36-9 line 5).

25. As per claim 106-107, Levin et al teach a computer program wherein rendering of the interpretation and the construction of the navigation query are performed on a computing device located locally with or remotely from the user (see abstract, fig. 1-3, column 3 line 36-9 line 5).

26. As per claim 108-114, Levin et al teach a code segment that solicits additional input display on option menu is performed by speaking in response deficiency including unresolved word encountered after the first navigation of the data source, required element of the navigational query, data recorded within the data source, failure to identify data record responsive to navigational query (see column 6 lines 20-59).

27. As per claim 115, Levin et al teach a computer program the act of selecting from the display is performed by speaking (see column 6 lines 20-59)

28. As per claim 116, Levin et al teach a code segment of the computer program operate with respect to a plurality of simultaneous user and corresponding client devices (see abstract, fig. 1-3, column 3 line 36-9 line 5).

29. As per claim 117, Levin et al teach a code segment that select data source form a plurality of electronic data source .... content (see abstract, fig. 1-3, column 3 line 36-9 line 5).

30. As per claim 118, Levin et al teach a computer program of selecting data source from plurality of electronic data source storing multimedia content including audio and video content (see abstract, fig. 1-3, column 3 line 36-9 line 5).

31. As per claim 119, Levin et al teach a computer program wherein the additional input is solicited upon receiving a user-input statement...(see column 6 lines 20-59).

32. As per claim 120-123, Levin et al teach a code segment of soliciting additional input from the user, including presenting: a menu, a textual or an audible request, a list of portions of data source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

33. As per claim 124-125, Levin et al teach a computer program wherein additional input received from the user is speech based, of no spoken input source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

34. As per claim 126, Levin et al teach a code segment wherein steps (d)-(e) are repeated until the navigational query if deemed adequate source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

Response to Arguments

Applicant's arguments filed on September 26th, 2001 have been fully considered but they are not persuasive.

Applicant argues that the prior art (Levin et al) fail to teach or suggest an inventive concept wherein "soliciting additional input from the user including user interaction in a modality different than the original request." Examiner respectfully disagrees with the applicant's perspective and characterization of Levin's inventive concept. Levin et al teach a system and method of using natural language to retrieve information. In that particular if the service host 112, based on the rules, decides that there is enough information for performing a database access, the database query is generated. The database query is generally in one of the standard query languages (e.g. SQL). The service host 112 also determines if there are any ambiguities with respect to the response (step 222) and, if so, forwards additional queries to the user to help to resolve the ambiguities (step 224). The service host 112 then sends the responses to the information server 110 (step 226). If there are too many potential answers (for instance if there are two pizza places on Main Street in Westfield), one or more questions to the user are generated in order to disambiguate the query (e.g. Do you mean "Venezia" or "Bella Roma?").

The answers to the additional questions are used to formulate a new logical search query. For this there might be additional rules like: if(Action_Object=Pizza_Restaurant and Too-Many_Answers) then User must provide further clarifying information such as, for example, the name of restaurant OR exact address. If the user does not provide enough information to achieve a single answer, the service host 112 might then list the possibilities and ask the user to chose one of them (*see column 6 lines 28-59*). This is a way to require additional information from the user in order to generate user's request.

Conclusion

THIS ACTION IS MADE FINAL. 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 mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Firmin Backer whose telephone number is 703-305-0624. The examiner can normally be reached on Mon-Thu 8:30-6:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Sheikh Ayaz can be reached on 703-305-9648. The fax phone numbers for the organization where this application or proceeding is assigned are 703-305-3718 for regular communications and 703-305-5352 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 703-305-3900.

Firmin Backer
October 2, 2001

AYAZ SHEIKH
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/524,095	03/13/2000	Christine Halverson	SRI1P037	6294

25696 7590 01/15/2002

OPPENHEIMER WOLFF & DONNELLY
P. O. BOX 10356
PALO ALTO, CA 94303

EXAMINER	
BACKER, FIRMIN	

ART UNIT	PAPER NUMBER
2155	21

DATE MAILED: 01/15/2002

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

PTO-90C (Rev. 07-01)

Interview Summary	**Application No.** 09/524,095	**Applicant(s)** HALVERSON ET AL.	
	Examiner Firmin Backer	**Art Unit** 2155	

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

(1) *Firmin Backer (examiner)*. (3) *Kin-Wah Tong (Attorney)*.

(2) *Ario Etienne (primary examiner)*. (4)_____.

Date of Interview: *08 January 2002* .

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: *56* .

Identification of prior art discussed: *6,173,279* .

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: *Applicant argues that the statutory double patenting rejection is improper and should be withdrawn. Applicant argues that the prior art fails to teach all the limitations of the inventive concept especially the concept of transmitting the selected portion of the electronic data source from the network server to a client device of the user* .

(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.)

i)☐ It is not necessary for applicant to provide a separate record of the substance of the interview(if box is checked).

Unless the paragraph above has been checked, 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.

Examiner's signature, if required

22
BT 1/17/02
Not Entered

09/524,095

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

PATENT APPLICATION

Applicant: **Halverson et al.**

Case: **SRI1P037**

Serial No.: **09/524,095** Filed: **March 13, 2000**

Group Art Unit: **2155**

Examiner: **Firmin Backer**

Title: **NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK**

ASSISTANT COMMISSIONER FOR PATENTS
Box AF
Washington, D. C. 20231

S I R:

RESPONSE UNDER 37 C.F.R. § 1.116

This response addresses the Final Office Action dated October 10, 2001. The Final Office Action appears to be labeled as Paper No. 20.

REMARKS

Applicants' representative would like to thank Examiner Backer and Primary Examiner Etienne for kindly taking a substantial amount of time on January 8, 2002 to discuss the merits of the subject invention. Applicants' representative is aware of the time constraint that is placed on the Examiners and is appreciative of the Examiners' willingness to devote such large quantity of time to discuss the case on the merit.

In view of the following discussion, the Applicants submit that none of the claims now pending in the application are anticipated under the provisions of 35 U.S.C. § 102. Thus, the Applicants believe that all of these claims are now in allowable form.

09/524,095

I. REJECTION OF CLAIMS 56-126 UNDER 35 U.S.C. § 102

The Examiner has rejected claims 56-126 in Paragraphs 2-34 of the Final Office Action as being anticipated by the Levin et al. patent (US Patent 6,173,279 issued January 9, 2001, hereinafter referred to as Levin). The rejection is respectfully traversed.

Levin teaches "a method of using at least one natural language query to retrieve information from one or more data resources and further performing a requested action using the retrieved information is disclosed". (See Levin, Column 2, lines 15-18) Namely, Levin teaches a method for using natural language query to obtain information, where upon receipt of the requested information, a desired action is executed based upon the requested information. To illustrate, Levin provides the example, where a user employs natural language to request the telephone number of a restaurant. Upon receipt of the telephone number, the telephone number is actually dialed for the user. (See Levin, Column 3 line 62 to Column 4, line 1)

In contrast, Levin fails to teach or suggest the novel concept of speech-based navigation where the method solicits additional input from the user, including user interaction in a modality different than the original request. Specifically, Applicants' independent claims 56, 82 and 101 positively recite:

> 56. 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 modality different than the original request;
> (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. (emphasis added)

82. A system 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, the system comprising:

(a) a portable microphone operable to receive a spoken request for desired information from the user;

(b) language processing logic, operable to render an interpretation of the spoken request;

(c) query construction logic, operable to construct a navigation query in response to the interpretation of the spoken request;

(d) user interaction logic, operable to solicit additional input from the user, including user interaction in a modality different than the original request;

(e) query refining logic, operable to refine the navigation query, based upon the additional input;

(f) navigation logic, operable to select a portion of the electronic data source using the navigation query; and

(g) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user. (emphasis added)

101. A computer program embodied on a computer readable medium 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:

(a) a code segment that receives a spoken request for desired information from the user;

(b) a code segment that renders an interpretation of the spoken request;

(c) a code segment that constructs at least part of a navigation query based upon the interpretation;

(d)a code segment that solicits additional input from the user, including user interaction in a modality different than the original request;

(e) a code segment that refines the navigation query, based upon the additional input;

(f) a code segment that uses the refined navigation query to select a portion of the electronic data source; and

(g) a code segment that transmits the selected portions of the electronic data source from the network server to a primarily stationary, display device located locally with the user. (emphasis added)

Pursuant to the Examiner Interview, Applicants directed the Examiner's attention to the fact that Applicants' invention teaches a novel method and apparatus for speech-based navigation where the method solicits additional input from the user, including user interaction **in a modality different than the original request**. Specifically, Applicants address the criticality of errors and deficiencies via user interface modalities in addition to spoken natural language. It has been observed that users are often frustrated by ineffective or non optimal speech-based navigation that simply engages the user repeatedly in a long series of questions and answers, i.e., "single modal interaction", to perfect the navigation query. This single modal approach is often tedious and uninspiring for a user who must refine the navigation query repeatedly to achieve the desired result, thereby increasing the time the user must interact with a system. In fact, one goal of the speech-based navigation is to relieve this very tedium where the user must engage a system repeatedly, e.g., via a long sequence of menus to achieve the desired result.

To address this criticality, Applicants' navigation query can be refined via input from the user, where the user interaction is in a modality different than the original request. To illustrate, if a portion of the navigation query can be achieved, then the result can be presented to the user in a way that the user can provide additional input via interaction that is in a modality that is different than the original request. For example, if the "partial" navigation query produces three possible results, then the results can be presented to the user via a menu with the most likely result being highlighted. The user can then press a button on a remote unit to accept the highlighted result or simply scroll to one of the other three choices. Thus, the pressing of the button by the user is a **user interaction that is in a different modality than the original request, e.g., a natural language request that originally started the navigation request**. This is an important aspect of the invention because of the psychological and real effect where the user perceives that the navigation query is actually progressing closer to the achieved result.

In contrast, Levin teaches that "the service host 112 determines if there are any ambiguities with respect to the response (step 222) and, if so, forwards **additional queries** to the user to help to resolve the ambiguities (step 224)". (emphasis added) (See Levin, Column 6, lines 40-43). Additionally, Levin states that "[t]he service host 112 includes a dialog control program that manages interactions with users **over several turns (e.g., it decides when to ask a question, when to give an answer,** provides means for clarifying ambiguities, and provides error control and recovery during an interaction)". (emphasis added) (See Levin, Column 5, lines 15-20). Levin's single modal approach is contrary to Applicants' invention and is one of the criticalities that Applicants' invention is designed to address. To further support Applicants' position, Levin states that "[t]he invention is independent of the actual modality of call placement". (See Levin, Column 4, lines 29-31) This statement is another clear indication that Levin is totally unconcerned with the modality of the user interaction and is simply teaching a single modal approach via queries and answers.

As discussed during the Examiner Interview, the support cited by the Examiner in the Final Office Action only discloses the teaching that the user is requested to provide additional information, but it does not require the user to provide the additional inputs in **a different modality than the original request** as claimed by the Applicants. The Examiners indicated that they would reconsider the present rejections.

Therefore, the Applicants respectfully submit that independent claims 56, 82 and 101 are not anticipated by the Levin reference. As such, claims 56, 82 and 101 fully satisfy the requirements of 35 U.S.C. §102 and are patentable thereunder.

Claims 57-81, 83-100 and 102-126 depend, either directly or indirectly, from claims 56, 82 and 101 and recite additional features therefor. Since Levin fails to anticipate Applicants' invention as recited in Applicants' independent claims 56, 82 and 101, dependent claims 57-81, 83-100 and 102-126 are also not anticipated under 35 U.S.C. § 102 and are allowable for the same reason noted above.

Conclusion

Thus, the Applicants submit that all of these claims now fully satisfy the requirements of 35 U.S.C. §102. Consequently, the Applicants believe that all these claims are presently in condition for allowance. Accordingly, both reconsideration of this application and its swift passage to issue are earnestly solicited.

If, however, the Examiner believes that there are any unresolved issues requiring the maintenance of the present final office action in any of the claims now pending in the application, it is requested that the Examiner telephone Mr. Kin-Wah Tong, Esq. at

(732) 530-9404 so that appropriate arrangements can be made for resolving such issues as expeditiously as possible.

Respectfully submitted,

1/10/02

Kin-Wah Tong, Attorney
Reg. No. 39,400
(732) 530-9404

Moser, Patterson & Sheridan, LLP
595 Shrewsbury Avenue
First Floor,
Shrewsbury, New Jersey 07702

TELEFAX COVER SHEET

MOSER, PATTERSON & SHERIDAN, LLP
ATTORNEYS AT LAW
595 SHREWSBURY AVENUE
FIRST FLOOR
SHREWSBURY, NJ 07702
TELEPHONE (732) 530-9404
TELEFAX (732) 530-9808

Official

RECEIVED 1/10/02

****

THIS TELEFAX MESSAGE IS ADDRESSED TO THE PERSON OR COMPANY LISTED BELOW. IF IT WAS SENT OR RECEIVED INCORRECTLY, OR YOU ARE NOT THE INTENDED RECIPIENT, PLEASE TAKE NOTICE THAT THIS MESSAGE MAY CONTAIN PRIVILEGED OR CONFIDENTIAL MATERIAL, AND YOUR DUE REGARD FOR THIS INFORMATION IS NECESSARY. YOU MAY ARRANGE TO RETURN THIS MATERIAL BY CALLING THE FIRM LISTED ABOVE AT (732) 530-9404

****

THIS MESSAGE HAS 9 PAGES INCLUDING THIS SHEET

TO: Assistant Commissioner of Patents

FAX NO.: 703-746-7238

FROM: Kin-Wah Tong

DATE: January 10, 2002

MATTER: Serial No. 09/524,095 Filed: March 13, 2000

DOCKET NO.: SRI 1P037

APPLICANT: HALVERSON, et al

The following has been received in the U.S. Patent and Trademark Office on the date of this facsimile:

___ Petition
___ Disclosure Statement & PTO-1449
___ Priority Document
___ Drawings (___ sheets) informal
X Response Under 37 CFR 1.116

X Transmittal Letter (2 copies)
___ Fee Transmittal (2 copies)
___ Deposit Account Transaction
X Facsimile Transmission Certificate dated January 10, 2002

CERTIFICATE OF TRANSMISSION UNDER 37 C.F.R. §1.6

I hereby certify that this correspondence is being transmitted by facsimile to the Assistant Commissioner for Patents, Box AF, Washington, DC 20231 on January 10, 2002,
Facsimile No. 703-746-7238

Linda DeNardi
Name of person signing this certificate

Linda DeNardi January 10, 2002
Signature and date

PTO/SB/21 (08-00)
Approved for use through 10/31/2002. OMB 0651-0031
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Please type a plus sign (+) inside this box → +

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.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	09/524,095
Filing Date	March 13, 2000
First Named Inventor	HALVERSON
Group Art Unit	2155
Examiner Name	F. BACKER
Attorney Docket Number	SRI 1 P 037

Total Number of Pages In This Submission: 9

ENCLOSURES *(check all that apply)*

- ☐ Fee Transmittal Form
  - ☐ Fee Attached
- ☒ Amendment / Response
  - ☐ After Final
  - ☐ Affidavits/declaration(s)
- ☐ Extension of Time Request
- ☐ Express Abandonment Request
- ☐ Information Disclosure Statement
- ☐ Certified Copy of Priority Document(s)
- ☐ Response to Missing Parts/ Incomplete Application
  - ☐ Response to Missing Parts under 37 CFR 1.52 or 1.53
- ☐ Assignment Papers *(for an Application)*
- ☐ Drawing(s)
- ☐ Licensing-related Papers
- ☐ Petition
- ☐ Petition to Convert to a Provisional Application
- ☐ Power of Attorney, Revocation Change of Correspondence Address
- ☐ Terminal Disclaimer
- ☐ Request for Refund
- ☐ CD, Number of CD(s)
- ☐ After Allowance Communication to Group
- ☐ Appeal Communication to Board of Appeals and Interferences
- ☐ Appeal Communication to Group *(Appeal Notice, Brief, Reply Brief)*
- ☐ Proprietary Information
- ☐ Status Letter
- ☐ Other Enclosure(s) *(please identify below)*:

Remarks: **It is believed no fee is due. However, in the event a fee is due, kindly charge that fee to deposit account number 20-0782. To facilitate that charge, a duplicate copy of this letter is enclosed**

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm or Individual name	PATRICIA A. VERLANGIERI, Reg. No. 42,201
Signature	Patricia A. Verlangieri
Date	January 10, 2002

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be send to the Chief Information Officer, U.S. Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Washington, DC 20231.

PTO/SB/21 (08-00)
Approved for use through 10/31/2002. OMB 0651-0031
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Please type a plus sign (+) inside this box → [+]

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.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	09/524,095
Filing Date	March 13, 2000
First Named Inventor	HALVERSON
Group Art Unit	2155
Examiner Name	F. BACKER
Attorney Docket Number	SRI 1 P 037
Total Number of Pages in This Submission	9

ENCLOSURES *(check all that apply)*

- ☐ Fee Transmittal Form
  - ☐ Fee Attached
- ☒ Amendment / Response
  - ☐ After Final
  - ☐ Affidavits/declaration(s)
- ☐ Extension of Time Request
- ☐ Express Abandonment Request
- ☐ Information Disclosure Statement
- ☐ Certified Copy of Priority Document(s)
- ☐ Response to Missing Parts/ Incomplete Application
  - ☐ Response to Missing Parts under 37 CFR 1.52 or 1.53
- ☐ Assignment Papers *(for an Application)*
- ☐ Drawing(s)
- ☐ Licensing-related Papers
- ☐ Petition
- ☐ Petition to Convert to a Provisional Application
- ☐ Power of Attorney, Revocation Change of Correspondence Address
- ☐ Terminal Disclaimer
- ☐ Request for Refund
- ☐ CD, Number of CD(s)
- ☐ After Allowance Communication to Group
- ☐ Appeal Communication to Board of Appeals and Interferences
- ☐ Appeal Communication to Group *(Appeal Notice, Brief, Reply Brief)*
- ☐ Proprietary Information
- ☐ Status Letter
- ☐ Other Enclosure(s) *(please identify below)*:

Remarks	It is believed no fee is due. However, in the event a fee is due, kindly charge that fee to deposit account number 20-0782. To facilitate that charge, a duplicate copy of this letter is enclosed

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm or Individual name	PATRICIA A. VERLANGIERI, Reg. No. 42,201
Signature	Patricia A. Verlangieri
Date	January 10, 2002

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be send to the Chief Information Officer, U.S. Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Washington, DC 20231.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/524,095	03/13/2000	Christine Halverson	SRI1P037	6294

25696 7590 02/19/2002

OPPENHEIMER WOLFF & DONNELLY
P. O. BOX 10356
PALO ALTO, CA 94303

EXAMINER	
BACKER, FIRMIN	

ART UNIT	PAPER NUMBER
2155	23

DATE MAILED: 02/19/2002

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

PTO-90C (Rev. 07-01)

Advisory Action	**Application No.**	**Applicant(s)**	
	09/524,095	HALVERSON ET AL.	
	Examiner	**Art Unit**	
	Firmin Backer	2155	

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

THE REPLY FILED 07 January 2002 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: _____.

3.☐ Applicant's reply has overcome the following rejection(s): _____.

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: 56-126

Claim(s) withdrawn from consideration: _____.

8.☐ The proposed 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 Sheet (PTO-303) **Application No.**
09/524,095

Continuation of 5. does NOT place the application in condition for allowance because: Applicant request for reconsideration has been considered but does not place the application in condition for allowance. Applicant argues that Levin fail to teach the limitation of soliciting additional input from the user, including user interaction in a modality different than the original request. Examiner respectfully disagree with applicant characterization of Levin et al' inventive concept. As examiner has indicated before, Levin et al teach a system and method of using natural language to retrieve information. In that particular if the service host 112, based on the rules, decides that there is enough information for performing a database access, the database query is generated. The database query is generally in one of the standard query languages (e.g. SQL). The service host 112 also determines if there are any ambiguities with respect to the response (step 222) and, if so, forwards additional queries to the user to help to resolve the ambiguities (step 224). The service host 112 then sends the responses to the information server 110 (step 226). If there are too many potential answers (for instance if there are two pizza places on Main Street in Westfield), one or more questions to the user are generated in order to disambiguate the query (e.g. Do you mean "Venezia" or "Bella Roma?"). The answers to the additional questions are used to formulate a new logical search query. For this there might be additional rules like: if(Action_Object=Pizza_Restaurant and Too-Many_Answers) then User must provide further clarifying information such as, for example, the name of restaurant OR exact address. If the user does not provide enough information to achieve a single answer, the service host 112 might then list the possibilities and ask the user to choose one of them (see column 6 lines 28-59). Levin cleary indicate that in the user does not provide enough information to achieve a sinlge answer then the service host might the list the possibilites and ask the user to chose on of them. To the examiner that is a different modality then the original mode. It can be seen that in the original mode, the user was requesting the service. In this mode, the host provides a list of service for the user to choose from. In the original mode, the user did not have any choices, however, in this mode the user has a list to choose from. Therefore, the final action is sustained.

AYAZ SHEIKH
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

8

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/524,095	03/13/2000	Christine Halverson	SRI1P037	6294

7590 04/03/2002

THOMASON, MOSER & PATTERSON, LLP
595 SHREWSBURY AVENUE
SUITE 100
SHREWSBURY, NJ 07702

EXAMINER	
BACKER, FIRMIN	
ART UNIT	**PAPER NUMBER**
2161	

DATE MAILED: 04/03/2002 #24

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

PTO-90C (Rev. 07-01)

SUPPLEMENTAL **Advisory Action**	**Application No.** *09/524,095*	**Applicant(s)** HALVERSON ET AL.	
	Examiner Firmin Backer	**Art Unit** 2155	

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

THE REPLY FILED 07 January 2002 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: _____.

3.☐ Applicant's reply has overcome the following rejection(s): _____.

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: *56-126*.

Claim(s) withdrawn from consideration: _____.

8.☐ The proposed 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: _____

JAMES P. TRAMMELL
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Continuation Sheet (PTO-303) **Application No.**
09/524,095

Continuation of 5. does NOT place the application in condition for allowance because: Applicant request for reconsideration has been considered but does not place the application in condition for allowance. Applicant argues that Levin fail to teach the limitation of soliciting additional input from the user, including user interaction in a modality different than the original request. Examiner respectfully disagree with applicant characterization of Levin et al' inventive concept. As examiner has indicated before, Levin et al teach a system and method of using natural language to retrieve information. In that particular if the service host 112, based on the rules, decides that there is enough information for performing a database access, the database query is generated. The database query is generally in one of the standard query languages (e.g. SQL). The service host 112 also determines if there are any ambiguities with respect to the response (step 222) and, if so, forwards additional queries to the user to help to resolve the ambiguities (step 224). The service host 112 then sends the responses to the information server 110 (step 226). If there are too many potential answers (for instance if there are two pizza places on Main Street in Westfield), one or more questions to the user are generated in order to disambiguate the query (e.g. Do you mean "Venezia" or "Bella Roma?"). The answers to the additional questions are used to formulate a new logical search query. For this there might be additional rules like: if(Action_Object=Pizza_Restaurant and Too-Many_Answers) then User must provide further clarifying information such as, for example, the name of restaurant OR exact address. If the user does not provide enough information to achieve a single answer, the service host 112 might then list the possibilities and ask the user to choose one of them (see column 6 lines 28-59). Levin cleary indicate that in the user does not provide enough information to achieve a sinlge answer then the service host might the list the possibilites and ask the user to chose on of them. To the examiner that is a different modality then the original mode. It can be seen that in the original mode, the user was requesting the service. In this mode, the host provides a list of service for the user to choose from. In the original mode, the user did not have any choices, however, in this mode the user has a list to choose from. Therefore, the final action is sustained.

PTO/SB/22 (10-00)
Approved for through 10/31/2002. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
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.

PETITION FOR EXTENSION OF TIME UNDER 37 CFR 1.136(a)	Docket Number (Optional) SRI 1P037

OIPE JC98 APR 1 0 2002 PATENT & TRADEMARK OFFICE

In re Application of HALVERSON	
Application Number 09/524,095	Filed March 13, 2000
For Navigating Network-Based Electronic Information Using Spoken Natural Language Input With Multimodal Error Feedback	
Group Art Unit 2155	Examiner F. Backer

#25
LDJ
4-16-02

This is a request under the provisions of 37 CFR 1.136(a) to extend the period for filing a response in the above identified application.

The requested extension and appropriate non-small-entity fee are as follows (check time period desired):

- ☐ One month (37 CFR 1.17(a)(1)) $
- ☒ Two months (37 CFR 1.17(a)(2)) $400.00
- ☐ Three months (37 CFR 1.17(a)(3)) $
- ☐ Four months (37 CFR 1.17(a)(4)) $
- ☐ Five months (37 CFR 1.17(a)(5)) $

☒ Applicant claims small entity status. See 37 CFR 1.27. Therefore, the fee amount shown above is reduced by one-half, and the resulting fee is: $ 200.00 .

☐ A check in the amount of the fee is enclosed.

☐ Payment by credit card. Form PTO-2038 is attached.

☐ The Commissioner has already been authorized to charge fees in this application to a Deposit Account.

☒ The Commissioner is hereby authorized to charge any fees which may be required, or credit any overpayment, to Deposit Account Number 20-0782 .
I have enclosed a duplicate copy of this sheet.

RECEIVED
APR 1 2 2002
Technology Center 2100

I am the ☐ applicant/inventor.

☐ assignee of record of the entire interest. See 37 CFR 3.71
Statement under 37 CFR 3.73(b) is enclosed. (Form PTO/SB/96).

☒ attorney or agent of record.

☐ attorney or agent under 37 CFR 1.34(a).
Registration number if acting under 37 CFR 1.34(a). ______ .

04/17/2002 LJOHNSON 00000001 200782 09524095
01 FC:217 260.00 CH

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

April 10, 2002
Date

Signature

KIN-WAH TONG
Typed or printed name

04/12/2002 AWONDAF1 00000141 200782 09524095
02 FC:216 200.00 CH

NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below*.

☐ *Total of ____ forms are submitted.

Burden Hour Statement: This form is estimated to take 0.1 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Washington, DC 20231.

04-11-02

RCE / 2700

OIPE JC96 APR 1 0 2002 PATENT & TRADEMARK OFFICE

PTO/SB/30 (8/2000)
Approved for use through 10/31/2002 OMB 0651-0031
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

#26 LDJ 4-16-0

REQUEST FOR CONTINUED EXAMINATION (RCE) TRANSMITTAL

Subsection (b) of 35 U.S.C. § 132, effective on May 29, 2000, provides for continued examination of a utility or plant application filed on or after June 8, 1995, See the American Inventors Protection Act of 1999 (AIPA).

Application Number	09/524,095
Filing Date	March 13, 2000
First Named Inventor	HALVERSON
Group Art Unit	2155
Examiner Name	F. Backer
Attorney Docket Number	SRI 1P037

This is a Request for Continued Examination (RCE) under 37 C.F.R. § 1.114 of the above-identified application.
NOTE: 37 C.F.R. § 1.114 is effective on May 29, 2000. If the above-identified application was filed prior to May 29, 2000, applicant may wish to consider filing a continued prosecution application (CPA) under 37 C.F.R. § 1.53 (d) (PTO/SB/29) instead of an RCE to be eligible for the patent term adjustment provisions of the AIPA. See Changes to Application Examination and Provisional Application Practice, Final Rule, 65 Fed. Reg. 50092 (Aug. 16, 2000); Interim Rule, 65 Fed. Reg. 14865 (Mar. 20, 2000), 1233 Off. Gaz. Pat. Office 47 (Apr. 11, 2000), which established RCE Practice.

1. Submission required under 37 C.F.R. § 1.114.
   a. ☐ Previously submitted
      i. ☐ Consider the amendment(s)/reply under 37 C.F.R. § 1.116 previously filed on ____
      ii. ☐ Consider the arguments in the Appeal Brief or Reply Brief previously filed on ____
      iii. ☐ Other ____
   b. ☒ Enclosed
      i. ☐ Amendment/Reply
      ii. ☐ Affidavit(s)/Declaration(s)
      iii. ☐ Information Disclosure Statement (IDS)
      iv. ☒ Other **Preliminary Amendment**

RECEIVED
APR 1 2 2002
Technology Center 2100

2. Miscellaneous
   a. ☐ Suspension of action on the above-identified application is requested under 37 C.F.R. § 1.103(c) for a period of ____ months (Period of suspension shall not exceed 3 months; Fee under 37 C.F.R.§ 1.17(i) required)
   b. ☐ Other
3. Fees The RCE fee under 37 C.F.R. § 1.17(e) is required by 37 C.F.R. § 1.114 when the RCE is filed.
   a. ☒ The Director is hereby authorized to charge the following fees, or credit any overpayments, to Deposit Account No. **20-0782**
      i. ☒ RCE fee required under 37 C.F.R. § 1.17(e)
      ii. ☒ Extension of time fee (37 C.F.R. §§ 1.136 and 1.17)
      iii. ☐ Other ____
   b. ☐ Check in the amount of $ ____ enclosed
   c. ☐ Payment by credit card (Form PTO-2038 enclosed)

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT REQUIRED

Name (Print / Type)	KIN-WAH TONG	*Registration No. (Attorney / Agent)*	39,400
Signature	K. W. Tong	*Date*	April 10, 2002

04/12/2002 AWONDAF1 00000141 200782 09524095
01 FC:179 370.00 CH

09/524,095

OIPE JC95
APR 1 0 2002
PATENT & TRADEMARK OFFICE

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

PATENT APPLICATION

RECEIVED
APR 1 2 2002
Technology Center 2100

Applicant: **Halverson et al.**

Case: **SRI1P037**

Serial No.: **09/524,095** Filed: **March 13, 2000**

Group Art Unit: **2155**

Examiner: **Firmin Backer**

Title: **NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK**

ASSISTANT COMMISSIONER FOR PATENTS
Box RCE
Washington, D. C. 20231

S I R:

Please be advised that the enclosed RCE and Preliminary Amendment are filed with a two-month extension request instead of a three-month extension request. The reason is that the Advisory Action dated February 19, 2002 was erroneously forwarded to a different law firm by the USPTO. This error was communicated to Examiner Backer and the Examiner subsequently issued a supplemental Advisory Action to the Applicants' representative on April 3, 2002. As such, Applicants have informed the Examiner that the enclosed RCE and Preliminary Amendment will be filed with a two-month extension request instead of a three-month extension request.

However, in the event that a three-month extension request is required, Applicants' representative hereby requests for a three-month extension request and authorizes the payment of the necessary extension fee via **Deposit Account: 20-0782**.

09/524,095

OIPE JC95
APR 1 0 2002
PATENT & TRADEMARK OFFICE

4/10/02

Respectfully submitted,

Kin-Wah Tong, Attorney
Reg. No. 39,400
(732) 530-9404

Moser, Patterson & Sheridan, LLP
595 Shrewsbury Avenue
First Floor,
Shrewsbury, New Jersey 07702

OIPE JCS
APR 1 0 2002
PATENT & TRADEMARK OFFICE

EXPRESS MAIL CERTIFICATION

"Express Mail" mailing label number EL 849341069 US

Date of deposit APRIL 10, 2002

I hereby certify that this paper and/or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to Assistant Commissioner of Patents, BOX RCE, Washington, D.C. 20231.

Linda DeNardi

Signature of person mailing paper or fee

Linda DeNardi

Name of person mailing paper or fee

RECEIVED

APR 1 2 2002

Technology Center 2100

09/524,095

OIPE APR 1 0 2002 JC96 PATENT & TRADEMARK OFFICE

#27/RC
LDJ
4-16-02
entered

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

PATENT APPLICATION

RECEIVED
APR 1 2 2002
Technology Center 2100

Applicant: **Halverson et al.**

Case: **SRI1P037**

Serial No.: **09/524,095** Filed: **March 13, 2000**

Group Art Unit: **2155**

Examiner: **Firmin Backer**

Title: **NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK**

ASSISTANT COMMISSIONER FOR PATENTS
Box RCE
Washington, D. C. 20231

S I R:

Preliminary Amendment

This Preliminary Amendment is filed in conjunction with an RCE and addresses the Advisory Action dated April 3, 2002.

IN THE CLAIMS

Please amend claims 56, 82, and 101 as shown below. The claims are "clean version" of the amended claims, i.e., with changes incorporated into the claims, whereas the Appendix to this Amendment illustrates the amended claims using underlines and brackets to indicate addition and deletion, respectively.

56. (Amended) 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:

C

09/524,095

(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;

C1 end

(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.

---

82. (Amended) A system 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, the system comprising:

(a) a portable microphone operable to receive a spoken request for desired information from the user;

(b) language processing logic, operable to render an interpretation of the spoken request;

(c) query construction logic, operable to construct a navigation query in response to the interpretation of the spoken request;

C2

(d) user interaction logic, operable to solicit additional input from the user, including user interaction in a non-spoken modality different than the original request;

(e) query refining logic, operable to refine the navigation query, based upon the additional input;

(f) navigation logic, operable to select a portion of the electronic data source using the navigation query; and

(g) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

---

101. A computer program embodied on a computer readable medium 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:

(a) a code segment that receives a spoken request for desired information from the user;

(b) a code segment that renders an interpretation of the spoken request;

(c) a code segment that constructs at least part of a navigation query based upon the interpretation;

(d) a code segment that solicits additional input from the user, including user interaction in a non-spoken modality different than the original request;

(e) a code segment that refines the navigation query, based upon the additional input;

(f) a code segment that uses the refined navigation query to select a portion of the electronic data source; and

(g) a code segment that transmits the selected portions of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

C3

REMARKS

In view of the above Amendment and the following discussion, the Applicants submit that none of the claims now pending in the application are anticipated under the provisions of 35 U.S.C. § 102. Thus, the Applicants believe that all of these claims are now in allowable form.

I. REJECTION OF CLAIMS 56-126 UNDER 35 U.S.C. § 102

The Examiner has rejected claims 56-126 in Paragraphs 2-34 of the Final Office Action and in the Advisory Action as being anticipated by the Levin et al. patent (US Patent 6,173,279 issued January 9, 2001, hereinafter referred to as Levin). The rejection is respectfully traversed.

Levin teaches "a method of using at least one natural language query to retrieve information from one or more data resources and further performing a requested action using the retrieved information is disclosed". (See Levin, Column 2, lines 15-18) Namely, Levin teaches a method for using natural language query to obtain information, where upon receipt of the requested information, a desired action is executed based upon the requested information. To illustrate, Levin provides the example, where a user employs natural language to request the telephone number of a restaurant. Upon receipt of the telephone number, the telephone number is actually dialed for the user. (See Levin, Column 3 line 62 to Column 4, line 1)

In contrast, Levin fails to teach or suggest the novel concept of speech-based navigation where the method solicits additional input from the user, including user interaction in a non-spoken modality different than the original request. Specifically, Applicants' amended independent claims 56, 82 and 101 positively recite:

> 56. 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;
> (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. (emphasis added)
>
> 82. A system 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, the system comprising:
> (a) a portable microphone operable to receive a spoken request for desired information from the user;
> (b) language processing logic, operable to render an interpretation of the spoken request;
> (c) query construction logic, operable to construct a navigation query in response to the interpretation of the spoken request;

(d) <u>user interaction logic, operable to solicit additional input from the user, including user interaction in a non-spoken modality different than the original request</u>;
(e) query refining logic, operable to refine the navigation query, based upon the additional input;
(f) navigation logic, operable to select a portion of the electronic data source using the navigation query; and
(g) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user. (emphasis added)

101. A computer program embodied on a computer readable medium 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:
(a) a code segment that receives a spoken request for desired information from the user;
(b) a code segment that renders an interpretation of the spoken request;
(c) a code segment that constructs at least part of a navigation query based upon the interpretation;
<u>(d)a code segment that solicits additional input from the user, including user interaction in a non-spoken modality different than the original request</u>;
(e) a code segment that refines the navigation query, based upon the additional input;
(f) a code segment that uses the refined navigation query to select a portion of the electronic data source; and
(g) a code segment that transmits the selected portions of the electronic data source from the network server to a primarily stationary, display device located locally with the user. (emphasis added)

Applicants direct the Examiner's attention to the fact that Applicants' invention teaches a novel method and apparatus for speech-based navigation <u>where the method solicits additional input from the user, including user interaction **in a non-spoken modality different than the original request**</u>. Specifically, Applicants address the criticality of errors and deficiencies via user interface modalities in addition to spoken natural language. It has been observed that users are often frustrated by ineffective or non optimal speech-based navigation that simply engages the user repeatedly in a long series of questions and answers, i.e., "single modal interaction", to perfect the navigation query. This single modal approach is often tedious and uninspiring for a user who must refine the navigation query repeatedly to achieve the desired result, thereby increasing the time the user must interact with a system. In fact, one goal of the speech-based navigation is to relieve this very tedium where the user must engage a system repeatedly, e.g., via a long sequence of menus to achieve the desired result.

To address this criticality, Applicants' navigation query can be refined via input from the user, where the user interaction is in a non-spoken modality different than the original request. To illustrate, if a portion of the navigation query can be achieved, then the result can be presented to the user in a way that the user can provide additional input via interaction that is in a non-spoken modality that is different than the original request. For example, if the "partial" navigation query produces three possible results, then the results can be presented to the user via a menu with the most likely result being highlighted. The user can then press a button on a remote unit to accept the highlighted result or simply scroll to one of the other three choices. Thus, the pressing of the button by the user is a **user interaction that is in a non-spoken modality different than the original request, e.g., a natural language request that originally started the navigation request**. This is an important aspect of the invention because of the psychological and real effect where the user perceives that the navigation query is actually progressing closer to the achieved result.

In contrast, Levin teaches that "the service host 112 determines if there are any ambiguities with respect to the response (step 222) and, if so, forwards **additional queries** to the user to help to resolve the ambiguities (step 224)". (emphasis added) (See Levin, Column 6, lines 40-43). Additionally, Levin states that "[t]he service host 112 includes a dialog control program that manages interactions with users **over several turns (e.g., it decides when to ask a question, when to give an answer**, provides means for clarifying ambiguities, and provides error control and recovery during an interaction)". (emphasis added) (See Levin, Column 5, lines 15-20). Levin's single modal approach is contrary to Applicants' invention and is one of the criticalities that Applicants' invention is designed to address. To further support Applicants' position, Levin states that "[t]he invention is independent of the actual modality of call placement". (See Levin, Column 4, lines 29-31) This statement is another clear indication that Levin is totally unconcerned with the modality of the user interaction and

09/524,095

is simply teaching a single modal approach via queries and answers.

However, the Examiner in the Advisory Action indicated that Levin's teaching of forwarding additional queries to the user constitutes a different modality. Applicants do not believe that the scope of Applicants' originally filed claims would read on this broad interpretation of different modality. Nevertheless, Applicants have agreed to clarify the independent claims to recite the term "a non-spoken modality different than the original request". The Examiner in several telephone conversations with Applicants' representative have indicated that this clarification will likely overcome the present rejection.

Additionally, it should be noted that this amendment is **not** made to overcome the cited prior art because it is Applicants' belief that the originally filed claims would not read on the invention disclosed by Levin. Thus, this clarifying amendment should not be interpreted in a manner that would limit the future application of Doctrine of Equivalents to Applicants' claims.

Therefore, the Applicants respectfully submit that independent claims 56, 82 and 101 are not anticipated by the Levin reference. As such, claims 56, 82 and 101 fully satisfy the requirements of 35 U.S.C. §102 and are patentable thereunder.

Claims 57-81, 83-100 and 102-126 depend, either directly or indirectly, from claims 56, 82 and 101 and recite additional features therefor. Since Levin fails to anticipate Applicants' invention as recited in Applicants' amended independent claims 56, 82 and 101, dependent claims 57-81, 83-100 and 102-126 are also not anticipated under 35 U.S.C. § 102 and are allowable for the same reason noted above.

II. Claims added in Preliminary Amendment dated September 12, 2000

Applicants have previously directed the Examiner's attention to the fact that it appears that the additional claims added in the Preliminary Amendment dated September 12, 2000 have not be addressed. Applicants respectfully request that the Examiner should verify the status of these added claims.

09/524,095

Conclusion

Thus, the Applicants submit that all of these claims now fully satisfy the requirements of 35 U.S.C. §102. Consequently, the Applicants believe that all these claims are presently in condition for allowance. Accordingly, both reconsideration of this application and its swift passage to issue are earnestly solicited.

If, however, the Examiner believes that there are any unresolved issues requiring the maintenance of the present final office action in any of the claims now pending in the application, it is requested that the Examiner telephone Mr. Kin-Wah Tong, Esq. at (732) 530-9404 so that appropriate arrangements can be made for resolving such issues as expeditiously as possible.

Respectfully submitted,

4/10/02

Kin-Wah Tong, Attorney
Reg. No. 39,400
(732) 530-9404

Moser, Patterson & Sheridan, LLP
595 Shrewsbury Avenue
First Floor,
Shrewsbury, New Jersey 07702

09/524,095

Appendix

(Marked-up version of amended claims)

56. (Amended) 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;

(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.

82. (Amended) A system 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, the system comprising:

(a) a portable microphone operable to receive a spoken request for desired information from the user;

(b) language processing logic, operable to render an interpretation of the spoken request;

(c) query construction logic, operable to construct a navigation query in response to the interpretation of the spoken request;

(d) user interaction logic, operable to solicit additional input from the user, including user interaction in a non-spoken modality different than the original request;

(e) query refining logic, operable to refine the navigation query, based upon the additional input;

(f) navigation logic, operable to select a portion of the electronic data source using the navigation query; and

(g) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

101. A computer program embodied on a computer readable medium 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:

(a) a code segment that receives a spoken request for desired information from the user;

(b) a code segment that renders an interpretation of the spoken request;

(c) a code segment that constructs at least part of a navigation query based upon the interpretation;

(d) a code segment that solicits additional input from the user, including user interaction in a <u>non-spoken</u> modality different than the original request;

(e) a code segment that refines the navigation query, based upon the additional input;

(f) a code segment that uses the refined navigation query to select a portion of the electronic data source; and

(g) a code segment that transmits the selected portions of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

Under the Paperwork Reduction Act of 1995, nc ons are required to respond to a collection of information unle lisplays a valid OMB control number.

FEE TRANSMITTAL for FY 2002

OIPE JC95 APR 1 0 2002 PATENT & TRADEMARK OFFICE

Patent fees are subject to annual revision.

TOTAL AMOUNT OF PAYMENT ($) 570.00

Complete If Known	
Application Number	09/524,095
Filing Date	March 13, 2000
First Named Inventor	Halverson
Examiner Name	F. Backer
Group / Art Unit	2155
Attorney Docket No.	SRI 1P037

RECEIVED APR 1 2 2002 Technology Center 2100

METHOD OF PAYMENT (check one)

1. ☒ The Commissioner is hereby authorized to charge indicated fees and credit any over payments to:

Deposit Account Number: 20-0782

Deposit Account Name: Moser, Patterson & Sheridan, LLP

☒ Charge Any Additional Fee Required Under 37 CFR 1.16 and 1.17

☒ Applicant claims small entity status. See 37 CFR 1.27

2. ☐ Payment Enclosed:

☐ Check ☐ Credit card ☐ Money Order ☐ Other

FEE CALCULATION

1. BASIC FILING FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
101	740	201	370	Utility filing fee	
106	330	206	165	Design filing fee	
107	510	207	255	Plant filing fee	
108	740	208	370	Reissue filing fee	
114	160	214	80	Provisional filling fee	

SUBTOTAL (1) ($) 0

2. EXTRA CLAIM FEES

			Extra Claims		Fee from below		Fee Paid
Total Claims		-20 ** =	0	X		=	0
Independent Claims		-3 ** =	0	X		=	0
Multiple Dependent				X		=	0

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description
103	18	203	9	Claims in excess of 20
102	84	202	42	Independent claims in excess of 3
104	280	204	140	Multiple dependent claim, if not paid
109	84	209	42	** Reissue independent claims over original patent
110	18	210	9	** Reissue claims in excess of 20 and over original patent

SUBTOTAL (2) ($) 0

**or number previously paid, if greater; For Reissues, see above

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
105	130	205	65	Surcharge - late filing fee or oath	
127	50	227	25	Surcharge - late provisional filing fee or cover sheet.	
139	130	139	130	Non-English specification	
147	2,520	147	2,520	For filing a request for reexamination	
112	920*	112	920*	Requesting publication of SIR prior to Examiner action	
113	1,840*	113	1,840*	Requesting publication of SIR after Examiner action	
115	110	215	55	Extension for reply within first month	
116	400	216	200	Extension for reply within second month	
117	920	217	460	Extension for reply within third month	200.00
118	1,440	218	720	Extension for reply within fourth month	
128	1,960	228	980	Extension for reply within fifth month	
119	320	219	160	Notice of Appeal	
120	320	220	160	Filing a brief in support of an appeal	
121	280	221	140	Request for oral hearing	
138	1,510	138	1,510	Petition to institute a public use proceeding	
140	110	240	55	Petition to revive – unavoidable	
141	1,280	241	640	Petition to revive – unintentional	
142	1,280	242	640	Utility issue fee (or reissue)	
143	460	243	230	Design issue fee	
144	620	244	310	Plant issue fee	
122	130	122	130	Petitions to the Commissioner	
123	50	123	50	Processing fee under 37 CFR 1.17 (q)	
126	180	126	180	Submission of Information Disclosure Stmt	
581	40	581	40	Recording each patent assignment per property (times number of properties)	
146	740	246	370	Filing a submission after final rejection (37 CFR § 1.129(a))	
149	740	249	370	For each additional invention to be examined (37 CFR § 1.129(b))	
179	740	279	370	Request for Continued Examination (RCE)	370.00
169	900	169	900	Request for expedited examination of a design application	

Other fee (specify)

*Reduced by Basic Filing Fee Paid SUBTOTAL (3) ($) 570.00

SUBMITTED BY			Complete (*if applicable*)		
Name (Print/Type)	KIN-WAH TONG	*Registration No. Attorney/Agent)*	39,400	*Telephone*	(732)530-9404
Signature				*Date*	APRIL 10, 2002

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Washington, DC 20231.

8U

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/524,095	03/13/2000	Christine Halverson	SRI1P037	6294

7590 05/07/2002

THOMASON, MOSER & PATTERSON, LLP
595 SHREWSBURY AVENUE
SUITE 100
SHREWSBURY, NJ 07702

EXAMINER	
BACKER, FIRMIN	
ART UNIT	PAPER NUMBER
3621	

DATE MAILED: 05/07/2002 #28

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

PTO-90C (Rev. 07-01)

Office Action Summary	Application No. 09/524,095	Applicant(s) HALVERSON ET AL.
	Examiner Firmin Backer	Art Unit 3621

-- The MAILING DATE of this communication appears on the cover sheet with the 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 *10 April 2002* .

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) *56-126* 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) *56-126* is/are rejected.

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

8)☐ Claims _____ 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 objected to by the Examiner.

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. § 119

13)☐ 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.

14)☐ Acknowledgement is made of a claim for domestic priority under 35 U.S.C. § 119(e).

Attachment(s)

15)☐ Notice of References Cited (PTO-892)

16)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

17)☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .

18)☐ Interview Summary (PTO-413) Paper No(s). _____ .

19)☐ Notice of Informal Patent Application (PTO-152)

20)☐ Other: .

Continued Examination Under 37 CFR 1.114

A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on April 10th, 2002 has been entered.

Response to Arguments

1. Applicant's arguments with respect to claims 56-126 have been considered but are moot in view of the new ground(s) of rejection.

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 56-126 are rejected under 35 U.S.C. 103(a) as being unpatentable over Levin et al. (U.S. Patent No. 6,173,279) in view of French-St. George et al (U.S. Patent 6,012,030 *(applicant submitted IDS)).*

4. As per claim 56, Levin et al teach a method for speech-based navigation (*information server, 110)* of an electronic data source located at one or more network servers located remotely

from a user, (*see abstract, fig 1, column 3 lines 5-35*), comprising receiving a spoken request (*receive a natural language query*) for desired information from the user (*user*); rendering an interpretation (*creating a semantic representation*) of the spoken request, constructing a navigation (*generating search)* query based upon the interpretation, refining the navigation query, based upon the additional input (*see column 6 lines 20-59*), using the 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 primarily stationary, display device located locally with the user (*see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claims 1, 10, 22*). Levin et al fail to teach an inventive concept of soliciting additional input from the user including user interaction in a non-spoken modality different that the original request. However, French-St. George et al. teach inventive concept of soliciting additional input from the user including user interaction in a non-spoken modality different that the original request (*see column 9 lines 36-65).* Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Levin et al's inventive concept to include French-St. George et al's inventive concept of soliciting additional input from the user including user interaction in a non-spoken modality different that the original request because this would have avoided or reduces error as the system search for user request thereby enhance the flexibility and the efficiency of the system.

5. As per claim 57, Levin et al teach a method of rendering the interpretation includes deriving linguistic information by using a speech recognition and a linguistic parser (see abstract, fig 1, column 3 lines 37-5 lines 40).

6. As per claim 58-62, Levin et al teach a method of constructing a navigation query in the form of a database query on a computing device located on a network including extracting an input template for an online scripted interface to the data source to be used for the construction of the navigation query and dynamically scraping the online scripted interface (see abstract, fig. 1-3, column 3 line 36-9 line 5)

7. As per claim 63-68, Levin et al teach a method of soliciting additional input is performed in response deficiency including unresolved word encountered after the first navigation of the data source, required element of the navigational query, data recorded within the data source, failure to identify data record responsive to navigational query (see column 6 lines 20-59).

8. As per claim 69, Levin et al teach a method wherein the additional input is solicited upon receiving a user-input statement…(see column 6 lines 20-59).

9. As per claim 70-73, Levin et al teach a method of soliciting additional input from the user, including presenting: a menu, a textual or an audible request, a list of portions of data source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

10. As per claim 74-75, Levin et al teach a method wherein additional input received from the user is speech based, of no spoken input source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

11. As per claim 76, Levin et al teach a method wherein steps (d)-(e) are repeated until the navigational query if deemed adequate source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

12. As per claim 77, 78, Levin et al teach a method wherein the input modality includes selecting (by speaking) from a displayed option menu (see abstract, fig. 1-3, column 3 line 36-9 line 5).

13. As per claim 79, Levin et al teach a method performed with respect to a plurality of user and corresponding client devices (see abstract, fig. 1-3, column 3 line 36-9 line 5).

14. As per claim 80-81, Levin et al teach a method of selecting data source from plurality of electronic data source storing multimedia content including audio and video content (see abstract, fig. 1-3, column 3 line 36-9 line 5)

15. As per claim 82, Levin et al teach a system for speech-based navigation (*information server, 110*) of an electronic data source located at one or more network servers located remotely from a user, (*see abstract, fig 1, column 3 lines 5-35*), comprising a portable microphone (*microphone, 105*) receiving a spoken request (*receive a natural language query*) for desired information from the user (user) a language processing logic (*natural language server, 114*) rendering an interpretation (*creating a semantic representation*) of the spoken request, (see *abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22*) a query construction logic

(*service host, 112*) constructing a navigation (*generating search)* query based upon the interpretation; a query interaction logic (*service host, 112)* a query refining logic (*service host, 112)*) refining the navigation query, based upon the additional input (*see column 6 lines 20-59*), a navigation logic (*service host, 112)* using the navigation query to select a portion of the electronic data source; electronic infrastructure (*network, 108*) transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user. (*see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22)*. However, French-St. George et al. teach inventive concept of soliciting additional input from the user including user interaction in a non-spoken modality different that the original request (*see column 9 lines 36-65).* Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Levin et al's inventive concept to include French-St. George et al's inventive concept of soliciting additional input from the user including user interaction in a non-spoken modality different that the original request because this would have avoided or reduces error as the system search for user request thereby enhance the flexibility and the efficiency of the system.

16. As per claim 83, Levin et al teach a system of rendering the interpretation includes deriving linguistic information by using a speech recognition and a linguistic parser (see abstract, fig 1, column 3 lines 37-5 lines 40).

17. As per claim 84-86, Levin et al teach a system of constructing a navigation query in the form of a database query on a computing device located on a network including extracting an

input template for an online scripted interface to the data source to be used for the construction of the navigation query and dynamically scraping the online scripted interface (see abstract, fig. 1-3, column 3 line 36-9 line 5).

18. As per claim 87, 88, 100, Levin et al teach a system wherein at least a portion of the language processing if hosted on a computing device coupled with a microphone located locally with a user and a network computing device located remotely and data in a two-way communication infrastructure (coaxial, DSL, satellite, wireless/cellular, fiber-optic) (see abstract, fig. 1-3, column 3 line 36-9 line 5).

19. As per claim 89-94, Levin et al teach a system of soliciting additional input is performed in response deficiency including unresolved word encountered after the first navigation of the data source, required element of the navigational query, data recorded within the data source, failure to identify data record responsive to navigational query (see column 6 lines 20-59).

20. As per claim 95, 96, Levin et al teach a system wherein the input modality includes selecting (by speaking) from a displayed option menu (see abstract, fig. 1-3, column 3 line 36-9 line 5).

21. As per claim 97-98, Levin et al teach a system of selecting data source from plurality of electronic data source storing multimedia content including audio and video content (see abstract, fig. 1-3, column 3 line 36-9 line 5).

22. As per claim 99, Levin et al teach a system wherein the display device receives data from the electronic device on the network via a communication box (see abstract, fig. 1-3, column 3 line 36-9 line 5).

23. As per claim 101, Levin et al teach a computer program for speech-based navigation (*information server, 110)* of an electronic data source located at one or more network servers located remotely from a user, (*see abstract, fig 1, column 3 lines 5-35*), comprising code segment receiving a spoken request (*receive a natural language query*) for desired information from the user (user); code segment rendering an interpretation (*creating a semantic representation*) of the spoken request, code segment constructing a navigation (*generating search)* query based upon the interpretation code segment, refining the navigation query, based upon the additional input (*see column 6 lines 20-59*), code segment using the navigation query to select a portion of the electronic data source; and code segment transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user (*see abstract, fig. 1-3, column 3 line 36-9 line 5, see also claim 1, 10, 22*). However, French-St. George et al. teach inventive concept of soliciting additional input from the user including user interaction in a non-spoken modality different that the original request (*see column 9 lines 36-65).* Therefore it would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Levin et al's inventive concept to include French-St. George et al's inventive concept of soliciting additional input from the user including user interaction in a non-spoken modality different that the original request because this would have avoided or

reduces error as the system search for user request thereby enhance the flexibility and the efficiency of the system.

24. As per claim 102, Levin et al teach a code segment deriving linguistic information by using a speech recognition and a linguistic parser (see abstract, fig 1, column 3 lines 37-5 lines 40).

25. As per claim 103-105, Levin et al teach a code segment of constructing a navigation query in the form of a database query on a computing device located on a network including extracting an input template for an online scripted interface to the data source to be used for the construction of the navigation query and dynamically scraping the online scripted interface (see abstract, fig. 1-3, column 3 line 36-9 line 5).

26. As per claim 106-107, Levin et al teach a computer program wherein rendering of the interpretation and the construction of the navigation query are performed on a computing device located locally with or remotely from the user (see abstract, fig. 1-3, column 3 line 36-9 line 5).

27. As per claim 108-114, Levin et al teach a code segment that solicits additional input display on option menu is performed by speaking in response deficiency including unresolved word encountered after the first navigation of the data source, required element of the navigational query, data recorded within the data source, failure to identify data record responsive to navigational query (see column 6 lines 20-59).

28. As per claim 115, Levin et al teach a computer program the act of selecting from the display is performed by speaking (see column 6 lines 20-59)

29. As per claim 116, Levin et al teach a code segment of the computer program operate with respect to a plurality of simultaneous user and corresponding client devices (see abstract, fig. 1-3, column 3 line 36-9 line 5).

30. As per claim 117, Levin et al teach a code segment that select data source form a plurality of electronic data source .... content (see abstract, fig. 1-3, column 3 line 36-9 line 5).

31. As per claim 118, Levin et al teach a computer program of selecting data source from plurality of electronic data source storing multimedia content including audio and video content (see abstract, fig. 1-3, column 3 line 36-9 line 5).

32. As per claim 119, Levin et al teach a computer program wherein the additional input is solicited upon receiving a user-input statement...(see column 6 lines 20-59).

33. As per claim 120-123, Levin et al teach a code segment of soliciting additional input from the user, including presenting: a menu, a textual or an audible request, a list of portions of data source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

34. As per claim 124-125, Levin et al teach a computer program wherein additional input received from the user is speech based, of no spoken input source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

35. As per claim 126, Levin et al teach a code segment wherein steps (d)-(e) are repeated until the navigational query if deemed adequate source (see abstract, fig. 1-3, column 3 line 36-9 line 5).

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Firmin Backer whose telephone number is (703) 305-0624. The examiner can normally be reached on Mon-Thu 8:30-6:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, James Trammel can be reached on (703) 305-9768. 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) 306-5484.

Firmin Backer
May 3, 2002

JAMES P. TRAMMELL
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

2155

PATENT

#29
LDD
7-1-02

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: JULIA, LUC

Serial No.: 09/524,095

Filed: 3/13/2000

GAU/Examiner: 2155/BACKER, F.

For: NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE

INPUT WITH MULTIMODAL CONVERGENT ERROR FEEDBACK

OIPE JC184
JUN 25 2002
PATENT & TRADEMARK OFFICE

Certificate of Mailing
I hereby certify that this correspondence is being deposited with the United States Postal Service as First Class Mail to: Commissioner for Patents, Washington, D.C. 20231
on: 06/18/02

Signed: ________________

RECEIVED
JUL 01 2002
Technology Center 2100

CHANGE OF ATTORNEY'S ADDRESS IN APPLICATION

Commissioner for Patents
Washington, D. C. 20231

Sir:

Please send all correspondence for this application as follows:

PERKINS COIE LLP
101 Jefferson Drive
Menlo Park, CA 94025-1114

Please direct any calls to Paul L. Hickman at (650) 838-4443.

Dated: 06/18/02

Respectfully submitted,

PERKINS COIE LLP

Paul L. Hickman
Reg. No. 28,516

101 Jefferson Drive
Menlo Park, CA 94025
Telephone: (650) 838-4443
Facsimile: (650) 838-4350

Attorney Docket No. SRI1P037 USA

UNITED STAT[illegible] DEPARTMENT OF COMMERCE
Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231

SERIAL NUMBER	FILING DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKETT NO.

EXAMINER

ART UNIT	PAPER NUMBER
	29

DATE MAILED:

EXAMINER INTERVIEW SUMMARY RECORD

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

(1) David Wiley (3)

(2) Kuwah Fong 39,400 (4)

Date of interview

Type: ☐ Telephonic ☒ Personal (copy is given to ☐ applicant ☐ applicant's representative).

Exhibit shown or demonstration conducted: ☐ Yes ☒ No. If yes, brief description:

Agreement ☒ was reached with respect to some or all of the claims in question. ☐ was not reached.

Claims discussed: 56-126

Identification of prior art discussed: French

Description of the general nature of what was agreed to if an agreement was reached, or any other comments: The applicant agreed to ~~the~~ amend element (d) in claim 56, 82, 101 and ~~the~~ examiner agreed to withdraw the previous rejection upon the amendment.

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

☒ 1. It is not necessary for applicant to provide a separate record of the substance of the interview.

Unless the paragraph below has been checked to indicate to the contrary, A FORMAL WRITTEN RESPONSE TO THE LAST OFFICE ACTION IS NOT WAIVED AND MUST INCLUDE THE SUBSTANCE OF THE INTERVIEW (e.g., items 1-7 on the reverse side of this form). If a response to the last Office action has already been filed, then applicant is given one month from this interview date to provide a statement of the substance of the interview.

☐ 2. Since the examiner's interview summary above (including any attachments) reflects a complete response to each of the objections, rejections and requirements that may be present in the last Office action, and since the claims are now allowable, this completed form is considered to fulfill the response requirements of the last Office action. Applicant is not relieved from providing a separate record of the substance of the interview unless box 1 above is also checked.

Examiner's Signature

PTOL-413 (REV. 2-93)

ORIGINAL FOR INSERTION IN RIGHT HAND FLAP OF FILE WRAPPER

09/524,095

8/7/2

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

PATENT APPLICATION

Applicant: **Halverson et al.**

Case: **SRI1P037**

Serial No.: **09/524,095** Filed: **March 13, 2000**

Group Art Unit: **2155**

Examiner: **Firmin Backer**

Title: **NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK**

ASSISTANT COMMISSIONER FOR PATENTS
Box Non-Fee Amendment
Washington, D. C. 20231

SIR:

AMENDMENT AND RESPONSE UNDER 37 C.F.R. § 1.111

This response addresses the Office Action dated May 7, 2002. The Office Action appears to be labeled as Paper No. 10.

IN THE CLAIMS

Please amend claims 56-181 as shown below. The claims are "clean version" of the amended claims, i.e., with changes incorporated into the claims, whereas the Appendix to this Amendment illustrates the amended claims using underlines and brackets to indicate addition and deletion, respectively.

56. (Twice Amended) A method for speech-based navigation of an electronic data source, the electronic data source being located at one or more network servers

09/524,095

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.

2 ~~57~~. (Amended) The method of claim 1 ~~56~~, wherein the step of rendering an interpretation further includes deriving linguistic information by using a speech recognition engine and a linguistic parser.

3 ~~58~~. (Amended) The method of claim 1 ~~56~~, wherein the step of constructing a navigation query further includes the steps of extracting an input template for an online scripted interface to the data source, and using the input template to construct the navigation query.

4 ~~59~~. (Amended) The method of claim 3 ~~58~~, wherein the step of extracting the input template includes dynamically scraping the online scripted interface.

5 ~~60~~. (Amended) The method of claim 1 ~~56~~, wherein the navigation query is constructed in the format of a database query language.

09/524,095

~~61~~ 6. (Amended) The method of claim ~~56~~ 1, wherein the step of rendering an interpretation and the step of constructing a navigation query are performed, at least in part, on a computing device located locally with the user.

~~62~~ 7. (Amended) The method of claim ~~56~~ 1, wherein the step of rendering an interpretation and the step of constructing a navigation query are performed, at least in part, on a network computing device located remotely from the user.

~~63~~ 8. (Amended) The method of claim ~~56~~ 1, wherein the step of soliciting additional input is performed in response to one or more deficiencies encountered during the step of constructing a navigation query.

~~64~~ 9. (Amended) The method of claim ~~63~~ 8, wherein the deficiencies include unresolved words of the spoken request.

~~65~~ 10. (Amended) The method of claim ~~63~~ 8, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken request.

~~66~~ 11. (Amended) The method of claim ~~56~~ 1, wherein the step of soliciting additional input is performed in response to one or more deficiencies encountered after a first navigation of the data source using the navigation query constructed in step (c).

~~67~~ 12. (Amended) The method of claim ~~66~~ 11, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

~~68~~ 13. (Amended) The method of claim ~~66~~ 11, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

09/524,095

~~69~~ 14. (Amended) The method of claim ~~56~~ 1, wherein the additional input is solicited upon receiving a user-input statement that additional information is required.

~~70~~ 15. (Amended) The method of claim ~~56~~ 1, wherein the step of soliciting the additional input includes presenting a menu to the user on the client device of the user.

~~71~~ 16. (Amended) The method of claim ~~56~~ 1, wherein the step of soliciting the additional input includes presenting a textual request for the additional input.

~~72~~ 17. (Amended) The method of claim ~~56~~ 1, wherein the step of soliciting the additional input includes an audible request for the additional input.

~~73~~ 18. (Amended) The method of claim ~~56~~ 1, wherein the step of soliciting the additional input includes presenting a list of portions of the electronic data source that match the navigational query.

~~74~~ 19. (Amended) The method of claim ~~56~~ 1, wherein additional input received from the user is at least partially speech based.

~~75~~ 20. (Amended) The method of claim ~~56~~ 1, wherein additional input received from the user includes no spoken input.

~~76~~ 21. (Amended) The method of claim ~~56~~ 1, wherein steps (d)-(e) are repeated until the navigational query is deemed adequate.

~~77~~ 22. (Amended) The method of claim ~~56~~ 1, wherein the input modality of step (d) includes selecting from a displayed option menu.

~~78~~ 23. (Amended) The method of claim ~~77~~ 22, wherein the act of selecting from the displayed option menu is performed by speaking.

24
~~79~~. (Amended) The method of claim ~~58~~ 1, wherein the method is performed with respect to a plurality of simultaneous users and corresponding client devices.

25
~~80~~. (Amended) The method of claim ~~58~~ 1, further including the step of selecting the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken request.

26
~~81~~. (Amended) The method of claim ~~56~~ 1, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

27
~~82~~. (Twice amended) A system 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, the system comprising:

(a) a portable microphone operable to receive a spoken request for desired information from the user;

(b) language processing logic, operable to render an interpretation of the spoken request;

(c) query construction logic, operable to construct a navigation query in response to the interpretation of the spoken request;

(d) user interaction logic, operable to 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 said non-spoken modality;

(e) query refining logic, operable to refine the navigation query, based upon the additional input;

(f) navigation logic, operable to select a portion of the electronic data source using the navigation query; and

(g) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

28 ~~83~~. (Amended) The system of claim 27 ~~82~~, wherein the language processing logic includes speech recognition logic and an linguistic parsing logic for deriving linguistic information.

29 ~~84~~. (Amended) The system of claim 27 ~~82~~, wherein the language processing logic extracts an input template for an online scripted interface to the data source, and uses the input template to construct the navigation query.

30 ~~85~~. (Amended) The system of claim 29 ~~84~~, wherein the language processing logic dynamically scrapes the online scripted interface.

31 ~~86~~. (Amended) The system of claim 27 ~~82~~, wherein the query construction logic constructs the query in the format of a database query language.

32 ~~87~~. (Amended) The system of claim 27 ~~82~~, wherein at least a portion of the language processing logic is hosted on a computing device located locally with the user, and wherein the portable microphone is electronically coupled to the local computing device.

33 ~~88~~. (Amended) The system of claim 27 ~~82~~, wherein at least a portion of the language processing logic is hosted on a network computing device located remotely from the user, and wherein the portable microphone sends data to the remote network computing device via the communications infrastructure.

34 ~~89~~. (Amended) The system of claim 27 ~~82~~, wherein the user interaction logic solicits additional input in response to one or more deficiencies encountered during construction of the navigation query.

09/524,095

35 ~~90~~. (Amended) The system of claim 34 ~~89~~, wherein the deficiencies include unresolved words of the spoken request.

36 ~~91~~. (Amended) The system of claim 34 ~~89~~, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken request.

37 ~~92~~. (Amended) The system of claim 27 ~~82~~, wherein the user interaction logic solicits additional input in response to one or more deficiencies encountered after a first navigation of the data source performed by the navigation logic.

38 ~~93~~. (Amended) The system of claim 37 ~~92~~, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

39 ~~94~~. (Amended) The system of claim 37 ~~92~~, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

40 ~~95~~. (Amended) The system of claim 27 ~~82~~, wherein the user interaction logic displays an option menu.

41 ~~96~~. (Amended) The system of claim 40 ~~95~~, wherein the act of selecting from the displayed option menu is performed by speaking.

42 ~~97~~. (Amended) The system of claim 27 ~~82~~, wherein the navigation logic selects the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken request.

43 ~~98~~. (Amended) The system of claim 27 ~~82~~, wherein the electronic data source stores

multimedia content including at least one of video content and audio content.

44 ~~99~~. (Amended) The system of claim ~~82~~ 27, wherein the display device receives data from the electronic data source on the network servers via a communications box.

45 ~~100~~. (Amended) The system of claim ~~82~~ 27, wherein the electronic communication infrastructure is a two-way infrastructure and is selected from among one or more of the following group: {coaxial cable, DSL, satellite, wireless/cellular, fiber-optic}.

46 ~~101~~. (Twice amended) A computer program embodied on a computer readable medium 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:

(a) a code segment that receives a spoken request for desired information from the user;

(b) a code segment that renders an interpretation of the spoken request;

(c) a code segment that constructs at least part of a navigation query based upon the interpretation;

(d) a code segment that solicits 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) a code segment that refines the navigation query, based upon the additional input;

(f) a code segment that uses the refined navigation query to select a portion of the electronic data source; and

(g) a code segment that transmits the selected portions of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

47 ~~102~~. (Amended) The computer program of claim ~~101~~ 46, further comprising a code

09/524,095

segment that derives linguistic information by using a speech recognition engine and a linguistic parser.

48 ~~103~~. (Amended) The computer program of claim 46 ~~101~~, further comprising a code segment that extract an input template for an online scripted interface to the data source, and a code segment that uses the input template to construct the navigation query.

49 ~~104~~. (Amended) The computer program of claim 48 ~~103~~, further comprising a code segment that dynamically scrapes the online scripted interface.

50 ~~105~~. (Amended) The computer program of claim 46 ~~101~~, wherein the navigation query is constructed in the format of a database query language.

51 ~~106~~. (Amended) The computer program of claim 46 ~~101~~, wherein rendering of the interpretation and the construction of the navigation query are performed, at least in part, on a computing device located locally with the user.

52 ~~107~~. (Amended) The compute program of claim 46 ~~101~~, wherein the rendering of the interpretation and the construction of a navigation query are performed, at least in part, on a network computing device located remotely from the user.

53 ~~108~~. (Amended) The computer program of claim 46 ~~101~~, wherein code segment that solicits additional input solicits the additional input in response to one or more deficiencies encountered during the constructing of the navigation query.

54 ~~109~~. (Amended) The computer program of claim 53 ~~108~~, wherein the deficiencies include unresolved words of the spoken request.

09/524,095

55 ~~110~~. (Amended) The computer program of claim 53 ~~108~~, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken request.

56 ~~111~~. (Amended) The computer program of claim 46 ~~101~~, wherein the code segment that solicits the additional input solicits the additional input in response to one or more deficiencies encountered after a first navigation of the data source.

57 ~~112~~. (Amended) The computer program of claim 56 ~~111~~, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

58 ~~113~~. (Amended) The computer program of claim 57 ~~112~~, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

59 ~~114~~. (Amended) The computer program of claim 46 ~~101~~, wherein code segment that solicits additional input displays an option menu.

60 ~~115~~. (Amended) The computer program of claim 59 ~~114~~, wherein the act of selecting from the displayed option menu is performed by speaking.

61 ~~116~~. (Amended) The computer program of claim 46 ~~101~~, wherein the code segments of the computer program operate with respect to a plurality of simultaneous users and corresponding client devices.

62 ~~117~~. (Amended) The computer program of claim 46 ~~101~~, further comprising a code segment that selects the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken request.

09/524,095

63 ~~118~~. (Amended) The computer program of claim 46 ~~101~~, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

64 ~~119~~. (Amended) The computer program of claim 46 ~~101~~, wherein the additional input is solicited upon receiving a user-input statement that additional information is required.

65 ~~120~~. (Amended) The computer program of claim 46 ~~101~~, wherein the code segment that solicits the additional input includes a code segment that presents a menu to the user on the client device of the user.

66 ~~121~~. (Amended) The computer program of claim 46 ~~101~~, wherein the code segment that solicits the additional input includes a code segment that presents a textual request for the additional input.

67 ~~122~~. (Amended) The computer program of claim 46 ~~101~~, wherein the code segment that solicits the additional input includes a code segment that produces an audible request for the additional input.

68 ~~123~~. (Amended) The computer program of claim 46 ~~101~~, wherein the code segment that solicits the additional input includes a code segment that presents a list of portions of the electronic data source that match the navigational query.

69 ~~124~~. (Amended) The computer program of claim 46 ~~101~~, wherein additional input received from the user is at least partially speech based.

70 ~~125~~. (Amended) The computer program of claim 46 ~~101~~, wherein additional input received from the user includes no spoken input.

39

09/524,095

71 ~~126~~. (Amended) The compute program of claim ~~101~~ 46, wherein code segments (d)-(e) are repeated until the navigational query is deemed adequate.

72 ~~127~~. (Amended) A method for utilizing spoken natural language for navigating 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 natural language ("NL") 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.

73 ~~128~~. (Amended) The method of claim ~~127~~ 72, wherein the step of rendering an interpretation further includes deriving linguistic information by using a speech recognition engine and an NL parser.

74 ~~129~~. (Amended) The method of claim ~~127~~ 72, wherein the step of constructing a navigation query further includes the steps of extracting an input template for an online scripted interface to the data source, and using the input template to construct the navigation query.

09/524,095

75 ~~130~~. (Amended) The method of claim ~~129~~ 74, wherein the step of extracting an input template includes dynamically scraping the online scripted interface.

76 ~~131~~. (Amended) The method of claim ~~127~~ 72, wherein the navigation query is constructed in the format of a database query language.

77 ~~132~~. (Amended) The method of claim ~~127~~ 72, wherein the step of rendering an interpretation and the step of constructing a navigation query are performed, at least in part, on a computing device located locally with the user.

78 ~~133~~. (Amended) The method of claim ~~127~~ 72, wherein the step of rendering an interpretation and the step of constructing a navigation query are performed, at least in part, on a network computing device located remotely from the user.

79 ~~134~~. (Amended) The method of claim ~~127~~ 72, wherein the step of soliciting additional input is performed in response to one or more deficiencies encountered during the step of constructing a navigation query.

80 ~~135~~. (Amended) The method of claim ~~134~~ 79, wherein the deficiencies include unresolved words of the spoken NL request.

81 ~~136~~. (Amended) The method of claim ~~134~~ 79, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken NL request.

82 ~~137~~. (Amended) The method of claim ~~127~~ 72, wherein the step of soliciting additional input is performed in response to one or more deficiencies encountered after a first navigation of the data source using the navigation query constructed in step (c).

09/524,095

83 ~~138~~. (Amended) The method of claim 82 ~~137~~, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

84 ~~139~~. (Amended) The method of claim 82 ~~137~~, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

85 ~~140~~. (Amended) The method of claim 72 ~~127~~, wherein the input modality of step (d) includes selecting from a displayed option menu.

86 ~~141~~. (Amended) The method of claim 85 ~~140~~, wherein the act of selecting from the displayed option menu is performed by speaking.

87 ~~142~~. (Amended) The method of claim 72 ~~127~~, wherein the method is performed with respect to a plurality of simultaneous users and corresponding client devices.

88 ~~143~~. (Amended) The method of claim 72 ~~127~~, further including the step of selecting the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken NL request.

89 ~~144~~. (Amended) The method of claim 72 ~~127~~, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

90 ~~145~~. (Amended) A system for utilizing spoken natural language to navigate an electronic data source, the electronic data source being located at one or more network servers located remotely from a user, the system comprising:

(a) a portable microphone operable to receive a spoken natural language ("NL") request for desired information from the user;

09/524,095

(b) spoken language processing logic, operable to render an interpretation of the spoken natural language request;

(c) query construction logic, operable to construct a navigation query in response to the interpretation of the spoken natural language request;

(d) user interaction logic, operable to 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 said non-spoken modality;

(e) query refining logic, operable to refine the navigation query, based upon the additional input;

(f) navigation logic, operable to select a portion of the electronic data source using the navigation query; and

(g) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

91 ~~146~~. (Amended) The system of claim 90 ~~145~~, wherein the spoken language processing logic includes speech recognition logic and an NL parsing logic for deriving linguistic information.

92 ~~147~~. (Amended) The system of claim 90 ~~145~~, wherein the spoken language processing logic extracts an input template for an online scripted interface to the data source, and uses the input template to construct the navigation query.

93 ~~148~~. (Amended) The system of claim 90 ~~145~~, wherein the spoken language processing logic dynamically scrapes the online scripted interface.

94 ~~149~~. (Amended) The system of claim 90 ~~145~~, wherein the query construction logic constructs the query in the format of a database query language.

09/524,095

95 ~~150~~. (Amended) The system of claim 90 ~~145~~, wherein at least a portion of the spoken language processing logic is hosted on a computing device located locally with the user, and wherein the portable microphone is electronically coupled to the local computing device.

96 ~~151~~. (Amended) The system of claim 90 ~~145~~, wherein at least a portion of the spoken language processing logic is hosted on a network computing device located remotely from the user, and wherein the portable microphone sends data to the remote network computing device via the communications infrastructure.

97 ~~152~~. (Amended) The system of claim 90 ~~145~~, wherein the user interaction logic solicits additional input in response to one or more deficiencies encountered during construction of the navigation query.

98 ~~153~~. (Amended) The system of claim 97 ~~152~~, wherein the deficiencies include unresolved words of the spoken NL request.

99 ~~154~~. (Amended) The system of claim 97 ~~152~~, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken NL request.

100 ~~155~~. (Amended) The system of claim 90 ~~145~~, wherein the user interaction logic solicits additional input in response to one or more deficiencies encountered after a first navigation of the data source performed by the navigation logic.

101 ~~156~~. (Amended) The system of claim 100 ~~155~~, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

09/524,095

~~157~~ 102. (Amended) The system of claim ~~155~~ 100, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

~~158~~ 103. (Amended) The system of claim ~~155~~ 100, wherein the user interaction logic displays an option menu.

~~159~~ 104. (Amended) The system of claim ~~158~~ 103, wherein the act of selecting from the displayed option menu is performed by speaking.

~~160~~ 105. (Amended) The system of claim ~~145~~ 90, wherein the navigation logic selects the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken NL request.

~~161~~ 106. (Amended) The system of claim ~~145~~ 90, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

~~162~~ 107. (Amended) The system of claim ~~145~~ 90, wherein the display device receives data from the electronic data source on the network servers via a communications box.

~~163~~ 108. (Amended) The system of claim ~~145~~ 90, wherein the electronic communication infrastructure is a two-way infrastructure and is selected from among one or more of the following group: {coaxial cable, DSL, satellite, wireless/cellular, fiber-optic}.

~~164~~ 109. (Amended) A computer program embodied on a computer readable medium for utilizing spoken natural language for navigating an electronic data source, the electronic data source being located at one or more network servers located remotely from a user, comprising:

(a) a code segment that receives a spoken natural language ("NL") request for desired information from the user;

09/524,095

(b) a code segment that renders an interpretation of the spoken natural language request;

(c) a code segment that constructs at least part of a navigation query based upon the interpretation;

(d) a code segment that solicits 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) a code segment that refines the navigation query, based upon the additional inputs;

(f) a code segment that uses the refined navigation query to select a portion of the electronic data source; and

(g) a code segment that transmits the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

110 ~~165~~. (Amended) The computer program of claim 109 ~~164~~, further comprising a code segment that derives linguistic information by using a speech recognition engine and an NL parser.

111 ~~166~~. (Amended) The computer program of claim 109 ~~164~~, further comprising a code segment that extract an input template for an online scripted interface to the data source, and a code segment that uses the input template to construct the navigation query.

112 ~~167~~. (Amended) The computer program of claim 111 ~~166~~, further comprising a code segment that dynamically scrapes the online scripted interface.

113 ~~168~~. (Amended) The computer program of claim 109 ~~164~~, wherein the navigation query is constructed in the format of a database query language.

114 ~~169~~. (Amended) The computer program of claim 109 ~~164~~, wherein rendering of the interpretation and the construction of the navigation query are performed, at least in part, on a computing device located locally with the user.

115 ~~170~~. (Amended) The computer program of claim 109 ~~164~~, wherein the rendering of the interpretation and the construction of a navigation query are performed, at least in part, on a network computing device located remotely from the user.

116 ~~171~~. (Amended) The computer program of claim 109 ~~164~~, wherein code segment that solicits additional input solicits the additional input in response to one or more deficiencies encountered during the constructing of the navigation query.

117 ~~172~~. (Amended) The computer program of claim 116 ~~171~~, wherein the deficiencies include unresolved words of the spoken NL request.

118 ~~173~~. (Amended) The computer program of claim 116 ~~171~~, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken NL request.

119 ~~174~~. (Amended) The computer program of claim 109 ~~164~~, wherein the code segment that solicits the additional input solicits the additional input in response to one or more deficiencies encountered after a first navigation of the data source.

120 ~~175~~. (Amended) The computer program of claim 119 ~~174~~, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

09/524,095

121 119

~~176~~. (Amended) The computer program of claim ~~174~~, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

122 109

~~177~~. (Amended) The computer program of claim ~~164~~, wherein code segment that solicits additional input displays an option menu.

123 122

~~178~~. (Amended) The computer program of claim ~~177~~, wherein the act of selecting from the displayed option menu is performed by speaking.

124 109

~~179~~. (Amended) The computer program of claim ~~164~~, wherein the code segments of the computer program operate with respect to a plurality of simultaneous users and corresponding client devices.

125 109

~~180~~. (Amended) The computer program of claim ~~164~~, further comprising a code segment that selects the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken NL request.

126 109

~~181~~. (Amended) The computer program of claim ~~164~~, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

Please add the following new claims:

127

~~182~~. (New) A method for utilizing spoken natural language for navigating 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 natural language ("NL") request for desired information from the user;

09/524,095

(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, in accordance with results generated from said at least part of a navigation query;

(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.

128 ~~183~~. (New) The method of claim 127 ~~182~~, wherein the input modality of step (d) includes selecting from a displayed option menu.

129 ~~184~~. (New) The method of claim 128 ~~183~~, wherein the act of selecting from the displayed option menu is performed by speaking.

130 ~~185~~. (New) A method for utilizing spoken natural language for navigating 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 natural language ("NL") 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, in response to one or more deficiencies encountered during the step of constructing said at least part of a navigation query;

(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.

91

131 ~~186~~. (New) The method of claim 130 ~~185~~, wherein the input modality of step (d) includes selecting from a displayed option menu.

132 ~~187~~. (New) The method of claim 131 ~~186~~, wherein the act of selecting from the displayed option menu is performed by speaking.

REMARKS

Applicants' representative would like to thank Primary Examiner David Wiley for kindly taking a substantial amount of time on May 23, 2002 to discuss the merits of the subject invention in a face-to-face Examiner Interview. Applicants' representative is aware of the time constraint that is placed on the Examiner and is appreciative of the Examiner's willingness to devote such large quantity of time to discuss the case on the merit.

In view of the following discussion, the Applicants submit that none of the claims now pending in the application are anticipated under the provisions of 35 U.S.C. § 103. Thus, the Applicants believe that all of these claims are now in allowable form.

I. MISNUMBERING OF CLAIMS

The Examiner has correctly detected that the claims (1-71) added in the Preliminary Amendment dated June 30, 2000 to replace the originally filed claims 1-55 were incorrectly numbered. The Examiner, in turn, renumbered these claims as 56-126 in the Office Action dated April 24, 2001.

However, Applicants also filed a second Preliminary Amendment "B" on September 12, 2000 that re-inserted the original claims 1-55. Again, Applicants misnumbered these claims as 72-126. Applicants now believe that these claims should be renumbered as claims 127-181. In fact, Applicants previously requested Examiner Backer to acknowledge these claims in Applicants' Preliminary Amendment dated April 10, 2002 that was filed in conjunction with a RCE. However, the Examiner is completely silent as to the status of these claims in the present Office Action.

This issue was brought to the attention of Primary Examiner David Wiley during the Examiner Interview. The Examiner acknowledged the existence of these claims and indicated that the agreement reached during the Examiner Interview is equally applicable to these claims.

To assist the Examiner and as agreed during the Examiner Interview, Applicants have affirmatively amended claims 56-181 as shown above to reflect the proper numbering. Thus, renumbering these claims are purely cosmetic and do not narrow the scope of the claims. Applicants believe that claims 127-181 are also currently pending in the present application.

Applicants sincerely apologize for the confusion created by the misnumbering, but Applicants respectfully request that the Examiner verify the status of claims 127-181 in the next Office Action or Notice of Allowance. Namely, these claims have not been rejected or allowed.

II. REJECTION OF CLAIMS 56-126 UNDER 35 U.S.C. § 103

The Examiner has rejected claims 56-126 in Paragraphs 2-35 of the Office Action as being unpatentable over the Levin et al. patent (US Patent 6,173,279 issued January 9, 2001, hereinafter referred to as Levin) in view of French-St. George et al. (US Patent 6,012,030, issued January 4, 2000, hereinafter referred to as French). The rejection is respectfully traversed.

Levin teaches "a method of using at least one natural language query to retrieve information from one or more data resources and further performing a requested action using the retrieved information is disclosed". (See Levin, Column 2, lines 15-18) Namely, Levin teaches a method for using natural language query to obtain information, where upon receipt of the requested information, a desired action is executed based upon the requested information. To illustrate, Levin provides the example, where a user employs natural language to request the telephone number of a restaurant. Upon receipt of the telephone number, the telephone number is actually dialed for the user. (See Levin, Column 3 line 62 to Column 4, line 1)

French teaches a management of speech and audio prompts and interface, in multimodal user interfaces. Specifically, the system is designed to detect and dynamically switches the speech interface into background mode or foreground mode in response to the user's current interaction modality. In the background mode, the speech interface can only respond to a very limited set of voice commands. (See French, Column 3, lines 20-57)

During the Examiner Interview, Applicants directed the Examiner's attention to the fact that French is a layer by layer system, i.e., a system that repeatedly asks questions and waits for a response before issuing the next response, whereas Levin is a natural language query system. Thus, the combination of the alleged references was challenged by the Applicants.

Second, assuming, *arguendo*, that the alleged combination was proper, the combination still falls short of making Applicants' Invention obvious. Namely, Applicants' invention solicits additional input from the user, including user interaction in a non-spoken modality different than the original request <u>without requiring the user to request the non-spoken modality</u>. In contrast, Levin is completely devoid of any disclosure pertaining to a different modality of interaction and French's invention is tied to the constant need to detect what the user is doing and shifting the speech interface back and forth between background and foreground modes. The Examiner agreed during the Examiner Interview that the alleged combination would not make Applicants' invention obvious.

However, the Examiner suggested that a clarification of step d) in the independent claims would be appropriate. Although Applicants believe that the current language would overcome the present obviousness rejection, Applicants nevertheless agreed to clarify step d) in the independent claims. Specifically, Applicants amended all the independent claims to recite the term "without requiring the user to request said non-spoken modality".

However, for the record, Applicants' position is that this term is provided purely to clarify the claim. The Examiner indicated that such clarification would be acceptable.

Therefore, the Applicants respectfully submit that independent claims 56, 82, 101, 127, 145, and 164 are not made obvious by the Levin and French references. As such, claims 56, 82, 101, 127, 145, and 164 fully satisfy the requirements of 35 U.S.C. §103 and are patentable thereunder.

Claims 57-81, 83-100, 102-126, 128-144, 146-163 and 165-181 depend, either directly or indirectly, from claims 56, 82, 101, 127, 145, and 164 and recite additional features therefor. Since Levin and French fail to make obvious Applicants' invention as recited in Applicants' independent claims 56, 82, 101, 127, 145, and 164, dependent claims 57-81, 83-100, 102-126, 128-144, 146-163 and 165-181 are also not made obvious under 35 U.S.C. § 103 and are allowable for the same reason noted above.

III. NEW CLAIMS 182-187

In addressing the Examiner's concern pertaining to the clarification of step d) in Applicants' independent claims, Applicants have added new independent claims 182 and 185 to address this issue in a different manner. Specifically, Applicants' new independent claims recite the term "in accordance with results generated from said at least part of a navigation query" and "in response to one or more deficiencies encountered during the step of constructing said at least part of a navigation query", respectively. Support for these claims can be found in Applicants' specification, page 17, line 7 to page 19, line 9. However, support for these claims may also exist in other sections of Applicants' application.

In brief, Applicants' invention allows the system to present a non-spoken modality of interaction to the user based upon the results generated by performing the partial navigation query. For example, the system evaluates the results (e.g., a short list of choices) generated by the partial navigation query, and may realize that additional user input is necessary. At this point, the system elects to interact with the user in a non-spoken modality, e.g., presenting the short list of choices on a display according to results generated.

Alternatively, the system may evaluate the navigation query itself, (i.e., without performing the navigation query) and may realize that additional user input is necessary to fully construct the navigation query. Based on the deficiencies encountered, the system will elect to interact with the user in a non-spoken modality. These approaches will allow the user to quickly refine the navigation query, thereby providing a sense of progress to the user.

For the reasons presented above, Applicants submit that independent claims 182 and 185 and dependent claims 183-184 and 186-187 are also patentable over the cited references. Since claims 182-187 are supported by Applicants' specification, no new matter is introduced.

Conclusion

Thus, the Applicants submit that all of these claims now fully satisfy the requirements of 35 U.S.C. §103. Consequently, the Applicants believe that all these claims are presently in condition for allowance. Accordingly, both reconsideration of this application and its swift passage to issue are earnestly solicited.

If, however, the Examiner believes that there are any unresolved issues requiring the issuance of a final office action in any of the claims now pending in the application, it is requested that the Examiner telephone Mr. Kin-Wah Tong, Esq. at (732) 530-9404 so that appropriate arrangements can be made for resolving such issues as expeditiously as possible.

Respectfully submitted,

8/5/02

Kin-Wah Tong, Attorney
Reg. No. 39,400
(732) 530-9404

09/524,095

Moser, Patterson & Sheridan, LLP
595 Shrewsbury Avenue
First Floor,
Shrewsbury, New Jersey 07702

09/524,095

Appendix

(Marked-up version of amended claims)

[1] 56. (Twice Amended) 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.

[2] 57. (Amended) The method of claim [1] 56, wherein the step of rendering an interpretation further includes deriving linguistic information by using a speech recognition engine and a linguistic parser.

[3] 58. (Amended) The method of claim [1] 56, wherein the step of constructing a navigation query further includes the steps of extracting an input template for an online scripted interface to the data source, and using the input template to construct the navigation query.

[4] 59. (Amended) The method of claim [3] 58, wherein the step of extracting the input template includes dynamically scraping the online scripted interface.

[5] <u>60</u>. (Amended) The method of claim [1] <u>56</u>, wherein the navigation query is constructed in the format of a database query language.

[6] <u>61</u>. (Amended) The method of claim [1] <u>56</u>, wherein the step of rendering an interpretation and the step of constructing a navigation query are performed, at least in part, on a computing device located locally with the user.

[7] <u>62</u>. (Amended) The method of claim [1] <u>56</u>, wherein the step of rendering an interpretation and the step of constructing a navigation query are performed, at least in part, on a network computing device located remotely from the user.

[8] <u>63</u>. (Amended) The method of claim [1] <u>56</u>, wherein the step of soliciting additional input is performed in response to one or more deficiencies encountered during the step of constructing a navigation query.

[9] <u>64</u>. (Amended) The method of claim [8] <u>63</u>, wherein the deficiencies include unresolved words of the spoken request.

[10] <u>65</u>. (Amended) The method of claim [8] <u>63</u>, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken request.

[11] <u>66</u>. (Amended) The method of claim [1] <u>56</u>, wherein the step of soliciting additional input is performed in response to one or more deficiencies encountered after a first navigation of the data source using the navigation query constructed in step (c).

[12] <u>67</u>. (Amended) The method of claim [11] <u>66</u>, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

D

[13] 68. (Amended) The method of claim [11] 66, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

[14] 69. (Amended) The method of claim [1] 56, wherein the additional input is solicited upon receiving a user-input statement that additional information is required.

[15] 70. (Amended) The method of claim [1] 56, wherein the step of soliciting the additional input includes presenting a menu to the user on the client device of the user.

[16] 71. (Amended) The method of claim [1] 56, wherein the step of soliciting the additional input includes presenting a textual request for the additional input.

[17] 72. (Amended) The method of claim [1] 56, wherein the step of soliciting the additional input includes an audible request for the additional input.

[18] 73. (Amended) The method of claim [1] 56, wherein the step of soliciting the additional input includes presenting a list of portions of the electronic data source that match the navigational query.

[19] 74. (Amended) The method of claim [1] 56, wherein additional input received from the user is at least partially speech based.

[20] 75. (Amended) The method of claim [1] 56, wherein additional input received from the user includes no spoken input.

[21] 76. (Amended) The method of claim [1] 56, wherein steps (d)-(e) are repeated until the navigational query is deemed adequate.

[22] 77. (Amended) The method of claim [1] 56, wherein the input modality of step (d) includes selecting from a displayed option menu.

[23] 78. (Amended) The method of claim [22] 77, wherein the act of selecting from the displayed option menu is performed by speaking.

[24] 79. (Amended) The method of claim [1] 56, wherein the method is performed with respect to a plurality of simultaneous users and corresponding client devices.

[25] 80. (Amended) The method of claim [1] 56, further including the step of selecting the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken request.

[26] 81. (Amended) The method of claim [1] 56, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

[27] 82. (Twice amended) A system 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, the system comprising:

(a) a portable microphone operable to receive a spoken request for desired information from the user;

(b) language processing logic, operable to render an interpretation of the spoken request;

(c) query construction logic, operable to construct a navigation query in response to the interpretation of the spoken request;

(d) user interaction logic, operable to 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 said non-spoken modality;

P

09/524,095

(e) query refining logic, operable to refine the navigation query, based upon the additional input;

(f) navigation logic, operable to select a portion of the electronic data source using the navigation query; and

(g) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

[28] 83. (Amended) The system of claim [27] 82, wherein the language processing logic includes speech recognition logic and an linguistic parsing logic for deriving linguistic information.

[29] 84. (Amended) The system of claim [27] 82, wherein the language processing logic extracts an input template for an online scripted interface to the data source, and uses the input template to construct the navigation query.

[30] 85. (Amended) The system of claim [29] 84, wherein the language processing logic dynamically scrapes the online scripted interface.

[31] 86. (Amended) The system of claim [27] 82, wherein the query construction logic constructs the query in the format of a database query language.

[32] 87. (Amended) The system of claim [27] 82, wherein at least a portion of the language processing logic is hosted on a computing device located locally with the user, and wherein the portable microphone is electronically coupled to the local computing device.

[33] 88. (Amended) The system of claim [27] 82, wherein at least a portion of the language processing logic is hosted on a network computing device located remotely

from the user, and wherein the portable microphone sends data to the remote network computing device via the communications infrastructure.

[34] 89. (Amended) The system of claim [27] 82, wherein the user interaction logic solicits additional input in response to one or more deficiencies encountered during construction of the navigation query.

[35] 90. (Amended) The system of claim [34] 89, wherein the deficiencies include unresolved words of the spoken request.

[36] 91. (Amended) The system of claim [34] 89, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken request.

[37] 92. (Amended) The system of claim [27] 82, wherein the user interaction logic solicits additional input in response to one or more deficiencies encountered after a first navigation of the data source performed by the navigation logic.

[38] 93. (Amended) The system of claim [37] 92, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

[39] 94. (Amended) The system of claim [37] 92, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

[40] 95. (Amended) The system of claim [27] 82, wherein the user interaction logic displays an option menu.

[41] <u>96</u>. (Amended) The system of claim [40] <u>95</u>, wherein the act of selecting from the displayed option menu is performed by speaking.

[42] <u>97</u>. (Amended) The system of claim [27] <u>82</u>, wherein the navigation logic selects the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken request.

[43] <u>98</u>. (Amended) The system of claim [27] <u>82</u>, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

[44] <u>99</u>. (Amended) The system of claim [27] <u>82</u>, wherein the display device receives data from the electronic data source on the network servers via a communications box.

[45] <u>100</u>. (Amended) The system of claim [27] <u>82</u>, wherein the electronic communication infrastructure is a two-way infrastructure and is selected from among one or more of the following group: {coaxial cable, DSL, satellite, wireless/cellular, fiber-optic}.

[46] <u>101</u>. (Twice amended) A computer program embodied on a computer readable medium 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:

(a) a code segment that receives a spoken request for desired information from the user;

(b) a code segment that renders an interpretation of the spoken request;

(c) a code segment that constructs at least part of a navigation query based upon the interpretation;

(d) a code segment that solicits additional input from the user, including user interaction in a non-spoken modality different than the original request <u>without requiring</u>

the user to request said non-spoken modality;

(e) a code segment that refines the navigation query, based upon the additional input;

(f) a code segment that uses the refined navigation query to select a portion of the electronic data source; and

(g) a code segment that transmits the selected portions of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

[47] 102. (Amended) The computer program of claim [46] 101, further comprising a code segment that derives linguistic information by using a speech recognition engine and a linguistic parser.

[48] 103. (Amended) The computer program of claim [46] 101, further comprising a code segment that extract an input template for an online scripted interface to the data source, and a code segment that uses the input template to construct the navigation query.

[49] 104. (Amended) The computer program of claim [48] 103, further comprising a code segment that dynamically scrapes the online scripted interface.

[50] 105. (Amended) The computer program of claim [46] 101, wherein the navigation query is constructed in the format of a database query language.

[51] 106. (Amended) The computer program of claim [46] 101, wherein rendering of the interpretation and the construction of the navigation query are performed, at least in part, on a computing device located locally with the user.

[52] 107. (Amended) The compute program of claim [46] 101, wherein the rendering of

the interpretation and the construction of a navigation query are performed, at least in part, on a network computing device located remotely from the user.

[53] 108. (Amended) The computer program of claim [46] 101, wherein code segment that solicits additional input solicits the additional input in response to one or more deficiencies encountered during the constructing of the navigation query.

[54] 109. (Amended) The computer program of claim [53] 108, wherein the deficiencies include unresolved words of the spoken request.

[55] 110. (Amended) The computer program of claim [53] 108, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken request.

[56] 111. (Amended) The computer program of claim [46] 101, wherein the code segment that solicits the additional input solicits the additional input in response to one or more deficiencies encountered after a first navigation of the data source.

[57] 112. (Amended) The computer program of claim [56] 111, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

[58] 113. (Amended) The computer program of claim [57] 112, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

[59] 114. (Amended) The computer program of claim [46] 101, wherein code segment that solicits additional input displays an option menu.

[60] 115. (Amended) The computer program of claim [59] 114, wherein the act of selecting from the displayed option menu is performed by speaking.

[61] 116. (Amended) The computer program of claim [46] 101, wherein the code segments of the computer program operate with respect to a plurality of simultaneous users and corresponding client devices.

[62] 117. (Amended) The computer program of claim [46] 101, further comprising a code segment that selects the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken request.

[63] 118. (Amended) The computer program of claim [46] 101, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

[64] 119. (Amended) The computer program of claim [46] 101, wherein the additional input is solicited upon receiving a user-input statement that additional information is required.

[65] 120. (Amended) The computer program of claim [46] 101, wherein the code segment that solicits the additional input includes a code segment that presents a menu to the user on the client device of the user.

[66] 121. (Amended) The computer program of claim [46] 101, wherein the code segment that solicits the additional input includes a code segment that presents a textual request for the additional input.

[67] 122. (Amended) The computer program of claim [46] 101, wherein the code segment that solicits the additional input includes a code segment that produces an

D

audible request for the additional input.

[68] <u>123</u>. (Amended) The computer program of claim [46] <u>101</u>, wherein the code segment that solicits the additional input includes a code segment that presents a list of portions of the electronic data source that match the navigational query.

[69] <u>124</u>. (Amended) The computer program of claim [46] <u>101</u>, wherein additional input received from the user is at least partially speech based.

[70] <u>125</u>. (Amended) The computer program of claim [46] <u>101</u>, wherein additional input received from the user includes no spoken input.

[71] <u>126</u>. (Amended) The compute program of claim [46] <u>101</u>, wherein code segments (d)-(e) are repeated until the navigational query is deemed adequate.

[72] <u>127</u>. (Amended) A method for utilizing spoken natural language for navigating 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 natural language ("NL") 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 <u>non-spoken</u> modality different than the original request <u>without requiring the user to request said non-spoken modality</u>;

(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.

[73] 128. (Amended) The method of claim [72] 127, wherein the step of rendering an interpretation further includes deriving linguistic information by using a speech recognition engine and an NL parser.

[74] 129. (Amended) The method of claim [72] 127, wherein the step of constructing a navigation query further includes the steps of extracting an input template for an online scripted interface to the data source, and using the input template to construct the navigation query.

[75] 130. (Amended) The method of claim [74] 129, wherein the step of extracting an input template includes dynamically scraping the online scripted interface.

[76] 131. (Amended) The method of claim [72] 127, wherein the navigation query is constructed in the format of a database query language.

[77] 132. (Amended) The method of claim [72] 127, wherein the step of rendering an interpretation and the step of constructing a navigation query are performed, at least in part, on a computing device located locally with the user.

[78] 133. (Amended) The method of claim [72] 127, wherein the step of rendering an interpretation and the step of constructing a navigation query are performed, at least in part, on a network computing device located remotely from the user.

[79] 134. (Amended) The method of claim [72] 127, wherein the step of soliciting additional input is performed in response to one or more deficiencies encountered during the step of constructing a navigation query.

[80] 135. (Amended) The method of claim [79] 134, wherein the deficiencies include unresolved words of the spoken NL request.

[81] 136. (Amended) The method of claim [79] 134, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken NL request.

[82] 137. (Amended) The method of claim [72] 127, wherein the step of soliciting additional input is performed in response to one or more deficiencies encountered after a first navigation of the data source using the navigation query constructed in step (c).

[83] 138. (Amended) The method of claim [82] 137, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

[84] 139. (Amended) The method of claim [82] 137, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

[85] 140. (Amended) The method of claim [72] 127, wherein the input modality of step (d) includes selecting from a displayed option menu.

[86] 141. (Amended) The method of claim [85] 140, wherein the act of selecting from the displayed option menu is performed by speaking.

[87] 142. (Amended) The method of claim [72] 127, wherein the method is performed with respect to a plurality of simultaneous users and corresponding client devices.

[88] 143. (Amended) The method of claim [72] 127, further including the step of selecting the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken NL request.

[89] 144. (Amended) The method of claim [72] 127, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

[90] 145. (Amended) A system for utilizing spoken natural language to navigate an electronic data source, the electronic data source being located at one or more network servers located remotely from a user, the system comprising:

(a) a portable microphone operable to receive a spoken natural language ("NL") request for desired information from the user;
(b) spoken language processing logic, operable to render an interpretation of the spoken natural language request;
(c) query construction logic, operable to construct a navigation query in response to the interpretation of the spoken natural language request;
(d) user interaction logic, operable to 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 said non-spoken modality;
(e) query refining logic, operable to refine the navigation query, based upon the additional input;
(f) navigation logic, operable to select a portion of the electronic data source using the navigation query; and
(g) electronic communications infrastructure for transmitting the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

09/524,095

[91] <u>146</u>. (Amended) The system of claim [90] <u>145</u>, wherein the spoken language processing logic includes speech recognition logic and an NL parsing logic for deriving linguistic information.

[92] <u>147</u>. (Amended) The system of claim [90] <u>145</u>, wherein the spoken language processing logic extracts an input template for an online scripted interface to the data source, and uses the input template to construct the navigation query.

[93] <u>148</u>. (Amended) The system of claim [90] <u>145</u>, wherein the spoken language processing logic dynamically scrapes the online scripted interface.

[94] <u>149</u>. (Amended) The system of claim [90] <u>145</u>, wherein the query construction logic constructs the query in the format of a database query language.

[95] <u>150</u>. (Amended) The system of claim [90] <u>145</u>, wherein at least a portion of the spoken language processing logic is hosted on a computing device located locally with the user, and wherein the portable microphone is electronically coupled to the local computing device.

[96] <u>151</u>. (Amended) The system of claim [90] <u>145</u>, wherein at least a portion of the spoken language processing logic is hosted on a network computing device located remotely from the user, and wherein the portable microphone sends data to the remote network computing device via the communications infrastructure.

[97] <u>152</u>. (Amended) The system of claim [90] <u>145</u>, wherein the user interaction logic solicits additional input in response to one or more deficiencies encountered during construction of the navigation query.

09/524,095

[98] 153. (Amended) The system of claim [97] 152, wherein the deficiencies include unresolved words of the spoken NL request.

[99] 154. (Amended) The system of claim [97] 152, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken NL request.

[100] 155. (Amended) The system of claim [90] 145, wherein the user interaction logic solicits additional input in response to one or more deficiencies encountered after a first navigation of the data source performed by the navigation logic.

[101] 156. (Amended) The system of claim [100] 155, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

[102] 157. (Amended) The system of claim [100] 155, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

[103] 158. (Amended) The system of claim [100] 155, wherein the user interaction logic displays an option menu.

[104] 159. (Amended) The system of claim [103] 158, wherein the act of selecting from the displayed option menu is performed by speaking.

[105] 160. (Amended) The system of claim [90] 145, wherein the navigation logic selects the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken NL request.

09/524,095

[106] 161. (Amended) The system of claim [90] 145, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

[107] 162. (Amended) The system of claim [90] 145, wherein the display device receives data from the electronic data source on the network servers via a communications box.

[108] 163. (Amended) The system of claim [90] 145, wherein the electronic communication infrastructure is a two-way infrastructure and is selected from among one or more of the following group: {coaxial cable, DSL, satellite, wireless/cellular, fiber-optic}.

[109] 164. (Amended) A computer program embodied on a computer readable medium for utilizing spoken natural language for navigating an electronic data source, the electronic data source being located at one or more network servers located remotely from a user, comprising:

(a) a code segment that receives a spoken natural language ("NL") request for desired information from the user;

(b) a code segment that renders an interpretation of the spoken natural language request;

(c) a code segment that constructs at least part of a navigation query based upon the interpretation;

(d) a code segment that solicits 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) a code segment that refines the navigation query, based upon the additional inputs;

D

09/524,095

(f) a code segment that uses the refined navigation query to select a portion of the electronic data source; and

(g) a code segment that transmits the selected portion of the electronic data source from the network server to a primarily stationary, display device located locally with the user.

[110] 165. (Amended) The computer program of claim [109] 164, further comprising a code segment that derives linguistic information by using a speech recognition engine and an NL parser.

[111] 166. (Amended) The computer program of claim [109] 164, further comprising a code segment that extract an input template for an online scripted interface to the data source, and a code segment that uses the input template to construct the navigation query.

[112] 167. (Amended) The computer program of claim [111] 166, further comprising a code segment that dynamically scrapes the online scripted interface.

[113] 168. (Amended) The computer program of claim [109] 164, wherein the navigation query is constructed in the format of a database query language.

[114] 169. (Amended) The computer program of claim [109] 164, wherein rendering of the interpretation and the construction of the navigation query are performed, at least in part, on a computing device located locally with the user.

[115] 170. (Amended) The computer program of claim [109] 164, wherein the rendering of the interpretation and the construction of a navigation query are performed, at least in part, on a network computing device located remotely from the user.

Received from < 732 530 9808 > at 8/5/02 4:33:12 PM [Eastern Daylight Time]

09/524,095

[116] 171. (Amended) The computer program of claim [109] 164, wherein code segment that solicits additional input solicits the additional input in response to one or more deficiencies encountered during the constructing of the navigation query.

[117] 172. (Amended) The computer program of claim [116] 171, wherein the deficiencies include unresolved words of the spoken NL request.

[118] 173. (Amended) The computer program of claim [116] 171, wherein the deficiencies include one or more required elements of the navigational query not determinable from the interpretation of the spoken NL request.

[119] 174. (Amended) The computer program of claim [109] 164, wherein the code segment that solicits the additional input solicits the additional input in response to one or more deficiencies encountered after a first navigation of the data source.

[120] 175. (Amended) The computer program of claim [119] 174, wherein the deficiencies include existence of more than one data record within the data source responsive to the navigation query.

[121] 176. (Amended) The computer program of claim [119] 174, wherein the deficiencies include failure to identify a single data record within the data source responsive to the navigation query.

[122] 177. (Amended) The computer program of claim [109] 164, wherein code segment that solicits additional input displays an option menu.

[123] 178. (Amended) The computer program of claim [122] 177, wherein the act of selecting from the displayed option menu is performed by speaking.

