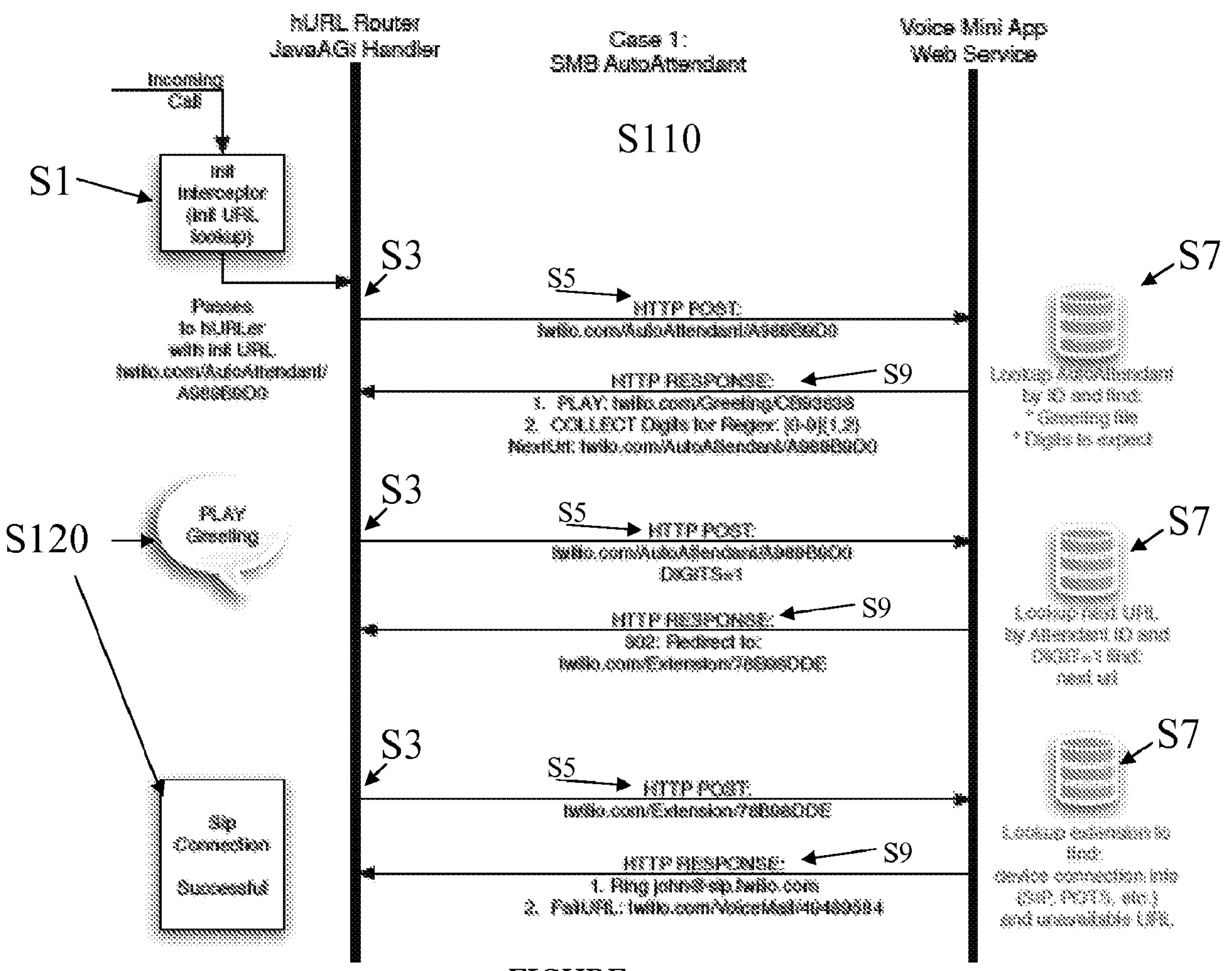
hURL Router
JavaAGI Handler
Case 1:
SMB AutoAttendant
Voice Mini App
Web Service
S110
Incoming
Call
S1
Init
Interceptor
(init URL
lookup)
Passes
to hURLer
with init URL
twilio.com/AutoAttendant/
S3
S5
HTTP POST:
S7
HTTP RESPONSE:
S9
1. PLAY: twilio.com/Greeting/
2. COLLECT Digits for Regex: [0-9]{1,2}
NextUrl: twilio.com/AutoAttendant/
Lookup AutoAttendant
by ID and find:
* Greeting file
* Digits to expect
S120
PLAY
Greeting
S3
S5
HTTP POST:
twilio.com/AutoAttendant/
DIGITS=1
S7
HTTP RESPONSE:
S9
302: Redirect to:
twilio.com/Extension/
Lookup next URL
by Attendant ID and
DIGITS=1 find:
next url
S3
S5
HTTP POST:
twilio.com/Extension/
S7
HTTP RESPONSE:
S9
1. Ring john@sip.twilio.com
2. FailURL: twilio.com/VoiceMail/
Sip
Connection
Successful
Lookup extension to
find:
device connection info
(SIP, POTS, etc.)
and unavailable URL

FIGURE 7

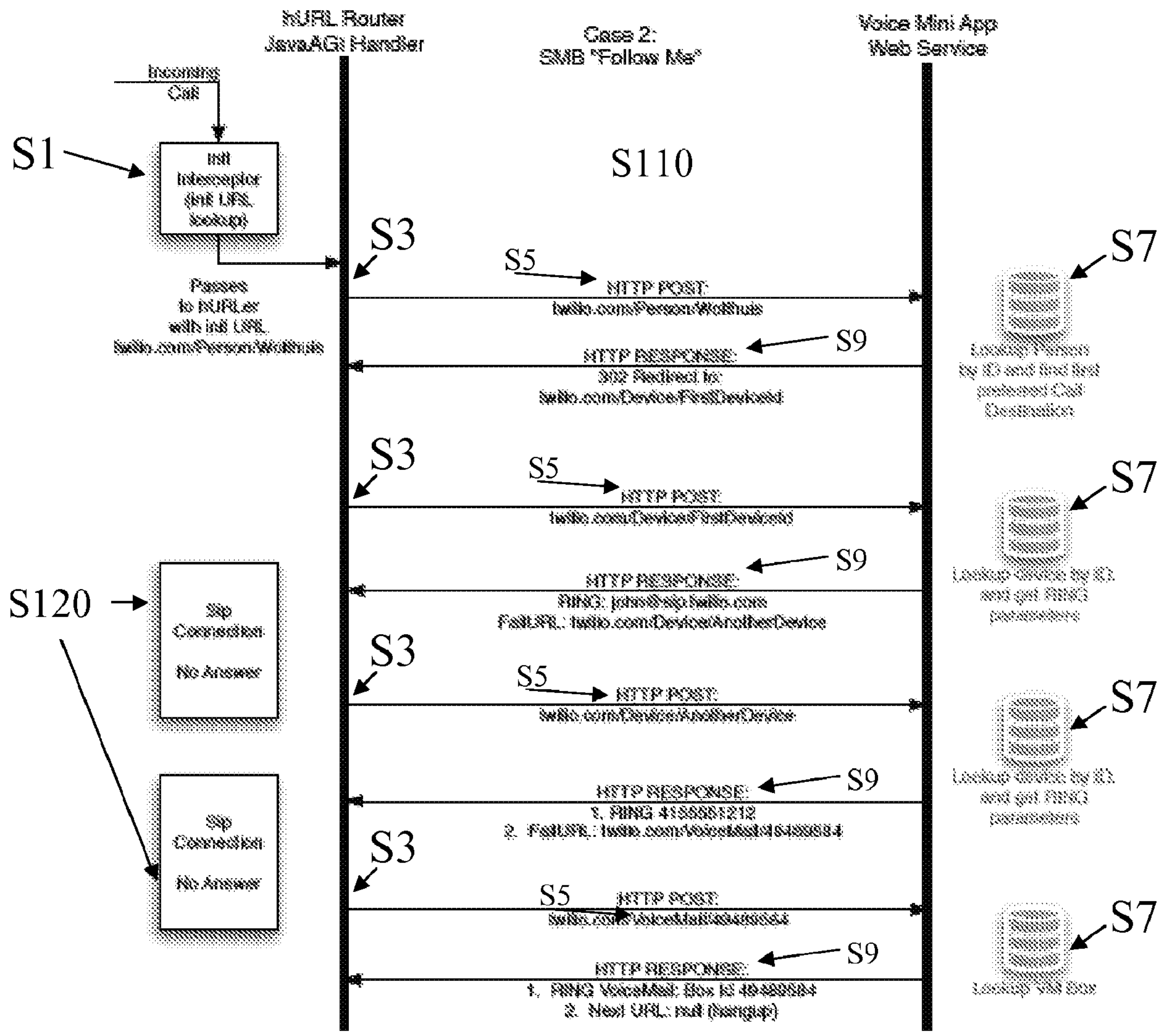
hURL Router
JavaAGI Handler
Case 2:
SMB "Follow Me"
Voice Mini App
Web Service
Incoming
Call
S1
Init
Interceptor
(init URL
lookup)
Passes
to hURLer
with init URL
twilio.com/Person/WoBhuis
S110
S3
S5
HTTP POST:
twilio.com/Person/WoBhuis
S7
Lookup Person
by ID and find first
preferred Call
Destination
S9
HTTP RESPONSE:
302 Redirect to:
twilio.com/Device/FirstDeviceId
S3
S5
HTTP POST:
twilio.com/Device/FirstDeviceId
S7
Lookup device by ID
and get RING
parameters
S9
HTTP RESPONSE:
RING: john@sip.twilio.com
FailURL: twilio.com/Device/AnotherDevice
S120
Sip
Connection
No Answer
S3
S5
HTTP POST:
twilio.com/Device/AnotherDevice
S7
Lookup device by ID
and get RING
parameters
S9
HTTP RESPONSE:
1. RING 4155551212
2. FailURL: twilio.com/VoiceMail/4548684
Sip
Connection
No Answer
S3
S5
HTTP POST:
twilio.com/VoiceMail/4548684
S7
Lookup VM Box
S9
HTTP RESPONSE:
1. RING VoiceMail: Box Id 4548684
2. Next URL: null (hangup)