09/524,095

[124] 179. (Amended) The computer program of claim [109] 164, wherein the code segments of the computer program operate with respect to a plurality of simultaneous users and corresponding client devices.

[125] 180. (Amended) The computer program of claim [109] 164, further comprising a code segment that selects the data source from among a plurality of candidate electronic data sources, in response to the interpretation of the spoken NL request.

[126] 181. (Amended) The computer program of claim [109] 164, wherein the electronic data source stores multimedia content including at least one of video content and audio content.

182. (New) A method for utilizing spoken natural language for navigating 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 natural language ("NL") 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, in accordance with results generated from said at least part of a navigation query;

(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.

183. (New) The method of claim 182, wherein the input modality of step (d) includes

selecting from a displayed option menu.

184. (New) The method of claim 183, wherein the act of selecting from the displayed option menu is performed by speaking.

185. (New) A method for utilizing spoken natural language for navigating 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 natural language ("NL") 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, in response to one or more deficiencies encountered during the step of constructing said at least part of a navigation query;

(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.

186. (New) The method of claim 185, wherein the input modality of step (d) includes selecting from a displayed option menu.

187. (New) The method of claim 186, wherein the act of selecting from the displayed option menu is performed by speaking.

TELEFAX COVER SHEET

MOSER, PATTERSON & SHERIDAN, LLP
ATTORNEYS AT LAW
595 SHREWSBURY AVENUE
FIRST FLOOR
SHREWSBURY, NJ 07702
TELEPHONE (732) 530-9404
TELEFAX (732) 530-9808

THIS MESSAGE HAS 51 PAGES INCLUDING THIS SHEET

TO: Assistant Commissioner of Patents

FAX NO.: 703-746-7239

FROM: Kin-Wah Tong

DATE: August 5, 2002

MATTER: Serial No. 09/524,095 Filed: March 13, 2000

DOCKET NO.: SRI 1P037

APPLICANT: HALVERSON, et al

The following has been received in the U.S. Patent and Trademark Office on the date of this facsimile:

___ Petition
___ Disclosure Statement & PTO-1449
___ Priority Document
___ Drawings (____ sheets) informal
X Response Under 37 CFR 1.111

X Transmittal Letter (2 copies)
___ Fee Transmittal (2 copies)
___ Deposit Account Transaction
X Facsimile Transmission Certificate dated AUGUST 5, 2002

CERTIFICATE OF TRANSMISSION UNDER 37 C.F.R. §1.8

I hereby certify that this correspondence is being transmitted by facsimile to the Assistant Commissioner for Patents, Washington, DC 20231 on AUGUST 5, 2002, Facsimile No. 703-746-7239.

Laura E. Crater
Name of person signing this certificate

Laura E. Crater AUGUST 5, 2002
Signature and date

Official

RECEIVED

PTO/SB/21 (08-00)

Please type a plus sign (+) inside this box → +

Approved for use through 10/31/2002. OMB 0651-0031
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

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.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	09/524,095
Filing Date	March 13, 2000
First Named Inventor	HALVERSON
Group Art Unit	2155
Examiner Name	F. BACKER
Attorney Docket Number	SRI 1 P 037

Total Number of Pages In This Submission: 9

ENCLOSURES *(check all that apply)*

- ☐ Fee Transmittal Form
  - ☐ Fee Attached
- ☒ Amendment / Response
  - ☐ After Final
  - ☐ Affidavits/declaration(s)
- ☐ Extension of Time Request
- ☐ Express Abandonment Request
- ☐ Information Disclosure Statement
- ☐ Certified Copy of Priority Document(s)
- ☐ Response to Missing Parts/ Incomplete Application
  - ☐ Response to Missing Parts under 37 CFR 1.52 or 1.53
- ☐ Assignment Papers *(for an Application)*
- ☐ Drawing(s)
- ☐ Licensing-related Papers
- ☐ Petition
- ☐ Petition to Convert to a Provisional Application
- ☐ Power of Attorney, Revocation Change of Correspondence Address
- ☐ Terminal Disclaimer
- ☐ Request for Refund
- ☐ CD, Number of CD(s)
- ☐ After Allowance Communication to Group
- ☐ Appeal Communication to Board of Appeals and Interferences
- ☐ Appeal Communication to Group *(Appeal Notice, Brief, Reply Brief)*
- ☐ Proprietary Information
- ☐ Status Letter
- ☐ Other Enclosure(s) *(please identify below)*:

Remarks: **Please charge the $138 additional claim fee (6 total claims at $9 each; 2 independent claims at $42 each) and any other fees due to Applicants' Attorneys' Deposit Account No. 20-0782. A duplicate copy of this transmittal is enclosed to facilitate the charge.**

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm *or* Individual name	KIN-WAH TONG, REG. NO. 39,400
Signature	
Date	August 5, 2002

PTO/SB/21 (08-00)
Approved for use through 10/31/2002. OMB 0651-0031
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
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.

Please type a plus sign (+) inside this box → [+]

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	09/524,095
Filing Date	March 13, 2000
First Named Inventor	HALVERSON
Group Art Unit	2155
Examiner Name	F. BACKER
Attorney Docket Number	SRI 1 P 037
Total Number of Pages in This Submission	9

ENCLOSURES *(check all that apply)*

- ☐ Fee Transmittal Form
  - ☐ Fee Attached
- ☒ Amendment / Response
  - ☐ After Final
  - ☐ Affidavits/declaration(s)
- ☐ Extension of Time Request
- ☐ Express Abandonment Request
- ☐ Information Disclosure Statement
- ☐ Certified Copy of Priority Document(s)
- ☐ Response to Missing Parts/ Incomplete Application
  - ☐ Response to Missing Parts under 37 CFR 1.52 or 1.53
- ☐ Assignment Papers *(for an Application)*
- ☐ Drawing(s)
- ☐ Licensing-related Papers
- ☐ Petition
- ☐ Petition to Convert to a Provisional Application
- ☐ Power of Attorney, Revocation Change of Correspondence Address
- ☐ Terminal Disclaimer
- ☐ Request for Refund
- ☐ CD, Number of CD(s)
- ☐ After Allowance Communication to Group
- ☐ Appeal Communication to Board of Appeals and Interferences
- ☐ Appeal Communication to Group *(Appeal Notice, Brief, Reply Brief)*
- ☐ Proprietary Information
- ☐ Status Letter
- ☐ Other Enclosure(s) *(please identify below)*:

ED
AUG - 5 2002
Technology Center 2100

Remarks	**Please charge the $138 additional claim fee (6 total claims at $9 each; 2 independent claims at $42 each) and any other fees due to Applicants' Attorneys' Deposit Account No. 20-0782. A duplicate copy of this transmittal is enclosed to facilitate the charge.**

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm *or* Individual name	KIN-WAH TONG, REG. NO. 39,400
Signature	K.W. Tong
Date	August 5, 2002

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be send to the Chief Information Officer, U.S. Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Washington, DC 20231.

Notice of Allowability	Application No.	Applicant(s)
	09/524,095	HALVERSON ET AL.
	Examiner	**Art Unit**
	Firmin Backer	3621

-- 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 *August 7th, 2002*.
2. ☒ The allowed claim(s) is/are *56-187*.
3. ☐ The drawings filed on _____ 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: _____.
5. ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).
   (a) ☐ The translation of the foreign language provisional application has been received.
6. ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

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.**

7. ☐ 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.

8. ☒ CORRECTED DRAWINGS 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. ____.
   (b) ☐ including changes required by the proposed drawing correction filed _____, which has been approved by the Examiner.
   (c) ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No. _____.

Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the top margin (not the back) of each sheet. The drawings should be filed as a separate paper with a transmittal letter addressed to the Official Draftsperson.

9. ☐ 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)
3☐ Notice of Draftperson's Patent Drawing Review (PTO-948)
5☒ Information Disclosure Statements (PTO-1449), Paper No. 13.
7☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material
2☐ Notice of Informal Patent Application (PTO-152)
4☐ Interview Summary (PTO-413), Paper No.____ .
6☐ Examiner's Amendment/Comment
8☒ Examiner's Statement of Reasons for Allowance
9☐ Other .

Response to Amendment

This is in response to an amendment file on August 7th, 2002. Claims 56, 82 and 101 have been amended and claims 127-187 have been added. Claims 56-187 are pending in the letter.

Allowable Subject Matter

1. Claims 56-187 are allowed.

2. The following is an examiner's statement of reasons for allowance:

a. Applicants teach an inventive concept for navigating network-based electronic data sources in response to spoken natural language input request. Applicants' inventive concept if novel and innovative in the sense that upon emerging of error or ambiguities in the interpretation of the spoken natural language, the system solicits additional input for the user in non-spoken modality that is different from the original request without requiring the user to request the non-spoken modality.

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."

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Firmin Backer whose telephone number is (703) 305-0624. The examiner can normally be reached on Mon-Thu 8:30-6:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, James Trammell can be reached on (703) 305-9768. The fax phone numbers for the organization where this application or proceeding is assigned are (703) 305-7687 for regular communications and (703) 305-7687 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) 308-1113.

Firmin Backer
November 21, 2002

JAMES P. TRAMMELL
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 3600

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

#31

7590 12/16/2002

PERKINS COIE LLP
101 JEFFERSON DRIVE
MENLO PARK, CA 94025-1114

EXAMINER	
BACKER, FIRMIN	
ART UNIT	CLASS-SUBCLASS
3621	709-218000

DATE MAILED: 12/16/2002

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/524,095	03/13/2000	Christine Halverson	SRI1P037	6294

TITLE OF INVENTION: NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK

APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	YES	$640	$0	$640	03/17/2003

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 is changed, pay the PUBLICATION FEE (if required) and twice the amount of the ISSUE FEE shown above and notify the United States Patent and Trademark Office of the change in status, 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 the box below and enclose the PUBLICATION FEE and 1/2 the ISSUE FEE shown above.

☐ Applicant claims SMALL ENTITY status.
See 37 CFR 1.27.

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 Box 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 **Box ISSUE FEE**
Commissioner for Patents
Washington, D.C. 20231
Fax (703)746-4000

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 4 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: Legibly mark-up with any corrections or use Block 1)

7590 12/16/2002

PERKINS COIE LLP
101 JEFFERSON DRIVE
MENLO PARK, CA 94025-1114

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 Box Issue Fee address above, or being facsimile transmitted to the USPTO, on the date indicated below.

(Depositor's name)

(Signature)

(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/524,095	03/13/2000	Christine Halverson	SRI1P037	6294

TITLE OF INVENTION: NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK

APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	YES	$640	$0	$640	03/17/2003

EXAMINER	ART UNIT	CLASS-SUBCLASS
BACKER, FIRMIN	3621	709-218000

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. Inclusion of assignee data is only appropriate when an assignment has been previously submitted to the USPTO or is being submitted under separate cover. 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

❑ 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 Commissioner is hereby authorized by charge the required fee(s), or credit any overpayment, to Deposit Account Number ____________(enclose an extra copy of this form).

Commissioner for Patents 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.

(Authorized Signature) (Date)

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.

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, Washington, D.C. 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, Washington, DC 20231.

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.

TRANSMIT THIS FORM WITH FEE(S)

PTOL-85 (REV. 04-02) Approved for use through 01/31/2004. OMB 0651-0033 U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/524,095	03/13/2000	Christine Halverson	SRI1P037	6294

7590 12/16/2002

PERKINS COIE LLP
101 JEFFERSON DRIVE
MENLO PARK, CA 94025-1114
UNITED STATES

EXAMINER	
BACKER, FIRMIN	
ART UNIT	**PAPER NUMBER**
3621	

DATE MAILED: 12/16/2002

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 days. 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) system. (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.

PTOL-85 (REV. 04-02) Approved for use through 01/31/2004.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/524,095	03/13/2000	Christine Halverson	SRI1P037	6294

7590 12/16/2002

PERKINS COIE LLP
101 JEFFERSON DRIVE
MENLO PARK, CA 94025-1114
UNITED STATES

EXAMINER	
BACKER, FIRMIN	
ART UNIT	**PAPER NUMBER**
3621	

DATE MAILED: 12/16/2002

Notice of Fee Increase on January 1, 2003

If a reply to a "Notice of Allowance and Fee(s) Due" is filed in the Office on or after January 1, 2003, then the amount due will be higher than that set forth in the "Notice of Allowance and Fee(s) Due" since there will be an increase in fees effective on January 1, 2003. See Revision of Patent and Trademark Fees for Fiscal Year 2003; Final Rule, 67 Fed. Reg. 70847, 70849 (November 27, 2002).

The current fee schedule is accessible from: http://www.uspto.gov/main/howtofees.htm.

If the issue 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 to Pay Balance of Issue Fee" will be mailed to applicant. In order to avoid processing delays associated with mailing of a "Notice to Pay Balance of Issue Fee," if the response to the Notice of Allowance and Fee(s) due form is to be filed on or after January 1, 2003 (or mailed with a certificate of mailing on or after January 1, 2003), the issue fee paid should be the fee that is required at the time the fee is paid. 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 Manual of Patent Examining Procedure, Section 1308.01 (Eighth Edition, August 2001).

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.

(2)

UW

09/524,095

#32
CA
4-25-03

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

PATENT APPLICATION

Applicant: **Halverson et al.**

Case: **SRI1P037**

Serial No.: **09/524,095** Filed: **March 13, 2000**

Group Art Unit: **3621**

Examiner: **Firmin Backer**

Title: **NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK**

ASSISTANT COMMISSIONER FOR PATENTS
Box Issue Fee
Washington, D. C. 20231

S I R:

Comments on Statement of Reasons for Allowance

This response addresses the Notice of Allowance dated December 16, 2002.

REMARKS

Applicants' representative would like to thank Examiner Firmin Backer for kindly allowing claims 56-187 of the present application. However, Applicants have reviewed the Examiner's Reasons for Allowance and have the following comments:

1. The Examiner stated that:

> "Applicants teach an inventive concept for navigating network-based electronic data sources in response to spoken natural language input request. Applicants' inventive concept **[If]** **<u>is</u>** novel and innovative in the sense that upon emerging of error or ambiguities in the interpretation of the spoken natural language, the system solicits additional input for the user in non-spoken modality that is different from the original request without requiring the user to request the non-spoken modality." (Emphasis and correction added)

It appears that there is a typographical error in the second sentence where the Examiner used the term "if" instead of "is". It is Applicants' interpretation that the Examiner intended to use the term "is". If the Examiner disagrees, it is respectfully requested that the Examiner resolve the ambiguity of the sentence.

Conclusion

Thus, the Applicants submit the present comments solely to clarify various issues raised by the Notice of Allowance. Once again, Applicants' representative would like to thank Firmin Backer for kindly allowing claims 56-187of the present application.

If, however, the Examiner believes that there are any unresolved issues, it is requested that the Examiner telephone Mr. Kin-Wah Tong, Esq. at (732) 530-9404 so that appropriate arrangements can be made for resolving such issues as expeditiously as possible.

Respectfully submitted,

3/17/03

Kin-Wah Tong, Attorney
Reg. No. 39,400
(732) 530-9404

Moser, Patterson & Sheridan, LLP
595 Shrewsbury Avenue
First Floor, Suite 100
Shrewsbury, New Jersey 07702

OIPE JC109
AUG 13 2002
PATENT & TRADEMARK OFFICE

2/53
#33/
5/2/03
5/rne

Attorney Docket 59501-8037.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: Assistant Commissioner for Patents, Washington, D.C., 20231, on:

Date: August 6, 2002

By: Jamie L. Hughes
Jamie L. Hughes

Match and Return

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF:

Halverson

APPLICATION NO.: 09/524,095

FILED: 03/13/2000

FOR: **NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK**

EXAMINER: BACKER

ART UNIT: 2155

RECEIVED
AUG 15 2002
Technology Center 2100

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

Assistant Commissioner for Patents
Washington, D.C. 20231

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.

08/14/2002 SMINASS1 00000008 502207 09524095
01 FC:126 180.00 CH

Match and Return

2. Cited Information

☒ Copies of the following references are enclosed:

☒ All cited references

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 elects to pay the fee under 37 CFR 1.17(p) $180.00.

☐ Check enclosed for $.
☒ Please charge the above fee(s) to Deposit Account No. 50-2207 this paper is provided in triplicate.

Respectfully submitted,
Perkins Coie LLP

Date: 6 Aug 2002

Brian R. Coleman
Registration No. 39,145

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

OIPE JC09 AUG 1 3 2002 PATENT & TRADEMARK OFFICE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
Form PTO-1449 (Modified)
(Use several sheets if necessary)

Sheet	1	of	3

COMPLE... KNOWN	
Application Number	09/542,09.
Confirmation Number	
Filing Date	March 13, 2000
First Named Inventor	Halverson
Group Art Unit	2155
Examiner Name	Backer
Attorney Docket No.	59501-8037.US01

RECEIVED AUG 1 5 2002 Technology Center 2100

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
F.B	1	5,197,005		Schwartz et al.	3/23/93	
	2	5,386,556		Hedin et al.	1/31/95	
	3	5,434,777		Luciw	7/18/95	
	4	5,519,608		Kupiec	5/21/96	
	5	5,608,624		Luciw	3/4/97	
	6	5,721,938		Stuckey	2/24/98	
	7	5,729,659		Potter	3/17/98	
	8	5,748,974		Johnson	5/5/98	
	9	5,774,859		Houser et al.	6/30/98	
F.B	10	5,794,050		Dahlgren et al.	8/11/98	

FOREIGN PATENT DOCUMENTS

Examiner Initial	Cite No.	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
F.B	11	WO	00/11869		Ellis et al.	3/2/00		
F.B	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
F.B	13	Dowding, John et al., "Gemini: A Natural Language System For Spoken-Language Understanding", SRI International	
	14	http://www.ai.sri.com/~oaa/infowiz.html, "InfoWiz: An Animated Voice Interactive Information System, May 8, 2000	
	15	Dowding, John, "Interleaving Syntax and Semantics in an Efficient Bottom-up Parser", SRI International	
F.B	16	Moore, Robert et al., "Combining Linguistic and Statistical Knowledge Sources in a Natural-Language Processing for ATIS", SRI International	

EXAMINER	DATE CONSIDERED
Firmin Backer	8/30/02

*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).

OIPE JC109 AUG 1 3 2002 PATENT & TRADEMARK OFFICE

RECEIVED AUG 1 5 2002 Technology Center 2100

INFORMATION DISCLOSURE STATEMENT BY APPLICANT Form PTO-1449 (Modified) (Use several sheets if necessary)	COMPLETE IF KNOWN	
	Application Number	09/542,095
	Confirmation Number	
	Filing Date	March 13, 2000
	First Named Inventor	Halverson
	Group Art Unit	2155
	Examiner Name	Backer
Sheet 2 of 3	Attorney Docket No.	59501-8037.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
F.B	17	5,802,526		Fawcett et al.	9/1/98	
	18	6,192,338		Haszto et al.	2/2001	
	19	6,173,279		Levin et al.	1/2001	
	20	5,805,775		Eberman et al.	9/8/98	
	21	5,855,002		Armstrong	12/29/98	
	22	5,890,123		Brown et al.	3/30/99	
	23	5,963,940		Liddy et al.	10/5/99	
	24	6,003,072		Gerritsen et al	12/14/99	
	25	6,012,030		French-St. George et al.	1/4/00	
	26	6,026,388		Liddy et al.	2/15/00	
	27	6,080,202		Strickland et al.	6/27/00	
F.B	28	6,021,427		Spagna et al.	1/1/00	

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
F.B	29	Stent, Amanda et al., "The CommandTalk Spoken Dialog System", SRI International	
	30	Moore, Robert et al., "CommandTalk: A Spoken-Language Interface for Battlefield Simulations:, October 23, 1997, SRI International	
	31	Dowding, John et al., "Interpreting Language in Context in CommandTalk", February 5, 1999, SRI International	
	32	Moran, Douglas B. et al., "Intelligent Agent-based User Interfaces", Article Intelligence center, SRI International	
F.B	33	Martin, David L. et al., "Building Distributed Software Systems with the Open Agent Architecture"	

EXAMINER	DATE CONSIDERED
Francis Backer	8/30/02

*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).

	COMPLETE IF KNOWN
Application Number	09/342,095
Confirmation Number	
Filing Date	March 13, 2000
First Named Inventor	Halverson
Group Art Unit	2155
Examiner Name	Backer
Attorney Docket No.	59501-8037.US01

OIPE
AUG 13 2002
PATENT & TRADEMARK OFFICE

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
Form PTO-1449 (Modified)
(Use several sheets if necessary)

Sheet 3 of 3

Technology Center 2100

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
F.B.	34	6,338,081		Furusawa et al.	1/8/02	
	35	6,144,989		Hodjat et al.	11/7/00	
F.B.	36	6,226,666		Chang et al.	5/1/01	

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
F.B.	37	Julia, Luc. et al., "Cooperative Agents and Recognition System (CARS) for Drivers and Passengers", SRI International	
	38	Moran, Douglas et al., "Multimodal User Interfaces in the Open Agent Architecture"	
	39	Cheyer, Adam et al., "Multimodal Maps: An Agent-based Approach", SRI International	
	40	Cutkosky, Mark R. et al., "An Experiment in Integrating Concurrent Engineering Systems"	
	41	Martin, David et al., "Development Tools for the Open Agent Architecture", The Practical Application of Intelleigent Agents and Multi-Agent Technology (PAAM96), London, April 1996	
	42	Cheyer, Adam et al.,"The Open Agent Architecturetm", SRI International, AI center	
	43	Dejima, Inc., http://www.dejima.com/	
	44	Cohen, Philip et al., "An Open Agent Architecture", AAAI Spring Symposium, pp1-8, March 1994	
F.B.	45	Martin, David et al., "Information Brokering in an Agent Architecture", Proceeding of the 2nd Int'l Conference on Practical Application of Intelligent Agents & Multi-Agent Technology, London, April 1997	

EXAMINER	DATE CONSIDERED
Frantz Jules	8/30/02

*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).

BY022170

OIPE JC64
MAR 2 4 2003
PATENT & TRADEMARK OFFICE

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

#35 MK

In re application of: Halverson, et al.
Serial No.: 09/524,095 Art Unit: 3621
Filing Date: March 13, 2000 Examiner: Backer, Firmin
For: NAVIGATING NETWORK-BASED ELECTRONIC INFOMRATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK
Docket No. SRI 4116-3

Assistant Commissioner for Patents
Washington, D.C. 20231
S I R:

SUBMISSION OF FORMAL DRAWINGS

The Applicants submit herewith 7 sheets of formal drawings (FIGS. 1 through 6), properly labeled, in connection with the above-captioned application. The Examiner is requested to substitute these formal drawings for the informal drawings previously submitted.

Respectfully submitted,

Dated: March 17, 2003

KIN-WAH TONG
Reg. No. 39,400
(732) 530-9404

Moser, Patterson & Sheridan, LLP
595 Shrewsbury Avenue
Suite 100
Shrewsbury, NJ 07702

CERTIFICATE OF MAILING under 37 C.F.R. 1.8(a)

I hereby certify that this correspondence is being deposited on March 17, 2003, with the United States Postal Service as first class mail, with sufficient postage, in an envelope addressed to the Commissioner for Patents, Box Issue Fee, Washington, D.C. 20231.

Linda DeNardi
Signature

March 17, 2003
Date of signature

Halverson, et al.
"NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK"
Serial No. 09/524,095 - SRI 4116-3/ KWT

09524095

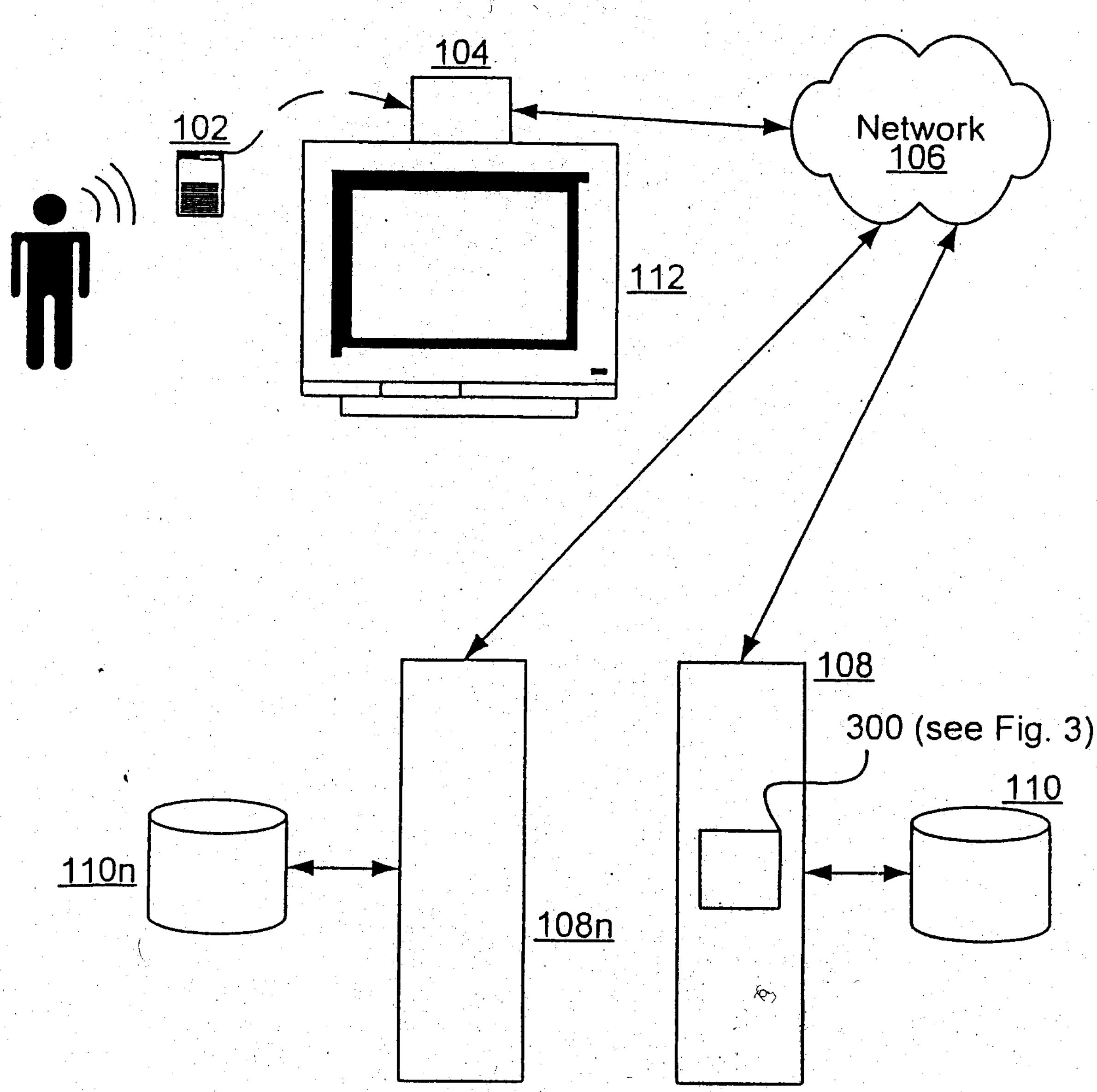


Fig. 1a

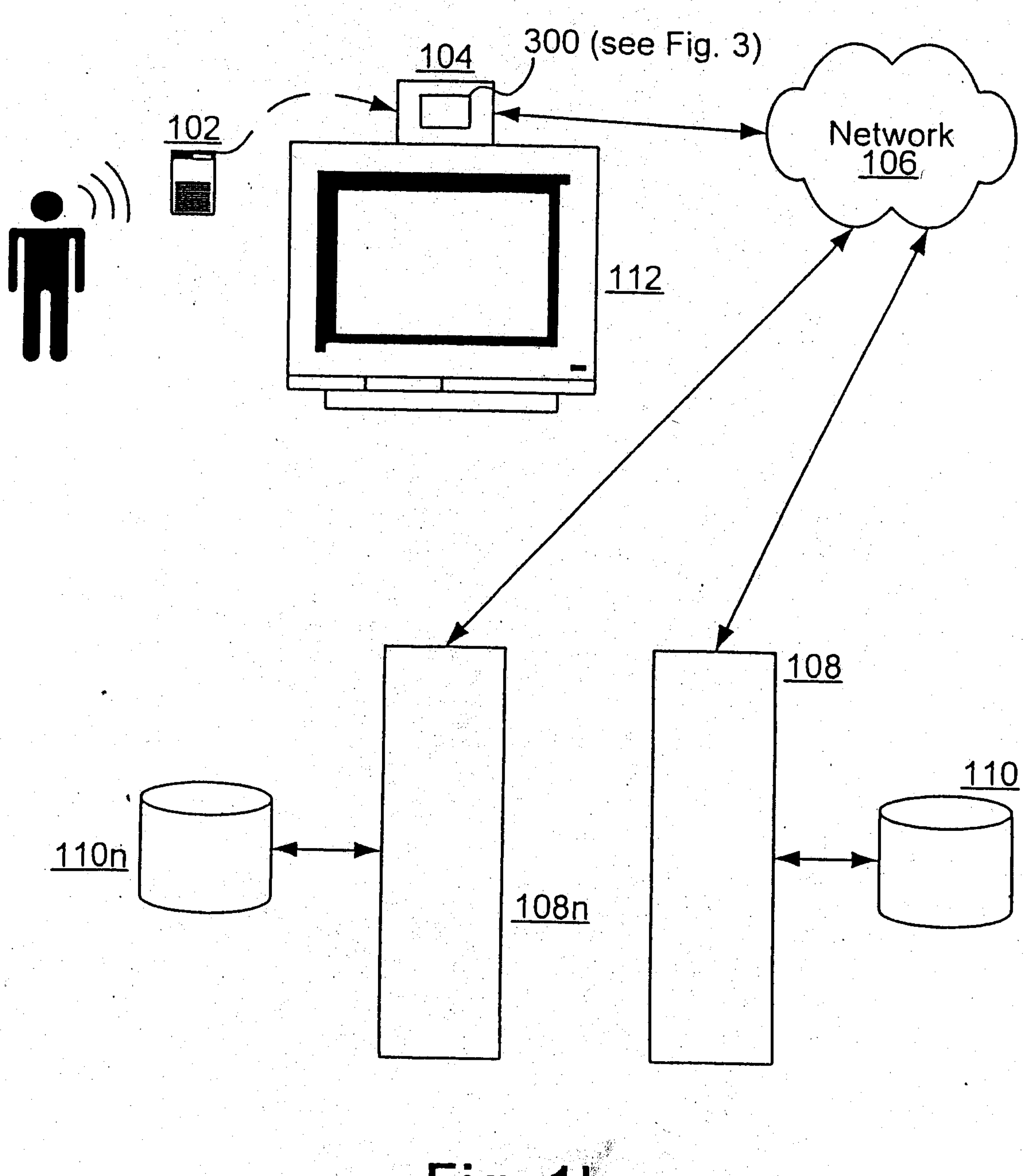


Fig. 1b

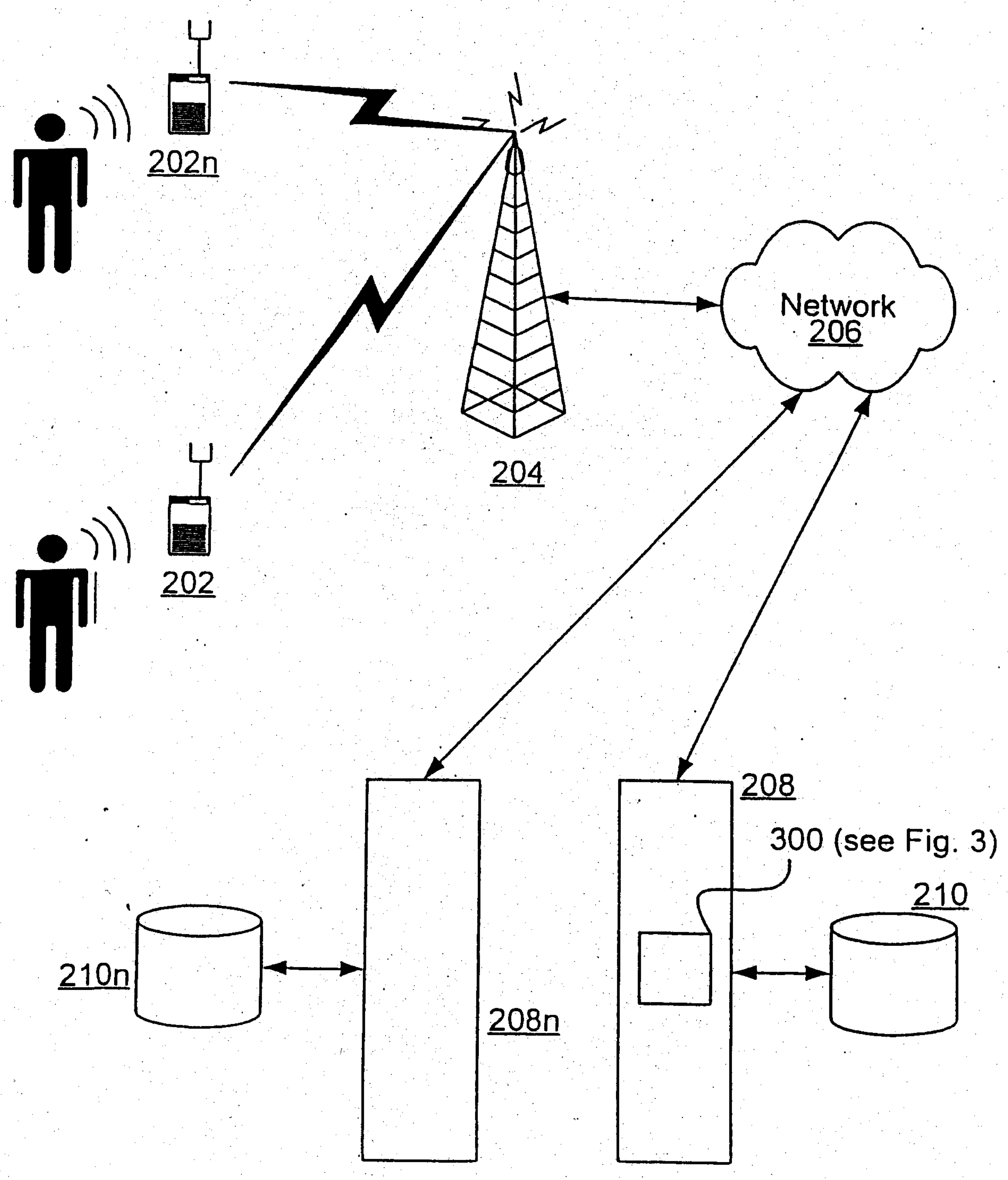


Fig. 2

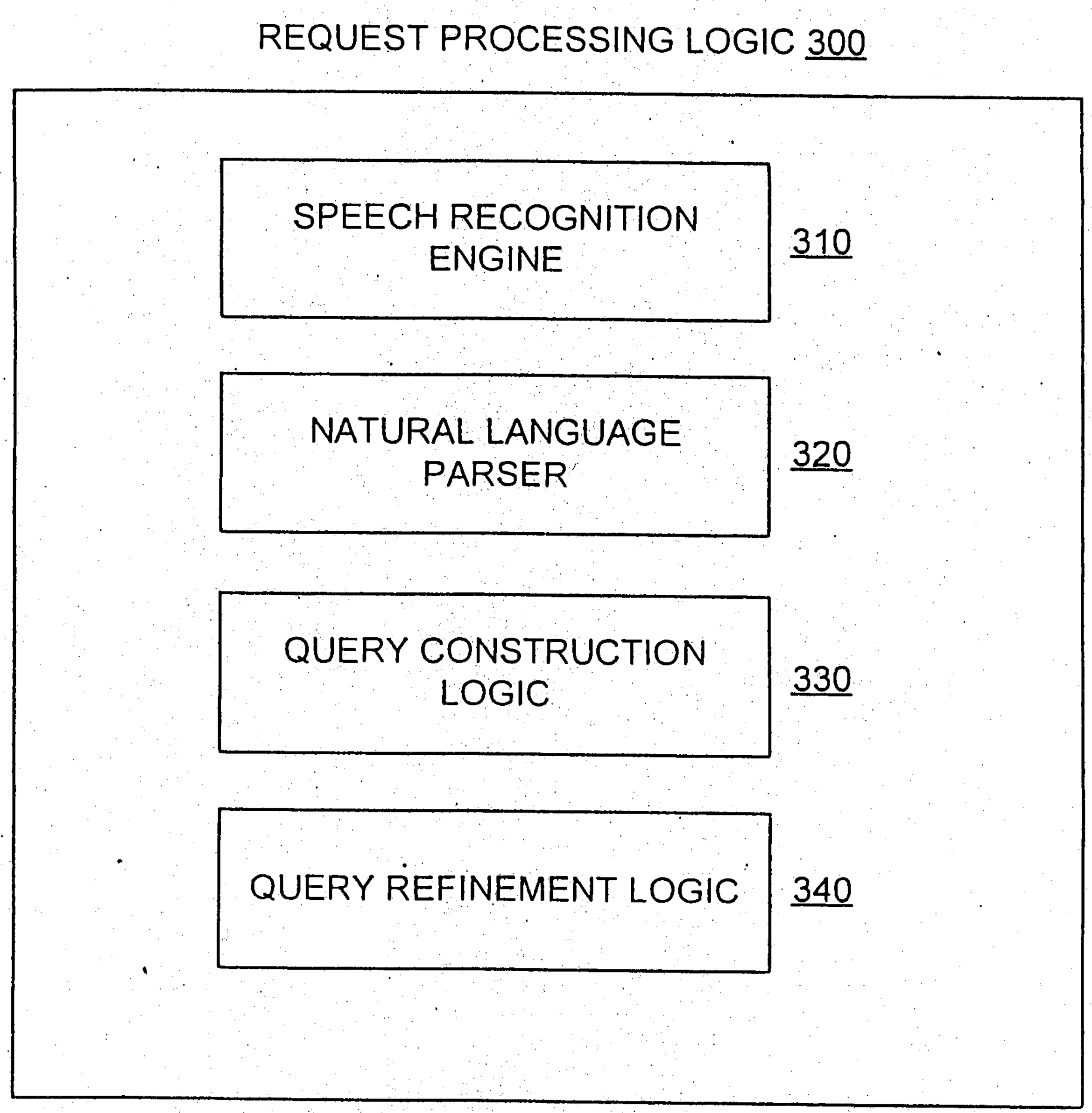

REQUEST PROCESSING LOGIC 300
SPEECH RECOGNITION ENGINE
310
NATURAL LANGUAGE PARSER
320
QUERY CONSTRUCTION LOGIC
330
QUERY REFINEMENT LOGIC
340

Fig. 3

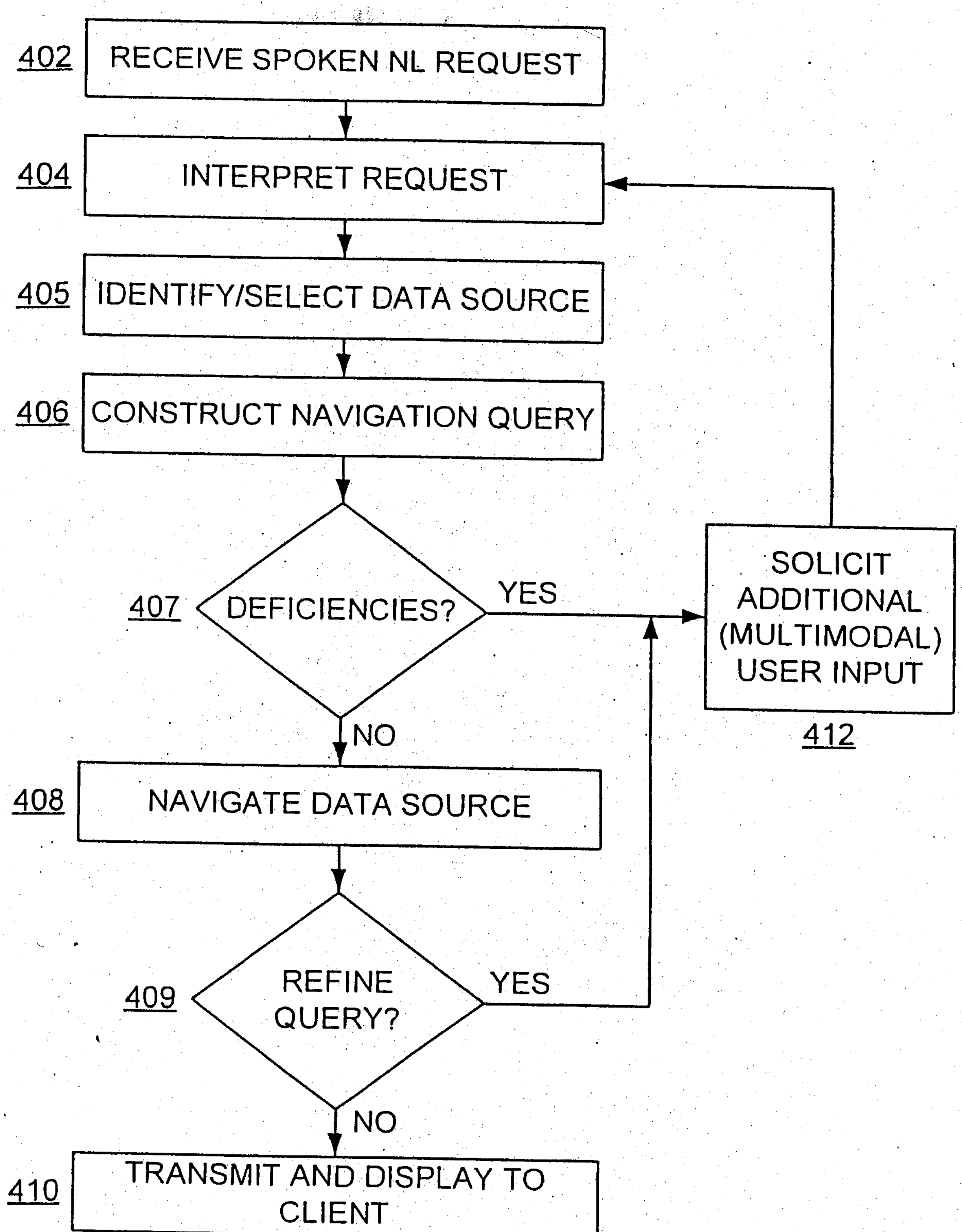

402 RECEIVE SPOKEN NL REQUEST
404 INTERPRET REQUEST
405 IDENTIFY/SELECT DATA SOURCE
406 CONSTRUCT NAVIGATION QUERY
407 DEFICIENCIES?
YES
NO
SOLICIT ADDITIONAL (MULTIMODAL) USER INPUT
412
408 NAVIGATE DATA SOURCE
409 REFINE QUERY?
YES
NO
410 TRANSMIT AND DISPLAY TO CLIENT

Fig. 4

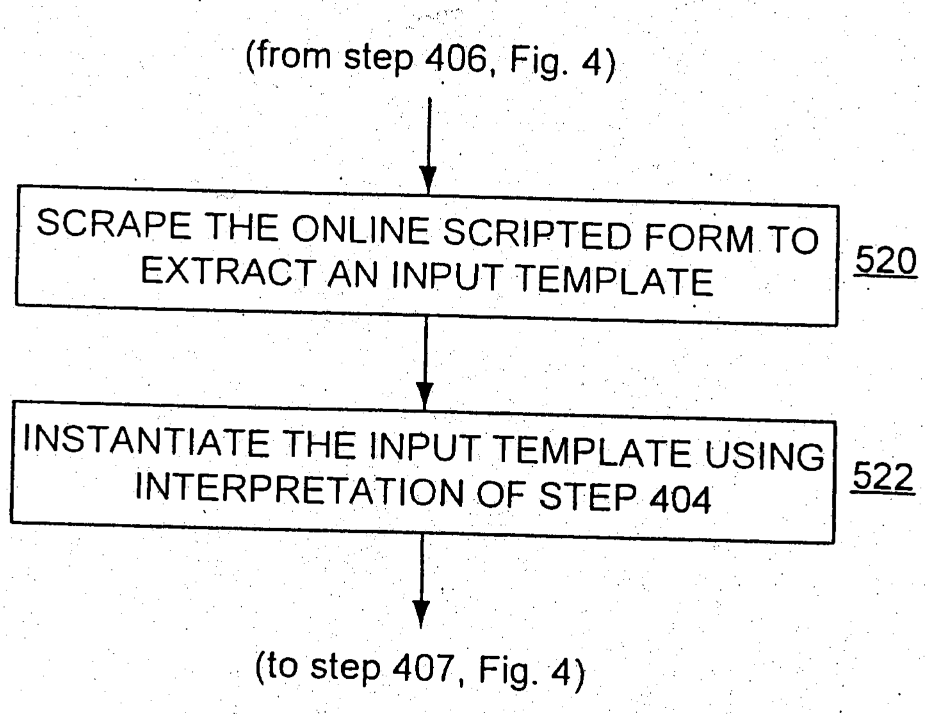

(from step 406, Fig. 4)
SCRAPE THE ONLINE SCRIPTED FORM TO EXTRACT AN INPUT TEMPLATE
520
INSTANTIATE THE INPUT TEMPLATE USING INTERPRETATION OF STEP 404
522
(to step 407, Fig. 4)

Fig. 5

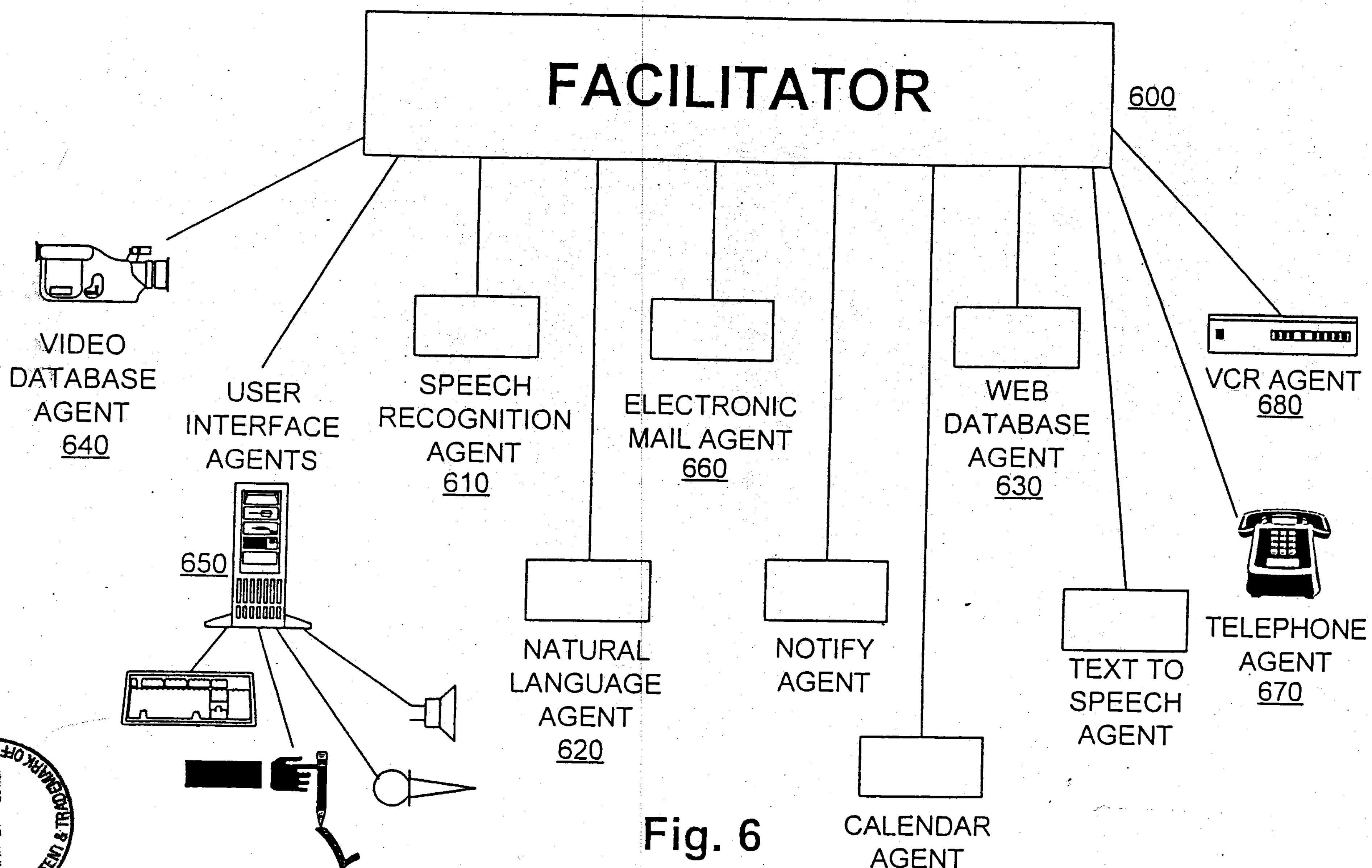

FACILITATOR
600
VIDEO DATABASE AGENT 640
USER INTERFACE AGENTS
650
SPEECH RECOGNITION AGENT 610
NATURAL LANGUAGE AGENT 620
ELECTRONIC MAIL AGENT 660
NOTIFY AGENT
CALENDAR AGENT
WEB DATABASE AGENT 630
TEXT TO SPEECH AGENT
VCR AGENT 680
TELEPHONE AGENT 670

Fig. 6

CW

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), to: Mail Box ISSUE FEE
Commissioner for Patents
Washington, D.C. 20231
Fax (703)746-4000

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 4 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: Legibly mark-up with any corrections or use Block 1)

7590 12/16/2002

PERKINS COIE LLP
101 JEFFERSON DRIVE
MENLO PARK, CA 94025-1114

03/28/2003 HMARZI2 00000050 200782 09524095

01 FC:2501 650.00 CH
02 FC:8001 3.00 CH

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 Box Issue Fee address above, or being facsimile transmitted to the USPTO, on the date indicated below.

LINDA DENARDI (Depositor's name)
Linda DeNardi (Signature)
MARCH 17, 2003 (Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/524,095	03/13/2000	Christine Halverson	SRI1P037	6294

TITLE OF INVENTION: NAVIGATING NETWORK-BASED ELECTRONIC INFORMATION USING SPOKEN NATURAL LANGUAGE INPUT WITH MULTIMODAL ERROR FEEDBACK

APPLN. TYPE	SMALL ENTITY	ISSUE FEE	PUBLICATION FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	YES	$640	$0	$640	03/17/2003

EXAMINER	ART UNIT	CLASS-SUBCLASS
BACKER, FIRMIN	3621	709-218000

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 Moser, Patterson & Sheridan, LLP.
2 Kin-Wah Tong, Esq.
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. Inclusion of assignee data is only appropriate when an assignment has been previously submitted to the USPTO or is being submitted under separate cover. Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE: SRI International, INc.

(B) RESIDENCE: (CITY and STATE OR COUNTRY): 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
☒ Advance Order - # of Copies 1

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 Commissioner is hereby authorized by charge the required fee(s), or credit any overpayment, to Deposit Account Number 20-0782 (enclose an extra copy of this form).

Commissioner for Patents 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.

(Authorized Signature) K-W T (Date) 3/17/03

Kin-Wah Tong, Esq.

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.

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, Washington, D.C. 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, Washington, DC 20231.

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.

TRANSMIT THIS FORM WITH FEE(S)

PTOL-85 (REV. 04-02) Approved for use through 01/31/2004. OMB 0651-0033 U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

#36 PAT REV

*RETURN TO FMF - LOCATION 7540

QUERY CONTROL FORM			RTIS USE ONLY
Application No. 09/522,1095	Prepared by BOB		Tracking Number
Examiner-GAU BACKER, F. 3621	Date 3-12-04		Week Date
	No. of queries 1		

JACKET			
a. Serial No.	f. Foreign Priority	k. Print Claim(s)	p. PTO-1449
b. Applicant(s)	g. Disclaimer	l. Print Fig.	q. PTOL-85b
c. Continuing Data	h. Microfiche Appendix	m. Searched Column	r. Abstract
d. PCT	i. Title	n. PTO-270/328	s. Sheets/Figs
e. Domestic Priority	j. Claims Allowed	o. PTO-892	t. Other

SPECIFICATION
a. Page Missing
b. Text Continuity
c. Holes through Data
d. Other Missing Text
e. Illegible Text
f. Duplicate Text
g. Brief Description
h. Sequence Listing
i. Appendix
j. Amendments
k. Other

CLAIMS
a. Claim(s) Missing
b. Improper Dependency
c. Duplicate Numbers
d. Incorrect Numbering
e. Index Disagrees
f. Punctuation
g. Amendments
h. Bracketing
i. Missing Text
j. Duplicate Text
k. Other

MESSAGE PTO-1449 (A) not completed. Please sign, initial and or line through. Please resolve

RECEIVED
MAR 16 2004
Publishing Division
13

Thanks
initials BOB

RESPONSE

initials

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

PAYOR NUMBER
25696

P75M

OPPENHEIMER WOLFF & DONNELLY
P. O. BOX 10356
PALO ALTO CA 94303

DATE PRINTED

12/03/07

MAINTENANCE FEE REMINDER

According to the records of the U.S. Patent and Trademark Office (USPTO) the maintenance fee for the patent(s) listed below (for which the above address is on record as the fee address under 37 CFR 1.363) has not been paid within the six-month period set forth in 37 CFR 1.362(d). THE MAINTENANCE FEE MAY STILL BE PAID WITH THE APPLICABLE SURCHARGE SET FORTH IN 37 CFR 1.20(h), WITHIN THE SIX-MONTH GRACE PERIOD SET FORTH IN 37 CFR 1.362(e).

Unless payment of the maintenance fee and the applicable surcharge is received in the USPTO within the six-month grace period, THE PATENT WILL EXPIRE AS OF THE END OF THE GRACE PERIOD. 35 U.S.C. 41(b).

The total payment due is the amount required on the date the fee is paid (and not necessarily the amount indicated below). All USPTO fees (including maintenance fees) are subject to change. Customers should refer to the USPTO Web site (www.uspto.gov) or call the Maintenance Fee Branch at 571-272-6500 for the most current fee amounts for the correct entity status before submitting payment. The total payment due indicated below is based on the entity status according to current Office records (shown below).

Timely payment of the total payment due is required in order to avoid expiration of the patent. A maintenance fee payment can be timely made using the certificate of mailing or transmission procedure set forth in 37 CFR 1.8.

PATENT NUMBER	FEE AMT	MAINT. SURCHG	U.S. APPL NUMBER	PATENT ISSUE DATE	APPL. FILING DATE	PAY-MENT YEAR	SMALL ENTITY?	TOTAL PYMT DUE	ATTORNEY DOCKET NUMBER
[redacted]	930	130	09524095	05/25/04	03/13/00	4	NO	1060	SRI1P037

The maintenance fee and the applicable surcharge can be paid quickly and easily over the Internet at www.uspto.gov by electronic funds transfer (EFT), credit card, or USPTO deposit account payment methods. The mailing address for all maintenance fee payments <u>not electronically submitted over the Internet</u> is: U.S. Patent and Trademark Office, P.O. Box 979070, St. Louis, MO 63197-9000.

Direct any questions about this notice to: Mail Stop M Correspondence, Director of the United States Patent and Trademark Office, P.O. Box 1450, Alexandria, VA 22313-1450.

NOTE: This notice was automatically generated based on the amount of time that elapsed since the date a patent was granted. It is possible that the patent term may have ended or been shortened due to a terminal disclaimer that was filed in the application. Also, for any patent that issued from an application filed on or after June 8, 1995 containing a specific reference to an earlier filed application or applications under 35 U.S.C. 120, 121, or 365(c), the patent term ends 20 years from the date on which the earliest such application was filed, unless the term was adjusted or extended under 35 U.S.C. 154 or 156. Patentee should determine the relevant patent term for a patent before paying the maintenance fee.

AO 120 (Rev. 3/04)

TO: Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court Delaware on the following X Patents or Trademarks:

DOCKET NO.	DATE FILED	U.S. DISTRICT COURT
16cv945-RGA	10/13/2016	DISTRICT OF DELAWARE

PLAINTIFF	DEFENDANT
IPA Technologies, Inc.	Acer America Corp.

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA Technologies, Inc.
2 6,523,061	2/18/2003	IP A Technologies, Inc.
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT
See attached Notice of Dismissal

CLERK	(BY) DEPUTY CLERK	DATE
JOHN A. CERINO, CLERK OF COURT		10/28/2016

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 08/10)

TO: **Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court District of Delaware on the following

☐ Trademarks or ☑ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED	U.S. DISTRICT COURT
16cv00947	10/13/2016	District of Delaware

PLAINTIFF	DEFENDANT
IPA Tech.	Dell

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6742021		
2 6523061		
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 08/10)

TO: **Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court District Court of Delaware on the following

☐ Trademarks or ☑ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED	U.S. DISTRICT COURT
116cv00948	10/13/2016	District Court of Delaware

PLAINTIFF	DEFENDANT
IPA Tech	HP Inc

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6742021		
2 6523061		
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 3/04)

TO: Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court Delaware on the following X Patents or Trademarks:

DOCKET NO.	DATE FILED	U.S. DISTRICT COURT
16cv949-RGA	10/13/2016	DISTRICT OF DELAWARE

PLAINTIFF	DEFENDANT
IPA Technologies, Inc.	Toshiba America, Inc., et al.

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA Technologies, Inc.
2 6,523,061	2/18/2003	IP A Technologies, Inc.
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT
See attached Notice of Dismissal

CLERK	(BY) DEPUTY CLERK	DATE
JOHN A. CERINO, CLERK OF COURT		10/28/2016

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 3/04)

TO: **Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court Delaware on the following X Patents or Trademarks:

DOCKET NO.	DATE FILED	U.S. DISTRICT COURT
16cv946-RGA	10/13/2016	DISTRICT OF DELAWARE

PLAINTIFF	DEFENDANT
IPA Technologies, Inc.	ASUS Computer Internationa, et al.

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA Technologies, Inc.
2 6,523,061	2/18/2003	IP A Technologies, Inc.
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT
See attached Notice of Dismissal

CLERK	(BY) DEPUTY CLERK	DATE
JOHN A. CERINO, CLERK OF COURT		10/28/2016

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 08/10)

TO: **Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court for the District of Delaware on the following

☐ Trademarks or ☑ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED 12/9/2016	U.S. DISTRICT COURT for the District of Delaware
PLAINTIFF IPA TECHNOLOGIES INC.		DEFENDANT ALCO ELECTRONICS LTD., ET AL.

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA TECHNOLOGIES INC.
2 6,523,061	2/18/2003	IPA TECHNOLOGIES INC.
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 08/10)

TO: **Mail Stop 8** **Director of the U.S. Patent and Trademark Office** **P.O. Box 1450** **Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court for the District of Delaware on the following

☐ Trademarks or ☑ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED 1/10/2017	U.S. DISTRICT COURT for the District of Delaware
PLAINTIFF IPA TECHNOLOGIES INC.		DEFENDANT ZTE CORPORATION, ET AL.

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA TECHNOLOGIES INC.
2 6,523,061	2/18/2003	IPA TECHNOLOGIES INC.
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 08/10)

TO: Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court for the District of Delaware on the following

☐ Trademarks or ☑ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED 1/19/2017	U.S. DISTRICT COURT for the District of Delaware
PLAINTIFF IPA TECHNOLOGIES INC.		DEFENDANT SONY CORPORATION, ET AL.

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA TECHNOLOGIES INC.
2 6,523,061	2/18/2003	IPA TECHNOLOGIES INC.
3 6,757,718	6/29/2004	IPA TECHNOLOGIES INC.
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	**DATE OF PATENT OR TRADEMARK**	**HOLDER OF PATENT OR TRADEMARK**
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 08/10)

TO: **Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court for the District of Delaware on the following

☐ Trademarks or ☑ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED 2/3/2017	U.S. DISTRICT COURT for the District of Delaware
PLAINTIFF IPA TECHNOLOGIES INC.		DEFENDANT ACER INC., ET AL.

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA TECHNOLOGIES INC.
2 6,523,061	2/18/2003	IPA TECHNOLOGIES INC.
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	**DATE OF PATENT OR TRADEMARK**	**HOLDER OF PATENT OR TRADEMARK**
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 08/10)

TO: **Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court for the District of Delaware on the following

☐ Trademarks or ☑ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED 2/3/2017	U.S. DISTRICT COURT for the District of Delaware

PLAINTIFF	DEFENDANT
IPA TECHNOLOGIES INC.	ASUS COMPUTER INTERNATIONAL, ET AL.

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA TECHNOLOGIES INC.
2 6,523,061	2/18/2003	IPA TECHNOLOGIES INC.
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 08/10)

TO: **Mail Stop 8 Director of the U.S. Patent and Trademark Office P.O. Box 1450 Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court for the District of Delaware on the following

☐ Trademarks or ☑ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED 2/3/2017	U.S. DISTRICT COURT for the District of Delaware

PLAINTIFF	DEFENDANT
IPA TECHNOLOGIES INC.	LG ELECTRONICS INC., ET AL.

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA TECHNOLOGIES INC.
2 6,523,061	2/18/2003	IPA TECHNOLOGIES INC.
3		
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

AO 120 (Rev. 08/10)

TO: **Mail Stop 8** **Director of the U.S. Patent and Trademark Office** **P.O. Box 1450** **Alexandria, VA 22313-1450**	**REPORT ON THE FILING OR DETERMINATION OF AN ACTION REGARDING A PATENT OR TRADEMARK**

In Compliance with 35 U.S.C. § 290 and/or 15 U.S.C. § 1116 you are hereby advised that a court action has been filed in the U.S. District Court for the District of Delaware on the following

☐ Trademarks or ☑ Patents. (☐ the patent action involves 35 U.S.C. § 292.):

DOCKET NO.	DATE FILED	U.S. DISTRICT COURT
	3/17/2017	for the District of Delaware

PLAINTIFF	DEFENDANT
IPA TECHNOLOGIES INC.	NVIDIA CORPORATION

PATENT OR TRADEMARK NO.	DATE OF PATENT OR TRADEMARK	HOLDER OF PATENT OR TRADEMARK
1 6,742,021	5/25/2004	IPA TECHNOLOGIES INC.
2 6,523,061	2/18/2003	IPA TECHNOLOGIES INC.
3 6,757,718	6/29/2004	IPA TECHNOLOGIES INC.
4		
5		

In the above—entitled case, the following patent(s)/ trademark(s) have been included:

DATE INCLUDED	INCLUDED BY ☐ Amendment ☐ Answer ☐ Cross Bill ☐ Other Pleading	
PATENT OR TRADEMARK NO.	**DATE OF PATENT OR TRADEMARK**	**HOLDER OF PATENT OR TRADEMARK**
1		
2		
3		
4		
5		

In the above—entitled case, the following decision has been rendered or judgement issued:

DECISION/JUDGEMENT

CLERK	(BY) DEPUTY CLERK	DATE

Copy 1—Upon initiation of action, mail this copy to Director Copy 3—Upon termination of action, mail this copy to Director
Copy 2—Upon filing document adding patent(s), mail this copy to Director Copy 4—Case file copy

Claim Final	Claim Original	Date							
	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								

SYMBOLS

- ✓ .... Rejected
- = ... Allowed
- - (Through numeral) Canceled
- ÷ .... Restricted
- N .... Non-elected
- I .... Interference
- A .... Appeal
- O .... Objected

Claim Final	Claim Original	Date 11/30/02							
96	151	=							
97	152								
98	153								
99	154								
100	155								
101	156								
102	157								
103	158								
104	159								
105	160								
106	161								
107	162								
108	163								
109	164								
110	165								
111	166								
112	167								
113	168								
114	169								
115	170								
116	171								
117	172								
118	173								
119	174								
120	175								
121	176								
122	177								
123	178								
124	179								
125	180								
126	181								
127	182								
128	183								
129	184								
130	185								
131	186								
132	187	=							
	188								
	189								
	190								
	191								
	192								
	193								
	194								
	195								
	196								
	197								
	198								
	199								
	200								

(LEFT INSIDE)

101

PATENT APPLICATION FEE DETERMINATION RECORD

Effective December 29, 1999

Application or Docket Number: 09/524095

CLAIMS AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	SMALL ENTITY TYPE RATE	FEE	OR	OTHER THAN SMALL ENTITY RATE	FEE
BASIC FEE				345.00	OR		690.00
TOTAL CLAIMS	55 minus 20=	* 35	X$ 9=		OR	X$18=	630
INDEPENDENT CLAIMS	3 minus 3 =	*	X39=		OR	X78=	—
MULTIPLE DEPENDENT CLAIM PRESENT		N	+130=		OR	+260=	—
			TOTAL		OR	TOTAL	1320

* If the difference in column 1 is less than zero, enter "0" in column 2

CLAIMS AS AMENDED - PART II

AMENDMENT A

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE	ADDITIONAL FEE	OR	OTHER THAN SMALL ENTITY RATE	ADDITIONAL FEE
Total	* 71	Minus	**	=	X$ 9=		OR	X$18=	
Independent	* 3	Minus	***	=	X39=		OR	X78=	
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					+130=		OR	+260=	
					TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT B

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE	ADDITIONAL FEE	OR	OTHER THAN SMALL ENTITY RATE	ADDITIONAL FEE
Total	* 126	Minus	** 71	= 56	X$ 9=	495	OR	X$18=	
Independent	* 6	Minus	*** 3	= 3	X39=	117	OR	X78=	
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					+130=		OR	+260=	
					TOTAL ADDIT. FEE	612	OR	TOTAL ADDIT. FEE	

AMENDMENT C

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE	ADDITIONAL FEE	OR	OTHER THAN SMALL ENTITY RATE	ADDITIONAL FEE
Total	* 132	Minus	** 126	= 6	X$ 9=	54	OR	X$18=	
Independent	* 8	Minus	*** 6	= 2	X ~~39~~ 42 =	84	OR	X78=	
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					+130=		OR	+260=	
					TOTAL ADDIT. FEE	138	OR	TOTAL ADDIT. FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20."
***If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3."
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

FORM PTO-875
(Rev. 12/99)

Patent and Trademark Office, U.S. DEPARTMENT OF COMMERCE

PATENT APPLICATION FEE DETERMINATION RECORD Effective October 1, 2000	Application or Docket Number 09/524,095

RCE

CLAIMS AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	SMALL ENTITY TYPE ☐ RATE	FEE	OR	OTHER THAN SMALL ENTITY RATE	FEE
BASIC FEE				$355	OR		$740
TOTAL CLAIMS	minus 20=	*	X$ 9=		OR	X$18=	
INDEPENDENT CLAIMS	minus 3 =	*	X40=		OR	X84=	
MULTIPLE DEPENDENT CLAIM PRESENT			+135 =		OR	+280=	
			TOTAL		OR	TOTAL	740 - pd

* If the difference in column 1 is less than zero, enter "0" in column 2

CLAIMS AS AMENDED - PART II

AMENDMENT A

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE	ADDITIONAL FEE	OR	OTHER THAN SMALL ENTITY RATE	ADDITIONAL FEE
Total	*	Minus	**	=	X$ 9=		OR	X$18=	
Independent	*	Minus	***	=	X40=		OR	X84=	
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					+135 =		OR	280 -	
					TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT B

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE	ADDITIONAL FEE	OR	OTHER THAN SMALL ENTITY RATE	ADDITIONAL FEE
Total	*	Minus	**	=	X$ 9=		OR	X$18=	
Independent	*	Minus	***	=	X40=		OR	X84	
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					+135=		OR	+280=	
					TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

AMENDMENT C

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE	ADDITIONAL FEE	OR	OTHER THAN SMALL ENTITY RATE	ADDITIONAL FEE
Total	*	Minus	**	=	X$ 9=		OR	X$18=	
Independent	*	Minus	***	=	X40=		OR	X84	
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM					+135=		OR	+280	
					TOTAL ADDIT. FEE		OR	TOTAL ADDIT. FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20."
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3."
The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 1.

ISSUE SLIP STAPLE AREA (for additional cross references)

POSITION	INITIALS	ID NO.	DATE
FEE DETERMINATION	SO	71058	3/21/00
O.I.P.E. CLASSIFIER	RSD		4-1-2000
FORMALITY REVIEW		71476	5/6/00
RESPONSE FORMALITY REVIEW		71476	9/8/00

INDEX OF CLAIMS

✓ .... Rejected
= .... Allowed
— (Through numeral)... Canceled
÷ .... Restricted
N .... Non-elected
I .... Interference
A .... Appeal
O .... Objected

Final	Original	Date
	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	
	32	
	33	
	34	
	35	
	36	
	37	
	38	
	39	
	40	
	41	
	42	
	43	
	44	
	45	
	46	
	47	
	48	
	49	
	50	

Final	Original	4/9/01	10/2/01	5/3/02	11/21/02
	51				
	52				
	53				
	54				
	55				
1	56	✓	✓	✓	=
2	57				
3	58				
4	59				
5	60				
6	61				
7	62				
8	63				
9	64				
10	65				
11	66				
12	67				
13	68				
14	69				
15	70				
16	71				
17	72				
18	73				
19	74				
20	75				
21	76				
22	77				
23	78				
24	79				
25	80				
26	81				
27	82				
28	83				
29	84				
30	85				
31	86				
32	87				
33	88				
34	89				
35	90				
36	91				
37	92				
38	93				
39	94				
40	95				
41	96				
42	97				
43	98				
44	99				
45	100	✓	✓	✓	=

Final	Original	4/9/01	10/2/01	5/3/02	11/21/02
46	101	✓	✓	✓	=
47	102				
48	103				
49	104				
50	105				
51	106				
52	107				
53	108				
54	109				
55	110				
56	111				
57	112				
58	113				
59	114				
60	115				
61	116				
62	117				
63	118				
64	119				
65	120				
66	121				
67	122				
68	123				
69	124				
70	125				
71	126	✓	✓	✓	=
72	127				=
73	128				
74	129				
75	130				
76	131				
77	132				
78	133				
79	134				
80	135				
81	136				
82	137				
83	138				
84	139				
85	140				
86	141				
87	142				
88	143				
89	144				
90	145				
91	146				
92	147				
93	148				
94	149				
95	150				

If more than 150 claims or 10 actions
staple additional sheet here

(LEFT INSIDE)

SEARCHED

Class	Sub.	Date	Exmr.
709	218	4/6/01	F.B
707	5		
	4		
	102		
704	257		
	231		
709	218	11/21/02	F.B

SEARCH NOTES
(INCLUDING SEARCH STRATEGY)

	Date	Exmr.
West search	4/6/01	F.B.
West search	4/28/02	F.B
NPL search		
IBM disclosure		
West search	11/20/02	F.B
NPL	11/20/02	
West	11/21/02	

INTERFERENCE SEARCHED

Class	Sub.	Date	Exmr.
709	218	11/21/02	F.B
707	5	11	
	4		

US006851115B1

(12) United States Patent
Cheyer et al.

(10) **Patent No.: US 6,851,115 B1**
(45) **Date of Patent: Feb. 1, 2005**

(54) **SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS**

(75) Inventors: **Adam J. Cheyer**, Palo Alto, CA (US); **David L. Martin**, Santa Clara, CA (US)

(73) Assignee: **SRI International**, Menlo Park, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/225,198**

(22) Filed: **Jan. 5, 1999**

(51) **Int. Cl.**[7] **G06F 9/54**
(52) **U.S. Cl.** **719/317**; 709/202; 717/114
(58) **Field of Search** 719/317; 709/202, 709/207, 317; 717/114

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,638,494	A	*	6/1997	Pinard et al.	709/202
5,802,396	A	*	9/1998	Gray	710/20
5,960,404	A	*	9/1999	Chaar et al.	705/11
6,047,053	A		4/2000	Miner et al.	
6,212,649	B1	*	4/2001	Yalowitz et al.	714/31
6,216,173	B1	*	4/2001	Jones et al.	135/77
6,256,771	B1		7/2001	O'Neil et al.	
6,338,081	B1	*	1/2002	Furusawa et al.	709/202
6,411,684	B1		6/2002	Cohn et al.	
6,484,155	B1	*	11/2002	Kiss et al.	706/46
2001/0039562	A1	*	11/2001	Sato	709/202
2003/0167247	A1	*	9/2003	Masuoka	706/46

OTHER PUBLICATIONS

Cheyer, Adam. "Mechanisms of Cooperation." Oct. 19, 1998.*

DeVoe, Deborah. "SRI distributed agents promise flexibility." InfoWorld. Dec. 30, 1996.*

Sycara, Katia et al. "Distributed Intelligent Agents." IEEE. Dec. 1996.*

Nwana, Hyacinth et al. "Software Agent Technologies". BT Technology Journal. 1996.*

Busetta, Paolo et al. "The BDIM Agent Toolkit Design." 1997.*

Mayfield, James et al. "Desiderata for Agent Communication Languages." Mar. 27–29, 1995.*

Khedro, Taha et al. Concurrent Endineering through Interoperable Software Agents. Aug. 1994.*

Moran et al. "Multimodal User Interfaces in the Open Agent Architecture." Proceedings of the International Conference on Intelligent User Interfaces. 6–9/1997.*

Martin, David et al. "The Open Agent Architecture: A Framework for Building Distributed Software Systems." Oct. 19, 1998.*

Wilkins, David et al. "Multiagent Planning Architecture." SRI International. Dec. 8, 1997.*

Moran, Douglas B. and Cheyer, Adam J., "Intelligent Agent–based User Interfaces", Article Intelligence center, SRI International.

Martin, David L., Cheyer, Adam J. and Moran, Douglas B., "Building Distributed Software Systems with the Open Agent Architecture".

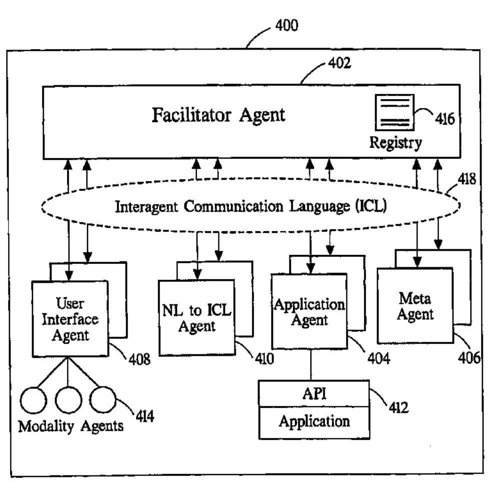
(List continued on next page.)

Primary Examiner—Lewis A. Bullock, Jr.
(74) *Attorney, Agent, or Firm*—Perkins Coie LLP

(57) **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.

89 Claims, 16 Drawing Sheets



OTHER PUBLICATIONS

Cohen, Philip R. and Cheyer, Adam, SRI International, Wang, Michelle, Stanford University, BAEG, Soon Cheol, ETRI, "An Open Agent Architecture".

Julia, Luc E. and Cheyer, Adam J., SRI International "Cooperative Agents and Recognition Systems (CARS) for Drivers and Passengers".

Moran, Douglas B., Cheyer, Adam J., Julia, Luc E., Martin, David L., SRI International, and Park, Sangkyu, Electronics and Telecommunications Research Institute, "Multimodal User Interfaces in the Open Agent Architecture".

Cheyer, Adam and Lulia, Luc, SRI International "Multimodal Maps: An Agent–based Approach".

Cutkosky, Mark R., Engelmore, Robert S., Fikes, Richard E., Genesereth, Michael R., Gruber, Thomas R., Stanford University, Mark, William, Lockheed Palo Alto Research Labs, Tenenbaum, Jay M., Weber, Jay C., Enterprise Integration Technologies, "An Experiment in Integrating Concurrent Engineering Systems".

Martin, David L., Cheyer, Adam, SRI International, LEE, Gowang–Lo, ETRI, "Development Tools for the Open Agent Architecture", The Practical Application of Intelligent Agents and Multi–Agent Technology (PAAM96), London, Apr. 1996.

Cheyer, Adam, Martin, David and Moran, Douglas, "The Open Agent architecture™", SRI International, AI Center.

Dejima, Inc., http://www.dejima.com/.

Cohen, Philip R, Cheyer, Adam, Wang, Michelle, Stanford University, BAEG, Soon Cheol ETRI: "An Open Agent Architecture," AAAI Spring Symposium, pp. 1–8, Mar. 1994.

Martin, David; Oohama, Hiroki; Moran, Douglas; Cheyer, Adam; "Information Brokering in an Agent Architecture," Proceeding of the 2nd International Conference on Practical Application of Intelligent Agents & Multi–Agent Technology, London, Apr. 1997.

* cited by examiner

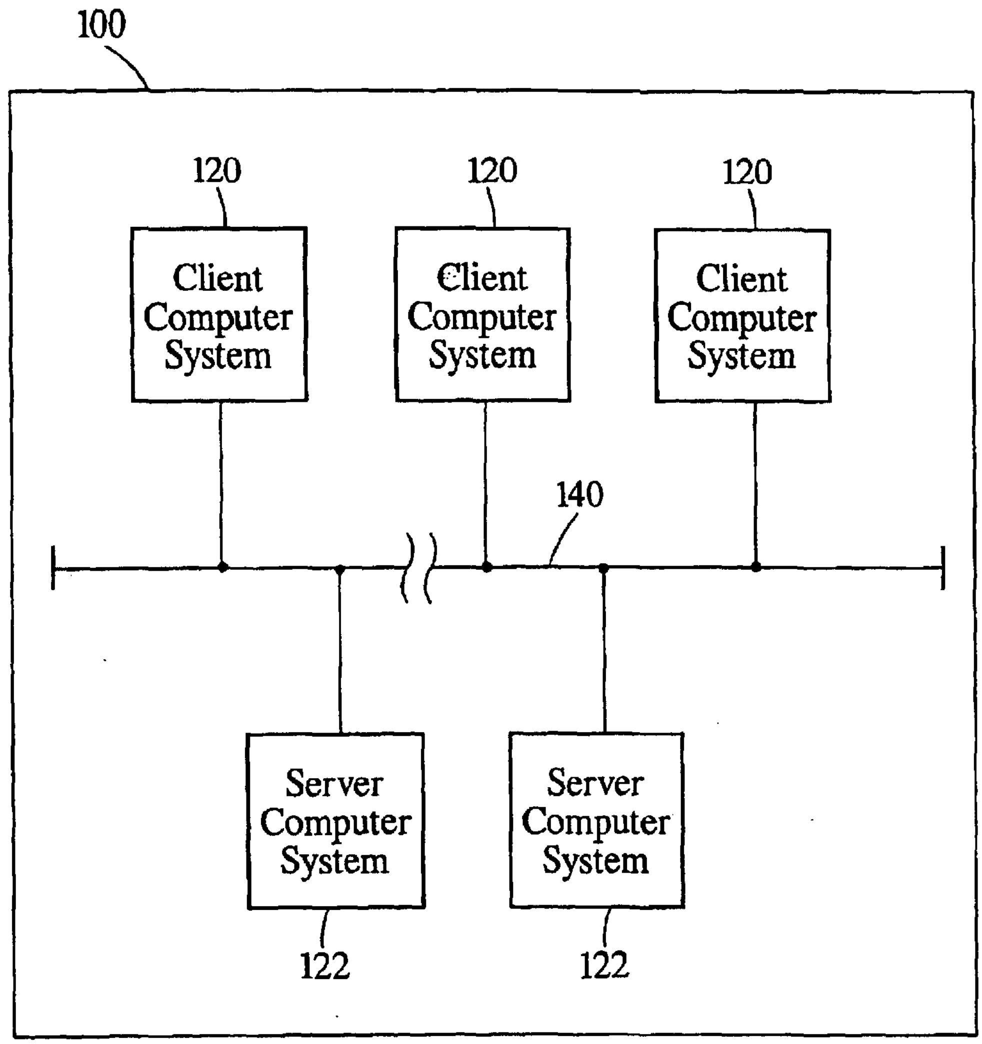


Fig. 1
(Prior Art)

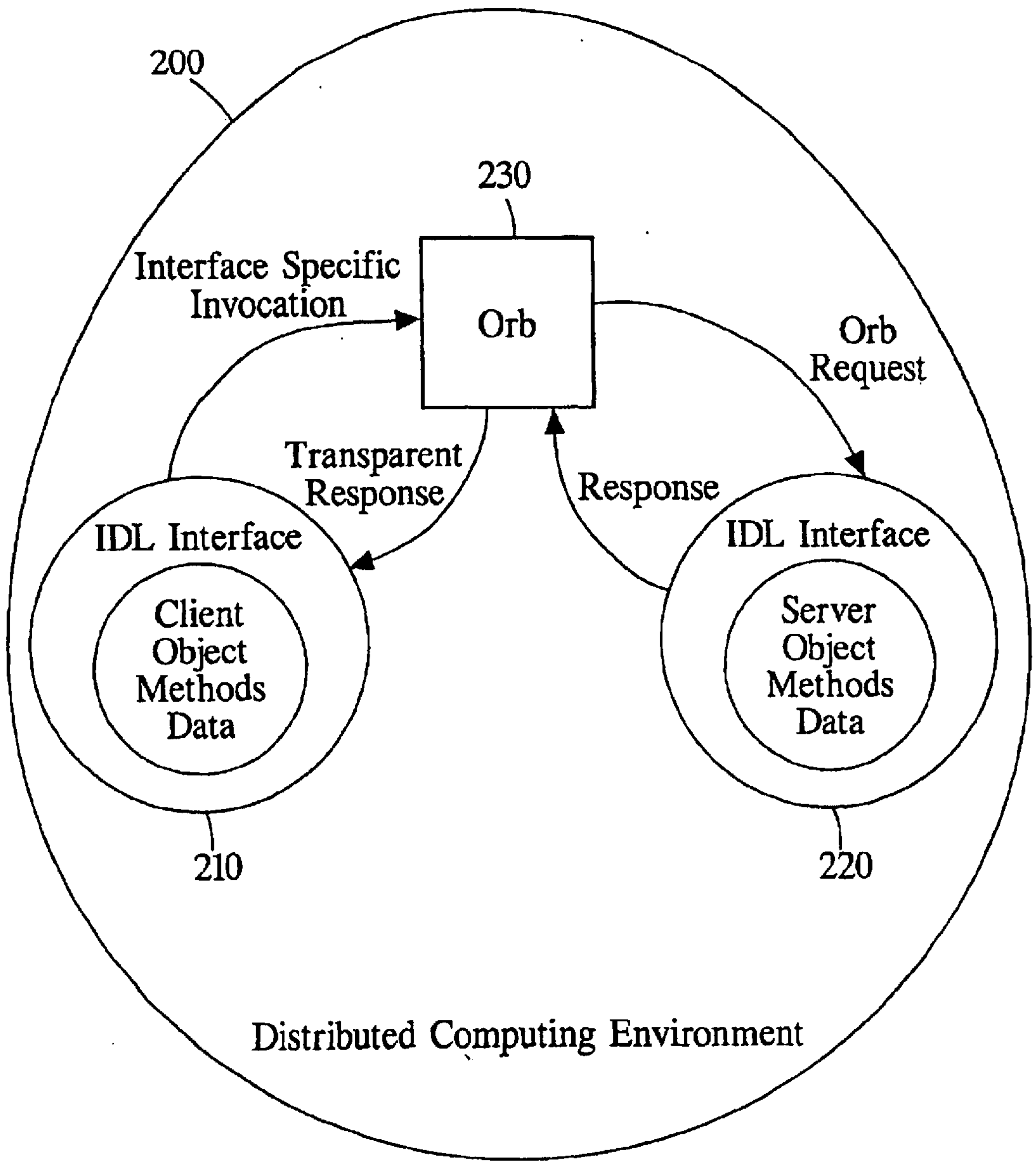


Fig. 2
(Prior Art)

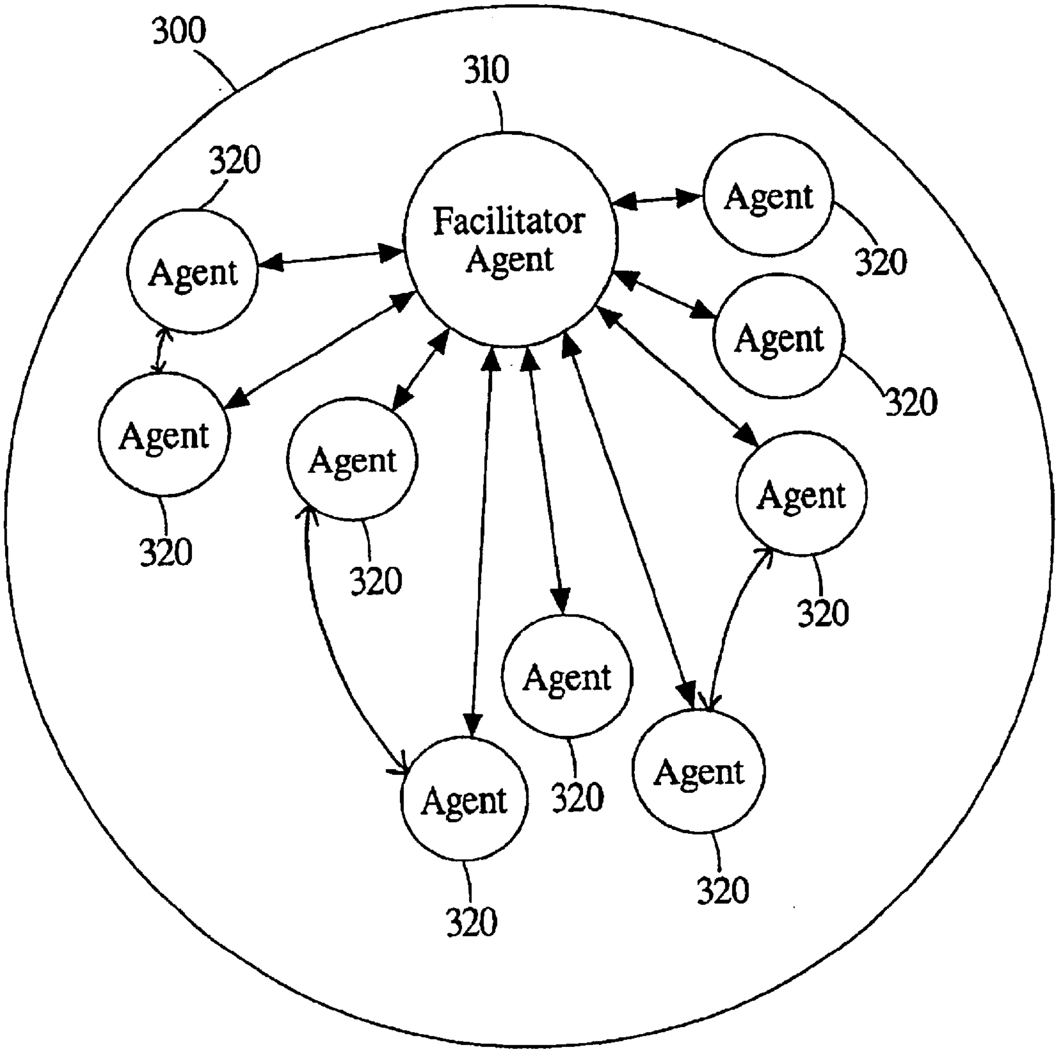

300
310
Facilitator
Agent
Agent
Agent
Agent
Agent
Agent
Agent
Agent
Agent
Agent
320
320
320
320
320
320
320
320
320
320

Fig. 3

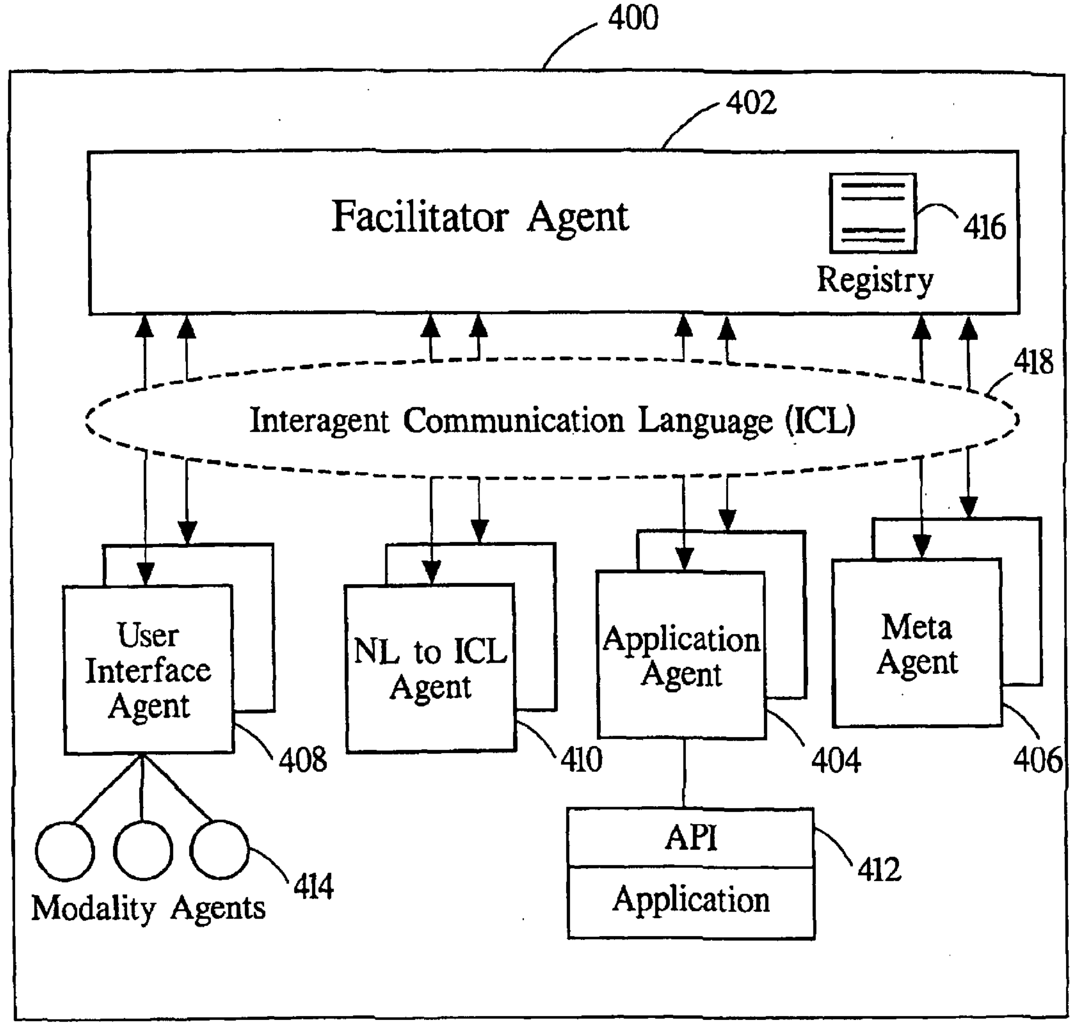

400
402
Facilitator Agent
416
Registry
418
Interagent Communication Language (ICL)
User Interface Agent
408
NL to ICL Agent
410
Application Agent
404
Meta Agent
406
API
Application
412
414
Modality Agents

Fig. 4

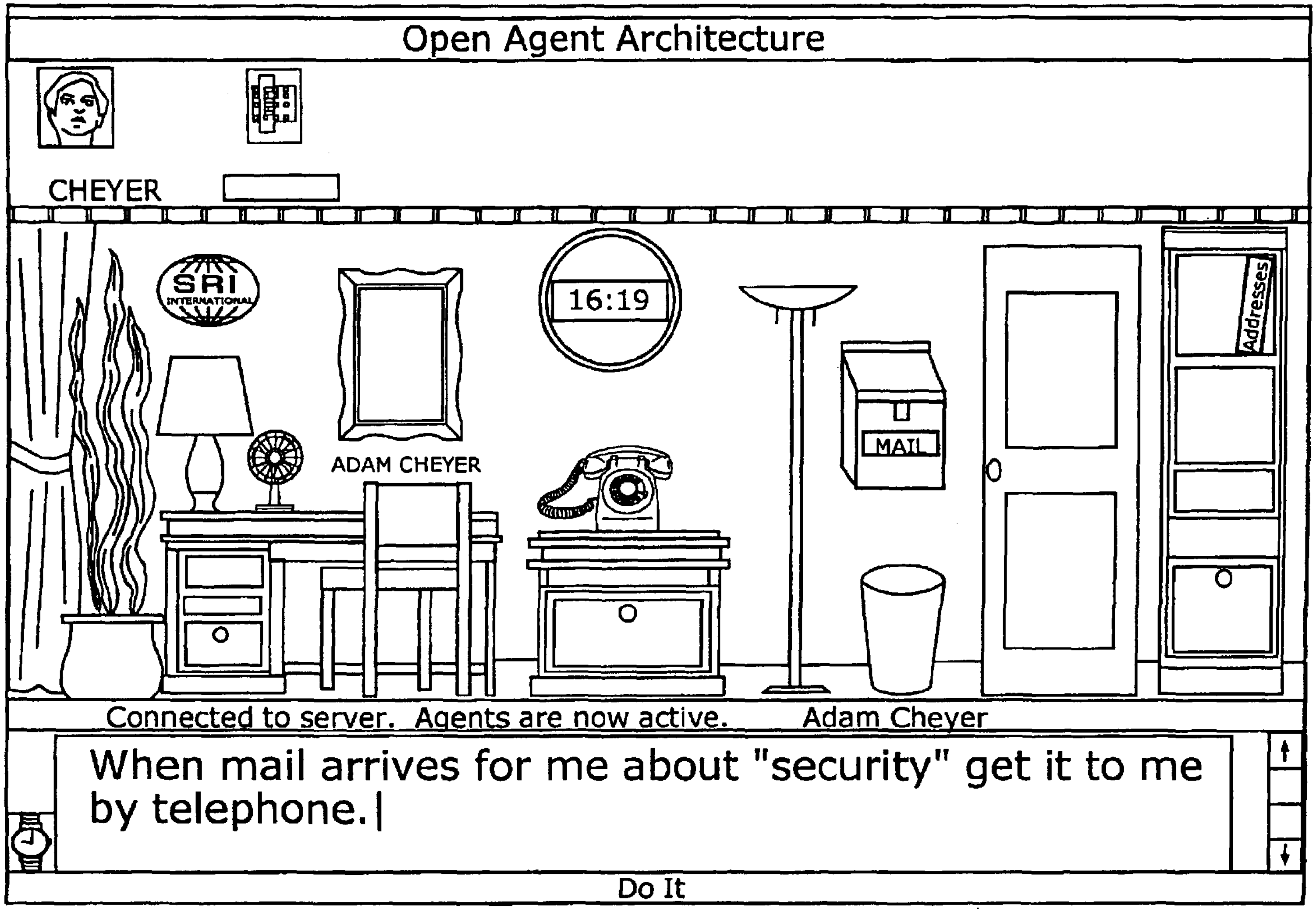

Open Agent Architecture
CHEYER
SRI INTERNATIONAL
16:19
ADAM CHEYER
MAIL
Addresses
Connected to server. Agents are now active.
Adam Cheyer
When mail arrives for me about "security" get it to me by telephone.
Do It

Fig. 5

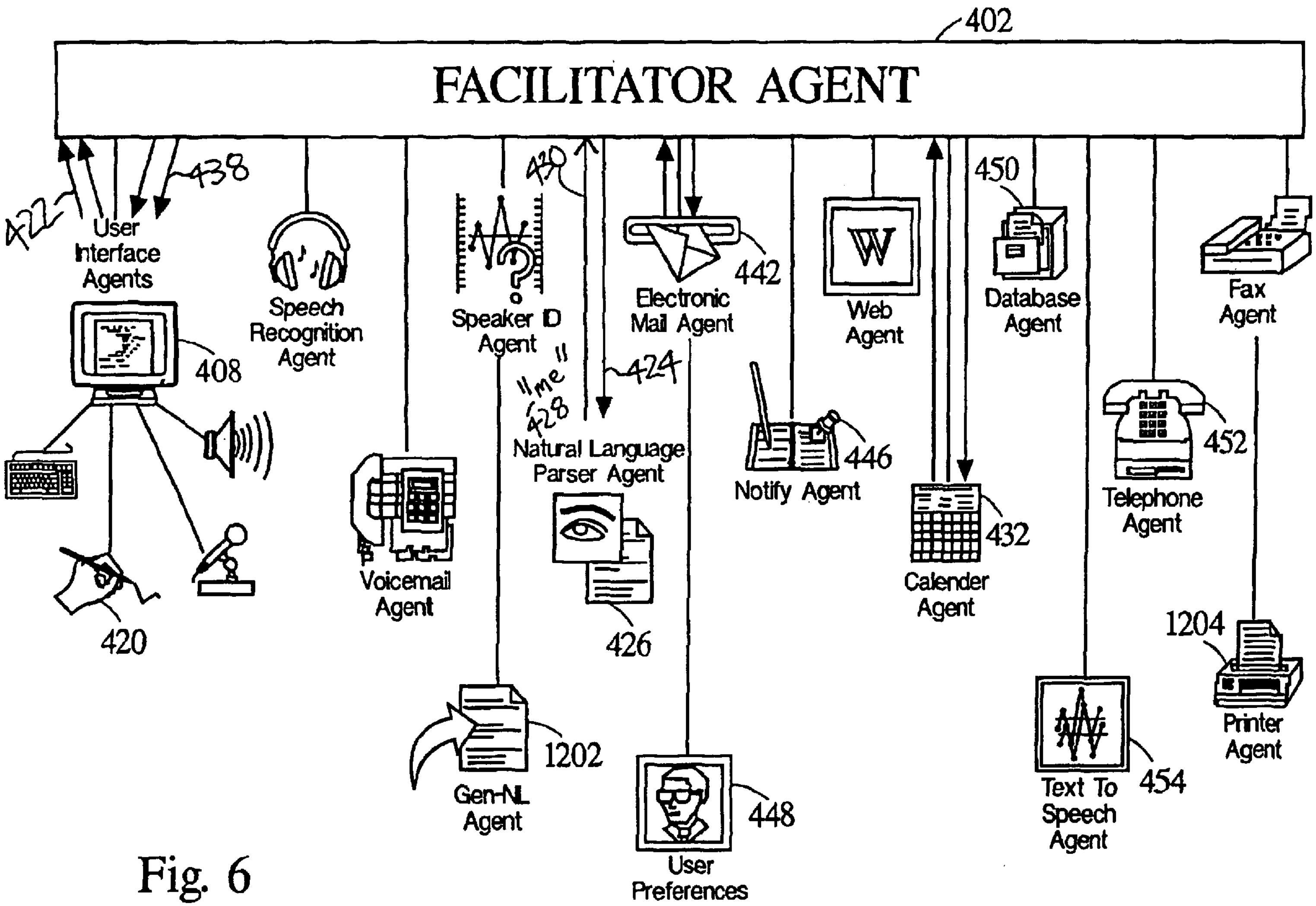

402
FACILITATOR AGENT
422
438
User Interface Agents
408
420
Speech Recognition Agent
Voicemail Agent
Speaker ID Agent
1202
Gen-NL Agent
430
"me"
428
424
Natural Language Parser Agent
426
Electronic Mail Agent
442
User Preferences
448
Web Agent
Notify Agent
446
450
Database Agent
432
Calender Agent
Text To Speech Agent
454
Telephone Agent
452
Fax Agent
1204
Printer Agent

Fig. 6

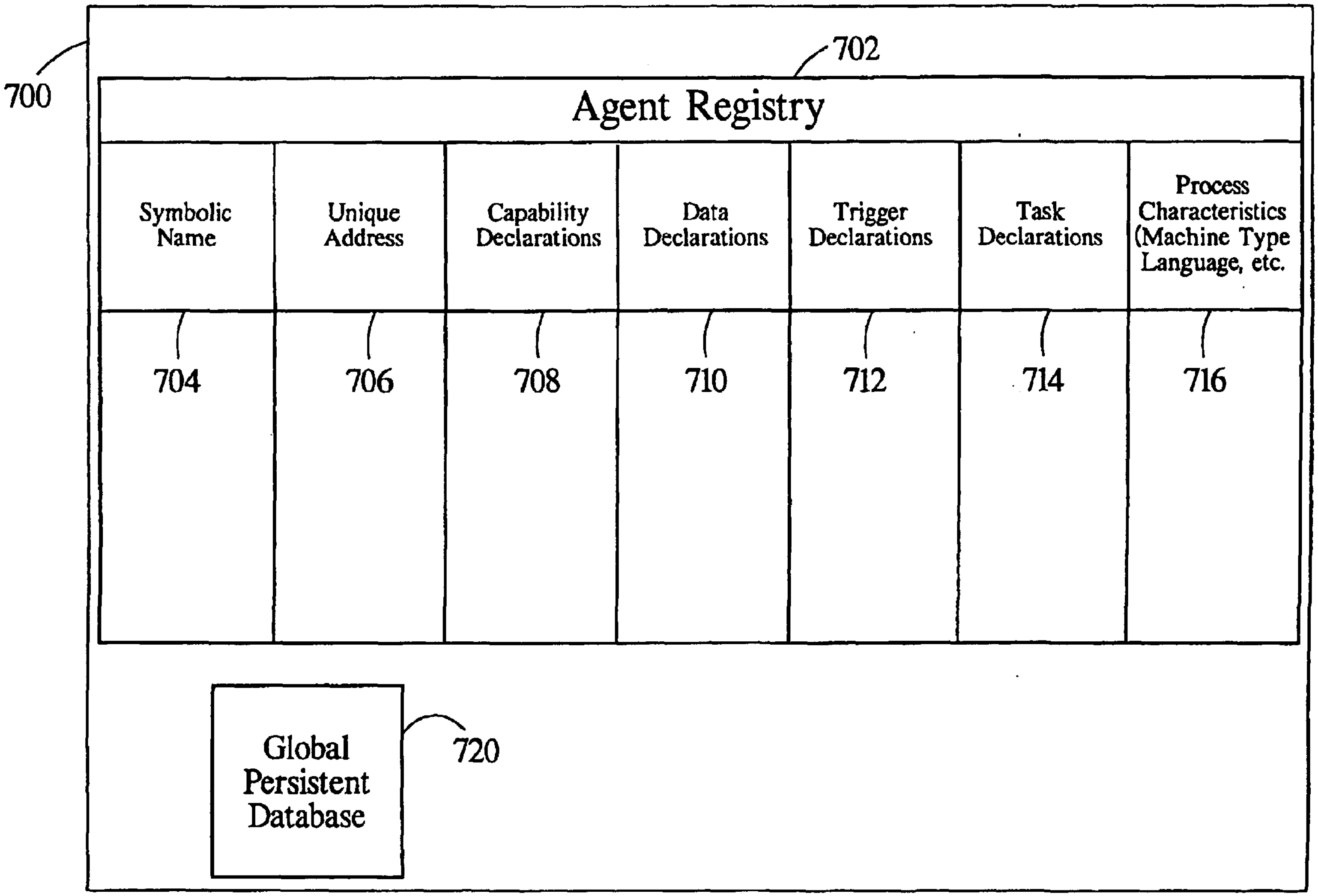
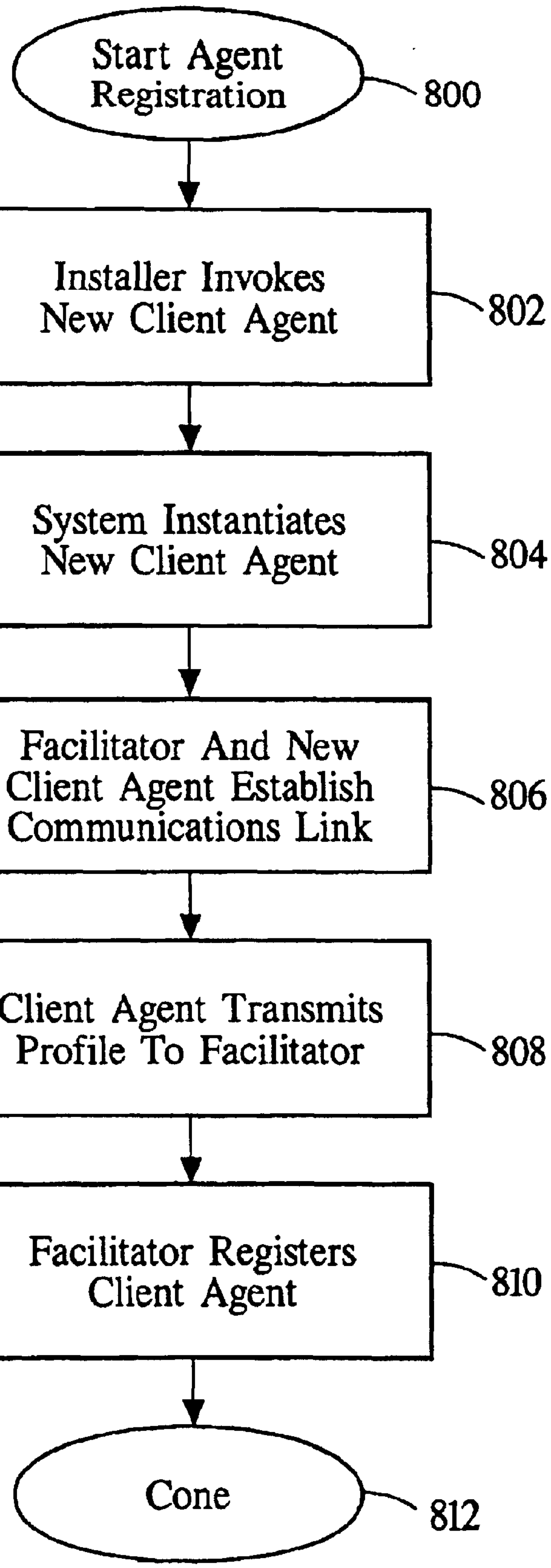

700
702
Agent Registry
Symbolic Name
Unique Address
Capability Declarations
Data Declarations
Trigger Declarations
Task Declarations
Process Characteristics (Machine Type Language, etc.
704
706
708
710
712
714
716
Global Persistent Database
720

Fig. 7



Start Agent Registration
800
Installer Invokes New Client Agent
802
System Instantiates New Client Agent
804
Facilitator And New Client Agent Establish Communications Link
806
Client Agent Transmits Profile To Facilitator
808
Facilitator Registers Client Agent
810
Cone
812

Fig. 8

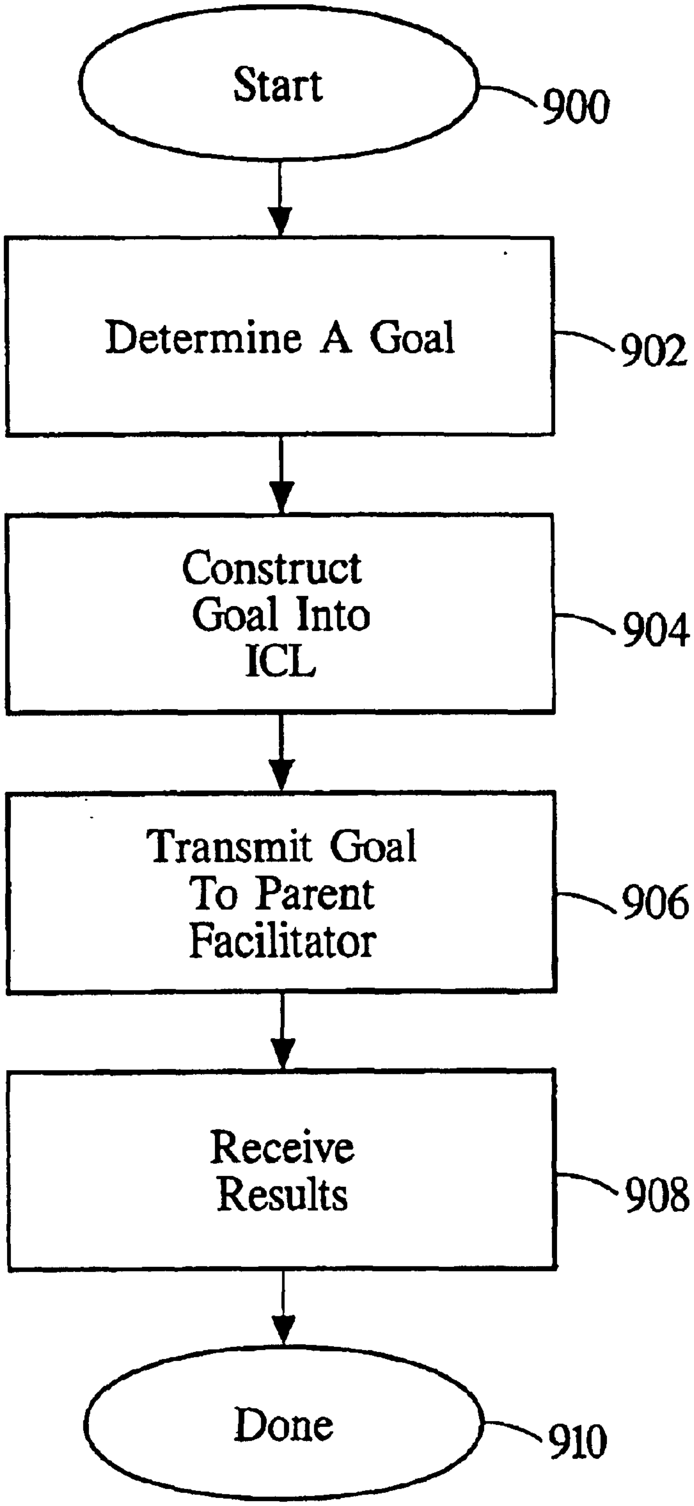

Start
900
Determine A Goal
902
Construct Goal Into ICL
904
Transmit Goal To Parent Facilitator
906
Receive Results
908
Done
910

Fig. 9

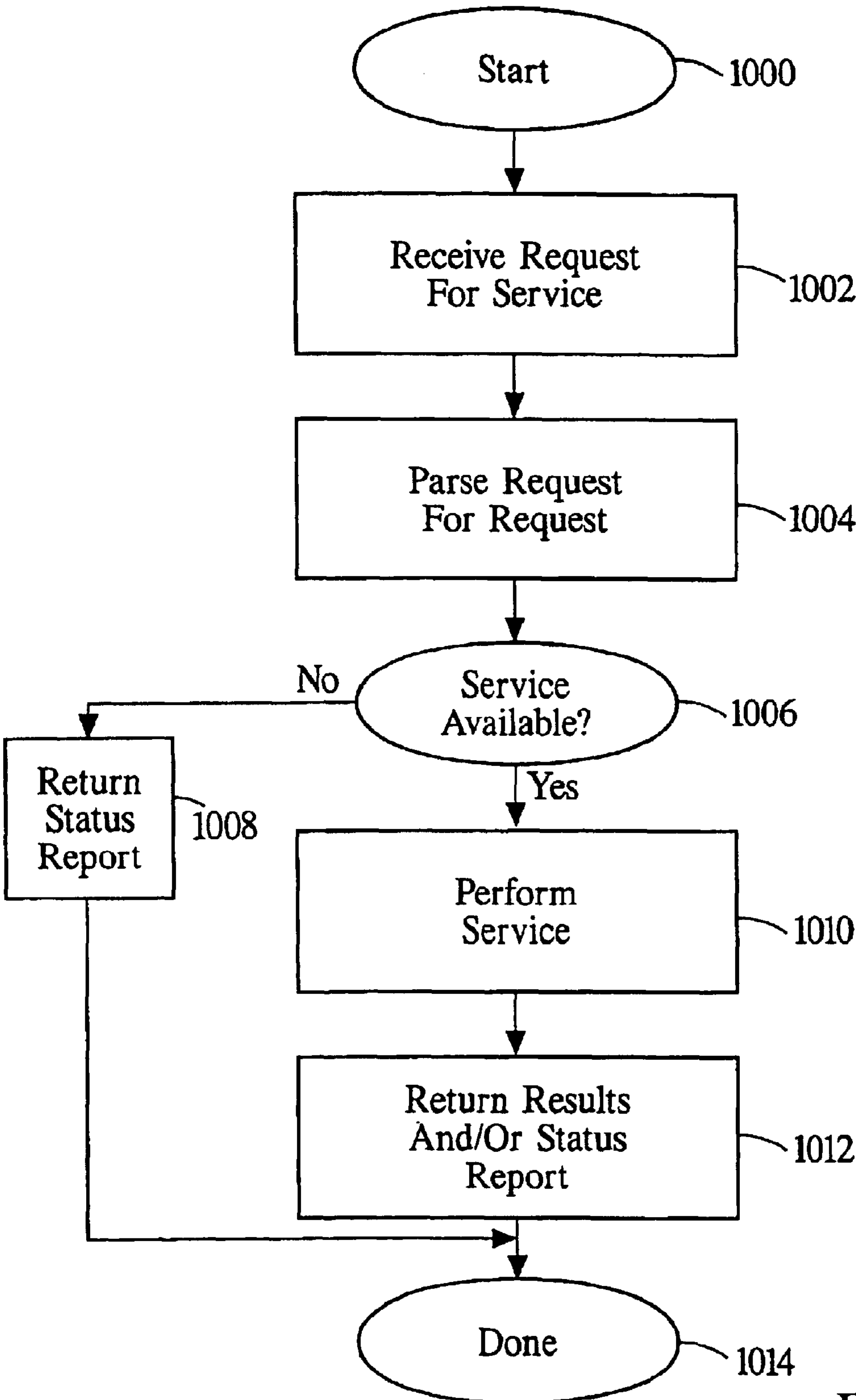
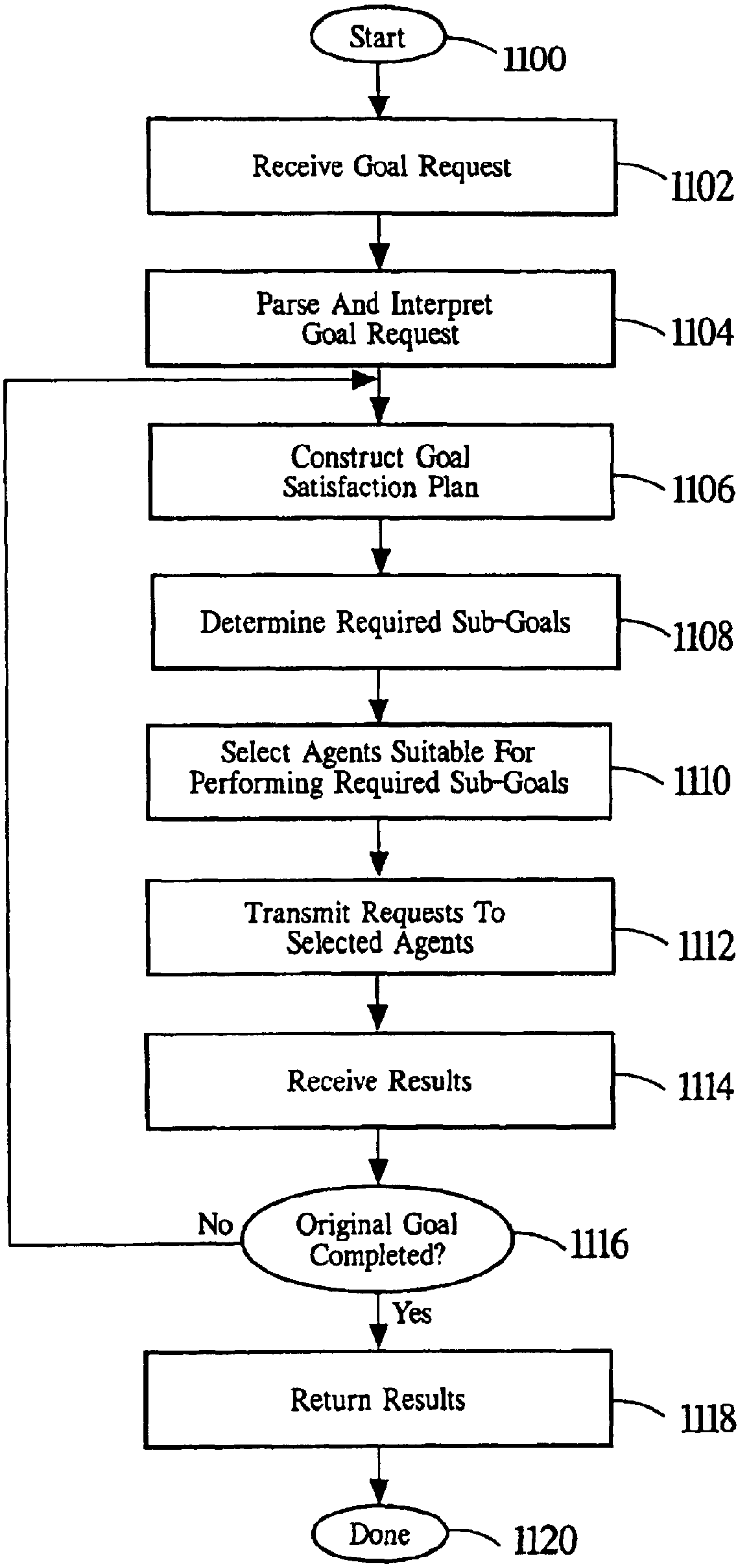

Start
1000
Receive Request For Service
1002
Parse Request For Request
1004
No
Service Available?
1006
Yes
Return Status Report
1008
Perform Service
1010
Return Results And/Or Status Report
1012
Done
1014

Fig. 10



Start
1100
Receive Goal Request
1102
Parse And Interpret Goal Request
1104
Construct Goal Satisfaction Plan
1106
Determine Required Sub-Goals
1108
Select Agents Suitable For Performing Required Sub-Goals
1110
Transmit Requests To Selected Agents
1112
Receive Results
1114
No
Original Goal Completed?
1116
Yes
Return Results
1118
Done
1120

Fig. 11

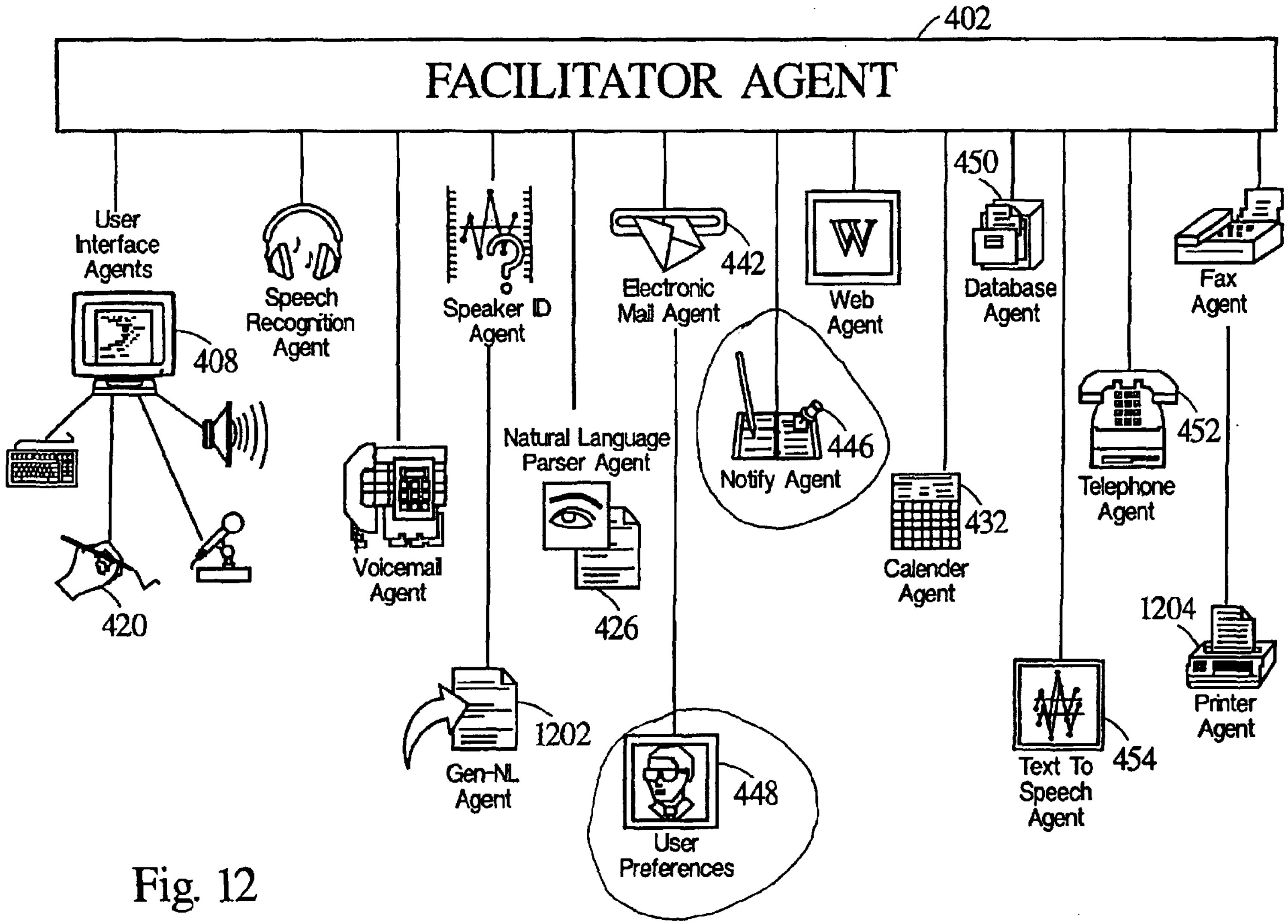

402
FACILITATOR AGENT
User Interface Agents
408
420
Speech Recognition Agent
Speaker ID Agent
Voicemail Agent
Natural Language Parser Agent
426
Gen-NL Agent
1202
Electronic Mail Agent
442
User Preferences
448
Notify Agent
446
Web Agent
450
Database Agent
Calender Agent
432
Text To Speech Agent
454
Telephone Agent
452
Fax Agent
1204
Printer Agent

Fig. 12

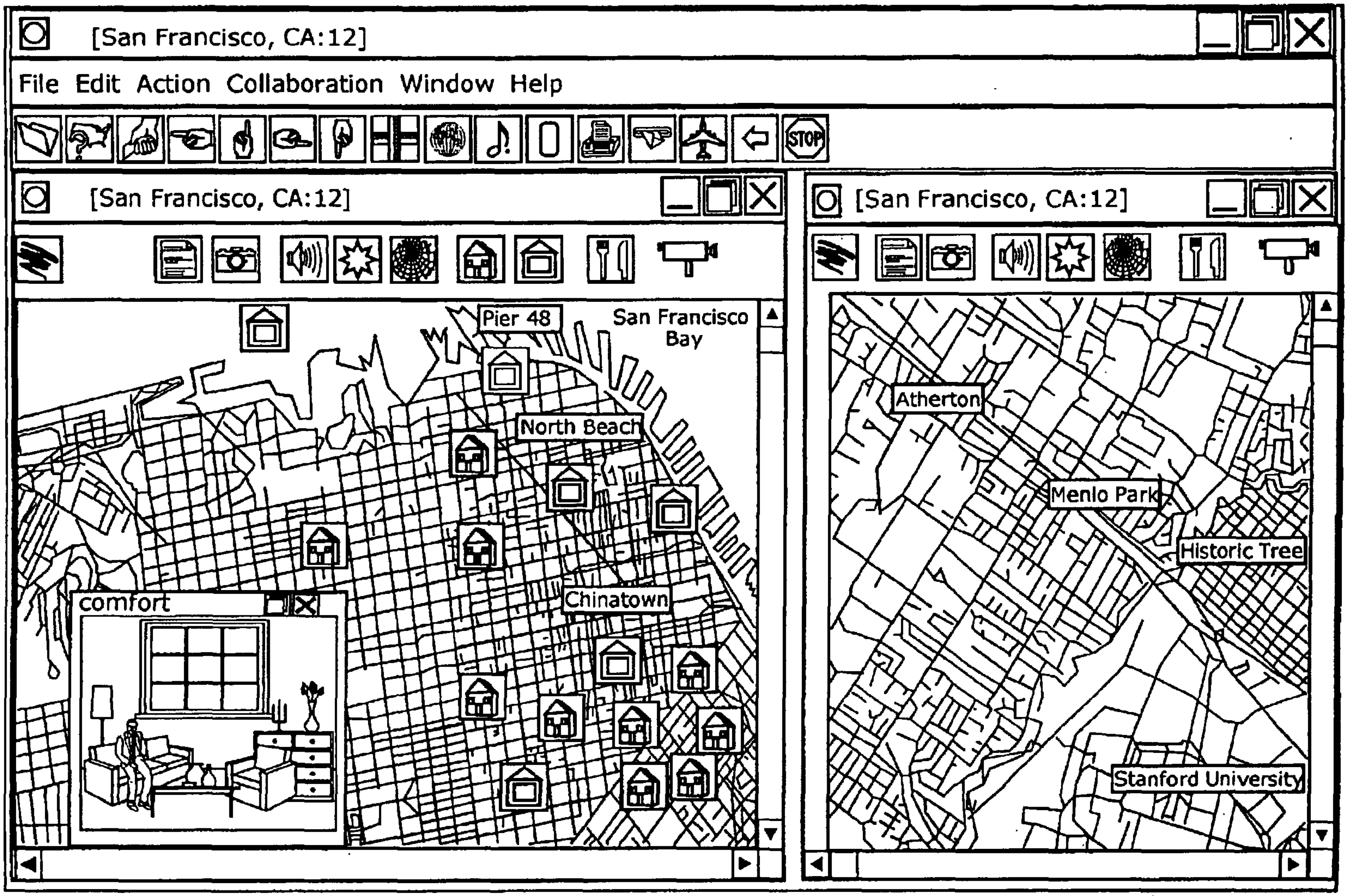

[San Francisco, CA:12]
File Edit Action Collaboration Window Help
STOP
[San Francisco, CA:12]
Pier 48
San Francisco Bay
North Beach
Chinatown
comfort
[San Francisco, CA:12]
Atherton
Menlo Park
Historic Tree
Stanford University

Fig. 13

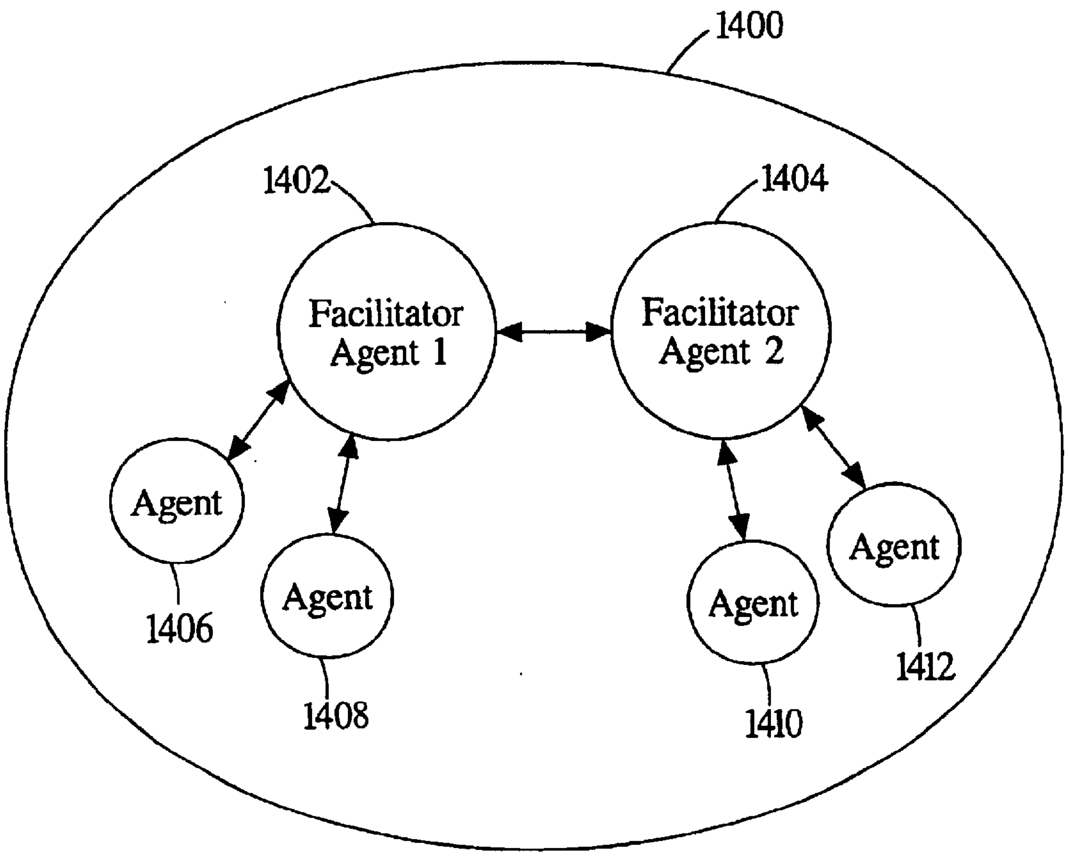

1400
1402
1404
Facilitator Agent 1
Facilitator Agent 2
Agent
Agent
Agent
Agent
1406
1408
1410
1412

Fig. 14

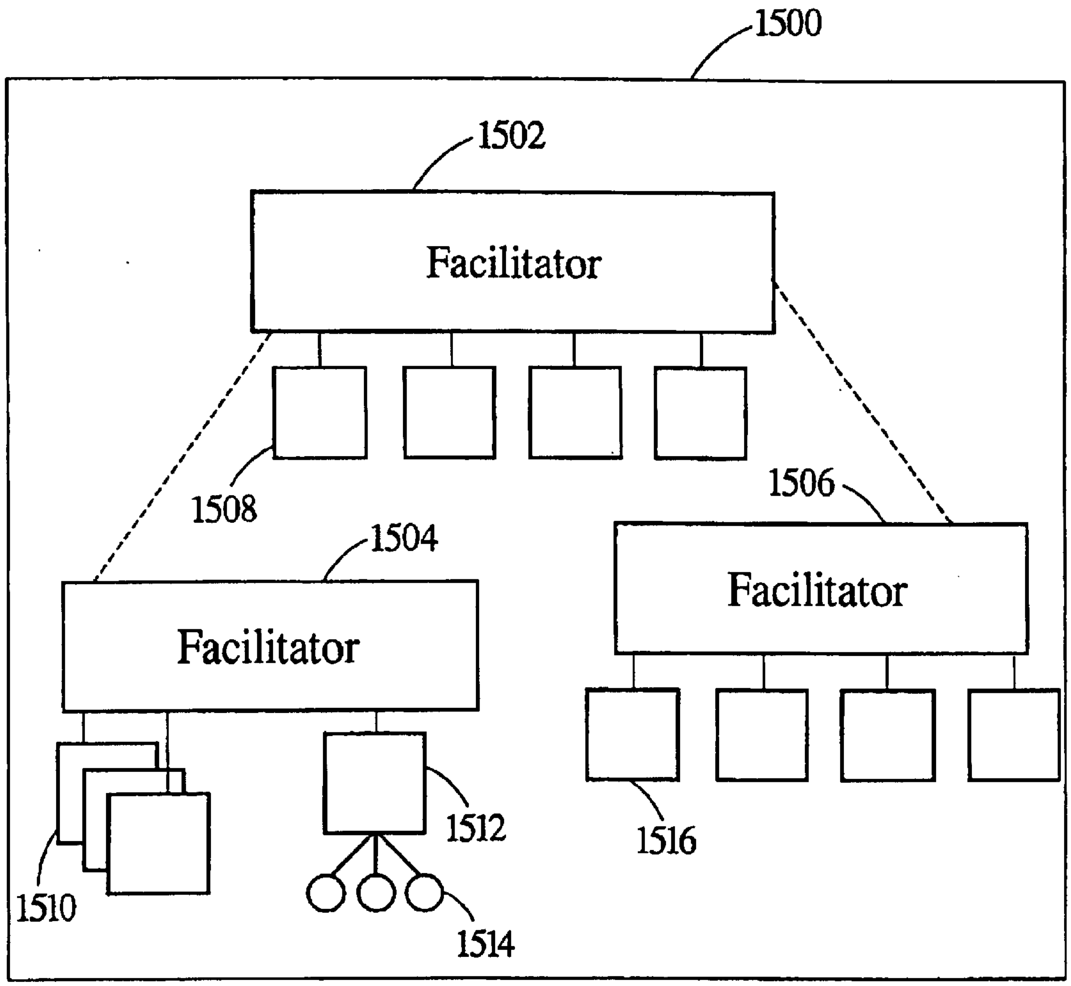

1500
1502
Facilitator
1508
1506
1504
Facilitator
Facilitator
1512
1516
1510
1514

Fig. 15

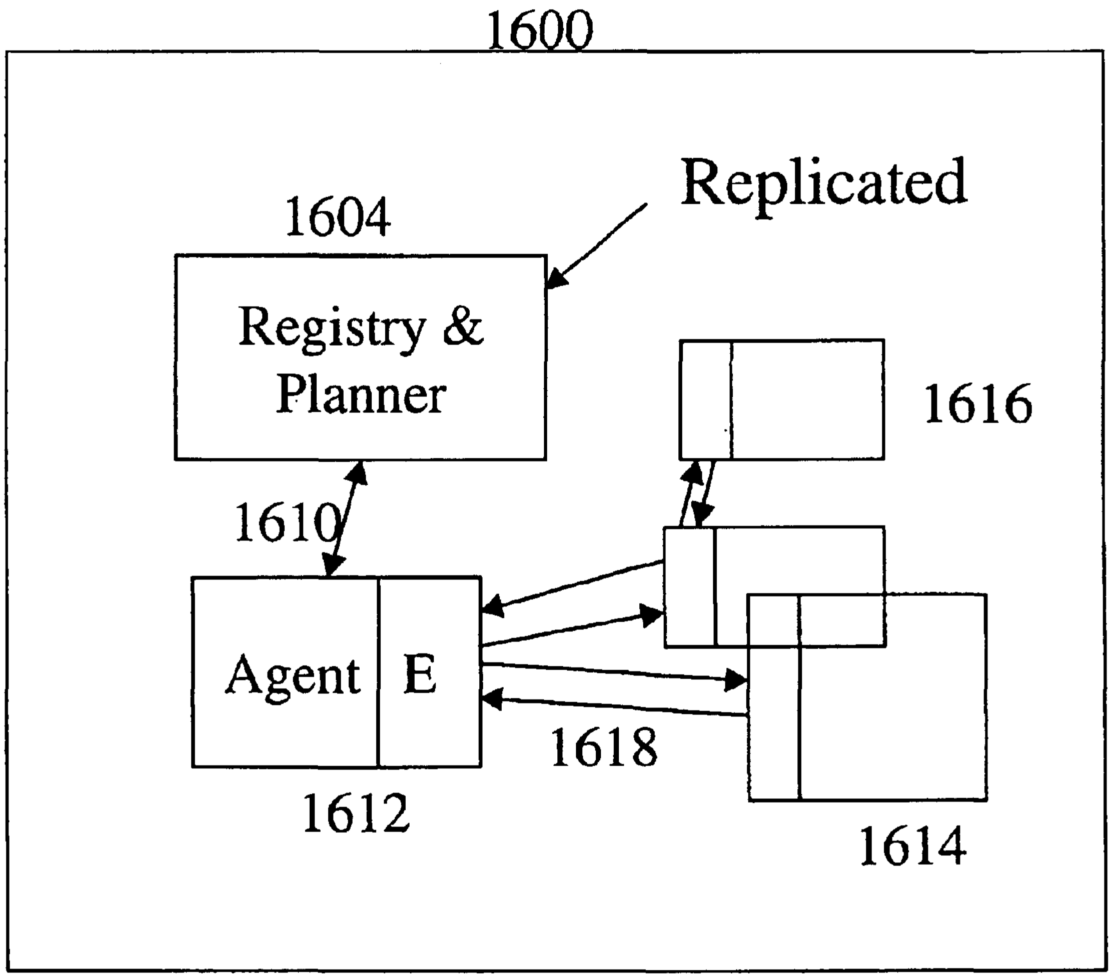

1600
Replicated
1604
Registry &
Planner
1616
1610
Agent
E
1618
1612
1614

Figure 16

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

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

1. 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 FIG. **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 computers (clients). Some of these computers provide resource to other computers and are known as server computers (servers). The servers **122** can vary greatly in the resources they possess, access they provide and services made available to other computers across a network. Servers may service other servers as well as clients.

The Internet is a computing system based upon this network computing model. The Internet is continually growing, stimulating a paradigm shift for computing away from requiring all relevant data and programs to reside on the user's desktop machine. The data now routinely accessed from computers spread around the world has become increasingly rich in format, comprising multimedia documents, and audio and video streams. With the popularization of programming languages such as JAVA, data transported between local and remote machines may also include programs that can be downloaded and executed on the local machine. There is an ever increasing reliance on networked computing, necessitating software design approaches that allow for flexible composition of distributed processing elements in a dynamically changing and relatively unstable environment.

In an increasing variety of domains, application designers and users are coming to expect the deployment of smarter, longer-lived, more autonomous, software applications. Push technology, persistent monitoring of information sources, and the maintenance of user models, allowing for personalized responses and sharing of preferences, are examples of the simplest manifestations of this trend. Commercial enterprises are introducing significantly more advanced approaches, in many cases employing recent research results from artificial intelligence, data mining, machine learning, and other fields.

More than ever before, the increasing complexity of systems, the development of new technologies, and the availability of multimedia material and environments are creating a demand for more accessible and intuitive user interfaces. Autonomous, distributed, multi-component systems providing sophisticated services will no longer lend themselves to the familiar "direct manipulation" model of interaction, in which an individual user masters a fixed selection of commands provided by a single application. Ubiquitous computing, in networked environments, has brought about a situation in which the typical user of many software services is likely to be a non-expert, who may access a given service infrequently or only a few times. Accommodating such usage patterns calls for new approaches, fortunately, input modalities now becoming widely available, such as speech recognition and pen-based handwriting/gesture recognition, and the ability to manage the presentation of systems' responses by using multiple media provide an opportunity to fashion a style of human-computer interaction that draws much more heavily on our experience with human-human interactions.

2. Prior Related Art

Existing approaches and technologies for distributed computing include to distributed objects, mobile objects, blackboard-style architectures, and agent-based software engineering.

The Distributed Object Approach

Object-oriented languages, such as C++ or JAVA, provide significant advances over standard procedural languages with respect to the reusability and modularity of code: encapsulation, inheritance and polymorhpism. Encapsulation encourages the creation of library interfaces that minimize dependencies on underlying algorithms or data structures. Changes to programming internals can be made at a later date with requiring modifications to the code that uses the library. Inheritance permits the extension and modification of a library of routines and data without requiring source code to the original library. Polymorphism allows one body of code to work on an arbitrary number of data types. For the sake of simplicity traditional objects may be seen to contain both methods and data. Methods provide the mechanisms by which the internal state of an object may be modified or by which communication may occur with another object or by which the instantiation or removal of objects may be directed.

With reference to FIG. **2**, a distributed object technology based around an Object Request Broker will now be described. Whereas "standard" object-oriented programming (OOP) languages can be used to build monolithic programs out of many object building blocks, distributed object technologies (DOOP) allow the creation of programs whose components may be spread across multiple machines. As shown in FIG. **2**, an object system **200** includes client objects **210** and server objects **220**. To implement a client-server relationship between objects, the distributed object system **200** uses a registry mechanism (CORBA's registry is called an object Request Broker, or ORB) **230** to store the interface descriptions of available objects. Through the services of the ORB **230**, a client can transparently invoke a method on a remote server object. The ORB **230** is then responsible for finding the object **220** that can implement the request, passing it the parameters, invoking its method, and returning the results. In the most sophisticated systems, the client **210** does not have to be aware of where the object is located, its programming language, its operating system, or any other system aspects that are not part of the server object's interface.

Although distributed objects offer a powerful paradigm for creating networked applications, certain aspects of the approach are not perfectly tailored to the constantly changing environment of the Internet. A major restriction of the DOOP approach is that the interactions among objects are fixed through explicitly coded instructions by the application developer. It is often difficult to reuse an object in a new application without bringing along all its inherent dependencies on other objects (embedded interface definitions and explicit method calls). Another restriction of the DOOP approach is the result of its reliance on a remote procedure call (RPC) style of communication. Although easy to debug, this single thread of execution model does not facilitate programming to exploit the potential for parallel computation that one would expect in a distributed environment. In addition, RPC uses a blocking (synchronous) scheme that does not scale well for high-volume transactions.

Mobile Objects

Mobile objects, sometimes called mobile agents, are bits of code that can move to another execution site (presumably on a different machine) under their own programmatic control, where they can then interact with the local environment. For certain types of problems, the mobile object paradigm offers advantages over more traditional distributed object approaches. These advantages include network bandwidth and parallelism. Network bandwidth advantages exist for some database queries or electronic commerce applications, where it is more efficient to perform tests on data by bringing the tests to the data than by bringing large amounts of data to the testing program. Parallelism advantages include situations in which mobile agents can be spawned in parallel to accomplish many tasks at once.

Some of the disadvantages and inconveniences of the mobile agent approach include the programmatic specificity of the agent interactions, lack of coordination support between participant agents and execution environment irregularities regarding specific programming languages supported by host processors upon which agents reside. In a fashion similar to that of DOOP programming, an agent developer must programmatically specify where to go and how to interact with the target environment. There is generally little coordination support to encourage interactions among multiple (mobile) participants. Agents must be written in the programming language supported by the execution environment, whereas many other distributed technologies support heterogeneous communities of components, written in diverse programming languages.

Blackboard Architectures

Blackboard architectures typically allow multiple processes to communicate by reading and writing tuples from a global data store. Each process can watch for items of interest, perform computations based on the state of the blackboard, and then add partial results or queries that other processes can consider. Blackboard architectures provide a flexible framework for problem solving by a dynamic community of distributed processes. A blackboard architecture provides one solution to eliminating the tightly bound interaction links that some of the other distributed technologies require during interprocess communication. This advantage can also be a disadvantage: although a programmer does not need to refer to a specific process during computation, the framework does not provide programmatic control for doing so in cases where this would be practical.

Agent-based Software Engineering

Several research communities have approached distributed computing by casting it as a problem of modeling communication and cooperation among autonomous entities, or agents. Effective communication among independent agents requires four components: (1) a transport mechanism carrying messages in an asynchronous fashion, (2) an interaction protocol defining various types of communication interchange and their social implications (for instance, a response is expected of a question), (3) a content language permitting the expression and interpretation of utterances, and (4) an agreed-upon set of shared vocabulary and meaning for concepts often called an ontology). Such mechanisms permit a much richer style of interaction among participants than can be expressed using a distributed object's RPC model or a blackboard architecture's centralized exchange approach.

Agent-based systems have shown much promise for flexible, fault-tolerant, distributed problem solving. Several agent-based projects have helped to evolve the notion of facilitation. However, existing agent-based technologies and architectures are typically very limited in the extent to which agents can specify complex goals or influence the strategies used by the facilitator. Further, such prior systems are not sufficiently attuned to the importance of integrating human agents (i.e., users) through natural language and other human-oriented user interface technologies.

The initial version of SRI International's Open Agent Architecture™ ("OAA®") technology provided only a very limited mechanism for dealing with compound goals. Fixed formats were available for specifying a flat list of either conjoined (AND) sub-goals or disjoined (OR) sub-goals; in both cases, parallel goal solving was hard-wired in, and only a single set of parameters for the entire list could be specified. More complex goal expressions involving (for example) combinations of different boolean connectors, nested expressions, or conditionally interdependent ("IF . . . THEN") goals were not supported. Further, system scalability was not adequately addressed in this prior work.

SUMMARY OF INVENTION

A first embodiment of the present invention discloses a highly flexible, software-based architecture for constructing distributed systems. The architecture supports cooperative task completion by flexible, dynamic configurations of autonomous electronic agents. Communication and cooperation between agents are brokered by one or more facilitators, which are responsible for matching requests, from users and agents, with descriptions of the capabilities of other agents. It is not generally required that a user or agent know the identities, locations, or number of other agents involved in satisfying a request, and relatively minimal effort is involved in incorporating new agents and "wrapping" legacy applications. Extreme flexibility is achieved through an architecture organized around the declaration of capabilities by service-providing agents, the construction of arbitrarily complex goals by users and service-requesting agents, and the role of facilitators in delegating and coordinating the satisfaction of these goals, subject to advice and constraints that may accompany them. Additional mechanisms and features include facilities for creating and maintaining shared repositories of data; the use of triggers to instantiate commitments within and between agents; agent-based provision of multi-modal user interfaces, including natural language; and built-in support for including the user as a privileged member of the agent community. Specific embodiments providing enhanced scalability are also described.

BRIEF DESCRIPTION OF THE DRAWINGS

Prior Art

Prior Art FIG. **1** depicts a networked computing model;

Prior Art FIG. **2** depicts a distributed object technology based around an Object Resource Broker;

Examples of the Invention

FIG. **3** depicts a distributed agent system based around a facilitator agent;

FIG. **4** presents a structure typical of one small system of the present invention;

FIG. **5** depicts an Automated Office system implemented in accordance with an example embodiment of the present invention supporting a mobile user with a laptop computer and a telephone;

FIG. **6** schematically depicts an Automated Office system implemented as a network of agents in accordance with a preferred embodiment of the present invention;

FIG. **7** schematically shows data structures internal to a facilitator in accordance with a preferred embodiment of the present invention;

FIG. **8** depicts operations involved in instantiating a client agent with its parent facilitator in accordance with a preferred embodiment of the present invention;

FIG. **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;

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

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

FIG. **12** depicts an Open Agent Architecture™ based system of agents implementing a unified messaging application in accordance with a preferred embodiment of the present invention;

FIG. **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;

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

FIG. **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

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

DETAILED DESCRIPTION OF THE INVENTION

FIG. **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 FIG. **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 FIG. **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 FIG. **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.

FIG. **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, FIG. **4** places more structure upon the system **400** than shown in FIG. **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 shown in FIG. **4**. All agents that are not facilitators are referred to herein generically as client agents—so called because each acts (in some respects) as a client of some facilitator, which provides communication and other essential services for the client.

The variety of possible client agents is essentially unlimited. Some typical categories of client agents would include application agents **404**, meta-agents **406**, and user interface agents **408**, as depicted in FIG. **4**. Application agents **404** denote specialists that provide a collection of services of a particular sort. These services could be domain-independent technologies (such as speech recognition, natural language processing **410**, email, and some forms of data retrieval and data mining) or user-specific or domain-specific (such as a travel planning and reservations agent). Application agents may be based on legacy applications or libraries, in which case the agent may be little more than a wrapper that calls a pre-existing API **412**, for example. Meta-agents **406** are agents whose role is to assist the facilitator agent **402** in coordinating the activities of other agents. While the facilitator **402** possesses domain-independent coordination strategies, meta-agents **406** can augment these by using domain- and application-specific knowledge or reasoning (including but not limited to rules, learning algorithms and planning).

With further reference to FIG. **4**, user interface agents **408** can play an extremely important and interesting role in certain embodiments of the present invention. By way of explanation, in some systems, a user interface agent can be implemented as a collection of "micro-agents", each monitoring a different input modality (point-and-click, handwriting, pen gestures, speech), and collaborating to produce the best interpretation of the current inputs. These micro-agents are depicted in FIG. **4**, for example, as Modality Agents **414**. While describing such subcategories of client agents is useful for purposes of illustration and understanding, they need not be formally distinguished within the system in preferred implementations of the present invention.

The operation of one preferred embodiment of the present invention will be discussed in greater detail below, but may be briefly outlined as follows. When invoked, a client agent makes a connection to a facilitator, which is known as its parent facilitator. These connections are depicted as a double headed arrow between the client agent and the facilitator agent in FIGS. **3** and **4**, for example. Upon connection, an agent registers with its parent facilitator a specification of the capabilities and services it can provide. For example, a natural language agent may register the characteristics of its available natural language vocabulary. (For more details regarding client agent connections, see the discussion of FIG. **8** below.) Later during task completion, when a facilitator determines that the registered services **416** of one of its client agents will help satisfy a goal, the facilitator sends that client a request expressed in the Interagent Communication Language (ICL) **418**. (See FIG. **11** below for a more detailed discussion of the facilitator operations involved.) The agent parses this request, processes it, and returns answers or status reports to the facilitator. In processing a request, the client agent can make use of a variety of infrastructure capabilities provided in the preferred embodiment. For example, the client agent can use ICL **418** to request services of other agents, set triggers, and read or write shared data on the facilitator or other client agents that maintain shared data. (See the discussion of FIGS. **9**–**11** below for a more detailed discussion of request processing.)

The functionality of each client agent are made available to the agent community through registration of the client agent's capabilities with a facilitator **402**. A software "wrapper" essentially surrounds the underlying application program performing the services offered by each client. The common infrastructure for constructing agents is preferably supplied by an agent library. The agent library is preferably accessible in the runtime environment of several different programming languages. The agent library preferably minimizes the effort required to construct a new system and maximizes the ease with which legacy systems can be "wrapped" and made compatible with the agent-based architecture of the present invention.

By way of further illustration, a representative application is now briefly presented with reference to FIGS. **5** and **6**. In the Automated Office system depicted in FIG. **5**, a mobile user with a telephone and a laptop computer can access and task commercial applications such as calendars, databases, and email systems running back at the office. A user interface (UI) agent **408**, shown in FIG. **6**, runs on the user's local laptop and is responsible for accepting user input, sending requests to the facilitator **402** for delegation to appropriate agents, and displaying the results of the distributed computation. The user may interact directly with a specific remote application by clicking on active areas in the interface, calling up a form or window for that application, and making queries with standard interface dialog mechanisms. Conversely, a user may express a task to be executed by using typed, handwritten, or spoken (over the telephone) English sentences, without explicitly specifying which agent or agents should perform the task.

For instance, if the question "What is my schedule?" is written **420** in the user interface **408**, this request will be sent **422** by the UI **408** to the facilitator **402**, which in turn will ask **424** a natural language (NL) agent **426** to translate the query into JCL **18**. To accomplish this task, the NL agent **426** may itself need to make requests of the agent community to resolve unknown words such as "me" **428** (the UI agent **408** can respond **430** with the name of the current user) or "schedule" **432** (the calendar agent **434** defines this word **436**). The resulting ICL expression is then routed by the facilitator **402** to appropriate agents (in this case, the calendar agent **434**) to execute the request. Results are sent back **438** to the UI agent **408** for display.

The spoken request "When mail arrives for me about security, notify me immediately." produces a slightly more complex example involving communication among all agents in the system. After translation into ICL as described above, the facilitator installs a trigger **440** on the mail agent **442** to look for new messages about security. When one such message does arrive in its mail spool, the trigger fires, and the facilitator matches the action part of the trigger to capabilities published by the notification agent **446**. The notification agent **446** is a meta-agent, as it makes use of rules concerning the optimal use of different output modalities (email, fax, speech generation over the telephone) plus information about an individual user's preferences **448** to determine the best way of relaying a message through available media transfer application agents. After some competitive parallelism to locate the user (the calendar agent **434** and database agent **450** may have different guesses as to where to find the user) and some cooperative parallelism to produce required information (telephone number of location, user password, and an audio file containing a text-to-speech representation of the email message), a telephone agent **452** calls the user, verifying its identity through touchtones, and then play the message.

The above example illustrates a number of inventive features. As new agents connect to the facilitator, registering capability specifications and natural language vocabulary, what the user can say and do dynamically changes; in other words, the ICL is dynamically expandable. For example, adding a calendar agent to the system in the previous example and registering its capabilities enables users to ask natural language questions about their "schedule" without any need to revise code for the facilitator, the natural language agents, or any other client agents. In addition, the interpretation and execution of a task is a distributed process, with no single agent defining the set of possible inputs to the system. Further, a single request can produce cooperation and flexible communication among many agents, written in different programming languages and spread across multiple machines.

Design Philosophy and Considerations

One preferred embodiment provides an integration mechanism for heterogeneous applications in a distributed infrastructure, incorporating some of the dynamism and extensibility of blackboard approaches, the efficiency associated with mobile objects, plus the rich and complex interactions of communicating agents. Design goals for preferred embodiments of the present invention may be categorized under the general headings of interoperation and cooperation, user interfaces, and software engineering. These design goals are not absolute requirements, nor will they necessarily be satisfied by all embodiments of the present invention, but rather simply reflect the inventor's currently preferred design philosophy.

Versatile Mechanisms of Interoperation and Cooperation

Interoperation refers to the ability of distributed software components—agents—to communicate meaningfully. While every system-building framework must provide mechanisms of interoperation at some level of granularity, agent-based frameworks face important new challenges in this area. This is true primarily because autonomy, the hallmark of individual agents, necessitates greater flexibility in interactions within communities of agents. Coordination refers to the mechanisms by which a community of agents is able to work together productively on some task. In these areas, the goals for our framework are to provide flexibility in assembling communities of autonomous service providers, provide flexibility in structuring cooperative interactions, impose the right amount of structure, as well as include legacy and "owned-elsewhere" applications.

Provide flexibility in assembling communities of autonomous service providers—both at development time and at runtime. Agents that conform to the linguistic and ontological requirements for effective communication should be able to participate in an agent community, in various combinations, with minimal or near minimal prerequisite knowledge of the characteristics of the other players. Agents with duplicate and overlapping capabilities should be able to coexist within the same community, with the system making optimal or near optimal use of the redundancy.

Provide flexibility in structuring cooperative interactions among the members of a community of agents. A framework preferably provides an economical mechanism for setting up a variety of interaction patterns among agents, without requiring an inordinate amount of complexity or infrastructure within the individual agents. The provision of a service should be independent or minimally dependent upon a particular configuration of agents.

Impose the right amount of structure on individual agents. Different approaches to the construction of multi-agent systems impose different requirements on the individual agents. For example, because KQML is neutral as to the content of messages, it imposes minimal structural requirements on individual agents. On the other hand, the BDI paradigm tends to impose much more demanding requirements, by making assumptions about the nature of the programming elements that are meaningful to individual agents. Preferred embodiments of the present invention should fall somewhere between the two, providing a rich set of interoperation and coordination capabilities, without precluding any of the software engineering goals defined below.

Include legacy and "owned-elsewhere" applications. Whereas legacy usually implies reuse of an established system fully controlled by the agent-based system developer, owned-elsewhere refers to applications to which the developer has partial access, but no control. Examples of owned-elsewhere applications include data sources and services available on the World Wide Web, via simple form-based interfaces, and applications used cooperatively within a virtual enterprise, which remain the properties of separate corporate entities. Both classes of application must preferably be able to interoperate, more or less as full-fledged members of the agent community, without requiring an overwhelming integration effort.

Human-Oriented User Interfaces

Systems composed of multiple distributed components, and possibly dynamic configurations of components, require the crafting of intuitive user interfaces to provide conceptually natural interaction mechanisms, treat users as privileged members of the agent community and support collaboration.

Provide conceptually natural interaction mechanisms with multiple distributed components. When there are numerous disparate agents, and/or complex tasks implemented by the system, the user should be able to express requests without having detailed knowledge of the individual agents. With speech recognition, handwriting recognition, and natural language technologies becoming more mature, agent architectures should preferably support these forms of input playing increased roles in the tasking of agent communities.

Preferably treat users as privileged members of the agent community by providing an appropriate level of task specification within software agents, and reusable translation mechanisms between this level and the level of human requests, supporting constructs that seamlessly incorporate interactions between both human-interface and software types of agents.

Preferably support collaboration (simultaneous work over shared data and processing resources) between users and agents.

Realistic Software Engineering Requirements

System-building frameworks should preferably address the practical concerns of real-world applications by the specification of requirements which preferably include: Minimize the effort required to create new agents, and to wrap existing applications. Encourage reuse, both of domain-independent and domain-specific components. The concept of agent orientation, like that of object orientation, provides a natural conceptual framework for reuse, so long as mechanisms for encapsulation and interaction are structured appropriately. Support lightweight mobile platforms. Such platforms should be able to serve as hosts for agents, without requiring the installation of a massive environment. It should also be possible to construct individual agents that are relatively small and modest in their processing requirements. Minimize platform and language barriers. Creation of new agents, as well as wrapping of existing applications, should not require the adoption of a new language or environment.

Mechanisms of Cooperation

Cooperation among agents in accordance with the present invention is preferably achieved via messages expressed in a common language, ICL. Cooperation among agent is further preferably structured around a three-part approach: providers of services register capabilities specifications with a facilitator, requesters of services construct goals and relay them to a facilitator, and facilitators coordinate the efforts of the appropriate service providers in satisfying these goals.

The Interagent Communication Language (ICL)

Interagent Communication Language ("ICL") **418** refers to an interface, communication, and task coordination language preferably shared by all agents, regardless of what platform they run on or what computer language they are programmed in. ICL may be used by an agent to task itself or some subset of the agent community. Preferably, ICL allows agents to specify explicit control parameters while simultaneously supporting expression of goals in an underspecified, loosely constrained manner. In a further preferred embodiment, agents employ ICL to perform queries, execute actions, exchange information, set triggers, and manipulate data in the agent community.

In a further preferred embodiment, a program element expressed in ICL is the event. The activities of every agent, as well as communications between agents, are preferably structured around the transmission and handling of events. In communications, events preferably serve as messages between agents; in regulating the activities of individual agents, they may preferably be thought of as goals to be satisfied. Each event preferably has a type, a set of parameters, and content. For example, the agent library procedure oaa_Solve can be used by an agent to request services of other agents. A call to oaa_Solve, within the code of agent A, results in an event having the form

ev_post_solve(Goal, Params)

going from A to the facilitator, where ev_post_solve is the type, Goal is the content, and Params is a list of parameters. The allowable content and parameters preferably vary according to the type of the event.

The ICL preferably includes a layer of conversational protocol and a content layer. The conversational layer of ICL is defined by the event types, together with the parameter lists associated with certain of these event types. The content layer consists of the specific goals, triggers, and data elements that may be embedded within various events.

The ICL conversational protocol is preferably specified using an orthogonal, parameterized approach, where the conversational aspects of each element of an interagent conversation are represented by a selection of an event type and a selection of values from at least one orthogonal set of parameters. This approach offers greater expressiveness than an approach based solely on a fixed selection of speech acts, such as embodied in KQML. For example, in KQML, a request to satisfy a query can employ either of the performatives ask_all or ask_one. In ICL, on the other hand, this type of request preferably is expressed by the event type evost solve, together with the solution_limit(N) parameter—where N can be any positive integer. (A request for all solutions is indicated by the omission of the solution_limit parameter.) The request can also be accompanied by other parameters, which combine to further refine its semantics. In KQML, then, this example forces one to choose between two possible conversational options, neither of which may be precisely what is desired. In either case, the performative chosen is a single value that must capture the entire conversational characterization of the communication. This requirement raises a difficult challenge for the language designer, to select a set of performatives that provides the desired functionality without becoming unmanageably large. Consequently, the debate over the right set of performatives has consumed much discussion within the KQML community.

The content layer of the ICL preferably supports unification and other features found in logic programming language environments such as PROLOG. In some embodiments, the content layer of the ICL is simply an extension of at least one programming language. For example, the Applicants have found that PROLOG is suitable for implementing and extending into the content layer of the ICL. The agent libraries preferably provide support for constructing, parsing, and manipulating ICL expressions. It is possible to embed content expressed in other languages within an ICL event. However, expressing content in ICL simplifies the facilitator's access to the content, as well as the conversational layer, in delegating requests. This gives the facilitator more information about the nature of a request and helps the facilitator decompose compound requests and delegate the sub-requests.

Further, ICL expressions preferably include, in addition to events, at least one of the following: capabilities declarations, requests for services, responses to requests, trigger specifications, and shared data elements. A further preferred embodiment of the present invention incorporates ICL expressions including at least all of the following: events, capabilities declarations, requests for services, responses to requests, trigger specifications, and shared data elements.

Providing Services: Specifying "Solvables"

In a preferred embodiment of the present invention, every participating agent defines and publishes a set of capability declarations, expressed in ICL, describing the services that it provides. These declarations establish a high-level interface to the agent. This interface is used by a facilitator in communicating with the agent, and, most important, in delegating service requests (or parts of requests) to the agent. Partly due to the use of PROLOG as a preferred basis for ICL, these capability declarations are referred as solvables. The agent library preferably provides a set of procedures allowing an agent to add, remove, and modify its solvables, which it may preferably do at any time after connecting to its facilitator.

There are preferably at least two major types of solvables: procedure solvables and data solvables. Intuitively, a procedure solvable performs a test or action, whereas a data solvable provides access to a collection of data. For example, in creating an agent for a mail system, procedure solvables might be defined for sending a message to a person, testing whether a message about a particular subject has arrived in the mail queue, or displaying a particular message onscreen. For a database wrapper agent, one might define a distinct data solvable corresponding to each of the relations present in the database. Often, a data solvable is used to provide a shared data store, which may be not only queried, but also updated, by various agents having the required permissions.

There are several primary technical differences between these two types of solvables. First, each procedure solvable must have a handler declared and defined for it, whereas this is preferably not necessary for a data solvable. The handling of requests for a data solvable is preferably provided transparently by the agent library. Second, data solvables are preferably associated with a dynamic collection of facts (or clauses), which may be further preferably modified at runtime, both by the agent providing the solvable, and by other agents (provided they have the required permissions). Third, special features, available for use with data solvables, preferably facilitate maintaining the associated facts. In spite of these differences, it should be noted that the mechanism of use by which an agent requests a service is the same for the two types of solvables.

In one embodiment, a request for one of an agent's services normally arrives in the form of an event from the agent's facilitator. The appropriate handler then deals with this event. The handler may be coded in whatever fashion is most appropriate, depending on the nature of the task, and the availability of task-specific libraries or legacy code, if any. The only hard requirement is that the handler return an appropriate response to the request, expressed in ICL. Depending on the nature of the request, this response could be an indication of success or failure, or a list of solutions (when the request is a data query).

A solvable preferably has three parts: a goal, a list of parameters, and a list of permissions, which are declared using the format:

solvable(Goal, Parameters, Permissions)

The goal of a solvable, which syntactically takes the preferable form of an ICL structure, is a logical representation of the service provided by the solvable. (An ICL structure consists of a functor with 0 or more arguments. For example, in the structure a(b,c), 'a' is the functor, and 'b' and 'c' the arguments.) As with a PROLOG structure, the goal's arguments themselves may preferably be structures.

Various options can be included in the parameter list, to refine the semantics associated with the solvable. The type parameter is preferably used to say whether the solvable is data or procedure. When the type is procedure, another parameter may be used to indicate the handler to be associated with the solvable. Some of the parameters appropriate for a data solvable are mentioned elsewhere in this application. In either case (procedure or data solvable), the private parameter may be preferably used to restrict the use of a solvable to the declaring agent when the agent intends the solvable to be solely for its internal use but wishes to take advantage of the mechanisms in accordance with the present invention to access it, or when the agent wants the solvable to be available to outside agents only at selected times. In support of the latter case, it is preferable for the agent to change the status of a solvable from private to non-private at any time.

The permissions of a solvable provide mechanisms by which an agent may preferably control access to its services allowing the agent to restrict calling and writing of a solvable to itself and/or other selected agents. (Calling means requesting the service encapsulated by a solvable, whereas Writing means modifying the collection of facts associated with a data solvable.) The default permission for every solvable in a further preferred embodiment of the present invention is to be callable by anyone, and for data solvables to be writable by anyone. A solvable's permissions can preferably be changed at any time, by the agent providing the solvable.

For example, the solvables of a simple email agent might include:

```
solvable(send_message(email, +ToPerson, +Params),
  [type(procedure), callback(send_mail)],
    [ ]
solvable(last_message(email, -MessageId),
  [type(data), single_value(true)],
  [write(true)]),
solvable (get_message (email, +MessageId, -Msg),
  [type(procedure), callback(get_mail)], [ ])
```

The symbols '+' and '−', indicating input and output arguments, are at present used only for purposes of documentation. Most parameters and permissions have default values, and specifications of default values may be omitted from the parameters and permissions lists.

Defining an agent's capabilities in terms of solvable declarations effectively creates a vocabulary with which other agents can communicate with the new agent. Ensuring that agents will speak the same language and share a common, unambiguous semantics of the vocabulary involves ontology. Agent development tools and services (automatic translations of solvables by the facilitator) help address this issue; additionally, a preferred embodiment of the present invention will typically rely on vocabulary from either formally engineered ontologies for specific domains or from ontologies constructed during the incremental development of a body of agents for several applications or from both specific domain ontologies and incrementally developed ontologies. Several example tools and services are described in Cheyer et al.'s paper entitled "Development Tools for the Open Agent Architecture," as presented at the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM 96), London, April 1996.

Although the present invention imposes no hard restrictions on the form of solvable declarations, two common usage conventions illustrate some of the utility associated with solvables.

Classes of services are often preferably tagged by a particular type. For instance, in the example above, the "last_message" and "get_message" solvables are specialized for email, not by modifying the names of the services, but rather by the use of the 'email' parameter, which serves during the execution of an ICL request to select (or not) a specific type of message.

Actions are generally written using an imperative verb as the functor of the solvable in a preferred embodiment of the present invention, the direct object (or item class) as the first argument of the predicate, required arguments following, and then an extensible parameter list as the last argument. The parameter list can hold optional information usable by the function. The ICL expression generated by a natural language parser often makes use of this parameter list to store prepositional phrases and adjectives.

As an illustration of the above two points, "Send mail to Bob about lunch" will be translated into an ICL request send message(email, 'Bob Jones', [subject(lunch)]), whereas "Remind Bob about lunch" would leave the transport unspecified (send_message(KIND, 'Bob Jones', [subject (lunch)])), enabling an available message transfer agents (e.g., fax, phone, mail, pager) to compete for the opportunity to carry out the request.

Requesting Services

An agent preferably requests services of the community of agent by delegating tasks or goals to its facilitator. Each request preferably contains calls to one or more agent solvables, and optionally specifies parameters containing advice to help the facilitator determine how to execute the task. Calling a solvable preferably does not require that the agent specify (or even know of) a particular agent or agents to handle the call. While it is possible to specify one or more agents using an address parameter (and there are situations in which this is desirable), in general it is advantageous to leave this delegation to the facilitator. This greatly reduces the hard-coded component dependencies often found in other distributed frameworks. The agent libraries of a preferred embodiment of the present invention provide an agent with a single, unified point of entry for requesting services of other agents: the library procedure oaa_Solve. In the style of logic programming, oaa_Solve may preferably be used both to retrieve data and to initiate actions, so that calling a data solvable looks the same as calling a procedure solvable.

Complex Goal Expressions

A powerful feature provided by preferred embodiments of the present invention is the ability of a client agent (or a user) to submit compound goals of an arbitrarily complex nature to a facilitator. A compound goal is a single goal expression that specifies multiple sub-goals to be performed. In speaking of a "complex goal expression" we mean that a single goal expression that expresses multiple sub-goals can potentially include more than one type of logical connector (e.g., AND, OR, NOT), and/or more than one level of logical nesting (e.g., use of parentheses), or the substantive equivalent. By way of further clarification, we note that when speaking of an "arbitrarily complex goal expression" we mean that goals are expressed in a language or syntax that allows expression of such complex goals when appropriate or when desired, not that every goal is itself necessarily complex.

It is contemplated that this ability is provided through an interagent communication language having the necessary syntax and semantics. In one example, the goals may take the form of compound goal expressions composed using operators similar to those employed by PROLOG, that is, the comma for conjunction, the semicolon for disjunction, the arrow for conditional execution, etc. The present invention also contemplates significant extensions to PROLOG syntax and semantics. For example, one embodiment incorporates a "parallel disjunction" operator indicating that the disjuncts are to be executed by different agents concurrently. A further embodiment supports the specification of whether a given sub-goal is to be executed breadth-first or depth-first.

A further embodiment supports each sub-goal of a compound goal optionally having an address and/or a set of parameters attached to it. Thus, each sub-goal takes the form

Address:Goal::Parameters

where both Address and Parameters are optional.

An address, if present, preferably specifies one or more agents to handle the given goal, and may employ several different types of referring expression: unique names, symbolic names, and shorthand names. Every agent has preferably a unique name, assigned by its facilitator, which relies upon network addressing schemes to ensure its global uniqueness. Preferably, agents also have self-selected symbolic names (for example, "mail"), which are not guaranteed to be unique. When an address includes a symbolic name, the facilitator preferably takes this to mean that all agents having that name should be called upon. Shorthand names include 'self' and 'parent' (which refers to the agent's facilitator). The address associated with a goal or sub-goal is preferably always optional. When an address is not present, it is the facilitator's job to supply an appropriate address.

The distributed execution of compound goals becomes particularly powerful when used in conjunction with natural language or speech-enabled interfaces, as the query itself may specify how functionality from distinct agents will be combined. As a simple example, the spoken utterance "Fax it to Bill Smith's manager." can be translated into the following compound ICL request:

oaa_Solve((manager('Bill Smith', M), fax(it,M,[ ])), [strategy(action)])

Note that in this ICL request there are two sub-goals, "manager('Bill Smith',M)" and "fax(it,M,[ ])," and a single global parameter "strategy(action)." According to the present invention, the facilitator is capable of mapping global parameters in order to apply the constraints or advice across the separate sub-goals in a meaningful way. In this instance, the global parameter strategy(action) implies a parallel constraint upon the first sub-goal; i.e., when there are multiple agents that can respond to the manager sub-goal, each agent should receive a request for service. In contrast, for the second sub-goal, parallelism should not be inferred from the global parameter strategy(action) because such an inference would possibly result in the transmission of duplicate facsimiles.

Refining Service Requests

In a preferred embodiment of the present invention, parameters associated with a goal (or sub-goal) can draw on useful features to refine the request's meaning. For example, it is frequently preferred to be able to specify whether or not solutions are to be returned synchronously; this is done using the reply parameter, which can take any of the values synchronous, asynchronous, or none. As another example, when the goal is a non-compound query of a data solvable, the cache parameter may preferably be used to request local caching of the facts associated with that solvable.

Many of the remaining parameters fall into two categories: feedback and advice. Feedback parameters allow a service requester to receive information from the facilitator about how a goal was handled. This feedback can include such things as the identities of the agents involved in satisfying the goal, and the amount of time expended in the satisfaction of the goal.

Advice parameters preferably give constraints or guidance to the facilitator in completing and interpreting the goal. For example, a solution_limit parameter preferably allows the requester to say how many solutions it is interested in; the facilitator and/or service providers are free to use this information in optimizing their efforts. Similarly, a time_limit is preferably used to say how long the requester is willing to wait for solutions to its request, and, in a multiple facilitator system, a level_limit may preferably be used to say how remote the facilitators may be that are consulted in the search for solutions. A priority parameter is preferably used to indicate that a request is more urgent than previous requests that have not yet been satisfied. Other preferred advice parameters include but are not limited to parameters used to tell the facilitator whether parallel satisfaction of the parts of a goal is appropriate, how to combine and filter results arriving from multiple solver agents, and whether the requester itself may be considered a candidate solver of the sub-goals of a request.

Advice parameters preferably provide an extensible set of low-level, orthogonal parameters capable of combining with the ICL goal language to fully express how information should flow among participants. In certain preferred embodiments of the present invention, multiple parameters can be grouped together and given a group name. The resulting high-level advice parameters can preferably be used to express concepts analogous to KQML's performatives, as well as define classifications of problem types. For instance, KQML's "ask_all" and "ask_one" performatives would be represented as combinations of values given to the parameters reply, parallel ok, and solution_limit. As an example of a higher-level problem type, the strategy "math_problem" might preferably send the query to all appropriate math solvers in parallel, collect their responses, and signal a conflict if different answers are returned. The strategy "essay_question" might preferably send the request to all appropriate participants, and signal a problem (i.e., cheating) if any of the returned answers are identical.

Facilitation

In a preferred embodiment of the present invention, when a facilitator receives a compound goal, its job is to construct a goal satisfaction plan and oversee its satisfaction in an optimal or near optimal manner that is consistent with the specified advice. The facilitator of the present invention maintains a knowledge base that records the capabilities of a collection of agents, and uses that knowledge to assist requesters and providers of services in making contact.

FIG. **7** schematically shows data structures **700** internal to a facilitator in accordance with one embodiment of the present invention. Consider the function of a Agent Registry **702** in the present invention. Each registered agent may be seen as associated with a collection of fields found within its parent facilitator such as shown in the figure. Each registered agent may optionally possess a Symbolic Name which would be entered into field **704**. As mentioned elsewhere, Symbolic Names need not be unique to each instance of an agent. Note that an agent may in certain preferred embodiments of the present invention possess more than one Symbolic Name. Such Symbolic Names would each be found through their associations in the Agent Registry entries. Each agent, when registered, must possess a Unique Address, which is entered into the Unique Address field **706**.

With further reference to FIG. **7**, each registered agent may be optionally associated with one or more capabilities, which have associated Capability Declaration fields **708** in the parent facilitator Agent Registry **702**. These capabilities may define not just functionality, but may further provide a utility parameter indicating, in some manner (e.g., speed, accuracy, etc), how effective the agent is at providing the declared capability. Each registered agent may be optionally associated with one or more data components, which have associated Data Declaration fields **710** in the parent facilitator Agent Registry **702**. Each registered agent may be optionally associated with one or more triggers, which preferably could be referenced through their associated Trigger Declaration fields **712** in the parent facilitator Agent Registry **702**. Each registered agent may be optionally associated with one or more tasks, which preferably could be referenced through their associated Task Declaration fields **714** in the parent facilitator Agent Registry **702**. Each registered agent may be optionally associated with one or more Process Characteristics, which preferably could be referenced through their associated Process Characteristics Declaration fields **716** in the parent facilitator Agent Registry **702**. Note that these characteristics in certain preferred embodiments of the present invention may include one or more of the following: Machine Type (specifying what type of computer may run the agent), Language (both computer and human interface).

A facilitator agent in certain preferred embodiments of the present invention further includes a Global Persistent Database **720**. The database **720** is composed of data elements which do not rely upon the invocation or instantiation of client agents for those data elements to persist. Examples of data elements which might be present in such a database include but are not limited to the network address of the facilitator agent's server, facilitator agent's server accessible network port list, firewalls, user lists, and security options regarding the access of server resources accessible to the facilitator agent.

A simplified walk through of operations involved in creating a client agent, a client agent initiating a service request, a client agent responding to a service request and a facilitator agent responding to a service request are including hereafter by way of illustrating the use of such a system. These figures and their accompanying discussion are provided by way of illustration of one preferred embodiment of the present invention and are not intended to limit the scope of the present invention.

FIG. **8** depicts operations involved in instantiating a client agent with its parent facilitator in accordance with a preferred embodiment of the present invention. The operations begin with starting the Agent Registration in a step **800**. In a next step **802**, the Installer, such as a client or facilitator agent, invokes a new client agent. It will be appreciated that any computer entity is capable of invoking a new agent. The system then instantiates the new client agent in a step **804**. This operation may involve resource allocations somewhere in the network on a local computer system for the client agent, which will often include memory as well as placement of references to the newly instantiated client agent in internal system lists of agents within that local computing system. Once instantiated, the new client and its parent facilitator establish a communications link in a step **806**. In certain preferred embodiments, this communications link involves selection of one or more physical transport mechanisms for this communication. Once established, the client agent transmits it profile to the parent facilitator in a step **808**. When received, the parent facilitator registers the client agent in a step **810**. Then, at a step **812**, a client agent has been instantiated in accordance with one preferred embodiment of the present invention.

FIG. **9** depicts operations involved in a client agent initiating a service request and receiving the response to that service request in accordance with a preferred embodiment of the present invention. The method of FIG. **9** begins in a step **900**, wherein any initialization or other such procedures may be performed. Then, in a step **902**, the client agent determines a goal to be achieved (or solved). This goal is then translated in a step **904** into ICL, if it is not already formulated in it. The goal, now stated in ICL, is then transmitted to the client agent's parent facilitator in a step **906**. The parent facilitator responds to this service request and at a later time, the client agent receives the results of the request in a step **908**, operations of FIG. **9** being complete in a done step **910**.

FIG. **10** depicts operations involved in a client agent responding to a service request in accordance with a preferred embodiment of the present invention. Once started in a step **1000**, the client agent receives the service request in a step **1002**. In a next step **1004**, the client agent parses the received request from ICL. The client agent then determines if the service is available in a step **1006**. If it is not, the client agent returns a status report to that effect in a step **1008**. If the service is available, control is passed to a step **1010** where the client performs the requested service. Note that in completing step **1010** the client may form complex goal expressions, requesting results for these solvables from the facilitator agent. For example, a fax agent might fax a document to a certain person only after requesting and receiving a fax number for that person. Subsequently, the client agent either returns the results of the service and/or a status report in a step **1012**. The operations of FIG. **10** are complete in a done step **1014**.

FIG. **11** depicts operations involved in a facilitator agent response to a service request in accordance with a preferred embodiment of the present invention. The start of such operations in step **1100** leads to the reception of a goal request in a step **1102** by the facilitator. This request is then parsed and interpreted by the facilitator in a step **1104**. The facilitator then proceeds to construct a goal satisfaction plan in a next step **1106**. In steps **1108** and **1110**, respectively, the facilitator determines the required sub-goals and then selects agents suitable for performing the required sub-goals. The facilitator then transmits the sub-goal requests to the selected agents in a step **1112** and receives the results of these transmitted requests in a step **1114**. It should be noted that the actual implementation of steps **1112** and **1114** are dependent upon the specific goal satisfaction plan. For instance, certain sub-goals may be sent to separate agents in parallel, while transmission of other sub-goals may be postponed until receipt of particular answers. Further, certain requests may generate multiple responses that generate additional sub-goals. Once the responses have been received, the facilitator determines whether the original requested goal has been completed in a step **1118**. If the original requested goal has not been completed, the facilitator recursively repeats the operations **1106** through **1116**. Once the original requested goal is completed, the facilitator returns the results to the requesting agent **1118** and the operations are done at **1120**.

A further preferred embodiment of the present invention incorporates transparent delegation, which means that a requesting agent can generate a request, and a facilitator can manage the satisfaction of that request, without the requester needing to have any knowledge of the identities or locations of the satisfying agents. In some cases, such as when the request is a data query, the requesting agent may also be oblivious to the number of agents involved in satisfying a request. Transparent delegation is possible because agents' capabilities (solvables) are treated as an abstract description of a service, rather than as an entry point into a library or body of code.

A further preferred embodiment of the present invention incorporates facilitator handling of compound goals, preferably involving three types of processing: delegation, optimization and interpretation.

Delegation processing preferably supports facilitator determination of which specific agents will execute a compound goal and how such a compound goal's sub-goals will be combined and the sub-goal results routed. Delegation involves selective application of global and local constraint and advice parameters onto the specific sub-goals. Delegation results in a goal that is unambiguous as to its meaning and as to the agents that will participate in satisfying it.

Optimization processing of the completed goal preferably includes the facilitator using sub-goal parallelization where appropriate. Optimization results in a goal whose interpretation will require as few exchanges as possible, between the facilitator and the satisfying agents, and can exploit parallel efforts of the satisfying agents, wherever this does not affect the goal's meaning.

Interpretation processing of the optimized goal. Completing the addressing of a goal involves the selection of one or more agents to handle each of its sub-goals (that is, each sub-goal for which this selection has not been specified by the requester). In doing this, the facilitator uses its knowledge of the capabilities of its client agents (and possibly of other facilitators, in a multi-facilitator system). It may also use strategies or advice specified by the requester, as explained below. The interpretation of a goal involves the coordination of requests to the satisfying agents, and assembling their responses into a coherent whole, for return to the requester.

A further preferred embodiment of present invention extends facilitation so the facilitator can employ strategies and advice given by the requesting agent, resulting in a variety of interaction patterns that may be instantiated in the satisfaction of a request.

A further preferred embodiment of present invention handles the distribution of both data update requests and requests for installation of triggers, preferably using some of the same strategies that are employed in the delegation of service requests.

Note that the reliance on facilitation is not absolute; that is, there is no hard requirement that requests and services be matched up by the facilitator, or that interagent communications go through the facilitator. There is preferably support in the agent library for explicit addressing of requests. However, a preferred embodiment of the present invention encourages employment the paradigm of agent communities, minimizing their development effort, by taking advantage of the facilitator's provision of transparent delegation and handling of compound goals.

A facilitator is preferably viewed as a coordinator, not a controller, of cooperative task completion. A facilitator preferably never initiates an activity. A facilitator preferably responds to requests to manage the satisfaction of some goal, the update of some data repository, or the installation of a trigger by the appropriate agent or agents. All agents can preferably take advantage of the facilitator's expertise in delegation, and its up-to-date knowledge about the current membership of a dynamic community. The facilitator's coordination services often allows the developer to lessen the complexity of individual agents, resulting in a more manageable software development process, and enabling the creation of lightweight agents.

Maintaining Data Repositories

The agent library supports the creation, maintenance, and use of databases, in the form of data solvables. Creation of a data solvable requires only that it be declared. Querying a data solvable, as with access to any solvable, is done using oaa_Solve.

A data solvable is conceptually similar to a relation in a relational database. The facts associated with each solvable are maintained by the agent library, which also handles incoming messages containing queries of data solvables. The default behavior of an agent library in managing these facts may preferably be refined, using parameters specified with the solvable's declaration. For example, the parameter single_value preferably indicates that the solvable should only contain a single fact at any given point in time. The parameter unique_values preferably indicates that no duplicate values should be stored.

Other parameters preferably allow data solvables use of the concepts of ownership and persistence. For implementing shared repositories, it is often preferable to maintain a record of which agent created each fact of a data solvable with the creating agent being preferably considered the fact's owner. In many applications, it is preferable to remove an agent's facts when that agent goes offline (for instance, when the agent is no longer participating in the agent community, whether by deliberate termination or by malfunction). When a data solvable is declared to be non-persistent, its facts are automatically maintained in this way, whereas a persistent data solvable preferably retains its facts until they are explicitly removed.

A further preferred embodiment of present invention supports an agent library through procedures by which agents can update (add, remove, and replace) facts belonging to data solvables, either locally or on other agents, given that they have preferably the required permissions. These procedures may preferably be refined using many of the same parameters that apply to service requests. For example, the address parameter preferably specifies one or more particular agents to which the update request applies. In its absence, just as with service requests, the update request preferably goes to all agents providing the relevant data solvable. This default behavior can be used to maintain coordinated "mirror" copies of a data set within multiple agents, and can be useful in support of distributed, collaborative activities.

Similarly, the feedback parameters, described in connection with oaa_Solve, are preferably available for use with data maintenance requests.

A further preferred embodiment of present invention supports ability to provide data solvables not just to client agents, but also to facilitator agents. Data solvables can preferably created, maintained and used by a facilitator. The facilitator preferably can, at the request of a client of the facilitator, create, maintain and share the use of data solvables with all the facilitator's clients. This can be useful with relatively stable collections of agents, where the facilitator's workload is predictable.

Using a Blackboard Style of Communication

In a further preferred embodiment of present invention, when a data solvable is publicly readable and writable, it acts essentially as a global data repository and can be used cooperatively by a group of agents. In combination with the use of triggers, this allows the agents to organize their efforts around a "blackboard" style of communication.

As an example, the "DCG-NL" agent (one of several existing natural language processing agents), provides natural language processing services for a variety of its peer agents, expects those other agents to record, on the facilitator, the vocabulary to which they are prepared to respond, with an indication of each word's part of speech, and of the logical form (ICL sub-goal) that should result from the use of that word. In a further preferred embodiment of present invention, the NL agent, preferably when it comes online, preferably installs a data solvable for each basic part of speech on its facilitator. For instance, one such solvable would be:

solvable(noun(Meaning, Syntax), [ ], [ ])

Note that the empty lists for the solvable's permissions and parameters are acceptable here, since the default permissions and parameters provide appropriate functionality.

A further preferred embodiment of present invention incorporating an Office Assistant system as discussed herein or similar to the discussion here supports several agents making use of these or similar services. For instance, the database agent uses the following call, to library procedure oaa_AddData, to post the noun 'boss', and to indicate that the "meaning" of boss is the concept 'manager':

oaa_AddData(noun(manager, atom(boss)), [address (parent)])

Autonomous Monitoring with Triggers

A further preferred embodiment of present invention includes support for triggers, providing a general mechanism for requesting some action be taken when a set of conditions is met. Each agent can preferably install triggers either locally, for itself, or remotely, on its facilitator or peer agents. There are preferably at least four types of triggers: communication, data, task, and time. In addition to a type, each trigger preferably specifies at least a condition and an action, both preferably expressed in ICL. The condition indicates under what circumstances the trigger should fire, and the action indicates what should happen when it fires. In addition, each trigger can be set to fire either an unlimited number of times, or a specified number of times, which can be any positive integer.

Triggers can be used in a variety of ways within preferred embodiments of the present invention. For example, triggers can be used for monitoring external sensors in the execution environment, tracking the progress of complex tasks, or coordinating communications between agents that are essential for the synchronization of related tasks. The installation of a trigger within an agent can be thought of as a representation of that agent's commitment to carry out the specified action, whenever the specified condition holds true.

Communication triggers preferably allow any incoming or outgoing event (message) to be monitored. For instance, a simple communication trigger may say something like: "Whenever a solution to a goal is returned from the facilitator, send the result to the presentation manager to be displayed to the user."

Data triggers preferably monitor the state of a data repository (which can be maintained on a facilitator or a client agent). Data triggers' conditions may be tested upon the addition, removal, or replacement of a fact belonging to a data solvable. An example data trigger is: "When 15 users are simultaneously logged on to a machine, send an alert message to the system administrator."

Task triggers preferably contain conditions that are tested after the processing of each incoming event and whenever a timeout occurs in the event polling. These conditions may specify any goal executable by the local ICL interpreter, and most often are used to test when some solvable becomes satisfiable. Task triggers are useful in checking for task-specific internal conditions. Although many cases such conditions are captured by solvables, in other cases they may not be. 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. An example task trigger is: "When mail arrives for me about security, notify me immediately."

Time triggers preferably monitor time conditions. For instance, an alarm trigger can be set to fire at a single fixed point in time (e.g., "On December 23rd at 3 pm"), or on a recurring basis (e.g., "Every three minutes from now until noon").

Triggers are preferably implemented as data solvables, declared implicitly for every agent. When requesting that a trigger be installed, an agent may use many of the same parameters that apply to service and data maintenance requests.

A further preferred embodiment of present invention incorporates semantic support, in contrast with most programming methodologies, of the agent on which the trigger is installed only having to know how to evaluate the conditional part of the trigger, not the consequence. When the trigger fires, the action is delegated to the facilitator for execution. Whereas many commercial mail programs allow rules of the form "When mail arrives about XXX, [forward it, delete it, archive it]", the possible actions are hard-coded and the user must select from a fixed set.

A further preferred embodiment of present invention, the consequence of a trigger may be any compound goal executable by the dynamic community of agents. Since new agents preferably define both functionality and vocabulary, when an unanticipated agent (for example, a fax agent) joins the community, no modifications to existing code is required for a user to make use of it—"When mail arrives, fax it to Bill Smith."

The Agent Library

In a preferred embodiment of present invention, the agent library provides the infrastructure for constructing an agent-based system. The essential elements of protocol (involving the details of the messages that encapsulate a service request and its response) are preferably made transparent to simplify the programming applications. This enables the developer to focus functionality, rather than message construction details and communication details. For example, to request a service of another agent, an agent preferably calls the library procedure oaa_Solve. This call results in a message to a facilitator, which will exchange messages with one or more service providers, and then send a message containing the desired results to the requesting agent. These results are returned via one of the arguments of oaa_Solve. None of the messages involved in this scenario is explicitly constructed by the agent developer. Note that this describes the synchronous use of oaa_Solve.

In another preferred embodiment of present invention, an agent library provides both intraagent and interagent infrastructure; that is, mechanisms supporting the internal structure of individual agents, on the one hand, and mechanisms of cooperative interoperation between agents, on the other. Note that most of the infrastructure cuts across this boundary with many of the same mechanisms supporting both agent internals and agent interactions in an integrated fashion. For example, services provided by an agent preferably can be accessed by that agent through the same procedure (oaa_Solve) that it would employ to request a service of another agent (the only difference being in the address parameter accompanying the request). This helps the developer to reuse code and avoid redundant entry points into the same functionality.

Both of the preferred characteristics described above (transparent construction of messages and integration of intraagent with interagent mechanisms) apply to most other library functionality as well, including but not limited to data management and temporal control mechanisms.

Source Code Appendix

Source code for version 2.0 of the OAA software product is included as an appendix hereto, and is incorporated herein by reference. The code includes an agent library, which provides infrastructure for constructing an agent-based system. The library's several families of procedures provide the functionalities discussed above, as well as others that have not been discussed here but that will be sufficiently clear to the interested practitioner. For example, declarations of an agent's solvables, and their registration with a facilitator, are managed using procedures such as oaa_Declare, oaa_Undeclare, and oaa_Redeclare. Updates to data solvables can be accomplished with a family of procedures including oaa_AddData, oaa_RemoveData, and oaa_ReplaceData. Similarly, triggers are maintained using procedures such as oaa_AddTrigger, oaa_RemoveTrigger, and oaa_ReplaceTrigger. The provided source code also includes source code for an OAA Facilitator Agent.

The source code appendix is offered solely as a means of further helping practitioners to construct a preferred embodiment of the invention. By no means is the source code intended to limit the scope of the present invention.

Illustrative Applications

To further illustrate the technology of the preferred embodiment, we will next present and discuss two sample applications of the present inventions.

Unified Messaging

A further preferred embodiment of present invention incorporates a Unified Messaging application extending the Automated Office application presented previously herein with an emphasis on ubiquitous access and dynamic presentation of the information and services supported by the agent community. The agents used in this application are depicted in FIG. **12**.

A hypothetical example of realistic dialog using a preferred embodiment of the present invention can provide insight into how systems may preferably be built using the present invention. In this scenario, the user, with only a telephone as an interface, is planning a trip to Boston where he will soon give a presentation. Capitalized sentences are phrases spoken by the user into the telephone and processed by a phone agent **452**.

Responses, unless otherwise indicated, are spoken by the system using text-to-speech generation agent **454**.

1.1 Welcome to SRI International. Please enter your user ID and password.

<User enters touchtone ID and password>

Good to see you again Adam Cheyer. I am listening to you.

Every user interface agent **408**, including the telephone agent **452**, should know the identify of its user. This information is used in resolving anaphoric references such as "Me" and "I", and allows multiple user interfaces operated by the same user to work together.

1.2 WHAT IS TODAY'S SCHEDULE?

Here is today's schedule for Adam Cheyer:

At 10 am for 1 hour, meeting with Dave.

At 3 pm for 1 hour, presentation about software agents. End of schedule.

If the user is operating both a graphical user interface and a telephone, as described in conjunction with the Automated Office application, the result of this spoken request is to display a calendar window containing the current schedule. In this case, with no graphical display available, the GEN_NL agent **1202** is tasked to produce a spoken response that can be played over the phone. GEN_NL shares the same dynamic vocabulary and phrasal rules as the natural language parser DCG_NL **426**, and contains strategies for producing responses to queries using either simple or list-based multimedia utterances.

1.3 FIND FRIDAY'S WEATHER IN BOSTON.

The weather in Boston for Friday is as follows:

Sunny in the morning. Partly cloudy in the afternoon with a 20

percent chance of thunderstorms late. Highs in the mid 70s.

In addition to data accessible from legacy applications, content may be retrieved by web-reading agents which provide wrappers around useful websites.

1.4 FIND ALL NEW MAIL MESSAGES.

There are 2 messages available.

Message 1, from Mark Tierny, entitled "OAA meeting."

1.5 NEXT MESSAGE

Message 2, from Jennifer Schwefler, entitled "Presentation Summary."

1.6 PLAY IT.

This message is a multipart MIME-encoded message. There are two parts.

Part **1**. (Voicemail message, not text-to speech):

Thanks for taking part as a speaker in our conference.

The schedule will be posted soon on our homepage.

1.7 NEXT PART

Part **2**. (read using text-to-speech):

The presentation home page is http://www . . .

1.8 PRINT MESSAGE

Command executed.

Mail messages are no longer just simple text documents, but often consist of multiple subparts containing audio files, pictures, webpages, attachments and so forth. When a user asks to play a complex email message over the telephone, many different agents may be implicated in the translation process, which would be quite different given the request "print it." The challenge is to develop a system which will enable agents to cooperate in an extensible, flexible manner that alleviates explicit coding of agent interactions for every possible input/output combination.

In a preferred embodiment of the present invention, each agent concentrates only on what it can do and on what it knows, and leaves other work to be delegated to the agent community. For instance, a printer agent **1204**, defining the solvable print(Object,Parameters), can be defined by the following pseudo-code, which basically says, "If someone can get me a document, in either POSTSCRIPT or text form, I can print it.".

```
print(Object, Parameters) {
  ' If Object is reference to "it", find an appropriate document
  if (Object ="ref(it)")
    oaa_Solve(resolve_reference(the, document, Params, Object),[ ]);
  ' Given a reference to some document, ask for the document in POSTSCRIPT
  if (Object ="id(Pointer)")
    oaa_Solve(resolve_id_as(id(Pointer), postscript, [ ], Object), [ ]);
  ' If Object is of type text or POSTSCRIPT, we can print it.
  if ((Object is of type Text) or (Object is of type Postscript))
    do print (Object);
}
```

In the above example, since an email message is the salient document, the mail agent **442** will receive a request to produce the message as POSTSCRIPT. Whereas the mail agent **442** may know how to save a text message as POSTSCRIPT, it will not know what to do with a webpage or voicemail message. For these parts of the message, it will simply send oaa_Solve requests to see if another agent knows how to accomplish the task.

Until now, the user has been using only a telephone as user interface. Now, he moves to his desktop, starts a web browser **436**, and accesses the URL referenced by the mail message.

1.9 RECORD MESSAGE

Recording voice message. Start speaking now.

1.10 THIS IS THE UPDATED WEB PAGE CONTAINING THE PRESENTATION SCHEDULE.

Message one recorded.

1.11 IF THIS WEB PAGE CHANGES, GET IT TO ME WITH NOTE ONE.

Trigger added as requested.

In this example, a local agent **436** which interfaces with the web browser can return the current page as a solution to the request "oaa_Solve(resolve_reference(this, web_page, [ ], Ref),[ ])", sent by the NL agent **426**. A trigger is installed on a web agent **436** to monitor changes to the page, and when the page is updated, the notify agent **446** can find the user and transmit the webpage and voicemail message using the most appropriate media transfer mechanism.

This example based on the Unified Messaging application is intended to show how concepts in accordance with the present invention can be used to produce a simple yet extensible solution to a multi-agent problem that would be difficult to implement using a more rigid framework. The application supports adaptable presentation for queries across dynamically changing, complex information; shared context and reference resolution among applications; and flexible translation of multimedia data. In the next section, we will present an application which highlights the use of parallel competition and cooperation among agents during multi-modal fusion.

Multimodal Map

A further preferred embodiment of present invention incorporates the Multimodal Map application. This application demonstrates natural ways of communicating with a community of agents, providing an interactive interface on which the user may draw, write or speak. In a travel-planning domain illustrated by FIG. **13**, available information includes hotel, restaurant, and tourist-site data retrieved by distributed software agents from commercial Internet sites. Some preferred types of user interactions and multimodal issues handled by the application are illustrated by a brief scenario featuring working examples taken from the current system.

Sara is planning a business trip to San Francisco, but would like to schedule some activities for the weekend while she is there. She turns on her laptop PC, executes a map application, and selects San Francisco.

2.1 [Speaking] Where is downtown?

Map scrolls to appropriate area.

2.2 [Speaking and drawing region] Show me all hotels near here.

Icons representing hotels appear.

2.3 [Writes on a hotel] Info?

A textual description (price, attributes, etc.) appears.

2.4 [Speaking] I only want hotels with a pool. Some hotels disappear.

2.5 [Draws a crosscut on a hotel that is too close to a highway)

Hotel disappears

2.6 [Speaking and circling] Show me a photo of this hotel.

Photo appears.

2.7 (Points to another hotel]

Photo appears.

2.8 [Speaking] Price of the other hotel?

Price appears for previous hotel.

2.9 [Speaking and drawing an arrow] Scroll down.

Display adjusted.

2.10 [Speaking and drawing an arrow toward a hotel]

What is the distance from this hotel to Fisherman's Wharf?

Distance displayed.

2.11 [Pointing to another place and speaking] And the distance to here?

Distance displayed.

Sara decides she could use some human advice. She picks up the phone, calls Bob, her travel agent, and writes Start collaboration to synchronize his display with hers. At this point, both are presented with identical maps, and the input and actions of one will be remotely seen by the other.

3.1 [Sara speaks and circles two hotels]

Bob, I'm trying to choose between these two hotels. Any opinions?

3.2 [Bob draws an arrow, speaks, and points]

Well, this area is really nice to visit. You can walk there from

this hotel.

Map scrolls to indicated area. Hotel selected.

3.3 [Sara speaks] Do you think I should visit Alcatraz?

3.4 [Bob speaks] Map, show video of Alcatraz.

Video appears.

3.5 [Bob speaks] Yes, Alcatraz is a lot of fun.

A further preferred embodiment of present invention generates the most appropriate interpretation for the incoming streams of multimodal input. Besides providing a user interface to a dynamic set of distributed agents, the application is preferably built using an agent framework. The present invention also contemplates aiding the coordinate competition and cooperation among information sources, which in turn works in parallel to resolve the ambiguities arising at every level of the interpretation process: low-level processing of the data stream, anaphora resolution, cross-modality influences and addressee.

Low-level processing of the data stream: Pen input may be preferably interpreted as a gesture (e.g., 2.5: cross-out) by one algorithm, or as handwriting by a separate recognition process (e.g., 2.3: "info?"). Multiple hypotheses may preferably be returned by a modality recognition component.

Anaphora resolution: When resolving anaphoric references, separate information sources may contribute to resolving the reference: context by object type, deictic, visual context, database queries, discourse analysis. An example of information provided through context by object type is found in interpreting an utterance such as "show photo of the hotel", where the natural language component can return a list of the last hotels talked about. Deictic information in combination with a spoken utterance like "show photo of this hotel" may preferably include pointing, circling, or arrow gestures which might indicate the desired object (e.g., 2.7). Deictic references may preferably occur before, during, or after an accompanying verbal command. Information provided in a visual context, given for the request "display photo of the hotel" may preferably include the user interface agent might determine that only one hotel is currently visible on the map, and therefore this might be the desired reference object. Database queries preferably involving information from a database agent combined with results from other resolution strategies. Examples are "show me a photo of the hotel in Menlo Park" and 2.2. Discourse analysis preferably provides a source of information for phrases such as "No, the other one" (or 2.8).

The above list of preferred anaphora resolution mechanisms is not exhaustive. Examples of other preferred resolution methods include but are not limited to spatial reasoning ("the hotel between Fisherman's Wharf and Lombard Street") and user preferences ("near my favorite restaurant").

Cross-modality influences: When multiple modalities are used together, one modality may preferably reinforce or remove or diminish ambiguity from the interpretation of another. For instance, the interpretation of an arrow gesture may vary when accompanied by different verbal commands (e.g., "scroll left" vs. "show info about this hotel"). In the latter example, the system must take into account how accurately and unambiguously an arrow selects a single hotel.

Addressee: With the addition of collaboration technology, humans and automated agents all share the same workspace. A pen doodle or a spoken utterance may be meant for either another human, the system (3.1), or both (3.2).

The implementation of the Multimodal Map application illustrates and exploits several preferred features of the present invention: reference resolution and task delegation by parallel parameters of oaa_Solve, basic multi-user collaboration handled through built-in data management services, additional functionality readily achieved by adding new agents to the community, domain-specific code cleanly separated from other agents.

A further preferred embodiment of present invention provides reference resolution and task delegation handled in a distributed fashion by the parallel parameters of oaa_Solve, with meta-agents encoding rules to help the facilitator make context- or user-specific decisions about priorities among knowledge sources.

A further preferred embodiment of present invention provides basic multi-user collaboration handled through at least one built-in data management service. The map user interface preferably publishes data solvables for elements such as icons, screen position, and viewers, and preferably defines these elements to have the attribute "shareable". For every update to this public data, the changes are preferably automatically replicated to all members of the collaborative session, with associated callbacks producing the visible effect of the data change (e.g., adding or removing an icon).

Functionality for recording and playback of a session is preferably implemented by adding agents as members of the collaborative community. These agents either record the data changes to disk, or read a log file and replicate the changes in the shared environment.

The domain-specific code for interpreting travel planning dialog is preferably separated from the speech, natural language, pen recognition, database and map user interface agents. These components were preferably reused without modification to add multimodal map capabilities to other applications for activities such as crisis management, multi-robot control, and the MVIEWS tools for the video analyst.

Improved Scalability and Fault Tolerance

Implementations of a preferred embodiment of present invention which rely upon simple, single facilitator architectures may face certain limitations with respect to scalability, because the single facilitator may become a communications bottleneck and may also represent a single, critical point for system failure.

Multiple facilitator systems as disclosed in the preferred embodiments to this point can be used to construct peer-to-peer agent networks as illustrated in FIG. **14**. While such embodiments are scalable, they do possess the potential for communication bottlenecks as discussed in the previous paragraph and they further possess the potential for reliability problems as central, critical points of vulnerability to systems failure.

A further embodiment of present invention supports a facilitator implemented as an agent like any other, whereby multiple facilitator network topologies can be readily constructed. One example configuration (but not the only possibility) is a hierarchical topology as depicted in FIG. **15**, where a top level Facilitator manages collections of both client agents **1508** and other Facilitators, **1504** and **1506**. Facilitator agents could be installed for individual users, for a group of users, or as appropriate for the task.

Note further, that network work topologies of facilitators can be seen as graphs where each node corresponds to an instance of a facilitator and each edge connecting two or more nodes corresponds to a transmission path across one or more physical transport mechanisms. Some nodes may represent facilitators and some nodes may represent clients. Each node can be further annotated with attributes corresponding to include triggers, data, capabilities but not limited to these attributes.

A further embodiment of present invention provides enhanced scalability and robustness by separating the planning and execution components of the facilitator. In contrast with the centralized facilitation schemes described above, the facilitator system **1600** of FIG. **16** separates the registry/planning component from the execution component. As a result, no single facilitator agent must carry all communications nor does the failure of a single facilitator agent shut down the entire system.

Turning directly to FIG. **16**, the facilitator system **1600** includes a registry/planner **1602** and a plurality of client agents **1612–1616**. The registry/planner **1604** is typically replicated in one or more locations accessible by the client agents. Thus if the registry/planner **1604** becomes unavailable, the client agents can access the replicated registry/planner(s).

This system operates, for example, as follows. An agent transmits a goal **1610** to the registry planner **1602**. The registry/planner **1604** translates the goal into an unambiguous execution plan detailing how to accomplish any sub-goals developed from the compound goal, as well as specifying the agents selected for performing the sub-goals. This execution plan is provided to the requesting agent which in turn initiates peer-to-peer interactions **1618** in order to implement the detailed execution plan, routing and combining information as specified within the execution plan. Communication is distributed thus decreasing sensitivity of the system to bandwidth limitations of a single facilitator agent. Execution state is likewise distributed thus enabling system operation even when a facilitator agent fails.

Further embodiments of present invention incorporate into the facilitator functionality such as load-balancing, resource management, and dynamic configuration of agent locations and numbers, using (for example) any of the topologies discussed. Other embodiments incorporate into a facilitator the ability to aid agents in establishing peer-to-peer communications. That is, for tasks requiring a sequence of exchanges between two agents, the facilitator assists the agents in finding one another and establishing communication, stepping out of the way while the agents communicate peer-to-peer over a direct, perhaps dedicated channel.

Further preferred embodiments of the present invention incorporate mechanisms for basic transaction management, such as periodically saving the state of agents (both facilitator and client) and rolling back to the latest saved state in the event of the failure of an agent.

What is claimed is:

1. 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. 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. 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. 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.

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 include a natural language query, and the second 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 compute-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. 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.

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-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. 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), where in 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 request 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 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 agents registry

establishing a bi-directional communication 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;

instating 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. 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 format 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 at 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 format 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 in the 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.

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 instating and executing the trigger within the facilitator agent.

44. A computer program as recited in claim **38** further including computer executable instructions for instating 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. 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 perform queries of other agents;

enabling agents to exchange information with other gents; 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. An ICL as recited in claim **48**, wherein the ICL is computer platform independent.

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 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.

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. 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 requesting 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. 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.

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. 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. 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 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.

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. 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. A computer architecture as recited in claim **71** wherein an execution component of the facilitating engine is distributed across at least two computer process.

86. 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. 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. 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.

* * * * *

define user preferences comprises means for providing a user with an opportunity to designate a preference level for a plurality of preference attributes.

23. The system defined in claim 20 further comprising means for providing software from the program guide server to the program guide client according to the user preferences.

24. The system defined in claim 20 further comprising means for providing Internet links from the program guide server to the program guide client according to the user preferences.

25. A client-server interactive television program guide system for scheduling reminders according to user defined expressions, comprising:

means for providing a user with an opportunity to define an expression with an interactive television program guide client implemented on user television equipment, without requiring the user to navigate the Internet;

means for processing the expression with a program guide server to find programs that satisfy the expression; and

means for scheduling with the program guide server reminders for programs that satisfy the expression.

26. The system defined in claim 25 wherein the means for scheduling with the program guide server reminders for programs that satisfy the expression comprises means for providing at least one message from the program guide server to the program guide client

before each of the programs that satisfy the expression begin.

27. The system defined in claim 25 wherein the means for scheduling with the program guide server reminders for programs that satisfy the expression comprises means for providing program identifiers for each of the programs that satisfy the expression from the program guide server to the program guide client.

28. A client-server interactive television program guide system for scheduling programs for recording according to user defined expressions, comprising:

means for providing a user with an opportunity to define an expression with an interactive television program guide client implemented on user television equipment, without requiring the user to navigate the Internet;

means for processing the expression with a program guide server to find programs that satisfy the expression; and

means for scheduling with the program guide server the programs that satisfy the expression for recording.

29. The system defined in claim 28 wherein the means for scheduling with the program guide server the programs that satisfy the expression for recording comprises means for scheduling with the program guide server the programs that satisfy the expression for recording by the user television equipment.

30. The system defined in claim 28 wherein the means for scheduling with the program guide server

the programs that satisfy the expression for recording comprises means for scheduling with the program guide server the programs that satisfy the expression for recording by the program guide server.

31. A client-server interactive television program guide system for parentally controlling programs according to user defined expressions, comprising:

means for providing a user with an opportunity to define an expression with an interactive television program guide client implemented on user television equipment, without requiring the user to navigate the Internet;

means for processing the expression with a program guide server to find programs that satisfy the expression; and

means for locking with the program guide server programs that satisfy the expression.

32. The system defined in claim 31 wherein the means for locking with the program guide server programs that satisfy the expression comprises means for indicating to the program guide client that the programs that satisfy the expression are locked.

33. A client-server interactive television program guide system for tracking a user's viewing history, comprising:

means for tracking a user's viewing history with a program guide server;

means for indicating on user television equipment programs that are consistent with the user's viewing history and that the user has not watched, with an interactive television program guide client implemented on the user television equipment.

34. The system defined in claim 33 wherein the means for tracking the user's viewing history comprises means for storing a user defined expression with the program guide server.

35. The system defined in claim 33 wherein the means for tracking the user's viewing history comprises means for calculating user demographic values with the program guide server.

36. The system defined in claim 33 further comprising:

means for providing a user with an opportunity to define a user preference profile with the interactive television program guide client implemented on user television equipment; and

means for finding programs with the program guide server that are consistent with the user preference profile, wherein:

the means for indicating on user television equipment the programs found by the program guide server that are consistent with the user's viewing history and that the user has not watched comprises means for indicating on user television equipment the programs found by the program guide server that are consistent with the user's viewing history and the user preference profile and that the user has not watched.

37. The system defined in claim 36 further comprising:

means for targeting advertising with the program guide server based on the user's viewing history; and

means for displaying the advertising with the interactive television program guide client on the user television equipment.

38. The system defined in claim 36 further comprising means for collecting program ratings information with the program guide server based on the user's viewing history.

39. A client-server interactive television program guide system comprising:

a program guide server;

user television equipment on which an interactive television program guide client is implemented, wherein the interactive television program guide client is programmed to provide a user with an opportunity to define user preferences without requiring the user to navigate the Internet; and

a communications path over which the user preferences are provided by the interactive television program guide client to the program guide server.

40. The system defined in claim 39 wherein:

the program guide server is programmed to generate a viewing recommendation based on the user preferences; and

the interactive television program guide client is further programmed to display the viewing recommendation on the user television equipment.

41. The system defined in claim 39 wherein the interactive television program guide client is further programmed to provide a user with an opportunity to designate a preference level for a plurality of preference attributes.

42. The system defined in claim 39 wherein the program guide server is programmed to provide software to the interactive television program guide client according to the user preferences.

43. The system defined in claim 39 wherein the program guide server is programmed to provide Internet links to the interactive television program guide client according to the user preferences.

44. A client-server interactive television program guide system for scheduling reminders according to user defined expressions, comprising:

user television equipment on which an interactive television program guide client is implemented, wherein the program guide client is programmed to provide a user with an opportunity to define an expression without requiring the user to navigate the Internet;

a communications path over which the expression is provided by the interactive television program guide client to a program guide server, wherein the program guide server is programmed to find programs that satisfy the expression and schedule reminders for programs that satisfy the expression.

45. The system defined in claim 44 wherein scheduling with the program guide server reminders for programs that satisfy the expression comprises

providing at least one message from the program guide server to the program guide client before each of the programs that satisfy the expression begin.

46. The system defined in claim 44 wherein the program guide server is further programmed to provide program identifiers for each of the programs that satisfy the expression to the interactive television program guide client over the communications path.

47. A client-server interactive television program guide system for scheduling programs for recording according to user defined expressions, comprising:

user television equipment on which an interactive television program guide client is implemented, wherein the interactive television program guide client is programmed to provide a user with an opportunity to define an expression without requiring the user to navigate the Internet;

a communications path over which the expression is provided by the interactive television program guide client to a program guide server, wherein the program guide server is programmed to find programs that satisfy the expression and schedule the programs that satisfy the expression for recording.

48. The system defined in claim 47 wherein:

the user television equipment comprises a storage device; and

the program guide server is further programmed to schedule the programs that satisfy the expression for recording by the storage device.

49. The system defined in claim 47 wherein the program guide server comprises a storage device on which the programs that satisfy the expression are stored.

50. A client-server interactive television program guide system for parentally controlling programs according to user defined expressions, comprising:

user television equipment on which an interactive television program guide client is implemented, wherein the interactive television program guide client is programmed to provide a user with an opportunity to define an expression without requiring the user to navigate the Internet;

a communications path over which the interactive television program guide client provides the expression to a program guide server, wherein the program guide server is programmed to find programs that satisfy the expression and lock programs that satisfy the expression.

51. The system defined in claim 50 wherein the program guide server is programmed to indicate to the interactive television program guide client the locked programs over the communications path; and

the interactive television program guide client is further programmed to indicate to the user the locked programs with the user television equipment.

52. A client-server interactive television program guide system for tracking a user's viewing history, comprising:

user television equipment on which an interactive television program guide client is implemented, wherein the interactive television program guide client is programmed to provide viewing history information to a program guide server over a communications path, wherein:

the program guide server is programmed to find programs based on the viewing history information and to indicate the programs to the interactive television program guide client over the communications path; and

the interactive television program guide client is further programmed to indicate on the user television equipment a subset of the programs wherein the subset of the programs are programs that the user has not watched.

53. The system defined in claim 52 wherein the program guide server is further programmed to calculate user demographic values based on the viewing history information.

54. The system defined in claim 52 wherein:

the interactive television program guide client is further programmed to provide user preference information to the program guide server over the communications path; and

the program guide server is further programmed to obtain programs based on the user preference information and to indicate the programs to the interactive television program guide client.

55. The system defined in claim 54 wherein:

the program guide server is programmed to target advertisements based on the user preference information and to provide the advertisements to the interactive television program guide client over the communications path; and

the interactive television program guide client is further programmed to display the advertisements on the user television equipment.

56. The system defined in claim 54 wherein the program guide server is further programmed to collect program ratings information based on the viewing history information.