FIGURE 8

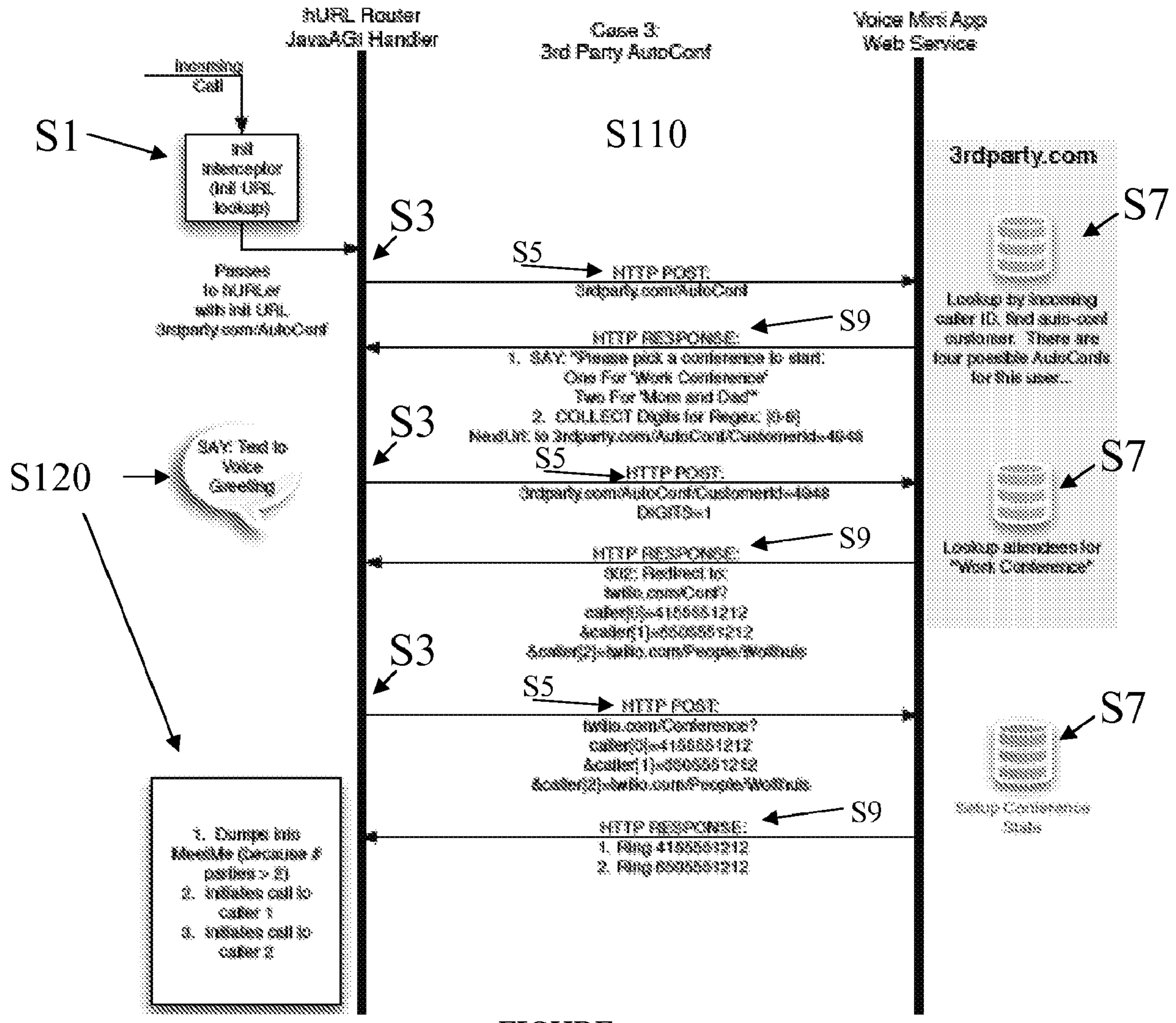
hURL Router
JavaAGI Handler
Case 3:
3rd Party AutoConf
Voice Mini App
Web Service
S110
Incoming
Call
S1
init
interceptor
(init URL
lookup)
Passes
to hURLer
with init URL
3rdparty.com/AutoConf
S3
S5
HTTP POST:
3rdparty.com/AutoConf
3rdparty.com
S7
Lookup by incoming
caller ID, find auto-conf
customer. There are
four possible AutoConfs
for this user...
S9
HTTP RESPONSE:
1. SAY: "Please pick a conference to start:
One For "Work Conference"
Two For "Mom and Dad"
2. COLLECT Digits for Regex: [0-4]
NextUrl: to 3rdparty.com/AutoConf/CustomerId=4848
S120
SAY: Text to
Voice
Greeting
HTTP POST:
3rdparty.com/AutoConf/CustomerId=4848
DIGITS=1
Lookup attendees for
"Work Conference"
HTTP RESPONSE:
302: Redirect to:
twilio.com/Conf?
caller[0]=4155551212
&caller[1]=6505551212
&caller[2]=twilio.com/People/WolfHale
HTTP POST:
twilio.com/Conference?
caller[0]=4155551212
&caller[1]=6505551212
&caller[2]=twilio.com/People/WolfHale
Setup Conference
State
HTTP RESPONSE:
1. Ring 4155551212
2. Ring 6505551212
1. Dumps into
MeetMe (because #
parties > 2)
2. initiates call to
caller 1
3. initiates call to
caller 2