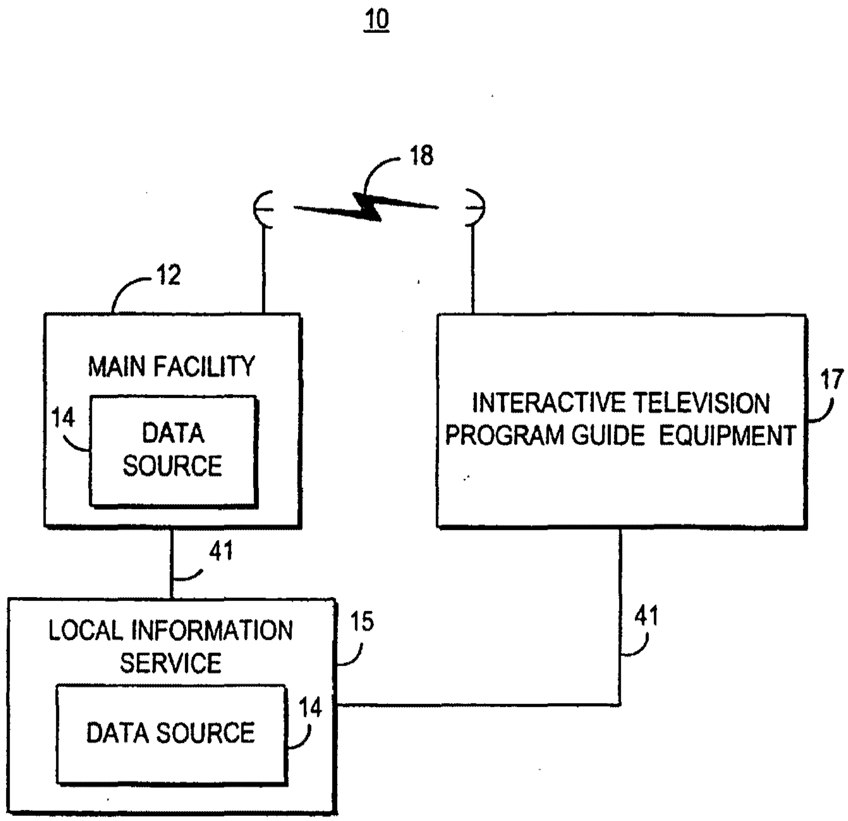

10
18
12
MAIN FACILITY
14
DATA SOURCE
INTERACTIVE TELEVISION PROGRAM GUIDE EQUIPMENT
17
41
LOCAL INFORMATION SERVICE
15
DATA SOURCE
14
41

FIG. 1

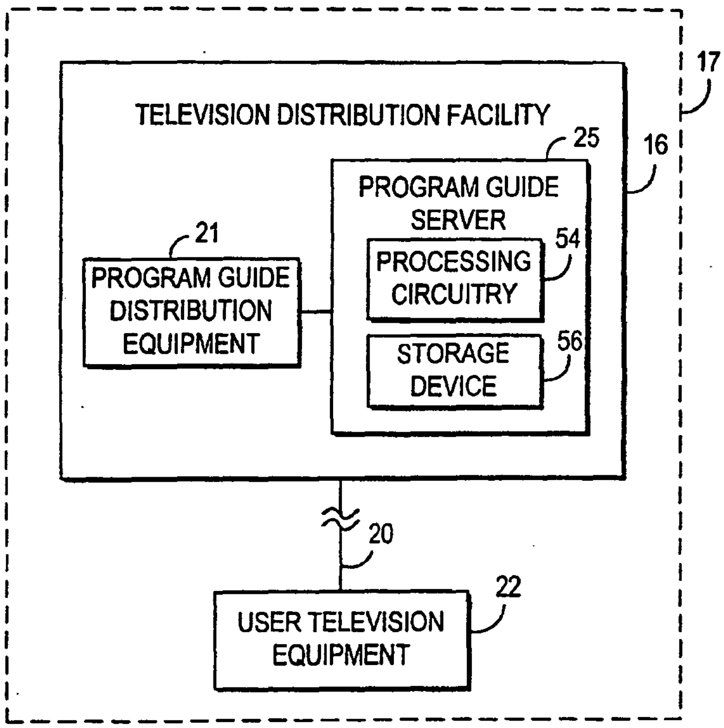
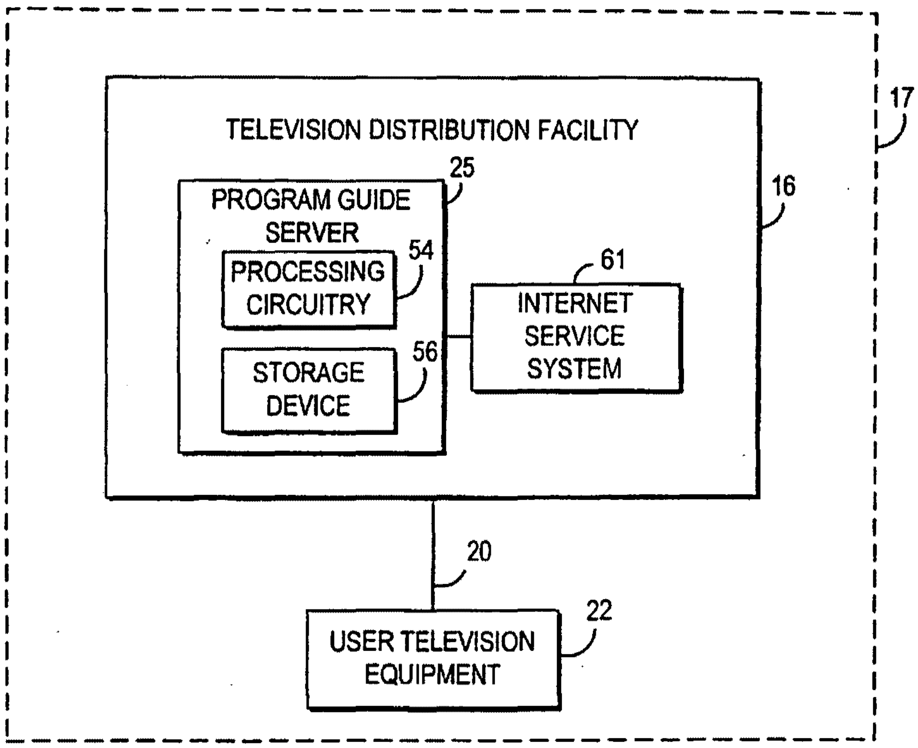

TELEVISION DISTRIBUTION FACILITY
PROGRAM GUIDE SERVER
PROCESSING CIRCUITRY
STORAGE DEVICE
PROGRAM GUIDE DISTRIBUTION EQUIPMENT
USER TELEVISION EQUIPMENT
17
16
25
21
54
56
20
22

FIG. 2a



TELEVISION DISTRIBUTION FACILITY
PROGRAM GUIDE SERVER
PROCESSING CIRCUITRY
STORAGE DEVICE
INTERNET SERVICE SYSTEM
USER TELEVISION EQUIPMENT
17
16
25
54
56
61
20
22

FIG. 2b

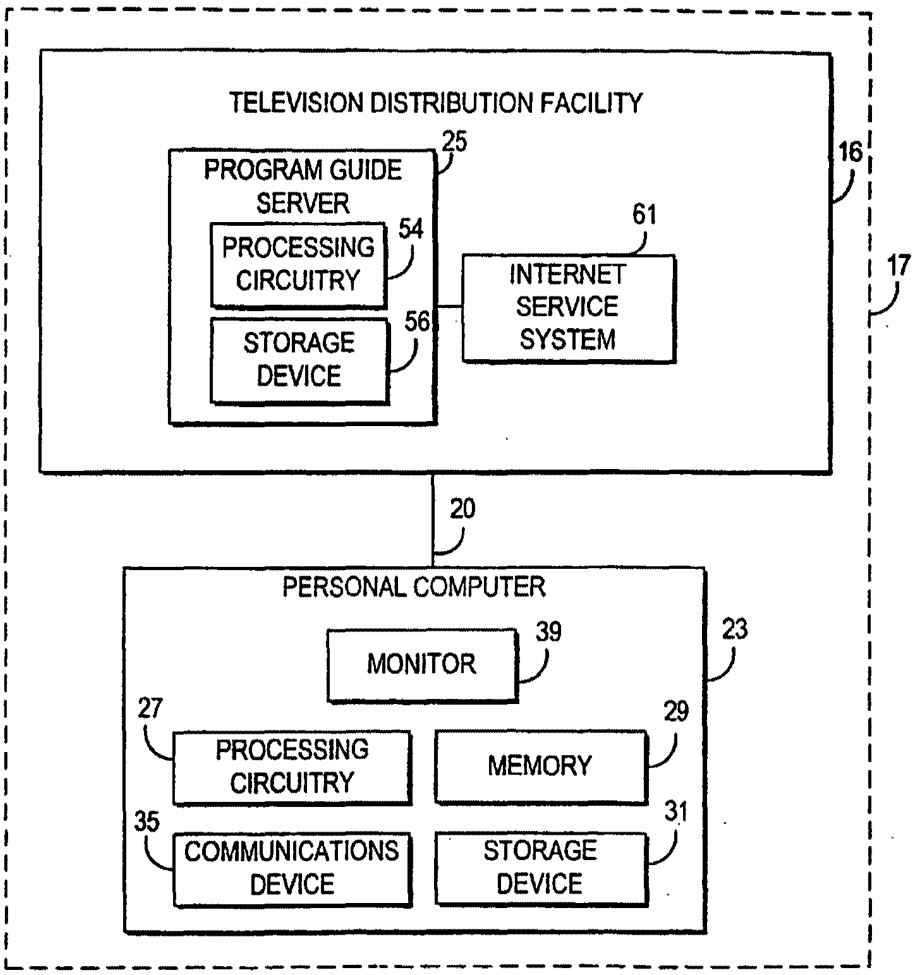

TELEVISION DISTRIBUTION FACILITY
PROGRAM GUIDE SERVER
25
16
PROCESSING CIRCUITRY
54
61
INTERNET SERVICE SYSTEM
17
STORAGE DEVICE
56
20
PERSONAL COMPUTER
MONITOR
39
23
27
29
PROCESSING CIRCUITRY
MEMORY
35
31
COMMUNICATIONS DEVICE
STORAGE DEVICE

FIG. 2c

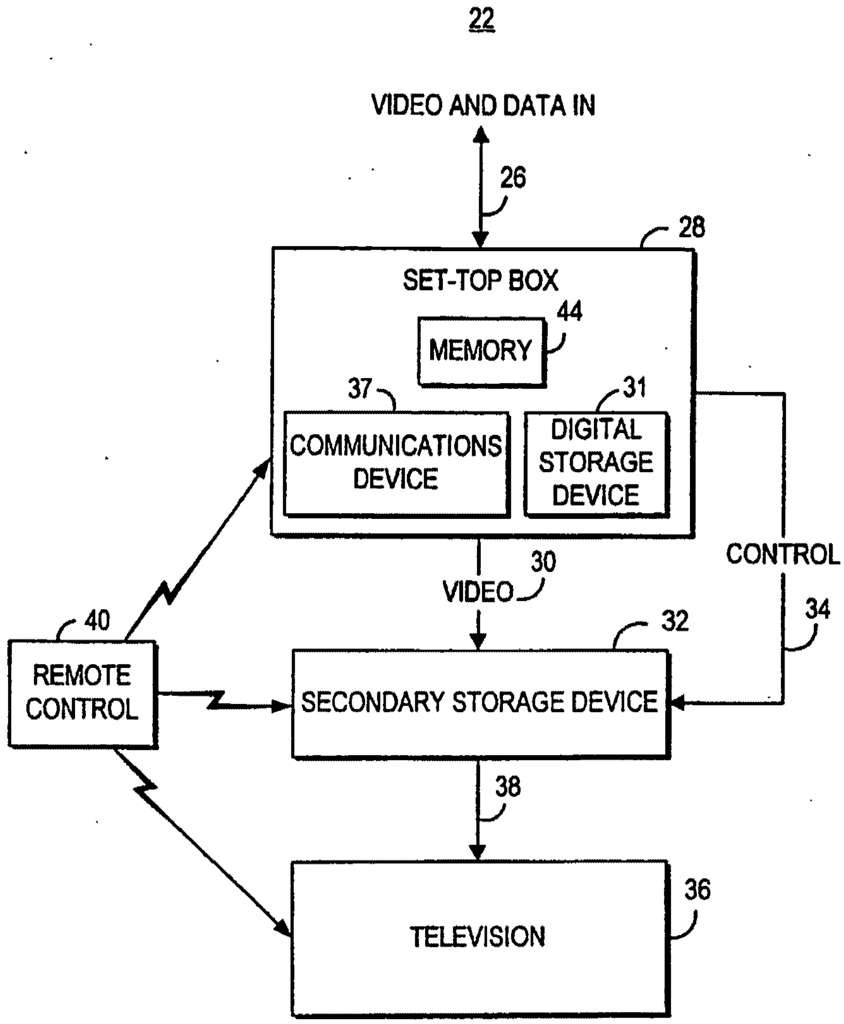

22
VIDEO AND DATA IN
26
28
SET-TOP BOX
MEMORY
44
37
31
COMMUNICATIONS DEVICE
DIGITAL STORAGE DEVICE
CONTROL
30
VIDEO
34
40
32
REMOTE CONTROL
SECONDARY STORAGE DEVICE
38
36
TELEVISION

FIG. 3

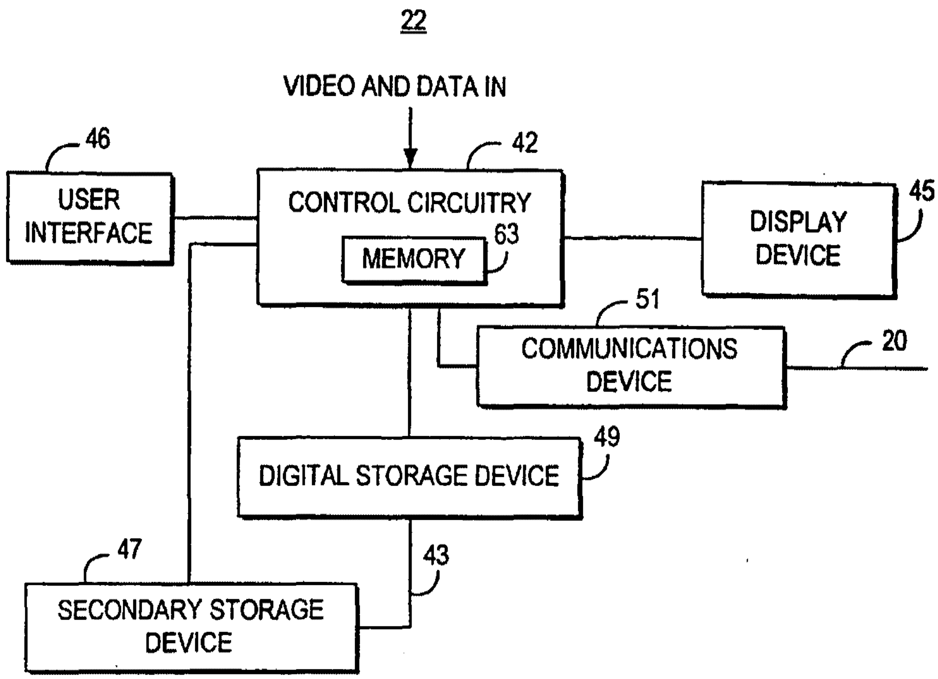

22
VIDEO AND DATA IN
46
USER INTERFACE
42
CONTROL CIRCUITRY
63
MEMORY
45
DISPLAY DEVICE
51
COMMUNICATIONS DEVICE
20
49
DIGITAL STORAGE DEVICE
47
43
SECONDARY STORAGE DEVICE

FIG. 4

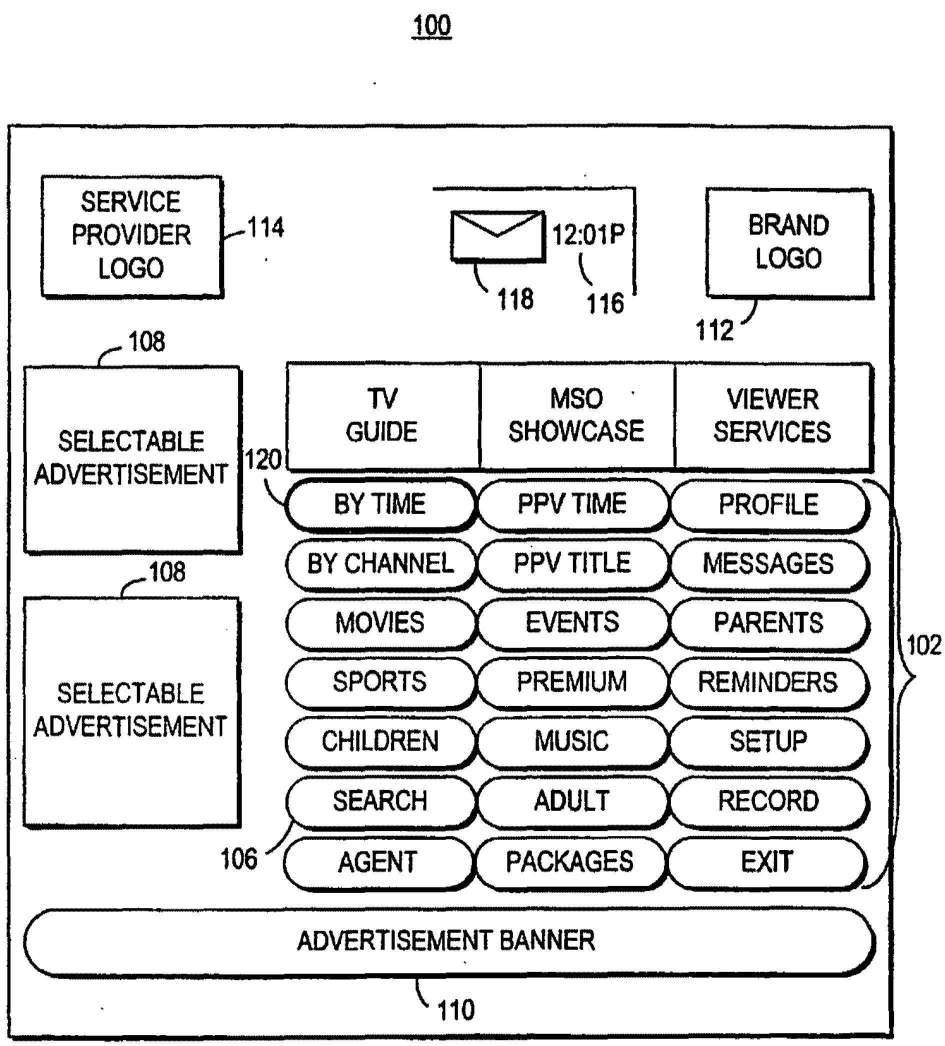

100
SERVICE PROVIDER LOGO
114
12:01P
118
116
BRAND LOGO
112
108
SELECTABLE ADVERTISEMENT
TV GUIDE
MSO SHOWCASE
VIEWER SERVICES
120
BY TIME
PPV TIME
PROFILE
BY CHANNEL
PPV TITLE
MESSAGES
108
MOVIES
EVENTS
PARENTS
SPORTS
PREMIUM
REMINDERS
102
SELECTABLE ADVERTISEMENT
CHILDREN
MUSIC
SETUP
SEARCH
ADULT
RECORD
106
AGENT
PACKAGES
EXIT
ADVERTISEMENT BANNER
110

FIG. 5

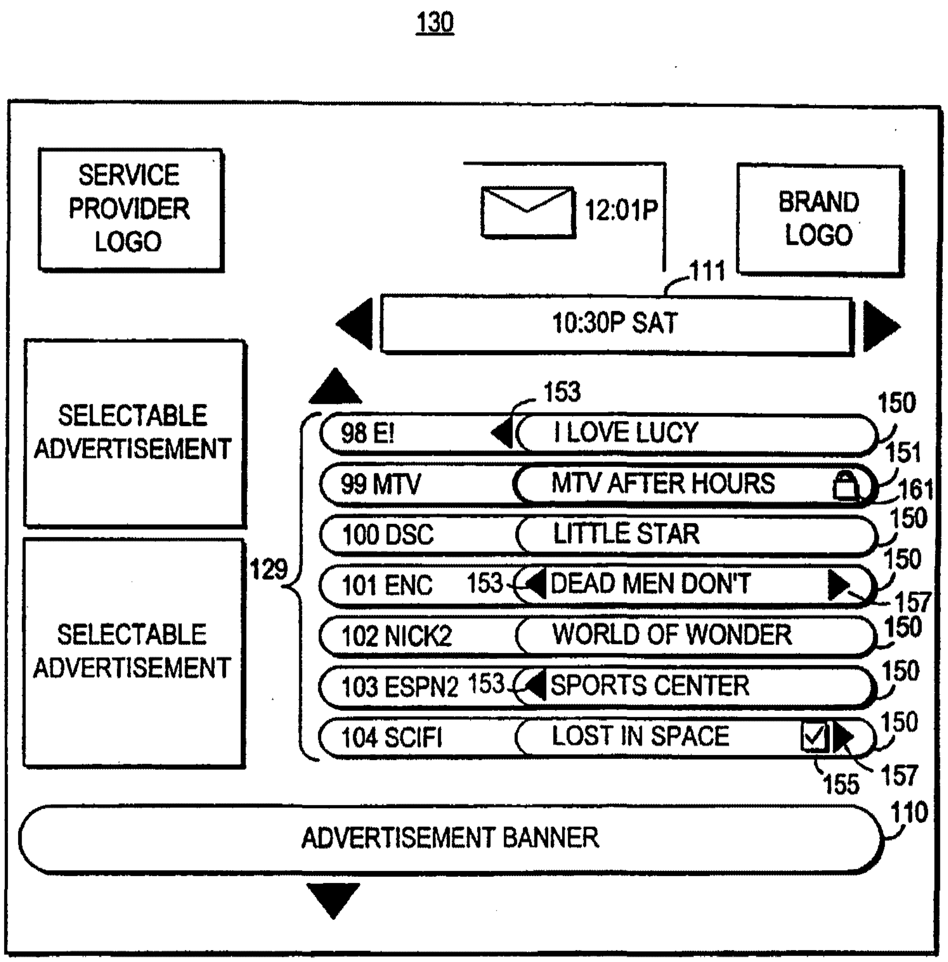

130
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
111
10:30P SAT
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
129
153
150
98 E!
I LOVE LUCY
151
99 MTV
MTV AFTER HOURS
161
100 DSC
LITTLE STAR
150
101 ENC
153
DEAD MEN DON'T
157
102 NICK2
WORLD OF WONDER
103 ESPN2
SPORTS CENTER
104 SCIFI
LOST IN SPACE
155
110
ADVERTISEMENT BANNER

FIG. 6

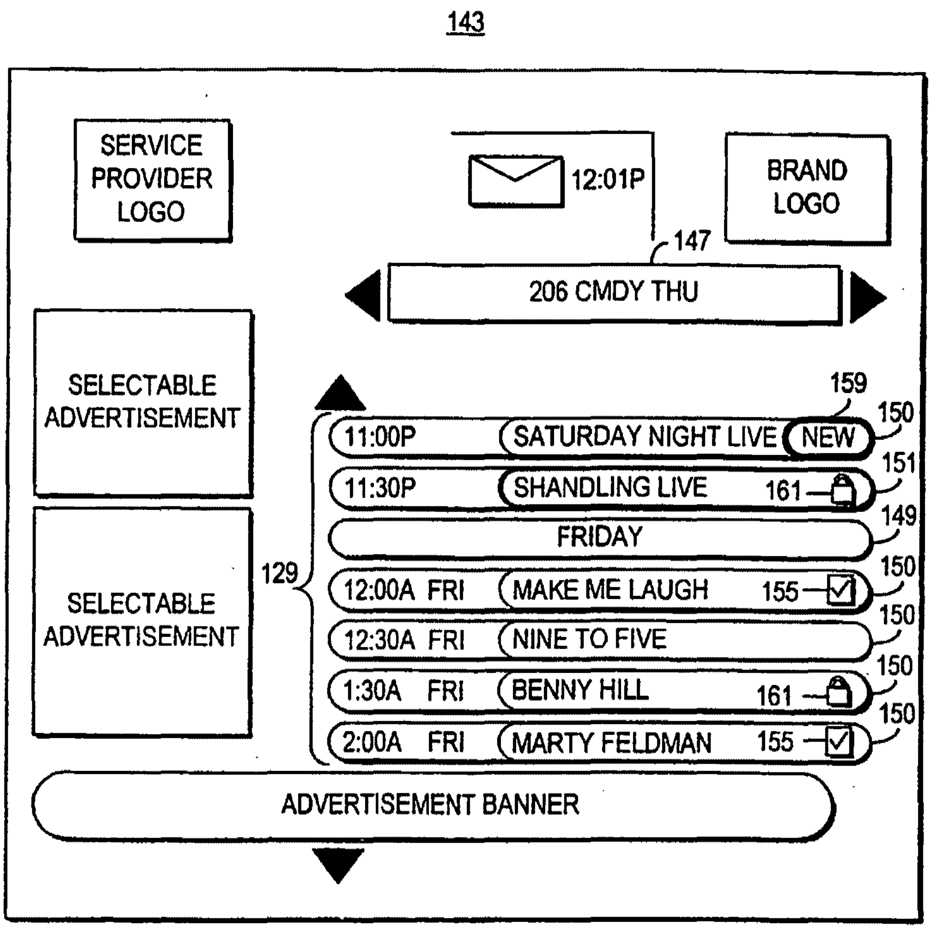

143
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
147
206 CMDY THU
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
159
150
151
149
129
150
150
150
150
11:00P
SATURDAY NIGHT LIVE
NEW
11:30P
SHANDLING LIVE
161
FRIDAY
12:00A FRI
MAKE ME LAUGH
155
12:30A FRI
NINE TO FIVE
1:30A FRI
BENNY HILL
161
2:00A FRI
MARTY FELDMAN
155
ADVERTISEMENT BANNER

FIG. 7

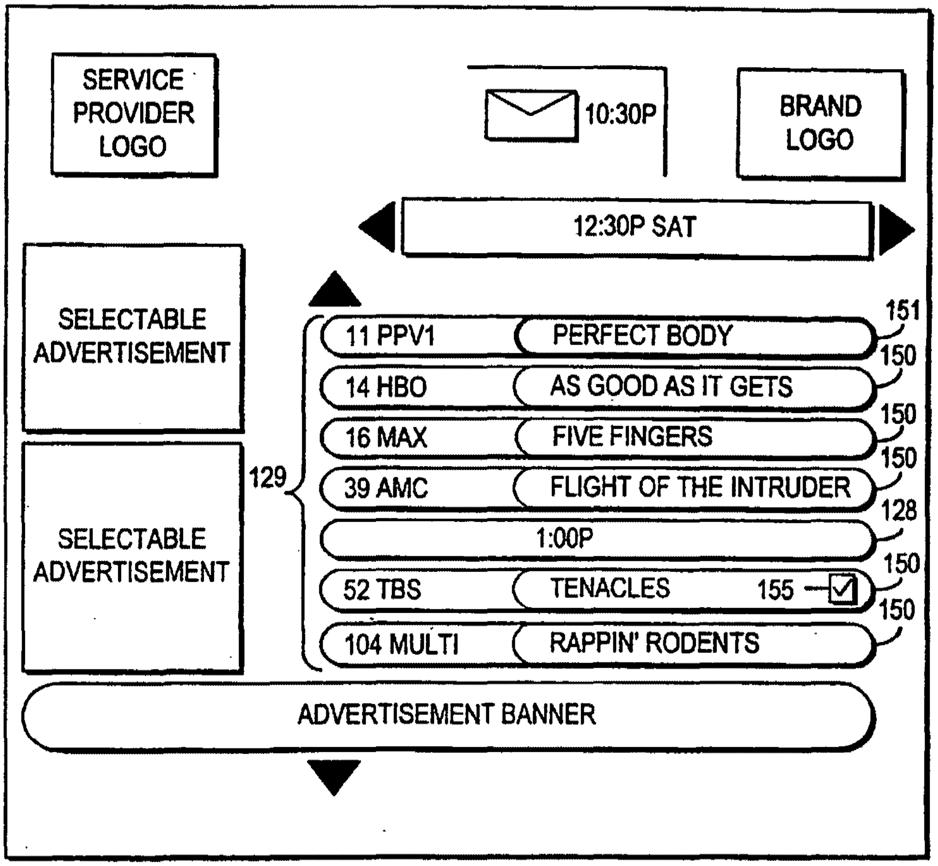

SERVICE PROVIDER LOGO
10:30P
BRAND LOGO
12:30P SAT
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
129
11 PPV1
PERFECT BODY
151
150
14 HBO
AS GOOD AS IT GETS
150
16 MAX
FIVE FINGERS
150
39 AMC
FLIGHT OF THE INTRUDER
128
1:00P
150
52 TBS
TENACLES
155
150
104 MULTI
RAPPIN' RODENTS
ADVERTISEMENT BANNER

FIG. 8a

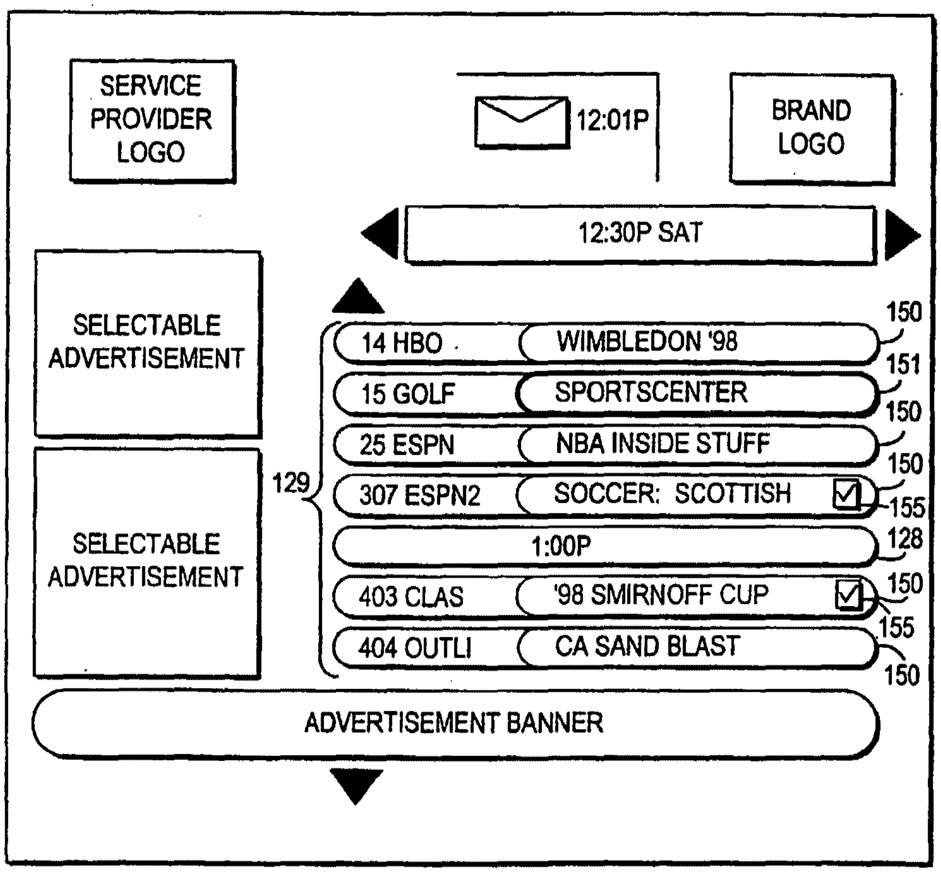

SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
12:30P SAT
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
129
14 HBO
WIMBLEDON '98
15 GOLF
SPORTSCENTER
25 ESPN
NBA INSIDE STUFF
307 ESPN2
SOCCER: SCOTTISH
1:00P
403 CLAS
'98 SMIRNOFF CUP
404 OUTLI
CA SAND BLAST
150
151
150
150
155
128
150
155
150
ADVERTISEMENT BANNER

FIG. 8b

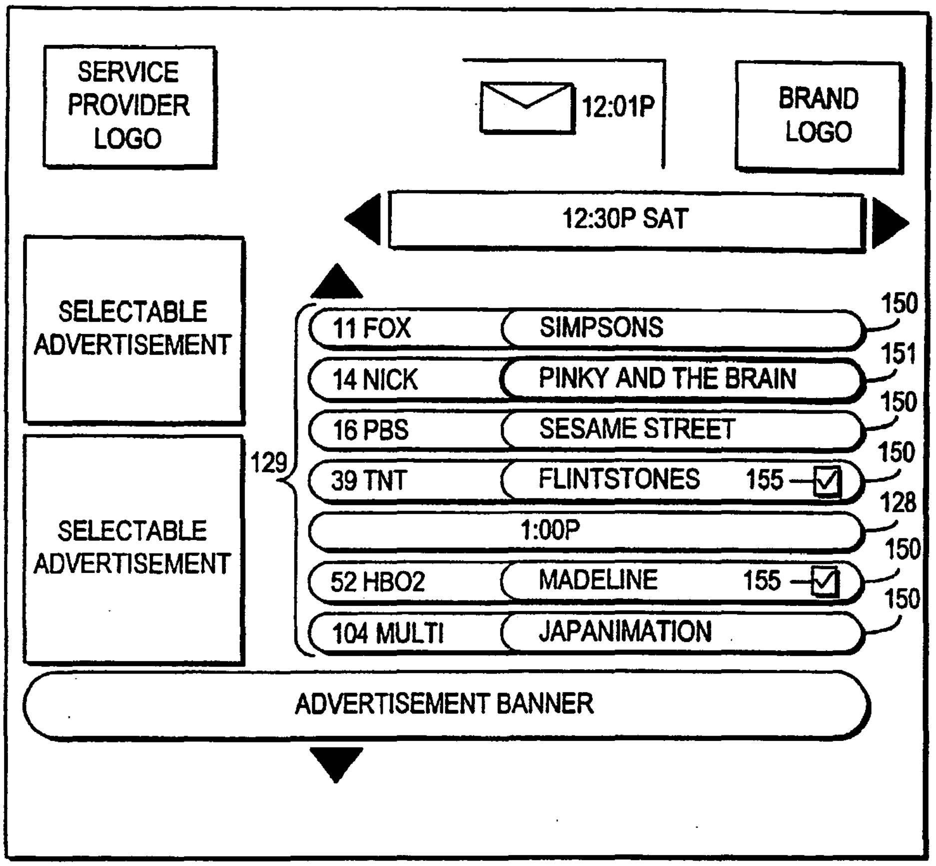

SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
12:30P SAT
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
129
11 FOX
SIMPSONS
150
14 NICK
PINKY AND THE BRAIN
151
16 PBS
SESAME STREET
150
39 TNT
FLINTSTONES
155
150
1:00P
128
52 HBO2
MADELINE
155
150
104 MULTI
JAPANIMATION
150
ADVERTISEMENT BANNER

FIG. 8c

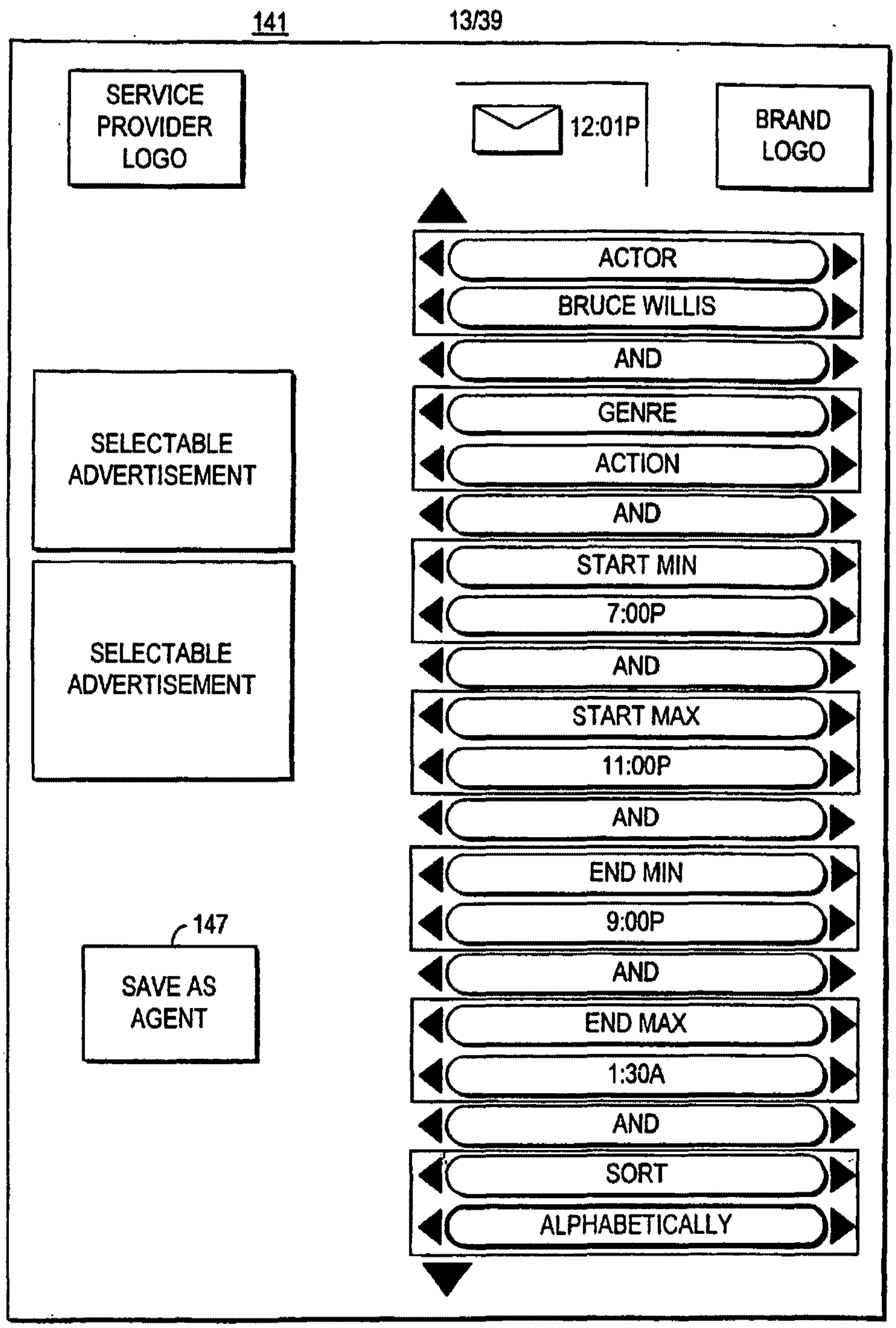

141
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
147
SAVE AS AGENT
ACTOR
BRUCE WILLIS
AND
GENRE
ACTION
AND
START MIN
7:00P
AND
START MAX
11:00P
AND
END MIN
9:00P
AND
END MAX
1:30A
AND
SORT
ALPHABETICALLY

FIG. 9a

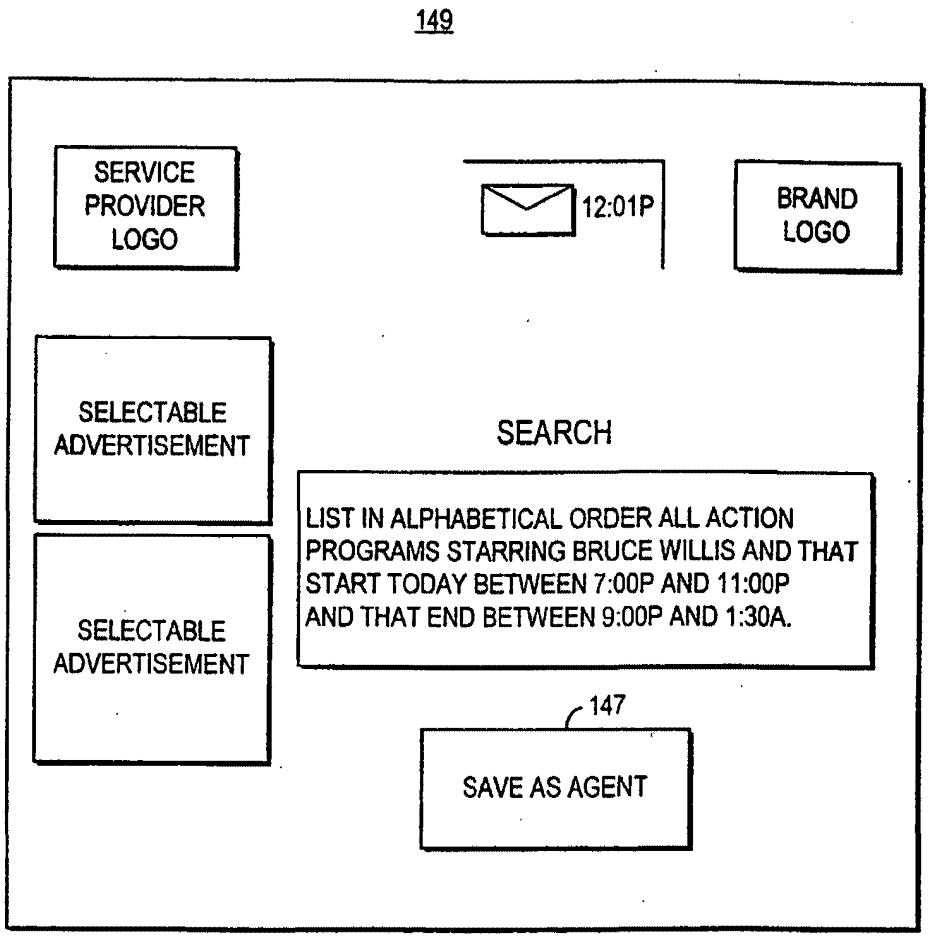

149
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
SEARCH
LIST IN ALPHABETICAL ORDER ALL ACTION PROGRAMS STARRING BRUCE WILLIS AND THAT START TODAY BETWEEN 7:00P AND 11:00P AND THAT END BETWEEN 9:00P AND 1:30A.
147
SAVE AS AGENT

FIG. 9b

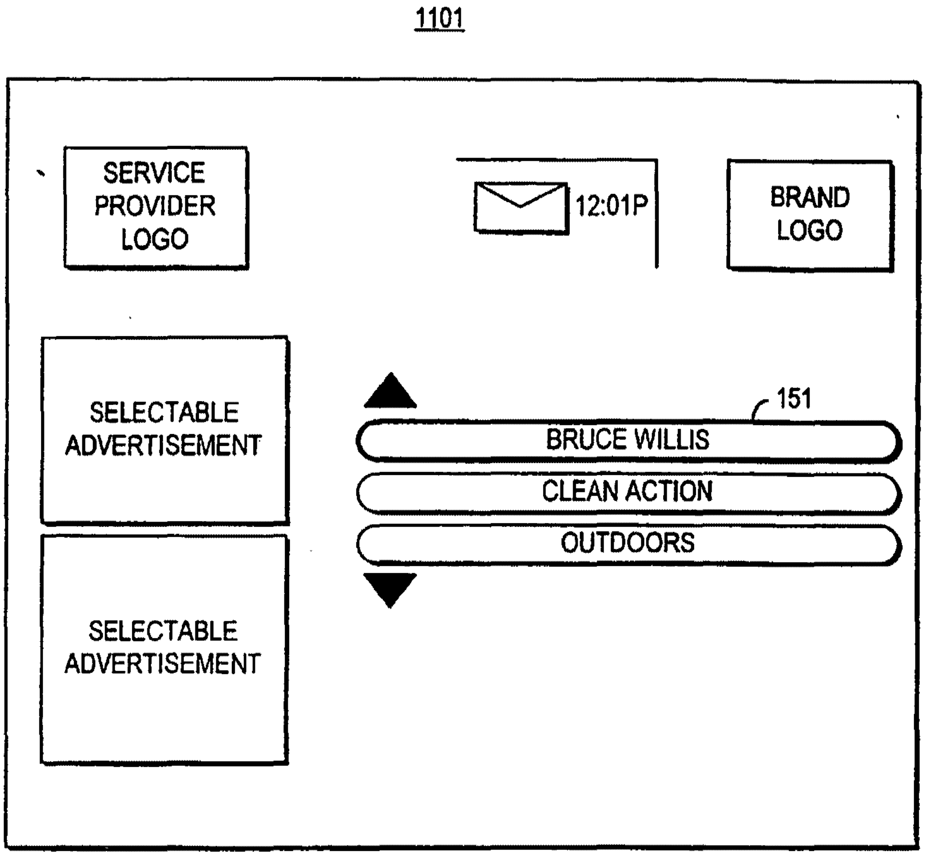

1101
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
SELECTABLE ADVERTISEMENT
151
BRUCE WILLIS
CLEAN ACTION
OUTDOORS
SELECTABLE ADVERTISEMENT

FIG. 10

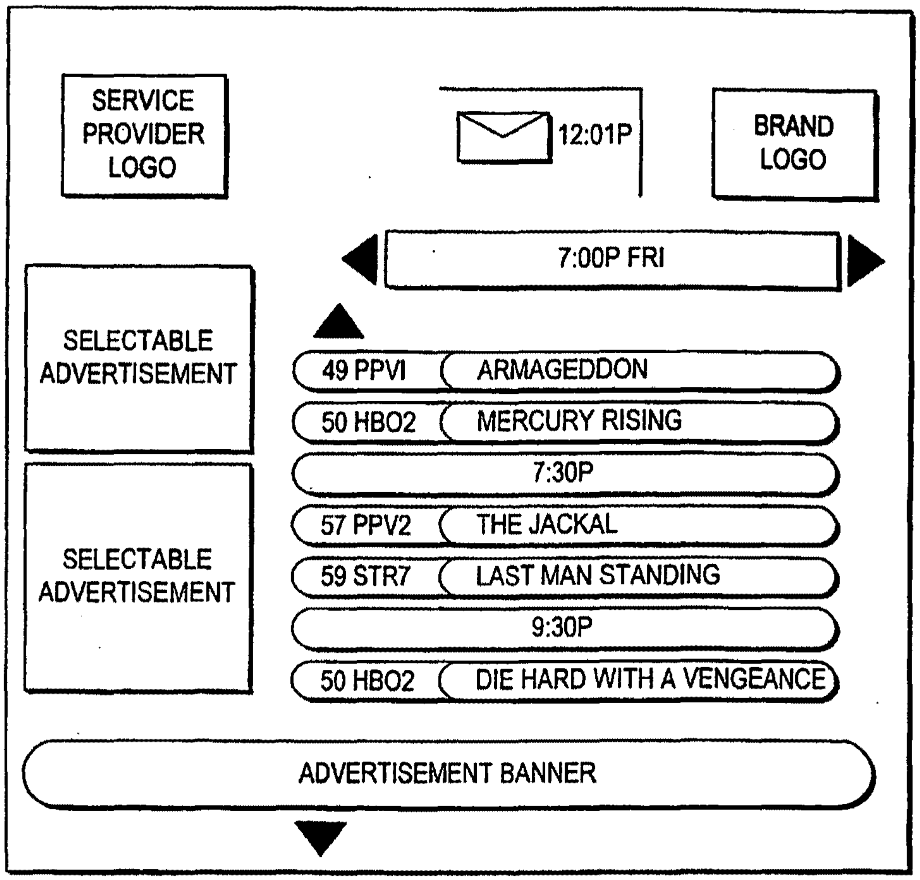

SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
7:00P FRI
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
49 PPVI
ARMAGEDDON
50 HBO2
MERCURY RISING
7:30P
57 PPV2
THE JACKAL
59 STR7
LAST MAN STANDING
9:30P
50 HBO2
DIE HARD WITH A VENGEANCE
ADVERTISEMENT BANNER

FIG. 11

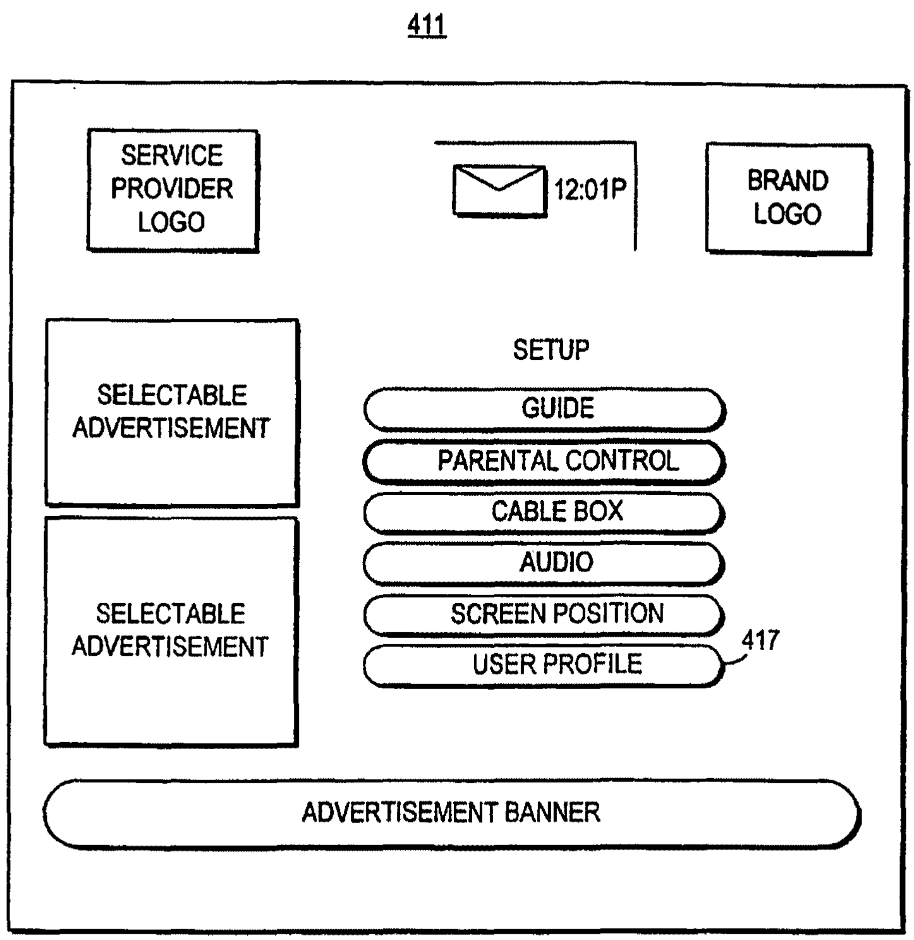

411
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
SELECTABLE ADVERTISEMENT
SETUP
GUIDE
PARENTAL CONTROL
CABLE BOX
AUDIO
SCREEN POSITION
USER PROFILE
417
SELECTABLE ADVERTISEMENT
ADVERTISEMENT BANNER

FIG. 12

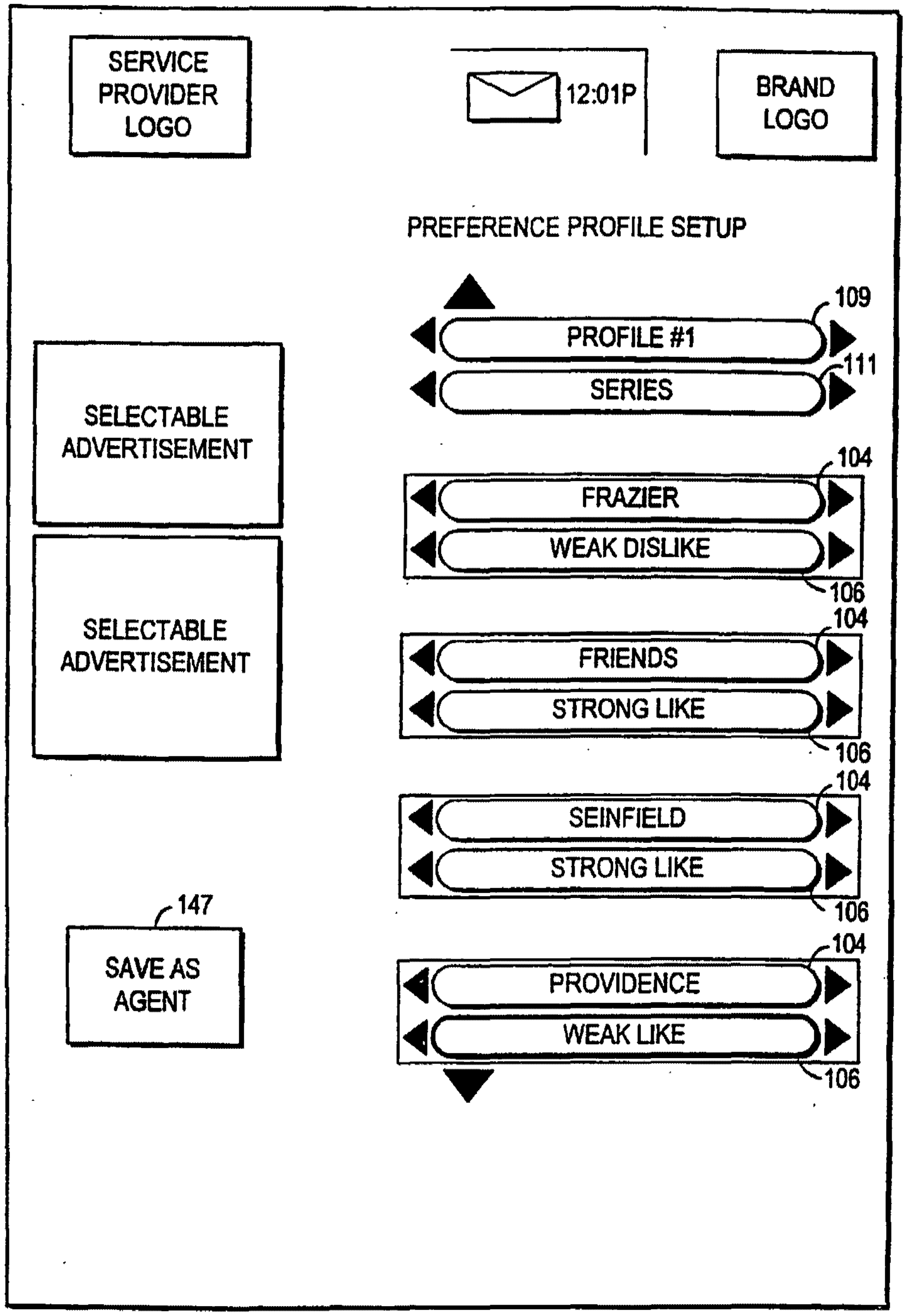


FIG. 13a

SUBSTITUTE SHEET (RULE 26)

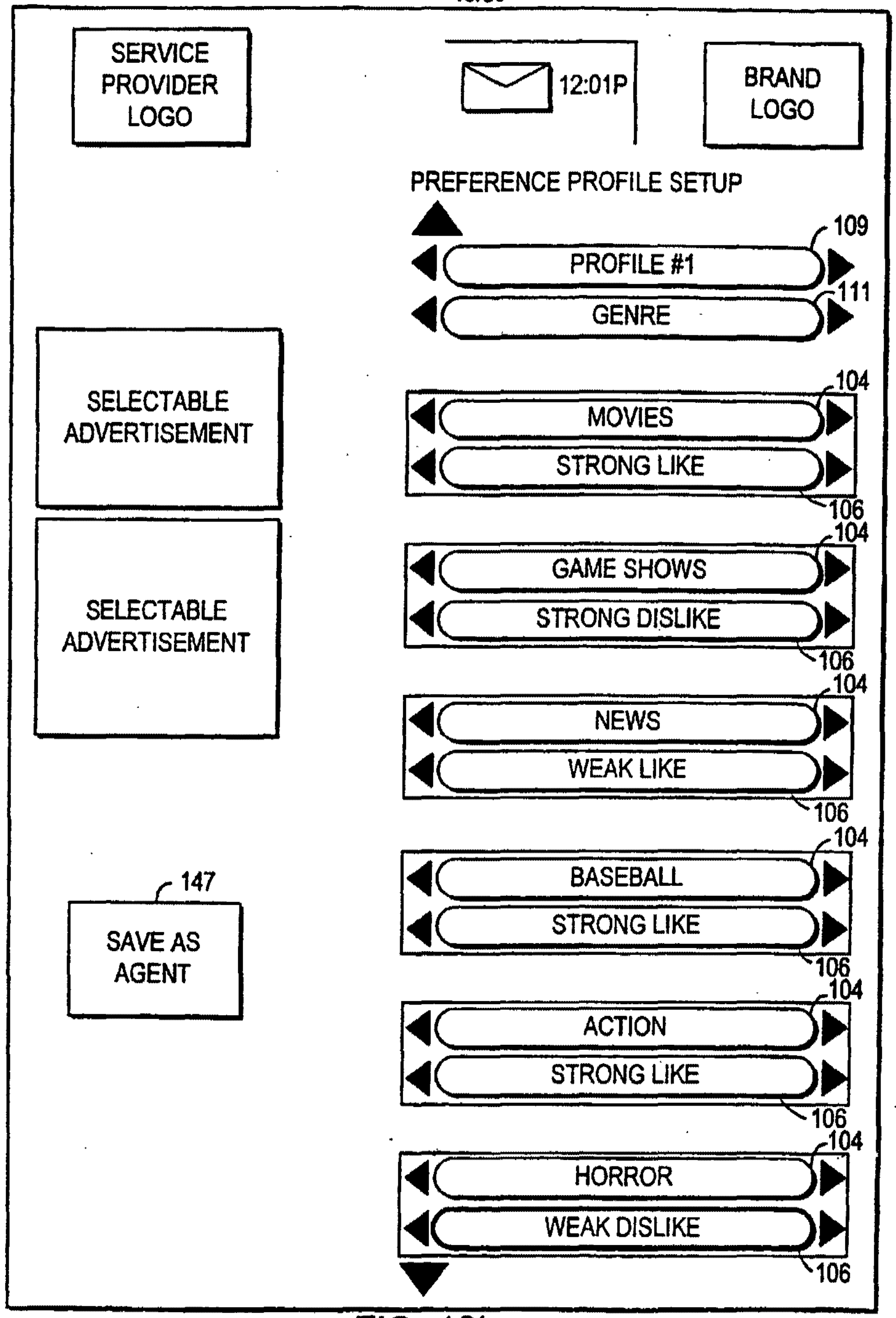

SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
PREFERENCE PROFILE SETUP
109
PROFILE #1
111
GENRE
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
104
MOVIES
STRONG LIKE
106
104
GAME SHOWS
STRONG DISLIKE
106
104
NEWS
WEAK LIKE
106
147
SAVE AS AGENT
104
BASEBALL
STRONG LIKE
106
104
ACTION
STRONG LIKE
106
104
HORROR
WEAK DISLIKE
106

FIG. 13b

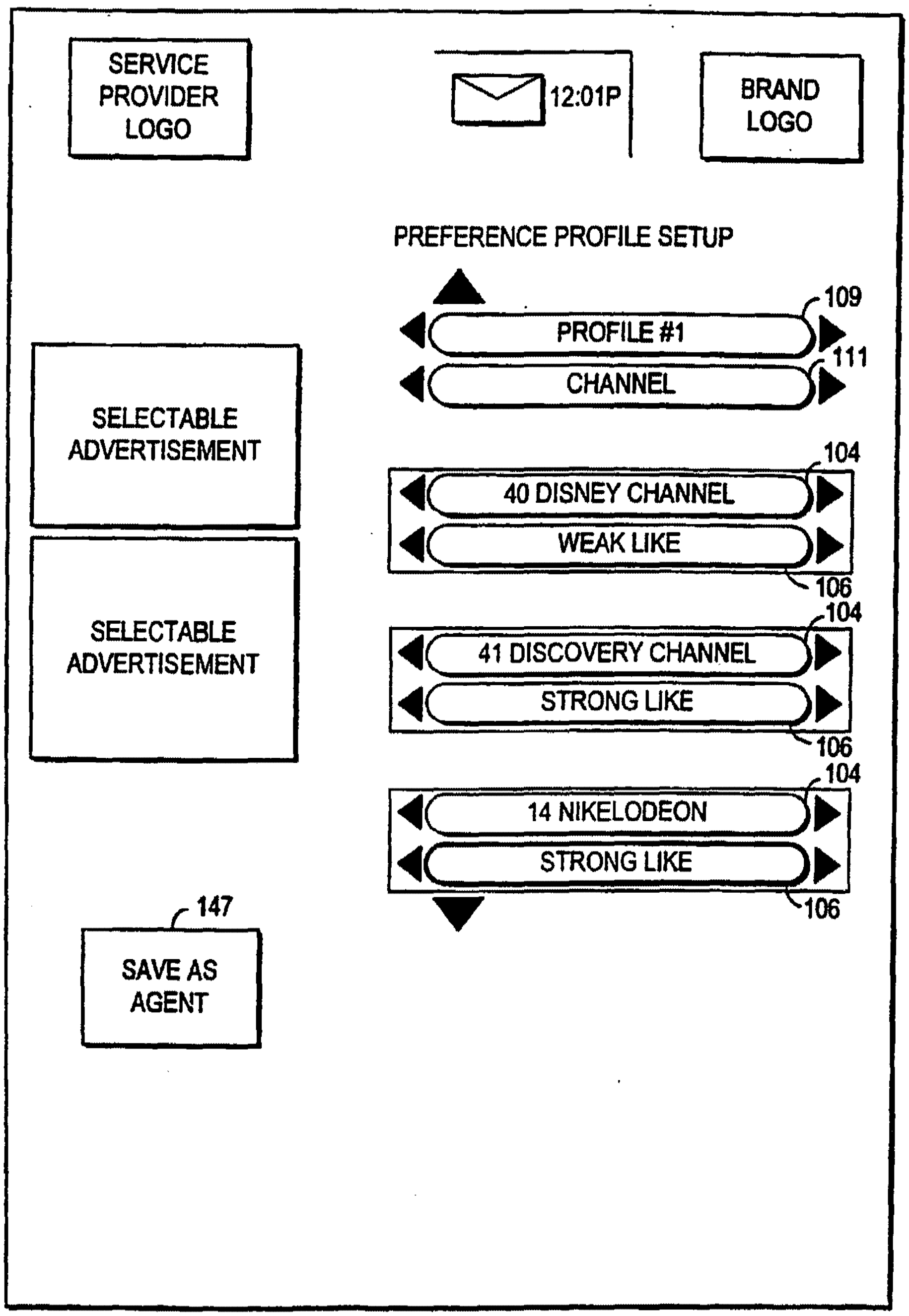


FIG. 13c

SUBSTITUTE SHEET (RULE 26)

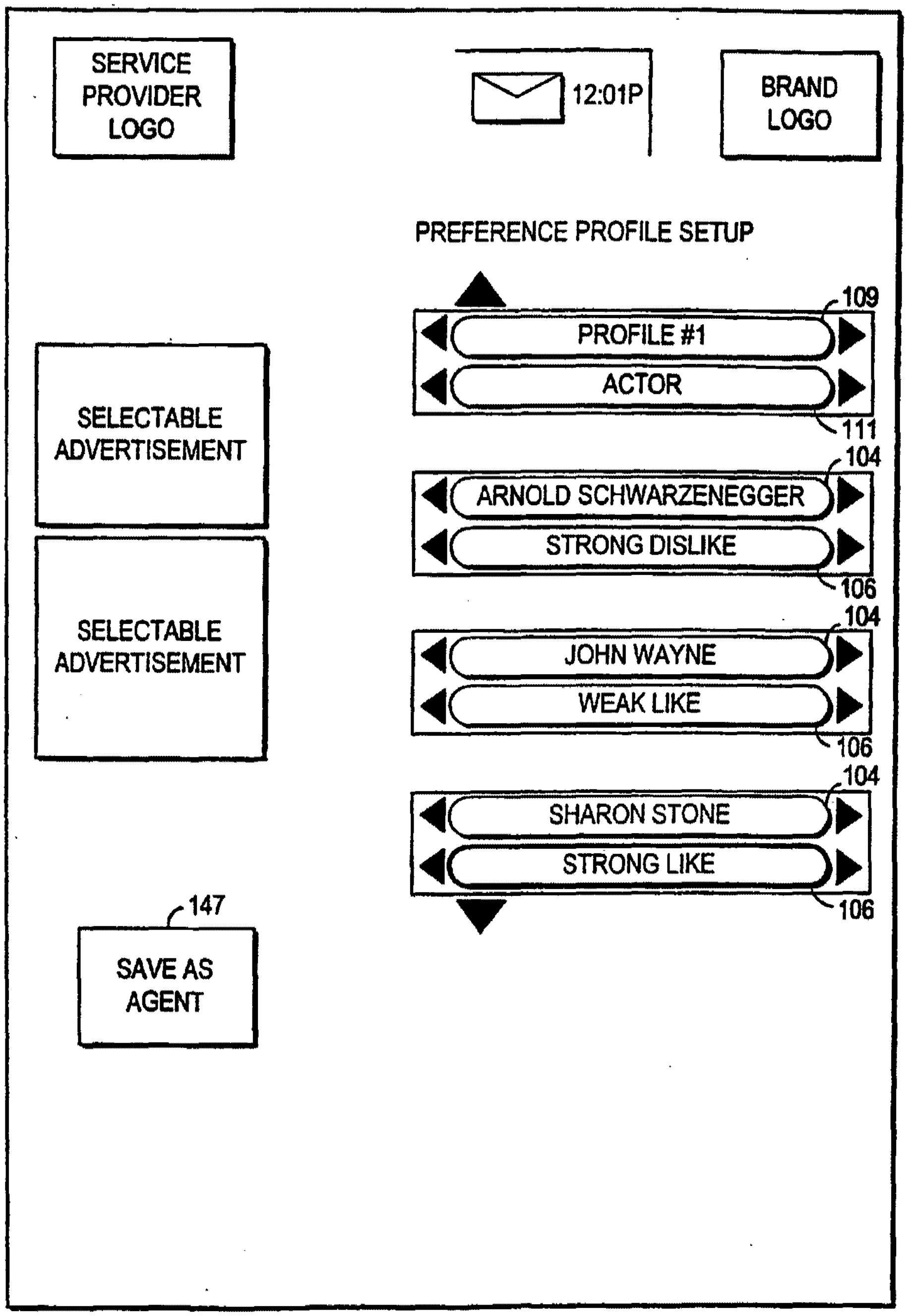

SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
PREFERENCE PROFILE SETUP
109
PROFILE #1
ACTOR
111
SELECTABLE ADVERTISEMENT
104
ARNOLD SCHWARZENEGGER
STRONG DISLIKE
106
SELECTABLE ADVERTISEMENT
104
JOHN WAYNE
WEAK LIKE
106
104
SHARON STONE
STRONG LIKE
106
147
SAVE AS AGENT

FIG. 13d

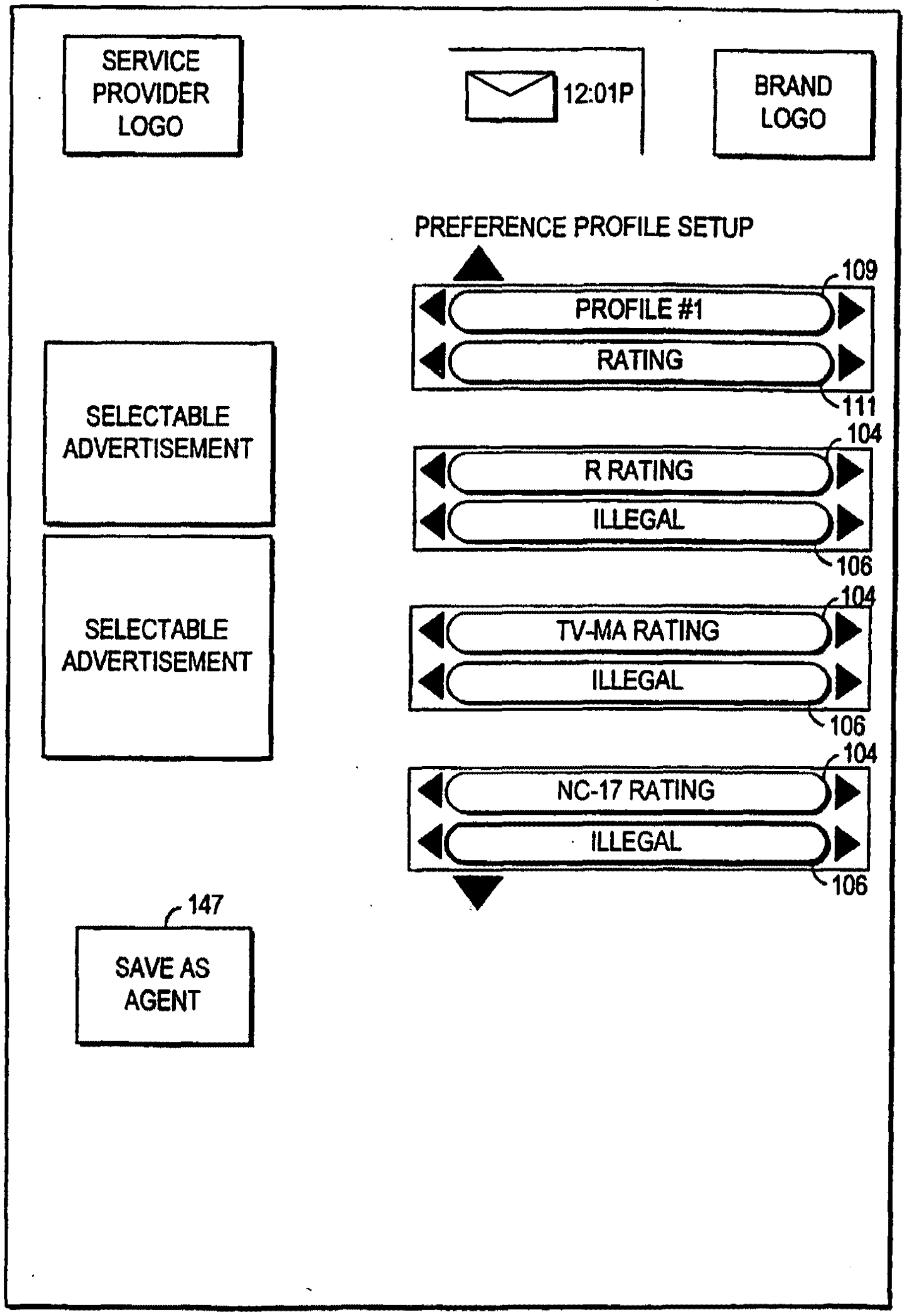

SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
PREFERENCE PROFILE SETUP
109
PROFILE #1
RATING
111
SELECTABLE ADVERTISEMENT
104
R RATING
ILLEGAL
106
SELECTABLE ADVERTISEMENT
104
TV-MA RATING
ILLEGAL
106
104
NC-17 RATING
ILLEGAL
106
147
SAVE AS AGENT

FIG. 13e

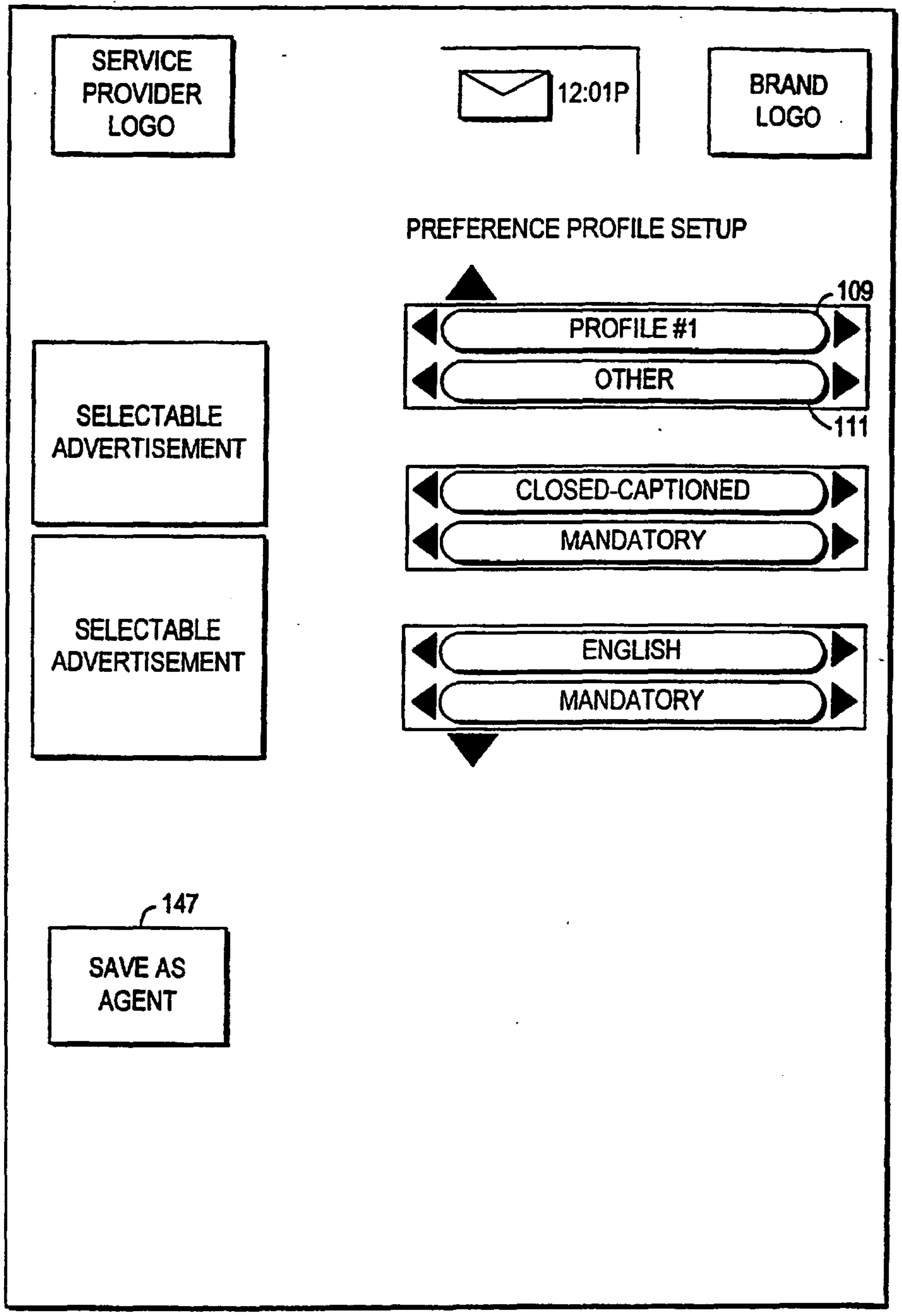

SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
PREFERENCE PROFILE SETUP
109
PROFILE #1
OTHER
111
SELECTABLE ADVERTISEMENT
CLOSED-CAPTIONED
MANDATORY
SELECTABLE ADVERTISEMENT
ENGLISH
MANDATORY
147
SAVE AS AGENT

FIG. 13f

130

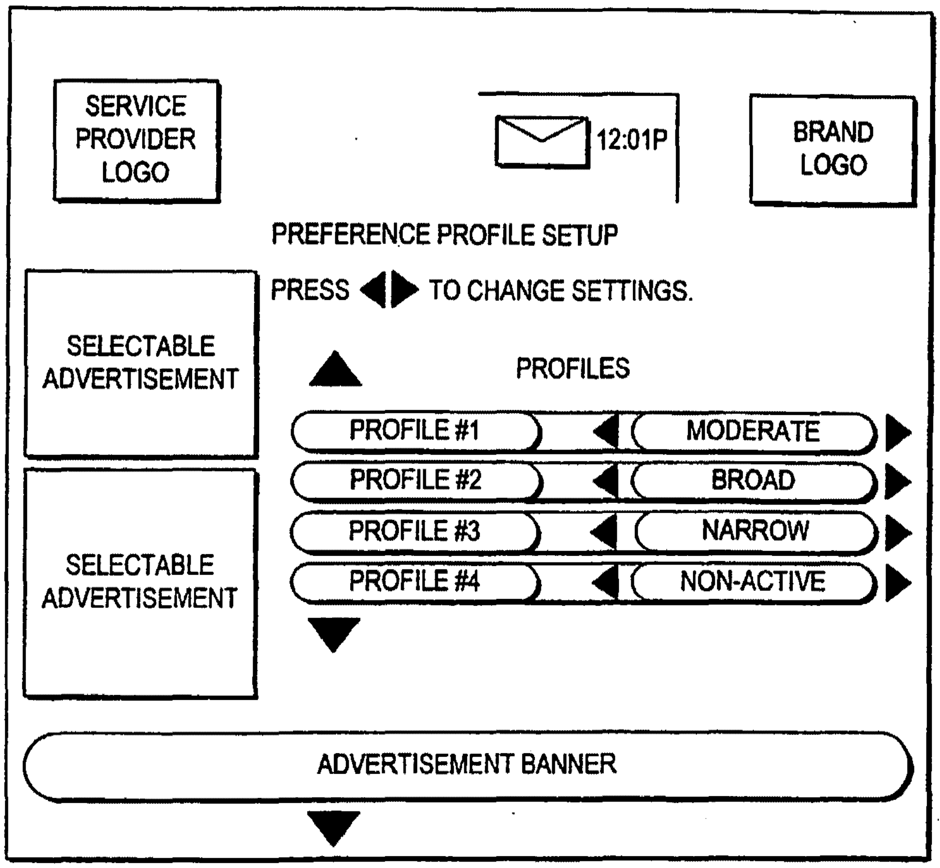

SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
PREFERENCE PROFILE SETUP
PRESS ◀▶ TO CHANGE SETTINGS.
SELECTABLE ADVERTISEMENT
PROFILES
PROFILE #1
MODERATE
PROFILE #2
BROAD
PROFILE #3
NARROW
PROFILE #4
NON-ACTIVE
SELECTABLE ADVERTISEMENT
ADVERTISEMENT BANNER

FIG. 14

NARROW SCOPE	MODERATE SCOPE	WIDE SCOPE	TITLE	GENRE	CC	RATING	MANDATORY+ NOT ILLEGAL	HIGHEST LEVEL
Y	Y	Y	SEINFELD	COMEDY	Y	TV-PG	Y	SL
N	N	Y	THE SHINING	HORROR	Y	PG-13	Y	WD
N	N	N	DANTE'S PEAK	COMEDY	Y	R	N	SL
N	N	N	NIGHT AT THE OPERA	COMEDY	N	G	N	SL
N	Y	Y	ER	DRAMA	Y	TV-PG	Y	NEUTRAL
N	N	Y	TERMINATOR	ACTION HORROR	Y	PG-13	Y	SD
N	Y	Y	MY STEPMOTHER IS AN ALIEN	COMEDY HORROR	Y	PG-13	Y	SL+WD

FIG. 15

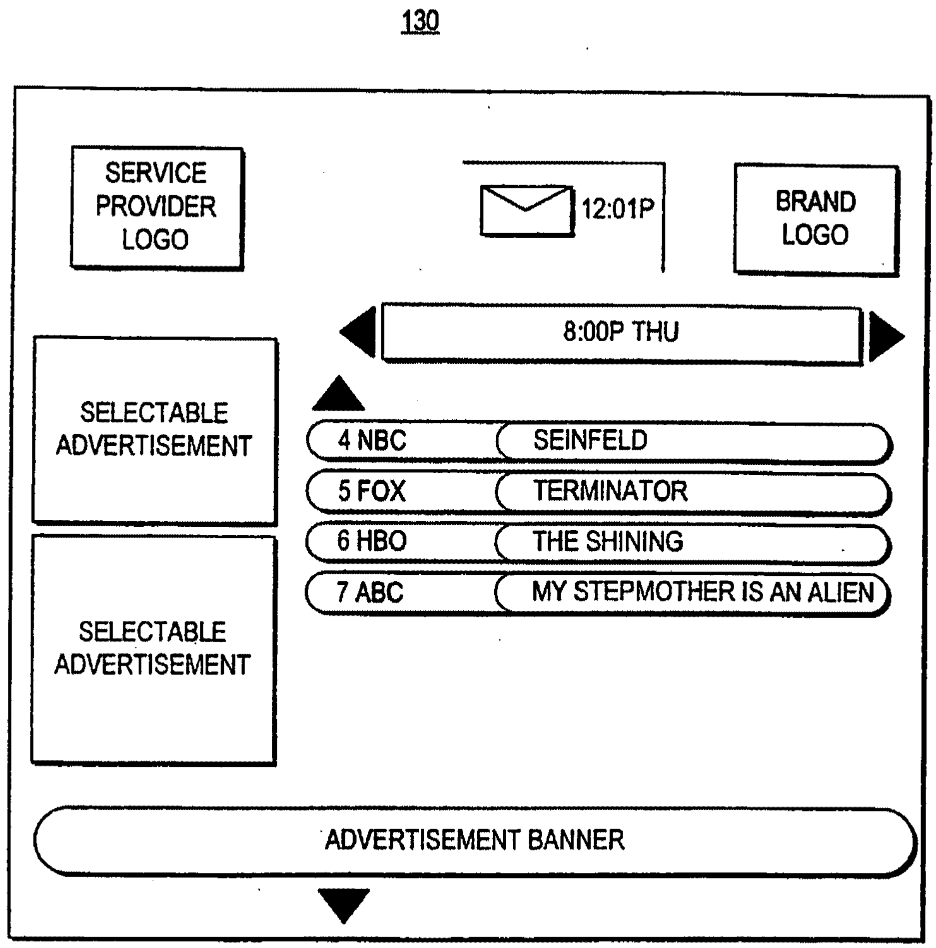

130
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
8:00P THU
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
4 NBC
SEINFELD
5 FOX
TERMINATOR
6 HBO
THE SHINING
7 ABC
MY STEPMOTHER IS AN ALIEN
ADVERTISEMENT BANNER

FIG. 16a

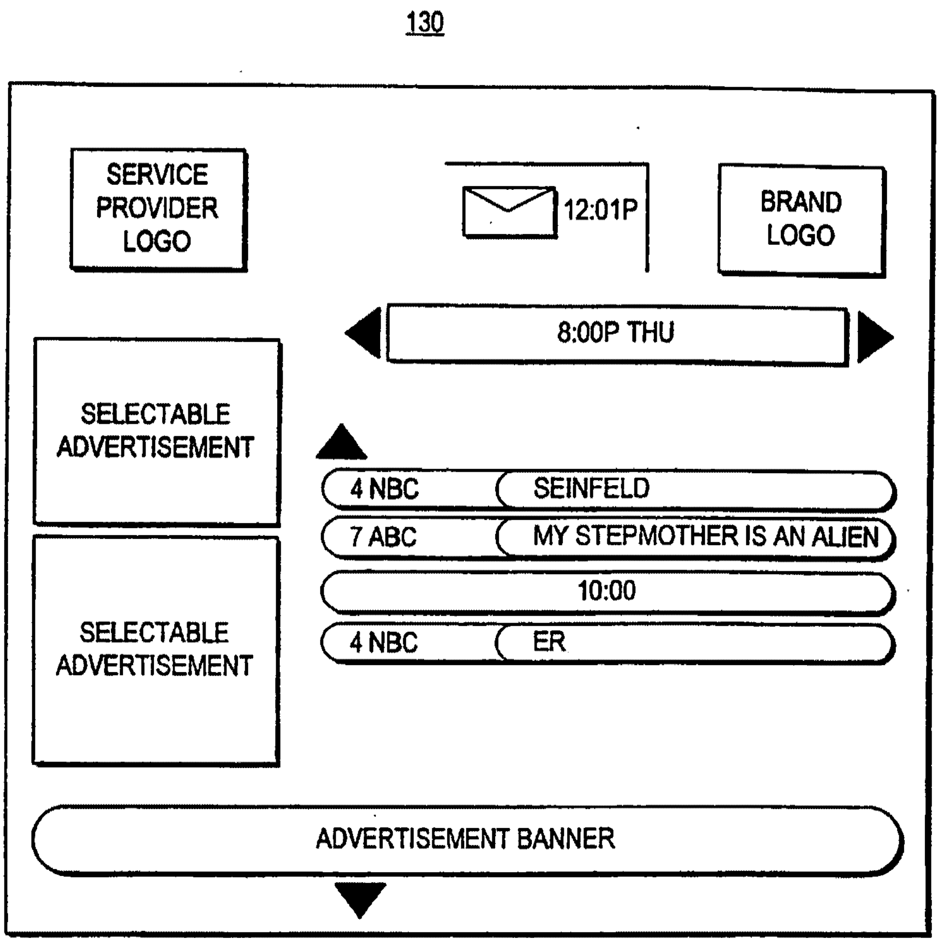

130
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
8:00P THU
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
4 NBC
SEINFELD
7 ABC
MY STEPMOTHER IS AN ALIEN
10:00
4 NBC
ER
ADVERTISEMENT BANNER

FIG. 16b

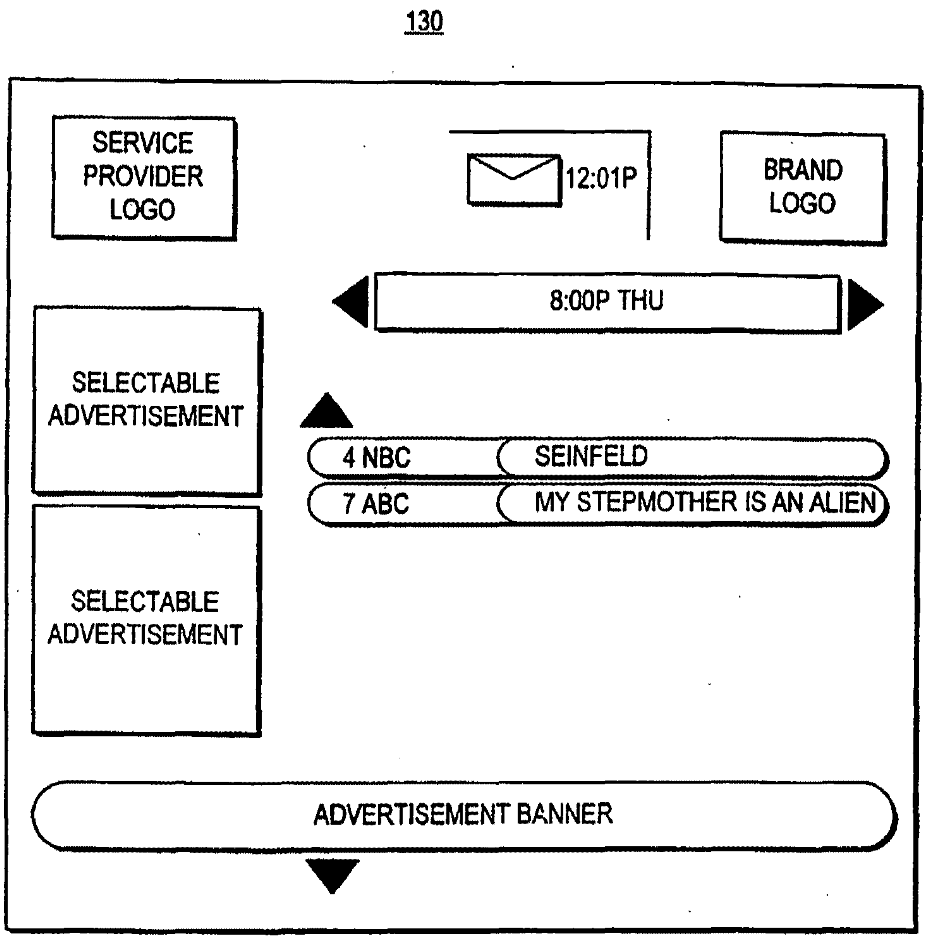

130
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
8:00P THU
SELECTABLE ADVERTISEMENT
4 NBC
SEINFELD
7 ABC
MY STEPMOTHER IS AN ALIEN
SELECTABLE ADVERTISEMENT
ADVERTISEMENT BANNER

FIG. 16c

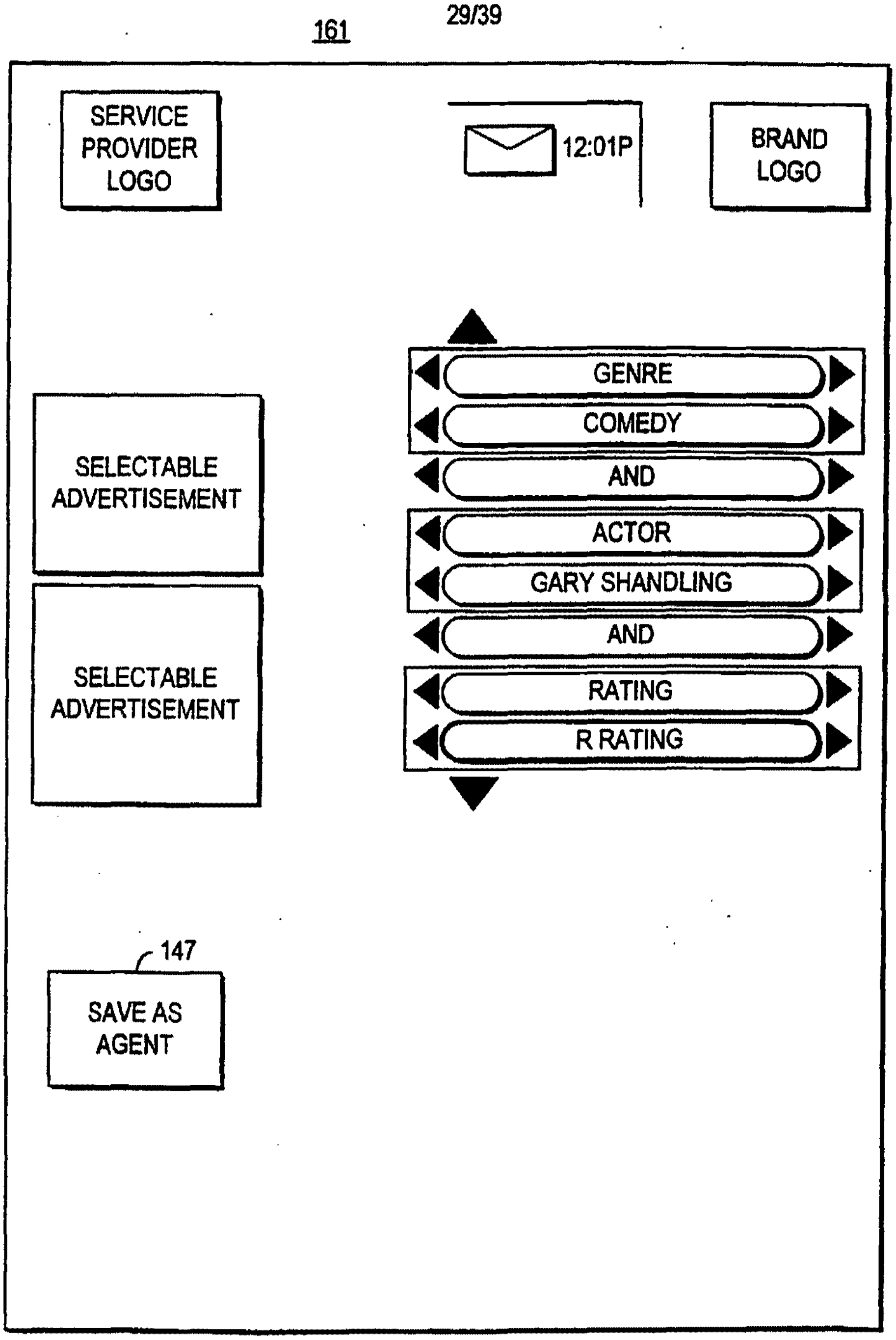

161
SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
GENRE
COMEDY
AND
ACTOR
GARY SHANDLING
AND
RATING
R RATING
147
SAVE AS AGENT

FIG. 17a

169

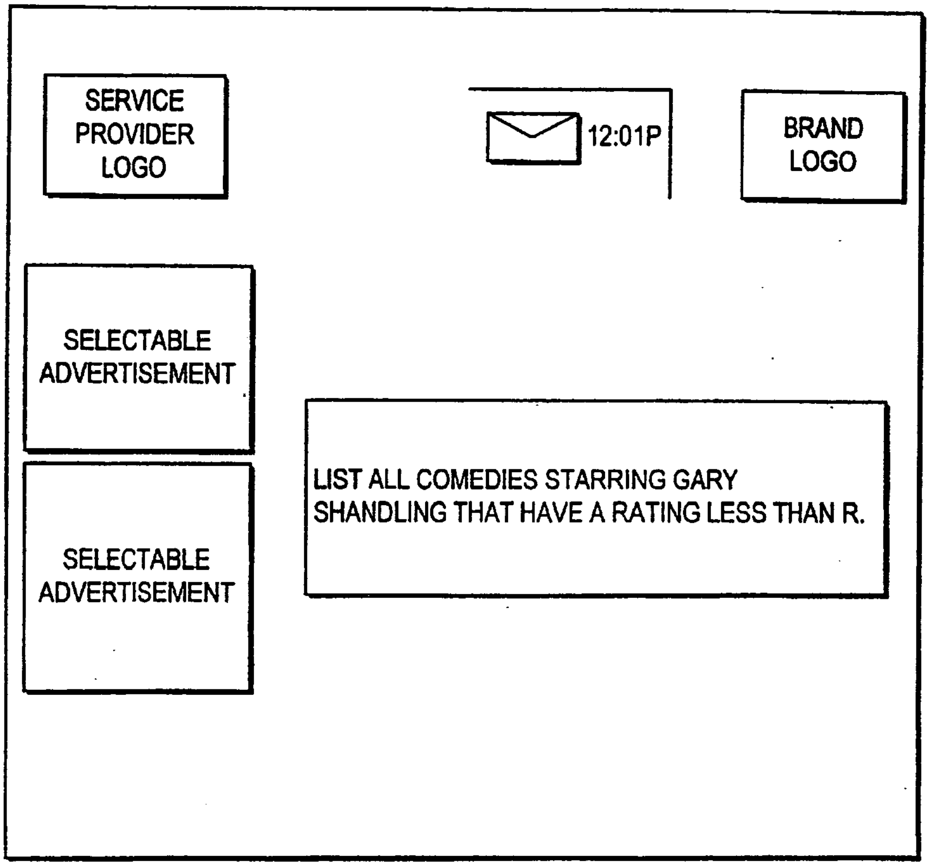

SERVICE PROVIDER LOGO
12:01P
BRAND LOGO
SELECTABLE ADVERTISEMENT
SELECTABLE ADVERTISEMENT
LIST ALL COMEDIES STARRING GARY SHANDLING THAT HAVE A RATING LESS THAN R.

FIG. 17b

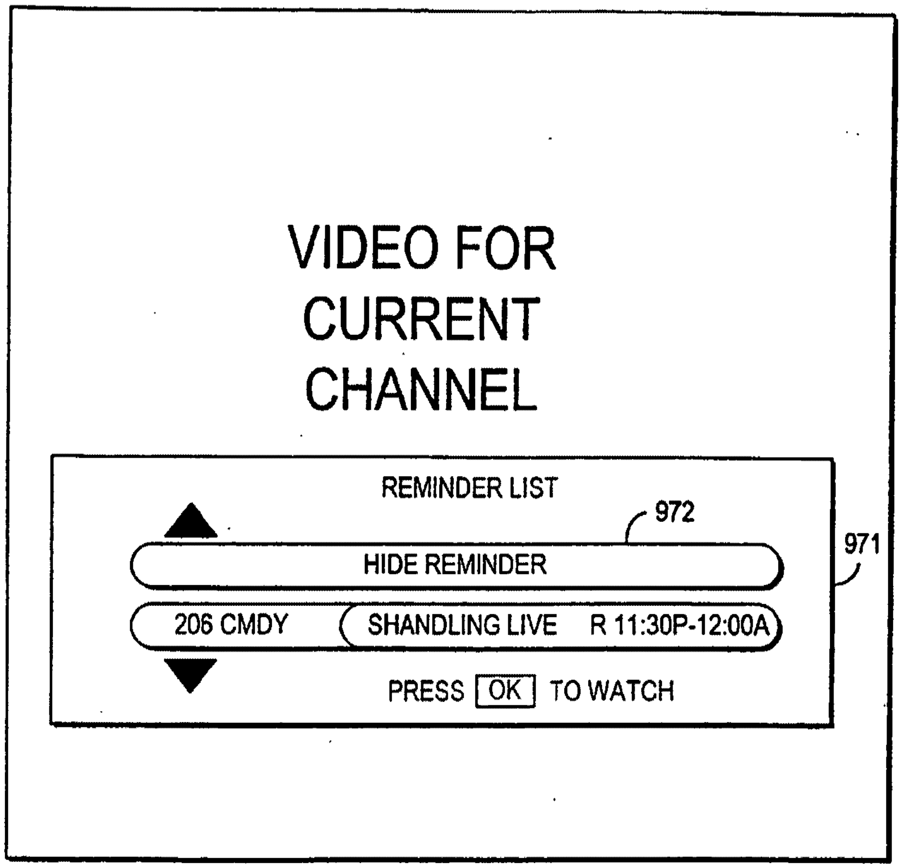

VIDEO FOR
CURRENT
CHANNEL
REMINDER LIST
972
971
HIDE REMINDER
206 CMDY
SHANDLING LIVE R 11:30P-12:00A
PRESS OK TO WATCH

FIG. 18

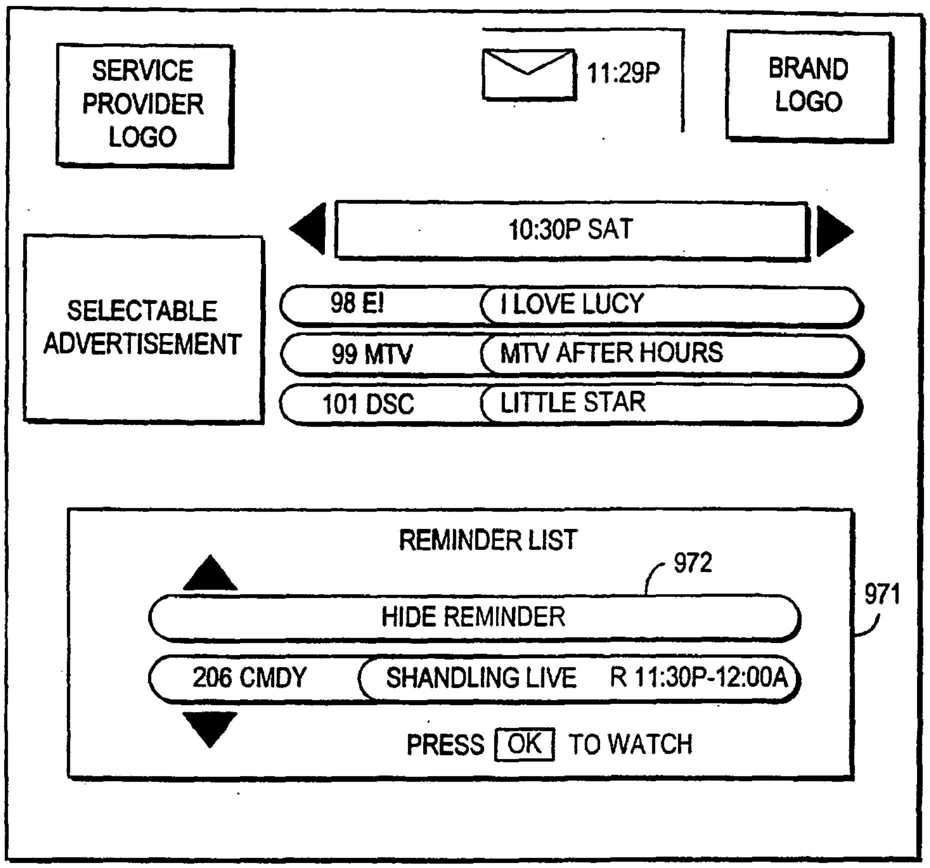

SERVICE PROVIDER LOGO
11:29P
BRAND LOGO
10:30P SAT
SELECTABLE ADVERTISEMENT
98 E! I LOVE LUCY
99 MTV MTV AFTER HOURS
101 DSC LITTLE STAR
REMINDER LIST
972
971
HIDE REMINDER
206 CMDY SHANDLING LIVE R 11:30P-12:00A
PRESS OK TO WATCH

FIG. 19

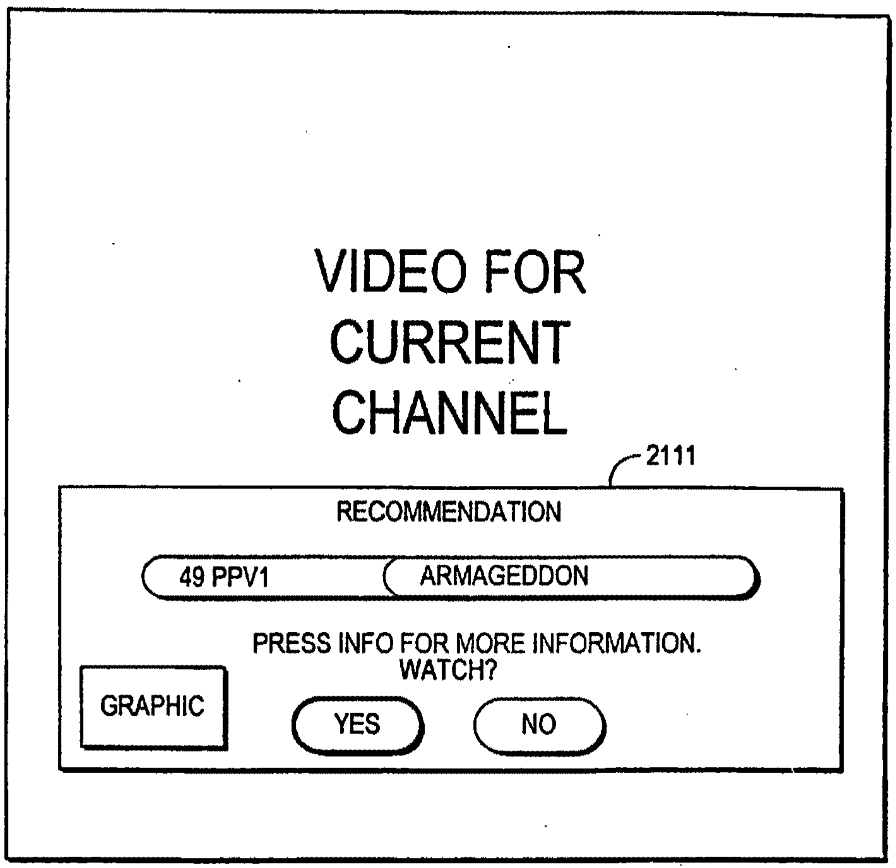

VIDEO FOR
CURRENT
CHANNEL
2111
RECOMMENDATION
49 PPV1
ARMAGEDDON
PRESS INFO FOR MORE INFORMATION.
WATCH?
GRAPHIC
YES
NO

FIG. 20a

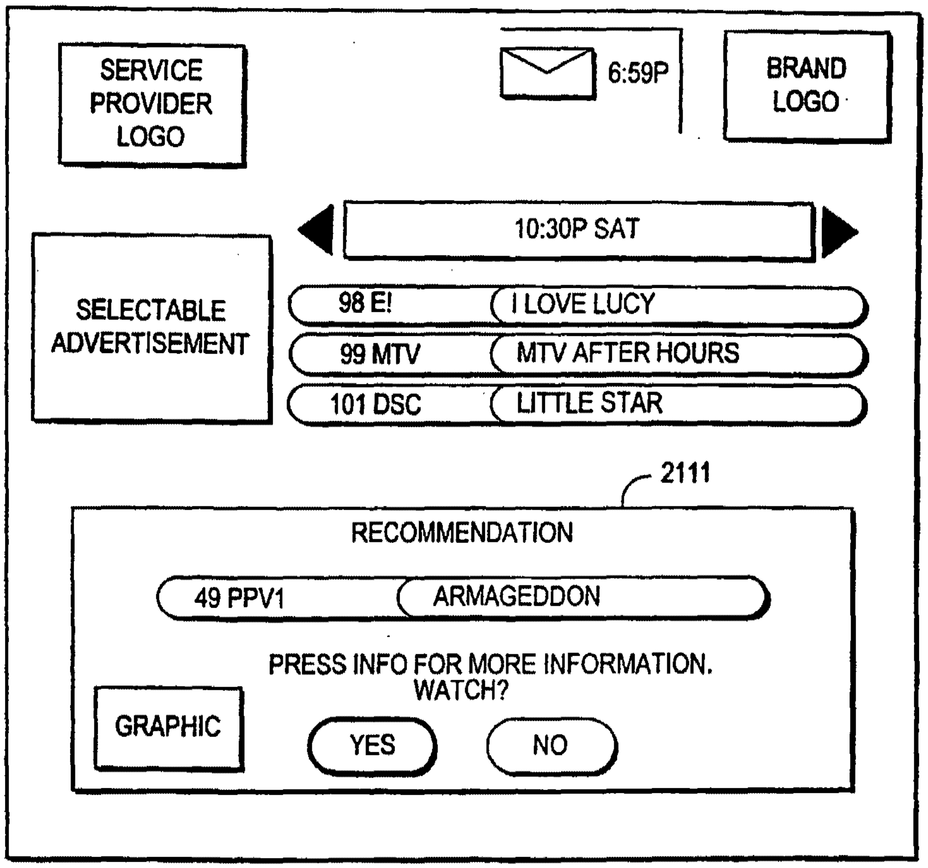

SERVICE PROVIDER LOGO
6:59P
BRAND LOGO
10:30P SAT
SELECTABLE ADVERTISEMENT
98 E!
I LOVE LUCY
99 MTV
MTV AFTER HOURS
101 DSC
LITTLE STAR
2111
RECOMMENDATION
49 PPV1
ARMAGEDDON
PRESS INFO FOR MORE INFORMATION.
WATCH?
GRAPHIC
YES
NO

FIG. 20b

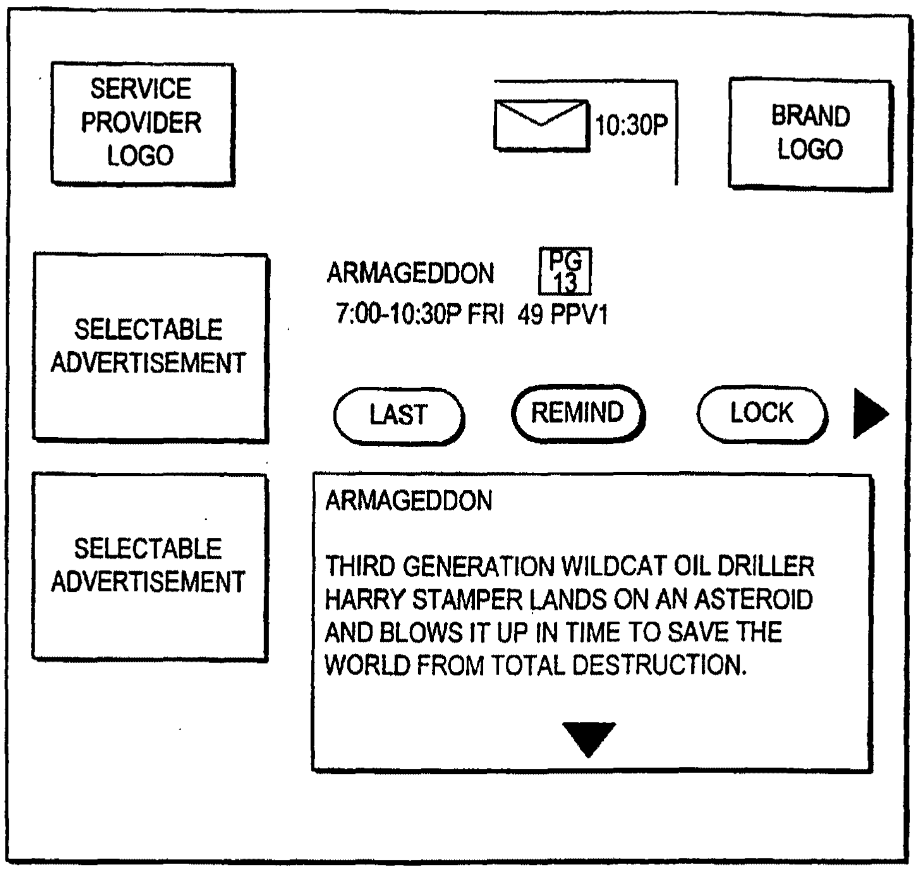

SERVICE PROVIDER LOGO
10:30P
BRAND LOGO
SELECTABLE ADVERTISEMENT
ARMAGEDDON
PG 13
7:00-10:30P FRI 49 PPV1
LAST
REMIND
LOCK
SELECTABLE ADVERTISEMENT
ARMAGEDDON
THIRD GENERATION WILDCAT OIL DRILLER HARRY STAMPER LANDS ON AN ASTEROID AND BLOWS IT UP IN TIME TO SAVE THE WORLD FROM TOTAL DESTRUCTION.