FIGURE 9

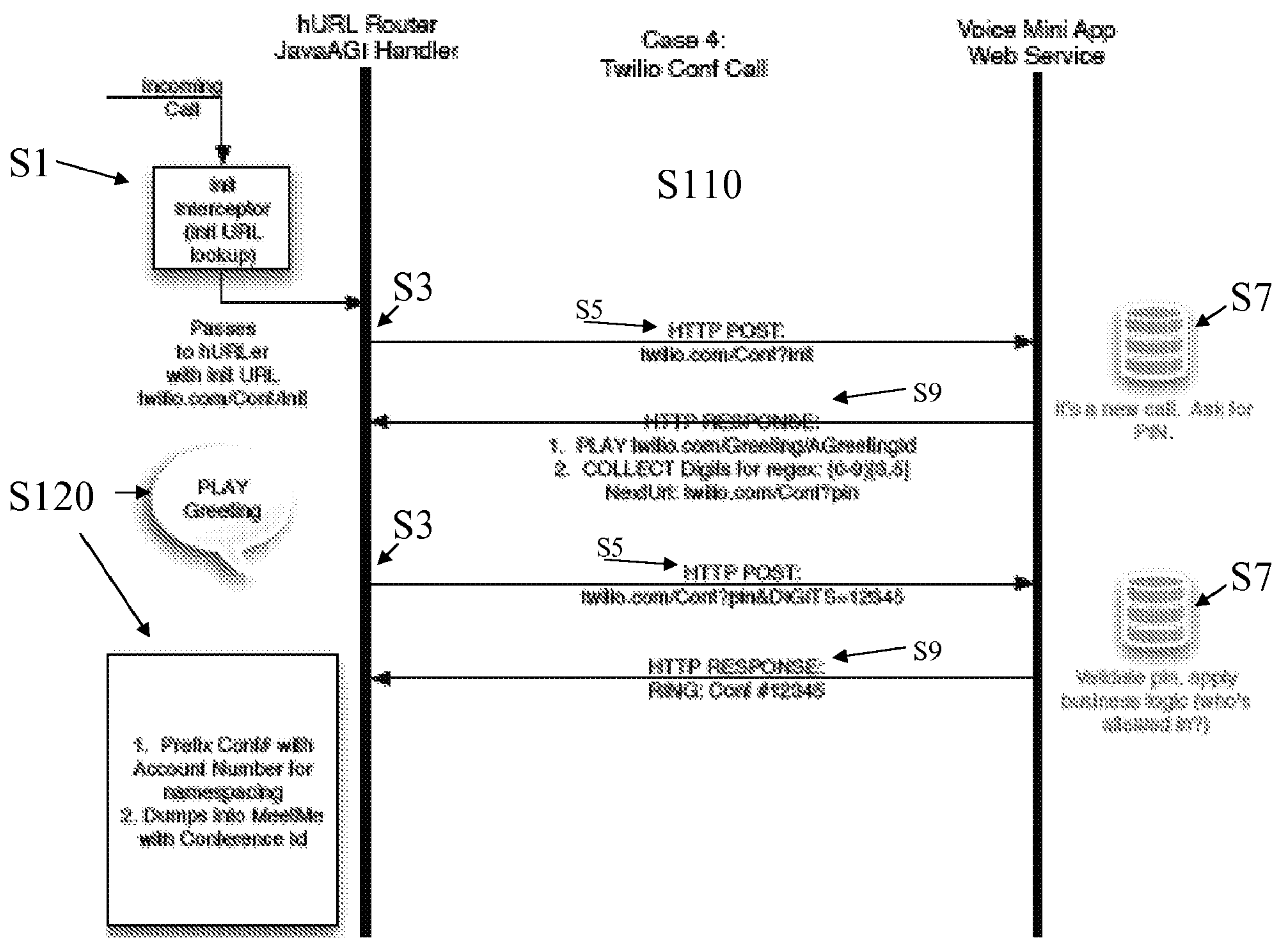
hURL Router
JavaAGI Handler
Case 4:
Twilio Conf Call
Voice Mini App
Web Service
Incoming
Call
S1
Init
Interceptor
(init URL
lookup)
S110
S3
Passes
to hURLer
with init URL
twilio.com/Conf/init
S5
HTTP POST:
twilio.com/Conf?init
S7
It's a new call. Ask for
PIN.
S9
HTTP RESPONSE:
1. PLAY twilio.com/Greetings/AGreetingId
2. COLLECT Digits for regex: [0-9]{3,5}
NextUrl: twilio.com/Conf?pin
S120
PLAY
Greeting
S3
S5
HTTP POST:
twilio.com/Conf?pin&DIGITS=12345
S7
S9
HTTP RESPONSE:
RING: Conf #12345
Validate pin, apply
business logic (who's
allowed in?)
1. Prefix Conf# with
Account Number for
namespacing
2. Dumps into MeetMe
with Conference Id

FIGURE 10

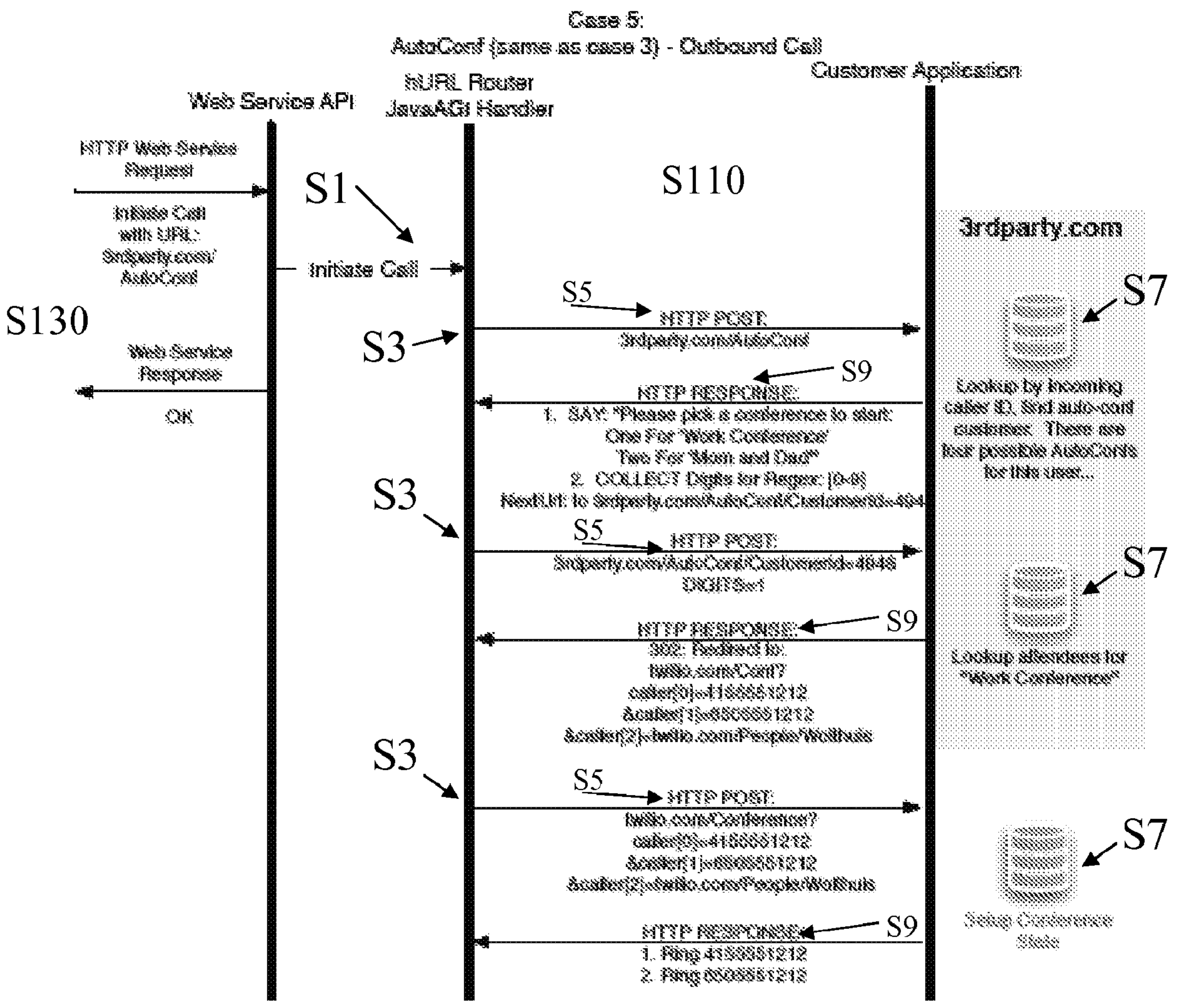
Case 5:
AutoConf (same as case 3) - Outbound Call
Web Service API
hURL Router
JavaAGI Handler
Customer Application
S110
S1
S130
S3
S5
S7
S9
HTTP Web Service Request
Initiate Call with URL: 3rdparty.com/AutoConf
Initiate Call
Web Service Response
OK
3rdparty.com
HTTP POST: 3rdparty.com/AutoConf
HTTP RESPONSE:
1. SAY: "Please pick a conference to start: One For 'Work Conference' Two For 'Mom and Dad'"
2. COLLECT Digits for Regex: [0-9]
NextUrl: to 3rdparty.com/AutoConf/Customerid=4848
Lookup by incoming caller ID, find auto-conf customer. There are four possible AutoConfs for this user...
HTTP POST: 3rdparty.com/AutoConf/Customerid=4848 DIGITS=1
HTTP RESPONSE: 302: Redirect to: twilio.com/Conf? caller[0]=4155551212 &caller[1]=6505551212 &caller[2]=twilio.com/People/WolfNuts
Lookup attendees for "Work Conference"
HTTP POST: twilio.com/Conference? caller[0]=4155551212 &caller[1]=6505551212 &caller[2]=twilio.com/People/WolfNuts
HTTP RESPONSE:
1. Ring 4155551212
2. Ring 6505551212
Setup Conference State