FIG. 20c

WO 00/1869

SUBSTITUTE SHEET (RULE 26)

36/39

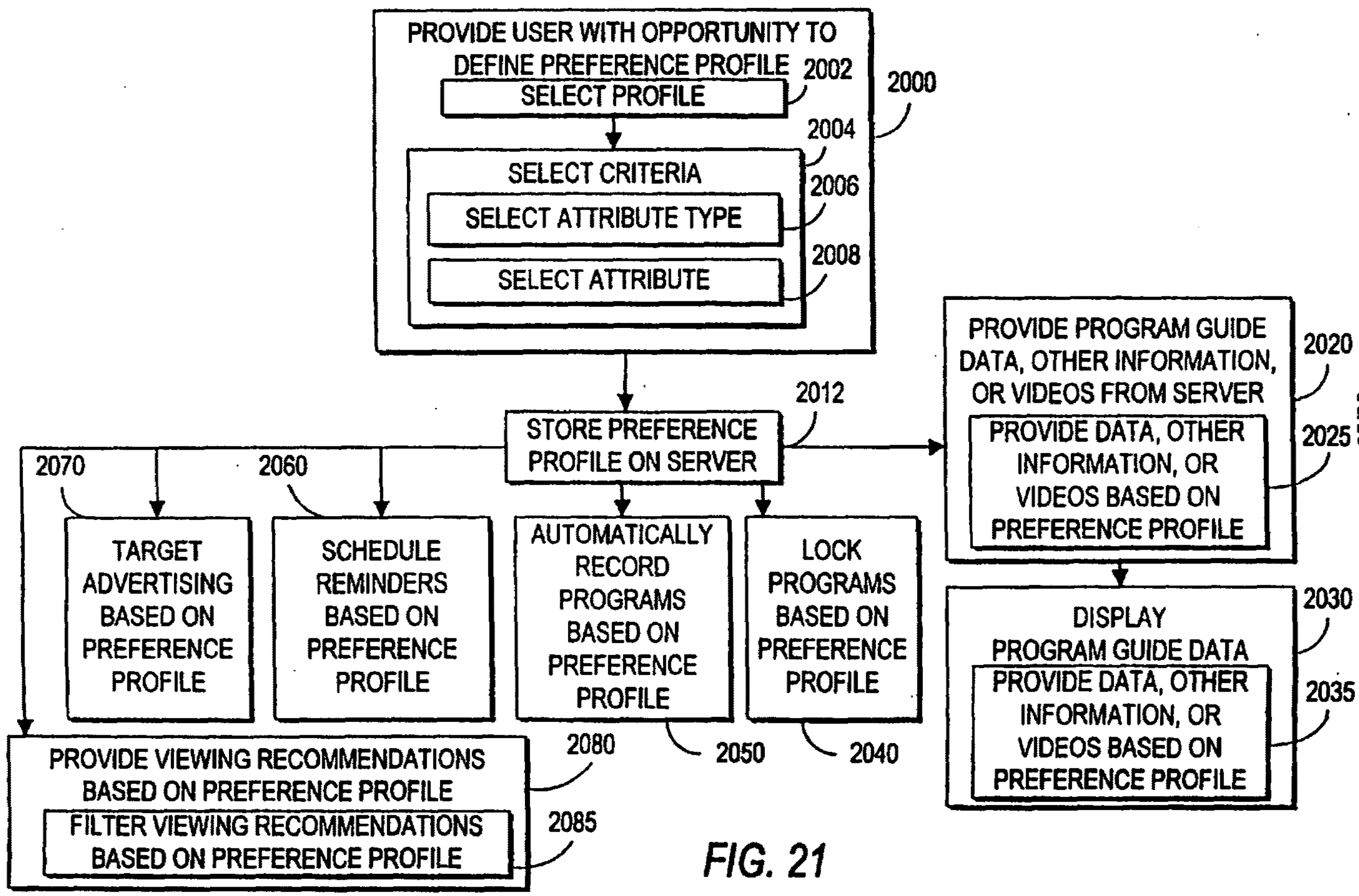


FIG. 21

PCT/US99/1951

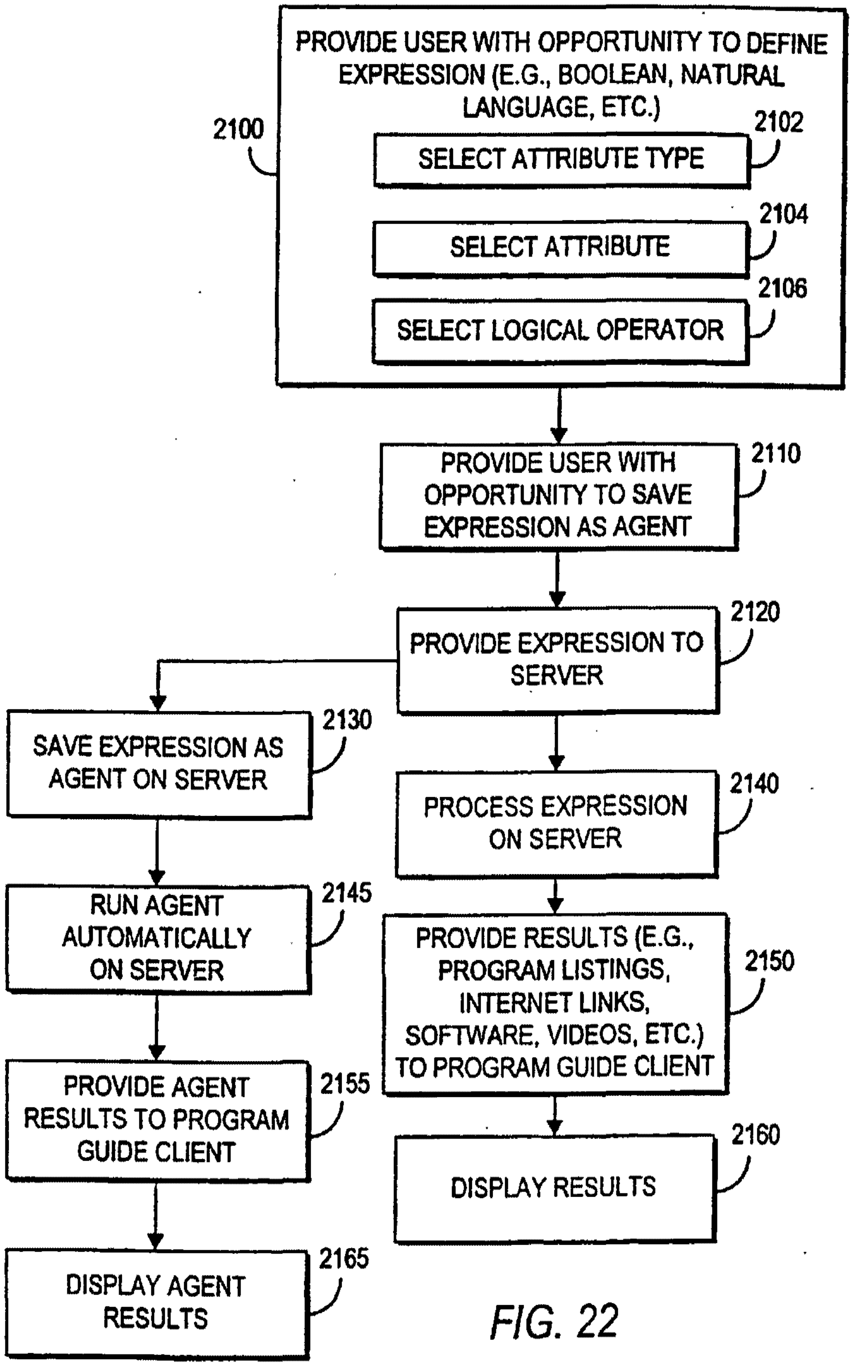

PROVIDE USER WITH OPPORTUNITY TO DEFINE EXPRESSION (E.G., BOOLEAN, NATURAL LANGUAGE, ETC.)
2100
SELECT ATTRIBUTE TYPE
2102
SELECT ATTRIBUTE
2104
SELECT LOGICAL OPERATOR
2106
PROVIDE USER WITH OPPORTUNITY TO SAVE EXPRESSION AS AGENT
2110
PROVIDE EXPRESSION TO SERVER
2120
SAVE EXPRESSION AS AGENT ON SERVER
2130
PROCESS EXPRESSION ON SERVER
2140
RUN AGENT AUTOMATICALLY ON SERVER
2145
PROVIDE RESULTS (E.G., PROGRAM LISTINGS, INTERNET LINKS, SOFTWARE, VIDEOS, ETC.) TO PROGRAM GUIDE CLIENT
2150
PROVIDE AGENT RESULTS TO PROGRAM GUIDE CLIENT
2155
DISPLAY RESULTS
2160
DISPLAY AGENT RESULTS
2165

FIG. 22

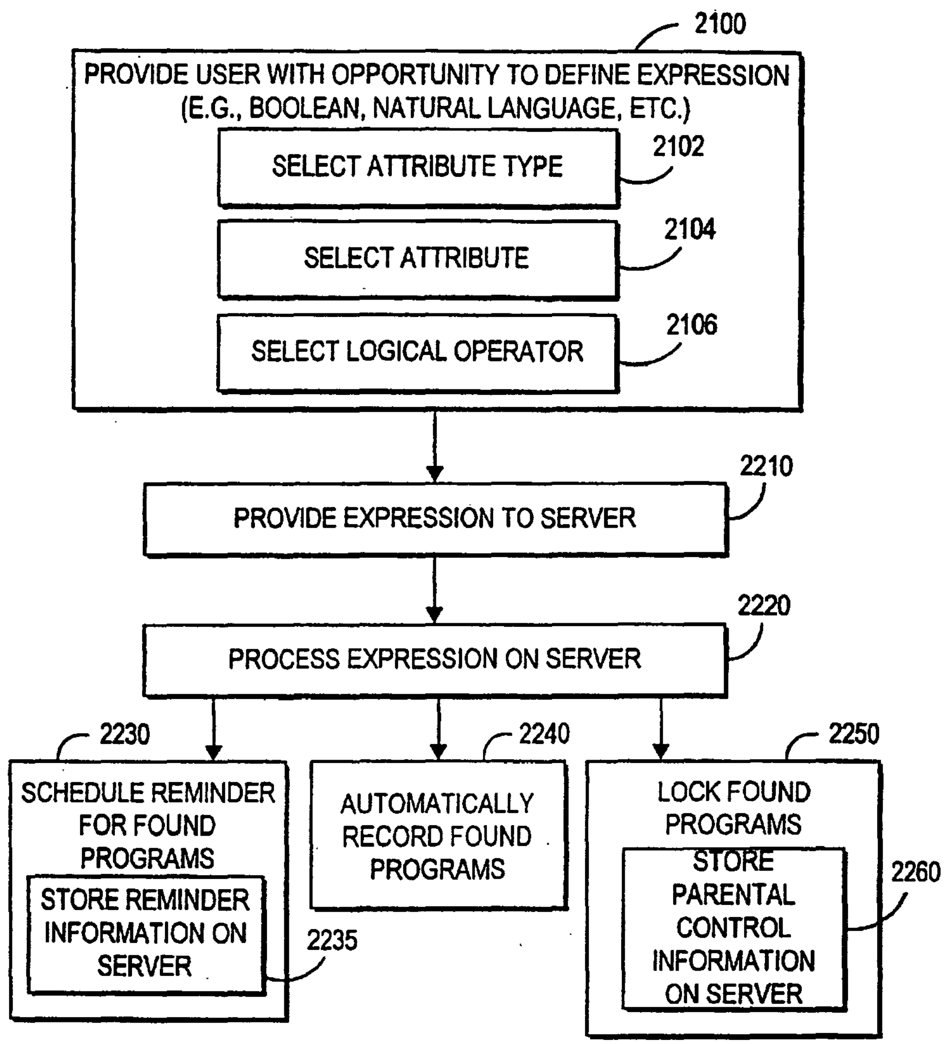

2100
PROVIDE USER WITH OPPORTUNITY TO DEFINE EXPRESSION (E.G., BOOLEAN, NATURAL LANGUAGE, ETC.)
SELECT ATTRIBUTE TYPE
2102
SELECT ATTRIBUTE
2104
SELECT LOGICAL OPERATOR
2106
PROVIDE EXPRESSION TO SERVER
2210
PROCESS EXPRESSION ON SERVER
2220
2230
SCHEDULE REMINDER FOR FOUND PROGRAMS
STORE REMINDER INFORMATION ON SERVER
2235
2240
AUTOMATICALLY RECORD FOUND PROGRAMS
2250
LOCK FOUND PROGRAMS
STORE PARENTAL CONTROL INFORMATION ON SERVER
2260

FIG. 23

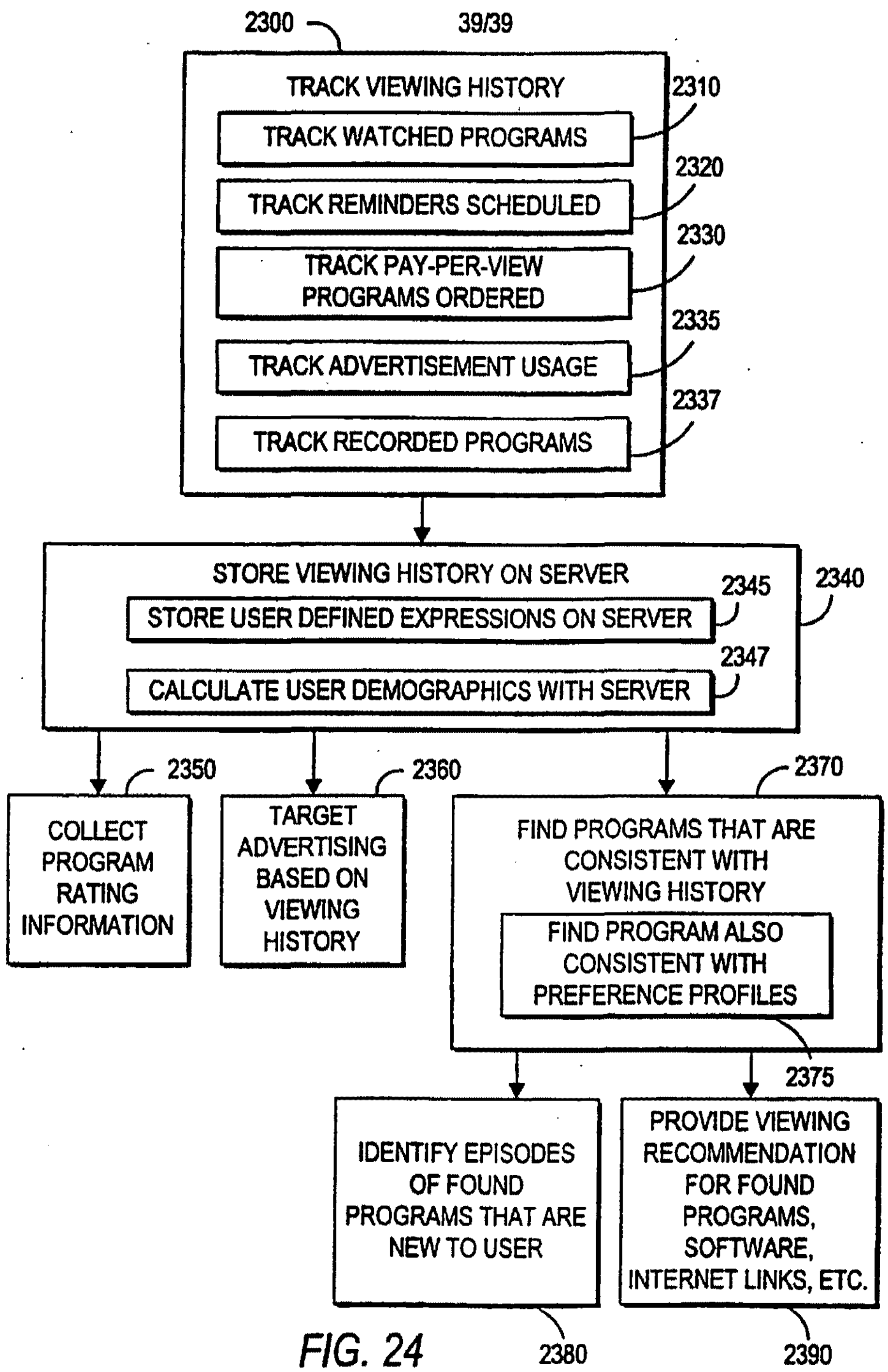

2300
TRACK VIEWING HISTORY
TRACK WATCHED PROGRAMS
2310
TRACK REMINDERS SCHEDULED
2320
TRACK PAY-PER-VIEW PROGRAMS ORDERED
2330
TRACK ADVERTISEMENT USAGE
2335
TRACK RECORDED PROGRAMS
2337
STORE VIEWING HISTORY ON SERVER
STORE USER DEFINED EXPRESSIONS ON SERVER
2345
CALCULATE USER DEMOGRAPHICS WITH SERVER
2347
2340
COLLECT PROGRAM RATING INFORMATION
2350
TARGET ADVERTISING BASED ON VIEWING HISTORY
2360
FIND PROGRAMS THAT ARE CONSISTENT WITH VIEWING HISTORY
FIND PROGRAM ALSO CONSISTENT WITH PREFERENCE PROFILES
2370
2375
IDENTIFY EPISODES OF FOUND PROGRAMS THAT ARE NEW TO USER
2380
PROVIDE VIEWING RECOMMENDATION FOR FOUND PROGRAMS, SOFTWARE, INTERNET LINKS, ETC.
2390

FIG. 24

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 99/19051

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04N7/16

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 94 14284 A (DISCOVERY COMMUNICAT INC) 23 June 1994 (1994-06-23) page 11, line 16 -page 13, line 30 page 15, line 22 -page 18, line 12 page 19, line 21 -page 21, line 10 page 32, line 11 -page 38, line 12 page 45, line 1 -page 46, line 3 page 59, line 11 -page 61, line 14 page 67, line 18 -page 70, line 32 figures 1-14 --- -/--	1-4, 6-11, 14-23, 25-30, 33-42, 44-49, 52-56

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
18 November 1999	24/11/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Van der Zaal, R

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 99/19051

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 96 41478 A (TV GUIDE ON SCREEN) 19 December 1996 (1996-12-19) page 12, line 32 -page 15, line 24 page 16, line 18 -page 35, line 19 page 36, line 12 -page 39, line 11 figures 1-58 ---	1-13, 20-32, 39-51
A	WO 98 17064 A (GEMSTAR DEVELOPMENT CORPORATION) 23 April 1998 (1998-04-23) page 5, line 6 -page 7, line 11 -----	5,24,43

1

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
F../US 99/19051

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9641478 A		PL 323914 A	27-04-1998
WO 9817064 A	23-04-1998	AU 4823197 A	11-05-1998
		EP 0932979 A	04-08-1999

Form PCT/ISA/210 (patent family annex) (July 1992)

(19) Europäisch Patentamt
European Pat nt Office
Office européen des br v t

(11) EP 0 803 826 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
29.10.1997 Bulletin 1997/44

(51) Int Cl.6: G06F 17/30, H04N 7/173

(21) Application number: 97302676.8

(22) Date of filing: 18.04.1997

(84) Designated Contracting States:
DE FR GB IT NL SE

(30) Priority: 22.04.1996 US 636118

(71) Applicant: SUN MICROSYSTEMS, INC.
Mountain View, CA 94043 (US)

(72) Inventors:
- Lindblad, Christopher
Stanford, California 94309 (US)
- Cachat, Stephan E.
Mountain View, California 94041 (US)

(74) Representative: W.P. Thompson & Co.
Coopers Building,
Church Street
Liverpool L1 3AB (GB)

(54) Video on demand applet method and apparatus for inclusion of motion video in multimedia documents

(57) The present specification describes a computer process which requests streams of motion video titles and decodes and displays the motion video signals of the stream for display in a computer display device is constructed in the form of an applet 212 of a multimedia document viewer 202 such as a World Wide Web browser. Accordingly, a designer of multimedia documents such as HTML pages can easily incorporate motion video titles into such HTML pages by specifying a few parameters of a desired title or a desired portion of a title to be requested from a video server 250. The applet 212 builds bit stream control signals from the specification of the title or the portion of the title. The bit stream control signals request transmission of the title or the portion of the title from a bit stream server such as a video server 250 and are in a form appropriate for processing by the bit stream server. The applet 212 transmits the bit stream control signals to the bit stream server 250 to thereby request that the bit stream server 250 initiate transmission of a bit stream representing the requested title or the requested portion of the title. The applet 212 also builds decoder control signals from the specification of the title or the portion of the title. The decoder control signals direct a bit stream decoder 204 to receive the requested bit stream from the bit stream server 250 and to decode a motion video signal from the bit stream. The applet 212 transmits the decoder control signals to the decoder 204 to cause the decoder 204 to receive the bit stream and to decode the motion video signal from the bit stream.

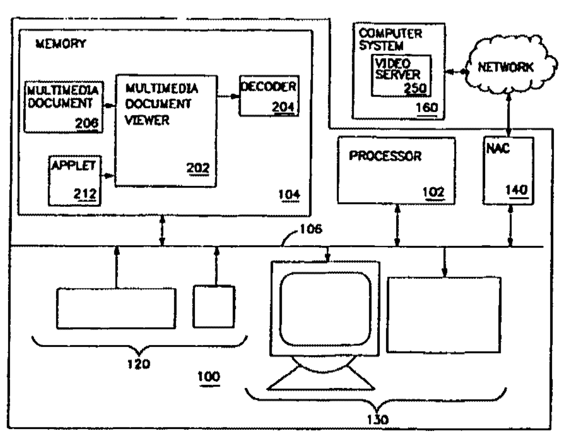


FIG. 1

EP 0 803 826 A2

Printed by Jouve, 75001 PARIS (FR)

Description

FIELD OF THE INVENTION

The present invention relates to computer graphical display of motion video and, in particular, to a method and apparatus for facilitating inclusion of motion video in multimedia computer displays.

BACKGROUND OF THE INVENTION

Video servers, including networked video servers, transmit "bit streams" to a video client. Such bit streams, which are sometimes referred to as "streams," generally represent video and/or audio signals which represent titles in a library of multimedia sources. Examples of titles of such a library typically include recordings of motion pictures. In general, a video server receives from a video client a request for a particular title and transmits a stream of the particular title to the video client. An example of a video client is a set top box which is generally known and which decodes the stream received from the video server and transmits the decoded signal to a connected television. The requesting of a particular title, receiving the stream of the particular title, and decoding the stream for display on a television are collectively and generally referred to as video on demand.

Examples of such video on demand servers are described in U.S. Patent Application Serial Number 08/572,639, filed December 14, 1995 by Kallol Mandal and Steven Kleiman and entitled "Method and Apparatus for Delivering Simultaneous Constant Bit Rate Compressed Video Streams at Arbitrary Bit Rates with Constrained Drift and Jitter" (hereinafter the '639 Application) and in U.S. Patent Application Serial Number 08/572,648, filed December 14, 1995 by Kallol Mandal and Steven Kleiman and entitled "Method and Apparatus for Distributing Network Bandwidth on a Video Server for Transmission of Bit Streams Across Multiple Network Interfaces Connected to a Single Internet Protocol (IP) Network" (hereinafter the '648 Application). Both the '639 Application and the '648 Application are incorporated herein in their entirety by reference.

The popularity of the Internet global network is growing extremely rapidly, and perhaps the most popular protocol of the Internet is the Hyper Text Transfer Protocol (HTTP) of the World Wide Web. According to the HTTP protocol of the World Wide Web, documents, which are generally referred to as "pages," incorporate text, graphical images, sound, and motion video which, when viewed, form a multimedia presentation to user. Such pages are typically viewed using a World Wide Web browser, which is a computer process capable of retrieving HTTP pages and presenting the contents of such pages to a user of a computer system through output devices such as a computer video display device and a computer audio circuit coupled to one or more audio speakers. An example of a World Wide Web browser is the Netscape browser available from Netscape Communications Corporation of Mountain View, California.

To display motion video, conventional browsers typically (i) transfer to the computer system in which the browser executes an entire data file which includes data representing a title and (ii) subsequently initiate execution of a player computer process which displays the title to the user on a computer display device. The player computer process is separate from the browser and therefore displays the motion video of the title outside of the page displayed by the browser. In addition, transferring the entire data file prior to displaying the motion video of the title delays substantially the display of the motion video since such data files are typically quite large, e.g., typically 1.8 gigabytes of data to represent a two-hour, VHS-quality motion picture.

Currently, no browser is capable of seamlessly integrating motion video streams into a page of the World Wide Web.

SUMMARY OF THE INVENTION

In accordance with the present invention, a computer process which requests streams of motion video titles and decodes and displays the motion video signals of the stream for display in a computer display device is constructed in the form of an applet of a multimedia document viewer such as a World Wide Web browser. Accordingly, a designer of multimedia documents such as HTML pages can easily incorporate motion video titles into such HTML pages by specifying a few parameters of a desired title or a desired portion of a title to be requested from a video server. The specification of the parameters is in the general form of a well-known parameter specification format dictated by the particular interface of the computer instruction language in which the applet is written.

The applet builds bit stream control signals from the specification of the title or the portion of the title. The bit stream control signals request transmission of the title or the portion of the title from a bit stream server such as a video server and ar in a form appropriate for processing by th bit stream server. The applet transmits the bit stream control signals to the bit stream server to thereby request that th bit stream server initiate transmission of a bit stream r presenting the requested title or the requested portion of th title.

The applet also builds decoder control signals from the specification of the title or the portion of the title. The decoder control signals direct a bit stream decoder to receive the requested bit stream from the bit stream server and to decode a motion video signal from the bit stream. Th applet transmits the decoder control signals to the decoder to cause the decoder to receive the bit stream and to decode the motion video signal from the bit stream.

By using an applet of a multimedia document viewer to request and control receipt by a decoder of a motion video bit stream and to control decoding of the motion video bit stream by the decoder, a designer of a multimedia document can easily and conveniently include motion video images in multimedia documents. In addition, since the applet transmits bit stream control signals to a video server, the motion video signals which can be incorporated into a multimedia document are any such motion video signals stored in such a video server. Such video servers will likely include a large number and wide variety of motion video signals, thereby providing a wealth of motion video content for inclusion in multimedia documents.

The present invention will now be further described, by way of example, with reference to the accompanying drawings, in which:-

Figure 1 is a block diagram of a computer system which is connected to a video server through a network and which includes a multimedia document viewer which in turn processes an applet to include motion video images in a representation of a multimedia document in accordance with the presenting invention.

Figure 2 is a block diagram showing the multimedia document viewer, applet, and video server of Figure 1 in greater detail.

Figure 3 is a block diagram of an applet tag of Figure 2 in greater detail.

Figure 4 is a block diagram of the applet of Figure 2 in greater detail.

DETAILED DESCRIPTION

In accordance with the present invention, a multimedia document 206 (Figure 2) includes an applet 214 which causes a multimedia document viewer 202 to execute an applet 212. Execution of applet 212 requests transmission of a bit stream of a particular title from a video server 250 and controls receipt and decoding of the bit stream by a decoder 204. Decoder 204, in response to control signals received from applet 212, decodes the received bit stream to produce a motion video image and displays the motion video image as an integral part of the representation of multimedia document 206. To include a motion video image as an integral part of a multimedia document, a designer of the multimedia document simply includes in the multimedia document an applet tag, e.g., applet tag 214, which specifies (i) applet 212, (ii) video servoer 250 as the source of a bit stream, and (iii) the particular bit stream to request from video server 250. A brief description of the operating environment of multimedia document viewer 202 and applet 212 facilitates appreciation ofthe present invention.

Figure 1 is a block diagram of a computer system 100 which is generally of the architecture of most computer systems available today. Computer system 100 includes a processor 102 which fetches computer instructions from a memory 104 through a bus 106 and executes those computer instructions. In executing computer instructions fetched from memory 104, processor 102 can retrieve data from or write data to memory 104, display information on one or more computer display devices 130, or receive command signals from one or more user-input devices 120. Processor 102 can be, for example, any of the SPARC processors available from Sun Microsystems, Inc. of Mountain View, California. Memory 104 can include any type of computer memory including, without limitation, randomly accessible memory (RAM), read-only memory (ROM), and storage devices which include magnetic and optical storage media such as magnetic or optical disks. Computer 100 can be, for example, any of the SPARCstation workstation computer systems available from Sun Microsystems, Inc. of Mountain View, California.

Sun, Sun Microsystems, the Sun Logo, Java and Hot Java are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. All SPARC trademarks are used under license and are trademarks of SPARC International, Inc. in the United States and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

Computer display devices 130 can include generally any computer display device such as a printer, a cathode ray tube (CRT), light-emitting diode (LED) display, or a liquid crystal display (LCD). User input devices 120 can include generally any user input device such as a keyboard, a keypad, an electronic mouse, a trackball, a digitizing tablet, thumbwheels, a light-sensitive pen, a touch-sensitive pad, or voice-recognition circuitry.

Computer system 100 also includes network access circuitry 140 which is coupled to processor 102 and memory 104 through bus 106 and which is coupled to a network 150. In accordance with control signals received from processor 102 through bus 106, network access circuitry 140 coordinates transfer of data through network 150 between network access circuitry 140 and similar network access circuitry (not shown) in computer 100B or other computer systems coupled to comput r syst m 100 through network 150. Th transfer of data through network 150 is conventional. Since a video stream r pres nting a VHS-quality motion picture encoded in MPEG-1 format has a bit rat of approximately 1.5 Mbit/second to 2 Mbit/second, a useful minimum threshold is that network access circuitry 140 is capable of r - ceiving data at a rate of at least 2 Mbit/second. Higher quality motion video images hav bit rates as high as 8 Mbit/second or high r. Ther fore, in one mbodiment, network acc ss circuitry 140 is capable of receiving data at a rate of at least 8 Mbit/second. Network access circuitry 140 can be generally any circuitry which is used to transfer data between a computer system and network such as computer system 100 and network 150 and can be, for example, an Ethernet controller chip.

A number of computer processes execute in processor 102 from memory 104, including a multimedia document viewer 202 and a decoder 204. Multimedia document viewer 202 is a computer process which reads a multimedia document 206 and displays the multimedia information specified in multimedia document 206 in one or more of computer display devices 130. In one embodiment, multimedia document 206 is a document in HTML format and multimedia document viewer 202 is an HTML viewer such as the Netscape World Wide Web browser available from Netscape Communications Corporation of Mountain View, California. Multimedia document viewer 202 and multimedia document 206 are shown in greater detail in Figure 2.

Multimedia document viewer 202 retrieves data and tags from a multimedia document such as multimedia document 206. A tag is data which is not itself substantive content of a multimedia document but instead provides format information and can include specification of substantive content which is to be included in the multimedia document and which is located in memory 104 outside of multimedia document 206. For example, a tag can specify a file stored in memory 104 as containing a graphical image which is to be included as substantive content of multimedia document 206. The data and tags of multimedia document 206 collectively define the composition, including substantive content and formatting, of multimedia document 206; and multimedia document viewer 202 displays such substantive content in one or more of computer display devices 130 (Figure 1) in accordance with the data and tags of multimedia document 206. In one embodiment, multimedia document 206 is an HTML document, and the data and tags of multimedia document 206 comport with the HTML language. Multimedia document 206 includes an applet tag 214 (Figure 2) which specifies an applet 212 and a number of operational characteristics of applet 212 as described more completely below.

Multimedia document viewer 202 includes an applet interpreter 210 which retrieves from applet 212 computer instructions and translates such computer instructions into computer instructions of a form appropriate for execution by processor 102 (Figure 1) and submits the translated computer instructions to processor 102 for execution. In one embodiment, applet interpreter 210 (Figure 2) translates and submits for execution a single computer instruction of applet 212 prior to translation and submission for execution of a subsequent computer instruction of applet 212. Applet interpreter 210 can be, for example, the Java applet interpreter or the Hot Java World Wide Web browser available from Sun Microsystems, Inc. and, in such an embodiment, applet 212 comports with the Java computer instruction language interpreted by the Java applet interpreter. As described more completely below, applet 212 is a novel applet which, when executed by processor 102 (Figure 1) through applet interpreter 210 (Figure 2), requests a title from a video server 250 and causes the received bit stream representing the requested title to be decoded in a decoder 204 and displayed in a computer display device as an integral part of a multimedia display of multimedia document 206.

In executing the computer instructions of applet 212, applet interpreter 210 transmits, through network 150 (Figure 1), control signals to an applications programming interface (API) 252 (Figure 2) of a video server 250 which executes within a computer system 160 (Figure 1). Illustrative examples of video server 250 of computer system 160 are described in the '639 and '648 Applications. API 252 (Figure 2) of video server 250 implements a remote procedure calling (RPC) protocol in which API 252 controls video server 250 in response to control signals received by API 252. For example, in response to control signals which request a title and which are transmitted to API 252 by applet interpreter 210, API 252 causes a bit pump 254 of video server 250 to initiate transmission through network 150 (Figure 1) to decoder 204 (Figure 2) of a bit stream representing the requested title. In addition, API 252 can transmit to applet interpreter 210 status information regarding a title stored within video server 250 or regarding a bit stream transmitted by bit pump 254 in response to control signals requesting such status information.

Decoder 204 is a computer process executing within processor 102 (Figure 1) from memory 104. Decoder 204 receives data representing a motion video display encoded in a particular format. In one embodiment, decoder 204 is the MPEG Expert (MPX) decoder available from Applied Vision and decodes motion video signals according to the MPEG-1 encoding format. Applet interpreter 210 transmits to decoder 204 control signals which control the decoding by decoder 204 of the bit stream received from bit pump 254 of video server 250. Specifically, applet interpreter 210 transmits to decoder 204 control signals directing decoder 204 to start or stop decoding the bit stream received from bit pump 254 or specifying characteristics of the bit stream received from bit pump 254 such as the bit rate, encoding format, and the coordinates of a particular location within one or more of computer display devices 130 (Figure 1) in which to display the decoded motion video images. In addition, applet 212 determines which communications port through network access circuitry 140 (Figure 1) the bit stream is to be received and transmits to decoder 204 (Figure 2) control signals identifying the selected communications port. Applet 212 can therefore determine which communications ports are used by other applications and can avoid conflicts resulting from access of decoder 204 of a communications port by selecting a communications port which is not used by another computer process of computer system 100 (Figure 1).

Applet tag 214 is shown in greater detail in Figure 3. Applet tag 214 includes a number of fields which collectively define a bit stream to be received and decoded for display by decoder 204 (Figure 2). A field is a collection of data which collectively define a item of information. Applet tag 214 includes (i) an applet identifier field 302, (ii) a width field 304, (iii) a height field 306, (iv) a server identifier field 308, and (v) an encoding format field 310. Applet tag 214 can also include any of the following optional fields: (vi) a title field 312, (vii) an image field 314, (viii) a play/pause field 316, (ix) a start field 318, and (x) a duration field 320.

Applet identifier field 302 specifies applet 212 as the applet to be retrieved and executed by applet interpreter 210. Width field 304 and height field 306 specify the width and height, respectively, in display coordinate space of a computer display device, i.e., specify the size of the viewport in which the decoded motion video image is displayed. Server identifier field 308 specifies video server 250 (Figure 2) as the source of the desired bit stream. Encoding format field 310 (Figure 3) specifies the particular encoding format, e.g., MPEG1SYS encoding format, ofthe bit stream received by decoder 204 (Figure 2). Title field 312 (Figure 3) specifies the particular title to be retrieved from server 250 (Figure 2). Alternatively, title field 312 can specify the address of a multicast bit stream.

Image field 314 (Figure 3), if included, specifies a still video image to be displayed in the space specified by width field 304 and height field 306 if the title specified by title field 312 is unavailable. Play/pause field 316, if included, specifies whether the motion video image received from video server 250 (Figure 2) is initially in a play state or in a paused state. Start field 318 (Figure 3), if included, specifies an offset into the title of a portion of the title, i.e., the point within the title at which the bit stream should begin. For example, start field 318 can specify that the requested bit stream begin at 3 minutes and 10 seconds into the title. Duration field 320, if included specifies the duration of a desired portion of the title. For example, duration field 320 can specify that a 30-minute portion of the title is requested. In one embodiment, start field 318 and duration field 320 are specified in terms of an integer number of nanoseconds.

Thus, by specifying the few fields described above and shown in Figure 3, a designer of multimedia document 206 can include as an integral part of multimedia document 206 a motion video image retrieved from video server 250. The following is an illustrative example of applet tag 214 in HTML format.

```
<applet code="SunMediaCenterPlayer.class" width=704 height=520>
<param name=port value="1973">
<param name=format value="MPEG1SYS">
<param name=host value="sqas-6">
<param name=img value="/images/bkgx.gif">
</applet>
```

Applet 212 (Figure 2) includes computer instructions which, when executed, request a title from video server 250 and control decoding and display of the decoded motion video signals by decoder 204 and is shown in greater detail in Figure 4. The computer instructions of applet 212 are organized into various levels, each of which defines a respective component of the behavior of applet 212. Applet 212 includes a player level 402, an API level 404, a decoder level 406, and a detailed decoder level 408.

Player level 402 includes computer instructions which, when executed, implement a graphical user interface in which a user can control the bit stream received by video server 250 (Figure 2) and the display of the decoded motion video signals of the bit stream by physical manipulation of one or more of user input devices 120 (Figure 1). In one embodiment, the computer instructions of player level 402 (Figure 4), when executed, cause graphical and/or textual representation of control mechanisms to be displayed in one or more of computer display devices 130 (Figure 1). Such control mechanisms are known and conventional and include, without limitation, virtual buttons, pull-down menus, virtual radio buttons, virtual check boxes, and sliding scroll bars. In a conventional manner, a user activates one or more of such control mechanisms by physical manipulation of one or more of user input devices 120 (Figure 1) and such physical manipulation results in receipt by player level 402 (Figure 4) of applet 212 of signals and/or data representing such activation.

API level 404 includes computer instructions which, when executed, implement the RPC protocol of API 252 (Figure 2) of video server 250 and invoke RPC calls to API 252 to control the bit stream transmitted by bit pump 254 in accordance with interaction of a user with the graphical user interface implemented by player level 402 (Figure 4).

Decoder level 406 and detailed decoder level 408 collectively control operation of decoder 204 (Figure 2), generally controlling the decoding of the bit stream received from video server 250 by decoder 204 and the display in a computer display device of the decoded motion video image. Decoder level 406 includes computer instructions and data structures which are not specific to any particular decoder, while detailed decoder level 408 includes computer instructions and data structures which are specific to decoder 204. It is generally preferred that detailed decoder level 408 is as small and simple as possible such that the majority of computer instructions of decoder levels 406 and 408 ar included in decoder level 406. Accordingly, adapting applet 212 (Figure 2) to operat in conjunction with a decoder oth r than decoder 204 requir s modification of only detailed decoder level 408 and, therefor , as littl modification as possible.

App ndix A is a computer sourc code listing of a preferred embodiment of applet 212. Th modules of Appendix A ar written in the Java applet computer instruction language developed by Sun Microsystems, Inc. of Mountain View, California. The computer instructions of the Java applet computer instruction language are object-oriented, and each of the modules of Appendix A represents a respective class of objects. Player level 402 (Figure 4), in this embodiment, includes classes SunMediaCenterPlayer, Player, and PositionSlider as defined in the computer source code listing of Appendix A. API level 404, in this embodiment, includes classes MsmPlayer, MsmSession, MsmAccessRight, Msm-Persistence, MsmPlaylist, MsmToString, MsmItem, MsmTitleItem, MsmDeadAirItem, MsmException, XdrBlock, and PortMapper as defined in the computer source code listing of Appendix A. Decoder level 406, in this embodiment, includes classes Decoder and DecoderImpl as defined in the computer source code listing of Appendix A. Detailed decoder level 408, in this embodiment, includes class MpxDecoderImpl as defined in the computer source code listing of Appendix A.

In the preferred embodiment of the present invention defined by Appendix A, a module "loop" includes computer instructions of the C computer instruction language and defines a loop computer process which executes independently of multimedia document viewer 202 (Figure 2). The loop computer process cooperates with multimedia document viewer 202 and decoder 204 to request and receive from video server 250 bit streams representing multicast motion video signals.

The above description is illustrative only and is not limiting. The present invention is therefore defined solely and completely by the appended claims together with their full scope of equivalents.

APPENDIX A

SunMediaCenterPlayer

```
/*
 * @(#)SunMediaCenterPlayer.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

import java.applet.*;
import java.awt.*;
import java.net.*;
import java.io.*;
import COM.Sun.isg.smcjc.*;

public class SunMediaCenterPlayer extends Applet {
    private Player player;
    private TextArea reporter;
    private Thread thread;

    public SunMediaCenterPlayer() {
     setLayout(new BorderLayout());
     player = new Player();
     add("Center", player);
    }

    public synchronized void init() {
     if (reporter != null && reporter.getParent() == this) {
         remove(reporter);
         reporter.setText("");
         validate();
     }
     try {
         int port=getParameterInt("port",-1);
         int vc=getParameterInt("vc",-1);
            if (vc!=-1){
             player.init(
             getParameterRequired("host"),
             getParameterRequired("title"),
```

```
                getParameterLong("start", 0L),
                getParameterLong("duration", 0L),
                getParameterString("loop",
"false").equalsIgnoreCase("true"),
                getParameterString("cmd", "play"),
                getParameterImage("img", null),
                      vc,"",
                      getParameterURL("CC"),
                getParameterRequired("interface"));
               }else{
                 if (port==-1){
                player.init(
                getParameterRequired("host"),
                getParameterRequired("title"),
                getParameterLong("start", 0L),
                getParameterLong("duration", 0L),
                getParameterString("loop",
"false").equalsIgnoreCase("true"),
                getParameterString("cmd", "play"),
                getParameterImage("img", null),
                      port,"",
                      getParameterURL("CC"),null);
                 }else{
                player.init(
                getParameterRequired("host"),
                "none",0L,0L,false,"play",
                getParameterImage("img", null),
                      port,
                getParameterRequired("format"),
                      getParameterURL("CC"),null);
                 }
               }
     } catch (IOException e) {
         report(e, "parsing Sun MediaCenter player parameters");
     }
    }

    public synchronized void start() {
     try player.start(); catch (IOException e)
         report(e, "starting a Sun MediaCenter player");
    }

    public synchronized void stop() {
     try player.stop(); catch (IOException e)
         report(e, "stopping a Sun MediaCenter player");
    }
```

```
    private String getParameterRequired(String key) throws
IOException {
         String val = getParameter(key);
         if (val != null) return val;
         throw new IOException("missing required parameter " + key);
        }

    private int getParameterIntRequired(String key) throws
IOException {
         String val = getParameter(key);
         if (val != null)
               try return Integer.parseInt(val); catch
(NumberFormatException e)
                  throw new IOException(
                     "parameter " + key + " is not a valid int: " +
val);
;
         throw new IOException("missing required parameter " + key);
        }

    private URL getParameterURL(String key) {
           URL res=null;
         String val = getParameter(key);
         if (val == null) return null;
           try res=new URL(val);
             catch (MalformedURLException e) try res=new
URL(getDocumentBase(),val);
               catch (MalformedURLException f)
System.out.println("MalformedURLException");
           return res;
        }

    private String getParameterString(String key, String dflt) {
         String val = getParameter(key);
         if (val == null) return dflt;
         return val;
        }

    private int getParameterInt(String key, int dflt) throws
IOException {
         String val = getParameter(key);
         if (val == null) return dflt;
         try return Integer.parseInt(val); catch
(NumberFormatException e)
            throw new IOException(
             "parameter " + key + " is not a valid int: " + val);
        }
```

```
    private long getParameterLong(String key, long dflt) throws
IOException {
     String val = getParameter(key);
     if (val == null) return dflt;
     try return Long.parseLong(val); catch (NumberFormatException
e)
          throw new IOException(
           "parameter " + key + " is not a valid long: " + val);
    }

    private Image getParameterImage(String key, Image dflt) {
     String val = getParameter(key);
     if (val == null) return dflt;
     return getImage(getDocumentBase(), val);
    }

    private synchronized void report(Exception e, String doing) {
     ByteArrayOutputStream os = new ByteArrayOutputStream();
     PrintStream ps = new PrintStream(os);
     ps.print("An error occurred while ");
     ps.print(doing);
     ps.println(":");
     e.printStackTrace(ps);
     if (reporter == null) {
         reporter = new TextArea("");
         reporter.setEditable(false);
     }
     reporter.appendText(os.toString());
     if (reporter.getParent() != this) {
         add("North", reporter);
         validate();
     }
    }

}
```

Player

```
/*
 * @(#)Player.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.1sc
 * author Christopher Lindblad   ( Msm API & Mpx API )
 * author Stephane CACHAT        (Closed Caption & Multicasting)
 *
 */

package COM.Sun.isg.smcjc;

import java.applet.*;
import java.awt.*;
import java.io.*;
import java.net.*;

public class Player extends Panel implements Runnable {
    private long playDuration;
    private long startOffset;
    private long seekPosition;
    private long tellPosition;
    private double tellPositiond;
    private MsmPlayer player;
    private String host;
    private String titleName;
    private String msg;
    private String format;
    private Image img;
    private Thread thread;
    private Panel controlLine;
    private Panel controlButtons;
    private TextArea reporter;
    private Decoder decoder;
    private PositionSlider positionSlider;
    private Button[] buttons;
    private int cmd = 999;
    private int initialCmd;
    private int port;
    private boolean loop;
    private boolean Msm;
    private URL CC;
    private List CCt;
```

```
        private int CCz=0;
        private String[] CCb=new String[1024];
        private Double[] CCi=new Double[1024];
        private int CCl=0;
        private int CCo=0;
        private int CCm=0;
        private boolean playing = false;
        private TextField CCs;
        private String ATM;

        public Player() {
         setLayout(new BorderLayout());
         decoder = new Decoder();
         add("Center", decoder);
        }

        public synchronized void init(
         String host, String titleName,
         long startOffset, long playDuration, boolean loop,
         String cmd, Image img,int port,String format,URL CC,String
    ATM)
         throws IOException {
                URLConnection uc;
                Double d;
                String str;
                int i=0;
                int j=0;

                this.port=port;
                if ((port!=-1)&&(ATM==null)){
                    Msm=false;
                }else{
                    Msm=true;
                 this.initialCmd = parseCmd(cmd);
                }
                this.CC=CC;
                this.ATM=ATM;
             this.host = host;
             this.titleName = titleName;
             this.startOffset = startOffset;
             this.playDuration = playDuration;
             this.loop = loop;
             this.img = img;
             this.format = format;
                if (CC!=null){
                  CCt= new List();
                  CCt.minimumSize(6);
```

```
                    CCt.preferredSize(6);
                    uc= CC.openConnection();
                    DataInputStream in=new
      DataInputStream(uc.getInputStream());
                    str="-";
                CCb[i]=new String("*");
                    CCi[i]=new Double(0.0);
                    i++;
                    while (in.available()>0){
                      str=in.readLine();
                      while
      ((str.trim().length()==0)&&(in.available()>0)) str=in.readLine();
                      if (str!=null){
                        j=str.trim().indexOf(' ');
                        if (j>0){
                    CCb[i]=new String(str.substring(j+1)).trim();
                          CCt.addItem(CCb[i]);
                          if (CCb[i]==null) CCb[i]="*";
                         CCi[i]=new Double(str.substring(0,j).trim());
                          i++;
                      }
                      }
                    }
                    CCm=i-1;
                    in.close();
                  }
         }

       public synchronized void start() throws IOException {
        if (reporter != null && reporter.getParent() == this) {
            remove(reporter);
            reporter.setText("");
            validate();
        }
        if (thread == null) {
            cmd = initialCmd;
            thread = new Thread(this);
            thread.start();
        }
       }

       public synchronized void stop() throws IOException {
        if (thread != null) {
            thread = null;
            notify();
        }
       }
```

```
public synchronized boolean action(Event evt, Object arg) {
  if (buttons != null && evt.target instanceof Button) {
      Button b = (Button)evt.target;
      for (int i = 0; i < buttons.length; i++) {
       if (b == buttons[i]) cmd = i;
      }
      notify();
  };
  if (CC != null && evt.target ==CCt) {
      seekPosition = (long)(new
Double(CCi[CCt.getSelectedIndex()].doubleValue()*10).intValue())*
100000000;
      cmd = SEEK;
      notify();
     };
  if (CC != null && evt.target==CCs) {
         if (CCl<CCm){
           CCz=CCl+1;
         }else{
           CCz=0;
         };

while((CCz!=CCl)&&(CCb[CCz].indexOf(CCs.getText())<0)) {
           CCz++;
           if (CCz>CCm) CCz=0;
         }
         if (CCb[CCz].indexOf(CCs.getText())>=0){
           CCt.select(CCz);
           CCt.makeVisible(CCz+1);
        seekPosition = (long)(new
Double(CCi[CCt.getSelectedIndex()].doubleValue()*10).intValue())*
100000000;
        cmd = SEEK;
        notify();
         }
     }
  return true;
}

private void setConnect(MsmConnect connect) throws
IOException {
     try {
         player.setConnect(connect);
     } catch (MsmException e) {
         /* Try it with destTiAddr in beta 0.5 syntax. */
System.out.println("DesTiAddr="+connect.destTiAddr);
         InputStream is = new
```

```
StringBufferInputStream(connect.destTiAddr);
        StreamTokenizer st = new StreamTokenizer(is);
        String host;
        int udpport;
           if(ATM==null){
          if (st.nextToken() == StreamTokenizer.TT_WORD &&
          st.sval.equals("host") &&
          st.nextToken() == '=' &&
          st.nextToken() == StreamTokenizer.TT_WORD &&
          (host = st.sval) != null &&
          st.nextToken() == ',' &&
          st.nextToken() == StreamTokenizer.TT_WORD &&
          st.sval.equals("udpport") &&
          st.nextToken() == '=' &&
          st.nextToken() == StreamTokenizer.TT_NUMBER &&
          (udpport = (int)st.nval) != 0) {
          connect.destTiAddr = "be0,"+host+","+udpport;
          player.setConnect(connect);
           } else {
          throw e;
           }
           }else{
           throw e;
           }
    }
   }

   public synchronized void run() {
    Thread currentThread = Thread.currentThread();
    MsmSession session = null;
    MsmTitle title = null;
       MsmItem[] items = null;
       int speed=0;


    if (Msm){
           controlButtons = new Panel();
        controlButtons.setLayout(new FlowLayout());
        controlButtons.add(cmds[PAUSE], new
Button(labels[PAUSE]));
        controlLine = new Panel();
        controlLine.setLayout(new BorderLayout());
        controlLine.add("East", controlButtons);
        positionSlider = new PositionSlider(this);
        controlLine.add("Center", positionSlider);
        add("South", controlLine);
           if (CC!=null){
```

```
                    Panel CCp=new Panel();
                    CCp.setLayout(new BorderLayout());
                    Panel CCq=new Panel();
                    CCq.setLayout(new BorderLayout());

                    CCs= new TextField(15);
                    CCs.isEditable();
                 CCq.add("South", CCs);
                    Label l=new Label("Search");
                 CCq.add("Center", l);
                    CCp.add("East",CCq);
                    CCp.add("Center",CCt);
                 controlLine.add("North",CCp);
                 }
             }
        try {
            if (Msm){
                    items = new MsmItem[1];
                 session = new MsmSession(host);
                 title = session.getTitleStatus(titleName);
                 if (playDuration == 0L) playDuration =
title.totalPlayDuration;
                    format=title.format;
                }
            decoder.init(format, img,host,port,ATM);
            if (Msm){
                    titleInit(title);
                 player = new MsmPlayer(session, info(),
MsmPlayer.TIME_MAXTIME);
                 player.setPersistence(new MsmPersistence(
                  MsmPersistence.TYPE_NONE,
                  MsmPlayer.TIME_MAXTIME));
                 items[0] = new MsmTitleItem(
                  titleName, playDuration, startOffset, playDuration,
                  playDuration, false, true, title.maxBitRate);
                 player.setPlaylist(new MsmPlaylist(
                  MsmPlayer.TIME_CURRENT, loop, 0,
MsmPlayer.TIME_MAXTIME,
                  items, 0, 0));
                 setConnect(new MsmConnect(
                  decoder.destTiAddr(), decoder.encap(),
title.maxBitRate));
                 playing = false;
                 speed = MsmPlayer.SPEED_FORWARD;
                }else{
                 invalidate();
                 validate();
```

```
            }
            while (currentThread == thread) {
             switch (cmd) {
             case NOP: {
                 if (Msm) {
                            MsmPlayStatus status =
player.getPlayStatus();
                        if (tellPosition != status.currentPosition) {
                         tellPosition = status.currentPosition;
                         positionSlider.repaint();
                        }

tellPositiond=(tellPosition/1000000000)+3.0;
                            if (CC!=null){
                              CCo=CCl;
                              while
((CCi[CCl+1].doubleValue()<tellPositiond)&&(CCl+1<CCm))  CCl++;
                              while
((CCi[CCl].doubleValue()>tellPositiond)&&(CCl>0)) CCl--;
                              if (CCo!=CCl) {
                                 CCt.select(CCl-1);
                                 CCt.makeVisible(CCl);
                              }
                            }
                       player.setPersistence(new MsmPersistence(
                        MsmPersistence.TYPE_NONE,
                        status.currentDate+60*1000000000L));
                 }
                        break;
             }
             case PAUSE: {
                 decoder.pause();
                 if (Msm) player.pause(MsmPlayer.TIME_CURRENT);
                 decoder.flush();
                 playing = false;
                 decoder.play();
                 break;
             }
             case GOTO_START: {
                 tellPosition = 0L;
                 if (Msm) positionSlider.repaint();
                 decoder.stop();
                 if (Msm) player.play(MsmPlayer.SPEED_FORWARD,
                        0L,
                        0L,
                        MsmPlayer.TIME_CURRENT);
                 decoder.flush();
```

```
            break;
        }
        case GOTO_END: {
            tellPosition = playDuration;
            if (Msm) positionSlider.repaint();
            decoder.stop();
            if (Msm) player.play(MsmPlayer.SPEED_REVERSE,
                  playDuration,
                  0L,
                  MsmPlayer.TIME_CURRENT);
            decoder.flush();
            break;
        }
        case SEEK: {
            tellPosition = seekPosition;
            if (Msm) positionSlider.repaint();
            if (playing) {
             decoder.flush();
             if (Msm) player.play(speed,
                      seekPosition,
                      MsmPlayer.TIME_MAXTIME,
                      MsmPlayer.TIME_CURRENT);
            } else {
             long duration = SEEKDURATION;
             long position = seekPosition-duration;
             if (position < 0L) {
                 duration += position;
                 position -= position;
             }
             decoder.play();
             decoder.flush();
             if (Msm) player.play(MsmPlayer.SPEED_FORWARD,
                      position,
                      duration,
                      MsmPlayer.TIME_CURRENT);
            }
            break;
        }
        default: {
            decoder.play();
            decoder.flush();
            if (Msm){
                      speed = cmd;
                player.play(speed,
                  MsmPlayer.TIME_CURRENT,
                  MsmPlayer.TIME_MAXTIME,
                  MsmPlayer.TIME_CURRENT);
```

```
                        playing = true;
                              if (CC!=null)
                                if (CCo!=CCl) {
                                   CCt.select(CCl-1);
                                   CCt.makeVisible(CCl);
                                }
                          }
                }
                }
                cmd = NOP;
                try wait(100); catch (InterruptedException e);
               }
          } catch (Exception e) {
              report(e, "communicating with a Sun MediaCenter
server");
          } finally {
              try {
               try decoder.stop(); catch (Exception e)
                   report(e, "stopping a video decoder");
                     if (Msm){
                   if (player != null) {
                       try player.delete(); catch (Exception e)
                        report(e, "deleting a Sun MediaCenter
player");
                       player = null;
                         }
                }
              } finally {
                     if(Msm){
                   if (session != null) {
                       try session.close(); catch (Exception e)
                        report(e, "closing a Sun MediaCenter
connection");
                   }
                     }
              }
          }
        }

        /*
         * Callback from the PositionSlider.
         * Unsynchronized to avoid deadlock.
         * @return value between 0 and 1 indicating where in the file
we are.
         */
        public double tell() {
         if (playDuration == 0L) return 0.0D;
```

```
 return (double)tellPosition / (double)playDuration;
}

/*
 * Callback from the PositionSlider.
 * Seek to a relative position in a file.
 * @param position Value between 0 and 1
 * indicating where in the file to go.
 */
public synchronized void seek(double position) {
 if (playDuration == 0) return;
 seekPosition = (long)(position*playDuration);
 cmd = SEEK;
 notify();
}

private String info() throws UnknownHostException {
    String hostName =
InetAddress.getLocalHost().getHostName();
    String javaVersion = System.getProperty("java.version");
    String javaVendor = System.getProperty("java.vendor");
    String osArch = System.getProperty("os.arch");
    String osName = System.getProperty("os.name");
    String osVersion = System.getProperty("os.version");
    return hostName
        + " Java " + javaVersion + " (" + javaVendor + ")"
        + " (" + osArch + " " + osName + " " + osVersion +
")";
}

private void addButton(int i) {
 buttons[i] = new Button(labels[i]);
 controlButtons.add(cmds[i], buttons[i]);
}

/**
 * Initialize for a title.
 * @param title The title to play.
 */
private void titleInit(MsmTitle title) throws IOException {
 controlButtons.removeAll();
 buttons = new Button[labels.length];
 for (int i = MsmPlayer.SPEED_SLOWEST_FORWARD;
      i <= MsmPlayer.SPEED_SCENE_FORWARD;
      i++) {
     if (title.speedScale[i] != 0) {
      addButton(GOTO_START);
```

```
              break;
            }
          }
          for (int i = MsmPlayer.SPEED_SCENE_REVERSE;
               i <= MsmPlayer.SPEED_SLOWEST_REVERSE;
               i++) {
              if (title.speedScale[i] != 0) addButton(i);
          }
          addButton(PAUSE);
          for (int i = MsmPlayer.SPEED_SLOWEST_FORWARD;
               i <= MsmPlayer.SPEED_SCENE_FORWARD;
               i++) {
              if (title.speedScale[i] != 0) addButton(i);
          }
          for (int i = MsmPlayer.SPEED_SCENE_REVERSE;
               i <= MsmPlayer.SPEED_SLOWEST_REVERSE;
               i++) {
              if (title.speedScale[i] != 0) {
               addButton(GOTO_END);
               break;
              }
          }
          /* recompute layout */
          controlLine.invalidate();
          invalidate();
          validate();
          /* resize if we need to */
          Component c = getParent();
          while (c != null) {
              if (c instanceof Applet) {
               Dimension ps = c.preferredSize();
               Rectangle b = c.bounds();
               if (ps.width != b.width || ps.height != b.height) {
                   // This wedges Netscape Navigator 2.0
                   // c.resize(ps.width, ps.height);
               }
               break;
              }
          }
        }

        private void report(Exception e, String doing) {
         ByteArrayOutputStream os = new ByteArrayOutputStream();
         PrintStream ps = new PrintStream(os);
         ps.print("An error occurred while ");
         ps.print(doing);
         ps.println(":");
```

```
        e.printStackTrace(ps);
        if (reporter == null) {
            reporter = new TextArea("");
            reporter.setEditable(false);
        }
        reporter.appendText(os.toString());
        if (reporter.getParent() != this) {
            add("North", reporter);
            validate();
        }
    }

    private int parseCmd(String cmd) throws IOException {
        for (int i = 0; i < cmds.length; i++) {
            if (cmd.equalsIgnoreCase(cmds[i])) return i;
        }
        throw new IOException("Not a valid Player command: "+cmd);
    }

    private static final long SEEKDURATION = 4000000000L;

    private static final int PAUSE = 16;
    private static final int GOTO_START = 17;
    private static final int GOTO_END = 18;
    private static final int SEEK = 19;
    private static final int NOP = 20;

    private static final String[] labels = {
        "|<<<<",        // MsmPlayer.SPEED_SCENE_REVERSE
        "<<<<",              // MsmPlayer.SPEED_FASTEST_REVERSE
        "<<<",               // MsmPlayer.SPEED_FASTER_REVERSE
        "<<",                // MsmPlayer.SPEED_FAST_REVERSE
        "<",            // MsmPlayer.SPEED_REVERSE
        "|<",                // MsmPlayer.SPEED_SLOW_REVERSE
        "||<",               // MsmPlayer.SPEED_SLOWER_REVERSE
        "|||<",              // MsmPlayer.SPEED_SLOWEST_REVERSE
        ">|||",              // MsmPlayer.SPEED_SLOWEST_FORWARD
        ">||",               // MsmPlayer.SPEED_SLOWER_FORWARD
        ">|",                // MsmPlayer.SPEED_SLOW_FORWARD
        ">",            // MsmPlayer.SPEED_FORWARD
        ">>",                // MsmPlayer.SPEED_FAST_FORWARD
        ">>>",               // MsmPlayer.SPEED_FASTER_FORWARD
        ">>>>",              // MsmPlayer.SPEED_FASTEST_FORWARD
        ">>>>|",        // MsmPlayer.SPEED_SCENE_FORWARD
        "||",                // PAUSE
        "||<<<<",       // GOTO_START
        ">>>>||",       // GOTO_END
```

```
     "",              // SEEK
     "",              // NOP
    };

    private static final String[] cmds = {
     "scene_reverse",     // MsmPlayer.SPEED_SCENE_REVERSE
     "fastest_reverse",   // MsmPlayer.SPEED_FASTEST_REVERSE
     "faster_reverse",    // MsmPlayer.SPEED_FASTER_REVERSE
     "fast_reverse",          // MsmPlayer.SPEED_FAST_REVERSE
     "reverse",           // MsmPlayer.SPEED_REVERSE
     "slow_reverse",          // MsmPlayer.SPEED_SLOW_REVERSE
     "slower_reverse",    // MsmPlayer.SPEED_SLOWER_REVERSE
     "slowest_reverse",   // MsmPlayer.SPEED_SLOWEST_REVERSE
     "slowest_forward",   // MsmPlayer.SPEED_SLOWEST_FORWARD
     "slower_forward",    // MsmPlayer.SPEED_SLOWER_FORWARD
     "slow_forward",          // MsmPlayer.SPEED_SLOW_FORWARD
     "play",              // MsmPlayer.SPEED_FORWARD
     "fast_forward",          // MsmPlayer.SPEED_FAST_FORWARD
     "faster_forward",    // MsmPlayer.SPEED_FASTER_FORWARD
     "fastest_forward",       // MsmPlayer.SPEED_FASTEST_FORWARD
     "scene_forward",     // MsmPlayer.SPEED_SCENE_FORWARD
     "pause",        // PAUSE
     "goto_start",        // GOTO_START
        "goto_end",       // GOTO_END
        "seek",               // SEEK
        "nop",            // NOP
    };

}
```

P sitionSlider

```
/*
 * @(#)PositionSlider.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.awt.*;
import java.io.*;

class PositionSlider extends Canvas {
    private Player player;
    private int hgap;
    private int vgap;
    private int wid;

    public PositionSlider(Player player) {
     this(player, 5, 5, 6);
    }

    public PositionSlider(Player player, int hgap, int vgap, int
wid) {
     this.player = player;
     this.hgap = hgap;
     this.vgap = vgap;
     this.wid = wid;
    }

    public void update(Graphics g) {
     paint(g);
    }

    public synchronized void paint(Graphics g) {
     Rectangle r = bounds();
     int position = (int)((r.width-hgap*2)*player.tell())+hgap;
     g.setColor(getBackground());
     g.fillRect(0, 0, r.width, vgap*2);
     g.fillRect(0, r.height-vgap*2, r.width, vgap*2);
     g.fillRect(0, vgap*2, r.width-hgap*2, r.height-vgap*2);
```

```
    g.fillRect(r.width-hgap, vgap*2, r.width, r.height-vgap*2);
    g.fill3DRect(hgap, vgap*2, r.width-hgap*2, r.height-vgap*4,
false);
    g.fill3DRect(position-2, vgap, wid, r.height-vgap*2, true);
   }

   private synchronized void seek(int x) {
    Rectangle r = bounds();
    double position = ((double)(x-hgap)) /
((double)(r.width-hgap*2));
    if (position < 0.0D) position = 0.0D;
    if (position > 1.0D) position = 1.0D;
    player.seek(position);
   }

   public boolean mouseDown(Event e, int x, int y) {
    seek(x);
    return true;
   }

   public boolean mouseDrag(Event e, int x, int y) {
    seek(x);
    return true;
   }

}
```

MsmPlayer

```
/*
 * @(#)MsmPlayer.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.io.*;

/**
 * Media Stream Manager Client API
 *
 * MSM allows for the creation of "players".  A player is a
persistent entity
 * that provides for the scheduled delivery of isochronous data
to a
 * particular destination.  To accomplish this task, a player
maintains a
 * playlist of titles, the state of a "playhead" which traverses
this
 * playlist, and an access list controlling who can perform
various functions
 * on the player.
 *
 * MSM, when supplied with titles that have been prepared for
presentation at
 * multiple presentation rates, manages the position index
lookups and stream
 * switching necessary for "trick play".
 *
 * Associated with a player is a "playhead" that maintains a
destination for
 * the isochronous data (possibly different than the controlling
client) and a
 * "playPosition" which travels along the playlist at the
selected
 * presentation rate and delivers isochronous data as scheduled
to the
 * destination.  The position, presentation rate, and
```

```
presentation direction
 * of the playhead can be controlled via play(), pause(), and
resume(). The
 * initiation of play can be synchronized with "wall clock time"
via play();
 * presentation will then stay synchronized with wall-clock time
as long as
 * presentation rate and direction are Normal-Rate,
Forward-Direction.
 *
 * Latency from invocation of the play() request until actual
start of stream
 * may be reduced by "pre-rolling" with a play() request that has
zero
 * duration. This may also be used to set a current playlist
position without
 * actually starting play.
 *
 * MSM manages concurrent updates to a playlist by returning a
modification
 * timestamp with playlist status. The modification timestamp
indicates the
 * time of the last modification of the playlist. When a client
wishes to
 * update a playlist, the client will first obtain status
containing a
 * modification timestamp to understand the current state of the
playlist.
 * Based on this status, the client then determines the
appropriate updates
 * and passes those updates along with the modification timestamp
of the
 * status on which the updates were based to msm. If msm finds
that the
 * modification timestamp has not changed, implying that the
clients updates
 * are based on currently valid playlist state, the playlist
update will
 * succeed. If the modification timestamp indicates that the
playlist has
 * been modified since this client obtained status, the update
will be
 * rejected. In this case, the client should reobtain status,
reaccess the
 * update, and then if appropriate resubmit the update with the
modification
 * timestamp of the new status. There is a designated timestamp
```

```
that forces
 * playlist modifications, this may be used if some external
method of
 * concurrency control is preferred.
 *
 * MsmPlaylist may be edit while play is in progress.  Normally,
changes to the
 * playlist will not take effect until the current item in play
completes.  A
 * playlist modification can be forced to take effect immediately
by calling
 * resume().  resume() should be called with the speed argument
being the
 * current (or desired new speed) and the startPosition argument
being
 * TIME_CURRENT. If the contents of the playlist at the current
position of
 * the playhead have not been modified, this call will not
disturb the
 * outgoing data stream.
 *
 * MSM optionally maintains players persistently across server
outages. When
 * this option is selected, a successful return from a player
request
 * indicates that the player modifications have been made
persistently.
 * Persistent players may optionally restart play on state
recovery, play may
 * be restarted at the last played position or at the position
that the
 * position that play would be add had no outage occurred.
 *
 * Access to read and modify players is controlled by access
control lists
 * associated with the players.  These may be modified by
 * msmPlayerSetAccess().
 *
 * Access rights are "Read", "Control", and "Admin".  Read rights
all state to
 * be seen.  Control rights allow "trick-play" operations to be
controlled.
 * Admin rights allow creation of players, and connection,
access, and
 * persistence attributes of players to be set.  Access rights
are associated
 * with "agents" (eg users) appropriate for the authorization
```

```
mechanism
 * selected.  The reserved agent name "*" represents ALL agents,
those
 * granting a right to "*", grants the right to all agents.
 *
 */
public class MsmPlayer {
    private MsmSession session;
    private byte[] handle;

    /**
     * Creates a player.  The player is initialized
non-persistent.
     * @param session A server session.
     * @param info Saved, but uninterpreted by server. May be
null.
     *    Used to describe the player for administrative purposes.
     * @param terminateDate Date at which player should be
auto-deleted.
     *    If TIME_MAXTIME, the player will never be auto-deleted,
it must
     *    be deleted via delete.
     * @exception IOException If an error has occurred.
     */
    public MsmPlayer(MsmSession session, String info, long
terminateDate)
     throws IOException {
         this.session = session;
         XdrBlock call = session.newCall(PLAYER_CREATE);
         call.xdroutString(info);
         call.xdroutMsmTime(terminateDate);
         XdrBlock reply = session.rpc(call);
         handle = reply.xdrinBytes(HANDLELEN);
         reply.done();
     }

    MsmPlayer(MsmSession session, XdrBlock xdr) {
     this.session = session;
     handle = xdr.xdrinBytes(HANDLELEN);
    }

    void xdrout(XdrBlock xdr) {
     xdr.xdroutBytes(handle,HANDLELEN);
    }

    public MsmSession getSession() {
     return session;
```

```
    }

    public byte[] getHandle() {
     return handle;
    }

    /**
     * Opens an existing player.
     * @param session A server session.
     * @param handle An opaque handle to the player.
     */
    public MsmPlayer(MsmSession session, byte[] handle) {
     this.session = session;
     this.handle = handle;
    }

    /**
     * Deletes the player.  In progress play of the player is
stopped.
     * @exception IOException If an error has occurred.
     */
    public void delete() throws IOException {
     XdrBlock call = session.newCall(PLAYER_DELETE);
     this.xdrout(call);
     session.rpc(call).done();
    }

    /**
     * Modifies access control list for player.
     * @param rights The access modifications.
     * @exception IOException If an error has occurred.
     */
    public void setAccess(MsmAccessRight[] rights) throws
IOException {
     XdrBlock call = session.newCall(PLAYER_SETACCESS);
     this.xdrout(call);
     call.xdroutInt(rights.length);
     for (int i = 0; i < rights.length; i++)
rights[i].xdrout(call);
     session.rpc(call).done();
    }

    /**
     * Get access control list for player.
     * @return The access modifications.
     * @exception IOException If an error has occurred.
     */
```

```
    public MsmAccessRight[] getAccess() throws IOException {
     XdrBlock call = session.newCall(PLAYER_GETACCESS);
     this.xdrout(call);
     XdrBlock reply = session.rpc(call);
     MsmAccessRight[] result = new
MsmAccessRight[reply.xdrinInt()];
     for (int i = 0; i < result.length; i++) {
         result[i] = new MsmAccessRight(reply);
     }
     reply.done();
     return result;
    }

    /**
     * Sets persistence for player.
     * @param prstp A MsmPersistence containing the persistence
to be set.
     * @exception IOException If an error has occurred.
     */
    public void setPersistence(MsmPersistence prst) throws
IOException {
     XdrBlock call = session.newCall(PLAYER_SETPERSISTENCE);
     this.xdrout(call);
     prst.xdrout(call);
     session.rpc(call).done();
    }

    /**
     * Get persistence information for player.
     * @exception IOException If an error has occurred.
     */
    public MsmPersistence getPersistence() throws IOException {
     XdrBlock call = session.newCall(PLAYER_GETPERSISTENCE);
     this.xdrout(call);
     XdrBlock reply = session.rpc(call);
     MsmPersistence result = new MsmPersistence(reply);
     reply.done();
     return result;
    }

    /**
     * Replaces a portion of the playlist for this player. The
portion to be
     * replaced and the new titles to inserted are indicated via
MsmPlaylist
     * struct pointed to by playlistp.
     * @param playlist A MsmPlaylist that indicates the period on
```

```
the playlist
     *   to be (re)scheduled and the new titles to place within
that period.
     * @exception IOException If an error has occurred.
     */
    public void setPlaylist(MsmPlaylist playlist) throws
IOException {
     XdrBlock call = session.newCall(PLAYER_SETPLAYLIST);
     this.xdrout(call);
     playlist.xdrout(call);
     session.rpc(call).done();
    }

    /**
     * Obtains a portion of the playlist for this player.
     * @param startPosition The position within the playlist at
which to start
     *     returning status.
     * @param playlistDuration The number of milliseseconds of
the playlist for
     *   which to return status.
     * @exception IOException If an error has occurred.
     */
    public MsmPlaylist getPlaylist(long startPosition, long
playlistDuration)
     throws IOException {
         XdrBlock call = session.newCall(PLAYER_GETPLAYLIST);
         this.xdrout(call);
         call.xdroutMsmTime(startPosition);
         call.xdroutMsmTime(playlistDuration);
         XdrBlock reply = session.rpc(call);
         MsmPlaylist result = new MsmPlaylist(reply);
         reply.done();
         return result;
     }

    /**
     * Obtains the playlist for this player.
     * @exception IOException If an error has occurred.
     */
    public MsmPlaylist getPlaylist() throws IOException {
     return getPlaylist(TIME_ZERO, TIME_MAXTIME);
    }

    /**
     * MsmConnects a player to the specified destination address.
     * An error is return if play is in progress at the time of a
```

```
setConnect().
     * @param connect A MsmConnect instance containing a
transport-independent
     *   address string for the destination of Media Server data
controlled
     *   by this player.  A connectp of NULL disconnects the
player from the
     *   current destination.
     * @exception IOException If an error has occurred.
     */
    public void setConnect(MsmConnect connect) throws IOException
{
     XdrBlock call = session.newCall(PLAYER_SETCONNECT);
     this.xdrout(call);
     connect.xdrout(call);
     session.rpc(call).done();
    }

    /**
     * Get current connection for player.
     * @exception IOException If an error has occurred.
     */
    public MsmConnect getConnect() throws IOException {
     XdrBlock call = session.newCall(PLAYER_GETCONNECT);
     this.xdrout(call);
     XdrBlock reply = session.rpc(call);
     MsmConnect result = new MsmConnect(reply);
     reply.done();
     return result;
    }

    /**
     * Schedules play to commence at startDate.  Play
     * will begin at playlist startPosition and continue for
playDuration NPT
     * seconds or until paused.  An error is returned if the
player is not
     * connected.
     * Only one play() command can be pending, a second play()
overrides any
     * pending play().
     * @param speed The speed at which to play.
     * @param startPosition The position within the playlist at
which to begin
     *   play.  TIME_CURRENT means the current play position.
     * @param playDuration The duration of play.
     *   TIME_MAXTIME indicates "forever".
```

```
     * @param startDate The wall-clock time of day at which to
begin play.
     *   A value of TIME_CURRENT means start play immediately.
     * @exception IOException If an error has occurred.
     */
    public void play(
     int speed, long startPosition, long playDuration, long
startDate)
     throws IOException {
         XdrBlock call = session.newCall(PLAYER_PLAY);
         this.xdrout(call);
         call.xdroutInt(speed);
         call.xdroutMsmTime(startPosition);
         call.xdroutMsmTime(playDuration);
         call.xdroutMsmTime(startDate);
         session.rpc(call).done();
     }

    /**
     * Pauses play on the player.
     * Only one pause() command can be pending, a second pause()
     * overrides any pending pause().
     * @param pausePosition The position within the playlist at
which to pause
     *   playing.  If current play position is later than
pausePosition
     *   (taking into account the direction of play), play pauses
immediately.
     *   A value of TIME_CURRENT means stop immediately.
     * @return The time at which play actually paused.
     * @exception IOException If an error has occurred.
     */
    public long pause(long pausePosition) throws Exception {
     XdrBlock call = session.newCall(PLAYER_PAUSE);
     this.xdrout(call);
     call.xdroutMsmTime(pausePosition);
     XdrBlock reply = session.rpc(call);
     long result = reply.xdrinMsmTime();
     reply.done();
     return result;
    }

    /**
     * Resumes playing.  Play will continue until paused
     * or the end of the playlist (looped playlists play
forever).
     * @param speed The speed at which to resume play.
```

```
     * @param startPosition The position within the playlist at
which to
     *   resume play.  TIME_CURRENT means the current play
position.
     * @exception IOException If an error has occurred.
     */
    public void resume(int speed, long startPosition) throws
IOException {
     XdrBlock call = session.newCall(PLAYER_RESUME);
     this.xdrout(call);
     call.xdroutInt(speed);
     call.xdroutMsmTime(startPosition);
     session.rpc(call).done();
    }

    /**
     * Get play state for a player.
     * @return  A MsmPlayStatus instance.
     * @exception IOException If an error has occurred.
     */
    public MsmPlayStatus getPlayStatus() throws IOException {
     XdrBlock call = session.newCall(PLAYER_GETPLAYSTATUS);
     this.xdrout(call);
     XdrBlock reply = session.rpc(call);
     MsmPlayStatus result = new MsmPlayStatus(reply);
     reply.done();
     return result;
    }

    public String toString() {
     return MsmToString.playerToString(this);
    }

    private static final int HANDLELEN = 12;

    public static final long TIME_BADTIME =                    -1L;
    public static final long TIME_CURRENT =                    -2L;
    public static final long TIME_ZERO    =                     0L;
    public static final long TIME_MAXTIME = 2147483647999999999L;
    public static final long TIME_MINTIME =                     1L;

    public static final int SPEED_SCENE_REVERSE = 0;
    public static final int SPEED_FASTEST_REVERSE = 1;
    public static final int SPEED_FASTER_REVERSE = 2;
    public static final int SPEED_FAST_REVERSE = 3;
    public static final int SPEED_REVERSE = 4;
    public static final int SPEED_SLOW_REVERSE = 5;
```

```
        public static final int SPEED_SLOWER_REVERSE = 6;
        public static final int SPEED_SLOWEST_REVERSE = 7;
        public static final int SPEED_SLOWEST_FORWARD = 8;
        public static final int SPEED_SLOWER_FORWARD = 9;
        public static final int SPEED_SLOW_FORWARD = 10;
        public static final int SPEED_FORWARD = 11;
        public static final int SPEED_FAST_FORWARD = 12;
        public static final int SPEED_FASTER_FORWARD = 13;
        public static final int SPEED_FASTEST_FORWARD = 14;
        public static final int SPEED_SCENE_FORWARD = 15;

        private static final int PROG = 0x206d736d;
        private static final int VERS = 1;

        private static final int SERVER_AUTHTYPE       =  1;
        private static final int PLAYER_CREATE         =  2;
        private static final int PLAYER_DELETE         =  3;
        private static final int PLAYER_LIST           =  4;
        private static final int PLAYER_SETACCESS      =  5;
        private static final int PLAYER_GETACCESS      =  6;
        private static final int PLAYER_SETPERSISTENCE =  7;
        private static final int PLAYER_GETPERSISTENCE =  8;
        private static final int PLAYER_SETPLAYLIST    =  9;
        private static final int PLAYER_GETPLAYLIST    = 10;
        private static final int PLAYER_SETCONNECT     = 11;
        private static final int PLAYER_GETCONNECT     = 12;
        private static final int PLAYER_PLAY           = 13;
        private static final int PLAYER_PAUSE          = 14;
        private static final int PLAYER_RESUME         = 15;
        private static final int PLAYER_GETPLAYSTATUS  = 16;
        private static final int TITLE_GETSTATUS       = 17;
    }
```

MsmSession

```
/*
 * @(#)MsmSession.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.io.*;
import java.net.*;
import java.util.*;

/**
 * Media Stream Manager Client API
 *
 * The Media Stream Manager (msm) API provides an RPC interface
for managing
 * the scheduling and play of isochronous media streams.
 */
public class MsmSession {
    private String serverHostName;
    private Socket socket;
    private InputStream is;
    private OutputStream os;
    private int prog;
    private int vers;

    /**
     * Create a RPC session for the named server.
     * @param serverHostName The host name of a MSM server.
     * @exception IOException If an error has occurred.
     */
    public MsmSession(String serverHostName) throws IOException {
     this.serverHostName = serverHostName;
     socket = new Socket(serverHostName, pmapGetPort());
     is = new BufferedInputStream(socket.getInputStream());
     os = new BufferedOutputStream(socket.getOutputStream());
    }

    private int pmapGetPort() throws IOException {
```

```
        PortMapper pmap = null;
        try {
            pmap = new PortMapper(serverHostName);
            int port;
            prog = 100236;
            vers = 1;
            port = pmap.getPort(prog, vers, PortMapper.IPPROTO_TCP);
            if (port != 0) return port;
            prog = 0x206d736d;
            vers = 1;
            port = pmap.getPort(prog, vers, PortMapper.IPPROTO_TCP);
            if (port != 0) return port;
        } finally {
            if (pmap != null) pmap.close();
        }
        throw new MsmException("no msm server on "+serverHostName);
    }

    /**
     * Closes a session with an MSM server.
     * @exception MsmException If an error has occurred.
     */
    public void close() throws IOException {
        socket.close();
    }

    /**
     * All players on this server.
     * @return an array of all players.
     * @exception IOException If an error has occurred.
     */
    public MsmPlayer[] players() throws IOException {
        XdrBlock reply = rpc(newCall(PLAYER_LIST));
        MsmPlayer[] result = new MsmPlayer[reply.xdrinInt()];
        for (int i = 0; i < result.length; i++) {
            result[i] = new MsmPlayer(this, reply);
        }
        reply.done();
        return result;
    }

    /**
     * Obtains status about titles.
     * @param titleName The name of the title on which to obtain
status.
     * @return the status of the title.
     * @exception IOException If an error has occurred.
```

```
    */
   public MsmTitle getTitleStatus(String titleName) throws
IOException {
    XdrBlock call = newCall(TITLE_GETSTATUS);
    call.xdroutString(titleName);
    XdrBlock reply = rpc(call);
       MsmTitle result = new MsmTitle(reply);
    reply.done();
    return result;
   }

   /**
    * Returns the server host name.
    */
   public String getServerHostName() {
    return serverHostName;
   }

   XdrBlock newCall(int proc) {
    return new XdrBlock(prog, vers, proc);
   }

   synchronized XdrBlock rpc(XdrBlock call) throws IOException {
    call.send(os);
    XdrBlock reply = new XdrBlock(is);
    try {
        reply.xdrinReplyHeader(call.callXid());
    } catch (IOException e) {
        throw new MsmException(call.callProc(), e.getMessage());
    }
    int err = reply.xdrinInt();
    if (err != 0) throw new MsmException(call.callProc(), err);
    return reply;
   }

   public String toString() {
    return MsmToString.sessionToString(this);
   }

   private static final int SERVER_AUTHTYPE        =  1;
   private static final int PLAYER_CREATE          =  2;
   private static final int PLAYER_DELETE          =  3;
   private static final int PLAYER_LIST            =  4;
   private static final int PLAYER_SETACCESS       =  5;
   private static final int PLAYER_GETACCESS       =  6;
   private static final int PLAYER_SETPERSISTENCE  =  7;
   private static final int PLAYER_GETPERSISTENCE  =  8;
```

```
    private static final int PLAYER_SETPLAYLIST    =  9;
    private static final int PLAYER_GETPLAYLIST    = 10;
    private static final int PLAYER_SETCONNECT     = 11;
    private static final int PLAYER_GETCONNECT     = 12;
    private static final int PLAYER_PLAY           = 13;
    private static final int PLAYER_PAUSE          = 14;
    private static final int PLAYER_RESUME         = 15;
    private static final int PLAYER_GETPLAYSTATUS  = 16;
    private static final int TITLE_GETSTATUS       = 17;

}
```

MsmAccessRight

```
/*
 * @(#)MsmAccessRight.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

/**
 * Access types, operations on access lists, and rights and
 * lists of access rights.
 * Access types (read, admin, control) are the access catagories
 * defined by the MSM server (see MSM doc for each request to
 * determine the access catagory of that request).  Access op's
 * are the operations that can be made to alter access rights of
 * a particular user.  An access right is the pairing of access
 * catagories with a particular user.  An access list is a
collection
 * of access rights for multiple users.
 */
public class MsmAccessRight {
    public String name;
    public int access;
    public int op;

    public MsmAccessRight(String name, int access, int op) {
     this.name = name;
     this.access = access;
     this.op = op;
    }

    MsmAccessRight(XdrBlock xdr) {
     name = xdr.xdrinString();
     access = xdr.xdrinInt();
     op = xdr.xdrinInt();
    }

    void xdrout(XdrBlock xdr) {
     xdr.xdroutString(name);
     xdr.xdroutInt(access);
```

```
 xdr.xdroutInt(op);
}

public String toString() {
 return MsmToString.accessRightToString(this);
}

public static final int ACCESS_NONE = 0;
public static final int ACCESS_ADMIN = 1;
public static final int ACCESS_READ = 2;
public static final int ACCESS_CONTROL = 4;
public static final int ACCESS_ALL = 7;

public static final int OP_ADD = 0;
public static final int OP_REMOVE = 1;
}
```

MsmPersistence

```
/*
 * @(#)MsmPersistence.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

/**
 * MsmPersistence information
 */
public class MsmPersistence {
    /**
     * Indicates the date at which the player should be
     * automatically deleted.  On terminateDate, play if in
progress, will
     * be stopped and the player deleted.  A terminateDate of
MSMTIME_MAXTIME
     * indicates the player should never be automatically
deleted.
     */
    public long terminateDate;

    public int type;

    public MsmPersistence(int type, long terminateDate) {
     this.type = type;
     this.terminateDate = terminateDate;
    }

    MsmPersistence(XdrBlock xdr) {
     type = xdr.xdrinInt();
     terminateDate = xdr.xdrinMsmTime();
    }

    void xdrout(XdrBlock xdr) {
     xdr.xdroutInt(type);
     xdr.xdroutMsmTime(terminateDate);
    }
```

```
    public String toString() {
     return MsmToString.persistenceToString(this);
    }

    /**
     * No persistence across server outage.
     */
    public static final int TYPE_NONE = 0;
    /**
     * Only public static state is preserved, play not is not
restarted.
     */
    public static final int TYPE_PLAYLIST = 1;
    /**
     * Play is restarted after outage at last known playPosition.
     */
    public static final int TYPE_PLAYPOSITION = 2;
    /**
     * Play is restarted after outage as appropriate for current
date.
     */
    public static final int TYPE_PLAYCURDATE = 3;
}
```

MsmPlaylist

```
/*
 * @(#)MsmPlaylist.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

/**
 * MsmPlaylist positions are measured in seconds and nanoseconds,
titles on a
 * playlist may be scheduled to start at any non-negative
position. (In some
 * cases it may be convenient to base playlists positions at 0;
in other
 * cases it may be better to base them with the OS representation
of
 * time-of-day.) The playlist maintains a contiguous sequence of
titles and
 * "dead air". A schedule may be edited by replacing any
contiguous
 * sub-sequence of the schedule with another sequence.  It is
also possible
 * to change the starting position of the scheduled list of
titles.  Because
 * of mfs "admission delays", title start times may slip; msm
optionally
 * allows a title to be padded with dead air that can absorb the
slip, or on
 * a slip the same title or a later title can be marked to be
truncated or a
 * later title may be "joined-in-progress" to absorb the slip and
maintain
 * schedule correspondence with clock time.
 */
public class MsmPlaylist {
    /**
     * On Get, the current modification status stamp.  On Put,
modstamp on
     * which mods are based, if modification status has changed.
```

```
Mods are
     * aborted unless modstamp == MsmPlayer.TIME_CURRENT, in
which case mods
     * are always done.
     */
    public long modstamp;

    /**
     * On Get, the starting playlist position for the returned
playlist items
     * on Put, the playlist position where items are to be
replaced.
     */
    public long editStartPosition;

    /**
     * On Get, the total duration of the items returned.  On Put,
the duration
     * of the existing playlist that is to be replaced with new
items.
     *
     * NOTE: On Put, edit range specified by editStartPosition
for length
     * editDuration must lie entirely within existing playlist.
Use
     * MsmPlayer.getPlaylist() to get listStartPosition and
listDuration to
     * determine playlist bounds.
     */
    public long editDuration;

    /**
     * On Get, the startPosition for the entire playlist.  On
Put, the new
     * startPosition for the playlist after edits.
     */
    public long listStartPosition;

    /**
     * On Get, the duration of the entire list.  On Put, ignored.
     */
    public long listDuration;

    public MsmItem[] items;

    /**
     * On Get, the current loop state of the playlist.  On Put,
```

```
if TRUE, the
     * playlist wraps from end->start, start-end.
     */
    public boolean isLoop;

    public MsmPlaylist(long modstamp, boolean isLoop, long
editStartPosition,
                    long editDuration, MsmItem[] items,
                    long listStartPosition, long listDuration) {
     this.modstamp = modstamp;
     this.isLoop = isLoop;
     this.editStartPosition = editStartPosition;
     this.editDuration = editDuration;
     this.items = items;
     this.listStartPosition = listStartPosition;
     this.listDuration = listDuration;
    }

    MsmPlaylist(XdrBlock xdr) {
     modstamp = xdr.xdrinMsmTime();
     isLoop = xdr.xdrinBoolean();
     editStartPosition = xdr.xdrinMsmTime();
     editDuration = xdr.xdrinMsmTime();
     items = new MsmItem[xdr.xdrinInt()];
     for (int i = 0; i < items.length; i++) {
         int itemType = xdr.xdrinInt();
         switch (itemType) {
         case TITLE:
          items[i] = new MsmTitleItem(xdr);
          break;
         case DEADAIR:
          items[i] = new MsmDeadAirItem(xdr);
          break;
         }
     }
     listStartPosition = xdr.xdrinMsmTime();
     listDuration = xdr.xdrinMsmTime();
    }

    void xdrout(XdrBlock xdr) {
     xdr.xdroutMsmTime(modstamp);
     xdr.xdroutBoolean(isLoop);
     xdr.xdroutMsmTime(editStartPosition);
     xdr.xdroutMsmTime(editDuration);
     xdr.xdroutInt(items.length);
     for (int i = 0; i < items.length; i++) {
         if (items[i] instanceof MsmTitleItem) {
```

```
            xdr.xdroutInt(TITLE);
            ((MsmTitleItem)items[i]).xdrout(xdr);
          } else {
            xdr.xdroutInt(DEADAIR);
            ((MsmDeadAirItem)items[i]).xdrout(xdr);
          }
      }
      xdr.xdroutMsmTime(listStartPosition);
      xdr.xdroutMsmTime(listDuration);
     }

     public String toString() {
      return MsmToString.playlistToString(this);
     }

     private static final int TITLE   = 0;
     private static final int DEADAIR = 1;

}
```

MsmConnect

```
/*
 * @(#)MsmConnect.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

/**
 * Connection paramaters.
 * These parameters are passed directly to mfs_str_open().
 */

public class MsmConnect {
    /**
     * The transport independent address.
     **/
    public String destTiAddr;

    /**
     * The packet encapsulation specifier (eg. MPEG Transport, *
DSS, etc).
     */
    public String encap;

    /**
     * The bits/second network bandwidth to request.
     */
    public int rate;

    public MsmConnect(String destTiAddr, String encap, int rate)
{
     this.destTiAddr = destTiAddr;
     this.encap = encap;
     this.rate = rate;
    }

    MsmConnect(XdrBlock xdr) {
     destTiAddr = xdr.xdrinString();
     encap = xdr.xdrinString();
```

```
 rate = xdr.xdrinInt();
}

void xdrout(XdrBlock xdr) {
 xdr.xdroutString(destTiAddr);
 xdr.xdroutString(encap);
 xdr.xdroutInt(rate);
}

public String toString() {
 return MsmToString.connectToString(this);
}

}
```

MsmPlayStatus

```
/*
 * @(#)MsmPlayStatus.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

/**
 * MsmPlayStatus indicates the current state of the player.
 * STATE_WAIT indicates that a play command has been given, but
 * that startDate has not arrived.
 */
public class MsmPlayStatus {
    public long pausePosition;
    public long currentDate;
    public long currentPosition;
    public String info;
    public int currentState;
    public int currentSpeed;
    public boolean pausePending;

    MsmPlayStatus(XdrBlock xdr) {
     info = xdr.xdrinString();
     pausePending = xdr.xdrinBoolean();
     pausePosition = xdr.xdrinMsmTime();
     currentState = xdr.xdrinInt();
     currentSpeed = xdr.xdrinInt();
     currentDate = xdr.xdrinMsmTime();
     currentPosition = xdr.xdrinMsmTime();
    }

    public String toString() {
     return MsmToString.playStatusToString(this);
    }

    public static final int STATE_STOP = 0;
    public static final int STATE_WAIT = 1;
    public static final int STATE_PLAY = 2;

    }
```

MsmToString

```
/*
 * @(#)MsmToString.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.util.*;

class MsmToString {
    static String sessionToString(MsmSession se) {
     return "MsmSession"
         + "[serverHostName=" + se.getServerHostName()
         + "]";
    }

    static String playerToString(MsmPlayer pl) {
     byte[] h = pl.getHandle();
     StringBuffer sb = new StringBuffer(h.length*2);
     for (int i = 0; i < h.length; i++) {
         byte b = h[i];
         sb.append(Character.forDigit((b >> 4) & 0xf, 16));
         sb.append(Character.forDigit( b       & 0xf, 16));
     }
     return "MsmPlayer"
         + "[serverHostName=" +
pl.getSession().getServerHostName()
         + " handle=" + sb.toString()
         + "]";
    }

    private static final String[] rights =
{"admin","read","control"};

    private static final String[] ops = {"add","remove"};

    static String accessRightToString(MsmAccessRight ar) {
     StringBuffer sb = new StringBuffer();
     for (int i = 0; i < rights.length; i++) {
```

```
            if ((ar.access & (1 << i)) != 0) {
             if (sb.length() > 0) sb.append("|");
             sb.append(rights[i]);
            }
        }
        if (sb.length() == 0) sb.append("none");
        String op;
        if (ar.op >= 0 && ar.op < ops.length) op = ops[ar.op];
        else op = String.valueOf(ar.op);
        return "MsmAccessRight"
            + "[name=" + ar.name
               + " access=" + sb.toString()
               + " op=" + op
            + "]";
       }

       static String connectToString(MsmConnect co) {
        return "MsmConnect"
            + "[destTiAddr=\"" + co.destTiAddr +"\""
            + " encap=\"" + co.encap +"\""
               + " rate=" + co.rate
            + "]";
       }

       static String deadAirItemToString(MsmDeadAirItem dai) {
        return "MsmDeadAirItem"
            + "[itemDuration=" + dai.itemDuration
               + " joinInDuration=" + dai.joinInDuration
            + "]";
       }

       private static final String[] types = {
        "none","playlist","playposition","playcurdate"};

       static String persistenceToString(MsmPersistence pe) {
        String type;
        if (pe.type >= 0 && pe.type < types.length) type =
 types[pe.type];
        else type = String.valueOf(pe.type);
        return "MsmPersistence"
            + "[type=" + type
               + "
 terminateDate=\""+dateToString(pe.terminateDate)+"\""
            + "]";
       }

       static String dateToString(long date) {
```

```
     if (date == MsmPlayer.TIME_MAXTIME) return "never";
     else return new Date(date/1000000L).toString();
    }

    private static final String[] states =
{"stop","wait","play"};

    private static final String[] speeds = {
     "scene_reverse","fastest_reverse","faster_reverse","fast_rev
erse",
     "reverse","slow_reverse","slower_reverse","slowest_reverse",
     "slowest_forward","slower_forward","slow_forward","forward",
     "fast_forward","faster_forward","fastest_forward","scene_for
ward"};

    static String playStatusToString(MsmPlayStatus ps) {
     String state;
     if (ps.currentState >= 0 && ps.currentState < states.length)
{
         state = states[ps.currentState];
     } else state = String.valueOf(ps.currentState);
     String speed;
     if (ps.currentSpeed >= 0 && ps.currentSpeed < speeds.length)
{
         speed = speeds[ps.currentSpeed];
     } else speed = String.valueOf(ps.currentSpeed);
     return "MsmPlayStatus"
         + "[info=\"" + ps.info +"\""
            + " pausePending=" + ps.pausePending
            + " pausePosition=" + ps.pausePosition
            + " currentState=" + state
            + " currentSpeed=" + speed
            + " currentDate=\"" + dateToString(ps.currentDate) +
"\""
            + " currentPosition=" + ps.currentPosition
         + "]";
    }

    static String playlistToString(MsmPlaylist pl) {
     StringBuffer sb = new StringBuffer();
     if (pl.items != null) {
         for (int i = 0; i < pl.items.length; i++) {
          if (i != 0) sb.append(",");
          sb.append(pl.items[i].toString());
         }
     }
     return "MsmPlaylist"
```

```
            + "[modstamp=\"" + dateToString(pl.modstamp) + "\""
            + " isLoop=" + pl.isLoop
            + " editStartPosition=" + pl.editStartPosition
            + " editDuration=" + pl.editDuration
            + " items=[" + sb.toString() + "]"
            + " listStartPosition=" + pl.listStartPosition
            + " listDuration=" + pl.listDuration
            + "]";
    }

    static String titleToString(MsmTitle ti) {
     StringBuffer sb = new StringBuffer();
     if (ti.speedScale != null) {
         for (int i = 0; i < ti.speedScale.length; i++) {
          if (i != 0) sb.append(",");
          sb.append(ti.speedScale[i]);
         }
     }
     return "MsmTitle"
         + "[name=\"" + ti.name + "\""
         + " speedScale=[" + sb.toString() + "]"
         + " maxBitRate=" + ti.maxBitRate
         + " totalPlayDuration=" + ti.totalPlayDuration
         + " format=\"" + ti.format + "\""
         + "]";
    }

    static String titleItemToString(MsmTitleItem ti) {
     return "MsmTitleItem"
         + "[titleName=\"" + ti.titleName + "\""
            + " itemDuration=" + ti.itemDuration
            + " startOffset=" + ti.startOffset
            + " playDuration=" + ti.playDuration
            + " joinInDuration=" + ti.joinInDuration
         + " isTimeLocked=" + ti.isTimeLocked
         + " playClosestSpeed=" + ti.playClosestSpeed
            + " maxBitRate=" + ti.maxBitRate
         + "]";
    }

}
```

MsmItem

```
/*
 * @(#)MsmItem.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

public abstract class MsmItem {
    /**
     * The number of milliseconds allocated to this item.
     */
    public long itemDuration;

    /**
     * Time of initial play that may be sacrificed to absorb
previous schedule
     * slips.   Silently limited to itemDuration.   If
TIME_CURRENT,
     * itemDuration is used.
     */
    public long joinInDuration;
}
```

MsmTitleItem

```
/*
 * @(#)MsmTitleItem.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

/*
 * A playlist title item.
 */
public class MsmTitleItem extends MsmItem {
    /**
     * The number of milliseconds into title where play should
begin.  It is
     * illegal for this to be greater than the total play time of
the title.
     */
    public long startOffset;

    /**
     * The number of milliseconds of title to play within this
item.
     * Values less than itemDuration allow some pad for absorbing
admission
     * delays (and the play truncation that would occur), but
should admission
     * delay be zero, dead air would occur for the remainder of
the item. It
     * is illegal for playDuration to be greater than
itemDuration or for
     * playDuration + startOffset to be greater than the total
play time of
     * the title. If TIME_CURRENT, the min of itemDuration and
total play time
     * minus startOffset is used.
     */
    public long playDuration;

    /**
```

```
    * The file pathname for title.
    */
   public String titleName;

   /**
    * Ignored on MsmPlayer.setPlaylist.  Returns max bit rate of
title on
    * MsmPlayer.getPlaylist.
    */
   public int maxBitRate;

   /**
    * If true, terminate play after itemDuration seconds (even
if admission
    * delays have caused schedule to slip and title has not
completed).  If
    * false, always play itemDuration seconds of title, allow
schedule to
    * slip if necessary.
    */
   public boolean isTimeLocked;

   /**
    * If true, plays closest available speed in same direction
if requested
    * speed is not available.  Search for closest is proceeds
towards normal
    * presentation rate.  Play is skipped if normal presentation
rate in
    * direction is not available.  If false, play of title is
skipped if
    * appropriate speed is not available.
    */
   public boolean playClosestSpeed;

   public MsmTitleItem(String titleName, long itemDuration, long
startOffset,
                long playDuration, long joinInDuration,
                boolean isTimeLocked, boolean playClosestSpeed,
                int maxBitRate) {
     this.titleName = titleName;
     this.itemDuration = itemDuration;
     this.startOffset = startOffset;
     this.playDuration = playDuration;
     this.joinInDuration = joinInDuration;
     this.isTimeLocked = isTimeLocked;
     this.playClosestSpeed = playClosestSpeed;
```

```
        this.maxBitRate = maxBitRate;
       }

       MsmTitleItem(XdrBlock xdr) {
        titleName = xdr.xdrinString();
        itemDuration = xdr.xdrinMsmTime();
        startOffset = xdr.xdrinMsmTime();
        playDuration = xdr.xdrinMsmTime();
        joinInDuration = xdr.xdrinMsmTime();
        isTimeLocked = xdr.xdrinBoolean();
        playClosestSpeed = xdr.xdrinBoolean();
        maxBitRate = xdr.xdrinInt();
       }

       void xdrout(XdrBlock xdr) {
        xdr.xdroutString(titleName);
        xdr.xdroutMsmTime(itemDuration);
        xdr.xdroutMsmTime(startOffset);
        xdr.xdroutMsmTime(playDuration);
        xdr.xdroutMsmTime(joinInDuration);
        xdr.xdroutBoolean(isTimeLocked);
        xdr.xdroutBoolean(playClosestSpeed);
        xdr.xdroutInt(maxBitRate);
       }

       public String toString() {
        return MsmToString.titleItemToString(this);
       }

}
```

MsmDeadAirItem

```
/*
 * @(#)MsmDeadAirItem.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

public class MsmDeadAirItem extends MsmItem {
    public MsmDeadAirItem(long itemDuration, long joinInDuration)
{
     this.itemDuration = itemDuration;
     this.joinInDuration = joinInDuration;
    }

    MsmDeadAirItem(XdrBlock xdr) {
     itemDuration = xdr.xdrinMsmTime();
     joinInDuration = xdr.xdrinMsmTime();
    }

    void xdrout(XdrBlock xdr) {
     xdr.xdroutMsmTime(itemDuration);
     xdr.xdroutMsmTime(joinInDuration);
    }

    public String toString() {
     return MsmToString.deadAirItemToString(this);
    }

}
```

MsmException

```
/*
 * @(#)MsmException.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.io.*;

/**
 * Signals that an Media Stream Manager exception has occurred.
 */
public class MsmException extends IOException {
    /**
     * Constructs an MsmException with no detail message.
     * A detail message is a String that describes this
particular exception.
     */
    MsmException() {
     super();
    }

    /**
     * Constructs an MsmException with the specified detail
message.
     * A detail message is a String that describes this
particular exception.
     * @param s the detail message
     */
    MsmException(String s) {
     super(s);
    }

    MsmException(int proc, String msg) {
     super(((proc >= 0 && proc < procNames.length) ?
              procNames[proc] : Integer.toString(proc))
            + ": " +
            msg);
    }
```

```
MsmException(int proc, int err) {
 super(((proc >= 0 && proc < procNames.length) ?
        procNames[proc] : Integer.toString(proc))
       + ": " +
       ((err >= 0 && err < errNames.length) ?
        errNames[err] : Integer.toString(err)));
}

private static final String[] procNames = {
 "null",
 "server authtype",
 "player create",
 "player delete",
 "player list",
 "player access set",
 "player access get",
 "player persistence set",
 "player persistence get",
 "player playlist set",
 "player playlist get",
 "player connect set",
 "player connect get",
 "player play",
 "player pause",
 "player resume",
 "player play status",
 "title status",
};

private static final String[] errNames = {
 "success",              /*  0 */
 "failed",               /*  1 */
 "badarg",               /*  2 */
 "no mem",               /*  3 */
 "no netname",           /*  4 */
 "des auth failed",      /*  5 */
 "kerb auth failed",     /*  6 */
 "no such player",       /*  7 */
 "old modstamp",         /*  8 */
 "item overlap",         /*  9 */
 "bad speed",            /* 10 */
 "bad start date",       /* 11 */
 "not connected",        /* 12 */
 "bad pause position",   /* 13 */
 "play active",          /* 14 */
 "bad file name",    /* 15 */
 "bad mfs file",         /* 16 */
```

```
        "bad file type",    /* 17 */
        "info too long",    /* 18 */
        "auth failed",      /* 19 */
        "bad position",         /* 20 */
        "kerberos unsupported", /* 21 */
        "bad credentials",  /* 22 */
        "insufficient authorization", /* 23 */
        "bad access op",    /* 24 */
        "bad access type",  /* 25 */
        "bad persist type", /* 26 */
        "bad time arg",         /* 27 */
        "bad start position",   /* 28 */
        "bad duration",         /* 29 */
        "bad start offset", /* 30 */
        "bad edit start pos",   /* 31 */
        "bad edit duration",    /* 32 */
        "bad list start pos",   /* 33 */
        "bad item duration",    /* 34 */
        "bad join in duration", /* 35 */
        "bad play duration",    /* 36 */
        "bad item type",    /* 37 */
        "bad title type",   /* 38 */
        "no such file",         /* 39 */
        "bad lut file",         /* 40 */
        "bad mfs fs",       /* 41 */
        "toc syntax",       /* 42 */
        "toc eof",          /* 43 */
        "toc bad char",         /* 44 */
        "no normal speed",  /* 45 */
        "dup speeds",       /* 46 */
        "bad file len",         /* 47 */
        "toc incomplete",   /* 48 */
        "toc can't map",    /* 49 */
        "toc bad filesize", /* 50 */
        "toc bad index",    /* 51 */
        "too low connect rate", /* 52 */
       };

   }
```

XdrBlock

```
/*
 * @(#)XdrBlock.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.io.*;
import java.net.*;

/**
 * Used to manipulate ONC RPC calls and replies.
 */
class XdrBlock {
    byte[] buf;
    int ptr;

    /*
     * Create a new empty block.
     * @param size The size of the block.
     */
    public XdrBlock(int size) {
     buf = new byte[size];
    }

    /*
     * Create a new empty block.
     */
    public XdrBlock() {
     this(256);
    }

    /*
     * Create a new block and initialize it with a call header.
     * @param prog The RPC program number.
     * @param vers The RPC version number.
     * @param proc The RPC procedure number.
     * @return The xid generated.
     */
```

```
public XdrBlock(int prog, int vers, int proc) {
 this();
 xdroutCallHeader(prog, vers, proc);
}

/**
 * Create a new block and receive it from an InputStream.
 * @param is The InputStream from which to receive the block.
 * @exception IOException If an IO error has occurred.
 */
public XdrBlock(InputStream is) throws IOException {
 synchronized (is) {
     int hdr;
     do {
      hdr  = readByte(is) << 24;
      hdr |= readByte(is) << 16;
      hdr |= readByte(is) <<  8;
      hdr |= readByte(is)      ;
      int start;
      int count = hdr & 0x7fffffff;
      if (buf == null) {
          start = 0;
          buf = new byte[count];
      } else {
          start = buf.length;
          byte[] tmp = new byte[start + count];
          System.arraycopy(buf, 0, tmp, 0, start);
          buf = tmp;
      }
      while (count > 0) {
          int done = is.read(buf, start, count);
          if (done < 0) throw new IOException("end of file");
          start += done;
          count -= done;
      }
     } while ((hdr & 0x80000000) == 0);
 }
}

private int readByte(InputStream is) throws IOException {
 int result = is.read();
 if (result < 0) throw new IOException("end of file");
 return result;
}

/**
 * Send the block to an output stream.
```

```
     * @param is The OutputStream ro which to send the block.
     * @exception IOException If an IO error has occurred.
     */
    public synchronized void send(OutputStream os) throws
IOException {
     int hdr = ptr | 0x80000000;
     synchronized (os) {
         os.write((hdr >> 24) & 0xff);
         os.write((hdr >> 16) & 0xff);
         os.write((hdr >>  8) & 0xff);
         os.write((hdr      ) & 0xff);
         os.write(buf, 0, ptr);
         if (os instanceof BufferedOutputStream) {
          ((BufferedOutputStream)os).flush();
         }
     }
    }

    /**
     * Input a fixed-length array of bytes from the block.
     * @param len The lenght of the array.
     * @return The byte array.
     */
    public synchronized byte[] xdrinBytes(int len) {
     byte[] result = new byte[len];
     System.arraycopy(buf, ptr, result, 0, len);
     ptr = (ptr + len + 3) & -4;
     return result;
    }

    /**
     * Input a variable-length array of bytes from the block.
     * @return The byte array.
     */
    public synchronized byte[] xdrinBytes() {
     return xdrinBytes(xdrinInt());
    }

    /**
     * Input an int from the block.
     * @return The int.
     */
    public synchronized int xdrinInt() {
     int result;
     result  = (buf[ptr    ] & 0xff) << 24;
     result |= (buf[ptr + 1] & 0xff) << 16;
     result |= (buf[ptr + 2] & 0xff) <<  8;
```

```
 result |= (buf[ptr + 3] & 0xff);
 ptr += 4;
 return result;
}

/**
 * Input an boolean from the block.
 * @return The boolean.
 */
public boolean xdrinBoolean() {
 return xdrinInt() != 0;
}

/**
 * Input a String from the block.
 * @return The String.
 */
public String xdrinString() {
 return new String(xdrinBytes(), 0);
}

/**
 * Input a Media Stream Manager Time value
 */
public synchronized long xdrinMsmTime() {
 long sec = xdrinInt();
 long nsec = xdrinInt();
 if (sec == nsec && sec < 0) return sec;
 return sec*1000000000L + nsec;
}

/**
 * Output a fixed-length array of bytes to the block.
 * @param val The array to output.
 * @param len The length of the array to output.
 */
public synchronized void xdroutBytes(byte[] val, int len) {
 int nxt = (ptr + len + 3) & -4;
 if (nxt > buf.length) grow(nxt);
 System.arraycopy(val, 0, buf, ptr, len);
 ptr = nxt;
}

/**
 * Output a variable-length array of bytes to the block.
 * @param val The array to output.
 */
```

```
public synchronized void xdroutBytes(byte[] val) {
 int len = val.length;
 xdroutInt(len);
 xdroutBytes(val, len);
}

/**
 * Output an int to the block.
 * @param val The int to output.
 */
public synchronized void xdroutInt(int val) {
 int nxt = ptr + 4;
 if (nxt > buf.length) grow(nxt);
 buf[ptr    ] = (byte)((val >> 24) & 0xff);
 buf[ptr + 1] = (byte)((val >> 16) & 0xff);
 buf[ptr + 2] = (byte)((val >>  8) & 0xff);
 buf[ptr + 3] = (byte)((val      ) & 0xff);
 ptr = nxt;
}

/**
 * Output an boolean to the block.
 * @param val The boolean to output.
 */
public void xdroutBoolean(boolean val) {
 xdroutInt(val? 1:0);
}

/**
 * Output a String to the block.
 * @param val The String to output.
 */
public void xdroutString(String val) {
 int len = val.length();
 byte[] tmp = new byte[len];
 val.getBytes(0, len, tmp, 0);
 xdroutBytes(tmp);
}

/**
 * Output a Media Stream Manager Time value
 * @param val The time to output.
 */
public synchronized void xdroutMsmTime(long val) {
 if (val < 0) {
     xdroutInt((int)val);
     xdroutInt((int)val);
```

```
        } else {
            xdroutInt((int)(val/1000000000L));
            xdroutInt((int)(val%1000000000L));
        }
    }

    private void grow(int needed) {
        int len = buf.length*2;
        while (len < needed) len *= 2;
        byte[] tmp = new byte[len];
        System.arraycopy(buf, 0, tmp, 0, buf.length);
        buf = tmp;
    }

    /**
     * Output a RPC Call header to the block.
     * @param prog The RPC program number.
     * @param vers The RPC version number.
     * @param proc The RPC procedure number.
     */
    public synchronized void xdroutCallHeader(int prog, int vers,
int proc) {
        xdroutInt(genXid());
        xdroutInt(CALL);
        xdroutInt(RPCVERS);
        xdroutInt(prog);
        xdroutInt(vers);
        xdroutInt(proc);
        xdroutInt(AUTH_UNIX);
        xdroutBytes(cred());
        xdroutInt(AUTH_NULL);
        xdroutBytes(verf());
    }

    public synchronized int callXid() {
        int tmp = ptr;
        ptr = 0;
        int result = xdrinInt();
        ptr = tmp;
        return result;
    }

    public synchronized int callProc() {
        int tmp = ptr;
        ptr = 20;
        int result = xdrinInt();
        ptr = tmp;
```

```
 return result;
}

private static int lastXid = 0;

private synchronized static int genXid() {
 if (lastXid != 0) lastXid += 1;
 else lastXid = (int)(Math.random() * 2147483648.0D);
 return lastXid;
}

private static byte[] lastCred;

private synchronized static byte[] cred() {
 if (lastCred == null) {
     XdrBlock xdr = new XdrBlock();
     xdr.xdroutInt((int)(System.currentTimeMillis()/1000L));
     String host;
     try host = InetAddress.getLocalHost().getHostName();
     catch (UnknownHostException e) host = "???";
     xdr.xdroutString(host);
     int uid;
     try uid =
Integer.parseInt(System.getProperty("user.uid"));
     catch (NumberFormatException e) uid = 0;
     xdr.xdroutInt(uid);
     int gid;
     try gid =
Integer.parseInt(System.getProperty("user.gid"));
     catch (NumberFormatException e) gid = 0;
     xdr.xdroutInt(gid);
     xdr.xdroutInt(0);     // no gids
     lastCred = new byte[xdr.ptr];
     System.arraycopy(xdr.buf, 0, lastCred, 0, xdr.ptr);
 }
 return lastCred;
}

private static byte[] lastVerf;

private synchronized static byte[] verf() {
 if (lastVerf == null) {
     lastVerf = new byte[0];
 }
 return lastVerf;
}
```

```
    /**
     * Input a RPC reply header from the block.
     * @param xid The expected xid.
     * @exception IOException If an error has occurred.
     */
    public synchronized void xdrinReplyHeader(int xid) throws
IOException {
     int replyXid = xdrinInt();
     if (replyXid != xid) {
         throw new IOException(
          "rpc xid mismatch: " +
          "expected " + xid + " but got " + replyXid);
     }
     int msgType = xdrinInt();
     if (msgType != REPLY) {
         throw new IOException(
          "rpc msg type mismatch: " +
          " expected " + REPLY + " but got " + msgType);
     }
     int replyStat = xdrinInt();
     switch (replyStat) {
     case MSG_ACCEPTED:
         int verfType = xdrinInt();
         byte[] verf = xdrinBytes();
         int acceptStat = xdrinInt();
         switch (acceptStat) {
            case SUCCESS:
          return;
         case PROG_UNAVAIL:
          throw new IOException(
              "rpc accepted: " +
              "remote hasn't exported program");
         case PROG_MISMATCH:
          int low = xdrinInt();
          int high = xdrinInt();
          throw new IOException(
              "rpc accepted: " +
              "version mismatch low=" + low + " high=" + high);
         case PROC_UNAVAIL:
          throw new IOException(
              "rpc accepted: " +
              "program can't support procedure");
         case GARBAGE_ARGS:
          throw new IOException(
              "rpc accepted: " +
              "procedure can't decode params");
         default:
```

```
            throw new IOException(
                "rpc accepted: " +
                "unknown status: " + acceptStat);
           }
        case MSG_DENIED:
            int rejectStat = xdrinInt();
            switch (rejectStat) {
            case RPC_MISMATCH:
             int low = xdrinInt();
             int high = xdrinInt();
             throw new IOException(
                 "rpc rejected: " +
                 "version mismatch low=" + low + " high=" + high);
            case AUTH_ERROR:
             int authStat = xdrinInt();
             switch (authStat) {
             case AUTH_BADCRED:
                 throw new IOException(
                  "rpc rejected: " +
                  "remote can't authenticate caller: " +
                  "bad credentials (seal broken)");
             case AUTH_REJECTEDCRED:
                 throw new IOException(
                  "rpc rejected: " +
                  "remote can't authenticate caller: " +
                  "client must begin new session");
             case AUTH_BADVERF:
                 throw new IOException(
                  "rpc rejected: " +
                  "remote can't authenticate caller: " +
                  "bad verifier (seal broken)");
             case AUTH_REJECTEDVERF:
                 throw new IOException(
                  "rpc rejected: " +
                  "remote can't authenticate caller: " +
                  "verifier expired or replayed");
             case AUTH_TOOWEAK:
                 throw new IOException(
                  "rpc rejected: " +
                  "remote can't authenticate caller: " +
                  "rejected for security reasons");
             default:
                 throw new IOException(
                  "rpc rejected: " +
                  "remote can't authenticate caller: " +
                  "unknown status: " + authStat);
             }
```

```
            default:
             throw new IOException(
                 "rpc rejected: " +
                 "unknown status: " + rejectStat);
            }
        default:
            throw new IOException("unknown rpc reply status: " +
replyStat);
        }
      }

      /*
       * Blow up if ptr hasn't reached the end of the block.
       */
      public void done() throws IOException {
       if (ptr != buf.length) {
           throw new IOException(
            (buf.length-ptr) + " extra bytes of data remaining in
reply");
       }
      }

      /*
       * Provisions for authentication of caller to service and
vice-versa are
       * provided as a part of the RPC protocol.  The call message
has two
       * authentication fields, the credentials and verifier.  The
reply
       * message has one authentication field, the response
verifier.  The RPC
       * protocol specification defines all three fields to be the
following
       * opaque type (in the eXternal Data Representation (XDR)
language [9]):
       */
      private static final int AUTH_NULL       = 0;
      private static final int AUTH_UNIX       = 1;
      private static final int AUTH_SHORT      = 2;
      private static final int AUTH_DES        = 3;

      /*
       * RPC Message protocol version 2
       */
      private static final int RPCVERS = 2;
      private static final int CALL    = 0;
      private static final int REPLY   = 1;
```

```
    /*
     * A reply to a call message can take on two forms: The
message was
     * either accepted or rejected.
     */
    private static final int MSG_ACCEPTED = 0;
    private static final int MSG_DENIED   = 1;

    /*
     * Given that a call message was accepted, the following is
the status
     * of an attempt to call a remote procedure.
     */
    private static final int SUCCESS       = 0;
    private static final int PROG_UNAVAIL  = 1;
    private static final int PROG_MISMATCH = 2;
    private static final int PROC_UNAVAIL  = 3;
    private static final int GARBAGE_ARGS  = 4;

    /*
     * Reasons why a call message was rejected:
     */
    private static final int RPC_MISMATCH = 0;
    private static final int AUTH_ERROR = 1;

    /*
     * Why authentication failed:
     */
    private static final int AUTH_BADCRED      = 1;
    private static final int AUTH_REJECTEDCRED = 2;
    private static final int AUTH_BADVERF      = 3;
    private static final int AUTH_REJECTEDVERF = 4;
    private static final int AUTH_TOOWEAK      = 5;

}
```

PortMapper

```
/*
 * @(#)PortMapper.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.io.*;
import java.net.*;

/**
 * Interface to the ONC port mapper.
 */
class PortMapper {
    private Socket socket;
    private InputStream is;
    private OutputStream os;

    /**
     * Create a port mapper client.
     * @param host The server for which we want to know the port
mappings.
     * @exception IOException If there is an error.
     */
    public PortMapper(String host) throws IOException {
     socket = new Socket(host, PMAP_PORT);
     is = new BufferedInputStream(socket.getInputStream());
     os = new BufferedOutputStream(socket.getOutputStream());
    }

    /**
     * Get the port number for a particular ONC service.
     * @param prog The RPC program number.
     * @param vers The RPC version number.
     * @param prot Either IPPROTO_TCP or IPPROTO_UDP.
     * @return The port number for the service.
     * @exception IOException If there is an error.
     */
    public synchronized int getPort(int prog, int vers, int prot)
```

```
        throws IOException {
            XdrBlock call = new XdrBlock();
            call.xdroutCallHeader(PMAP_PROG, PMAP_VERS,
PMAPPROC_GETPORT);
            call.xdroutInt(prog);
            call.xdroutInt(vers);
            call.xdroutInt(prot);
            call.xdroutInt(0);
            call.send(os);
            XdrBlock reply = new XdrBlock(is);
            reply.xdrinReplyHeader(call.callXid());
            int result = reply.xdrinInt();
            reply.done();
            return result;
        }

    /**
     * Closes the port mapper.
     */
    public synchronized void close() throws IOException {
     socket.close();
    }

    static final int IPPROTO_TCP = 6;
    static final int IPPROTO_UDP = 17;

    private static final int PMAP_PROG = 100000;
    private static final int PMAP_VERS = 2;
    private static final int PMAP_PORT = 111;

    private static final int PMAPPROC_NULL    = 0;
    private static final int PMAPPROC_SET     = 1;
    private static final int PMAPPROC_UNSET   = 2;
    private static final int PMAPPROC_GETPORT = 3;
    private static final int PMAPPROC_DUMP    = 4;
    private static final int PMAPPROC_CALLIT  = 5;

}
```

Decoder

```
/*
 * @(#)Decoder.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.awt.*;
import java.io.*;

public class Decoder extends Panel {
    private DecoderImpl impl;

    public Decoder() {
     setLayout(new BorderLayout());
    }

    public synchronized void init(String format, Image img,String
host,int port,String ATM)
     throws IOException {
         try {
          Class implClass = Class.forName(implClassName(format));
          if (impl == null || impl.getClass() != implClass) {
              removeAll();
              impl = (DecoderImpl)implClass.newInstance();
              add("Center", impl);
          }
          impl.init(format, img, host, port,ATM);
         } catch (ClassNotFoundException e) {
          throw new IOException(e.toString());
         } catch (IllegalAccessException e) {
          throw new IOException(e.toString());
         } catch (InstantiationException e) {
          throw new IOException(e.toString());
         }
     }

    public synchronized void paint(Graphics g) {
     if (impl != null) super.paint(g);
```

```
        else {
            Rectangle b = bounds();
            g.setColor(getBackground());
            g.fill3DRect(0, 0, b.width, b.height, true);
        }
    }

    public synchronized void stop() throws IOException {
        if (impl != null) impl.stop();
    }

    public synchronized void pause() throws IOException {
        if (impl != null) impl.pause();
    }

    public synchronized void play() throws IOException {
        if (impl != null) impl.play();
    }

    public synchronized void flush() throws IOException {
        if (impl != null) impl.flush();
    }

    public synchronized String destTiAddr() throws IOException {
        if (impl != null) return impl.destTiAddr();
        return "";
    }

    public synchronized String encap() throws IOException {
        if (impl != null) return impl.encap();
        return "";
    }

    /**
     * A hacky implementation factory
     */
    private static String implClassName(String format) throws
IOException {
        String osArch = System.getProperty("os.arch", "?os.arch");
        String osName = System.getProperty("os.name", "?os.name");
        String osVersion = System.getProperty("os.version",
"?os.version");
        String spec = format + " " + osArch + " " + osName + " " +
osVersion;
        if (format.equals("MPEG1SYS")) {
            if (osName.equals("Solaris") || osName.equals("SunOS"))
{
```

```
            if (osArch.equals("sparc")) {
                return "COM.Sun.isg.smcjc.MpxDecoderImpl";
            }
          }
      }
      throw new IOException("no decoder for " + spec);
    }
}
```

DecoderImpl

```
/*
 * @(#)DecoderImpl.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.awt.*;
import java.io.*;

abstract class DecoderImpl extends Canvas {
    public abstract void init(String format, Image img, String
host, int port,String ATM) throws IOException;
    public abstract void stop() throws IOException;
    public abstract void pause() throws IOException;
    public abstract void play() throws IOException;
    public abstract void flush() throws IOException;
    public abstract String destTiAddr() throws IOException;
    public abstract String encap() throws IOException;
}
```

MpxDecoderImpl

```java
/*
 * @(#)MpxDecoderImpl.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

import java.applet.*;
import java.io.*;
import java.awt.*;
import java.net.*;

class MpxDecoderImpl extends DecoderImpl implements Runnable {
    private String format;
    private String host;
    private int port;
    private int port0;
    private Image img;
    private long fadeTimeMillis;
    private DatagramSocket ctrlSckt;
    private Thread thread;
    private DatagramPacket ctrlPckt;
    private File logFile;
    private float luminance = 1.0F;
    private int dataPort;
    private int scale = 1;
    private int state=STOP;
    private boolean multi=false;
    private boolean ATM=false;
    private String ATMs=null;

    public MpxDecoderImpl() {
     super();
    }

    public synchronized void init(String format, Image img,
String host, int port,String ATMs)
     throws IOException {
         this.format = format;
```

```
            this.img = img;
            ATM=(ATMs!=null);
               this.port=port;
               this.host=host;
               if ((port==-1)&&(!ATM)){
                dataPort = genLocalPort();
               }else{
                dataPort = port;
                port0= genLocalPort();
                   multi=!ATM;
                   if (ATM) this.ATMs = ATMs;
               }
            ctrlPckt = new DatagramPacket(
             new
byte[128],128,InetAddress.getLocalHost(),genLocalPort());
            ctrlWord(0, 0x00000001); // sync
            ctrlWord(1, 0x00000002); // sync
            ctrlWord(2, 0x00000003); // sync
            ctrlWord(3, 0x00000004); // sync
            ctrlWord(4, 0xaaaa0001); // version = 1
            ctrlWord(5, 0xbbbb0001); // channel = 1
            ctrlWord(6, 0x00000000); // sequence = 0
            ctrlWord(7,     0xcccc0000); // flags = 0
            ctrlWord(8,     0xdddd0001); // type = 1
     }

    public Dimension minimumSize() {
     return new Dimension(WIDTH, HEIGHT);
    }

    public synchronized Dimension preferredSize() {
     Dimension dim = new Dimension(WIDTH*scale, HEIGHT*scale);
     return dim;
    }

    public synchronized void layout() {
     Rectangle b = bounds();
     double xscale = (double)b.width/(double)WIDTH;
     double yscale = (double)b.height/(double)HEIGHT;
     int scale = (int)((xscale + yscale) / 2.0 + 0.25);
     if (scale < 1) scale = 1;
     if (scale > 3) scale = 3;
     if (scale != this.scale) {
         this.scale = scale;
         if (state == PAUSE || state == PLAY) updateVideoMode();
     }
    }
```

```
    public synchronized void paint(Graphics g) {
     Dimension ps = preferredSize();
     g.setColor(getBackground());
     g.fill3DRect(0, 0, ps.width, ps.height, true);
     if (img != null) g.drawImage(img, 0, 0, ps.width, ps.height,
this);
    }

    public synchronized void stop() throws IOException {
     if (state == PAUSE || state == PLAY) {

           if (multi||ATM){
             StringBuffer sc= new StringBuffer();
             sc.append("kloop ");
System.out.println(sc.toString());
             String[] cmdarray0= new String[3];
             cmdarray0[0] = "/bin/sh";
             cmdarray0[1] = "-c";
             cmdarray0[2] = sc.toString();
             try Runtime.getRuntime().exec(cmdarray0);
             catch (SecurityException e)
System.out.println("Exec="+exec(cmdarray0[2]));
           }
         ctrlWord(9,     MCMD_EXIT);
         ctrlSckt.send(ctrlPckt);
         ctrlSckt.close();
         ctrlSckt = null;
         state = STOP;
         try {
          if (logFile.length() == 0) logFile.delete();
         } catch (SecurityException e) {
          String cmd = "/bin/rm -f "+logFile.getPath();
          try Runtime.getRuntime().exec(cmd);
          catch (SecurityException f) exec(cmd);
         }
     }
    }

    public synchronized void pause() throws IOException {
     if (state == PLAY) {
         ctrlWord(9,      MCMD_PLAYCTR); // identifier
         ctrlWord(10, PC_PAUSE); // action
         ctrlWord(11, Float.floatToIntBits(1.0F)); // speed
         ctrlSckt.send(ctrlPckt);
         state = PAUSE;
     }
```

```
}

    public synchronized void play() throws IOException {
     if (state == PAUSE) {
         ctrlWord(9,      MCMD_PLAYCTR); // identifier
         ctrlWord(10, PC_PLAY);    // action
         ctrlWord(11, Float.floatToIntBits(1.0F)); // speed
         ctrlSckt.send(ctrlPckt);
         state = PLAY;
     } else if (state == STOP) {
         StringBuffer sb = new StringBuffer();
         sb.append("exec mpx");
         if (!multi) {
               if (!ATM){
                 sb.append(" -fn udp,lp,");
              sb.append(dataPort);
               }else{
                 sb.append(" -fn udp,lp,");
              sb.append(port0);
               }
            }else{
               sb.append(" -fn udp,lp,");
            sb.append(port0);
            }
         sb.append(" -xn udp,lp,");
         sb.append(ctrlPckt.getPort());
         sb.append(" -u 2");
         sb.append(" -v ");
         int depth = getColorModel().getPixelSize();
         if (depth == 1) {
          sb.append("mono");
         } else {
          sb.append("col");
          sb.append(depth);
          if (depth == 24 && scale > 1) sb.append("B");
         }
         sb.append(",");
         sb.append(scale);
         sb.append(" -w ");
         sb.append(windowId());
         sb.append(" </dev/null");
         sb.append(" >");
System.out.println(sb.toString());
         logFile = new
File("/tmp/mpx."+System.currentTimeMillis());
         sb.append(logFile.getPath());
         sb.append(" 2>&1");
```

```
            String[] cmdarray = new String[3];
            cmdarray[0] = "/bin/sh";
            cmdarray[1] = "-c";
            cmdarray[2] = sb.toString();
            try Runtime.getRuntime().exec(cmdarray);
            catch (SecurityException e) exec(cmdarray[2]);
            ctrlSckt = new DatagramSocket();
            state = PLAY;
                if(ATM){
                  StringBuffer sc= new StringBuffer();
                  sc.append("loop a ");
                  sc.append(dataPort+" ");
                  sc.append(port0+" >sasa &");
System.out.println(sc.toString());
                String[] cmdarray0= new String[3];
                cmdarray0[0] = "/bin/sh";
                cmdarray0[1] = "-c";
                cmdarray0[2] = sc.toString();
                try Runtime.getRuntime().exec(cmdarray0);
                catch (SecurityException e)
System.out.println("Exec="+exec(cmdarray0[2]));
                 }else if (multi) {
                  StringBuffer sc= new StringBuffer();
                  sc.append("loop m ");
                  sc.append(host+" ");
                  sc.append(dataPort+" ");
                  sc.append(port0+" &");
System.out.println(sc.toString());
                String[] cmdarray0= new String[3];
                cmdarray0[0] = "/bin/sh";
                cmdarray0[1] = "-c";
                cmdarray0[2] = sc.toString();
                try Runtime.getRuntime().exec(cmdarray0);
                catch (SecurityException e)
System.out.println("Exec="+exec(cmdarray0[2]));
                 }
        }
       }

       public synchronized void flush() {
        if (thread == null) {
            thread = new Thread(this);
            thread.start();
        }
        fadeTimeMillis = System.currentTimeMillis() + 4000;
       }
```

```
    public synchronized String destTiAddr() throws
UnknownHostException {
        String phost;
     //return "be0,"+phost+","+dataPort;
     if (ATM){
          return "port=" + ATMs + ",vc=" + dataPort;
        }else {
       phost = InetAddress.getLocalHost().getHostName();
          return "host=" + phost + ",udpport=" + dataPort;
        }
    }

    public String encap() {
     return "MPEG1SYS";
    }

    private void ctrlWord(int idx, int val) {
     byte[] buf = ctrlPckt.getData();
     buf[idx*4    ] = (byte)((val >> 24) & 0xff);
     buf[idx*4 + 1] = (byte)((val >> 16) & 0xff);
     buf[idx*4 + 2] = (byte)((val >>  8) & 0xff);
     buf[idx*4 + 3] = (byte)((val      ) & 0xff);
    }

    private void updateVideoMode() {
     ctrlWord(9,  MCMD_PRESCTR); // identifier
     ctrlWord(10, PCTR_VMD|PCTR_LUM); // which
     int depth = getColorModel().getPixelSize();
     int col = (depth==1)? 0 : (depth==24&&scale>1) ? VDM_COLB :
VDM_COL;
     ctrlWord(11, (col<<8)|scale); // video mode
     ctrlWord(12, 0);           // audio mode
     ctrlWord(13, 0);           // audio_volume
     ctrlWord(14, Float.floatToIntBits(luminance)); // luminance
     ctrlWord(15, 0);           // saturation
     ctrlWord(16, 0);           // gamma
     try ctrlSckt.send(ctrlPckt); catch (IOException e);
    }

    public synchronized void run() {
     Thread currentThread = Thread.currentThread();
     try {
         while (currentThread==thread && (state==PAUSE ||
state==PLAY)) {
          long currentTimeMillis = System.currentTimeMillis();
          float last = luminance;
          if (fadeTimeMillis < currentTimeMillis) {
```

```
                if (luminance < 1.0F) luminance += 0.125F;
            } else {
                if (luminance > 0.0F) luminance -= 0.125F;
            }
            if (luminance != last) updateVideoMode();
            if (luminance >= 1.0F) return;
            try wait(125); catch (InterruptedException e);
           }
       } finally {
           if (thread == currentThread) thread = null;
       }
      }

      private int genLocalPort() throws IOException {
       DatagramSocket sckt = new DatagramSocket();
       int port = sckt.getLocalPort();
       sckt.close();
       return port;
      }

      private native int windowId();

      private native int exec(String cmd);

      protected void finalize() {
       try stop(); catch (IOException e);
      }

      private static final int WIDTH  = 352;
      private static final int HEIGHT = 240;

      private static final int STOP  = 0;
      private static final int PLAY  = 1;
      private static final int PAUSE = 2;

      /* command identifiers */
      private static final int MCMD_NULL        = 0;
      private static final int MCMD_EXIT        = 1;
      private static final int MCMD_OPENSRC     = 2;
      private static final int MCMD_CLOSESRC    = 3;
      private static final int MCMD_REENTER     = 4;
      private static final int MCMD_PLAYCTR     = 5;
      private static final int MCMD_PRESCTR     = 6;
      private static final int MCMD_STREAM      = 7;
      private static final int MCMD_SENDSTAT    = 8;
      private static final int MCMD_STATUS      = 9;
      private static final int MCMD_ACK         = 10;
```

```
/* command flags */
private static final int MCFL_SNDACK     = (1<<0);
private static final int MCFL_ORGMPX     = (1<<2);

/* command parameter values: */

/* source_type  :    MCMD_OPENSRC */
private static final int MSC_FNAME       = 1;
private static final int MSC_FDSCP       = 4;

/* flags   :    MCMD_REENTER */
private static final int MRE_FOFS        = (1<<0);
private static final int MRE_ASOPEN      = (1<<2);
private static final int MRE_STRMS       = (1<<3);
private static final int MRE_SEEKVSEQ    = (1<<4);

/* data_type    :    MCMD_OPENSRC, MCMD_REENTER */
private static final int BSTRM_11172     = (1<<0);
private static final int BSTRM_VSEQ      = (1<<1);
private static final int BSTRM_ASEQ      = (1<<2);

/* action       :    MCMD_PLAYCTR */
private static final int PC_PLAY         = (1<<0);
private static final int PC_FWDSPEED     = (1<<1);
private static final int PC_FWDSTEP      = (1<<2);
private static final int PC_PAUSE        = (1<<3);

/* which        :    MCMD_PRESCTR */
private static final int PCTR_VMD        = (1<<0);
private static final int PCTR_AMD        = (1<<1);
private static final int PCTR_AVOL       = (1<<2);
private static final int PCTR_LUM        = (1<<3);
private static final int PCTR_SAT        = (1<<4);
private static final int PCTR_GAM        = (1<<5);

/* video_mode   :    MCMD_PRESCTR
 * 0xvvzz
 *  vv : VDM_COL, VDM_COLB
 *  zz : zoom [1-3]
 */
private static final int VDM_COL         = 1;
private static final int VDM_COLB        = 2;

/* audio_mode   :    MCMD_PRESCTR
 *
 *    cccqqq
```

```
 *   ccc: channel listening selection
 *    Sxx : 1/0 -> Selection/ No Selection
 *    101 : Left
 *    110 : Right
 *    111 : Left & Right
 *   qqq: audio playback quality selection
 *    Sxx : 1/0 -> Selection/ No Selection
 *    100 : High
 *    101 : Medium
 *    110 : Low
 */

/* stream       :    MCMD_STREAM, MCMD_OPENSRC, MCMD_REENTER

 *   vvvvvvvv.aaaaaaaa
 *   aaaaaaaa:
 *     a7: 1-> ignore stream identifier part (bits a5-a0).
 *     a6: audio stream subscription 0/ON, 1/OFF
 *     a5: 1->auto subscribe to first encountered audio
stream,
 *        (a4-a0 = 00000).
 *   a4-a0: subscribe to a particular audio stream [0-31]
 *
 *   vvvvvvvv:
 *     v7: 1-> ignore stream identifier part, bits v5-v0
 *     v6: video stream subscription 0/ON, 1/OFF
 *     v5: 1->auto subscribe to first encountered video
stream,
 *        (v4-v0 = 00000).
 *     v4: 0
 *   v3-v0: subscribe to particular video stream [0-15]
 *
 */

    private static final int STRM_IGNOREID   = 0x80;
    private static final int STRM_SBCOFF     = 0x40;
    private static final int STRM_AUTOSBC    = 0x20;

    static {
     try System.loadLibrary("javampx"); catch
(UnsatisfiedLinkError e)
         System.load("/opt/SUNWsmcjc/lib/libjavampx.so");
    }
}
```

smcrm

```
/*
 * @(#)smcrm.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

public class smcrm {
    private static byte[] parseHandle(String s) {
      int len = s.length()/2;
      byte[] h = new byte[len];
      for (int i = 0; i < len; i++) {
          h[i] = (byte) Integer.parseInt(s.substring(i*2,
(i+1)*2), 16);
      }
      return h;
    }
    public static void main (String args[]) throws Exception {
      MsmSession session = null;
      MsmPlayer player;
      if (args.length != 2) {
          System.err.println("usage: smcrm <serverName>
<playerHandle>");
          return;
      }
      try {
          session = new MsmSession(args[0]);
          player = new MsmPlayer(session, parseHandle(args[1]));
          player.delete();
      } catch (Exception e) {
          System.err.println("smcrm: " + e);
      } finally {
          if (session != null) {
           try session.close(); catch (Exception e)
               System.err.println("smcrm: " + e);
          }
      }
    }
}
```

smcstat

```
/*
 * @(#)smcstat.java
 *
 * Copyright 1995 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version     1.0
 * author Christopher Lindblad
 *
 */

package COM.Sun.isg.smcjc;

public class smcstat {
    public static void main (String args[]) throws Exception {
     MsmSession session = null;
     MsmPlayer[] players;
     if (args.length != 1) {
         System.err.println("usage: smcstat <serverName>");
         return;
     }
     try {
         session = new MsmSession(args[0]);
         players = session.players();
         System.out.println(session);
         for (int i = 0; i < players.length; i++) {
          MsmPlayer player = players[i];
          MsmPersistence persistence = player.getPersistence();
          MsmConnect connect = player.getConnect();
          MsmPlayStatus status = player.getPlayStatus();
          MsmAccessRight[] rights = player.getAccess();
          MsmPlaylist playlist = player.getPlaylist();
          System.out.println(player);
          System.out.println(persistence);
          System.out.println(connect);
          System.out.println(status);
          for (int j = 0; j < rights.length; j++) {
              System.out.println(rights[j]);
          }
          System.out.println(playlist);
          for (int j = 0; j < playlist.items.length; j++) {
              if (playlist.items[j] instanceof MsmTitleItem) {
               MsmTitleItem ti = (MsmTitleItem)playlist.items[j];
               System.out.println(
                   session.getTitleStatus(ti.titleName));
```

```
                    }
                }
            }
        } catch (Exception e) {
            System.err.println("smcstat: " + e);
        } finally {
            if (session != null) {
             try session.close(); catch (Exception e)
                 System.err.println("smcstat: " + e);
            }
        }
    }
}
```

LOOP

```
/*
 * @(#)loop.c
 *
 * Copyright 1996 Sun Microsystems, Inc.  All Rights Reserved.
 *
 * version      1.0
 * author       Stephane CACHAT
 *
 */

#include <stdio.h>
#include <stdlib.h>

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <string.h>
#include <netdb.h>
#include <signal.h>
#include <errno.h>
#include <fcntl.h>
#include <assert.h>
#include <unistd.h>
#include <sys/time.h>
#include <sys/resource.h>
#include <time.h>
#include <thread.h>
#include <sys/errno.h>
#include <sys/stropts.h>
#include <fcntl.h>
#include <atm/atmioctl.h>

#ifdef TRUE
#undef TRUE
#endif

#ifdef FALSE
#undef FALSE
#endif

#define FALSE 0
#define TRUE 1
```

```
#define BUF 1024*8

  /*************************************************
  *** Global variables                          ***
  *************************************************/

/* Parameters */

char servername[256];
char * progName;
char *opt;
int port;
int port0;

/* Socket */

struct sockaddr_in adds;
int skt;
struct sockaddr_in addr;
struct sockaddr_in addx;
struct hostent * hp;
int len;

/* buffer */

char * buffer=NULL;

/* Multicast */

struct ip_mreq mreq;
char * host;

/* Thread */

thread_t Tpump;
int okdone=0;
int flag=1;

/* ATM */
int safd;
int ppa;
char ctlbuf[0x100];

#define vc port

  /*************************************************
  *** Receive&transmit info Multicast           ***
```

```
 ************************************************/

void * pumpM(void * result){
  while (flag) {                                    /*main loop*/
    len=recvfrom(skt,buffer,BUF,0,NULL,0);
    if (len) {
    sendto(skt,buffer,len,0,(struct sockaddr *)
&(addx),sizeof(addx));
    }
  }
  flag=1;
}

  /***********************************************
   *** Receive&transmit info ATM                ***
   ***********************************************/

void * pumpA(void * result){
  struct strbuf   ctl;
  struct strbuf   data;
  int             flags;
fprintf(stderr,"pumpA\n");
  ctl.buf = (char *) ctlbuf;
  ctl.maxlen = 0x100;
  ctl.len = 0;
  data.buf = (char *) buffer;
  data.maxlen = BUF;
  data.len = 0;
  flags = 0;
  while (flag) {                                    /*main loop*/
    if (getmsg(safd, &ctl, &data, &flags) < 0) {
      fprintf(stderr,"getmsg failed, errno=%d\n", errno);
      perror("");
      return;
    }
    len=data.len;
fprintf(stderr,"len=%d\n",len);
    if (len) {
    sendto(skt,buffer+4,len-4,0,(struct sockaddr *)
&(addx),sizeof(addx));
    }
  }
  flag=1;
}

  /***********************************************
   *** Collecting arguments                     ***
```

```
**************************************************/

void print_usage_and_exit (char* a){
    if (strlen(a)) fprintf(stderr,a);
    fprintf(stderr,"\n%s  redirect multicast or atm data stream
to lo0\n",progName);
    fprintf(stderr,"Usage\n");
    fprintf(stderr,"%s m <Multicast address> <in port> <out
port>\n",progName);
    fprintf(stderr,"%s a <VC> <out port>\n",progName);
    (void)exit(0);
}

static void collectArgs(int argc,char **argv){
    int i;
    int j=0;
    FILE * f;
    progName=*argv++;
    if (!*argv) print_usage_and_exit("");
    opt=*argv++;
    if (*opt=='a') {
      if (!*argv) print_usage_and_exit("");
      port=atoi(*argv++);
      if (!*argv) print_usage_and_exit("");
      port0=atoi(*argv++);
      if (port<=0) print_usage_and_exit("");
      if (*argv) print_usage_and_exit("");
      f=fopen("./loop.conf","r");
      if (!f){
        fprintf(stderr,"Can't open loop.conf");
        exit(-1);
      }
      host= (char*) malloc(256);
      fscanf(f,"%s",host);
      fclose(f);
    }else if (*opt=='m') {
      if (!*argv) print_usage_and_exit("");
      host=*argv++;
      if (!*argv) print_usage_and_exit("");
      port=atoi(*argv++);
      if (!*argv) print_usage_and_exit("");
      port0=atoi(*argv++);
      if (port<=0) print_usage_and_exit("");
      if (*argv) print_usage_and_exit("");
    } else print_usage_and_exit("");
}
```

```
/**************************************************
 *** Getting server IP adress                   ***
 **************************************************/

void getaddr(){
  int udpport;
  unsigned long inaddr;
  struct hostent * hp;
  char n[256];
  int i;

  if (gethostname(servername,256)==-1)
print_usage_and_exit("error while getting hostname");
  if ((inaddr=inet_addr(servername))!=-1){
    adds.sin_addr.s_addr=inaddr;
  }else{
    hp=gethostbyname(servername);
    if (hp!=NULL){
      adds.sin_addr.s_addr=((struct in_addr*)
hp->h_addr)->s_addr;
      adds.sin_port = htons(udpport);
    }
  }
  if ((inaddr=inet_addr(host))!=-1){/*hostname*/
    mreq.imr_multiaddr.s_addr=inaddr;
  }else{
    hp=gethostbyname(host);
    if (hp!=NULL){
      mreq.imr_multiaddr.s_addr=((struct in_addr*)
hp->h_addr)->s_addr;
    }else{
      fprintf(stderr,"Multicast connect failed\n");
    }
  }
 /* mreq.imr_interface.s_addr=INADDR_ANY; */
  gethostname(n,256);
  hp=gethostbyname(n);
  if (hp!=NULL){
      mreq.imr_interface.s_addr=((struct in_addr*)
hp->h_addr)->s_addr;
      addx.sin_addr.s_addr=((struct in_addr*)
hp->h_addr)->s_addr;
      addx.sin_port = htons(port0);
  }else{
      fprintf(stderr,"Multicast connect failed\n");
  }
}
```

```
/**************************************************
 *** Socket setting Multicast                  ***
 **************************************************/
void goM(){
  getaddr();
  skt=socket(AF_INET,SOCK_DGRAM,0);
  if (skt==0) {
    perror("Create socket");
    exit(EXIT_FAILURE);
  }
  addr.sin_family = AF_INET;
  addr.sin_addr.s_addr = INADDR_ANY;
  addr.sin_port = htons(port);
  bind(skt,(void *)&addr,sizeof(addr));
  if( setsockopt(skt, IPPROTO_IP, IP_ADD_MEMBERSHIP,(char*)&mreq,
sizeof(struct ip_mreq) ) == -1 ){
    fprintf(stderr,"Can't join multicast membership");
    exit(0);
  }
  if (fcntl(skt,F_SETFL,O_NDELAY)==-1){
    fprintf(stderr,"set socket options nb");
    exit(EXIT_FAILURE);
  }

  if (thr_create(0,0,pumpM,0,0,&Tpump)) perror("Can't create
Dispatcher");
}

/**************************************************
 *** ATM interface setting                     ***
 **************************************************/
void goA(){
  int udpport;
  unsigned long inaddr;
  struct hostent * hp;
  char n[256];

  char interface[10];
  memset(interface, 0, sizeof (interface));
  strcpy(interface, host);
  ppa = interface[strlen(interface) - 1] - '0';
  if ((safd = sa_open(interface)) < 0) {
    fprintf(stderr,"open failed, errno=%d\n", errno);
    perror("open");
    exit(-1);
  }
fprintf(stderr,"ready to attach\n");
```

```
  sa_attach(safd, ppa, -1);
fprintf(stderr,"attached\n");
  if (sa_add_vpci(safd, vc, NULL_ENCAP, BIG_BUF_TYPE) < 0) {
    fprintf(stderr,"sa_add_vpci failed, errno=%d\n", errno);
    exit(-1);
  }
  sa_setraw(safd);

  gethostname(n,256);
  hp=gethostbyname(n);
  if (hp!=NULL){
      addx.sin_addr.s_addr=((struct in_addr*)
hp->h_addr)->s_addr;
      addx.sin_port = htons(port0);
  }else{
      fprintf(stderr,"lo0 connect failed\n");
  }
  skt=socket(AF_INET,SOCK_DGRAM,0);
  if (skt==0)  {
    perror("Create socket");
    exit(EXIT_FAILURE);
  }
  addr.sin_family = AF_INET;
  addr.sin_addr.s_addr = INADDR_ANY;
  addr.sin_port = htons(port0);
  bind(skt,(void *)&addr,sizeof(addr));
  if (fcntl(skt,F_SETFL,O_NDELAY)==-1){
    fprintf(stderr,"set socket options nb");
    exit(EXIT_FAILURE);
  }

  if (thr_create(0,0,pumpA,0,0,&Tpump)) perror("Can't create
Dispatcher");
}

  /**************************************************
   *** Cleaning ATM                             ***
   **************************************************/

void doneA(int arg){
 fprintf(stderr,"loop killed by signal %d\n",arg);
 if (!okdone){okdone=1;
  flag=0;
  while (!flag) {
    sleep(1);
  }
  fprintf(stderr,"dispatcher killed\n");
```

```
  if (sa_delete_vpci(safd, vc) < 0) {
    fprintf(stderr,"sa_delete_vpci failed, errno=%d\n", errno);
  };
fprintf(stderr,"ready to detach\n");
  sa_detach(safd, -1);
fprintf(stderr,"detached\n");
  sa_close(safd);
  close(skt);
  printf("socket closed\n");
  if (buffer) free(buffer);
  printf("Buffer free\n");
  exit(0);
}}

  /************************************************
   *** Cleaning Multicast                       ***
   ************************************************/

void doneM(int arg){
 if (!okdone){okdone=1;
  if (setsockopt(skt,IPPROTO_IP, IP_DROP_MEMBERSHIP,(char *)
&mreq,sizeof(mreq))==-1){
    fprintf(stderr,"Can't drop multicast membership");
    exit(0);
  }
  printf("Multicast membership dropped\n");

  flag=0;
  while (!flag) {
    sleep(1);
  }
  printf("dispatcher killed\n");

  close(skt);
  printf("socket closed\n");
  if (buffer) free(buffer);
  printf("Buffer free\n");
  exit(0);
}}

  /************************************************
   *** Main                                     ***
   ************************************************/

int main(int argc, char** argv)
{
  int i;
```

```
 buffer=(char*) malloc(BUF);
 collectArgs(argc,argv);
if (*opt=='m'){
   printf("host=%s, port=%d, port0=%d\n",host,port,port0);
   signal(SIGQUIT,doneM);
   signal(SIGINT,doneM);
   signal(SIGUSR1,doneM);
   signal(SIGUSR2,doneM);

   printf("go M\n");
   goM();
}else if (*opt=='a'){
   printf("interface=%s, vc=%d,port0=%d\n",host,vc,port0);
   signal(SIGQUIT,doneA);
   signal(SIGINT,doneA);
   signal(SIGUSR1,doneA);
   signal(SIGUSR2,doneA);

   printf("go A\n");
   goA();
}

printf("loop\n");

while(1) sleep(60);
}
```

Claims

1. A method for processing in a computer which includes a memory a bit stream received from a bit stream server which is operatively coupled to the computer through a network, the method comprising:

   retrieving from a multimedia document stored in the memory a specification of a title;
   building from the specification of the title bit stream control signals which request a bit stream representing the title and which are in a form appropriate for processing by the bit stream server;
   transmitting the bit stream control signals to the bit stream server to thereby request from the bit stream server a bit stream representing the title;
   building from the specification of the title decoder control signals which direct a decoder to receive the bit stream from the bit stream server and which are in a form appropriate for processing by the decoder; and
   transmitting the decoder control signals to the decoder to thereby cause the decoder to receive and decode the bit stream.

2. An applet, capable of executing within a computer system, for requesting and controlling decoding of a bit stream specified in a multimedia document stored in a memory of the computer system, the applet comprising:

   an API module (i) which is configured to build from a specification of the bit stream in the multimedia document bit stream control signals which request transmission of the bit stream from a bit stream server and which are in a form appropriate for processing by the bit stream server and (ii) which is configured to transmit the bit stream control signals to the bit stream server to thereby request from the bit stream server a bit stream representing the title; and
   a decoder module (i) which is operatively coupled to the API module; (ii) which is configur d to build from the specification of the bit stream in the multimedia document decoder control signals which direct a decoder to

receiv the bit stream from th bit stream server and which are in a form appropriat for processing by the decoder; and (iii) which is configured to transmit the decoder control signals to the decoder to thereby cause the decoder to receive and decod the bit stream.

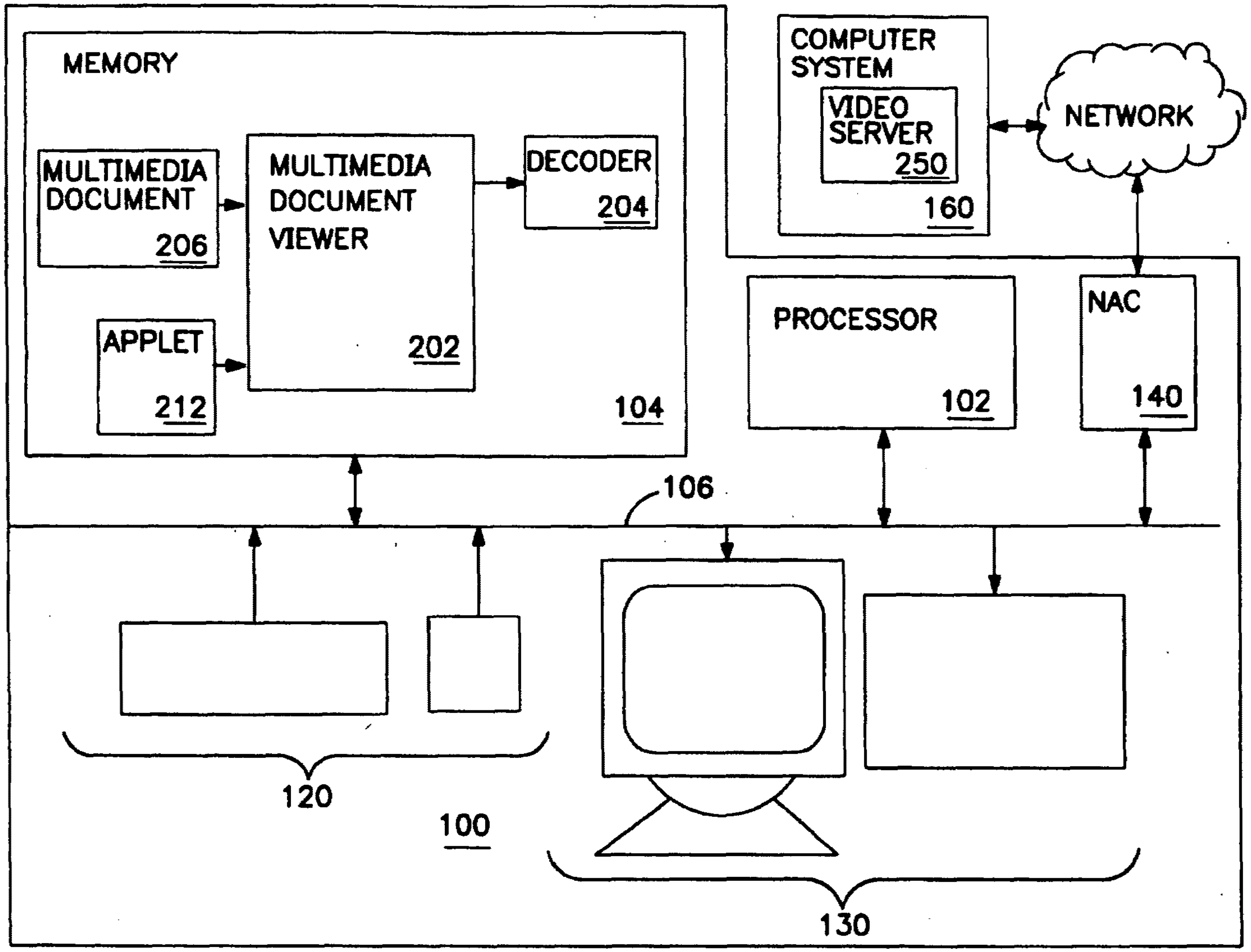

MEMORY
MULTIMEDIA DOCUMENT
206
MULTIMEDIA DOCUMENT VIEWER
202
DECODER
204
APPLET
212
104
COMPUTER SYSTEM
VIDEO SERVER
250
160
NETWORK
PROCESSOR
102
NAC
140
106
120
100
130

FIG. 1

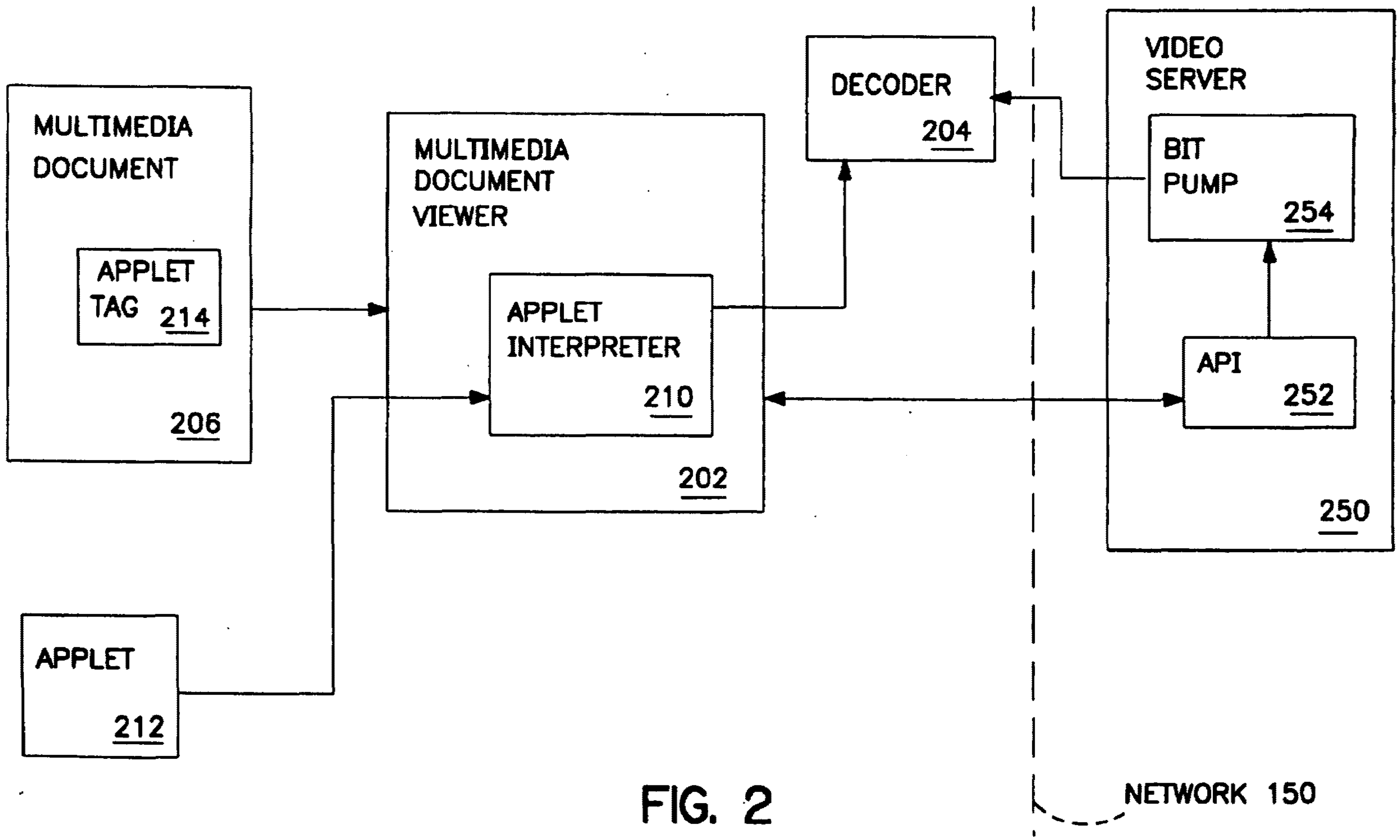
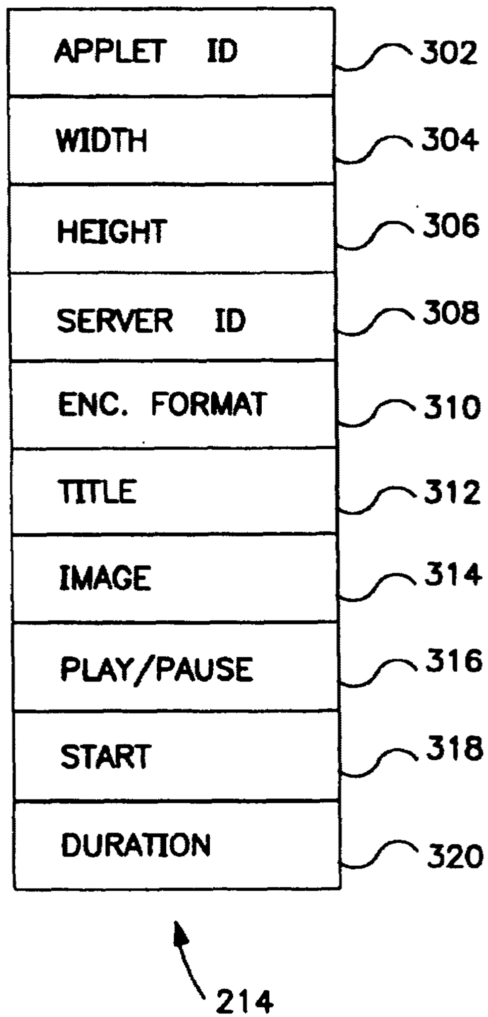

MULTIMEDIA DOCUMENT
APPLET TAG 214
206
MULTIMEDIA DOCUMENT VIEWER
APPLET INTERPRETER
210
202
DECODER
204
VIDEO SERVER
BIT PUMP
254
API
252
250
APPLET
212
NETWORK 150

FIG. 2

FIG. 3


APPLET ID
302
WIDTH
304
HEIGHT
306
SERVER ID
308
ENC. FORMAT
310
TITLE
312
IMAGE
314
PLAY/PAUSE
316
START
318
DURATION
320
214

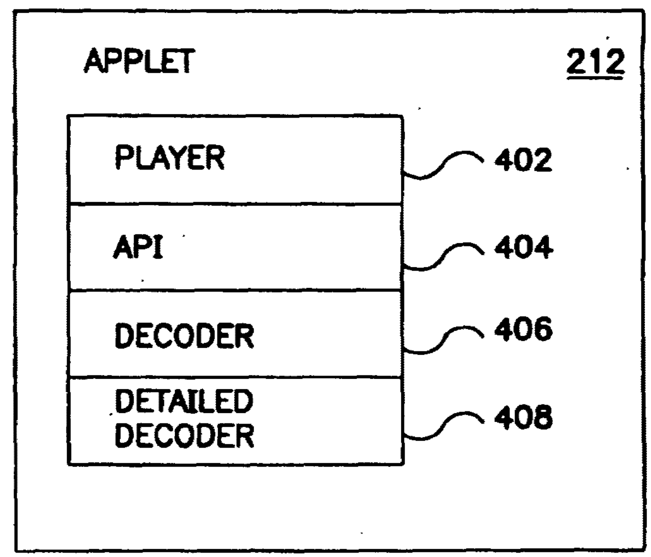

APPLET
212
PLAYER
402
API
404
DECODER
406
DETAILED DECODER
408

FIG. 4

NOV 25 2002 JC89 PATENT & TRADEMARK OFFICE

2151 $

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: Assistant Commissioner for Patents, Washington, D.C., 20231, on:

Date: November 18, 2002

By: Carina M. Tan
Carina M. Tan

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**

RECEIVED
NOV 27 2002
Technology Center 2100

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

1. Transmitted herewith are the following:

☒ Amendment and Response, with Version with Markings to Show Changes Made
☒ Declaration of Adam Cheyer
☒ Declaration of David L. Martin
☒ Applicants request one month extension of time

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

Check No. 1123 the amount of $55.00 is enclosed for the one month extension of time. Please charge any underpayment in fees for timely filing of this transmittal and enclosures to Deposit Account No. 50-2207.

Respectfully submitted,
Perkins Coie LLP

Date: November 18, 2002

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

Corr spondence Address:
Customer No. 22918
Perkins Coie LLP
P.O. Box 2168
Menlo Park, CA 94
(650) 838-4300

OIPE
NOV 2 5 2002
PATENT & TRADEMARK OFFICE

#5/Amd A
T. McBeth Brown
12/10/02

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: Assistant Commissioner for Patents, Washington, DC 20231

on November 18, 2002 by Carina M. Tan
Carina M. Tan

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

RECEIVED
NOV 2 7 2002
Technology Center 2100

Commissioner of Patents
Washington, D.C. 20231

AMENDMENT AND RESPONSE

Sir:

This is in response to the Office Action mailed July 17, 2002, the shortened statutory period for which runs until October 17, 2002.

IN THE CLAIMS

Please amend Claims 1-3, 48, 84-88. A set of "clean" claims have been provided herein. Further, a set of claims having markings that show the changes that are made in this amendment is attached herewith. The attached pages are captioned "**Version of claims with markings to show changes made**."

11/26/2002 NMOHAMM1 00000101 09225198
01 FC:2251 55.00 OP

AMENDED CLAIMS IN CLEAN FORM

IN THE CLAIMS:

1. (Once 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;

Sub B1

receiving a request for service as a base goal in the inter-agent language, in the form of an arbitrarily complex goal expression;
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
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. (Once 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. (Once amended) 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.

48. (Once 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:

Sub B1

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 operator that indicates that disjunct goals are to be performed by different agents.

84. (Once 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. (Once 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.

86. (Once 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 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. (Once 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. (Once 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.

REMARKS

The Examiner is thanked for the performance of a thorough search. By this amendment, Claims 1-3, 48, and 84-88 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.

SUMMARY OF REJECTIONS/OBJECTIONS

In the Office Action, Claim 2 is rejected under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claim 3 recites the limitation "from the specific agent to the facilitator agent" and is rejected under 35 U.S.C. § 112, second paragraph for lacking sufficient antecedent basis for this limitation in the claim.

Claims 84 and 85 are rejected under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claims 87 and 88 recite the limitation "A data wave carrier as recited in claim 85" and are rejected under 35 U.S.C. § 112, second paragraph for lacking sufficient antecedent basis for this limitation in the claim.

Claims 1, 2, 5-11, 15-28, 48-89 are rejected under 35 U.S.C. § 102(b) as being anticipated by "Building Distributed Software Systems With The Open Agent Architecture" by Martin et al.

Claims 1, 2, 5-11, and 15-25 are rejected under 35 U.S.C. 102(b) as being anticipated by "Development Tools for the Open Agent Architecture" by Martin et al.

Claims 3, 29-34, and 38-47 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Building Distributed Software Systems with the Open Agent Architecture" by Martin.

Claims 4, 12-14 and 35-37 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Building Distributed Software Systems with the Open Agent Architecture" by Martin 1 in view of "Information Brokering in an Agent Architecture" by Martin 2.

Claims 3, 29-34, 38-47, 61-71 and 84-89 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Developing Tools for the Open Agent Architecture" by Martin et al.

Claims 4, 12-14, 26-28, 35-37, 48-60, 72-83 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Development Tools for the Open Agent Architecture" by Martin 1 in view of "Information Brokering in an Agent Architecture" by Martin 2.

REJECTIONS UNDER 35 U.S.C. § 112

CLAIMS 2, 3, 84, 85, 87, and 88

In the Office Action, Claims 2, 3, 84, 85, 87, and 88 are rejected under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claims 2, 3, 84, 85, 87, and 88 are amended according to the suggestions of the Examiner. Thus, the amendments to the claims as indicated herein have been made in view of the Office Action's rejection under 35 U.S.C. § 112, second paragraph and to improve clarity of the claims.

AFFIDAVITS OF DAVID MARTIN AND ADAM CHEYER UNDER 37 CFR §1.132

Submitted herewith is a declaration under 37 CFR §1.132 by David Martin. In his declaration, David Martin avers that: 1) David Martin, Adam Cheyer and Douglas Moran are the co-authors of the reference, "Building Distributed Software Systems with the Open Agent

Architecture", 2) David Martin and Adam Cheyer are the only inventors of the subject application, 3) the reference, "Building Distributed Software Systems with the Open Agent Architecture" was published in March 1988, which is less than one year from the filing date of January 5, 1999.

Also, submitted herewith is a declaration under 37 CFR §1.132 by Adam Cheyer. In his declaration, Adam Cheyer avers that: 1) David Martin, Adam Cheyer and Douglas Moran are the co-authors of the reference, "Building Distributed Software Systems with the Open Agent Architecture", 2) David Martin and Adam Cheyer are the only inventors of the subject application, 3) the reference, "Building Distributed Software Systems with the Open Agent Architecture" was published in March 1988, which is less than one year from the filing date of January 5, 1999.

In accordance with MPEP 716.10, David Martin's declaration and Adam Cheyer's declaration render the reference, "Building Distributed Software Systems with the Open Agent Architecture" as inapplicable prior art.

REJECTIONS UNDER 35 U.S.C. § 102(b) and § 103(a)

CLAIM 1

Claim 1, as amended, recites in part:

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

The novel method recited in Claim 1 requires "**constructing a goal satisfaction plan that includes said one or more sub-goals.**" None of the cited references disclose, suggest or render obvious the limitation of "constructing a goal satisfaction plan that includes said one or more sub-goals." For example, Claim 1 requires constructing a goal satisfaction plan that includes said one or more sub-goals whenever the sub-goals cannot be generated by a simple decomposition of the "arbitrarily complex goal expression" in Claim 1. In other words, "**a goal satisfaction plan"** is needed to satisfy the "arbitrarily complex goal expression" in Claim 1 whenever there is no direct match between the components of arbitrarily complex goal expression and the "**registered functional capabilities**" of the client agents.

Since, none of the cited references disclose, suggest or render obvious the limitations of Claim 1 including the limitation of "constructing a goal satisfaction plan that includes said one or more sub-goals", Claim 1 is allowable over the art of record. It is respectfully submitted that Claim 1 be held in condition for allowance.

CLAIMS 2-28

Claims 2-28 are either directly or indirectly dependent upon independent Claim 1, and include all the features of Claim 1. Therefore, Claims 2-28 are allowable for at least the reasons provided herein with respect to Claim 1. Furthermore, it is respectfully submitted that Claims 2-28 recite additional features that independently render Claims 2-28 patentable over the art of record. Thus, it is respectfully submitted that Claims 2-28 be held in condition for allowance.

CLAIMS 29, 61, 71 and 86

Claims 29, 61, 71 and 86, each contain the limitation requiring the "construction of a goal satisfaction plan".

Claim 29, recites in part, the limitations of:

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

Claim 61, recites in part, the limitations of:

"the facilitating engine further operable to **construct a goal satisfaction plan** 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."

Claim 71, recites in part, the limitations of:

"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."

Claim 86, recites in part, the limitations of:

"wherein said at least one facilitator agent is operable to **construct a goal satisfaction plan** for satisfying one or more requests for service from said at least one active client agent,"

Thus, Claims 29, 61, 71 and 86 contain limitations that are similar to those described herein with respect to Claim 1. Therefore, based on the reasons stated herein, it is respectfully submitted that Claims 29, 61, 71 and 86, are allowable over the art of record for at least the reasons provided herein with respect to Claim 1. Furthermore, it is respectfully submitted that Claims 29, 61, 71 and 86 recite additional features that independently render Claims 29, 61, 71 and 86 patentable over the art of record. Therefore, it is respectfully submitted that Claims 29, 61, 71 and 86 be held in condition for allowance.

CLAIMS 30-47, 62-70, 72-85, 87-89

Claims 30-47, 62-70, 72-85, 87-89 are either directly or indirectly dependent upon independent Claims 29, 61, 71 and 86, respectively. Therefore, Claims 30-47, 62-70, 72-85, 87-89 are allowable for at least the reasons provided herein with respect to Claims 29, 61, 71, 86 and 1. Furthermore, it is respectfully submitted that Claims 30-47, 62-70, 72-85, 87-89 recite additional features that independently render Claims 30-47, 62-70, 72-85, 87-89 patentable over the art of record. Thus, it is respectfully submitted that Claims 30-47, 62-70, 72-85, 87-89 be held in condition for allowance.

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 conjunctive operator;**
> **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 Claim48 requires that "**goals within a single request**" are "**coupled by one or more operators from a set of operators**". In Claim 48, the set of operators comprise, **a conjunctive operator, a conditional execution operator**, and **a parallel disjunctive operator**.

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 conjunctive operator, a conditional execution operator**, and **a parallel disjunctive operator**. 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: November 18, 2002
(Monday)

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

VERSION OF CLAIMS WITH MARKINGS TO SHOW CHANGES MADE

1. (Once 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 <u>as corresponding registered functional capabilities,</u> 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;

dynamically interpreting the <u>arbitrarily complex</u> goal expression, said act of interpreting further comprising:

generating one or more sub-goals [using] <u>expressed in</u> the inter-agent language; [and]

<u>constructing a goal satisfaction plan that includes said one or more sub-goals; and</u>

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. (Once 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 [last] step of <u>dynamically interpreting the arbitrarily complex goal expression</u> [claim 1] in order to perform the new request for service.

3. (Once amended) 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 [the] a facilitator agent in response to the instantiation of the specific agent.

48. (Once 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

[in] 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 operator [parallel disjunctive operator] that indicates that disjunct goals are to be performed by different agents.

84. (Once amended) A computer architecture as recited in claim 71 wherein [the] a planning component of the facilitating engine is distributed across at least two computer processes.

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

86. (Once 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 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. (Once amended) A data wave carrier as recited in claim [85] 86, the data wave carrier further comprising a corresponding signal representation of [request] said one or more requests for service in the inter-agent language from a first agent to a second agent.

88. (Once amended) A data wave carrier as recited in claim [85] 86, the data wave carrier further comprising a signal representation of a goal dispatched to an agent for performance from a facilitator agent.

Serial No. 09/225,198

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: Assistant Commissioner for Patents, Washington, D.C., 20231, on:

Date: November 18, 2002 By: Carina M. Tan

DOCKET NO.: 59501-8016.US01

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF:

Cheyer *et al.*

SERIAL NO.: 09/225,198

FILED: 01/05/99

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

EXAMINER: Bullock Jr., L.

ART UNIT: 2151

DECLARATION UNDER 37 C.F.R. §1.132

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

I, David L. Martin, declare and affirm as follows:

1. I am a co-inventor, along with Adam J. Cheyer, of the subject matter described and claimed in U.S. Patent Application Serial No. 09/225,198, filed January 05, 1999, entitled SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONICS AGENTS.

2. I am co-author of an article published in March, 1998, entitled "Building Distributed Software Systems with the Open Agent Architecture." The article included as co-authors, Adam J. Cheyer and Douglas B. Moran. Thus, the article was published less than one year from the filing date of the instant application.

3. I and Adam J. Cheyer are the inventors of the subject matter, which is claimed in claims 1-

86 in the instant application.

4. Douglas B. Moran is not a co-inventor of the subject matter described in the subject matter disclosed and claimed in the instant application.

I declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the Unites States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Respectfully submitted,

11/14/2002
Date

David L. Martin

#/

NOV 25 2002 OIPE

Serial No. 09/225.198

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: Assistant Commissioner for Patents, Washington, D.C. 20231, on:

Date: November 15, 2002 By: Corina M. Tan

DOCKET NO.: 59501-8016.US01

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF: Cheyer *et al.*	EXAMINER: Bullock Jr., L.
SERIAL NO.: 09/225,198	ART UNIT: 2151
FILED: 01/05/99	
FOR: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONICS AGENTS	

RECEIVED
NOV 27 2002
Technology Center 2100

DECLARATION UNDER 37 C.F.R. §1.132

Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

I, Adam J Cheyer, declare and affirm as follows:

1. I am a co-inventor, along with David L. Martin, of the subject matter described and claimed in U.S. Patent Application Serial No. 09/225,198, filed January 05, 1999, entitled SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONICS AGENTS.

2. I am co-author of an article published in March, 1998, entitled "Building Distributed Software Systems with the Open Agent Architecture." The article included as co-authors, David L. Martin and Douglas B. Moran. Thus, the article was published less than one year from the filing date of the instant application.

3. I and David L. Martin are the inventors of the subject matter, which is claimed in claims 1-

Serial No. 09/225,198

86 in the instant application.

4. Douglas B. Moran is not a co-inventor of the subject matter described in the subject matter disclosed and claimed in the instant application.

I declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the Unites States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Respectfully submitted,

11/15/02

Date

Adam J. Cheyer

Adam J. Cheyer

3 M.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
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

25696 7590 03/03/2003

OPPENHEIMER WOLFF & DONNELLY
P. O. BOX 10356
PALO ALTO, CA 94303

EXAMINER	
BULLOCK JR, LEWIS ALEXANDER	8

ART UNIT	PAPER NUMBER
2126	

DATE MAILED: 03/03/2003

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

PTO-90C (Rev. 07-01)

8M

Office Action Summary	Application No.	Applicant(s)
	09/225,198	CHEYER ET AL.
	Examiner	Art Unit
	Lewis A. Bullock, Jr.	2126

-- The MAILING DATE of this communication appears on the cover sheet with the 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 25 November 2002.

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) 1-89 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) 1-89 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).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

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

13)☐ 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.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

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

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

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) 4.
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 A.I, 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.

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 1-89 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 "Information Brokering in an Agent Architecture" by MARTIN2.

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.

MARTIN2 teaches an agent architecture for request communication comprising the step of constructing a goal satisfaction plan (query execution plan) that includes one or more sub-goals (sub-queries) and dispatching each sub-goal (sub-queries) to a selected agent (source) for performance based on a match between the capabilities of the agent and the sub-goal ("for each chunk, rewrite it as a disjunction of translated sub-queries where each disjunct is the translation of the sub-query for one of the source s that can handle that chunk.") (pg. 11-12, Query Processing). Therefore, it would be obvious to one skilled in the art to combine the teachings of MARTIN1 with the teachings of MARTIN2 in order to facilitate query processing (pg. 11).

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."). However, MARTIN1 does not explicitly mention that the method is operable in a computer program product or the sending of advice or constraints. It would be obvious 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. 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.

MARTIN2 teaches an agent architecture for request communication comprising the step of constructing a goal satisfaction plan (query execution plan) comprising: determining whether the service is available (determine what set of sources provides solutions for that predicate), determining sub-goals required in completing the base goal (determine which are the largest sub-queries that can be treated as chunks and which sources can handle each chunk); selecting service-providing agents ("which sources can handle each chunk), and ordering a delegation of sub-gal request to best complete the requested service ("for each chunk, rewrite it as a disjunction of translated sub-queries...each translated subquery is labeled with the name of the source by which it is to be solved."); and implementing the base goal satisfaction plan ("The plan is then interpreted according to Prolog semantics.") (pg. 11-12, Query Processing). It would be obvious 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/or constraints. It would also be obvious to one skilled in the art to generate program code that would entail the method of MARTIN2 and thereby obvious that the method can be entailed in a computer program product. Refer to claim 1 for the motivation to combine.

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 of other agents (pg. 5, Agents share a common communication language...and may run on any network linked platform."). However, MARTIN1 does not teach the ICL supporting compound goal expressions.

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 (pg. 10, "Although the body of the broker predicate rule is characterized as a conjunction of predicates."). 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 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), 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.

MARTIN2 teaches a facilitator agent (facilitator) having a facilitating engine (broker agent) (pg. 7, "...the Information Broker agent, working in close cooperation with the OAA facilitotor.") operable to parse a service request in order to interpret a compound goal (pg. 7, "The Broker accepts request (queries) from..."; "The Broker delegates, translates, and relays the appropriate sub-queries to the available source agents.."; pg. 8, "Each query is syntactically the same as a Prolog goal, usually a compound goal."), the compound goal including constraints and parameters (built-in predicates) (pg. 11, "..ICL built-in predicates (including arithmetic comparisons) are included with chuncks to be solved by sources."), the service request formed according to an ICL (pg. 11), the engine further operable to construct a goal satisfaction plan (query execution plan) specifying the coordination of a suitable delegation of sub-goal (sub-queries) requests to complete the requested service satisfying the constraints and parameters (pg. 11, Query Processing). Refer to claim 1 for the motivation to combine.

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.

MARTIN2 teaches an agent system for information communication wherein a facilitation agent (broker agent) is operable to construct a goal satisfaction plan (query execution plan) for satisfying one or more request (query) for service from the at least one active client agent (source) (pg. 11-12, Query Processing). Refer to claim 1 for the motivation to combine.

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 claim 4, 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.").

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 12-14, 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).

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 26-28, MARTIN2 teaches the base goal is a compound goal having sub-goals (pg. 8, "Queries submitted to the Broker are expression...and backtracking in expressing and processing queries."). 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.

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

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

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

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 (pg. 5, "Every agent participating in an OAA-based system defines and publishes...we refer to these capabilities specifications as solvables.").

As to claim 62, MARTIN2 teaches the facilitating engine (broker agent) is able to receive events such as online and offline agents (pg. 8-9, The Broker agent). It would be obvious that the plan is modified if a particular agent goes offline since that agent is no longer available.

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 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.

U.S. Patent and Trademark Office
PTO-892 (Rev. 01-2001) Notice f R ferences Cited Part of Paper No. 8

OIPE JC109 AUG 1 3 2002 PATENT & TRADEMARK OFFICE

RECEIVED AUG 1 5 2002 Technology Center 2100

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
fab	1	5,197,005		Schwartz et al.	3/23/93	
fab	2	5,386,556		Hedin et al.	1/31/95	
fab	3	5,434,777		Luciw	7/18/95	
fab	4	5,519,608		Kupiec	5/21/96	
fab	5	5,608,624		Luciw	3/4/97	
fab	6	5,721,938		Stuckey	2/24/98	
fab	7	5,729,659		Potter	3/17/98	
fab	8	5,748,974		Johnson	5/5/98	
fab	9	5,774,859		Houser et al.	6/30/98	
fab	10	5,794,050		Dahlgren et al.	8/11/98	

FOREIGN PATENT DOCUMENTS

Examiner Initial	Cite No.	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
fab	11	WO	00/11869		Ellis et al.	3/2/00		
fab	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
fab	13	Dowding, John et al., "Gemini: A Natural Language System For Spoken-Language Understanding", SRI International	
fab	14	http://www.ai.sri.com/~oaa/infowiz.html, "InfoWiz: An Animated Voice Interactive Information System, May 8, 2000	
fab	15	Dowding, John, "Interleaving Syntax and Semantics in an Efficient Bottom-up Parser", SRI International	
fab	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	2/21/03

*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).

OIPE JC109
AUG 13 2002
PATENT & TRADEMARK OFFICE

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

RECEIVED
AUG 15 2002
Technology Center 2100

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).

BY022180

2126

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF:	ATTORNEY DOCKET NO.: 59501.8016.US01
ADAM CHEYER ET AL.	EXAMINER: LEWIS ALEXANDER BULLOCK JR.
APPLICATION NO.: 09/225,198	ART UNIT: 2126
FILING DATE: JANUARY 5, 1999	
FOR: **SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS**	

#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

pol

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

Lew 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 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 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.

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.

JUN 0 6 2003

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

RECEIVED
JUN 1 6 2003
Technology Center 2100

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
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

- [x] 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 *AMENDMENTS 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 <u>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</u>,

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.

B\

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.

B1

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.

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 <u>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</u> ~~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.

31

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.

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;

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 <u>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</u>.

B1

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.

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.

B1

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)

<u>CLAIMS 1, 29, 61, 71 and 86</u>

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
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)

Office Action Summary	Application N .	Applicant(s)
	09/225,198	CHEYER ET AL.
	Examin r	Art Unit
	Lewis A. Bullock, Jr.	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 <u>must</u> 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.

OIPE JC48
JUN 0 6 2003
PATENT & TRADEMARK OFFICE

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
LAB		6,047,053		Miner et al.	04/04/00	
LAB		6,256,771	B1	O'Neil et al.	07/03/01	
LAB		6,411,684	B1	Cohn et al.	06/25/02	
						RECEIVED
						JUN 0 9 2003
						Technology Center 2100

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
Lee 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).

Ŝ

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 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.*.

(2) *Corina Tan*.

(3) *David Stringer-Calbert*.

(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), <u>wherein the ICL includes:</u>

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

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

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:

<u>the ICL having one or more of:</u>

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

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

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, <u>and an Interagent Communication Language (ICL), wherein the ICL includes:</u>

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

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

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™ 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.

69

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. 09/225 198
Reviewed By: K. SMITH Phone: 308/215

- [x] 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/2100

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.

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

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:

<u>the ICL having one or more of:</u>

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

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

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.

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

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
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™ 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.

RECEIVED
CENTRAL FAX CENTER
JUN 08 2004

OFFICIAL

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

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

OIPE MAR 2 9 2004 PATENT & TRADEMARK OFFICE

2195

PERKINS COIE LLP
BAY AREA PATENT
101 JEFFERSON DRIVE
MENLO PARK, CA 94025-1114

US BA
1420 5TH
SEATTLE, WA 98101
19-10/1250

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

Shelina Berger

MEMO 59501-8016.US01

⑈002195⑈ ⑆125000105⑆153592271574⑈

PERKINS COIE LLP 2195

Commissioner for Patents	Carina Tan	3/26/2004	55.00

U. S. Bank 59501-8016.US01 55.00

PERKINS COIE LLP 2195

Commissioner for Patents	Carina Tan	3/26/2004	55.00

PAYMENT RECORD

BEST AVAILABLE COPY

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

BEST AVAILABLE COPY

Attorney Docket No. 59501-8016.US01

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

ply

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.

BEST AVAILABLE COPY

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: _____.

BEST AVAILABLE COPY

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: _____

Lewis A. Bullock Jr

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.

RECEIVED
CENTRAL FAX CENTER
JUN 08 2004

EXPRESS MAIL LABEL NO. EV 099152888 US

OFFICIAL

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

LAB 7/7/04

BEST AVAILABLE COPY

CONFIDENTIAL PRIVILEGED

Perkins Coie LLP-Menlo Park

RECEIVED
CENTRAL FAX CENTER
AUG 2 5 2004

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

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

RECEIVED
CENTRAL FAX CENTER
AUG 2 5 2004

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
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

[59501-8016/BY042380.033]

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

RECEIVED
CENTRAL FAX CENTER
AUG 2 5 2004

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:

CHEYER et al.

Serial No.: 09/225,198

Filed on: January 5, 1999

Atty Dkt. No. 59501-8016.US01

Group Art Unit No.: 2126

Examiner: L. A. Bullock, Jr.

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.

This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☑ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: ______________________

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.

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, <u>wherein the parameter lists further refine the one or more events</u>; 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.

This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☑ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: ______________________

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.

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

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

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. __________

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.

PTOL-85 (Rev. 09/04) Approved for use through 04/30/2007.

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.

PTOL-85 (Rev. 09/04) Approved for use through 04/30/2007.

Notice of Allowability	Application No.	Applicant(s)
	09/225,198	CHEYER ET AL.
	Examiner	Art Unit
	Lewis A. Bullock, Jr.	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.	Applicant(s)
	09/225,198	CHEYER ET AL.
	Examiner	Art Unit
	Lewis A. Bullock, Jr.	2126

All Participants: **Status of Application:** *Allowed*

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

(2) *Corina Tan*. (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 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

RECEIVED
CENTRAL FAX CENTER
AUG 25 2004

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:

CHEYER et al.

Serial No.: 09/225,198

Filed on: January 5, 1999

Atty Dkt. No. 59501-8016.US01

Group Art Unit No.: 2126

Examiner: L. A. Bullock, Jr.

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:

Do Not ENTER JCB

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

Foreign Priority claimed / 35 USC 119 (a-d) conditions met / Verified and Acknowledged	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 ACCOUNT NO. ______ for the following:	☐ All Fees ☐ 1.16 Fees (Filing) ☐ 1.17 Fees (Processing Ext. of time) ☐ 1.18 Fees (Issue) ☐ Other ______ ☐ Credit

Best Available Copy

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 Final	Claim 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 Final	Claim 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 Final	Claim Original	Date
	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	

Issue Classification	Application No. 09/225,198	Applicant(s) CHEYER ET AL.
	Examiner Lewis A. Bullock, Jr.	Art Unit 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)

(Legal Instruments Examiner) (Date)

Lewis A. Bullock, Jr. 09/03/04

(Primary Examiner) (Date)

Total Claims Allowed: 89

O.G. Print Claim(s)	O.G. Print Fig.
1	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

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

SEARCH NOTES (INCLUDING SEARCH STRATEGY)

	DATE	EXMR
EAST ACM IEEE CITESEER INTERNET	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

U.S. Patent and Trademark Office

Part of Paper No. 20040903

PART B - FEE(S) TRANSMITTAL

OIPE JCIS OCT 0 4 2004 PATENT & TRADEMARK OFFICE

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: SRI International

(B) RESIDENCE: (CITY and STATE OR COUNTRY): 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

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:	EXAMINER: L. A. BULLOCK, JR.
CHEYER ET AL.	ART UNIT: 2126
APPLICATION NO.: 09/225,198	
FILED: January 5, 1999	NOTICE OF ALLOWABILITY: SEPTEMBER 10, 2004
FOR: **SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS**	

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

OIPE JC22
OCT 0 4 2004
PATENT & TRADEMARK OFFICE

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
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

A

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF EXPRESS MAILING
I hereby certify that this paper and the documents and/or fees referred to as attached therein are being deposited with the United States Postal Service on January 05, 1999 in an envelope as "Express Mail Post Office to Addressee" service under 37 CFR §1.10, Mailing Label Number **EL221766053US**, addressed to the Assistant Commissioner for Patents, Washington, DC 20231.

Michael L. Gough

01/05/99 jc571 U.S. PTO

Attorney Docket No.: SRI1P016

First Named Inventor:

CHEYER, Adam J.

jc618 U.S. PTO 09/225198 01/05/99

UTILITY PATENT APPLICATION TRANSMITTAL (37 CFR § 1.53(b))

Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

☐ Duplicate for fee processing

Sir: This is a request for filing a patent application under 37 CFR § 1.53(b) in the name of inventors: Adam J. Cheyer and David L. Martin

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

Application Elements:

- ☒ 59 Pages of Specification, Claims and Abstract
- ☒ 16 Sheets of Drawings
- ☒ 01 Pages Combined Declaration and Power of Attorney

Accompanying Application Parts:

- ☒ Assignment and Assignment Recordation Cover Sheet (recording fee not enclosed)
- ☒ Return Receipt Postcard

Fee Calculation (37 CFR § 1.16)

	(Col. 1) NO. FILED		(Col. 2) NO. EXTRA	SMALL ENTITY RATE	SMALL ENTITY FEE	OR	LARGE ENTITY RATE	LARGE ENTITY FEE
BASIC FEE				$395	$	OR	$760	$ 760.00
TOTAL CLAIMS	89	-20 =	69	x11 =	$	OR	x18 =	$1242.00
INDEP CLAIMS	06	-03 =	03	x41 =	$	OR	x78 =	$ 234.00
* If the difference in Col. 1 is less than zero, enter "0" in Col. 2.				Total	$	OR	Total	$2236.00

Including filing fees and the assignment recordation fee of $40.00, the Commissioner is authorized to charge all required fees to Deposit Account No. 50-0384 (Order No. SRI1P016).

☒ The Commissioner is authorized to charge any fees beyond the amount enclosed which may be required, or to credit any overpayment, to Deposit Account No. 50-0384 (Order No. SRI1P016).

General Authorization for Petition for Extension of Time (37 CFR §1.136)

☒ Applicants hereby make and generally authorize any Petitions for Extensions of Time as may be needed for any subsequent filings. The Commissioner is also authorized to charge any extension fees under 37 CFR §1.17 as may be needed to Deposit Account No. 50-0384.

☒ Please send correspondence to the following address:

Brian R. Coleman
HICKMAN STEPHENS & COLEMAN, LLP
P.O. Box 52037
Palo Alto, CA 94303-0746

Tel (650) 470-7430
Fax (650) 470-7440

Date: 1/5/99

Brian R. Coleman
Registration No. **39,145**

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

By:

Adam J. Cheyer and David L. Martin

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*

computers (*clients*). Some of these computers provide resources to other computers and are known as *server computers* (*servers*). The servers 122 can vary greatly in the resources they possess, access they provide and services made available to other computers across a network. Servers may service other servers as well as clients.

The Internet is a computing system based upon this network computing model. The Internet is continually growing, stimulating a paradigm shift for computing away from requiring all relevant data and programs to reside on the user's desktop machine. The data now routinely accessed from computers spread around the world has become increasingly rich in format, comprising multimedia documents, and audio and video streams. With the popularization of programming languages such as JAVA, data transported between local and remote machines may also include programs that can be downloaded and executed on the local machine. There is an ever increasing reliance on networked computing, necessitating software design approaches that allow for flexible composition of distributed processing elements in a dynamically changing and relatively unstable environment.

In an increasing variety of domains, application designers and users are coming to expect the deployment of *smarter, longer-lived, more autonomous, software applications.* Push technology, persistent monitoring of information sources, and the maintenance of user models, allowing for personalized responses and sharing of preferences, are examples of the simplest manifestations of this trend. Commercial enterprises are introducing significantly more advanced approaches, in many cases employing recent research results from artificial intelligence, data mining, machine learning, and other fields.

More than ever before, the increasing complexity of systems, the development of new technologies, and the availability of multimedia material and environments are creating a demand for *more accessible and intuitive user interfaces.* Autonomous, distributed, multi-component systems providing sophisticated services will no longer lend themselves to the familiar "direct manipulation" model of interaction, in which an individual user masters a fixed selection of commands provided by a single application. Ubiquitous computing, in networked environments, has brought about a situation in which the typical user of many software services is likely to be a non-expert, who may access a given service infrequently or only a few times.

Accommodating such usage patterns calls for new approaches. Fortunately, input modalities now becoming widely available, such as speech recognition and pen-based handwriting/gesture recognition, and the ability to manage the presentation of systems' responses by using multiple media provide an opportunity to fashion a style of human-computer interaction that draws much more heavily on our experience with human-human interactions.

PRIOR RELATED ART

Existing approaches and technologies for distributed computing include distributed objects, mobile objects, blackboard-style architectures, and agent-based software engineering.

The Distributed Object Approach

Object-oriented languages, such as C++ or JAVA, provide significant advances over standard procedural languages with respect to the reusability and modularity of code: *encapsulation*, *inheritance* and *polymorhpism*. Encapsulation encourages the creation of library interfaces that minimize dependencies on underlying algorithms or data structures. Changes to programming internals can be made at a later date with requiring modifications to the code that uses the library. Inheritance permits the extension and modification of a library of routines and data without requiring source code to the original library. Polymorphism allows one body of code to work on an arbitrary number of data types. For the sake of simplicity traditional objects may be seen to contain both methods and data. Methods provide the mechanisms by which the internal state of an object may be modified or by which communication may occur with another object or by which the instantiation or removal of objects may be directed.

With reference to Figure 2, a distributed object technology based around an Object Request Broker will now be described. Whereas "standard" object-oriented programming (OOP) languages can be used to build monolithic programs out of many object building blocks, distributed object technologies (DOOP) allow the creation of programs whose components may be spread across multiple machines. As shown in Figure 2, an object system 200 includes client objects 210 and server objects 220. To implement a client-server relationship between objects, the distributed object system

200 uses a registry mechanism (CORBA's registry is called an Object Request Broker, or ORB) 230 to store the interface descriptions of available objects. Through the services of the ORB 230, a client can transparently invoke a method on a remote server object. The ORB 230 is then responsible for finding the object 220 that can implement the request, passing it the parameters, invoking its method, and returning the results. In the most sophisticated systems, the client 210 does not have to be aware of where the object is located, its programming language, its operating system, or any other system aspects that are not part of the server object's interface.

Although distributed objects offer a powerful paradigm for creating networked applications, certain aspects of the approach are not perfectly tailored to the constantly changing environment of the Internet. A major restriction of the DOOP approach is that the interactions among objects are fixed through explicitly coded instructions by the application developer. It is often difficult to reuse an object in a new application without bringing along all its inherent dependencies on other objects (embedded interface definitions and explicit method calls). Another restriction of the DOOP approach is the result of its reliance on a remote procedure call (RPC) style of communication. Although easy to debug, this single thread of execution model does not facilitate programming to exploit the potential for parallel computation that one would expect in a distributed environment. In addition, RPC uses a blocking (synchronous) scheme that does not scale well for high-volume transactions.

Mobile Objects

Mobile objects, sometimes called mobile agents, are bits of code that can move to another execution site (presumably on a different machine) under their own programmatic control, where they can then interact with the local environment. For certain types of problems, the mobile object paradigm offers advantages over more traditional distributed object approaches. These advantages include network bandwidth and parallelism. Network bandwidth advantages exist for some database queries or electronic commerce applications, where it is more efficient to perform tests on data by bringing the tests to the data than by bringing large amounts of data to the testing program. Parallelism advantages include situations in which mobile agents can be spawned in parallel to accomplish many tasks at once.

Some of the disadvantages and inconveniences of the mobile agent approach include the programmatic specificity of the agent interactions, lack of coordination support between participant agents and execution environment irregularities regarding specific programming languages supported by host processors upon which agents reside. In a fashion similar to that of DOOP programming, an agent developer must programmatically specify where to go and how to interact with the target environment. There is generally little coordination support to encourage interactions among multiple (mobile) participants. Agents must be written in the programming language supported by the execution environment, whereas many other distributed technologies support heterogeneous communities of components, written in diverse programming languages.

Blackboard Architectures

Blackboard architectures typically allow multiple processes to communicate by reading and writing tuples from a global data store. Each process can watch for items of interest, perform computations based on the state of the blackboard, and then add partial results or queries that other processes can consider. Blackboard architectures provide a flexible framework for problem solving by a dynamic community of distributed processes. A blackboard architecture provides one solution to eliminating the tightly bound interaction links that some of the other distributed technologies require during interprocess communication. This advantage can also be a disadvantage: although a programmer does not need to refer to a specific process during computation, the framework does not provide programmatic control for doing so in cases where this would be practical.

Agent-based Software Engineering

Several research communities have approached distributed computing by casting it as a problem of modeling communication and cooperation among autonomous entities, or agents. Effective communication among independent agents requires four components: (1) a transport mechanism carrying messages in an asynchronous fashion, (2) an interaction protocol defining various types of communication interchange and their social implications (for instance, a response is expected of a question), (3) a content language permitting the expression and interpretation of utterances, and (4) an agreed-upon set of shared vocabulary and meaning for concepts (often called an *ontology*). Such mechanisms permit a much richer style of interaction among participants than can be expressed using a distributed object's RPC model or a blackboard architecture's centralized exchange approach.

Agent-based systems have shown much promise for flexible, fault-tolerant, distributed problem solving. Several agent-based projects have helped to evolve the notion of facilitation. However, existing agent-based technologies and architectures are typically very limited in the extent to which agents can specify complex goals or influence the strategies used by the facilitator. Further, such prior systems are not sufficiently attuned to the importance of integrating human agents (i.e., users) through natural language and other human-oriented user interface technologies.

The initial version of SRI International's Open Agent Architecture™ ("*OAA*®") technology provided only a very limited mechanism for dealing with compound goals. Fixed formats were available for specifying a flat list of either conjoined (AND) sub-goals or disjoined (OR) sub-goals; in both cases, parallel goal solving was hard-wired in, and only a single set of parameters for the entire list could be specified. More complex goal expressions involving (for example) combinations of different boolean connectors, nested expressions, or conditionally interdependent ("IF .. THEN") goals were not supported. Further, system scalability was not adequately addressed in this prior work.

SUMMARY OF INVENTION

A first embodiment of the present invention discloses a highly flexible, software-based architecture for constructing distributed systems. The architecture supports cooperative task completion by flexible, dynamic configurations of autonomous electronic agents. Communication and cooperation between agents are brokered by one or more facilitators, which are responsible for matching requests, from users and agents, with descriptions of the capabilities of other agents. It is not generally required that a user or agent know the identities, locations, or number of other agents involved in satisfying a request, and relatively minimal effort is involved in incorporating new agents and "wrapping" legacy applications. Extreme flexibility is achieved through an architecture organized around the declaration of capabilities by service-providing agents, the construction of arbitrarily complex goals by users and service-requesting agents, and the role of facilitators in delegating and coordinating the satisfaction of these goals, subject to advice and constraints that may accompany them. Additional mechanisms and features include facilities for creating and maintaining shared repositories of data; the use of triggers to instantiate commitments within and between agents; agent-based provision of multi-modal user interfaces, including natural language; and built-in support for including the user as a privileged member of the agent community. Specific embodiments providing enhanced scalability are also described.

BRIEF DESCRIPTION OF THE DRAWINGS

Prior Art

Prior Art FIGURE 1 depicts a networked computing model;

Prior Art FIGURE 2 depicts a distributed object technology based around an Object Resource Broker;

Examples of the Invention

FIGURE 3 depicts a distributed agent system based around a facilitator agent;

FIGURE 4 presents a structure typical of one small system of the present invention;

FIGURE 5 depicts an Automated Office system implemented in accordance with an example embodiment of the present invention supporting a mobile user with a laptop computer and a telephone;

FIGURE 6 schematically depicts an Automated Office system implemented as a network of agents in accordance with a preferred embodiment of the present invention;

FIGURE 7 schematically shows data structures internal to a facilitator in accordance with a preferred embodiment of the present invention;

FIGURE 8 depicts operations involved in instantiating a client agent with its parent facilitator in accordance with a preferred embodiment of the present invention;

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™ 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.

BRIEF DESCRIPTION OF THE APPENDICES

The Appendices provide source code for an embodiment of the present invention written in the PROLOG programming language.

APPENDIX A: Source code file named compound.pl.

APPENDIX B: Source code file named fac.pl.

APPENDIX C: Source code file named libcom_tcp.pl.

APPENDIX D: Source code file named liboaa.pl.

APPENDIX E: Source code file named translations.pl.

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

shown in Figure 4. All agents that are not facilitators are referred to herein generically as *client* agents -- so called because each acts (in some respects) as a client of some facilitator, which provides communication and other essential services for the client.

The variety of possible client agents is essentially unlimited. Some typical categories of client agents would include application agents 404, meta-agents 406, and user interface agents 408, as depicted in Figure 4. Application agents 404 denote specialists that provide a collection of services of a particular sort. These services could be domain-independent technologies (such as speech recognition, natural language processing 410, email, and some forms of data retrieval and data mining) or user-specific or domain-specific (such as a travel planning and reservations agent). Application agents may be based on legacy applications or libraries, in which case the agent may be little more than a wrapper that calls a pre-existing API 412, for example. Meta-agents 406 are agents whose role is to assist the facilitator agent 402 in coordinating the activities of other agents. While the facilitator 402 possesses domain-independent coordination strategies, meta-agents 406 can augment these by using domain- and application-specific knowledge or reasoning (including but not limited to rules, learning algorithms and planning).

With further reference to Figure 4, user interface agents 408 can play an extremely important and interesting role in certain embodiments of the present invention. By way of explanation, in some systems, a user interface agent can be implemented as a collection of "micro-agents", each monitoring a different input modality (point-and-click, handwriting, pen gestures, speech), and collaborating to produce the best interpretation of the current inputs. These micro-agents are depicted in Figure 4, for example, as Modality Agents 414. While describing such subcategories of client agents is useful for purposes of illustration and understanding, they need not be formally distinguished within the system in preferred implementations of the present invention.

The operation of one preferred embodiment of the present invention will be discussed in greater detail below, but may be briefly outlined as follows. When invoked, a client agent makes a connection to a facilitator, which is known as its *parent facilitator*. These connections are depicted as a double headed arrow between the client agent and the facilitator agent in Figure 3 and 4, for example. Upon connection, an agent registers with its parent facilitator a specification of the capabilities and services it can provide. For example, a natural language agent may register the characteristics of its available natural language vocabulary. (For more details regarding client agent connections, see the discussion of Figure 8 below.) Later during task completion, when a facilitator determines that the registered services 416 of one of its client agents will help satisfy a goal, the facilitator sends that client a request expressed in the Interagent Communication Language (*ICL*) 418. (See Figure 11 below for a more detailed discussion of the facilitator operations involved.) The agent parses this request, processes it, and returns answers or status reports to the facilitator. In processing a request, the client agent can make use of a variety of infrastructure capabilities provided in the preferred embodiment. For example, the client agent can use *ICL* 418 to request services of other agents, set triggers, and read or write shared data on the facilitator or other client agents that maintain shared data. (See the discussion of Figures 9-11 below for a more detailed discussion of request processing.)

The functionality of each client agent are made available to the agent community through registration of the client agent's capabilities with a facilitator 402. A software "wrapper" essentially surrounds the underlying application program performing the services offered by each client. The common infrastructure for constructing agents is preferably supplied by an *agent library.* The agent library is preferably accessible in the runtime environment of several different programming languages. The agent library preferably minimizes the effort required to construct a new system and maximizes the ease with which legacy systems can be "wrapped" and made compatible with the agent-based architecture of the present invention.

By way of further illustration, a representative application is now briefly presented with reference to Figures 5 and 6. In the Automated Office system depicted in Figure 5, a mobile user with a telephone and a laptop computer can access and task commercial applications such as calendars, databases, and email systems running back at the office. A user interface (UI) agent 408, shown in Figure 6, runs on the user's local laptop and is responsible for accepting user input, sending requests to the facilitator 402 for delegation to appropriate agents, and displaying the results of the distributed computation. The user may interact directly with a specific remote application by clicking on active areas in the interface, calling up a form or window for that application, and making queries with standard interface dialog mechanisms. Conversely, a user may express a task to be executed by using typed, handwritten, or spoken (over the telephone) English sentences, without explicitly specifying which agent or agents should perform the task.

For instance, if the question "What is my schedule?" is written 420 in the user interface 408, this request will be sent 422 by the UI 408 to the facilitator 402, which in turn will ask 424 a natural language (NL) agent 426 to translate the query into *ICL* 18. To accomplish this task, the NL agent 426 may itself need to make requests of the agent community to resolve unknown words such as "me" 428 (the UI agent 408 can respond 430 with the name of the current user) or "schedule" 432 (the calendar agent 434 defines this word 436). The resulting *ICL* expression is then routed by the facilitator 402 to appropriate agents (in this case, the calendar agent 434) to execute the request. Results are sent back 438 to the UI agent 408 for display.

The spoken request "When mail arrives for me about security, notify me immediately." produces a slightly more complex example involving communication among all agents in the system. After translation into *ICL* as described above, the facilitator installs a trigger 440 on the mail agent 442 to look for new messages about security. When one such message does arrive in its mail spool, the trigger fires, and the facilitator matches the action part of the trigger to capabilities published by the notification agent 446. The notification agent 446 is a meta-agent, as it makes use of rules concerning the optimal use of different output modalities (email, fax, speech generation over the telephone) plus information about an individual user's preferences 448 to determine the best way of relaying a message through available media transfer application agents. After some competitive parallelism to locate the user (the calendar agent 434 and database agent 450 may have different guesses as to where to find the user) and some cooperative parallelism to produce required information (telephone number of location, user password, and an audio file containing a text-to-speech representation of the email message), a telephone agent 452 calls the user, verifying its identity through touchtones, and then play the message.

The above example illustrates a number of inventive features. As new agents connect to the facilitator, registering capability specifications and natural language vocabulary, what the user can say and do dynamically changes; in other words, the ICL is dynamically ***expandable***. For example, adding a calendar agent to the system in the previous example and registering its capabilities enables users to ask natural language questions about their "schedule" without any need to revise code for the facilitator, the natural language agents, or any other client agents. In addition, the interpretation and execution of a task is a distributed process, with no single agent defining the set of possible inputs to the system. Further, a single request can produce cooperation and flexible communication among many agents, written in different programming languages and spread across multiple machines.

Design Philosophy and Considerations

One preferred embodiment provides an integration mechanism for heterogeneous applications in a distributed infrastructure, incorporating some of the dynamism and extensibility of blackboard approaches, the efficiency associated with mobile objects, plus the rich and complex interactions of communicating agents. Design goals for preferred embodiments of the present invention may be categorized under the general headings of *interoperation and cooperation, user interfaces*, and *software engineering*. These design goals are not absolute requirements, nor will they necessarily be satisfied by all embodiments of the present invention, but rather simply reflect the inventor's currently preferred design philosophy.

Versatile mechanisms of interoperation and cooperation

Interoperation refers to the ability of distributed software components - agents - to communicate meaningfully. While every system-building framework must provide mechanisms of interoperation at some level of granularity, agent-based frameworks face important new challenges in this area. This is true primarily because autonomy, the hallmark of *individual* agents, necessitates greater flexibility in interactions within *communities* of agents. *Coordination* refers to the mechanisms by which a community of agents is able to work together productively on some task. In these areas, the goals for our framework are to *provide flexibility in assembling* *communities of autonomous service providers, provide flexibility in structuring cooperative interactions, impose the right amount of structure,* as well as *include legacy and "owned-elsewhere" applications.*

Provide flexibility in assembling communities of autonomous service providers -- both at development time and at runtime. Agents that conform to the linguistic and ontological requirements for effective communication should be able to participate in an agent community, in various combinations, with minimal or near minimal prerequisite knowledge of the characteristics of the other players. Agents with duplicate and overlapping capabilities should be able to coexist within the same community, with the system making optimal or near optimal use of the redundancy.

Provide flexibility in structuring cooperative interactions among the members of a community of agents. A framework preferably provides an economical mechanism for setting up a variety of interaction patterns among agents, without requiring an inordinate amount of complexity or infrastructure within the individual agents. The provision of a service should be independent or minimally dependent upon a particular configuration of agents.

Impose the right amount of structure on individual agents. Different approaches to the construction of multi-agent systems impose different requirements on the individual agents. For example, because KQML is neutral as to the content of messages, it imposes minimal structural requirements on individual agents. On the other hand, the BDI paradigm tends to impose much more demanding requirements, by making assumptions about the nature of the programming elements that are meaningful to individual agents. Preferred embodiments of the present invention should fall somewhere between the two, providing a rich set of interoperation and coordination capabilities, without precluding any of the software engineering goals defined below.

Include legacy and "owned-elsewhere" applications. Whereas *legacy* usually implies reuse of an established system fully controlled by the agent-based system developer, *owned-elsewhere* refers to applications to which the developer has partial access, but no control. Examples of owned-elsewhere applications include data sources and services available on the World Wide Web, via simple form-based interfaces, and applications used cooperatively within a virtual enterprise, which remain the properties of separate corporate entities. Both classes of application must preferably be able to interoperate, more or less as full-fledged members of the agent community, without requiring an overwhelming integration effort.

Human-oriented user interfaces

Systems composed of multiple distributed components, and possibly dynamic configurations of components, require the crafting of intuitive user interfaces to *provide conceptually natural interaction mechanisms*, *treat users as privileged members of the agent community* and *support collaboration.*

Provide conceptually natural interaction mechanisms with multiple distributed components. When there are numerous disparate agents, and/or complex tasks implemented by the system, the user should be able to express requests without having detailed knowledge of the individual agents. With speech recognition, handwriting recognition, and natural language technologies becoming more mature, agent architectures should preferably support these forms of input playing increased roles in the tasking of agent communities.

Preferably treat *users as privileged members* of the agent community by providing an appropriate level of task specification within *software* agents, and reusable translation mechanisms between this level and the level of *human* requests, supporting constructs that seamlessly incorporate interactions between both human-interface and software types of agents.

Preferably support *collaboration* (simultaneous work over shared data and processing resources) between users and agents.

Realistic software engineering requirements

System-building frameworks should preferably address the practical concerns of real-world applications by the specification of requirements which preferably include: *Minimize the effort* required to create new agents, and to wrap existing applications. *Encourage reuse,* both of domain-independent and domain-specific components. The concept of *agent orientation*, like that of object orientation, provides a natural conceptual framework for reuse, so long as mechanisms for encapsulation and interaction are structured appropriately. *Support lightweight, mobile platforms.* Such platforms should be able to serve as hosts for agents, without requiring the installation of a massive environment. It should also be possible to construct individual agents that are relatively small and modest in their processing requirements. *Minimize platform and language barriers.* Creation of new agents, as well as wrapping of existing applications, should not require the adoption of a new language or environment.

Mechanisms of Cooperation

Cooperation among agents in accordance with the present invention is preferably achieved via messages expressed in a common language, *ICL*. Cooperation among agent is further preferably structured around a three-part approach: providers of services register capabilities specifications with a facilitator, requesters of services construct goals and relay them to a facilitator, and facilitators coordinate the efforts of the appropriate service providers in satisfying these goals.

The Interagent Communication Language (ICL)

Interagent Communication Language (*"ICL"*) 418 refers to an interface, communication, and task coordination language preferably shared by all agents, regardless of what platform they run on or what computer language they are programmed in. *ICL* may be used by an agent to task itself or some subset of the agent community. Preferably, *ICL* allows agents to specify explicit control parameters while simultaneously supporting expression of goals in an underspecified, loosely constrained manner. In a further preferred embodiment, agents employ *ICL* to perform queries, execute actions, exchange information, set triggers, and manipulate data in the agent community.

In a further preferred embodiment, a program element expressed in *ICL* is the *event*. The activities of every agent, as well as communications between agents, are preferably structured around the transmission and handling of events. In communications, events preferably serve as messages between agents; in regulating the activities of individual agents, they may preferably be thought of as goals to be satisfied. Each event preferably has a type, a set of parameters, and content. For example, the agent library procedure *oaa_Solve* can be used by an agent to request services of other agents. A call to *oaa_Solve*, within the code of agent *A*, results in an event having the form

ev_post_solve(Goal, Params)

going from *A* to the facilitator, where *ev_post_solve* is the type, *Goal* is the content, and *Params* is a list of parameters. The allowable content and parameters preferably vary according to the type of the event.

The *ICL* preferably includes a layer of conversational protocol and a content layer. The conversational layer of *ICL* is defined by the event types, together with the parameter lists associated with certain of these event types. The content layer consists of the specific goals, triggers, and data elements that may be embedded within various events.

The *ICL* conversational protocol is preferably specified using an orthogonal, parameterized approach, where the conversational aspects of each element of an interagent conversation are represented by a selection of an event type and a selection of values from at least one orthogonal set of parameters. This approach offers greater expressiveness than an approach based solely on a fixed selection of *speech acts*, such as embodied in KQML. For example, in KQML, a request to satisfy a query can employ either of the performatives *ask_all* or *ask_one*. In *ICL*, on the other hand, this type of request preferably is expressed by the event type *ev_post_solve*, together with the *solution_limit(N)* parameter - where *N* can be any positive integer. (A request for all solutions is indicated by the omission of the *solution_limit* parameter.) The request can also be accompanied by other parameters, which combine to further refine its semantics. In KQML, then, this example forces one to choose between two possible conversational options, neither of which may be precisely what is desired. In either case, the performative chosen is a single value that must capture the entire conversational characterization of the communication. This requirement raises a difficult challenge for the language designer, to select a set of performatives that provides the desired functionality without becoming unmanageably large. Consequently, the debate over the right set of performatives has consumed much discussion within the KQML community.

The content layer of the *ICL* preferably supports unification and other features found in logic programming language environments such as PROLOG. In some embodiments, the content layer of the *ICL* is simply an extension of at least one programming language. For example, the Applicants have found that PROLOG is suitable for implementing and extending into the content layer of the *ICL*. The agent libraries preferably provide support for constructing, parsing, and manipulating *ICL* expressions. It is possible to embed content expressed in other languages within an *ICL* event. However, expressing content in *ICL* simplifies the facilitator's access to the content, as well as the conversational layer, in delegating requests. This gives the facilitator more information about the nature of a request and helps the facilitator decompose compound requests and delegate the sub-requests.