FIGURE 11

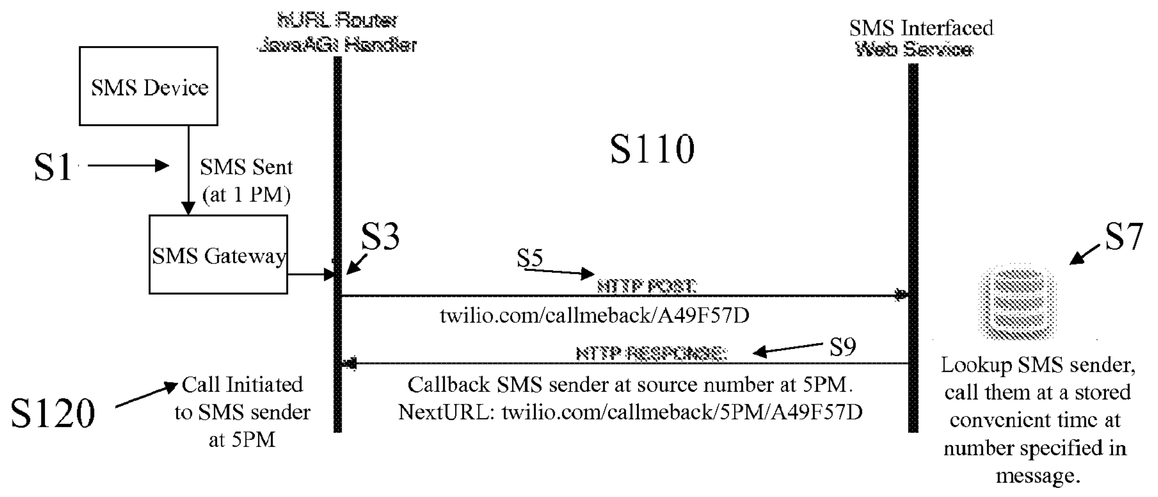
hURL Router
JavaAGI Handler
SMS Interfaced
Web Service
SMS Device
S1
SMS Sent
(at 1 PM)
SMS Gateway
S110
S3
S5
HTTP POST
twilio.com/callmeback/A49F57D
S7
HTTP RESPONSE
S9
Lookup SMS sender, call them at a stored convenient time at number specified in message.
S120
Call Initiated to SMS sender at 5PM
Callback SMS sender at source number at 5PM.
NextURL: twilio.com/callmeback/5PM/A49F57D

FIGURE 12

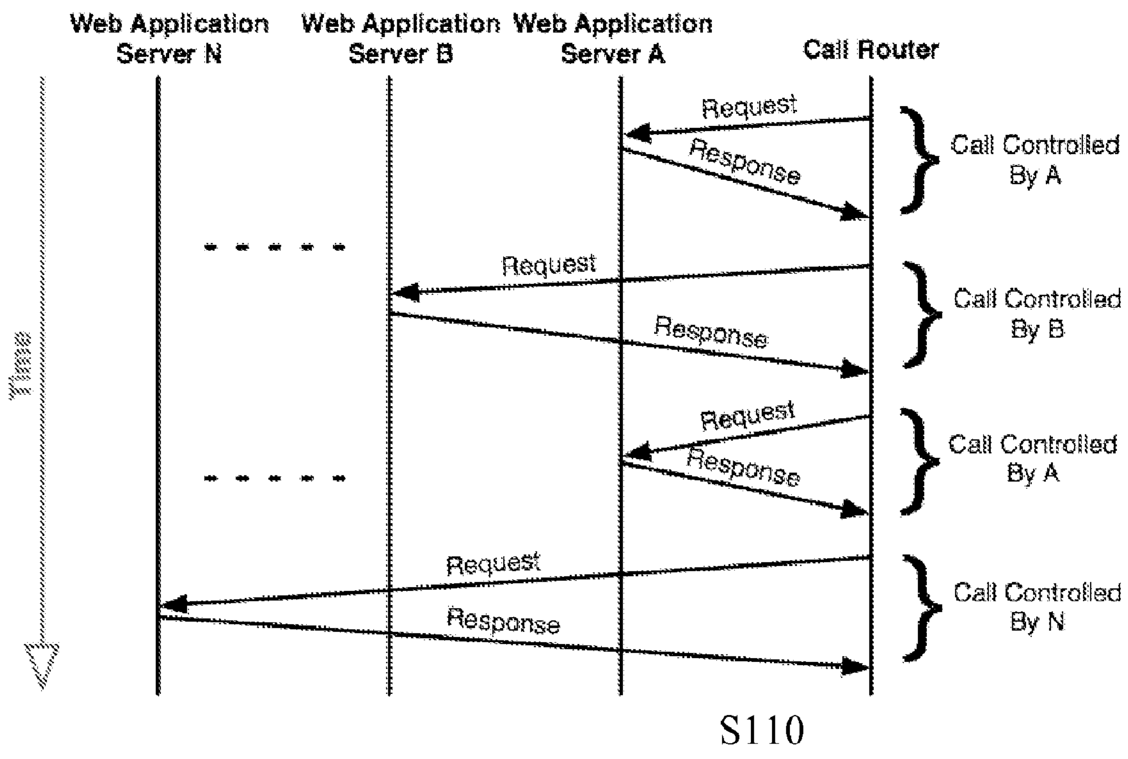

FIGURE 13

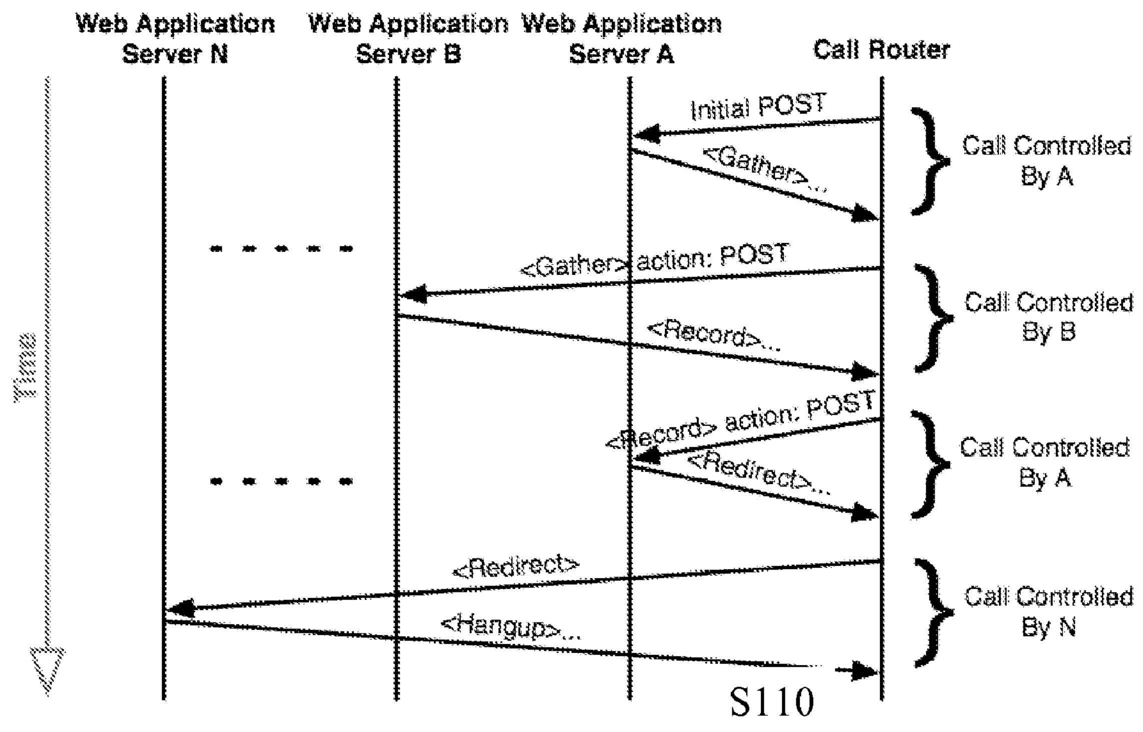
Web Application Server N
Web Application Server B
Web Application Server A
Call Router
Time
Initial POST
<Gather>...
Call Controlled By A
<Gather> action: POST
<Record>...
Call Controlled By B
<Record> action: POST
<Redirect>...
Call Controlled By A
<Redirect>
<Hangup>...
Call Controlled By N
S110

FIGURE 14

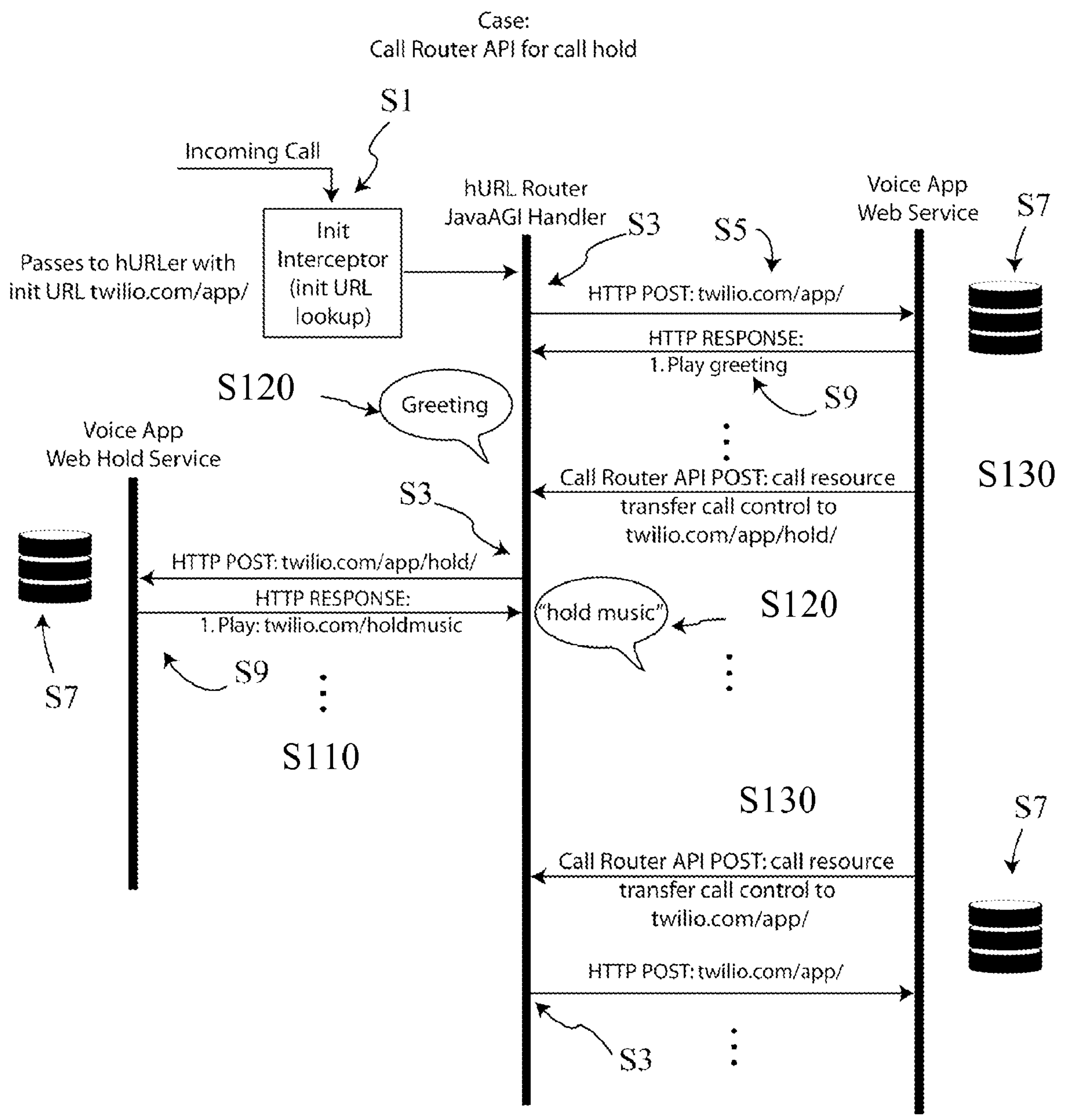
Case:
Call Router API for call hold
S1
Incoming Call
Init Interceptor (init URL lookup)
Passes to hURLer with init URL twilio.com/app/
hURL Router JavaAGI Handler
S3
S5
Voice App Web Service
S7
HTTP POST: twilio.com/app/
HTTP RESPONSE: 1. Play greeting
S9
S120
Greeting
Voice App Web Hold Service
Call Router API POST: call resource transfer call control to twilio.com/app/hold/
S130
S3
HTTP POST: twilio.com/app/hold/
HTTP RESPONSE: 1. Play: twilio.com/holdmusic
"hold music"
S120
S7
S9
S110
S130
S7
Call Router API POST: call resource transfer call control to twilio.com/app/
HTTP POST: twilio.com/app/
S3

FIGURE 15

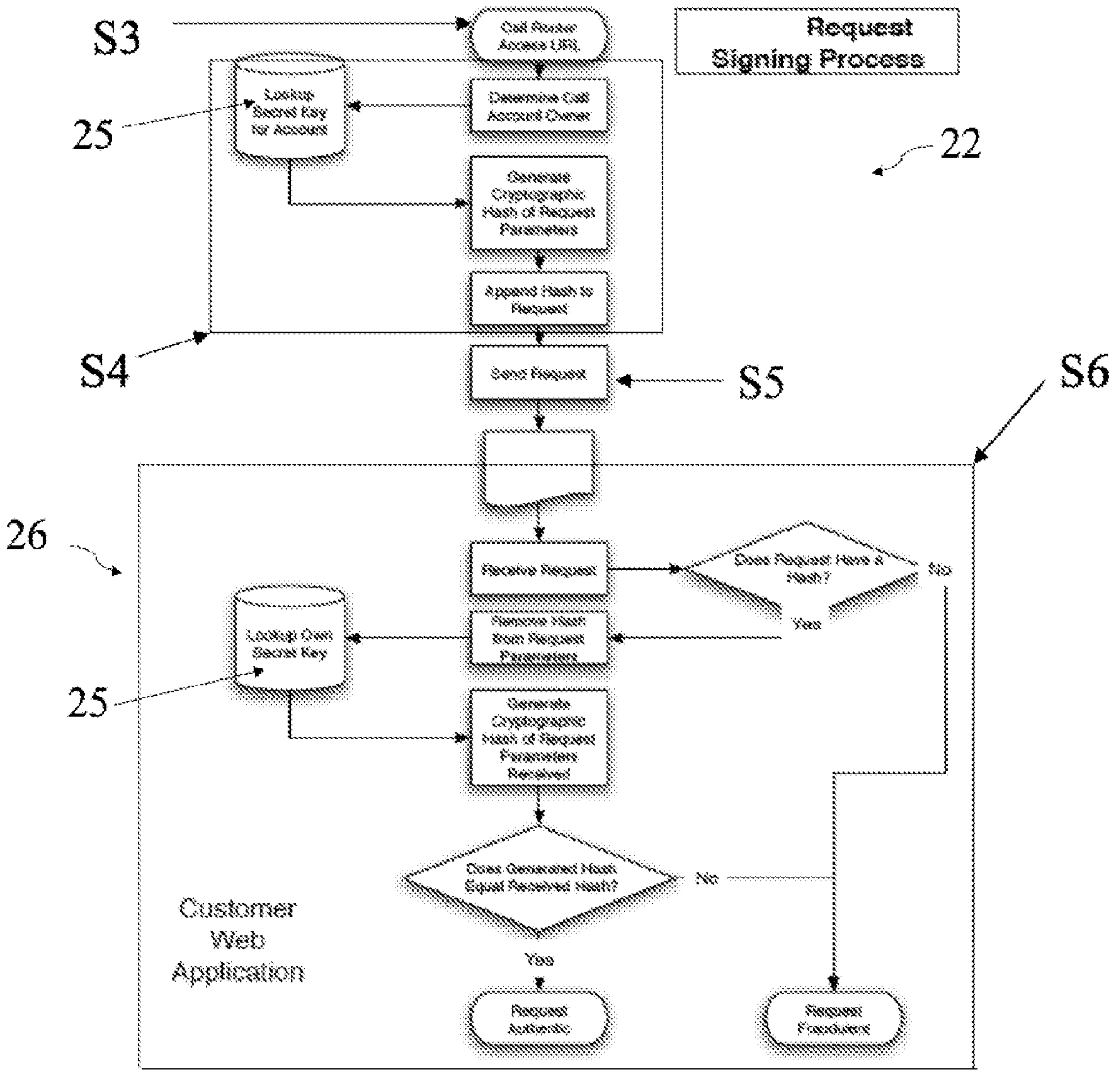
S3
Call Router Access URL
Request Signing Process
25
Lookup Secret Key for Account
Determine Call Account Owner
22
Generate Cryptographic Hash of Request Parameters
Append Hash to Request
S4
Send Request
S5
S6
26
Receive Request
Does Request Have a Hash?
No
Yes
Lookup Own Secret Key
Remove Hash from Request Parameters
25
Generate Cryptographic Hash of Request Parameters Received
Does Generated Hash Equal Received Hash?
No
Customer Web Application
Yes
Request Authentic
Request Fraudulent