Further, *ICL* expressions preferably include, in addition to events, at least one of the following: capabilities declarations, requests for services, responses to requests, trigger specifications, and shared data elements. A further preferred embodiment of the present invention incorporates *ICL* expressions including at least all of the following: events, capabilities declarations, requests for services, responses to requests, trigger specifications, and shared data elements.

Providing Services: Specifying "Solvables"

In a preferred embodiment of the present invention, every participating agent defines and publishes a set of capability declarations, expressed in *ICL*, describing the services that it provides. These declarations establish a high-level interface to the agent. This interface is used by a facilitator in communicating with the agent, and, most important, in delegating service requests (or parts of requests) to the agent. Partly due to the use of PROLOG as a preferred basis for *ICL*, these capability declarations are referred as *solvables*. The agent library preferably provides a set of procedures allowing an agent to add, remove, and modify its solvables, which it may preferably do at any time after connecting to its facilitator.

There are preferably at least two major types of solvables: *procedure* solvables and *data* solvables. Intuitively, a procedure solvable performs a test or action, whereas a data solvable provides access to a collection of data. For example, in creating an agent for a mail system, procedure solvables might be defined for sending a message to a person, testing whether a message about a particular subject has arrived in the mail queue, or displaying a particular message onscreen. For a database wrapper agent, one might define a distinct data solvable corresponding to each of the relations present in the database. Often, a data solvable is used to provide a *shared* data store, which may be not only queried, but also updated, by various agents having the required permissions.

There are several primary technical differences between these two types of solvables. First, each procedure solvable must have a handler declared and defined for it, whereas this is preferably not necessary for a data solvable. The handling of requests for a data solvable is preferably provided transparently by the agent library. Second, data solvables are preferably associated with a dynamic collection of facts (or clauses), which may be further preferably modified at runtime, both by the agent providing the solvable, and by other agents (provided they have the required permissions). Third, special features, available for use with data solvables, preferably facilitate maintaining the associated facts. In spite of these differences, it should be noted that the mechanism of *use* by which an agent requests a service is the same for the two types of solvables.

In one embodiment, a request for one of an agent's services normally arrives in the form of an event from the agent's facilitator. The appropriate handler then deals with this event. The handler may be coded in whatever fashion is most appropriate, depending on the nature of the task, and the availability of task-specific libraries or legacy code, if any. The only hard requirement is that the handler return an appropriate response to the request, expressed in *ICL*. Depending on the nature of the request, this response could be an indication of success or failure, or a list of solutions (when the request is a data query).

A solvable preferably has three parts: a *goal*, a list of *parameters*, and a list of *permissions*, which are declared using the format:

solvable(Goal, Parameters, Permissions)

The goal of a solvable, which syntactically takes the preferable form of an *ICL* structure, is a logical representation of the service provided by the solvable. (An *ICL* structure consists of a *functor* with 0 or more arguments. For example, in the structure a(b,c), `a' is the functor, and `b' and `c' the arguments.) As with a PROLOG structure, the goal's arguments themselves may preferably be structures.

Various options can be included in the parameter list, to refine the semantics associated with the solvable. The *type* parameter is preferably used to say whether the solvable is *data* or *procedure*. When the type is *procedure*, another parameter may be used to indicate the handler to be associated with the solvable. Some of the parameters appropriate for a *data* solvable are mentioned elsewhere in this application. In either case (procedure or data solvable), the *private* parameter may be preferably used to restrict the use of a solvable to the declaring agent when the agent intends the solvable to be solely for its internal use but wishes to take advantage of the mechanisms in accordance with the present invention to access it, or when the agent wants the solvable to be available to outside agents only at selected times. In support of the latter case, it is preferable for the agent to change the status of a solvable from private to non-private at any time.

The permissions of a solvable provide mechanisms by which an agent may preferably control access to its services allowing the agent to restrict calling and writing of a solvable to itself and/or other selected agents. (*Calling* means requesting the service encapsulated by a solvable, whereas *writing* means modifying the collection of facts associated with a data solvable.) The default permission for every solvable in a further preferred embodiment of the present invention is to be callable by anyone, and for data solvables to be writable by anyone. A solvable's permissions can preferably be changed at any time, by the agent providing the solvable.

For example, the solvables of a simple email agent might include:

```
solvable(send_message(email, +ToPerson, +Params),
        [type(procedure), callback(send_mail)],
           [])
solvable(last_message(email, -MessageId),
           [type(data), single_value(true)],
           [write(true)]),
        solvable(get_message(email, +MessageId, -
Msg),
            [type(procedure), callback(get_mail)],
       [])
```

The symbols `+' and `-', indicating input and output arguments, are at present used only for purposes of documentation. Most parameters and permissions have default values, and specifications of default values may be omitted from the parameters and permissions lists.

Defining an agent's capabilities in terms of solvable declarations effectively creates a vocabulary with which other agents can communicate with the new agent. Ensuring that agents will speak the same language and share a common, unambiguous semantics of the vocabulary involves *ontology*. Agent development tools and services (automatic translations of solvables by the facilitator) help address this issue; additionally, a preferred embodiment of the present invention will typically rely on vocabulary from either formally engineered ontologies for specific domains or from ontologies constructed during the incremental development of a body of agents for several applications or from both specific domain ontologies and incrementally developed ontologies. Several example tools and services are described in Cheyer et al.'s paper entitled "Development Tools for the Open Agent Architecture," as presented at the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM 96), London, April 1996.

Although the present invention imposes no hard restrictions on the form of solvable declarations, two common usage conventions illustrate some of the utility associated with solvables.

Classes of services are often preferably tagged by a particular type. For instance, in the example above, the "last_message" and "get_message" solvables are specialized for email, not by modifying the *names* of the services, but rather by the use of the `email' parameter, which serves during the execution of an *ICL* request to select (or not) a specific type of message.

Actions are generally written using an imperative verb as the functor of the solvable in a preferred embodiment of the present invention, the direct object (or item class) as the first argument of the predicate, required arguments following, and then an extensible parameter list as the last argument. The parameter list can hold optional information usable by the function. The *ICL* expression generated by a natural language parser often makes use of this parameter list to store prepositional phrases and adjectives.

As an illustration of the above two points, "Send mail to Bob about lunch" will be translated into an *ICL* request send_message(email, `Bob Jones', [subject(lunch)]), whereas "Remind Bob about lunch" would leave the transport unspecified (send_message(KIND, `Bob Jones', [subject(lunch)])), enabling all available message transfer agents (e.g., fax, phone, mail, pager) to compete for the opportunity to carry out the request.

Requesting Services

An agent preferably requests services of the community of agent by delegating tasks or goals to its facilitator. Each request preferably contains calls to one or more agent solvables, and optionally specifies parameters containing advice to help the facilitator determine how to execute the task. Calling a solvable preferably does *not* require that the agent specify (or even know of) a particular agent or agents to handle the call. While it is possible to specify one or more agents using an address parameter (and there are situations in which this is desirable), in general it is advantageous to leave this delegation to the facilitator. This greatly reduces the hard-coded component dependencies often found in other distributed frameworks. The agent libraries of a preferred embodiment of the present invention provide an agent with a single, unified point of entry for requesting services of other agents: the library procedure *oaa_Solve*. In the style of logic programming, *oaa_Solve* may preferably be used both to retrieve data and to initiate actions, so that calling a *data* solvable looks the same as calling a *procedure* solvable.

Complex Goal Expressions

A powerful feature provided by preferred embodiments of the present invention is the ability of a client agent (or a user) to submit compound goals of an arbitrarily complex nature to a facilitator. A compound goal is a single goal expression that specifies multiple sub-goals to be performed. In speaking of a *"complex goal expression"* we mean that a single goal expression that expresses multiple sub-goals can potentially include more than one type of logical connector (e.g., AND, OR, NOT), and/or more than one level of logical nesting (e.g., use of parentheses), or the substantive equivalent. By way of further clarification, we note that when speaking of an *"arbitrarily complex goal expression"* we mean that goals are expressed in a language or syntax that allows expression of such complex goals when appropriate or when desired, not that every goal is itself necessarily complex.

It is contemplated that this ability is provided through an interagent communication language having the necessary syntax and semantics. In one example, the goals may take the form of compound goal expressions composed using operators similar to those employed by PROLOG, that is, the comma for conjunction, the semicolon for disjunction, the arrow for conditional execution, etc. The present invention also contemplates significant extensions to PROLOG syntax and semantics. For example, one embodiment incorporates a "parallel disjunction" operator indicating that the disjuncts are to be executed by different agents concurrently. A further embodiment supports the specification of whether a given sub-goal is to be executed breadth-first or depth-first.

A further embodiment supports each sub-goal of a compound goal optionally having an address and/or a set of parameters attached to it. Thus, each sub-goal takes the form

Address:Goal::Parameters

where both *Address* and *Parameters* are optional.

An address, if present, preferably specifies one or more agents to handle the given goal, and may employ several different types of referring expression: unique names, symbolic names, and shorthand names. Every agent has preferably a unique name, assigned by its facilitator, which relies upon network addressing schemes to ensure its global uniqueness. Preferably, agents also have self-selected symbolic names (for example, "mail"), which are not guaranteed to be unique. When an address includes a symbolic name, the facilitator preferably takes this to mean that all agents having that name should be called upon. Shorthand names include `self' and `parent' (which refers to the agent's facilitator). The address associated with a goal or sub-goal is preferably always optional. When an address is not present, it is the facilitator's job to supply an appropriate address.

The distributed execution of compound goals becomes particularly powerful when used in conjunction with natural language or speech-enabled interfaces, as the query itself may specify how functionality from distinct agents will be combined. As a simple example, the spoken utterance "Fax it to Bill Smith's manager." can be translated into the following compound *ICL* request:

oaa_Solve((manager('Bill Smith', M), fax(it,M,[])), [strategy(action)])

Note that in this ICL request there are two sub-goals, "manager('Bill Smith',M)" and "fax(it,M,[])," and a single global parameter "strategy(action)." According to the present invention, the facilitator is capable of mapping global parameters in order to apply the constraints or advice across the separate sub-goals in a meaningful way. In this instance, the global parameter strategy(action) implies a parallel constraint upon the first sub-goal; i.e., when there are multiple agents that can respond to the manager sub-goal, each agent should receive a request for service. In contrast, for the second sub-goal, parallelism should not be inferred from the global parameter strategy(action) because such an inference would possibly result in the transmission of duplicate facsimiles.

Refining Service Requests

In a preferred embodiment of the present invention, parameters associated with a goal (or sub-goal) can draw on useful features to refine the request's meaning. For example, it is frequently preferred to be able to specify whether or not solutions are to be returned synchronously; this is done using the *reply* parameter, which can take any of the values *synchronous, asynchronous*, or *none*. As another example, when the goal is a non-compound query of a data solvable, the *cache* parameter may preferably be used to request local caching of the facts associated with that solvable. Many of the remaining parameters fall into two categories: feedback and advice.

Feedback parameters allow a service requester to receive information from the facilitator about how a goal was handled. This feedback can include such things as the identities of the agents involved in satisfying the goal, and the amount of time expended in the satisfaction of the goal.

Advice parameters preferably give constraints or guidance to the facilitator in completing and interpreting the goal. For example, a *solution_limit* parameter preferably allows the requester to say how many solutions it is interested in; the facilitator and/or service providers are free to use this information in optimizing their efforts. Similarly, a *time_limit* is preferably used to say how long the requester is willing to wait for solutions to its request, and, in a multiple facilitator system, a *level_limit* may preferably be used to say how remote the facilitators may be that are consulted in the search for solutions. A *priority* parameter is preferably used to indicate that a request is more urgent than previous requests that have not yet been satisfied. Other preferred advice parameters include but are not limited to parameters used to tell the facilitator whether parallel satisfaction of the parts of a goal is appropriate, how to combine and filter results arriving from multiple solver agents, and whether the requester itself may be considered a candidate solver of the sub-goals of a request.

Advice parameters preferably provide an extensible set of low-level, orthogonal parameters capable of combining with the *ICL* goal language to fully express how information should flow among participants. In certain preferred embodiments of the present invention, multiple parameters can be grouped together and given a group name. The resulting *high-level advice parameters* can preferably be used to express concepts analogous to KQML's performatives, as well as define classifications of problem types. For instance, KQML's "ask_all" and "ask_one" performatives would be represented as combinations of values given to the parameters *reply*, *parallel_ok*, and *solution_limit*. As an example of a higher-level problem type, the strategy "math_problem" might preferably send the query to all appropriate math solvers in parallel, collect their responses, and signal a conflict if different answers are returned. The strategy "essay_question" might preferably send the request to all appropriate participants, and signal a problem (i.e., cheating) if any of the returned answers are identical.

Facilitation

In a preferred embodiment of the present invention, when a facilitator receives a compound goal, its job is to construct a goal satisfaction plan and oversee its satisfaction in an optimal or near optimal manner that is consistent with the specified advice. The facilitator of the present invention maintains a knowledge base that records the capabilities of a collection of agents, and uses that knowledge to assist requesters and providers of services in making contact.

Figure 7 schematically shows data structures 700 internal to a facilitator in accordance with one embodiment of the present invention. Consider the function of a Agent Registry 702 in the present invention. Each registered agent may be seen as associated with a collection of fields found within its parent facilitator such as shown in the figure. Each registered agent may optionally possess a Symbolic Name which would be entered into field 704. As mentioned elsewhere, Symbolic Names need not be unique to each instance of an agent. Note that an agent may in certain preferred embodiments of the present invention possess more than one Symbolic Name. Such Symbolic Names would each be found through their associations in the Agent Registry entries. Each agent, when registered, must possess a Unique Address, which is entered into the Unique Address field 706.

With further reference to Figure 7, each registered agent may be optionally associated with one or more capabilities, which have associated Capability Declaration fields 708 in the parent facilitator Agent Registry 702. These capabilities may define not just functionality, but may further provide a utility parameter indicating, in some manner (e.g., speed, accuracy, etc), how effective the agent is at providing the declared capability. Each registered agent may be optionally associated with one or more data components, which have associated Data Declaration fields 710 in the parent facilitator Agent Registry 702. Each registered agent may be optionally associated with one or more triggers, which preferably could be referenced through their associated Trigger Declaration fields 712 in the parent facilitator Agent Registry 702. Each registered agent may be optionally associated with one or more tasks, which preferably could be referenced through their associated Task Declaration fields 714 in the parent facilitator Agent Registry 702. Each registered agent may be optionally associated with one or more Process Characteristics, which preferably could be referenced through their associated Process Characteristics Declaration fields 716 in the parent facilitator Agent Registry 702. Note that these characteristics in certain preferred embodiments of the present invention may include one or more of the following: Machine Type (specifying what type of computer may run the agent), Language (both computer and human interface).

A facilitator agent in certain preferred embodiments of the present invention further includes a Global Persistent Database 720. The database 720 is composed of data elements which do not rely upon the invocation or instantiation of client agents for those data elements to persist. Examples of data elements which might be present in such a database include but are not limited to the network address of the facilitator agent's server, facilitator agent's server accessible network port list, firewalls, user lists, and security options regarding the access of server resources accessible to the facilitator agent.

A simplified walk through of operations involved in creating a client agent, a client agent initiating a service request, a client agent responding to a service request and a facilitator agent responding to a service request are including hereafter by way of illustrating the use of such a system. These figures and their accompanying discussion are provided by way of illustration of one preferred embodiment of the present invention and are not intended to limit the scope of the present invention.

Figure 8 depicts operations involved in instantiating a client agent with its parent facilitator in accordance with a preferred embodiment of the present invention. The operations begin with starting the Agent Registration in a step 800. In a next step 802, the Installer, such as a client or facilitator agent, invokes a new client agent. It will be appreciated that any computer entity is capable of invoking a new agent. The system then instantiates the new client agent in a step 804. This operation may involve resource allocations somewhere in the network on a local computer system for the client agent, which will often include memory as well as placement of references to the newly instantiated client agent in internal system lists of agents within that local computing system. Once instantiated, the new client and its parent facilitator establish a communications link in a step 806. In certain preferred embodiments, this communications link involves selection of one or more physical transport mechanisms for this communication. Once established, the client agent transmits it profile to the parent facilitator in a step 808. When received, the parent facilitator registers the client agent in a step 810. Then, at a step 812, a client agent has been instantiated in accordance with one preferred embodiment of the present invention.

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 preferred embodiment of the present invention. The method of Figure 9 begins in a step 900, wherein any initialization or other such procedures may be performed. Then, in a step 902, the client agent determines a goal to be achieved (or solved). This goal is then translated in a step 904 into *ICL*, if it is not already formulated in it. The goal, now stated in *ICL*, is then transmitted to the client agent's parent facilitator in a step 906. The parent facilitator responds to this service request and at a later time, the client agent receives the results of the request in a step 908, operations of Figure 9 being complete in a done step 910.

FIGURE 10 depicts operations involved in a client agent responding to a service request in accordance with a preferred embodiment of the present invention. Once started in a step 1000, the client agent receives the service request in a step 1002. In a next step 1004, the client agent parses the received request from ICL. The client agent then determines if the service is available in a step 1006. If it is not, the client agent returns a status report to that effect in a step 1008. If the service is available, control is passed to a step 1010 where the client performs the requested service. Note that in completing step 1010 the client may form complex goal expressions, requesting results for these solvables from the facilitator agent. For example, a fax agent might fax a document to a certain person only after requesting and receiving a fax number for that person. Subsequently, the client agent either returns the results of the service and/or a status report in a step 1012. The operations of Figure 10 are complete in a done step 1014.

FIGURE 11 depicts operations involved in a facilitator agent response to a service request in accordance with a preferred embodiment of the present invention. The start of such operations in step 1100 leads to the reception of a goal request in a step 1102 by the facilitator. This request is then parsed and interpreted by the facilitator in a step 1104. The facilitator then proceeds to construct a goal satisfaction plan in a next step 1106. In steps 1108 and 1110, respectively, the facilitator determines the required sub-goals and then selects agents suitable for performing the required sub-goals. The facilitator then transmits the sub-goal requests to the selected agents in a step 1112 and receives the results of these transmitted requests in a step 1114. It should be noted that the actual implementation of steps 1112 and 1114 are dependent upon the specific goal satisfaction plan. For instance, certain sub-goals may be sent to separate agents in parallel, while transmission of other sub-goals may be postponed until receipt of particular answers. Further, certain requests may generate multiple responses that generate additional sub-goals. Once the responses have been received, the facilitator determines whether the original requested goal has been completed in a step 1118. If the original requested goal has not been completed, the facilitator recursively repeats the operations 1106 through 1116. Once the original requested goal is completed, the facilitator returns the results to the requesting agent 1118 and the operations are done at 1120.

A further preferred embodiment of the present invention incorporates *transparent delegation*, which means that a requesting agent can generate a request, and a facilitator can manage the satisfaction of that request, without the requester needing to have any knowledge of the identities or locations of the satisfying agents. In some cases, such as when the request is a data query, the requesting agent may also be oblivious to the *number* of agents involved in satisfying a request. Transparent delegation is possible because agents' capabilities (solvables) are treated as an abstract description of a service, rather than as an entry point into a library or body of code.

A further preferred embodiment of the present invention incorporates facilitator handling of compound goals, preferably involving three types of processing: delegation, optimization and interpretation.

Delegation processing preferably supports facilitator determination of which specific agents will execute a compound goal and how such a compound goal's sub-goals will be combined and the sub-goal results routed. *Delegation* involves selective application of global and local constraint and advice parameters onto the specific sub-goals. *Delegation* results in a goal that is unambiguous as to its meaning and as to the agents that will participate in satisfying it.

Optimization processing of the completed goal preferably includes the facilitator using sub-goal parallelization where appropriate. *Optimization* results in a goal whose interpretation will require as few exchanges as possible, between the facilitator and the satisfying agents, and can exploit parallel efforts of the satisfying agents, wherever this does not affect the goal's meaning.

Interpretation processing of the optimized goal. Completing the addressing of a goal involves the selection of one or more agents to handle each of its sub-goals (that is, each sub-goal for which this selection has not been specified by the requester). In doing this, the facilitator uses its knowledge of the capabilities of its client agents (and possibly of other facilitators, in a multi-facilitator system). It may also use strategies or advice specified by the requester, as explained below. The *interpretation* of a goal involves the coordination of requests to the satisfying agents, and assembling their responses into a coherent whole, for return to the requester.

A further preferred embodiment of present invention extends facilitation so the facilitator can employ strategies and advice given by the requesting agent, resulting in a variety of interaction patterns that may be instantiated in the satisfaction of a request.

A further preferred embodiment of present invention handles the distribution of both data update requests and requests for installation of triggers, preferably using some of the same strategies that are employed in the delegation of service requests.

Note that the reliance on facilitation is not absolute; that is, there is no hard requirement that requests and services be matched up by the facilitator, or that interagent communications go through the facilitator. There is preferably support in the agent library for explicit addressing of requests. However, a preferred embodiment of the present invention encourages employment the paradigm of agent communities, minimizing their development effort, by taking advantage of the facilitator's provision of transparent delegation and handling of compound goals.

A facilitator is preferably viewed as a *coordinator*, not a controller, of cooperative task completion. A facilitator preferably never initiates an activity. A facilitator preferably responds to requests to manage the satisfaction of some goal, the update of some data repository, or the installation of a trigger by the appropriate agent or agents. All agents can preferably take advantage of the facilitator's expertise in delegation, and its up-to-date knowledge about the current membership of a dynamic community. The facilitator's coordination services often allows the developer to lessen the complexity of individual agents, resulting in a more manageable software development process, and enabling the creation of lightweight agents.

Maintaining Data Repositories

The agent library supports the creation, maintenance, and use of databases, in the form of data solvables. Creation of a data solvable requires only that it be declared. Querying a data solvable, as with access to any solvable, is done using *oaa_Solve*.

A data solvable is conceptually similar to a relation in a relational database. The facts associated with each solvable are maintained by the agent library, which also handles incoming messages containing queries of data solvables. The default behavior of an agent library in managing these facts may preferably be refined, using parameters specified with the solvable's declaration. For example, the parameter *single_value* preferably indicates that the solvable should only contain a single fact at any given point in time. The parameter *unique_values* preferably indicates that no duplicate values should be stored.

Other parameters preferably allow data solvables use of the concepts of ownership and persistence. For implementing shared repositories, it is often preferable to maintain a record of which agent created each fact of a data solvable with the creating agent being preferably considered the fact's owner. In many applications, it is preferable to remove an agent's facts when that agent goes offline (for instance, when the agent is no longer participating in the agent community, whether by deliberate termination or by malfunction). When a data solvable is declared to be non-persistent, its facts are automatically maintained in this way, whereas a persistent data solvable preferably retains its facts until they are explicitly removed.

A further preferred embodiment of present invention supports an agent library through procedures by which agents can update (add, remove, and replace) facts belonging to data solvables, either locally or on other agents, given that they have preferably the required permissions. These procedures may preferably be refined using many of the same parameters that apply to service requests. For example, the *address* parameter preferably specifies one or more particular agents to which the update request applies. In its absence, just as with service requests, the update request preferably goes to *all* agents providing the relevant data solvable. This default behavior can be used to maintain coordinated "mirror" copies of a data set within multiple agents, and can be useful in support of distributed, collaborative activities.

Similarly, the *feedback* parameters, described in connection with *oaa_Solve*, are preferably available for use with data maintenance requests.

A further preferred embodiment of present invention supports ability to provide data solvables not just to client agents, but also to facilitator agents. Data solvables can preferably created, maintained and used by a facilitator. The facilitator preferably can, at the request of a client of the facilitator, create, maintain and share the use of data solvables with all the facilitator's clients. This can be useful with relatively stable collections of agents, where the facilitator's workload is predictable.

Using a Blackboard Style of Communication

In a further preferred embodiment of present invention, when a data solvable is publicly readable and writable, it acts essentially as a global data repository and can be used cooperatively by a group of agents. In combination with the use of triggers, this allows the agents to organize their efforts around a "blackboard" style of communication.

As an example, the "DCG-NL" agent (one of several existing natural language processing agents), provides natural language processing services for a variety of its peer agents, expects those other agents to record, on the facilitator, the vocabulary to which they are prepared to respond, with an indication of each word's part of speech, and of the logical form (*ICL* sub-goal) that should result from the use of that word. In a further preferred embodiment of present invention, the NL agent, preferably when it comes online, preferably installs a data solvable for each basic part of speech on its facilitator. For instance, one such solvable would be:

solvable(noun(Meaning, Syntax), [], [])

Note that the empty lists for the solvable's permissions and parameters are acceptable here, since the default permissions and parameters provide appropriate functionality.

A further preferred embodiment of present invention incorporating an Office Assistant system as discussed herein or similar to the discussion here supports several agents making use of these or similar services. For instance, the database agent uses the following call, to library procedure *oaa_AddData*, to post the noun `boss', and to indicate that the "meaning" of boss is the concept `manager':

oaa_AddData(noun(manager, atom(boss)), [address(parent)])

Autonomous Monitoring with Triggers

A further preferred embodiment of present invention includes support for triggers, providing a general mechanism for requesting some action be taken when a set of conditions is met. Each agent can preferably install triggers either locally, for itself, or remotely, on its facilitator or peer agents. There are preferably at least four types of triggers: communication, data, task, and time. In addition to a type, each trigger preferably specifies at least a condition and an action, both preferably expressed in *ICL*. The condition indicates under what circumstances the trigger should fire, and the action indicates what should happen when it fires. In addition, each trigger can be set to fire either an unlimited number of times, or a specified number of times, which can be any positive integer.

Triggers can be used in a variety of ways within preferred embodiments of the present invention. For example, triggers can be used for monitoring external sensors in the execution environment, tracking the progress of complex tasks, or coordinating communications between agents that are essential for the synchronization of related tasks. The installation of a trigger within an agent can be thought of as a representation of that agent's *commitment* to carry out the specified action, whenever the specified condition holds true.

Communication triggers preferably allow any incoming or outgoing event (message) to be monitored. For instance, a simple communication trigger may say something like: "Whenever a solution to a goal is returned from the facilitator, send the result to the presentation manager to be displayed to the user."

Data triggers preferably monitor the state of a data repository (which can be maintained on a facilitator or a client agent). Data triggers' conditions may be tested upon the addition, removal, or replacement of a fact belonging to a data solvable. An example data trigger is: "When 15 users are simultaneously logged on to a machine, send an alert message to the system administrator."

Task triggers preferably contain conditions that are tested after the processing of each incoming event and whenever a timeout occurs in the event polling. These conditions may specify any goal executable by the local *ICL* interpreter, and most often are used to test when some solvable becomes satisfiable. Task triggers are useful in checking for task-specific internal conditions. Although in many cases such conditions are captured by solvables, in other cases they may not be. 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. An example task trigger is: "When mail arrives for me about security, notify me immediately."

Time triggers preferably monitor time conditions. For instance, an alarm trigger can be set to fire at a single fixed point in time (e.g., "On December 23rd at 3pm"), or on a recurring basis (e.g., "Every three minutes from now until noon").

Triggers are preferably implemented as data solvables, declared implicitly for every agent. When requesting that a trigger be installed, an agent may use many of the same parameters that apply to service and data maintenance requests.

A further preferred embodiment of present invention incorporates semantic support, in contrast with most programming methodologies, of the agent on which the trigger is installed only having to know how to evaluate the conditional part of the trigger, not the consequence. When the trigger fires, the action is delegated to the facilitator for execution. Whereas many commercial mail programs allow rules of the form "When mail arrives about XXX, [forward it, delete it, archive it]", the possible actions are hard-coded and the user must select from a fixed set.

A further preferred embodiment of present invention, the consequence of a trigger may be any compound goal executable by the dynamic community of agents. Since new agents preferably define both functionality and vocabulary, when an unanticipated agent (for example, a fax agent) joins the community, no modifications to existing code is required for a user to make use of it - "When mail arrives, fax it to Bill Smith."

The Agent Library

In a preferred embodiment of present invention, the agent library provides the infrastructure for constructing an agent-based system. The essential elements of protocol (involving the details of the messages that encapsulate a service request and its response) are preferably made transparent to simplify the programming applications. This enables the developer to focus functionality, rather than message construction details and communication details. For example, to request a service of another agent, an agent preferably calls the library procedure *oaa_Solve*. This call results in a message to a facilitator, which will exchange messages with one or more service providers, and then send a message containing the desired results to the requesting agent. These results are returned via one of the arguments of *oaa_Solve*. None of the messages involved in this scenario is explicitly constructed by the agent developer. Note that this describes the *synchronous* use of *oaa_Solve*.

In another preferred embodiment of present invention, an agent library provides both *intra*agent and *inter*agent infrastructure; that is, mechanisms supporting the internal structure of individual agents, on the one hand, and mechanisms of cooperative interoperation between agents, on the other. Note that most of the infrastructure cuts across this boundary with many of the same mechanisms supporting both agent internals and agent interactions in an integrated fashion. For example, services provided by an agent preferably can be accessed by that agent through the same procedure (*oaa_Solve*) that it would employ to request a service of another agent (the only difference being in the *address* parameter accompanying the request). This helps the developer to reuse code and avoid redundant entry points into the same functionality.

Both of the preferred characteristics described above (transparent construction of messages and integration of *intra*agent with *inter*agent mechanisms) apply to most other library functionality as well, including but not limited to data management and temporal control mechanisms.

Source Code Appendix

Source code for version 2.0 of the*OAA* software product is included as an appendix hereto, and is incorporated herein by reference. The code includes an agent library, which provides infrastructure for constructing an agent-based system. The library's several families of procedures provide the functionalities discussed above, as well as others that have not been discussed here but that will be sufficiently clear to the interested practitioner. For example, declarations of an agent's solvables, and their registration with a facilitator, are managed using procedures such as *oaa_Declare*, *oaa_Undeclare*, and *oaa_Redeclare*. Updates to data solvables can be accomplished with a family of procedures including *oaa_AddData*, *oaa_RemoveData*, and *oaa_ReplaceData*. Similarly, triggers are maintained using procedures such as *oaa_AddTrigger*, *oaa_RemoveTrigger*, and *oaa_ReplaceTrigger*. The provided source code also includes source code for an OAA Facilitator Agent.

The source code appendix is offered solely as a means of further helping practitioners to construct a preferred embodiment of the invention. By no means is the source code intended to limit the scope of the present invention.

Illustrative Applications

To further illustrate the technology of the preferred embodiment, we will next present and discuss two sample applications of the present inventions.

Unified Messaging

A further preferred embodiment of present invention incorporates a Unified Messaging application extending the Automated Office application presented previously herein with an emphasis on ubiquitous access and dynamic presentation of the information and services supported by the agent community. The agents used in this application are depicted in Figure 12.

A hypothetical example of realistic dialog using a preferred embodiment of the present invention can provide insight into how systems may preferably be built using the present invention. In this scenario, the user, with only a telephone as an interface, is planning a trip to Boston where he will soon give a presentation. Capitalized sentences are phrases spoken by the user into the telephone and processed by a phone agent 452.

Responses, unless otherwise indicated, are spoken by the system using text-to-speech generation agent 454.

1.1 Welcome to SRI International. Please enter your user ID and password.
<User enters touchtone ID and password>

Good to see you again Adam Cheyer. I am listening to you.

Every user interface agent 408, including the telephone agent 452, should know the identify of its user. This information is used in resolving anaphoric references such as "me" and "I", and allows multiple user interfaces operated by the same user to work together.

1.2 WHAT IS TODAY'S SCHEDULE?
Here is today's schedule for Adam Cheyer:
At 10am for 1 hour, meeting with Dave.
At 3pm for 1 hour, presentation about software agents.

End of schedule.

If the user is operating both a graphical user interface and a telephone, as described in conjunction with the Automated Office application, the result of this spoken request is to display a calendar window containing the current schedule. In this case, with no graphical display available, the GEN_NL agent 1202 is tasked to produce a spoken response that can be played over the phone. GEN_NL shares the same dynamic vocabulary and phrasal rules as the natural language parser DCG_NL 426, and contains strategies for producing responses to queries using either simple or list-based multimedia utterances.

1.3 FIND FRIDAY'S WEATHER IN BOSTON.
The weather in Boston for Friday is as follows:
Sunny in the morning. Partly cloudy in the afternoon with a 20 percent chance of thunderstorms late. Highs in the mid 70s.

In addition to data accessible from legacy applications, content may be retrieved by web-reading agents which provide wrappers around useful websites.

1.4 FIND ALL NEW MAIL MESSAGES.
There are 2 messages available.
Message 1, from Mark Tierny, entitled "OAA meeting."
1.5 NEXT MESSAGE
Message 2, from Jennifer Schwefler, entitled "Presentation Summary."
1.6 PLAY IT.
This message is a multipart MIME-encoded message. There are two parts.
Part 1. (Voicemail message, not text-to speech):
Thanks for taking part as a speaker in our conference.
The schedule will be posted soon on our homepage.
1.7 NEXT PART
Part 2. (read using text-to-speech):
The presentation home page is http://www....
1.8 PRINT MESSAGE
Command executed.

Mail messages are no longer just simple text documents, but often consist of multiple subparts containing audio files, pictures, webpages, attachments and so forth. When a user asks to play a complex email message over the telephone, many different agents may be implicated in the translation process, which would be quite different given the request "print it." The challenge is to develop a system which will enable agents to cooperate in an extensible, flexible manner that alleviates explicit coding of agent interactions for every possible input/output combination.

In a preferred embodiment of the present invention, each agent concentrates only on what it can do and on what it knows, and leaves other work to be delegated to the agent community. For instance, a printer agent 1204, defining the solvable print(Object,Parameters), can be defined by the following pseudo-code, which basically says, "If someone can get me a document, in either POSTSCRIPT or text form, I can print it.".

```
print(Object, Parameters) {
   ' If Object is reference to "it", find an appropriate
document
   if (Object = "ref(it)")
      oaa_Solve(resolve_reference(the, document, Params,
Object),[]);
   ' Given a reference to some document, ask for the
document in POSTSCRIPT
   if (Object = "id(Pointer)")
      oaa_Solve(resolve_id_as(id(Pointer), postscript,
[], Object),[]);
   ' If Object is of type text or POSTSCRIPT, we can
print it.
   if ((Object is of type Text) or (Object is of type
Postscript))
      do_print(Object);
}
```

In the above example, since an email message is the salient document, the mail agent 442 will receive a request to produce the message as POSTSCRIPT. Whereas the mail agent 442 may know how to save a text message as POSTSCRIPT, it will not know what to do with a webpage or voicemail message. For these parts of the message, it will simply send oaa_Solve requests to see if another agent knows how to accomplish the task.

Until now, the user has been using only a telephone as user interface. Now, he moves to his desktop, starts a web browser 436, and accesses the URL referenced by the mail message.

```
1.9  RECORD MESSAGE
     Recording voice message.  Start speaking now.
1.10 THIS IS THE UPDATED WEB PAGE CONTAINING THE
PRESENTATION SCHEDULE.
     Message one recorded.
1.11 IF THIS WEB PAGE CHANGES, GET IT TO ME WITH NOTE
ONE.
     Trigger added as requested.
```

In this example, a local agent 436 which interfaces with the web browser can return the current page as a solution to the request "oaa_Solve(resolve_reference(this, web_page, [], Ref),[])", sent by the NL agent 426. A trigger is installed on a web agent 436 to monitor changes to the page, and when the page is updated, the notify agent 446 can find the user and transmit the webpage and voicemail message using the most appropriate media transfer mechanism.

This example based on the Unified Messaging application is intended to show how concepts in accordance with the present invention can be used to produce a simple yet extensible solution to a multi-agent problem that would be difficult to implement using a more rigid framework. The application supports adaptable presentation for queries across dynamically changing, complex information; shared context and reference resolution among applications; and flexible translation of multimedia data. In the next section, we will present an application which highlights the use of parallel competition and cooperation among agents during multi-modal fusion.

Multimodal Map

A further preferred embodiment of present invention incorporates the Multimodal Map application. This application demonstrates natural ways of communicating with a community of agents, providing an interactive interface on which the user may draw, write or speak. In a travel-planning domain illustrated by Figure 13, available information includes hotel, restaurant, and tourist-site data retrieved by distributed software agents from commercial Internet sites. Some preferred types of user interactions and multimodal issues handled by the application are illustrated by a brief scenario featuring working examples taken from the current system.

Sara is planning a business trip to San Francisco, but would like to schedule some activities for the weekend while she is there. She turns on her laptop PC, executes a map application, and selects San Francisco.

```
2.1    [Speaking] Where is downtown?
       Map scrolls to appropriate area.
2.2    [Speaking and drawing region] Show me all hotels near here.
       Icons representing hotels appear.
2.3    [Writes on a hotel] Info?
       A textual description (price, attributes, etc.) appears.
2.4    [Speaking] I only want hotels with a pool.
       Some hotels disappear.
2.5    [Draws a crossout on a hotel that is too close to a highway]
       Hotel disappears
2.6    [Speaking and circling] Show me a photo of this hotel.
       Photo appears.
2.7    [Points to another hotel]
       Photo appears.
2.8    [Speaking] Price of the other hotel?
       Price appears for previous hotel.
2.9    [Speaking and drawing an arrow] Scroll down.
       Display adjusted.
2.10   [Speaking and drawing an arrow toward a hotel]
       What is the distance from this hotel to Fisherman's Wharf?
       Distance displayed.
2.11   [Pointing to another place and speaking] And the distance to here?
       Distance displayed.
```

Sara decides she could use some human advice. She picks up the phone, calls Bob, her travel agent, and writes Start collaboration to synchronize his display with hers. At this point, both are presented with identical maps, and the input and actions of one will be remotely seen by the other.

```
3.1    [Sara speaks and circles two hotels]
       Bob, I'm trying to choose between these two hotels. Any opinions?
3.2    [Bob draws an arrow, speaks, and points]
       Well, this area is really nice to visit. You can walk there from
```

```
        this hotel.
        Map scrolls to indicated area.  Hotel selected.
  3.3   [Sara speaks] Do you think I should visit Alcatraz?
  3.4   [Bob speaks] Map, show video of Alcatraz.
        Video appears.
  3.5   [Bob speaks] Yes, Alcatraz is a lot of fun.
```

A further preferred embodiment of present invention generates the most appropriate interpretation for the incoming streams of multimodal input. Besides providing a user interface *to* a dynamic set of distributed agents, the application is preferably built *using* an agent framework. The present invention also contemplates aiding the coordinate competition and cooperation among information sources, which in turn works in parallel to resolve the ambiguities arising at every level of the interpretation process: *low-level processing of the data stream*, *anaphora resolution*, *cross-modality influences* and *addressee*.

Low-level processing of the data stream: Pen input may be preferably interpreted as a gesture (e.g., 2.5: cross-out) by one algorithm, or as handwriting by a separate recognition process (e.g., 2.3: "info?"). Multiple hypotheses may preferably be returned by a modality recognition component.

Anaphora resolution: When resolving anaphoric references, separate information sources may contribute to resolving the reference: context by object type, deictic, visual context, database queries, discourse analysis. An example of information provided through context by object type is found in interpreting an utterance such as "show photo of the hotel", where the natural language component can return a list of the last hotels talked about. Deictic information in combination with a spoken utterance like "show photo of this hotel" may preferably include pointing, circling, or arrow gestures which might indicate the desired object (e.g., 2.7). Deictic references may preferably occur before, during, or after an accompanying verbal command. Information provided in a visual context, given for the request "display photo of the hotel" may preferably include the user interface agent might determine that only one hotel is currently visible on the map, and therefore this might be the desired reference object. Database queries preferably involving information from a database agent combined with results from other resolution strategies. Examples are "show me a photo of the hotel in Menlo Park" and

2.2. Discourse analysis preferably provides a source of information for phrases such as "No, the other one" (or 2.8).

The above list of preferred anaphora resolution mechanisms is not exhaustive. Examples of other preferred resolution methods include but are not limited to spatial reasoning ("the hotel between Fisherman's Wharf and Lombard Street") and user preferences ("near my favorite restaurant").

Cross-modality influences: When multiple modalities are used together, one modality may preferably reinforce or remove or diminish ambiguity from the interpretation of another. For instance, the interpretation of an arrow gesture may vary when accompanied by different verbal commands (e.g., "scroll left" vs. "show info about this hotel"). In the latter example, the system must take into account how accurately and unambiguously an arrow selects a single hotel.

Addressee: With the addition of collaboration technology, humans and automated agents all share the same workspace. A pen doodle or a spoken utterance may be meant for either another human, the system (3.1), or both (3.2).

The implementation of the Multimodal Map application illustrates and exploits several preferred features of the present invention: reference resolution and task delegation by parallel parameters of oaa_Solve, basic multi-user collaboration handled through built-in data management services, additional functionality readily achieved by adding new agents to the community, domain-specific code cleanly separated from other agents.

A further preferred embodiment of present invention provides reference resolution and task delegation handled in a distributed fashion by the parallel parameters of oaa_Solve, with meta-agents encoding rules to help the facilitator make context- or user-specific decisions about priorities among knowledge sources.

A further preferred embodiment of present invention provides basic multi-user collaboration handled through at least one built-in data management service. The map user interface preferably publishes data solvables for elements such as icons, screen position, and viewers, and preferably defines these elements to have the attribute "shareable". For every update to this public data, the changes are preferably automatically replicated to all members of the collaborative session, with associated callbacks producing the visible effect of the data change (e.g., adding or removing an icon).

Functionality for recording and playback of a session is preferably implemented by adding agents as members of the collaborative community. These agents either record the data changes to disk, or read a log file and replicate the changes in the shared environment.

The domain-specific code for interpreting travel planning dialog is preferably separated from the speech, natural language, pen recognition, database and map user interface agents. These components were preferably reused without modification to add multimodal map capabilities to other applications for activities such as crisis management, multi-robot control, and the MVIEWS tools for the video analyst.

Improved Scalability and Fault Tolerance

Implementations of a preferred embodiment of present invention which rely upon simple, single facilitator architectures may face certain limitations with respect to scalability, because the single facilitator may become a communications bottleneck and may also represent a single, critical point for system failure.

Multiple facilitator systems as disclosed in the preferred embodiments to this point can be used to construct peer-to-peer agent networks as illustrated in Figure 14. While such embodiments are scalable, they do possess the potential for communication bottlenecks as discussed in the previous paragraph and they further possess the potential for reliability problems as central, critical points of vulnerability to systems failure.

A further embodiment of present invention supports a facilitator implemented as an agent like any other, whereby multiple facilitator network topologies can be readily constructed. One example configuration (but not the only possibility) is a hierarchical topology as depicted in Figure 15, where a top level Facilitator manages collections of both client agents 1508 and other Facilitators, 1504 and 1506. Facilitator agents could be installed for individual users, for a group of users, or as appropriate for the task.

Note further, that network work topologies of facilitators can be seen as graphs where each node corresponds to an instance of a facilitator and each edge connecting two or more nodes corresponds to a transmission path across one or more physical transport mechanisms. Some nodes may represent facilitators and some nodes may represent clients. Each node can be further annotated with attributes corresponding to include triggers, data, capabilities but not limited to these attributes.

A further embodiment of present invention provides enhanced scalability and robustness by separating the planning and execution components of the facilitator. In contrast with the centralized facilitation schemes described above, the facilitator system 1600 of Figure 16 separates the registry/planning component from the execution component. As a result, no single facilitator agent must carry all communications nor does the failure of a single facilitator agent shut down the entire system.

Turning directly to Figure 16, the facilitator system 1600 includes a registry/planner 1602 and a plurality of client agents 1612-1616. The registry/planner 1604 is typically replicated in one or more locations accessible by the client agents. Thus if the registry/planner 1604 becomes unavailable, the client agents can access the replicated registry/planner(s).

This system operates, for example, as follows. An agent transmits a goal 1610 to the registry planner 1602. The registry/planner 1604 translates the goal into an unambiguous execution plan detailing how to accomplish any sub-goals developed from the compound goal, as well as specifying the agents selected for performing the sub-goals. This execution plan is provided to the requesting agent which in turn initiates peer-to-peer interactions 1618 in order to implement the detailed execution plan, routing and combining information as specified within the execution plan. Communication is distributed thus decreasing sensitivity of the system to bandwidth limitations of a single facilitator agent. Execution state is likewise distributed thus enabling system operation even when a facilitator agent fails.

Further embodiments of present invention incorporate into the facilitator functionality such as load-balancing, resource management, and dynamic configuration of agent locations and numbers, using (for example) any of the topologies discussed. Other embodiments incorporate into a facilitator the ability to aid agents in establishing peer-to-peer communications. That is, for tasks requiring a sequence of exchanges between two agents, the facilitator assist the agents in finding one another and establishing communication, stepping out of the way while the agents communicate peer-to-peer over a direct, perhaps dedicated channel.

Further preferred embodiments of the present invention incorporate mechanisms for basic transaction management, such as periodically saving the state of agents (both facilitator and client) and rolling back to the latest saved state in the event of the failure of an agent.

IN THE CLAIMS:

1. 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, 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 goal expression, said act of interpreting further comprising:

generating one or more sub-goals using the inter-agent language; 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. 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 last step of claim 1 in order to perform the new request for service.

3. 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 the 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 as 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.

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.

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.

24. A computer implemented method as recited in claim 15 wherein the conditional functionality of the trigger is installed on a 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. 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), 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,

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.

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. A 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.

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.

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 a parallel disjunction operator that indicates that disjunct goals are to be performed by different agents.

48. 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, the ICL enabling agents to perform queries of other agents, exchange information with other agents, set triggers within other agents, 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.

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

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 within goal expressions.

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.

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. 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 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.

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. 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;

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.

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.

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. A computer architecture as recited in claim 71 wherein the planning component of the facilitating engine are distributed across at least two computer processes.

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

86. 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, the data wave carrier comprising a signal representation of an inter-agent language description of an active client agent's functional capabilities.

87. A data wave carrier as recited in claim 85, the data wave carrier further comprising a signal representation of a request for service in the inter-agent language from a first agent to a second agent.

88. A data wave carrier as recited in claim 85, 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.

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

ABSTRACT

A highly flexible, software-based architecture is disclosed for constructing distributed systems. The architecture supports cooperative task completion by flexible, dynamic configurations of autonomous electronic agents. Communication and cooperation between agents are brokered by one or more facilitators, which are responsible for matching requests, from users and agents, with descriptions of the capabilities of other agents. It is not generally required that a user or agent know the identities, locations, or number of other agents involved in satisfying a request, and relatively minimal effort is involved in incorporating new agents and "wrapping" legacy applications. Extreme flexibility is achieved through an architecture organized around the declaration of capabilities by service-providing agents, the construction of arbitrarily complex goals by users and service-requesting agents, and the role of facilitators in delegating and coordinating the satisfaction of these goals, subject to advice and constraints that may accompany them. Additional mechanisms and features include facilities for creating and maintaining shared repositories of data; the use of triggers to instantiate commitments within and between agents; agent-based provision of multi-modal user interfaces, including natural language; and built-in support for including the user as a privileged member of the agent community. Specialized embodiments providing enhanced scalability are also described.

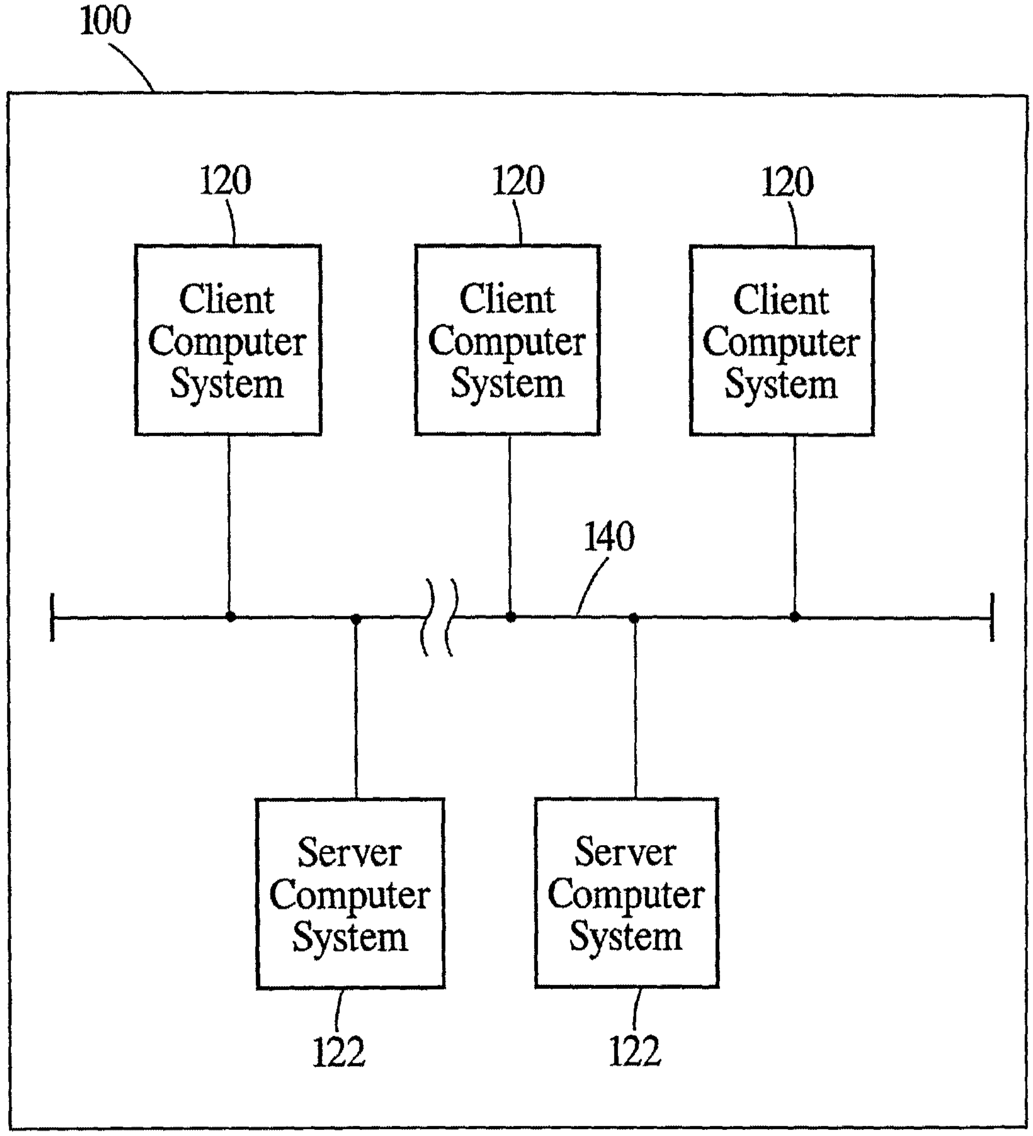


Fig. 1
(Prior Art)

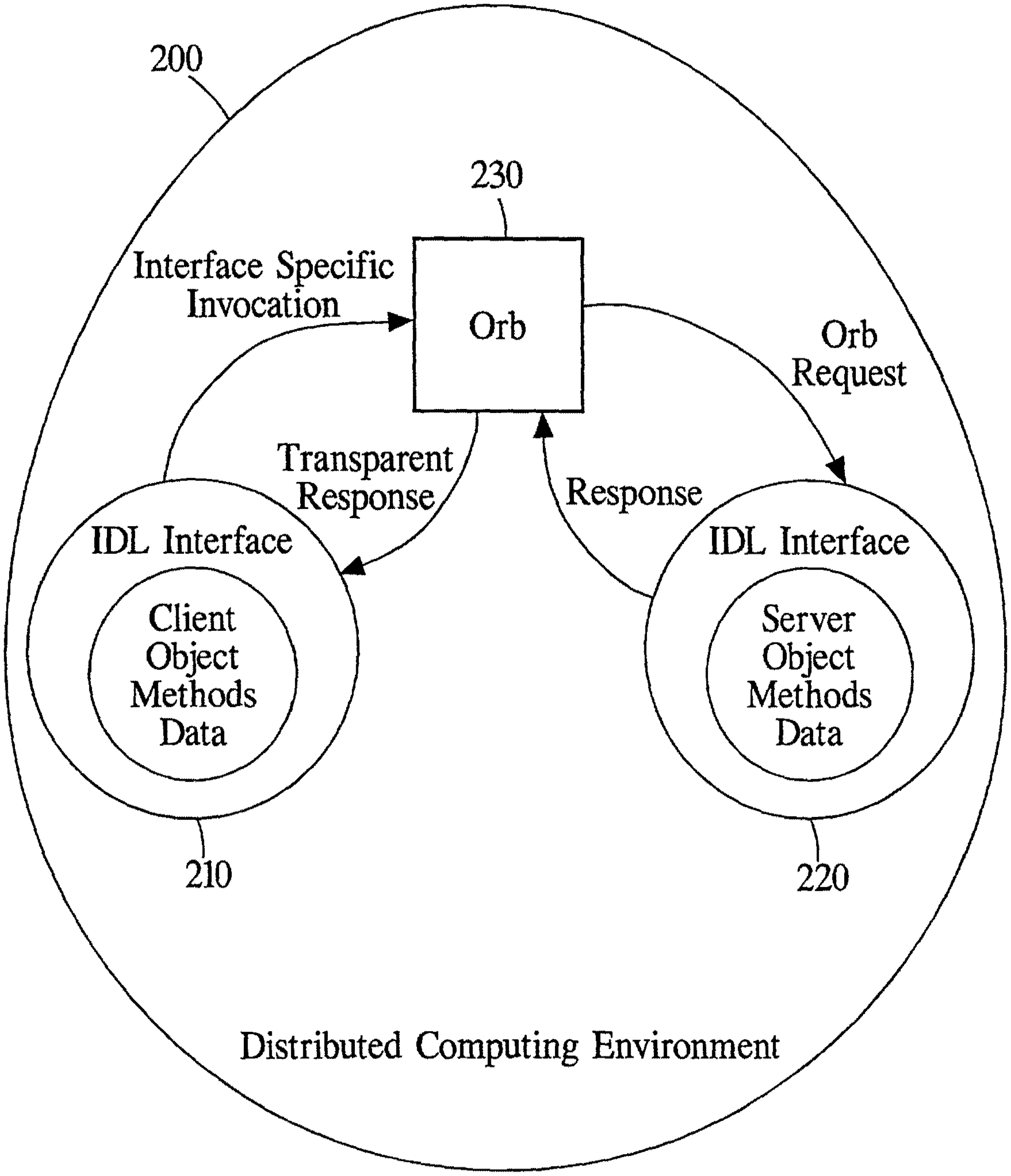


Fig. 2
(Prior Art)

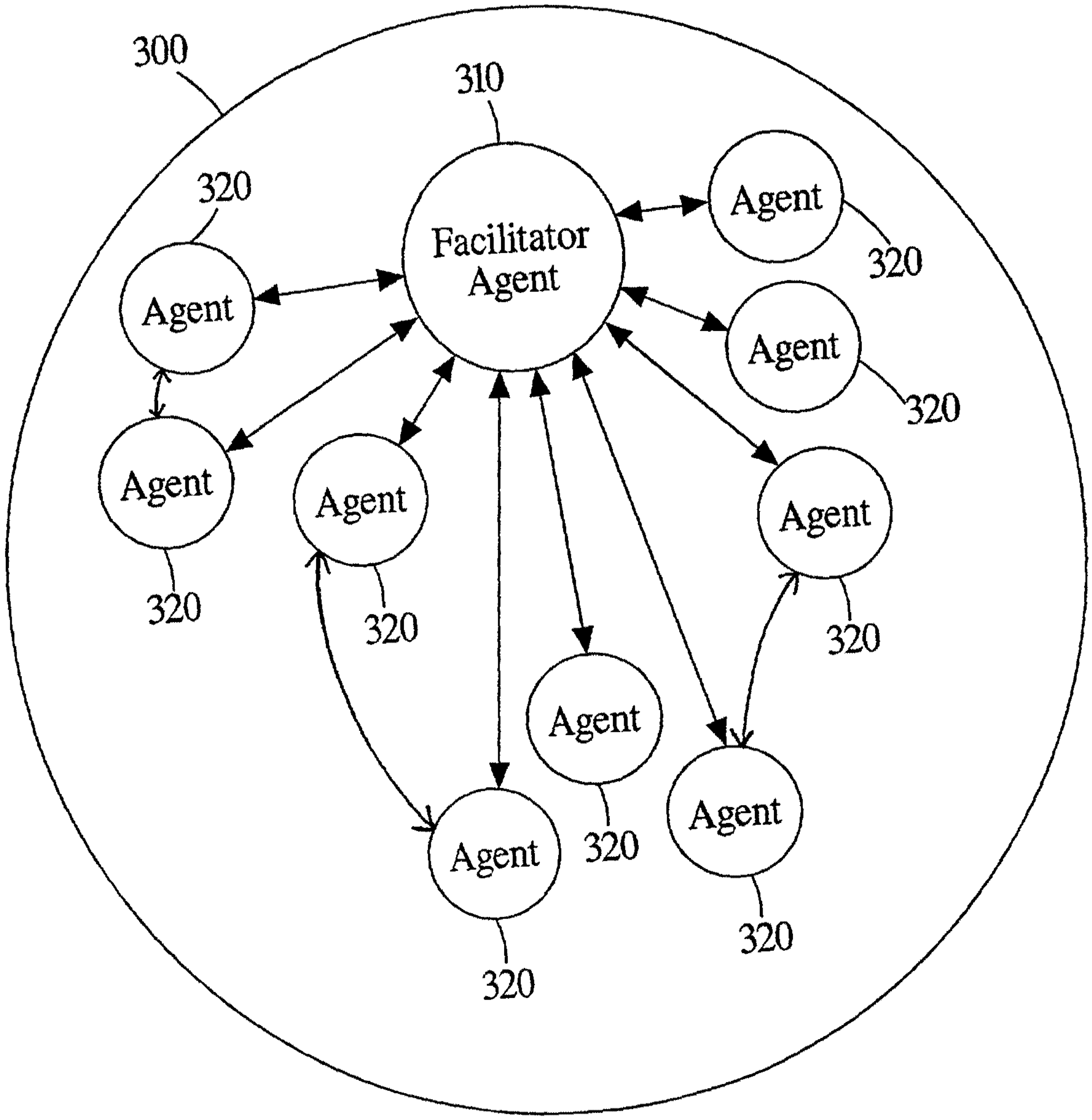

300
310
Facilitator Agent
Agent
Agent
Agent
Agent
Agent
Agent
Agent
Agent
Agent
320
320
320
320
320
320
320
320
320

Fig. 3

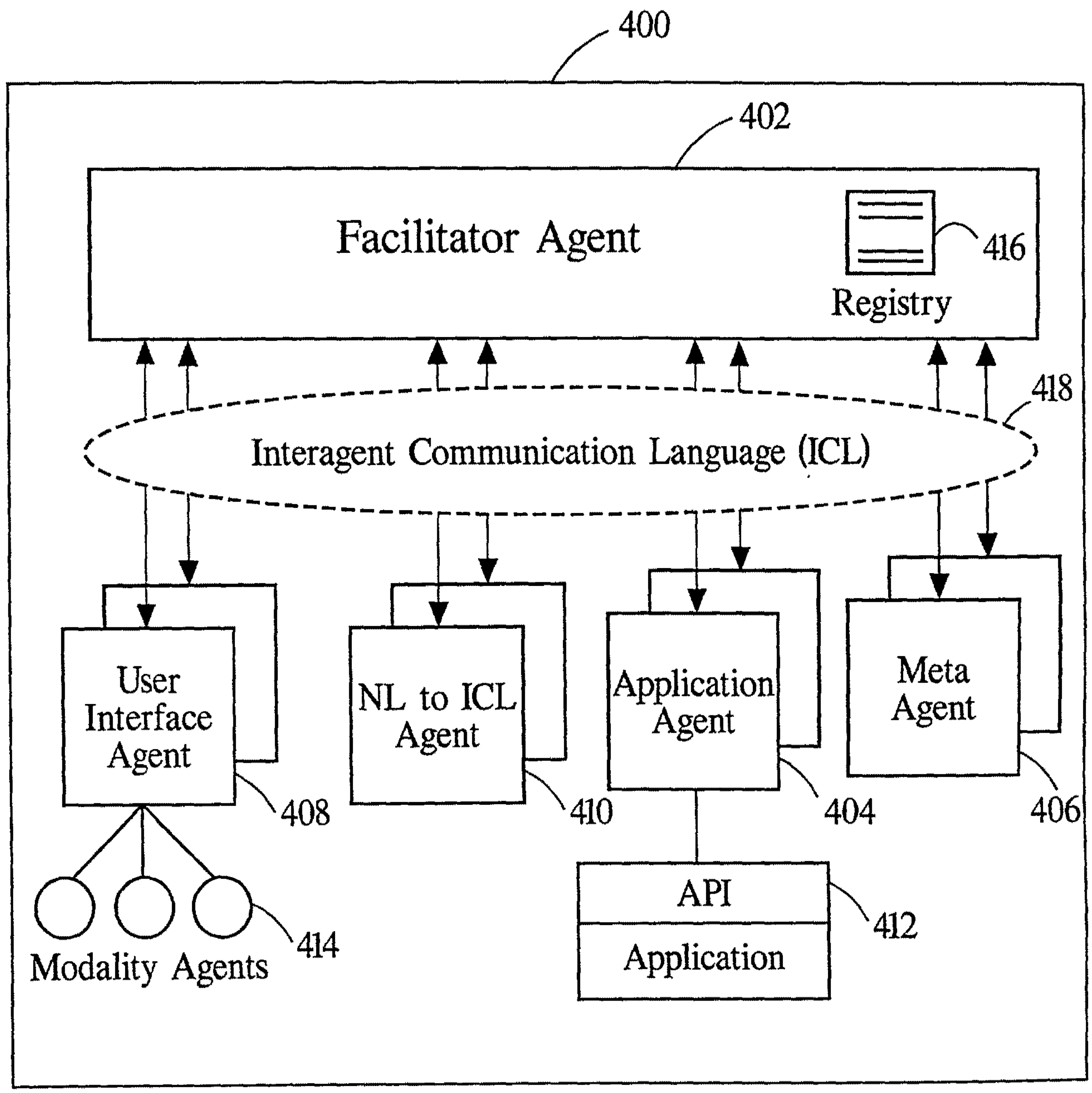

400
402
Facilitator Agent
416
Registry
418
Interagent Communication Language (ICL)
User Interface Agent
408
NL to ICL Agent
410
Application Agent
404
Meta Agent
406
Modality Agents
414
API
Application
412

Fig. 4

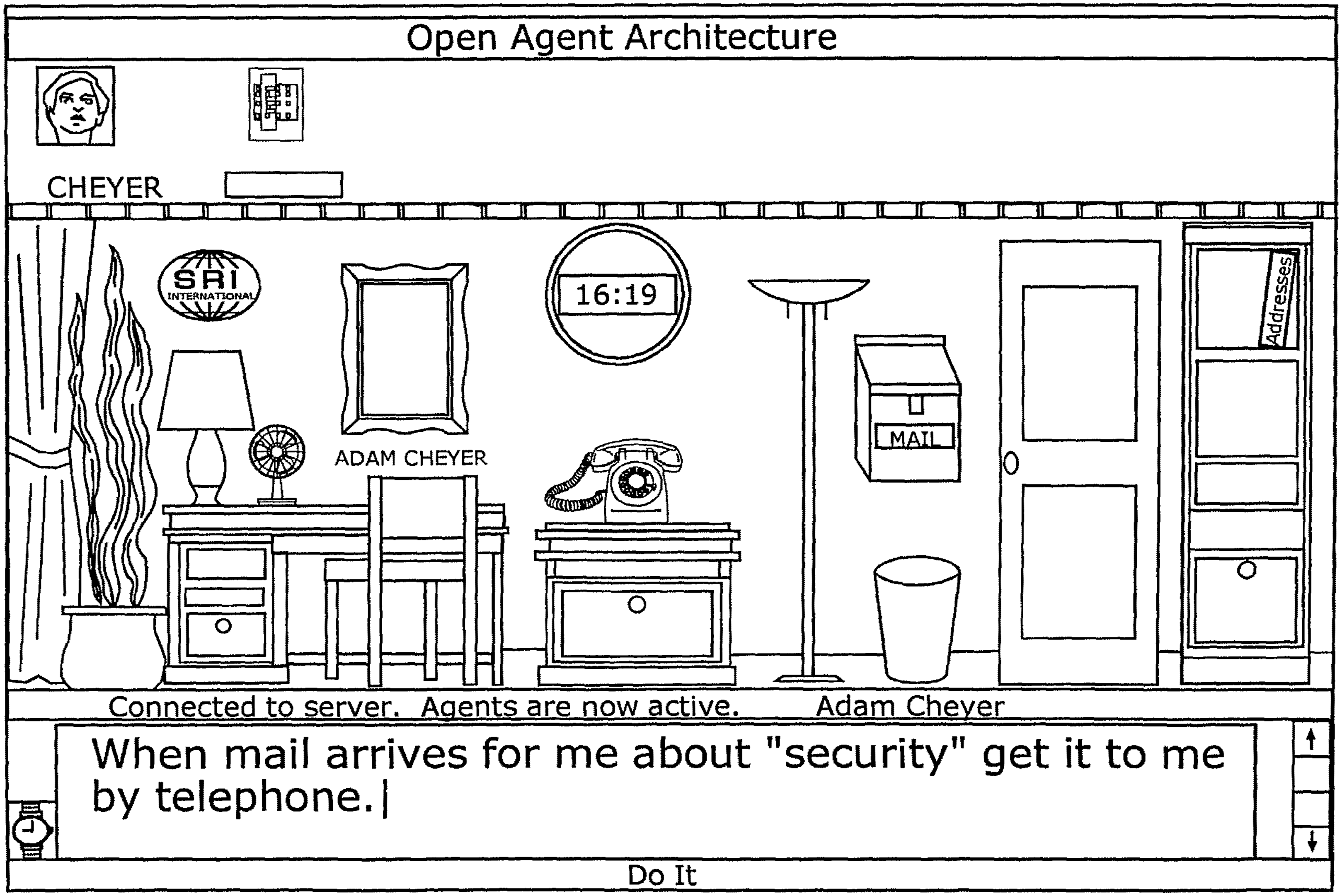
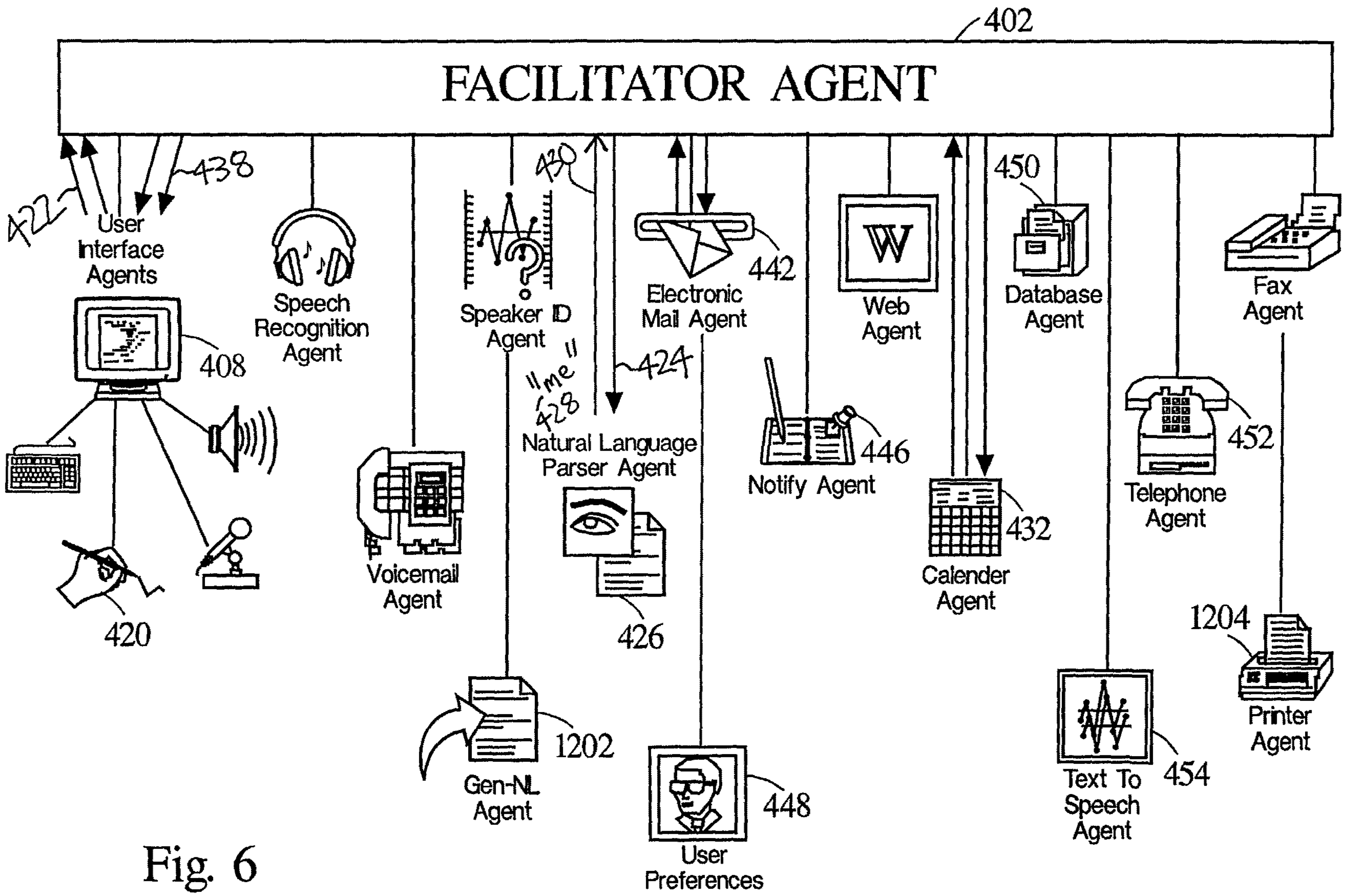

Open Agent Architecture
CHEYER
SRI
INTERNATIONAL
16:19
ADAM CHEYER
MAIL
Addresses
Connected to server. Agents are now active.
Adam Cheyer
When mail arrives for me about "security" get it to me by telephone.
Do It

Fig. 5



402
FACILITATOR AGENT
422
438
User Interface Agents
408
420
Speech Recognition Agent
Voicemail Agent
Speaker ID Agent
1202
Gen-NL Agent
430
"me"
428
424
Natural Language Parser Agent
426
Electronic Mail Agent
442
User Preferences
448
Notify Agent
446
Web Agent
Calender Agent
432
450
Database Agent
Text To Speech Agent
454
Telephone Agent
452
Fax Agent
1204
Printer Agent

Fig. 6

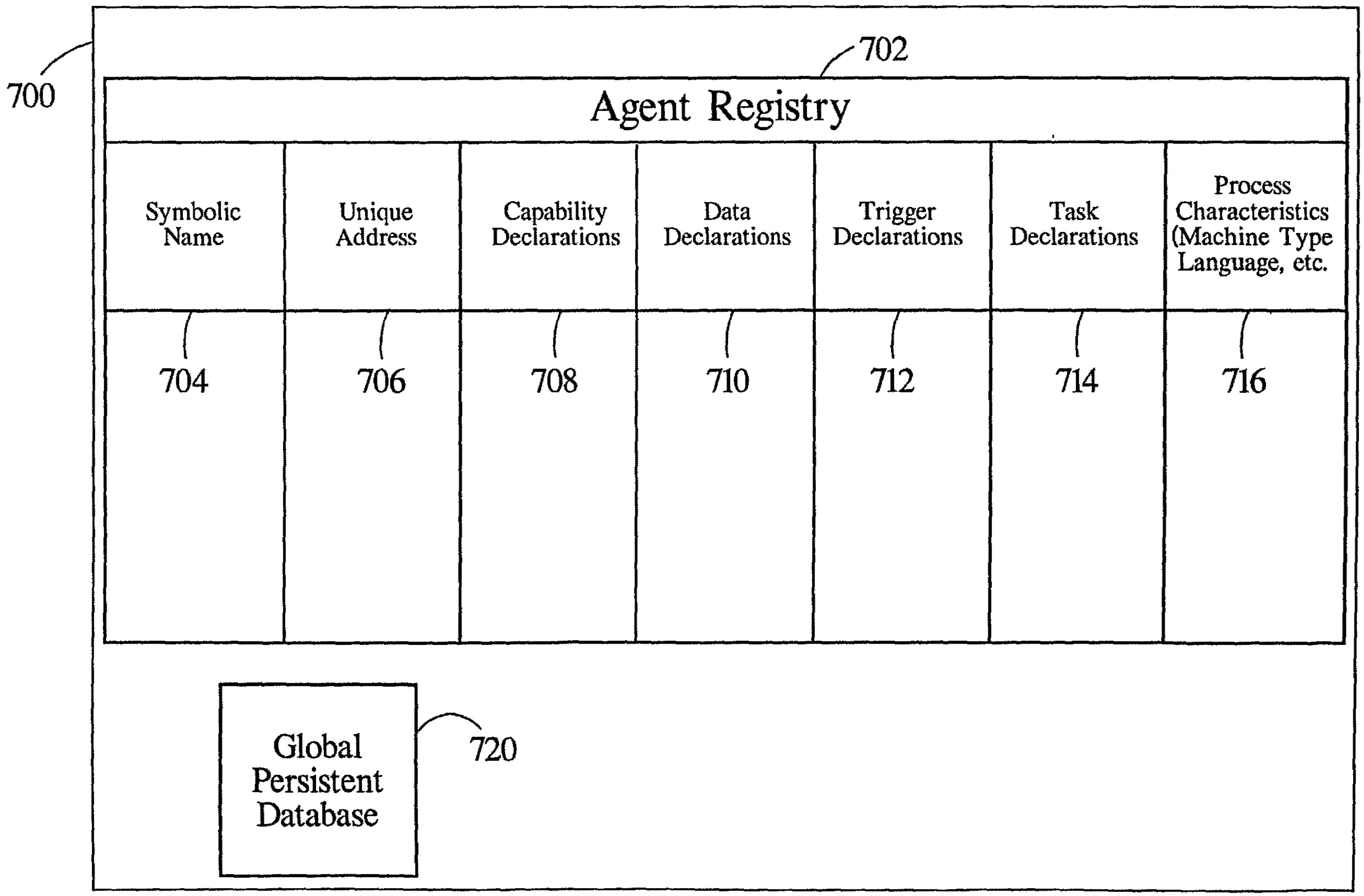

700
702
Agent Registry
Symbolic Name
Unique Address
Capability Declarations
Data Declarations
Trigger Declarations
Task Declarations
Process Characteristics (Machine Type Language, etc.
704
706
708
710
712
714
716
Global Persistent Database
720

Fig. 7

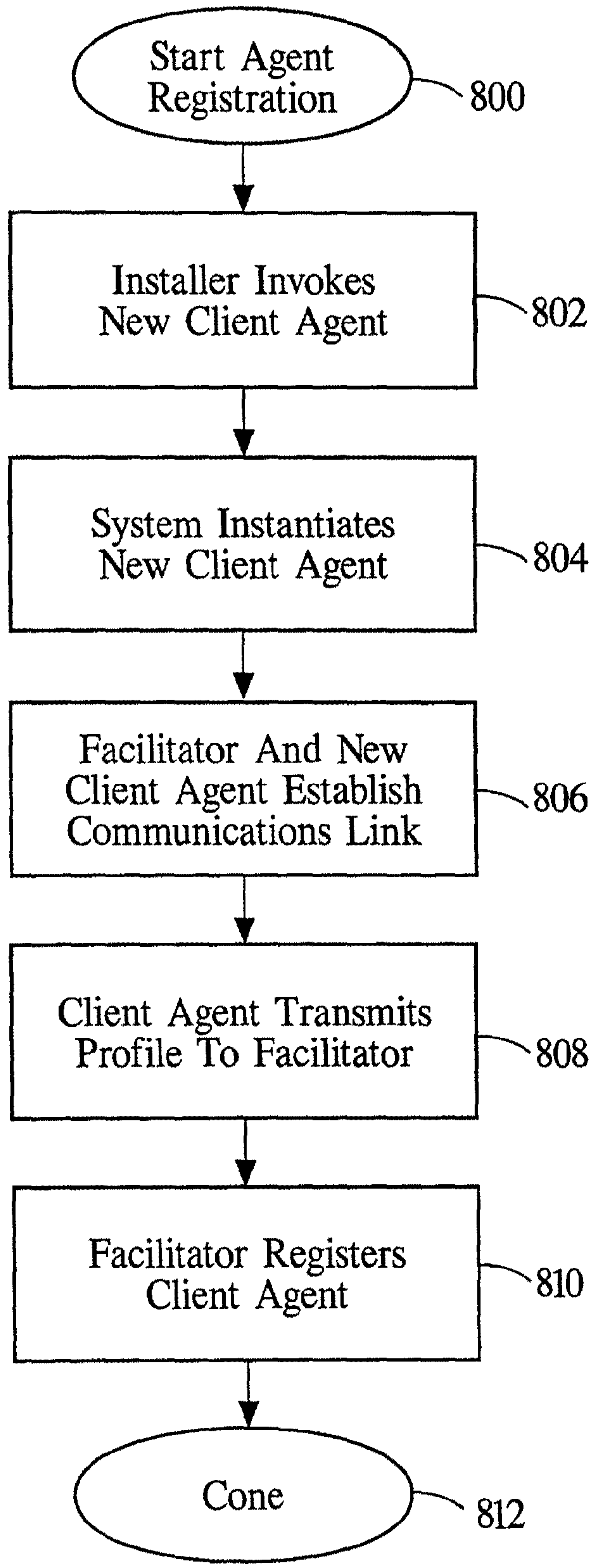

Start Agent Registration
800
Installer Invokes New Client Agent
802
System Instantiates New Client Agent
804
Facilitator And New Client Agent Establish Communications Link
806
Client Agent Transmits Profile To Facilitator
808
Facilitator Registers Client Agent
810
Cone
812

Fig. 8

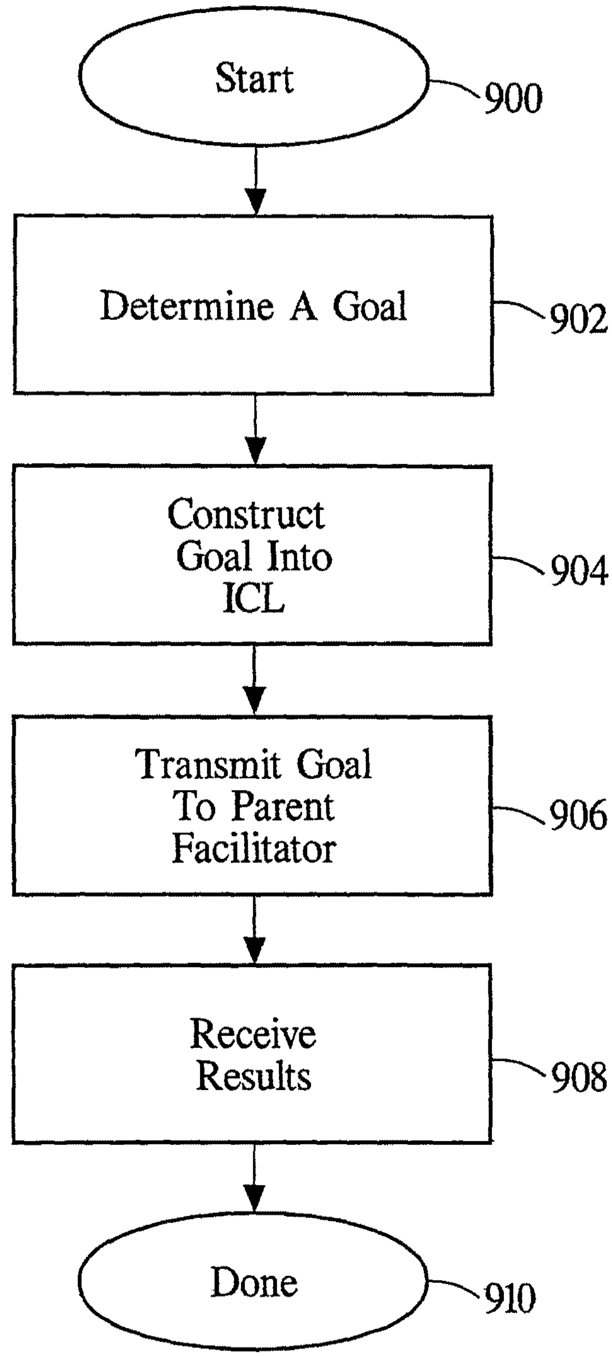

Start
900
Determine A Goal
902
Construct
Goal Into
ICL
904
Transmit Goal
To Parent
Facilitator
906
Receive
Results
908
Done
910

Fig. 9

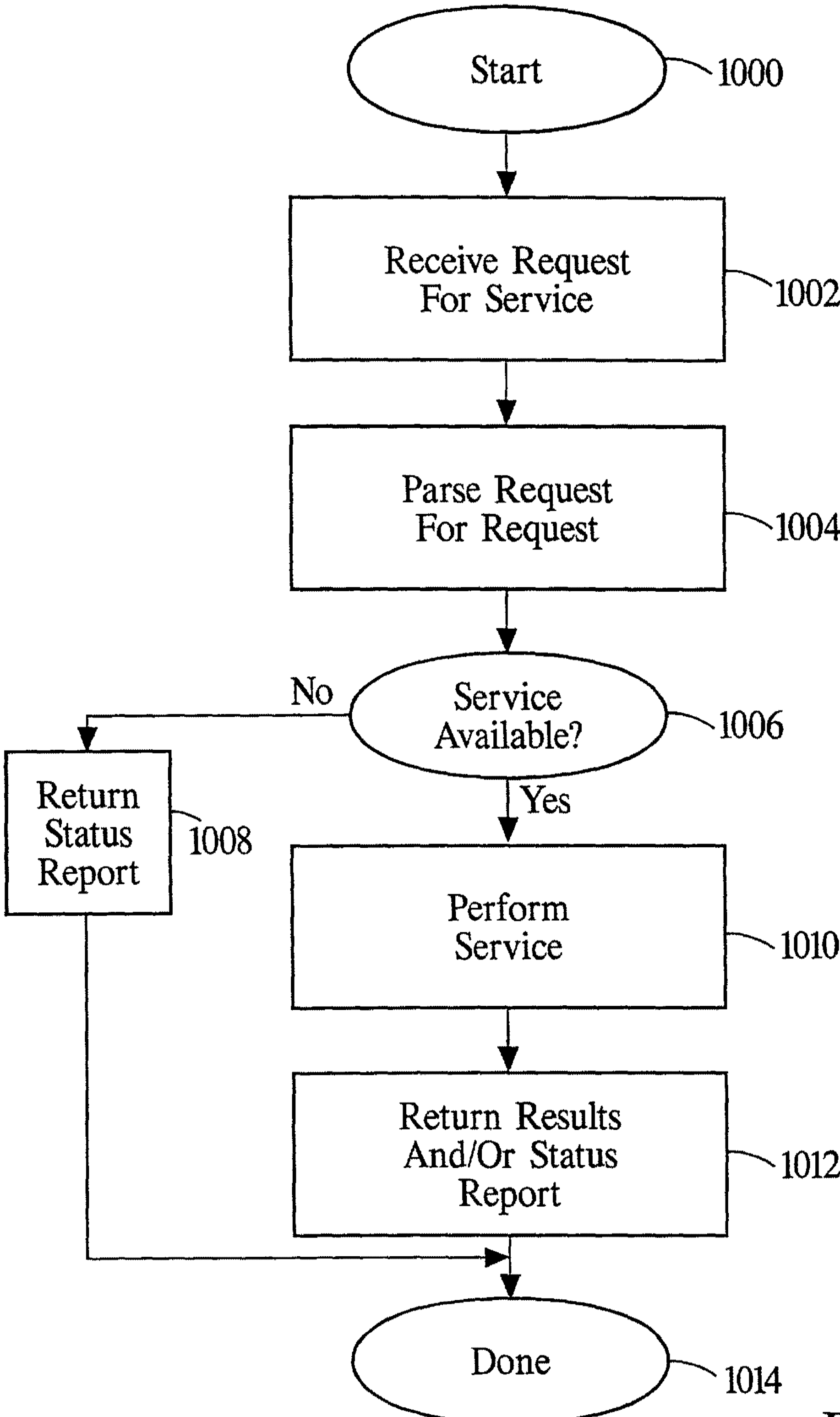

Start
1000
Receive Request For Service
1002
Parse Request For Request
1004
Service Available?
1006
No
Yes
Return Status Report
1008
Perform Service
1010
Return Results And/Or Status Report
1012
Done
1014

Fig. 10

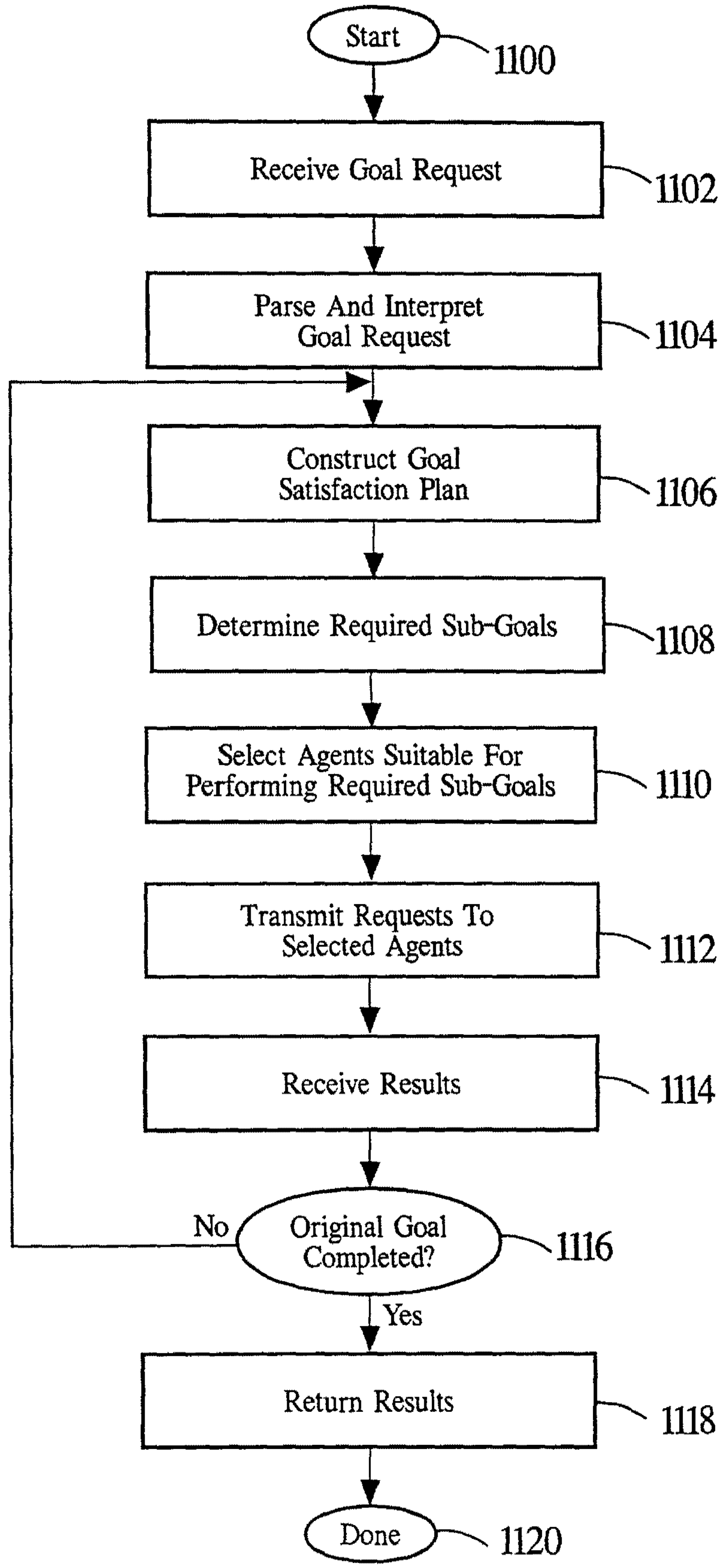

Start
1100
Receive Goal Request
1102
Parse And Interpret Goal Request
1104
Construct Goal Satisfaction Plan
1106
Determine Required Sub-Goals
1108
Select Agents Suitable For Performing Required Sub-Goals
1110
Transmit Requests To Selected Agents
1112
Receive Results
1114
No
Original Goal Completed?
1116
Yes
Return Results
1118
Done
1120

Fig. 11

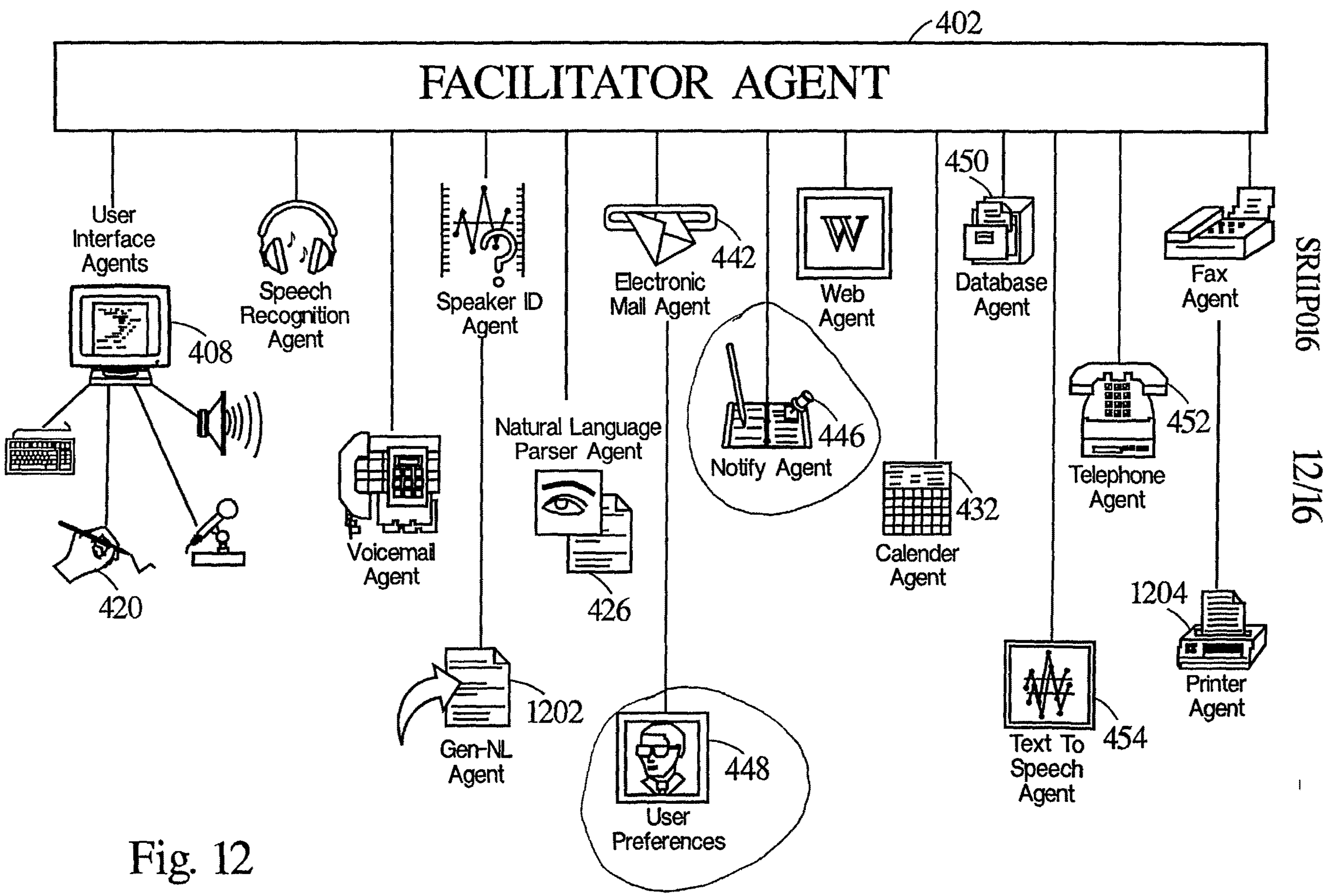
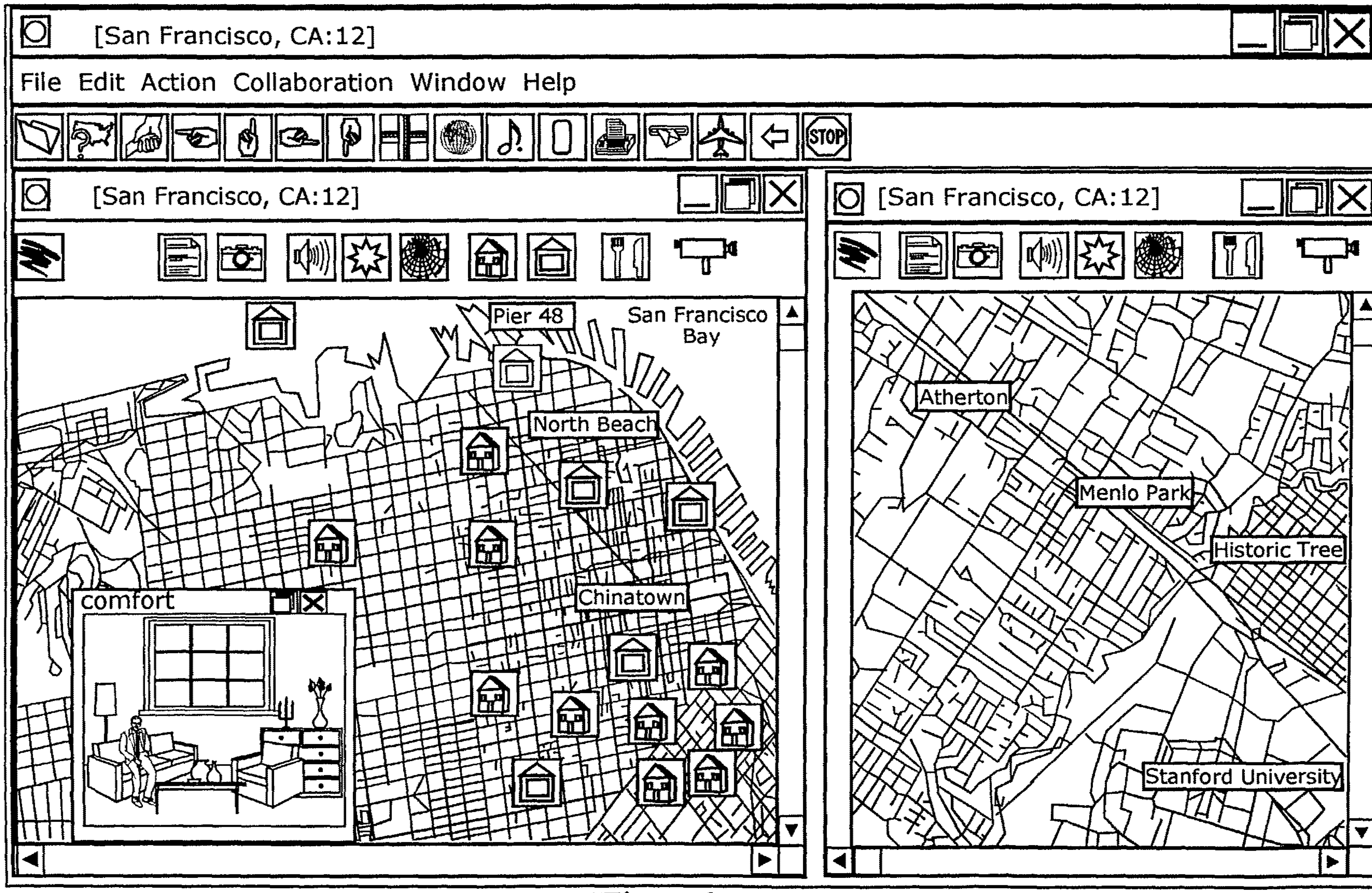

402
FACILITATOR AGENT
User Interface Agents
408
420
Speech Recognition Agent
Voicemail Agent
Speaker ID Agent
Gen-NL Agent
1202
Natural Language Parser Agent
426
Electronic Mail Agent
442
User Preferences
448
Notify Agent
446
Web Agent
W
450
Database Agent
Calender Agent
432
Text To Speech Agent
454
Telephone Agent
452
Fax Agent
1204
Printer Agent
SRI1P016
12/16

Fig. 12



SRI1P016 13/16

Fig. 13

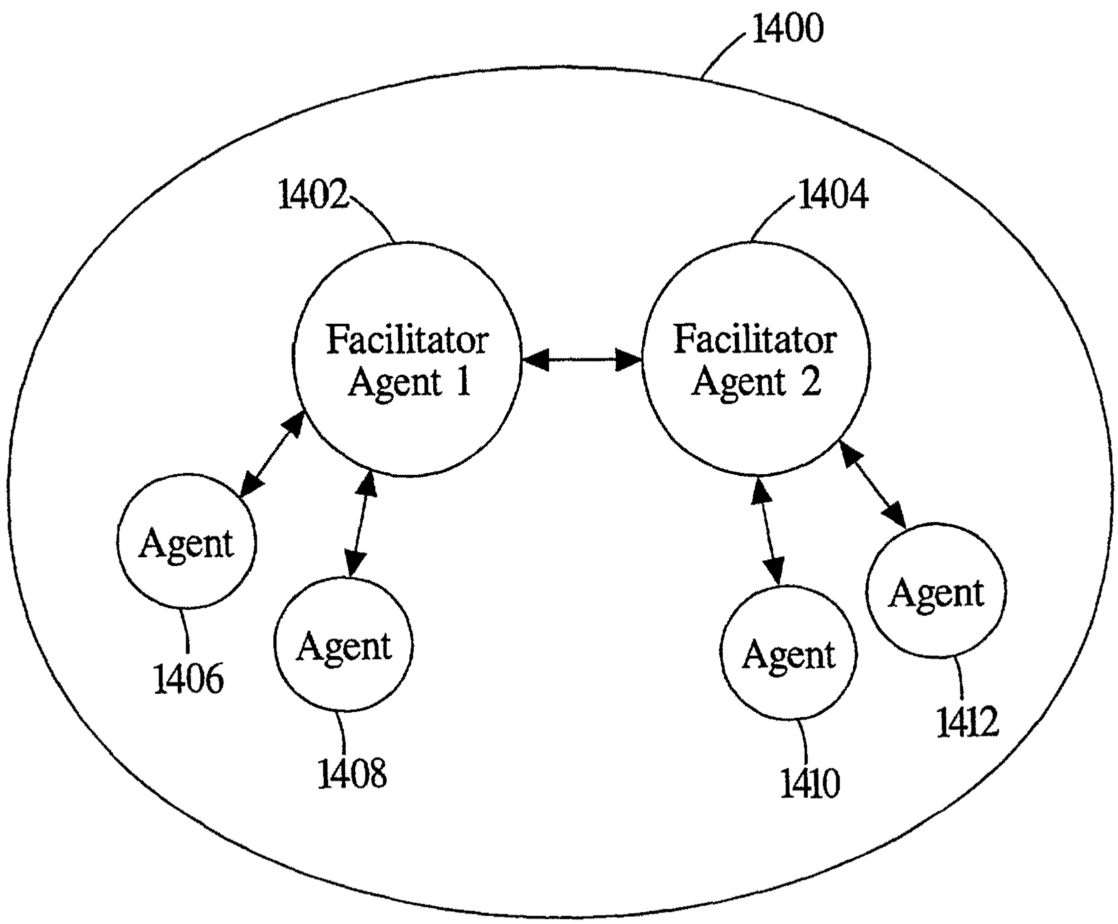

1400
1402
1404
Facilitator
Agent 1
Facilitator
Agent 2
Agent
Agent
Agent
Agent
1406
1408
1410
1412

Fig. 14

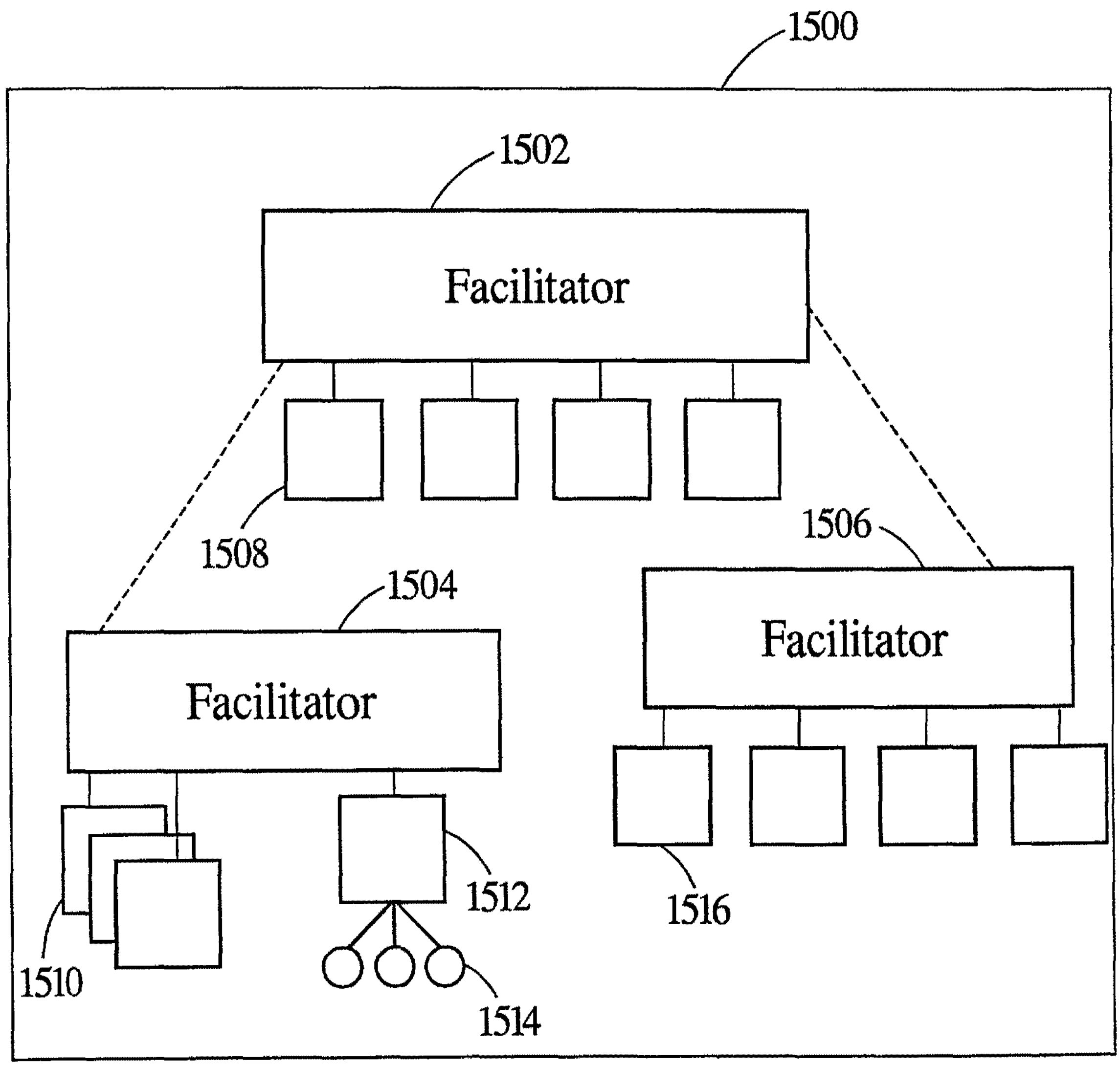

1500
1502
Facilitator
1508
1504
1506
Facilitator
Facilitator
1510
1512
1514
1516

Fig. 15

SRI1P016 16/16

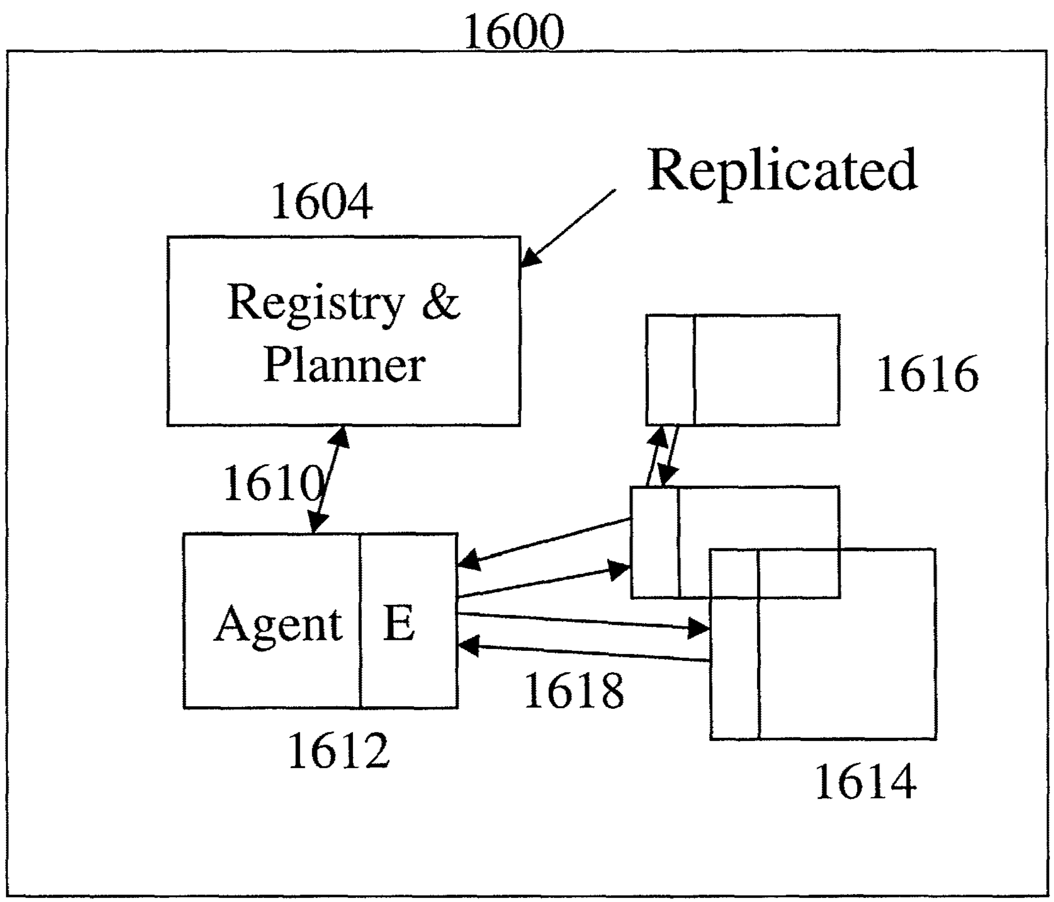


Figure 16

DECLARATION AND POWER OF ATTORNEY
FOR ORIGINAL U.S. PATENT APPLICATION

Attorney's Docket No. SRI1P016

As a below-named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name.

I believe that I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled: SOFTWARE-BASED ARCHITECTURE FOR COMMUNICATION AND COOPERATION AMONG DISTRIBUTED ELECTRONIC AGENTS, the specification of which is attached hereto.

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the examination of this application in accordance with Title 37, CFR § 1.56.