FIGURE 16

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2009/039371

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - H04L 12/66 (2009.01)
USPC - 370/352
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(8) - H04L 12/66 (2009.01)
USPC - 370/352

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 practicable, search terms used)
MicroPatent

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2007/0036143 A1 (ALT et al) 15 February 2007 (15.02.2007) entire document	1-14, 16-19, 21-25,27
Y		15, 20, 26
Y	US 2007/0242626 A1 (ALTBERG et al) 18 October 2007 (18.10.2007) entire document	15, 20, 26

☐ Further documents are listed in the continuation of Box C. ☐

* Special categories of cited documents:	"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
"A" document defining the general state of the art which is not considered to be of particular relevance	"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
"E" earlier application or patent but published on or after the international filing date	"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
"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)	"&" document member of the same patent family
"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	

Date of the actual completion of the international search 26 June 2009	Date of mailing of the international search report 14 JUL 2009
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Authorized officer: Blaine R. Copenheaver PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (April 2005)

WO2010037064

Publication Title:

SYSTEM AND METHOD FOR PROCESSING MEDIA REQUESTS DURING A TELEPHONY SESSIONS

Abstract:

Abstract of WO 2010037064

(A1) Translate this text In a preferred embodiment, the method of caching media used in a telephony application includes: receiving a media request; sending the media request to a media layer using HTTP; the a media layer performing the steps of checking in a cache for the media resource; processing the media request within a media processing server; and storing the processed media in the cache as a telephony compatible resource specified by a persistent address. The system of the preferred embodiment includes a call router and a media layer composed of a cache and media processing server.

Courtesy of http://v3.espacenet.com

This Patent PDF Generated by Patent Fetcher(R), a service of Stroke of Color, Inc.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
1 April 2010 (01.04.2010)

PCT

(10) International Publication Number
WO 2010/037064 A1

(51) International Patent Classification:
H04M 7/00 (2006.01)

(21) International Application Number:
PCT/US2009/058671

(22) International Filing Date:
28 September 2009 (28.09.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

61/100,578	26 September 2008 (26.09.2008)	US
61/100,630	26 September 2008 (26.09.2008)	US
61/100,627	26 September 2008 (26.09.2008)	US
61/156,751	2 March 2009 (02.03.2009)	US
61/156,746	2 March 2009 (02.03.2009)	US
12/417,630	2 April 2009 (02.04.2009)	US

(71) Applicant *(for all designated States except US)*: **TWILIO INC.** [US/US]; 548 Market St #14510, San Francisco, CA 94104 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **LAWSON, Jeffrey** [US/US]; 548 Market St #14510, San Francisco, CA 94104 (US). **WOLTHUIS, John** [US/US]; 548 Market St #14510, San Francisco, CA 94104 (US). **COOKE, Evan** [US/US]; 548 Market St #14510, San Francisco, CA 94104 (US).

(74) Agent: **SCHOX, Jeffrey**; 500 3rd Street #515, San Francisco, CA 94107 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

[Continued on next page]

(54) Title: SYSTEM AND METHOD FOR PROCESSING MEDIA REQUESTS DURING A TELEPHONY SESSIONS

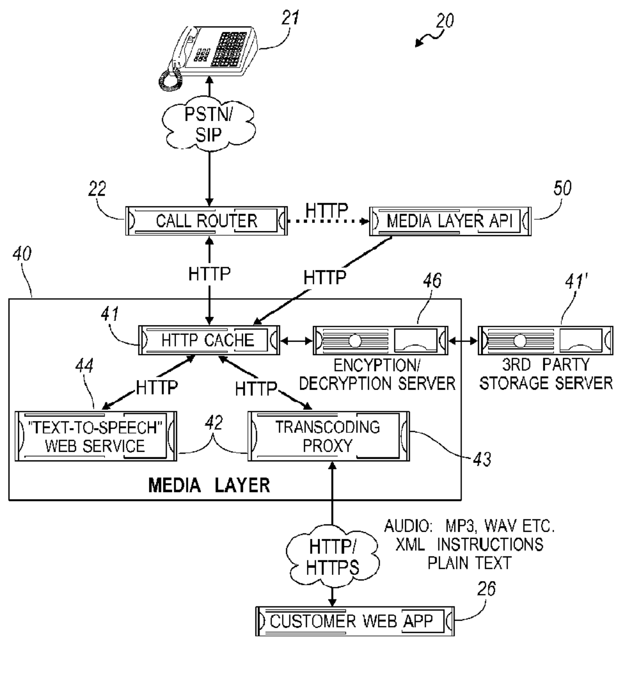

FIG. 11

(57) **Abstract**: In a preferred embodiment, the method of caching media used in a telephony application includes: receiving a media request; sending the media request to a media layer using HTTP; the a media layer performing the steps of checking in a cache for the media resource; processing the media request within a media processing server; and storing the processed media in the cache as a telephony compatible resource specified by a persistent address. The system of the preferred embodiment includes a call router and a media layer composed of a cache and media processing server.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

SYSTEM AND METHOD FOR PROCESSING MEDIA REQUESTS DURING A TELEPHONY SESSIONS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is a continuation in part of prior application number 12/417,630 filed 02 April 2009 and entitled "System and Method for Processing Telephony Sessions", which claims the benefit of the following: US Provisional Application number 61/041,829 filed 02 April 2008 and entitled "System and Method for Processing Telephony Sessions"; US Provisional Application number 61/055,417 filed on 22 May 2008 and entitled "System and Method for Processing SMS Messages", US Provisional Application number 61/100,578 filed on 26 September 2008 and entitled "System and Method for Processing Telephony Sessions", US Provisional Application number 61/156,746 filed on 02 March 2009 and entitled "System and Method for Processing Telephone Sessions", and US Provisional Application number 61/156,751 filed on 02 March 2009 and entitled "System and Method for Processing Telephony Sessions".

[0002] This application also claims the benefit of the following: US Provisional Application number 61/100,627 filed on 26 September 2008 and entitled "System and Method of Caching Media for Telephony Sessions" and US Provisional Application number 61/100,630 filed on 26 September 2008 and entitled "System and Method of Caching Media for Telephony Sessions."

[0003] All of priority documents identified in this section are incorporated in their entirety by this reference.

TECHNICAL FIELD

[0004] This invention relates generally to the telephony field, and more specifically to a new and useful system and method for processing media requests during telephony sessions in the telephony field.

BACKGROUND

[0005] In the last decade, legislation and the advent of Voice over Internet Protocol (VOIP) have revolutionized the communication industry with new technologies, business models, and service providers. Software and commodity hardware now provide an alternative to expensive carrier equipment. One can implement extensible call switching and voice application logic in Open source software applications, such as Asterisk and FreeSwitch. These new application stacks, however, usher in new complexities and challenges, requiring new skill sets to deploy, develop, and maintain. Deploying telephony services requires knowledge of voice networking and codecs, hardware or services to bridge servers to the public phone infrastructure, capital investment in hardware, and ongoing collocation of that hardware. These burdens are a mere prerequisite to developing the actual application, which requires developers to train in new languages, tools, and development environments. Even telephony applications that currently try to leverage a model more similar to web-development such as Voice Extensible Markup Language (VoiceXML), require the dedication to learn a new language and understand telephony interaction. Ongoing operation and maintenance of these services requires teams to adopt new analysis tools, performance metrics, and debugging methodologies. Developing even the simplest of voice services (such as a so-called “phone tree”) requires significant upfront and ongoing investment in specialized infrastructure, skills, and operations.

[0006] In similar manner to how multimedia has impacted the advance of the Internet, interacting with media through telephony services is also becoming more important for telephony applications. However, media consumption through an internet browser and a telephony device are completely different experiences, each having different user expectations. Unlike websites, where users have been conditioned for loading times and processing time, phone users expect real-time results and often view processing delays as application annoyances. Internet media is inherently multimedia: a combination of text, images, video, audio, and other forms of multimedia. Telephony devices are limited in the format of media consumable by a user. In the case of a typical phone, audio with 8-bit PCM mono with 8 kHz bandwidth format is the native form. Tremendous amounts of processing must be performed by telephony applications to convert from internet media to telephony compatible media. The processing increases infrastructure costs, slows down the responsiveness of a telephony application, and overall, limits the possibilities of telephony applications. The inefficiency of media processing impacts not only one telephony application but all applications operating on a system. Thus, there is a need in the telephony field to create a new and useful system and method for processing media requests during telephony sessions. This invention provides such a new and useful system and method.

SUMMARY

[0007] The method of the preferred embodiment for processing telephony sessions include the steps of communicating with an application server using an application layer protocol, processing telephony instructions with a call router, and creating call router resources accessible through an Application Programming Interface (API). The method and system of the preferred embodiments enables web developers to use their existing skills and tools with the esoteric world of telephony, making telephony application development as easy as web programming. The method and system use the familiar web site visitor model to interact with a web developer's application, with each step of the phone call analogous to a traditional page view. Within this model, developers reuse their existing tools and techniques, including familiar concepts such as HTTP redirects, accessing resources through an API, cookies, and mime-type responses to construct complex telephony applications. The method of processing telephony instructions and creating call router resources accessible through an API (a call router API) cooperatively function to enable a stateless and simple telephony language with more call router resources and information provided through the call router (preferably a REST API as is familiar to many web developers). In one embodiment, the telephony instructions set may have fewer than dozen verbs, simplifying the language so that developers can quickly learn and implement telephony applications, while the call router API compliments the simple telephony instructions to enable complex telephony applications.

[0008] Within this framework for processing a telephony session, a method and system is described for caching media of the telephony session. The method and system include a cache and a media layer that cooperatively works to minimize processing and create telephony compatible media files that are cacheable. This method and system further enhances the developer process by removing the complexities of telephone media formatting and creates an improved telephony application system.

BRIEF DESCRIPTION OF THE FIGURES

[0009] FIGURE 1 is a flowchart representation of a preferred method of the invention.

[0010] FIGURES 2A, 2B, 3A and 3B are schematic representations of preferred embodiments of the invention.

[0011] FIGURES 4A – 4C are examples of a HTTP GET request, a HTTP POST request, and a HTTP GET request, respectively.

[0012] FIGURES 4D – 4F are examples of a HTTP requests.

[0013] FIGURES 5A and 5B are examples of XML responses.

[0014] FIGURE 6 is an example of a call Router request and response.

[0015] FIGURES 7-9 are schematic representations of various applications that incorporate the principals of the preferred method of the invention.

[0016] FIGURE 10 is a flowchart representation of the sub-steps relating to the digital signature aspect of the preferred method of the invention.

[0017] FIGURE 11 is a schematic diagram of the preferred embodiment of the invention.

[0018] FIGURE 12 is a flowchart diagram of a preferred method.

[0019] FIGURE 13 is a flowchart diagram of a first preferred variation of the preferred method, including a Text-To-Speech audio conversion.

[0020] FIGURE 14 is a flowchart diagram of a second preferred variation including audio transcoding.

[0021] FIGURE 15 is a schematic diagram of the preferred embodiment interfacing with a professional recording server.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0022] The following description of the preferred embodiments of the invention is not intended to limit the invention to these preferred embodiments, but rather to enable any person skilled in the art to make and use this invention.

1. Method for Processing Telephony Sessions

[0023] As shown in FIGURES 1, 2A, 2B, 3A, and 3B, the method 10 of the preferred embodiment for processing telephony sessions include the steps of communicating with an application server using an application layer protocol S110, processing telephony instructions with a call router S120, and creating call router resources accessible through an Application Programming Interface (API) S130. The preferred method may also include other steps and/or sub-steps, as explained below.

1A. Communicating with an Application Server

[0024] As shown in FIGURE 1, the step of communicating with an application server using an application layer protocol S110 preferably includes the following sub-steps: initiating a telephony session S1, mapping a call to a Universal Resource Identifier (URI) S3, sending a request to a server associated with the URI S5, processing the request corresponding to the state of a telephony session S7, and receiving a response from the server S9. One of the challenges of using the familiar web site visitor model is that a third party web application may expose URIs that contain sensitive data or that suggest actions that could maliciously manipulate the application database. In the preferred embodiment, the call router cryptographically signs outbound requests to customer web applications using an account-specific key. More specifically, the step of communicating with the application server includes the additional steps of digitally signing the request parameters S4 and verifying the digital signature of the request parameters S6. Only the call router and the application server know that key, so any request that includes parameters (URL, POST data, headers, etc) signed with that key can be checked for authenticity before allowing such operations. This method also provides verification of authenticity over insecure links (HTTP) with low CPU overhead.

[0025] Step S1, which recites initiating a telephony session, functions to accept an incoming message. The message is preferably a call from a PSTN-connected (Public

Switched Telephone Network) or Internet addressable devices, such as landline phones, cellular phones, satellite phones, Voice-Over-Internet-Protocol (VOIP) phones, SIP devices, Skype, Gtalk, or any other suitable PSTN-connected or Internet addressable voice device. The message may alternatively be a Short Message Service (SMS) message. A SMS gateway server may alternatively connect to a SMS network through a Short Message Service Center ("SMS-C"), directly to the Signaling System #7 (SS7) telephony network, or by any other suitable SMS gateway provider, and the message is preferably received from the gateway by the call router and translated into a format (such as a URI) that can be sent over the public Internet such as HTTP, based on the recipient address of the SMS, such as a short code, or Direct Inward Dialing (DID), or other suitable unique recipient identifier. The message may alternatively be a multimedia message, a facsimile transmission, an email, or any other suitable messaging medium. The originating phone number of the PSTN device is preferably captured using caller ID, but any other suitable ID may be captured, such as a VOIP provider ID, SMS device number, email address, or a short code. The dialed phone number, the EIN, and/or billing identifier, and/or the date and time of the call are also preferably included in the session information. An authentication ID may additionally or alternatively be included in the session information.

[0026] In one variation, Step S1 also functions to initiate a telephony session (such as a phone call) via an HTTP or other request sent to a call router from an application running on a third-party server. In this variation, the application running on the server preferably specifies an initial URI for the call router to use for telephony session in step S3, as well as the phone number (or other addressable destination) to dial and the source phone number (caller id). In this variation, the call router API is preferably used by the application server to request an outgoing call from the call router.

[0027] Step S3, which recites mapping the call to a Universal Resource Identifier (URI), functions to enable a telephony session to be converted into a format that may be handled with standard web servers and web applications. The mapping is preferably performed using a call router. The initial URI is preferably pre-specified at the call router by a web application (which may be running on a third party server) or call router account owner. More preferably, the initial URI is assigned to the call via a unique identifier for the call destination, such as a DID (Direct Inbound Dial) phone number, or a VOIP SIP address. The URI may alternatively be specified by a remote server or other suitable device or method. In one variation, the URI may be used to encapsulate state information or a portion of state information from the initiated telephony session, such as the originating phone number, the dialed phone number, the date and time of the call, geographic location of the caller (e.g. country, city, state, and/or zip), and/or the unique call ID. The information included in the URI may be included in the form of a URI template. For example the URI default template could be: http://demo.twilio.com/myapp/{dialed phone number}/{originating phone number} or http://demo.twilio.com/myapp/foo.php?dialed_number={dialed phone number}& originating_number={originating phone number}.

[0028] Step S4 functions to digitally sign the request parameters. As shown in FIGURE 10, Step S4 preferably determines the call router account owner and, more preferably, looks up the account owner's unique ID or secret key and signs a set of request parameters. Step S4 is preferably accomplished by generating a cryptographic hash of the request parameters, preferably including the URI as well as any request body parameters (in the case of an HTTP POST, for example) with the unique key associated with the call router account owner. The cryptographic hash is preferably generated by appending the hash of the request parameters to the original set of request parameters. The hash is preferably appended to a URL, but if the hash is particularly long (i.e. for a very large number of parameters) the hash may be included in an HTTP header, where there is no limitation on size. In a variation of Step S4, at least one sensitive parameter may be individually encrypted using the account owner's secret key before the hash is processed. In another variation, a cryptographic credential delegation system, such as Oauth (oauth.net), may alternatively be used to electronically sign the request.

[0029] Step S5 functions to send the request to a server. Preferably, the request is sent to a URI and, more preferably, the request is sent to the URI mapped in S3. The request preferably includes a cryptographic hash computed from the set of request parameters (acting as a digital signature), but the request may alternatively include individually encrypted request parameters if the parameters are determined to contain sensitive data. The server is preferably a third party server and, more preferably, the server is running a web application. The request is preferably sent to a server over a network. In one variation, the request is sent to a local server on a local area network. In another variation, the request is sent to a server running locally on the device originating the call. In yet another variation, the request may be sent to multiple servers.