AJC DLM Stephens & Coleman, LLP,

And I hereby appoint the law firm of Hickman ~~& Martine~~ including **Paul L. Hickman (Reg. No. 28, 516); L. Keith Stephens (Reg. No. 32,632); Brian R. Coleman (Reg. No. 39,145); Dawn L. Palmer (Reg. No. 41,238); Jerray Wei (Reg. No. 43,247); and Ian L. Cartier (Reg. No. 38,406)** as my principal attorneys to prosecute this application and to transact all business in the Patent and Trademark Office connected therewith:

Send Correspondence To: **Brian R. Coleman**
HICKMAN STEPHENS & COLEMAN, LLP
P.O. BOX 52037
Palo Alto, California 94303-0746

Direct Telephone Calls To: **Brian R. Coleman at telephone number (650) 470-7430**

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

Typewritten Full Name of Sole or First Inventor:	Adam J. Cheyer	Citizenship:	USA
Inventor's signature:	Adam J. Cheyer	**Date of Signature:**	1/5/99
Residence: (City)	Palo Alto	(State/Country)	CA
Post Office Address:	757 Cereza Drive Palo Alto CA 94306		

Typewritten Full Name of Second Inventor:	David L. Martin	Citizenship:	USA
Inventor's signature:	David L. Martin	**Date of Signature:**	1/5/99
Residence: (City)	Santa Clara	(State/Country)	CA
Post Office Address:	167 CRONIN DR. Santa Clara, CA 95051		

jc618 U.S. PTO
09/225198
01/05/99

Class	Subclass

ISSUE CLASSIFICATION

PATENT NUMBER

U.S. UTILITY PATENT APPLICATION

O.I.P.E.	PATENT DATE
TM SCANNED AG.G Q.A. KB	

SECTOR	CLASS	SUBCLASS	ART UNIT	EXAMINER
	709	317	~~2151~~ 2126	LEOCK

FILED WITH: ☐ DISK (CRF) ☐ FICHE
(Attached in pocket on right inside flap)

PREPARED AND APPROVED FOR ISSUE

ISSUING CLASSIFICATION

ORIGINAL		CROSS REFERENCE(S)						
CLASS	SUBCLASS	CLASS	SUBCLASS (ONE SUBCLASS PER BLOCK)					
INTERNATIONAL CLASSIFICATION								
/								
/								
/								
/								
/				☐ Continued on Issue Slip Inside File Jacket				

☐ TERMINAL DISCLAIMER	DRAWINGS			CLAIMS ALLOWED	
	Sheets Drwg.	Figs. Drwg.	Print Fig.	Total Claims	Print Claim for O.G.
☐ a) The term of this patent subsequent to ________ (date) has been disclaimed.	(Assistant Examiner) (Date)			NOTICE OF ALLOWANCE MAILED	
☐ b) The term of this patent shall not extend beyond the expiration date of U.S Patent. No. ________	(Primary Examiner) (Date)			ISSUE FEE: Amount Due	Date Paid
☐ c) The terminal ____ months of this patent have been disclaimed.	(Legal Instruments Examiner) (Date)			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. 6/98)

Best Available Copy

(LABEL AREA)

[redacted]

Best Available Copy

SEARCHED

Class	Sub.	Date	Exmr.
709	317 202	7/10/02	[illegible]
709	317 202	2/20/03	[illegible]
UPDATED		11/20/03	[illegible]

INTERFERENCE SEARCHED

Class	Sub.	Date	Exmr.

SEARCH NOTES
(INCLUDING SEARCH STRATEGY)

	Date	Exmr.
EAST WEST ACM IEEE INTERNET	7/10/02	[illegible]
UPDATED	2/20/03	[illegible]
UPDATED	11/20/03	[illegible]

(RIGHT OUTSIDE)

ISSUE SLIP ST··PLE AREA (for additional cross references)

POSITION	INITIALS	ID NO.	DATE
FEE DETERMINATI N	NW	76534	01-19-99
O.I.P.E. CLASSIFIER		10	1/20
FORMALITY REVIEW	JC	71470	1/28/99

INDEX OF CLAIMS

✔ .... Rejected		N .... Non-elected	
= .... Allowed		I .... Interference	
— (Through numeral)... Canceled		A .... Appeal	
÷ .... Restricted		O .... Objected	

Claim Final	Claim Original	Date 7/11/02	Date 2/21/03	Date 11/20/03	Date	Date	Date	Date	Date	Date	Date
	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										
	32										
	33										
	34										
	35										
	36										
	37										
	38										
	39										
	40										
	41										
	42										
	43										
	44										
	45										
	46										
	47										
	48										
	49										
	50	✓	✓	✓							

Claim Final	Claim Original	Date 7/11/02	Date 2/21/03	Date 11/20/03	Date	Date	Date	Date	Date	Date	Date
	51	✓	✓	✓							
	52										
	53										
	54										
	55										
	56										
	57										
	58										
	59										
	60										
	61										
	62										
	63										
	64										
	65										
	66										
	67										
	68										
	69										
	70										
	71										
	72										
	73										
	74										
	75										
	76										
	77										
	78										
	79										
	80										
	81										
	82										
	83										
	84										
	85										
	86										
	87										
	88										
	89	✓	✓	✓							
	90										
	91										
	92										
	93										
	94										
	95										
	96										
	97										
	98										
	99										
	100										

Claim Final	Claim Original	Date	Date	Date	Date	Date	Date	Date	Date	Date	Date
	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										

If more than 150 claims or 10 actions
staple additional sheet here

(LEFT INSIDE)

A

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF EXPRESS MAILING
I hereby certify that this paper and the documents and/or fees referred to as attached therein are being deposited with the United States Postal Service on January 05, 1999 in an envelope as "Express Mail Post Office to Addressee" service under 37 CFR §1.10, Mailing Label Number EL221766053US, addressed to the Assistant Commissioner for Patents, Washington, DC 20231.

Michael L. Gough

Attorney Docket No.: SRI1P016
First Named Inventor:
CHEYER, Adam J.

UTILITY PATENT APPLICATION TRANSMITTAL (37 CFR § 1.53(b))

Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

☐ Duplicate for fee processing

Sir: This is a request for filing a patent application under 37 CFR § 1.53(b) in the name of inventors: Adam J. Cheyer and David L. Martin

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

Application Elements:

☒ 59 Pages of Specification, Claims and Abstract
☒ 16 Sheets of Drawings
☒ 01 Pages Combined Declaration and Power of Attorney

Accompanying Application Parts:

☒ Assignment and Assignment Recordation Cover Sheet (recording fee not enclosed)
☒ Return Receipt Postcard

Fee Calculation (37 CFR § 1.16)

	(Col. 1) NO. FILED		(Col. 2) NO. EXTRA	SMALL ENTITY RATE	FEE	OR	LARGE ENTITY RATE	FEE
BASIC FEE				$395	$	OR	$760	$ 760.00
TOTAL CLAIMS	89	-20 =	69	x11 =	$	OR	x18 =	$1242.00
INDEP CLAIMS	06	-03 =	03	x41 =	$	OR	x78 =	$ 234.00
* If the difference in Col. 1 is less than zero, enter "0" in Col. 2.				Total	$	OR	Total	$2236.00

Including filing fees and the assignment recordation fee of $40.00, the Commissioner is authorized to charge all required fees to Deposit Account No. 50-0384 (Order No. SRI1P016).

☒ The Commissioner is authorized to charge any fees beyond the amount enclosed which may be required, or to credit any overpayment, to Deposit Account No. 50-0384 (Order No. SRI1P016).

General Authorization for Petition [illegible] Extension of Time (37 CFR §1.136)

☒ Applicants hereby make and generally authorize any Petitions for Extensions of Time as may be needed for any subsequent filings. The Commissioner is also authorized to charge any extension fees under 37 CFR §1.17 as may be needed to Deposit Account No. 50-0384.

☒ Please send correspondence to the following address:

Brian R. Coleman
HICKMAN STEPHENS & COLEMAN, LLP
P.O. Box 52037
Palo Alto, CA 94303-0746

Tel (650) 470-7430
Fax (650) 470-7440

Date: 1/5/99

Brian R. Coleman
Registration No. **39,145**

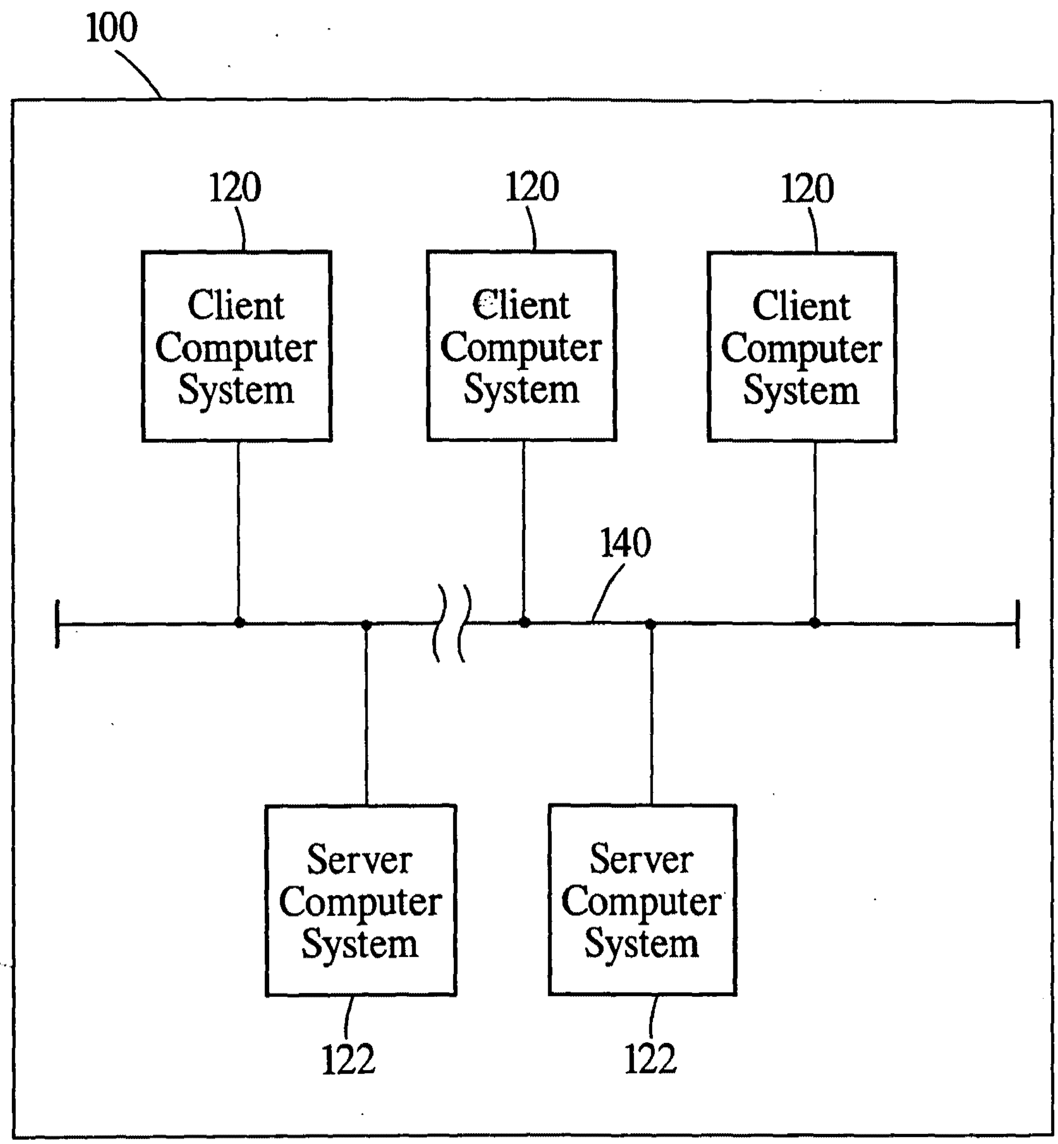


Fig. 1
(Prior Art)

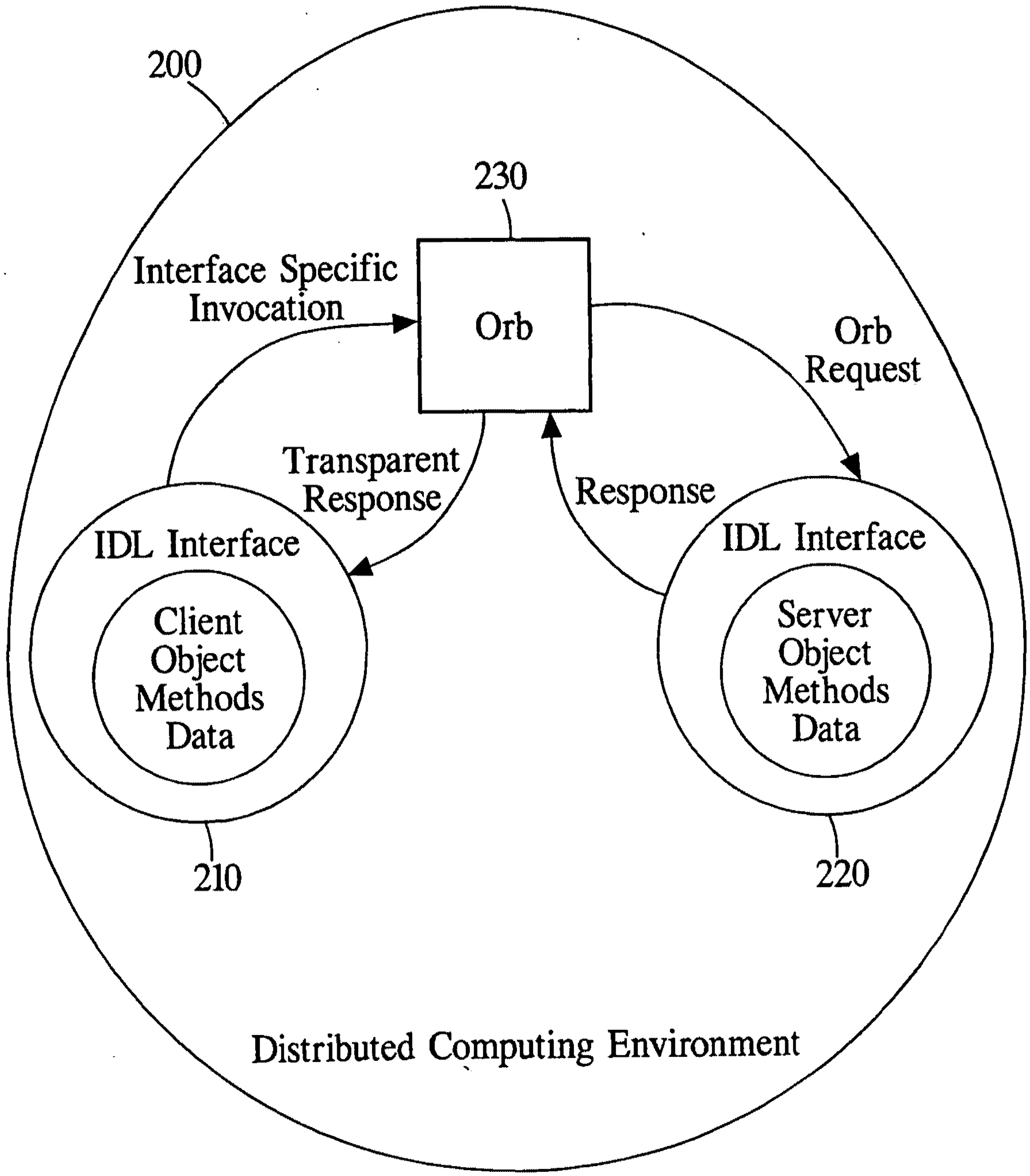


Fig. 2
(Prior Art)

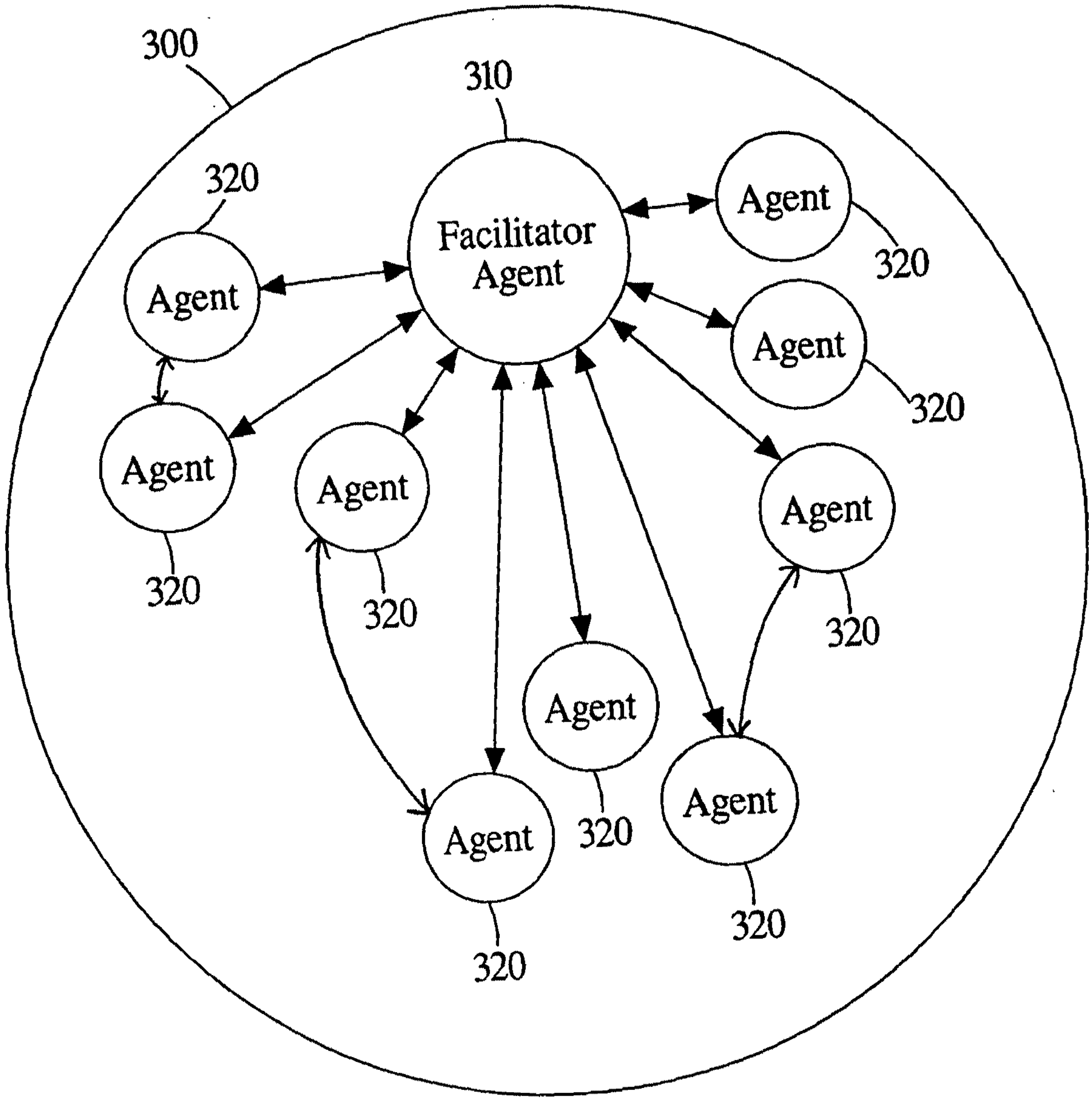

300
310
Facilitator Agent
Agent
320
Agent
320
Agent
320
Agent
320
Agent
320
Agent
320
Agent
320
Agent
320
Agent
320
Agent
320

Fig. 3

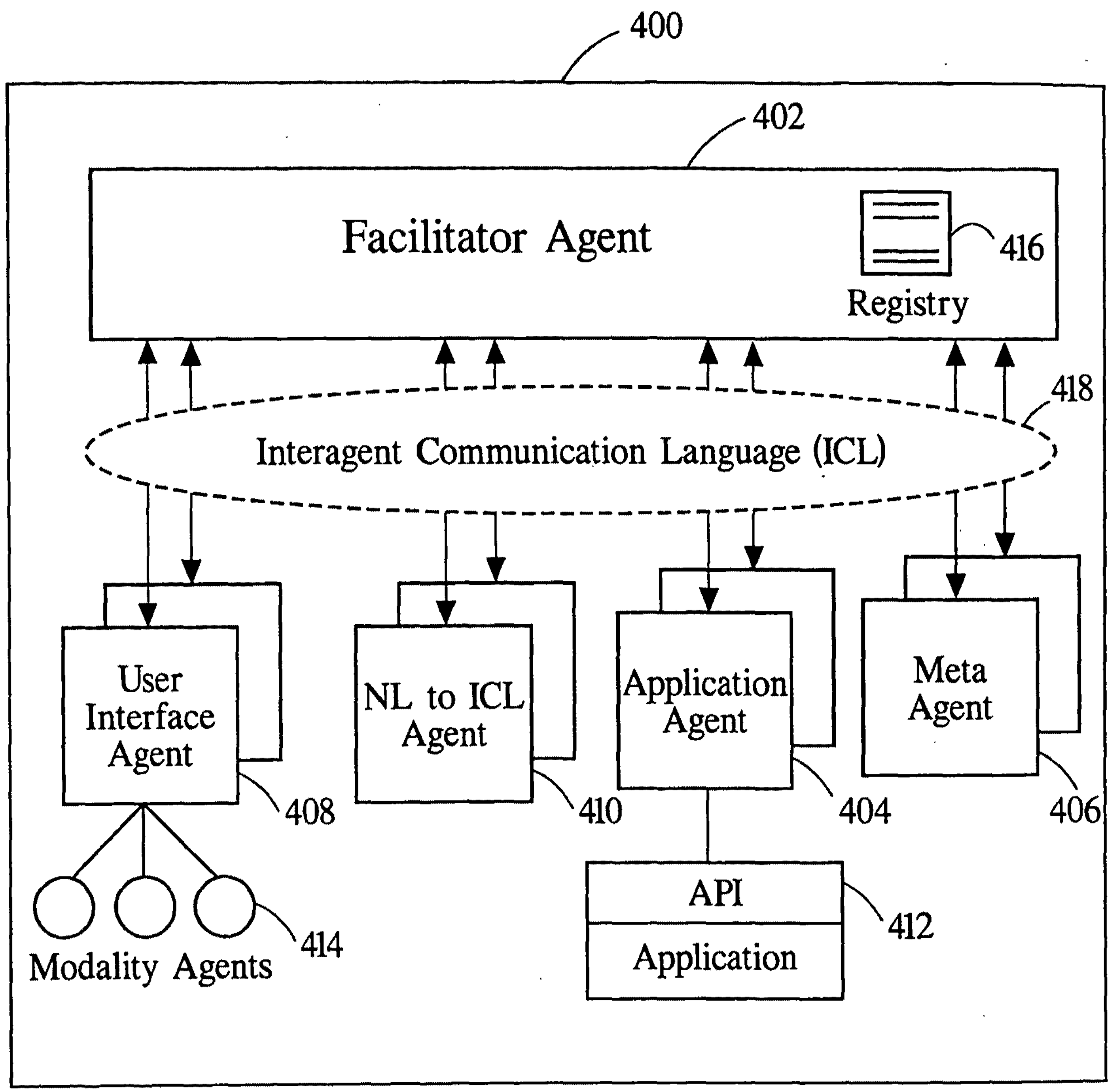

400
402
Facilitator Agent
416
Registry
418
Interagent Communication Language (ICL)
User Interface Agent
408
NL to ICL Agent
410
Application Agent
404
Meta Agent
406
Modality Agents
414
API
Application
412

Fig. 4

SRI1P016 5/16

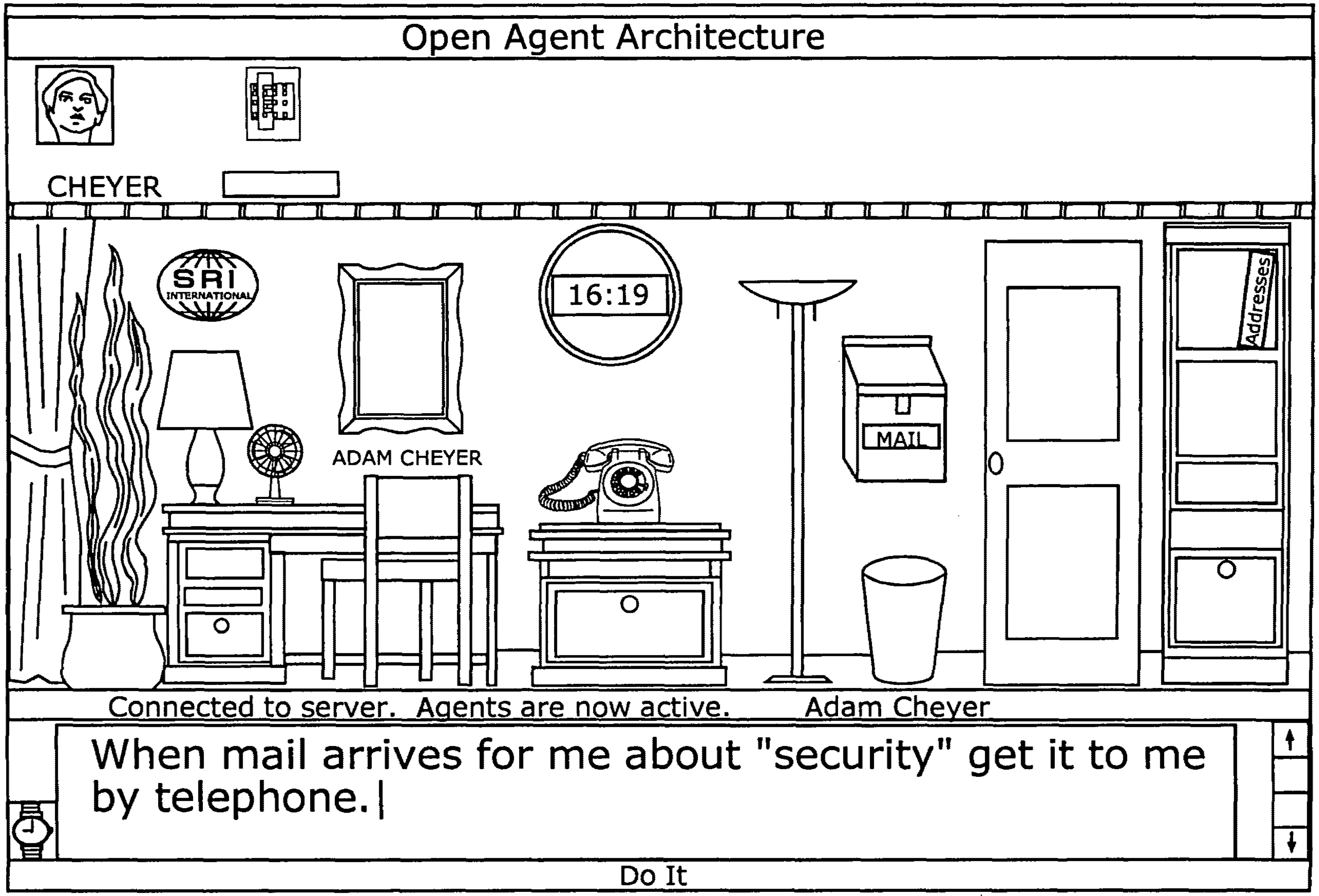


Fig. 5

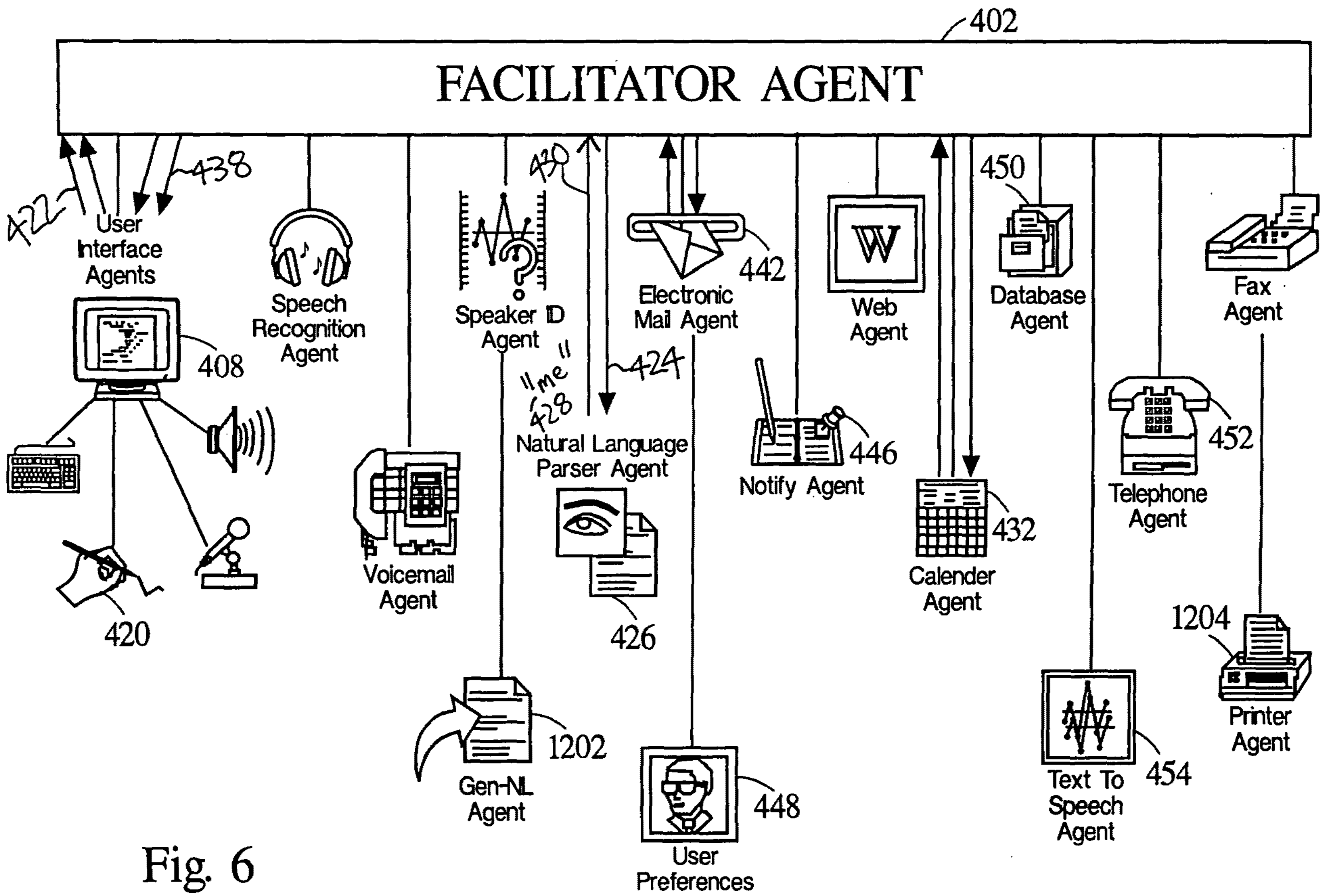


Fig. 6

SRI1P016 6/16

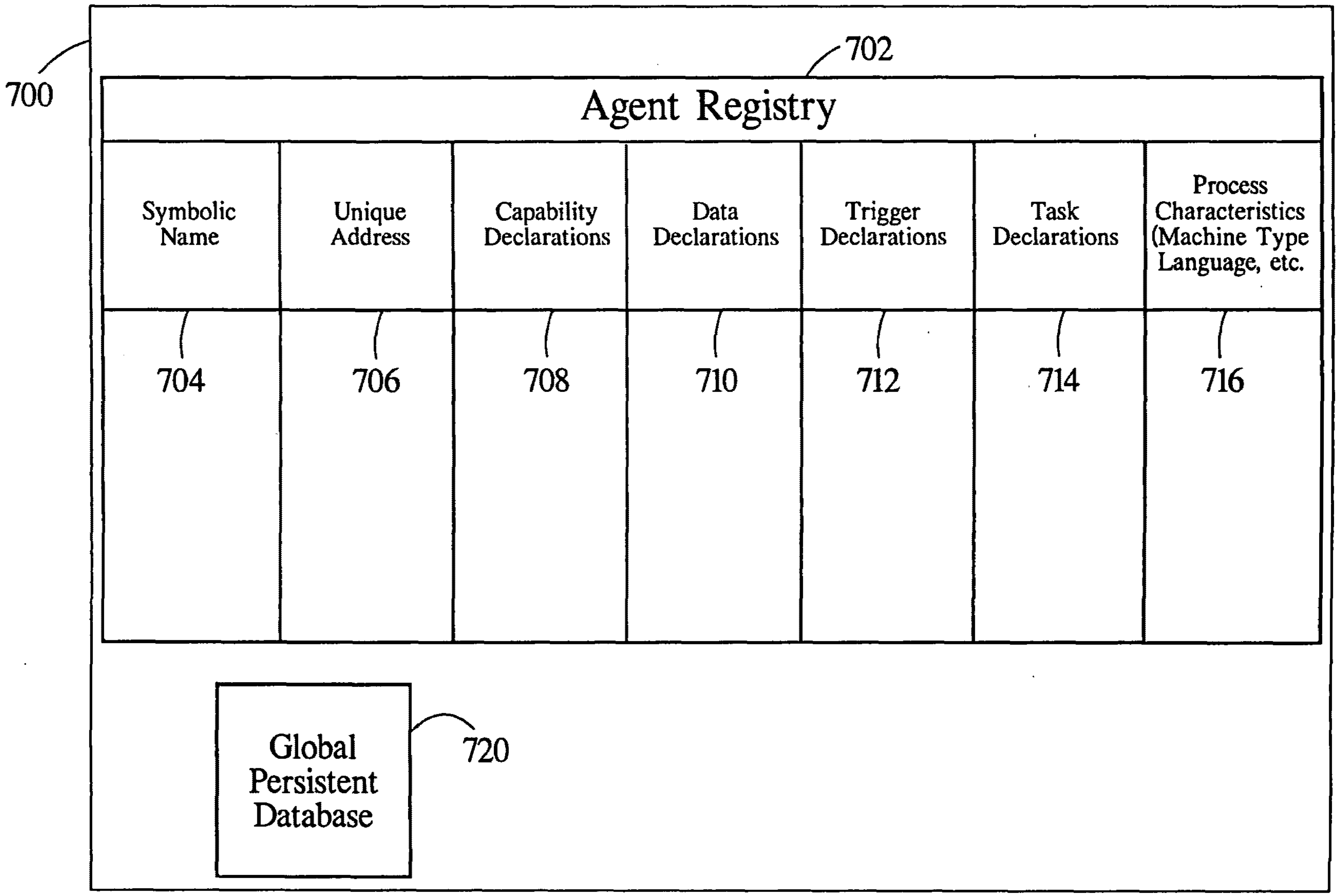
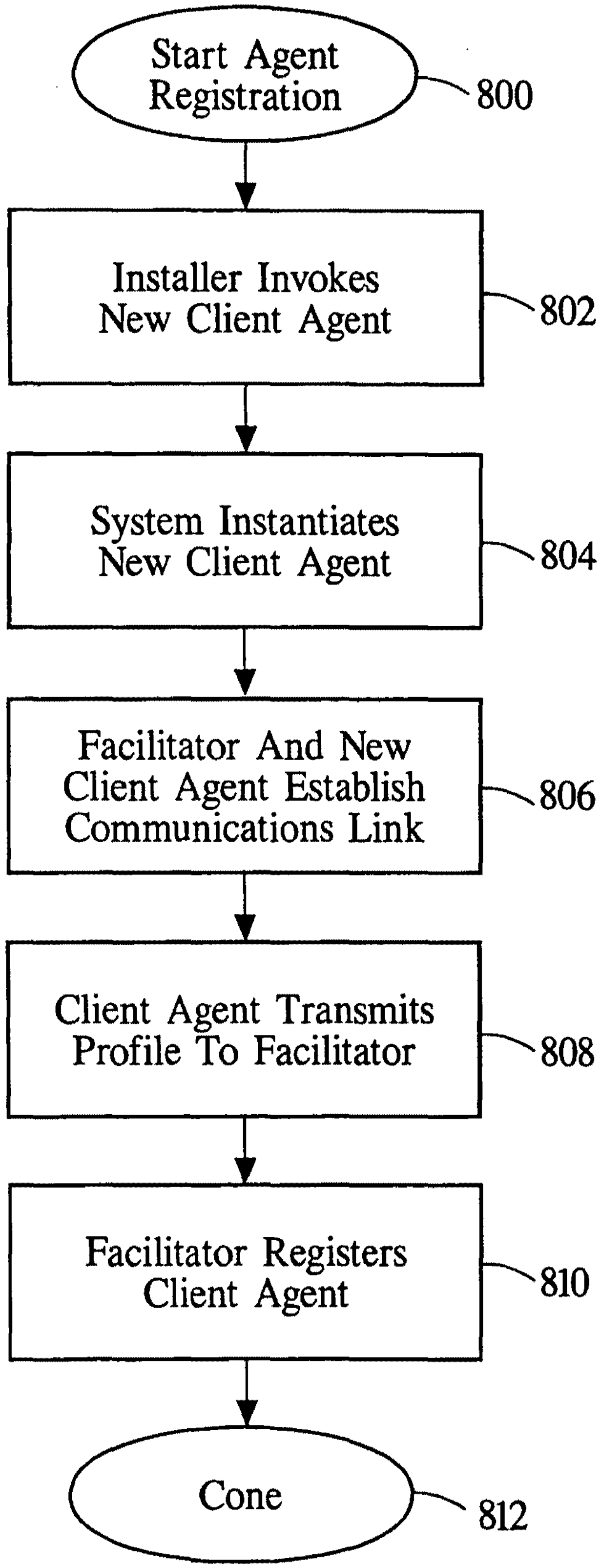

700
702
Agent Registry
Symbolic Name
Unique Address
Capability Declarations
Data Declarations
Trigger Declarations
Task Declarations
Process Characteristics (Machine Type Language, etc.
704
706
708
710
712
714
716
Global Persistent Database
720

Fig. 7



Start Agent Registration
800
Installer Invokes New Client Agent
802
System Instantiates New Client Agent
804
Facilitator And New Client Agent Establish Communications Link
806
Client Agent Transmits Profile To Facilitator
808
Facilitator Registers Client Agent
810
Cone
812

Fig. 8

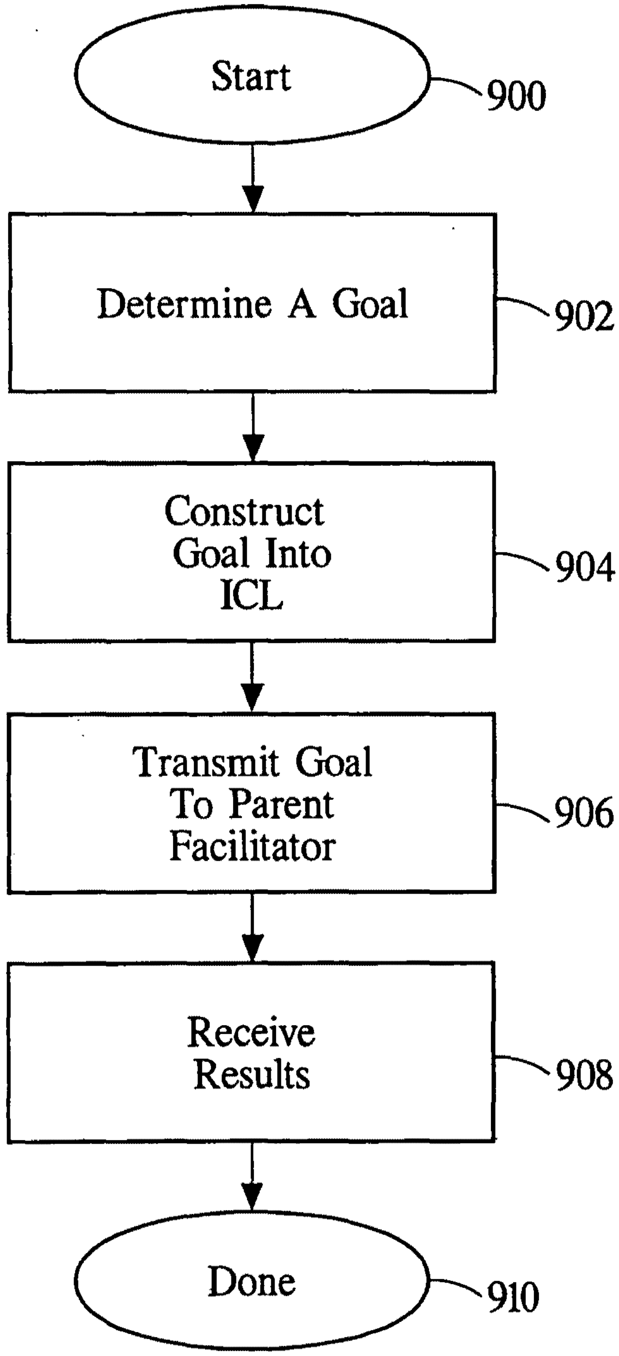

Start
900
Determine A Goal
902
Construct Goal Into ICL
904
Transmit Goal To Parent Facilitator
906
Receive Results
908
Done
910

Fig. 9

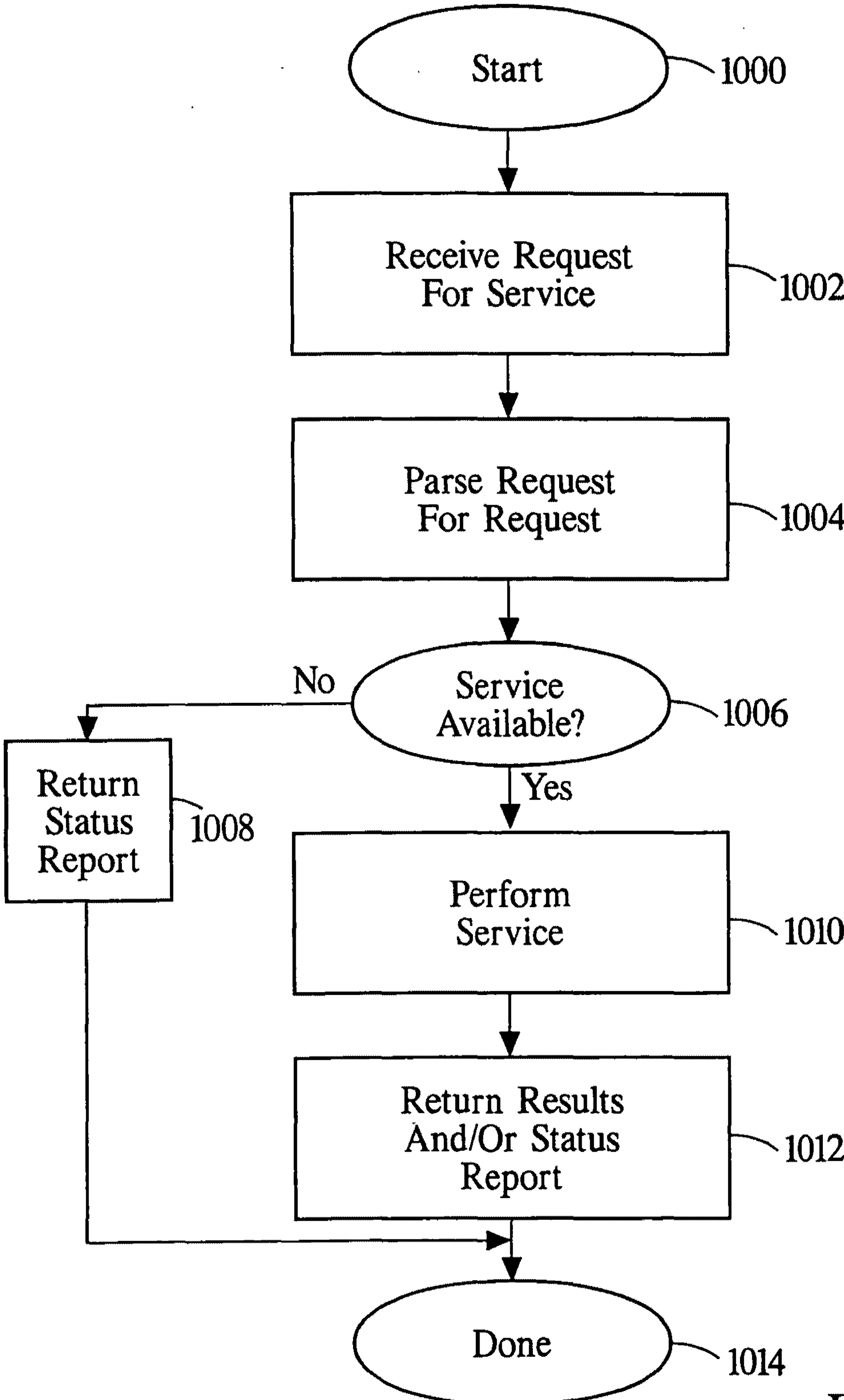

Start
1000
Receive Request For Service
1002
Parse Request For Request
1004
Service Available?
1006
No
Yes
Return Status Report
1008
Perform Service
1010
Return Results And/Or Status Report
1012
Done
1014

Fig. 10

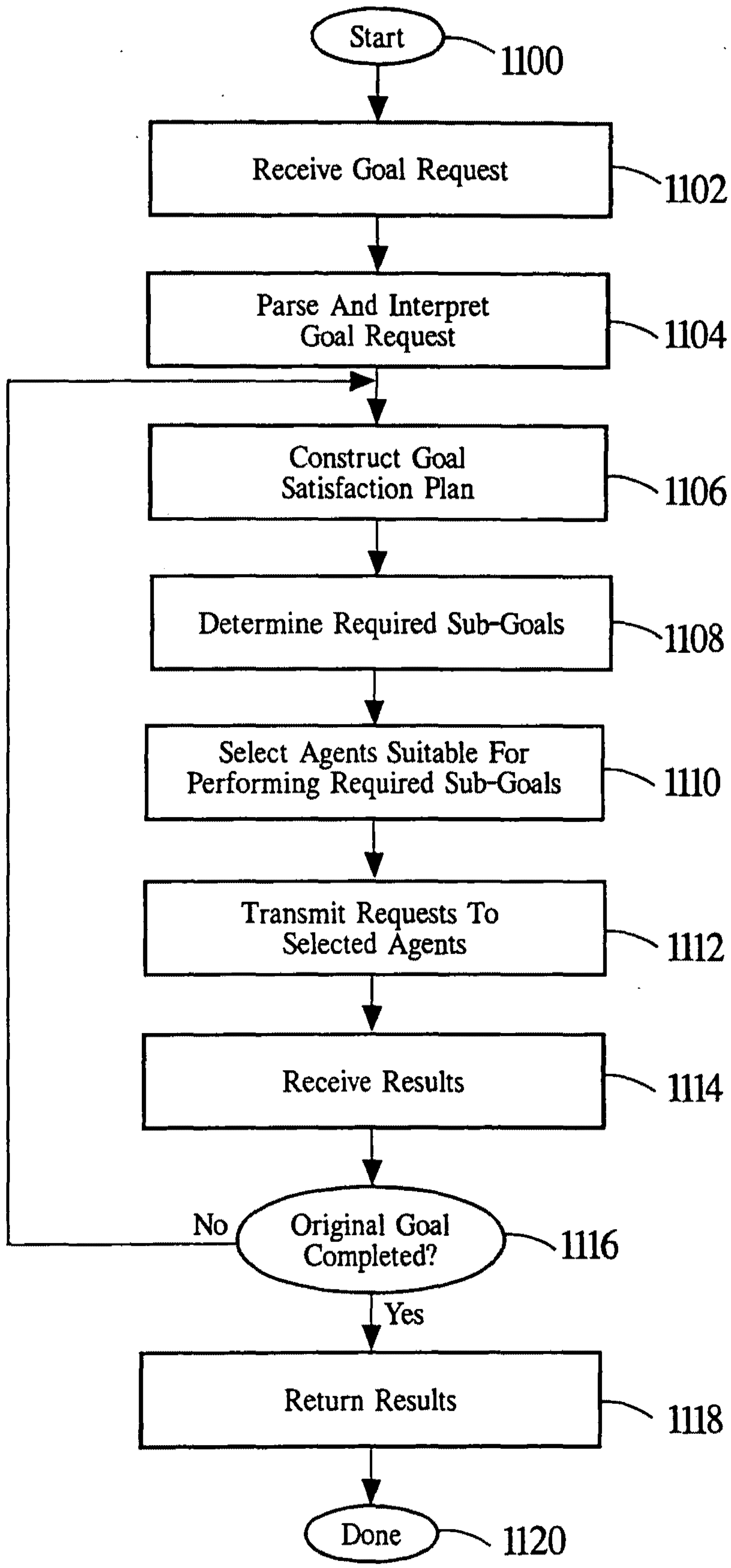
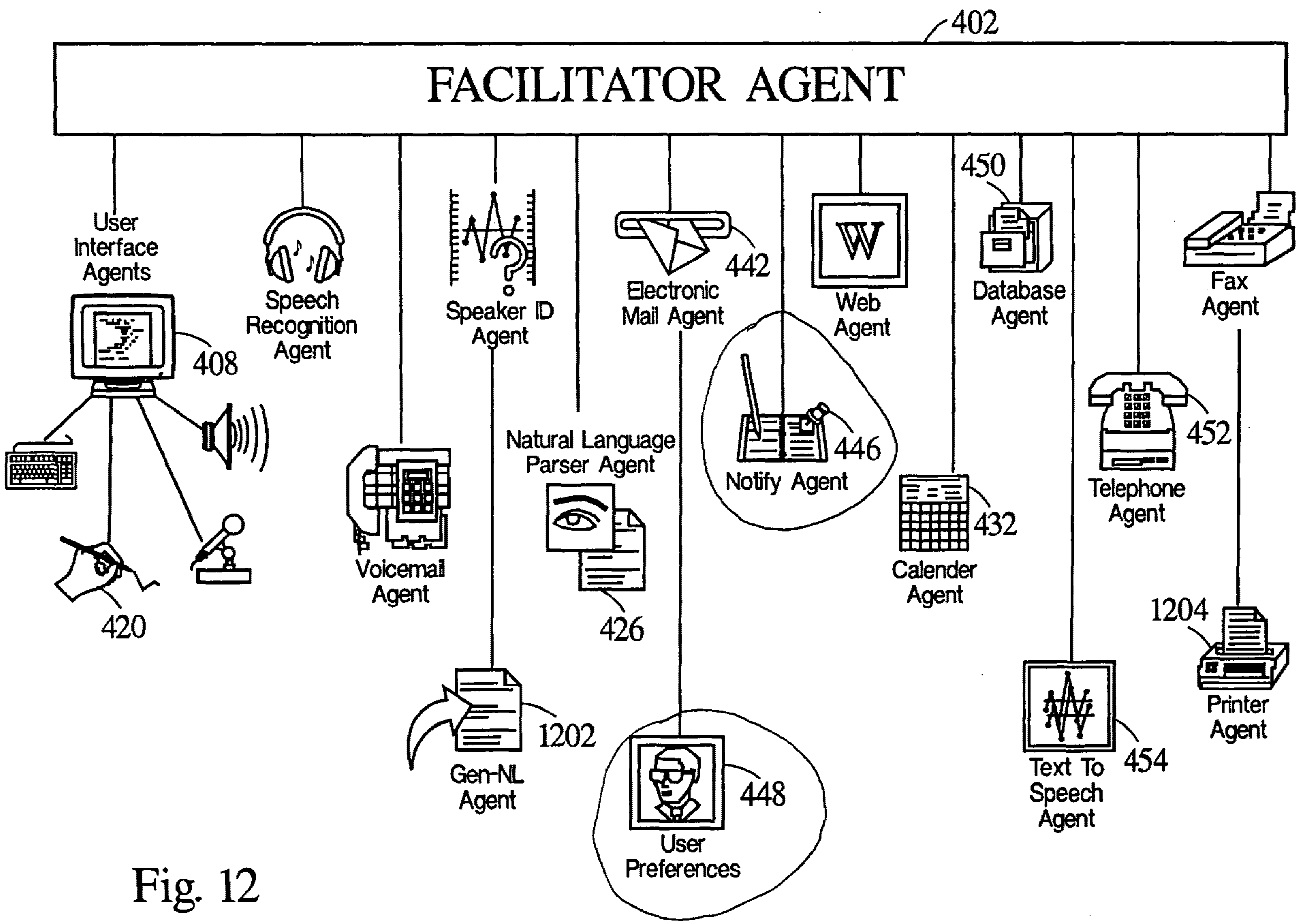

Start
1100
Receive Goal Request
1102
Parse And Interpret Goal Request
1104
Construct Goal Satisfaction Plan
1106
Determine Required Sub-Goals
1108
Select Agents Suitable For Performing Required Sub-Goals
1110
Transmit Requests To Selected Agents
1112
Receive Results
1114
No
Original Goal Completed?
1116
Yes
Return Results
1118
Done
1120

Fig. 11



SRI1P016 12/16

Fig. 12

APPROVED O.G. FIG.
BY CLASS SUBCLASS
DRAFTSMAN

SRIIP016 13/16

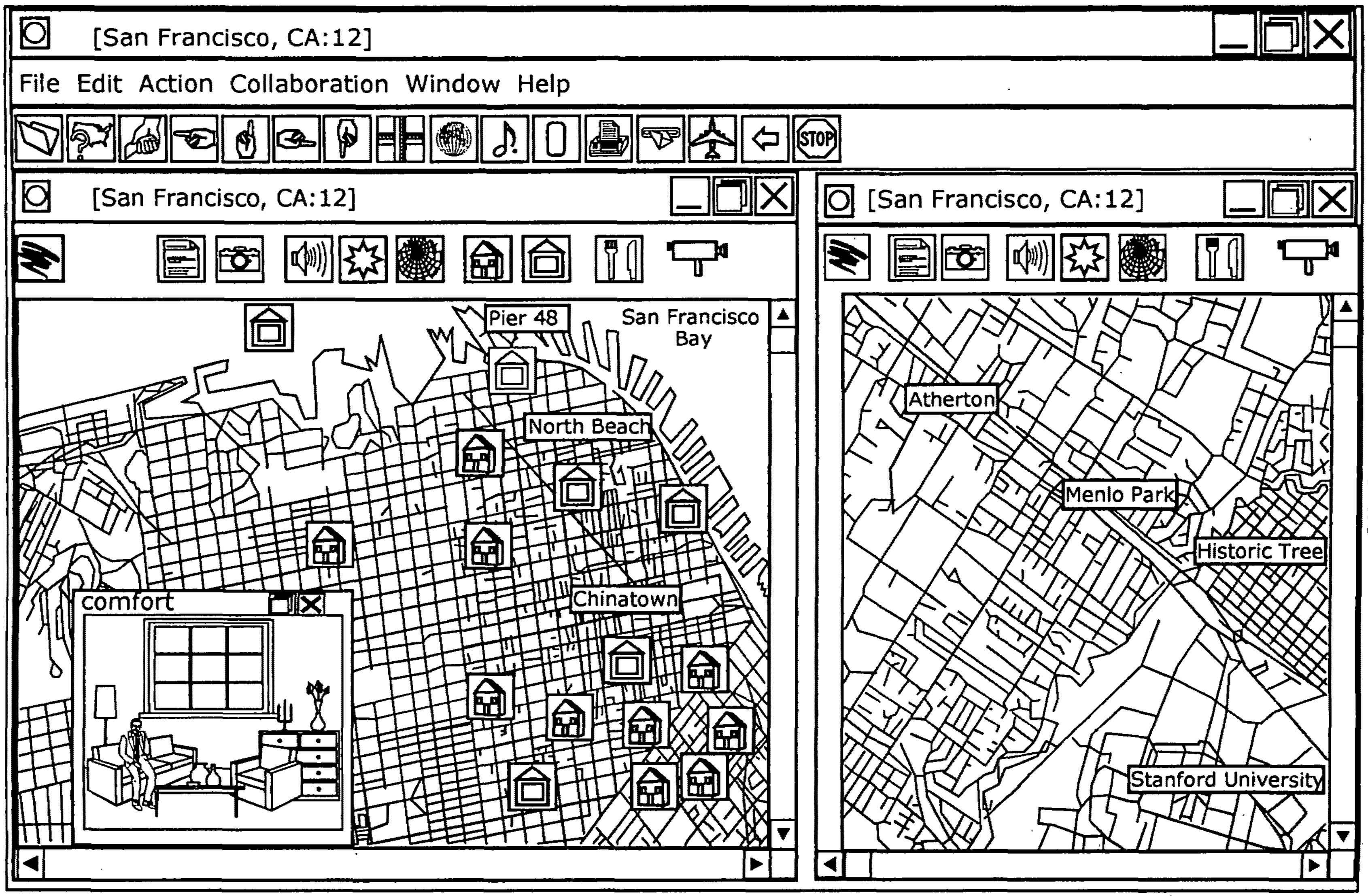


Fig. 13

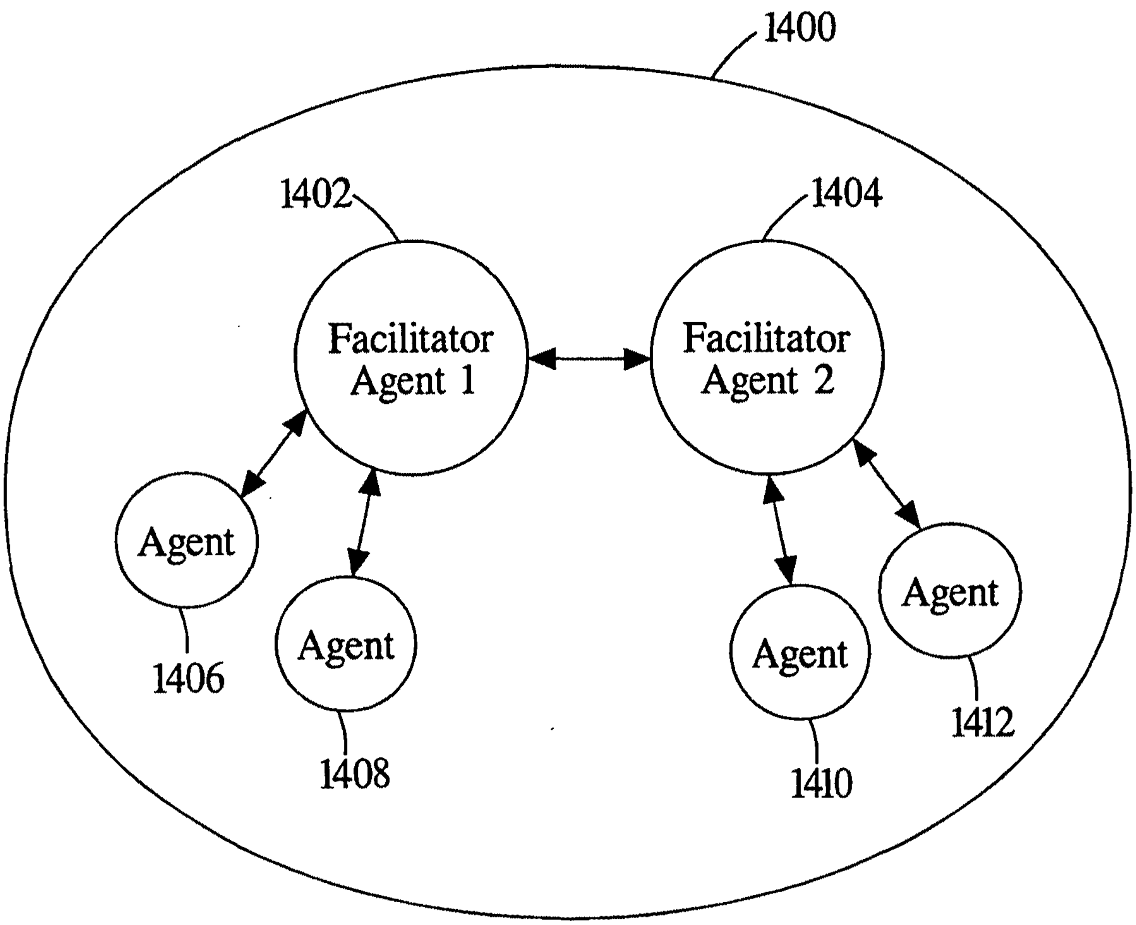

1400
1402
1404
Facilitator Agent 1
Facilitator Agent 2
Agent
Agent
Agent
Agent
1406
1408
1410
1412

Fig. 14

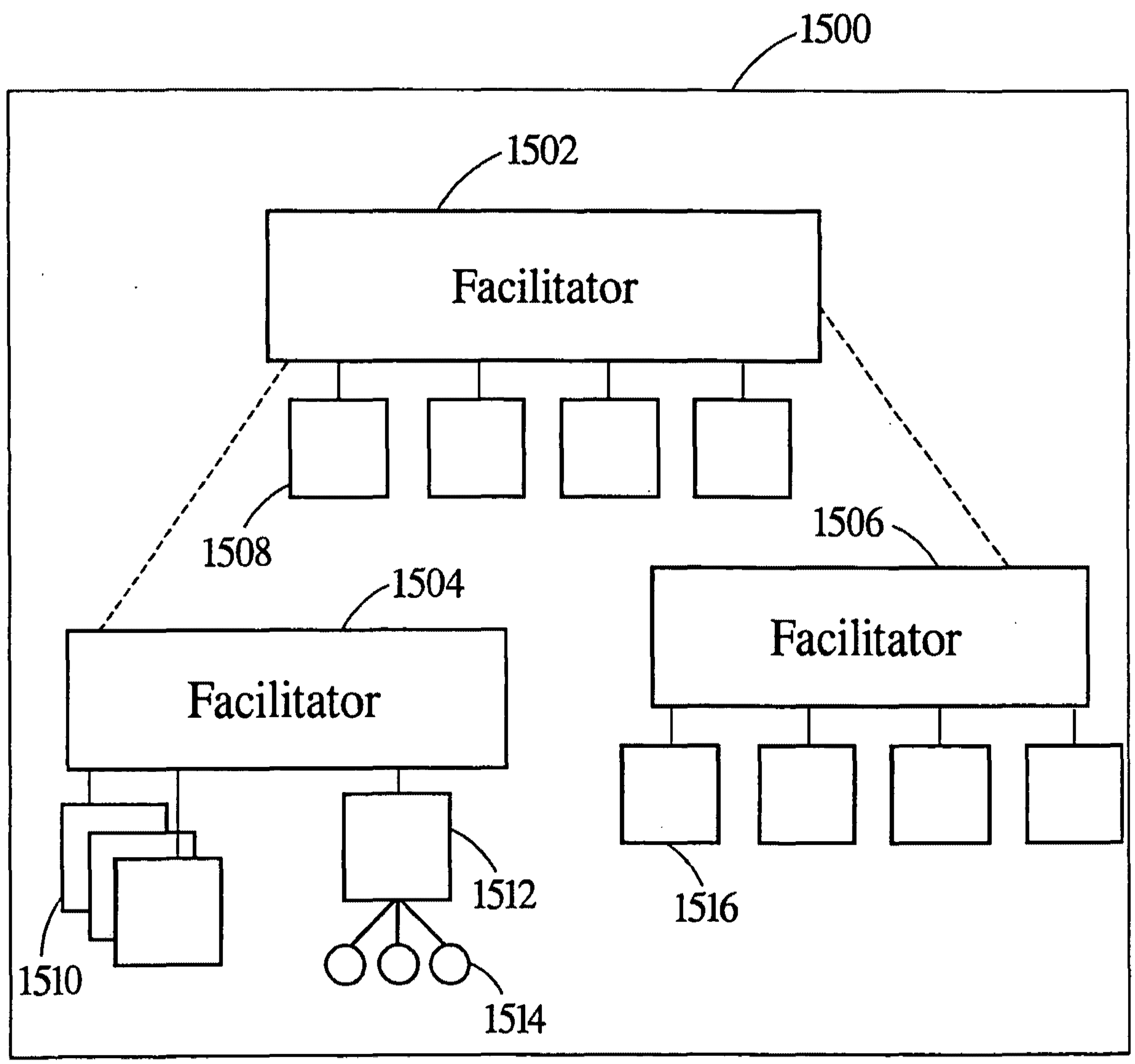

1500
1502
Facilitator
1508
1504
Facilitator
1506
Facilitator
1510
1512
1514
1516

Fig. 15

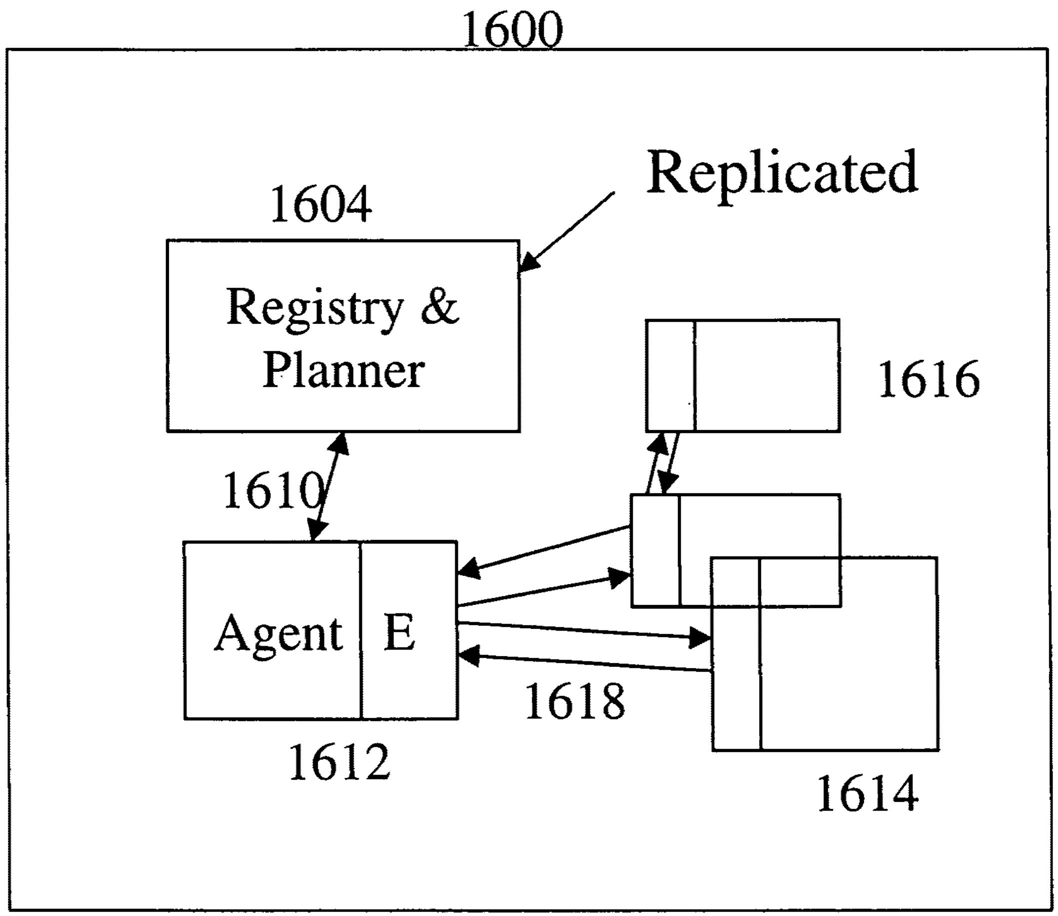


Figure 16

SRI1P016 16/

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

By:

Adam J. Cheyer and David L. Martin

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*

computers (*clients*). Some of these computers provide resources to other computers and are known as *server computers* (*servers*). The servers 122 can vary greatly in the resources they possess, access they provide and services made available to other computers across a network. Servers may service other servers as well as clients.

The Internet is a computing system based upon this network computing model. The Internet is continually growing, stimulating a paradigm shift for computing away from requiring all relevant data and programs to reside on the user's desktop machine. The data now routinely accessed from computers spread around the world has become increasingly rich in format, comprising multimedia documents, and audio and video streams. With the popularization of programming languages such as JAVA, data transported between local and remote machines may also include programs that can be downloaded and executed on the local machine. There is an ever increasing reliance on networked computing, necessitating software design approaches that allow for flexible composition of distributed processing elements in a dynamically changing and relatively unstable environment.

In an increasing variety of domains, application designers and users are coming to expect the deployment of *smarter, longer-lived, more autonomous, software applications*. Push technology, persistent monitoring of information sources, and the maintenance of user models, allowing for personalized responses and sharing of preferences, are examples of the simplest manifestations of this trend. Commercial enterprises are introducing significantly more advanced approaches, in many cases employing recent research results from artificial intelligence, data mining, machine learning, and other fields.

More than ever before, the increasing complexity of systems, the development of new technologies, and the availability of multimedia material and environments are creating a demand for *more accessible and intuitive user interfaces*. Autonomous, distributed, multi-component systems providing sophisticated services will no longer lend themselves to the familiar "direct manipulation" model of interaction, in which an individual user masters a fixed selection of commands provided by a single application. Ubiquitous computing, in networked environments, has brought about a situation in which the typical user of many software services is likely to be a non-expert, who may access a given service infrequently or only a few times.

Accommodating such usage patterns calls for new approaches. Fortunately, input modalities now becoming widely available, such as speech recognition and pen-based handwriting/gesture recognition, and the ability to manage the presentation of systems' responses by using multiple media provide an opportunity to fashion a style of human-computer interaction that draws much more heavily on our experience with human-human interactions.

PRIOR RELATED ART

Existing approaches and technologies for distributed computing include distributed objects, mobile objects, blackboard-style architectures, and agent-based software engineering.

The Distributed Object Approach

Object-oriented languages, such as C++ or JAVA, provide significant advances over standard procedural languages with respect to the reusability and modularity of code: *encapsulation*, *inheritance* and *polymorhpism*. Encapsulation encourages the creation of library interfaces that minimize dependencies on underlying algorithms or data structures. Changes to programming internals can be made at a later date with requiring modifications to the code that uses the library. Inheritance permits the extension and modification of a library of routines and data without requiring source code to the original library. Polymorphism allows one body of code to work on an arbitrary number of data types. For the sake of simplicity traditional objects may be seen to contain both methods and data. Methods provide the mechanisms by which the internal state of an object may be modified or by which communication may occur with another object or by which the instantiation or removal of objects may be directed.

With reference to Figure 2, a distributed object technology based around an Object Request Broker will now be described. Whereas "standard" object-oriented programming (OOP) languages can be used to build monolithic programs out of many object building blocks, distributed object technologies (DOOP) allow the creation of programs whose components may be spread across multiple machines. As shown in Figure 2, an object system 200 includes client objects 210 and server objects 220. To implement a client-server relationship between objects, the distributed object system

200 uses a registry mechanism (CORBA's registry is called an Object Request Broker, or ORB) 230 to store the interface descriptions of available objects. Through the services of the ORB 230, a client can transparently invoke a method on a remote server object. The ORB 230 is then responsible for finding the object 220 that can implement the request, passing it the parameters, invoking its method, and returning the results. In the most sophisticated systems, the client 210 does not have to be aware of where the object is located, its programming language, its operating system, or any other system aspects that are not part of the server object's interface.

Although distributed objects offer a powerful paradigm for creating networked applications, certain aspects of the approach are not perfectly tailored to the constantly changing environment of the Internet. A major restriction of the DOOP approach is that the interactions among objects are fixed through explicitly coded instructions by the application developer. It is often difficult to reuse an object in a new application without bringing along all its inherent dependencies on other objects (embedded interface definitions and explicit method calls). Another restriction of the DOOP approach is the result of its reliance on a remote procedure call (RPC) style of communication. Although easy to debug, this single thread of execution model does not facilitate programming to exploit the potential for parallel computation that one would expect in a distributed environment. In addition, RPC uses a blocking (synchronous) scheme that does not scale well for high-volume transactions.

Mobile Objects

Mobile objects, sometimes called mobile agents, are bits of code that can move to another execution site (presumably on a different machine) under their own programmatic control, where they can then interact with the local environment. For certain types of problems, the mobile object paradigm offers advantages over more traditional distributed object approaches. These advantages include network bandwidth and parallelism. Network bandwidth advantages exist for some database queries or electronic commerce applications, where it is more efficient to perform tests on data by bringing the tests to the data than by bringing large amounts of data to the testing program. Parallelism advantages include situations in which mobile agents can be spawned in parallel to accomplish many tasks at once.

Some of the disadvantages and inconveniences of the mobile agent approach include the programmatic specificity of the agent interactions, lack of coordination support between participant agents and execution environment irregularities regarding specific programming languages supported by host processors upon which agents reside. In a fashion similar to that of DOOP programming, an agent developer must programmatically specify where to go and how to interact with the target environment. There is generally little coordination support to encourage interactions among multiple (mobile) participants. Agents must be written in the programming language supported by the execution environment, whereas many other distributed technologies support heterogeneous communities of components, written in diverse programming languages.

Blackboard Architectures

Blackboard architectures typically allow multiple processes to communicate by reading and writing tuples from a global data store. Each process can watch for items of interest, perform computations based on the state of the blackboard, and then add partial results or queries that other processes can consider. Blackboard architectures provide a flexible framework for problem solving by a dynamic community of distributed processes. A blackboard architecture provides one solution to eliminating the tightly bound interaction links that some of the other distributed technologies require during interprocess communication. This advantage can also be a disadvantage: although a programmer does not need to refer to a specific process during computation, the framework does not provide programmatic control for doing so in cases where this would be practical.

Agent-based Software Engineering

Several research communities have approached distributed computing by casting it as a problem of modeling communication and cooperation among autonomous entities, or agents. Effective communication among independent agents requires four components: (1) a transport mechanism carrying messages in an asynchronous fashion, (2) an interaction protocol defining various types of communication interchange and their social implications (for instance, a response is expected of a question), (3) a content language permitting the expression and interpretation of utterances, and (4) an agreed-upon set of shared vocabulary and meaning for concepts (often called an *ontology*). Such mechanisms permit a much richer style of interaction among participants than can be expressed using a distributed object's RPC model or a blackboard architecture's centralized exchange approach.

Agent-based systems have shown much promise for flexible, fault-tolerant, distributed problem solving. Several agent-based projects have helped to evolve the notion of facilitation. However, existing agent-based technologies and architectures are typically very limited in the extent to which agents can specify complex goals or influence the strategies used by the facilitator. Further, such prior systems are not sufficiently attuned to the importance of integrating human agents (i.e., users) through natural language and other human-oriented user interface technologies.

The initial version of SRI International's Open Agent Architecture™ ("*OAA*®") technology provided only a very limited mechanism for dealing with compound goals. Fixed formats were available for specifying a flat list of either conjoined (AND) sub-goals or disjoined (OR) sub-goals; in both cases, parallel goal solving was hard-wired in, and only a single set of parameters for the entire list could be specified. More complex goal expressions involving (for example) combinations of different boolean connectors, nested expressions, or conditionally interdependent ("IF .. THEN") goals were not supported. Further, system scalability was not adequately addressed in this prior work.

SUMMARY OF INVENTION

A first embodiment of the present invention discloses a highly flexible, software-based architecture for constructing distributed systems. The architecture supports cooperative task completion by flexible, dynamic configurations of autonomous electronic agents. Communication and cooperation between agents are brokered by one or more facilitators, which are responsible for matching requests, from users and agents, with descriptions of the capabilities of other agents. It is not generally required that a user or agent know the identities, locations, or number of other agents involved in satisfying a request, and relatively minimal effort is involved in incorporating new agents and "wrapping" legacy applications. Extreme flexibility is achieved through an architecture organized around the declaration of capabilities by service-providing agents, the construction of arbitrarily complex goals by users and service-requesting agents, and the role of facilitators in delegating and coordinating the satisfaction of these goals, subject to advice and constraints that may accompany them. Additional mechanisms and features include facilities for creating and maintaining shared repositories of data; the use of triggers to instantiate commitments within and between agents; agent-based provision of multi-modal user interfaces, including natural language; and built-in support for including the user as a privileged member of the agent community. Specific embodiments providing enhanced scalability are also described.

BRIEF DESCRIPTION OF THE DRAWINGS

Prior Art

Prior Art FIGURE 1 depicts a networked computing model;

Prior Art FIGURE 2 depicts a distributed object technology based around an Object Resource Broker;

Examples of the Invention

FIGURE 3 depicts a distributed agent system based around a facilitator agent;

FIGURE 4 presents a structure typical of one small system of the present invention;

FIGURE 5 depicts an Automated Office system implemented in accordance with an example embodiment of the present invention supporting a mobile user with a laptop computer and a telephone;

FIGURE 6 schematically depicts an Automated Office system implemented as a network of agents in accordance with a preferred embodiment of the present invention;

FIGURE 7 schematically shows data structures internal to a facilitator in accordance with a preferred embodiment of the present invention;

FIGURE 8 depicts operations involved in instantiating a client agent with its parent facilitator in accordance with a preferred embodiment of the present invention;

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™ 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.

BRIEF DESCRIPTION OF THE APPENDICES

The Appendices provide source code for an embodiment of the present invention written in the PROLOG programming language.

APPENDIX A: Source code file named compound.pl.

APPENDIX B: Source code file named fac.pl.

APPENDIX C: Source code file named libcom_tcp.pl.

APPENDIX D: Source code file named liboaa.pl.

APPENDIX E: Source code file named translations.pl.

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

shown in Figure 4. All agents that are not facilitators are referred to herein generically as *client* agents -- so called because each acts (in some respects) as a client of some facilitator, which provides communication and other essential services for the client.

The variety of possible client agents is essentially unlimited. Some typical categories of client agents would include application agents 404, meta-agents 406, and user interface agents 408, as depicted in Figure 4. Application agents 404 denote specialists that provide a collection of services of a particular sort. These services could be domain-independent technologies (such as speech recognition, natural language processing 410, email, and some forms of data retrieval and data mining) or user-specific or domain-specific (such as a travel planning and reservations agent). Application agents may be based on legacy applications or libraries, in which case the agent may be little more than a wrapper that calls a pre-existing API 412, for example. Meta-agents 406 are agents whose role is to assist the facilitator agent 402 in coordinating the activities of other agents. While the facilitator 402 possesses domain-independent coordination strategies, meta-agents 406 can augment these by using domain- and application-specific knowledge or reasoning (including but not limited to rules, learning algorithms and planning).

With further reference to Figure 4, user interface agents 408 can play an extremely important and interesting role in certain embodiments of the present invention. By way of explanation, in some systems, a user interface agent can be implemented as a collection of "micro-agents", each monitoring a different input modality (point-and-click, handwriting, pen gestures, speech), and collaborating to produce the best interpretation of the current inputs. These micro-agents are depicted in Figure 4, for example, as Modality Agents 414. While describing such subcategories of client agents is useful for purposes of illustration and understanding, they need not be formally distinguished within the system in preferred implementations of the present invention.

The operation of one preferred embodiment of the present invention will be discussed in greater detail below, but may be briefly outlined as follows. When invoked, a client agent makes a connection to a facilitator, which is known as its *parent facilitator*. These connections are depicted as a double headed arrow between the client agent and the facilitator agent in Figure 3 and 4, for example. Upon connection, an agent registers with its parent facilitator a specification of the capabilities and services it can provide. For example, a natural language agent may register the characteristics of its available natural language vocabulary. (For more details regarding client agent connections, see the discussion of Figure 8 below.) Later during task completion, when a facilitator determines that the registered services 416 of one of its client agents will help satisfy a goal, the facilitator sends that client a request expressed in the Interagent Communication Language (*ICL*) 418. (See Figure 11 below for a more detailed discussion of the facilitator operations involved.) The agent parses this request, processes it, and returns answers or status reports to the facilitator. In processing a request, the client agent can make use of a variety of infrastructure capabilities provided in the preferred embodiment. For example, the client agent can use *ICL* 418 to request services of other agents, set triggers, and read or write shared data on the facilitator or other client agents that maintain shared data. (See the discussion of Figures 9-11 below for a more detailed discussion of request processing.)

The functionality of each client agent are made available to the agent community through registration of the client agent's capabilities with a facilitator 402. A software "wrapper" essentially surrounds the underlying application program performing the services offered by each client. The common infrastructure for constructing agents is preferably supplied by an *agent library*. The agent library is preferably accessible in the runtime environment of several different programming languages. The agent library preferably minimizes the effort required to construct a new system and maximizes the ease with which legacy systems can be “wrapped” and made compatible with the agent-based architecture of the present invention.

By way of further illustration, a representative application is now briefly presented with reference to Figures 5 and 6. In the Automated Office system depicted in Figure 5, a mobile user with a telephone and a laptop computer can access and task commercial applications such as calendars, databases, and email systems running back at the office. A user interface (UI) agent 408, shown in Figure 6, runs on the user's local laptop and is responsible for accepting user input, sending requests to the facilitator 402 for delegation to appropriate agents, and displaying the results of the distributed computation. The user may interact directly with a specific remote application by clicking on active areas in the interface, calling up a form or window for that application, and making queries with standard interface dialog mechanisms. Conversely, a user may express a task to be executed by using typed, handwritten, or spoken (over the telephone) English sentences, without explicitly specifying which agent or agents should perform the task.

For instance, if the question "What is my schedule?" is written 420 in the user interface 408, this request will be sent 422 by the UI 408 to the facilitator 402, which in turn will ask 424 a natural language (NL) agent 426 to translate the query into *ICL* 18. To accomplish this task, the NL agent 426 may itself need to make requests of the agent community to resolve unknown words such as "me" 428 (the UI agent 408 can respond 430 with the name of the current user) or "schedule" 432 (the calendar agent 434 defines this word 436). The resulting *ICL* expression is then routed by the facilitator 402 to appropriate agents (in this case, the calendar agent 434) to execute the request. Results are sent back 438 to the UI agent 408 for display.

The spoken request "When mail arrives for me about security, notify me immediately." produces a slightly more complex example involving communication among all agents in the system. After translation into *ICL* as described above, the facilitator installs a trigger 440 on the mail agent 442 to look for new messages about security. When one such message does arrive in its mail spool, the trigger fires, and the facilitator matches the action part of the trigger to capabilities published by the notification agent 446. The notification agent 446 is a meta-agent, as it makes use of rules concerning the optimal use of different output modalities (email, fax, speech generation over the telephone) plus information about an individual user's preferences 448 to determine the best way of relaying a message through available media transfer application agents. After some competitive parallelism to locate the user (the calendar agent 434 and database agent 450 may have different guesses as to where to find the user) and some cooperative parallelism to produce required information (telephone number of location, user password, and an audio file containing a text-to-speech representation of the email message), a telephone agent 452 calls the user, verifying its identity through touchtones, and then play the message.

The above example illustrates a number of inventive features. As new agents connect to the facilitator, registering capability specifications and natural language vocabulary, what the user can say and do dynamically changes; in other words, the ICL is dynamically ***expandable***. For example, adding a calendar agent to the system in the previous example and registering its capabilities enables users to ask natural language questions about their "schedule" without any need to revise code for the facilitator, the natural language agents, or any other client agents. In addition, the interpretation and execution of a task is a distributed process, with no single agent defining the set of possible inputs to the system. Further, a single request can produce cooperation and flexible communication among many agents, written in different programming languages and spread across multiple machines.

Design Philosophy and Considerations

One preferred embodiment provides an integration mechanism for heterogeneous applications in a distributed infrastructure, incorporating some of the dynamism and extensibility of blackboard approaches, the efficiency associated with mobile objects, plus the rich and complex interactions of communicating agents. Design goals for preferred embodiments of the present invention may be categorized under the general headings of *interoperation and cooperation, user interfaces*, and *software engineering*. These design goals are not absolute requirements, nor will they necessarily be satisfied by all embodiments of the present invention, but rather simply reflect the inventor's currently preferred design philosophy.

Versatile mechanisms of interoperation and cooperation

Interoperation refers to the ability of distributed software components - agents - to communicate meaningfully. While every system-building framework must provide mechanisms of interoperation at some level of granularity, agent-based frameworks face important new challenges in this area. This is true primarily because autonomy, the hallmark of *individual* agents, necessitates greater flexibility in interactions within *communities* of agents. *Coordination* refers to the mechanisms by which a community of agents is able to work together productively on some task. In these areas, the goals for our framework are to *provide flexibility in assembling* *communities of autonomous service providers, provide flexibility in structuring cooperative interactions, impose the right amount of structure*, as well as *include legacy and "owned-elsewhere" applications.*

Provide flexibility in assembling communities of autonomous service providers -- both at development time and at runtime. Agents that conform to the linguistic and ontological requirements for effective communication should be able to participate in an agent community, in various combinations, with minimal or near minimal prerequisite knowledge of the characteristics of the other players. Agents with duplicate and overlapping capabilities should be able to coexist within the same community, with the system making optimal or near optimal use of the redundancy.

Provide flexibility in structuring cooperative interactions among the members of a community of agents. A framework preferably provides an economical mechanism for setting up a variety of interaction patterns among agents, without requiring an inordinate amount of complexity or infrastructure within the individual agents. The provision of a service should be independent or minimally dependent upon a particular configuration of agents.

Impose the right amount of structure on individual agents. Different approaches to the construction of multi-agent systems impose different requirements on the individual agents. For example, because KQML is neutral as to the content of messages, it imposes minimal structural requirements on individual agents. On the other hand, the BDI paradigm tends to impose much more demanding requirements, by making assumptions about the nature of the programming elements that are meaningful to individual agents. Preferred embodiments of the present invention should fall somewhere between the two, providing a rich set of interoperation and coordination capabilities, without precluding any of the software engineering goals defined below.

Include legacy and "owned-elsewhere" applications. Whereas *legacy* usually implies reuse of an established system fully controlled by the agent-based system developer, *owned-elsewhere* refers to applications to which the developer has partial access, but no control. Examples of owned-elsewhere applications include data sources and services available on the World Wide Web, via simple form-based interfaces, and applications used cooperatively within a virtual enterprise, which remain the properties of separate corporate entities. Both classes of application must preferably be able to interoperate, more or less as full-fledged members of the agent community, without requiring an overwhelming integration effort.

Human-oriented user interfaces

Systems composed of multiple distributed components, and possibly dynamic configurations of components, require the crafting of intuitive user interfaces to *provide conceptually natural interaction mechanisms, treat users as privileged members of the agent community* and *support collaboration.*

Provide conceptually natural interaction mechanisms with multiple distributed components. When there are numerous disparate agents, and/or complex tasks implemented by the system, the user should be able to express requests without having detailed knowledge of the individual agents. With speech recognition, handwriting recognition, and natural language technologies becoming more mature, agent architectures should preferably support these forms of input playing increased roles in the tasking of agent communities.

Preferably treat *users as privileged members* of the agent community by providing an appropriate level of task specification within *software* agents, and reusable translation mechanisms between this level and the level of *human* requests, supporting constructs that seamlessly incorporate interactions between both human-interface and software types of agents.

Preferably support *collaboration* (simultaneous work over shared data and processing resources) between users and agents.

Realistic software engineering requirements

System-building frameworks should preferably address the practical concerns of real-world applications by the specification of requirements which preferably include: *Minimize the effort* required to create new agents, and to wrap existing applications. *Encourage reuse,* both of domain-independent and domain-specific components. The concept of *agent orientation*, like that of object orientation, provides a natural conceptual framework for reuse, so long as mechanisms for encapsulation and interaction are structured appropriately. *Support lightweight, mobile platforms.* Such platforms should be able to serve as hosts for agents, without requiring the installation of a massive environment. It should also be possible to construct individual agents that are relatively small and modest in their processing requirements. *Minimize platform and language barriers.* Creation of new agents, as well as wrapping of existing applications, should not require the adoption of a new language or environment.

Mechanisms of Cooperation

Cooperation among agents in accordance with the present invention is preferably achieved via messages expressed in a common language, *ICL*. Cooperation among agent is further preferably structured around a three-part approach: providers of services register capabilities specifications with a facilitator, requesters of services construct goals and relay them to a facilitator, and facilitators coordinate the efforts of the appropriate service providers in satisfying these goals.

The Interagent Communication Language (ICL)

Interagent Communication Language (*"ICL"*) 418 refers to an interface, communication, and task coordination language preferably shared by all agents, regardless of what platform they run on or what computer language they are programmed in. *ICL* may be used by an agent to task itself or some subset of the agent community. Preferably, *ICL* allows agents to specify explicit control parameters while simultaneously supporting expression of goals in an underspecified, loosely constrained manner. In a further preferred embodiment, agents employ *ICL* to perform queries, execute actions, exchange information, set triggers, and manipulate data in the agent community.

In a further preferred embodiment, a program element expressed in *ICL* is the *event*. The activities of every agent, as well as communications between agents, are preferably structured around the transmission and handling of events. In communications, events preferably serve as messages between agents; in regulating the activities of individual agents, they may preferably be thought of as goals to be satisfied. Each event preferably has a type, a set of parameters, and content. For example, the agent library procedure *oaa_Solve* can be used by an agent to request services of other agents. A call to *oaa_Solve*, within the code of agent *A*, results in an event having the form

ev_post_solve(Goal, Params)

going from *A* to the facilitator, where *ev_post_solve* is the type, *Goal* is the content, and *Params* is a list of parameters. The allowable content and parameters preferably vary according to the type of the event.

The *ICL* preferably includes a layer of conversational protocol and a content layer. The conversational layer of *ICL* is defined by the event types, together with the parameter lists associated with certain of these event types. The content layer consists of the specific goals, triggers, and data elements that may be embedded within various events.

The *ICL* conversational protocol is preferably specified using an orthogonal, parameterized approach, where the conversational aspects of each element of an interagent conversation are represented by a selection of an event type and a selection of values from at least one orthogonal set of parameters. This approach offers greater expressiveness than an approach based solely on a fixed selection of *speech acts*, such as embodied in KQML. For example, in KQML, a request to satisfy a query can employ either of the performatives *ask_all* or *ask_one*. In *ICL*, on the other hand, this type of request preferably is expressed by the event type *ev_post_solve*, together with the *solution_limit(N)* parameter - where *N* can be any positive integer. (A request for all solutions is indicated by the omission of the *solution_limit* parameter.) The request can also be accompanied by other parameters, which combine to further refine its semantics. In KQML, then, this example forces one to choose between two possible conversational options, neither of which may be precisely what is desired. In either case, the performative chosen is a single value that must capture the entire conversational characterization of the communication. This requirement raises a difficult challenge for the language designer, to select a set of performatives that provides the desired functionality without becoming unmanageably large. Consequently, the debate over the right set of performatives has consumed much discussion within the KQML community.

The content layer of the *ICL* preferably supports unification and other features found in logic programming language environments such as PROLOG. In some embodiments, the content layer of the *ICL* is simply an extension of at least one programming language. For example, the Applicants have found that PROLOG is suitable for implementing and extending into the content layer of the *ICL*. The agent libraries preferably provide support for constructing, parsing, and manipulating *ICL* expressions. It is possible to embed content expressed in other languages within an *ICL* event. However, expressing content in *ICL* simplifies the facilitator's access to the content, as well as the conversational layer, in delegating requests. This gives the facilitator more information about the nature of a request and helps the facilitator decompose compound requests and delegate the sub-requests.

Further, *ICL* expressions preferably include, in addition to events, at least one of the following: capabilities declarations, requests for services, responses to requests, trigger specifications, and shared data elements. A further preferred embodiment of the present invention incorporates *ICL* expressions including at least all of the following: events, capabilities declarations, requests for services, responses to requests, trigger specifications, and shared data elements.

Providing Services: Specifying "Solvables"

In a preferred embodiment of the present invention, every participating agent defines and publishes a set of capability declarations, expressed in *ICL*, describing the services that it provides. These declarations establish a high-level interface to the agent. This interface is used by a facilitator in communicating with the agent, and, most important, in delegating service requests (or parts of requests) to the agent. Partly due to the use of PROLOG as a preferred basis for *ICL*, these capability declarations are referred as *solvables*. The agent library preferably provides a set of procedures allowing an agent to add, remove, and modify its solvables, which it may preferably do at any time after connecting to its facilitator.

There are preferably at least two major types of solvables: *procedure* solvables and *data* solvables. Intuitively, a procedure solvable performs a test or action, whereas a data solvable provides access to a collection of data. For example, in creating an agent for a mail system, procedure solvables might be defined for sending a message to a person, testing whether a message about a particular subject has arrived in the mail queue, or displaying a particular message onscreen. For a database wrapper agent, one might define a distinct data solvable corresponding to each of the relations present in the database. Often, a data solvable is used to provide a *shared* data store, which may be not only queried, but also updated, by various agents having the required permissions.

There are several primary technical differences between these two types of solvables. First, each procedure solvable must have a handler declared and defined for it, whereas this is preferably not necessary for a data solvable. The handling of requests for a data solvable is preferably provided transparently by the agent library. Second, data solvables are preferably associated with a dynamic collection of facts (or clauses), which may be further preferably modified at runtime, both by the agent providing the solvable, and by other agents (provided they have the required permissions). Third, special features, available for use with data solvables, preferably facilitate maintaining the associated facts. In spite of these differences, it should be noted that the mechanism of *use* by which an agent requests a service is the same for the two types of solvables.

In one embodiment, a request for one of an agent's services normally arrives in the form of an event from the agent's facilitator. The appropriate handler then deals with this event. The handler may be coded in whatever fashion is most appropriate, depending on the nature of the task, and the availability of task-specific libraries or legacy code, if any. The only hard requirement is that the handler return an appropriate response to the request, expressed in *ICL*. Depending on the nature of the request, this response could be an indication of success or failure, or a list of solutions (when the request is a data query).

A solvable preferably has three parts: a *goal*, a list of *parameters*, and a list of *permissions*, which are declared using the format:

solvable(Goal, Parameters, Permissions)

The goal of a solvable, which syntactically takes the preferable form of an *ICL* structure, is a logical representation of the service provided by the solvable. (An *ICL* structure consists of a *functor* with 0 or more arguments. For example, in the structure a(b,c), \`a' is the functor, and \`b' and \`c' the arguments.) As with a PROLOG structure, the goal's arguments themselves may preferably be structures.

Various options can be included in the parameter list, to refine the semantics associated with the solvable. The *type* parameter is preferably used to say whether the solvable is *data* or *procedure*. When the type is *procedure*, another parameter may be used to indicate the handler to be associated with the solvable. Some of the parameters appropriate for a *data* solvable are mentioned elsewhere in this application. In either case (procedure or data solvable), the *private* parameter may be preferably used to restrict the use of a solvable to the declaring agent when the agent intends the solvable to be solely for its internal use but wishes to take advantage of the mechanisms in accordance with the present invention to access it, or when the agent wants the solvable to be available to outside agents only at selected times. In support of the latter case, it is preferable for the agent to change the status of a solvable from private to non-private at any time.

The permissions of a solvable provide mechanisms by which an agent may preferably control access to its services allowing the agent to restrict calling and writing of a solvable to itself and/or other selected agents. (*Calling* means requesting the service encapsulated by a solvable, whereas *writing* means modifying the collection of facts associated with a data solvable.) The default permission for every solvable in a further preferred embodiment of the present invention is to be callable by anyone, and for data solvables to be writable by anyone. A solvable's permissions can preferably be changed at any time, by the agent providing the solvable.

For example, the solvables of a simple email agent might include:

```
    solvable(send_message(email, +ToPerson, +Params),
           [type(procedure), callback(send_mail)],
              [])
    solvable(last_message(email, -MessageId),
              [type(data), single_value(true)],
              [write(true)]),
           solvable(get_message(email, +MessageId, -
Msg),
               [type(procedure), callback(get_mail)],
          [])
```

The symbols `+' and `-', indicating input and output arguments, are at present used only for purposes of documentation. Most parameters and permissions have default values, and specifications of default values may be omitted from the parameters and permissions lists.

Defining an agent's capabilities in terms of solvable declarations effectively creates a vocabulary with which other agents can communicate with the new agent. Ensuring that agents will speak the same language and share a common, unambiguous semantics of the vocabulary involves *ontology*. Agent development tools and services (automatic translations of solvables by the facilitator) help address this issue; additionally, a preferred embodiment of the present invention will typically rely on vocabulary from either formally engineered ontologies for specific domains or from ontologies constructed during the incremental development of a body of agents for several applications or from both specific domain ontologies and incrementally developed ontologies. Several example tools and services are described in Cheyer et al.'s paper entitled "Development Tools for the Open Agent Architecture," as presented at the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM 96), London, April 1996.

Although the present invention imposes no hard restrictions on the form of solvable declarations, two common usage conventions illustrate some of the utility associated with solvables.

Classes of services are often preferably tagged by a particular type. For instance, in the example above, the "last_message" and "get_message" solvables are specialized for email, not by modifying the *names* of the services, but rather by the use of the `email' parameter, which serves during the execution of an *ICL* request to select (or not) a specific type of message.

Actions are generally written using an imperative verb as the functor of the solvable in a preferred embodiment of the present invention, the direct object (or item class) as the first argument of the predicate, required arguments following, and then an extensible parameter list as the last argument. The parameter list can hold optional information usable by the function. The *ICL* expression generated by a natural language parser often makes use of this parameter list to store prepositional phrases and adjectives.

As an illustration of the above two points, "Send mail to Bob about lunch" will be translated into an *ICL* request send_message(email, `Bob Jones', [subject(lunch)]), whereas "Remind Bob about lunch" would leave the transport unspecified (send_message(KIND, 'Bob Jones', [subject(lunch)])), enabling all available message transfer agents (e.g., fax, phone, mail, pager) to compete for the opportunity to carry out the request.

Requesting Services

An agent preferably requests services of the community of agent by delegating tasks or goals to its facilitator. Each request preferably contains calls to one or more agent solvables, and optionally specifies parameters containing advice to help the facilitator determine how to execute the task. Calling a solvable preferably does *not* require that the agent specify (or even know of) a particular agent or agents to handle the call. While it is possible to specify one or more agents using an address parameter (and there are situations in which this is desirable), in general it is advantageous to leave this delegation to the facilitator. This greatly reduces the hard-coded component dependencies often found in other distributed frameworks. The agent libraries of a preferred embodiment of the present invention provide an agent with a single, unified point of entry for requesting services of other agents: the library procedure *oaa_Solve*. In the style of logic programming, *oaa_Solve* may preferably be used both to retrieve data and to initiate actions, so that calling a *data* solvable looks the same as calling a *procedure* solvable.

Complex Goal Expressions

A powerful feature provided by preferred embodiments of the present invention is the ability of a client agent (or a user) to submit compound goals of an arbitrarily complex nature to a facilitator. A compound goal is a single goal expression that specifies multiple sub-goals to be performed. In speaking of a *"complex goal expression"* we mean that a single goal expression that expresses multiple sub-goals can potentially include more than one type of logical connector (e.g., AND, OR, NOT), and/or more than one level of logical nesting (e.g., use of parentheses), or the substantive equivalent. By way of further clarification, we note that when speaking of an *"arbitrarily complex goal expression"* we mean that goals are expressed in a language or syntax that allows expression of such complex goals when appropriate or when desired, not that every goal is itself necessarily complex.

It is contemplated that this ability is provided through an interagent communication language having the necessary syntax and semantics. In one example, the goals may take the form of compound goal expressions composed using operators similar to those employed by PROLOG, that is, the comma for conjunction, the semicolon for disjunction, the arrow for conditional execution, etc. The present invention also contemplates significant extensions to PROLOG syntax and semantics. For example, one embodiment incorporates a "parallel disjunction" operator indicating that the disjuncts are to be executed by different agents concurrently. A further embodiment supports the specification of whether a given sub-goal is to be executed breadth-first or depth-first.

A further embodiment supports each sub-goal of a compound goal optionally having an address and/or a set of parameters attached to it. Thus, each sub-goal takes the form

Address:Goal::Parameters

where both *Address* and *Parameters* are optional.

An address, if present, preferably specifies one or more agents to handle the given goal, and may employ several different types of referring expression: unique names, symbolic names, and shorthand names. Every agent has preferably a unique name, assigned by its facilitator, which relies upon network addressing schemes to ensure its global uniqueness. Preferably, agents also have self-selected symbolic names (for example, "mail"), which are not guaranteed to be unique. When an address includes a symbolic name, the facilitator preferably takes this to mean that all agents having that name should be called upon. Shorthand names include `self' and `parent' (which refers to the agent's facilitator). The address associated with a goal or sub-goal is preferably always optional. When an address is not present, it is the facilitator's job to supply an appropriate address.

The distributed execution of compound goals becomes particularly powerful when used in conjunction with natural language or speech-enabled interfaces, as the query itself may specify how functionality from distinct agents will be combined. As a simple example, the spoken utterance "Fax it to Bill Smith's manager." can be translated into the following compound *ICL* request:

oaa_Solve((manager('Bill Smith', M), fax(it,M,[])), [strategy(action)])

Note that in this ICL request there are two sub-goals, "manager('Bill Smith',M)" and "fax(it,M,[])," and a single global parameter "strategy(action)." According to the present invention, the facilitator is capable of mapping global parameters in order to apply the constraints or advice across the separate sub-goals in a meaningful way. In this instance, the global parameter strategy(action) implies a parallel constraint upon the first sub-goal; i.e., when there are multiple agents that can respond to the manager sub-goal, each agent should receive a request for service. In contrast, for the second sub-goal, parallelism should not be inferred from the global parameter strategy(action) because such an inference would possibly result in the transmission of duplicate facsimiles.

Refining Service Requests

In a preferred embodiment of the present invention, parameters associated with a goal (or sub-goal) can draw on useful features to refine the request's meaning. For example, it is frequently preferred to be able to specify whether or not solutions are to be returned synchronously; this is done using the *reply* parameter, which can take any of the values *synchronous, asynchronous*, or *none*. As another example, when the goal is a non-compound query of a data solvable, the *cache* parameter may preferably be used to request local caching of the facts associated with that solvable. Many of the remaining parameters fall into two categories: feedback and advice.

Feedback parameters allow a service requester to receive information from the facilitator about how a goal was handled. This feedback can include such things as the identities of the agents involved in satisfying the goal, and the amount of time expended in the satisfaction of the goal.

Advice parameters preferably give constraints or guidance to the facilitator in completing and interpreting the goal. For example, a *solution_limit* parameter preferably allows the requester to say how many solutions it is interested in; the facilitator and/or service providers are free to use this information in optimizing their efforts. Similarly, a *time_limit* is preferably used to say how long the requester is willing to wait for solutions to its request, and, in a multiple facilitator system, a *level_limit* may preferably be used to say how remote the facilitators may be that are consulted in the search for solutions. A *priority* parameter is preferably used to indicate that a request is more urgent than previous requests that have not yet been satisfied. Other preferred advice parameters include but are not limited to parameters used to tell the facilitator whether parallel satisfaction of the parts of a goal is appropriate, how to combine and filter results arriving from multiple solver agents, and whether the requester itself may be considered a candidate solver of the sub-goals of a request.

Advice parameters preferably provide an extensible set of low-level, orthogonal parameters capable of combining with the *ICL* goal language to fully express how information should flow among participants. In certain preferred embodiments of the present invention, multiple parameters can be grouped together and given a group name. The resulting *high-level advice parameters* can preferably be used to express concepts analogous to KQML's performatives, as well as define classifications of problem types. For instance, KQML's "ask_all" and "ask_one" performatives would be represented as combinations of values given to the parameters *reply*, *parallel_ok*, and *solution_limit*. As an example of a higher-level problem type, the strategy "math_problem" might preferably send the query to all appropriate math solvers in parallel, collect their responses, and signal a conflict if different answers are returned. The strategy "essay_question" might preferably send the request to all appropriate participants, and signal a problem (i.e., cheating) if any of the returned answers are identical.

Facilitation

In a preferred embodiment of the present invention, when a facilitator receives a compound goal, its job is to construct a goal satisfaction plan and oversee its satisfaction in an optimal or near optimal manner that is consistent with the specified advice. The facilitator of the present invention maintains a knowledge base that records the capabilities of a collection of agents, and uses that knowledge to assist requesters and providers of services in making contact.

Figure 7 schematically shows data structures 700 internal to a facilitator in accordance with one embodiment of the present invention. Consider the function of a Agent Registry 702 in the present invention. Each registered agent may be seen as associated with a collection of fields found within its parent facilitator such as shown in the figure. Each registered agent may optionally possess a Symbolic Name which would be entered into field 704. As mentioned elsewhere, Symbolic Names need not be unique to each instance of an agent. Note that an agent may in certain preferred embodiments of the present invention possess more than one Symbolic Name. Such Symbolic Names would each be found through their associations in the Agent Registry entries. Each agent, when registered, must possess a Unique Address, which is entered into the Unique Address field 706.

With further reference to Figure 7, each registered agent may be optionally associated with one or more capabilities, which have associated Capability Declaration fields 708 in the parent facilitator Agent Registry 702. These capabilities may define not just functionality, but may further provide a utility parameter indicating, in some manner (e.g., speed, accuracy, etc), how effective the agent is at providing the declared capability. Each registered agent may be optionally associated with one or more data components, which have associated Data Declaration fields 710 in the parent facilitator Agent Registry 702. Each registered agent may be optionally associated with one or more triggers, which preferably could be referenced through their associated Trigger Declaration fields 712 in the parent facilitator Agent Registry 702. Each registered agent may be optionally associated with one or more tasks, which preferably could be referenced through their associated Task Declaration fields 714 in the parent facilitator Agent Registry 702. Each registered agent may be optionally associated with one or more Process Characteristics, which preferably could be referenced through their associated Process Characteristics Declaration fields 716 in the parent facilitator Agent Registry 702. Note that these characteristics in certain preferred embodiments of the present invention may include one or more of the following: Machine Type (specifying what type of computer may run the agent), Language (both computer and human interface).

A facilitator agent in certain preferred embodiments of the present invention further includes a Global Persistent Database 720. The database 720 is composed of data elements which do not rely upon the invocation or instantiation of client agents for those data elements to persist. Examples of data elements which might be present in such a database include but are not limited to the network address of the facilitator agent's server, facilitator agent's server accessible network port list, firewalls, user lists, and security options regarding the access of server resources accessible to the facilitator agent.

A simplified walk through of operations involved in creating a client agent, a client agent initiating a service request, a client agent responding to a service request and a facilitator agent responding to a service request are including hereafter by way of illustrating the use of such a system. These figures and their accompanying discussion are provided by way of illustration of one preferred embodiment of the present invention and are not intended to limit the scope of the present invention.

Figure 8 depicts operations involved in instantiating a client agent with its parent facilitator in accordance with a preferred embodiment of the present invention. The operations begin with starting the Agent Registration in a step 800. In a next step 802, the Installer, such as a client or facilitator agent, invokes a new client agent. It will be appreciated that any computer entity is capable of invoking a new agent. The system then instantiates the new client agent in a step 804. This operation may involve resource allocations somewhere in the network on a local computer system for the client agent, which will often include memory as well as placement of references to the newly instantiated client agent in internal system lists of agents within that local computing system. Once instantiated, the new client and its parent facilitator establish a communications link in a step 806. In certain preferred embodiments, this communications link involves selection of one or more physical transport mechanisms for this communication. Once established, the client agent transmits it profile to the parent facilitator in a step 808. When received, the parent facilitator registers the client agent in a step 810. Then, at a step 812, a client agent has been instantiated in accordance with one preferred embodiment of the present invention.

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 preferred embodiment of the present invention. The method of Figure 9 begins in a step 900, wherein any initialization or other such procedures may be performed. Then, in a step 902, the client agent determines a goal to be achieved (or solved). This goal is then translated in a step 904 into *ICL*, if it is not already formulated in it. The goal, now stated in *ICL*, is then transmitted to the client agent's parent facilitator