The request preferably encapsulates at least a portion of the state information from the initiated telephony session, such as the originating phone number, the dialed phone number, the date and time of the call, geographic location of the caller (e.g. country, city, and/or state, zip), and/or the unique call ID. The request, more preferably, encapsulates all the state information of the call, but may alternatively include no state information or partial state information. The state information from the initiated telephony session is preferably sent via HTTP POST in the request body, HTTP GET in the request URI, HTTP header parameters to mimic the data flow of a web browser, or by any combination or suitable alternative way. If new state information is generated in the course of the operation of the call router, a request to the application server is preferably made to communicate the new state and to request new telephony instructions. Preferably, new state information is not kept or acted upon internally by the call router, but is passed to the application server for processing. Alternatively, partial state information is preferably stored on the call router until a fully updated state is achieved, and then communicated to the application server. For example, the application server may specify that multiple digits should be pressed on the keypad, not just one, before new call state is derived and communicated to the application server. In one variation, the information from the initiated telephone session may be a web-form submission included in the HTTP POST request. The request may include any state information from the telephony session, such as the originating phone number, the dialed phone number, the date and time of the call, and/or the unique call ID, the current status of the phone call (pending, in-progress, completed, etc.), or the results of a telephony action, including Dual Tone Multi Frequency (DTMF) digit processing, or a representation of or a link to a sound recording, or the status of the last command, or other call state. Examples of a HTTP GET request, a HTTP POST request, and a HTTP GET request are shown in FIGURES 4A, 4B, and 4C, respectively. Further examples of HTTP communication used for SMS messaging are shown in FIGURES 4D, 4E, and 4F. The HTTP request (or any suitable request communication) to the server preferably observes the principles of a RESTful design. RESTful is understood in this document to describe a Representational State Transfer architecture as is known in the art. The RESTful HTTP requests are preferably stateless, thus each message communicated from the call router to the application server preferably contains all necessary information for operation of the application server and response generation of the application server. The call router and/or the application server preferably do not need to remember or store previous communications to be aware of the state. Documents, media, and application state are preferably viewed as addressable resources, combined with data provide to the resource via request parameter, such as HTTP GET or HTTP POST parameters, or request body contents. Such request data may include an updated representation of the call resource, or other call state data generated as a result of call router operation, such as digits pressed on the keypad or audio recordings generated. State information included with each request may include a unique call identifier, call status data such as whether the call is in—progress or completed, the caller ID of the caller, the phone number called, geographic data about the callers, and/or any suitable data. However, a varying level of a RESTful communication (statelessness) may be used, such as by using cookies, session tracking, or any suitable devices to simulate a normal website visitor model. Preferably, data sent with each request may fully enable the application server to determine the next state of the call to execute. RESTfulness preferably does not preclude using external datasource, such as a database, to lookup additional data to log call meta data, or determine application logic.

[0030] Step S6 functions to verify the digital signature of the request parameters. As shown in FIGURE 7, after the request is received at the server, the request parameters are preferably checked and/or parsed for a hash. The cryptographic hash is preferably included in the URL of an HTTP request, but may alternatively be included in the HTTP header of the request. If the request does not include a hash, and the web application server has enabled the hash function checking as a security measure, the request is preferably determined to be fraudulent, which would include – for example – malicious requests, mis-routed requests, corrupted requests and any other requests not intended for the application server. If the set of request parameters includes a hash, the hash is preferably extracted from the request, and the secret key of the customer web application (i.e. the same key that is stored on the call router as the customer account secret key) is preferably used to generate a server side cryptographic hash of the parameters received. The server side cryptographic hash is preferably compared to the hash included with the request and if the hashes do not match, the request is preferably determined to be fraudulent. However, if the server side cryptographic hash matches the request hash, the request is preferably determined to be authentic and ready for further processing at the application server. In the variation mentioned above in Step S4, where sensitive parameters may have been encrypted using the secret key, Step S6 preferably includes decrypting the sensitive parameters. The application server and the third parties operating the application are preferably responsible for completing this verification step, but the verification may alternatively be completed by a single party, such as when a single party operates the application server and the call router. The application server may alternatively be configured to ignore a hash included with the request parameters if request authentication is not important to the application.

[0031] Step S7, which recites processing the request corresponding to the state of a telephony session, functions to perform processing functions on at least a portion of the data included in the request. The processing functions are preferably performed on a third party server. The processing functions may include recording the data included in the request and/or metadata about the call session, routing to another URI, performing a database lookup of at least one portion of the data included in the request, voice recognition processing, or any other suitable processing function. The processing functions may re-use logic and data from other business applications, such as customer databases and/or shopping cart applications, which may be linked using caller-id or caller provided information. State information is preferably communicated with each request from the call router, and application state is preferably not required on the application server. Alternatively, the application server may store state between each request related to the call, by using HTTP cookies, sessions, and/or database records. In some cases, such as the case of a static HTML page running on a server or a stored media file such as an mp3 or wav file stored on a server, Step S7 may be simplified, and a file mapped to disk by the URI may be simply returned. In some situations, media files (such as an mp3 or wav audio file), are requested by the call router and returned by the application server.

[0032] Step S9 recites receiving a response from the server. This response is preferably an HTTP response. The response is preferably sent as XML, audio binary, or raw text, but may alternatively be any sort of messaging format, including HTML, delimited text, key/value text or binary encoded format. The HTTP response preferably includes directions to perform telephony actions. The response may alternatively or additionally include a new URI or a new URI template to use with the telephony action in Step S3. An additional example XML response is shown in FIGURES 5A and 5B. Additionally, the response preferably passes through a media layer. The media layer preferably performs any necessary caching and/or processing on returned media files and/or instructions to create a telephony compatible media file. The operation of the media layer is preferably transparent to the call router such that the media layer provides properly formatted media to the call router preferably without the call router being aware of the media conversion. The method of caching media during a telephony session is further described below.

1B. Processing Telephone Instructions

[0033] The step of processing telephone instructions with a call router S120 preferably functions to convert the server response into telephony actions or executable operations during a telephony session. The telephony actions may include, for example, playing a pre-recorded sound file at a server-specified URI (such as a static mp3 file located at http://demo.twilio.com/myapp/1234.mp3), reading text to the caller using text-to-speech technology, calling another number (such as creating a new voice connection through the PSTN, SIP/VoIP, or other IP technology system), collecting digits via DTMF input, recording voice response audio, TTY or other inputs, sending an SMS message, or any suitable combination or sequence of these or other suitable actions. This conversion of the server response is preferably performed at a call router. Preferably, Step S120 includes processing the response mime-types associated with the server response. For example, if the response mime-type is XML, it is considered to be a set of call router instructions. If the response mime-type is MP3, it is considered a sound file to be played for the caller. If the response type is plain text, it is considered to be text to be read, via Text-To-Speech, to the caller. Response mime-types associated with media handling are preferably passed through the media layer, and may be modified, processed, or created within the media layer. In the case where call router instruction includes playing a media file from an external server, the call router preferably sends the appropriate HTTP or HTTPS request to the external server. This request is preferably passed through the media layer. The media layer either completes the request by fetching the media file and performing any necessary media processing or the media file may alternatively be cached, in which case the media layer returns the a pre-processed, cached version of the media file. Response mime-types that involve the generation of media such as Text-To-Speech instructions, are additionally handled by the media layer.

The media layer preferably handles the querying a cache, generating the necessary media, and/or caching the media.

[0034] Contents of the server response, such as an XML document, are preferably converted into a telephony action by processing the document sequentially (e.g. line by line). Telephony instructions are preferably contained within the document in the form of a markup language, such as XML as shown in FIGURES 5A and 5B. This sequential approach to processing a document of telephony instructions is enabled when the communication is stateless and all the necessary information is contained within the URI. This stateless communication preferably allows telephony instructions (verbs or commands) to be used as the programming interface for a server application performing telephony services. Algorithmic interpretation (based on the state of the communication) of the telephony verbs or the document is preferably not necessary. The telephony actions are preferably executed in the order of telephony instructions found in the contents of the server response. For example, an XML document may include the necessary verbs to carry out the telephony actions of reading text to a caller, monitoring keys pressed by the caller, and redirecting the caller to a new URI using the pressed keys as part of the data within the new URI. Preferably, the telephony action (such as digits pressed) results in new state information, which may result in a repetition of some steps of the method, preferably beginning at Steps S3. The next URI is preferably provided by the server as part of the processing instructions. In another variation, the last URI is reused if the server fails to specify a next URI. In yet another variation, no repetition occurs if the server fails to specify a next URI, and processing continues below at the next call router instruction. The behavior may be determined by the nature of the call router instruction; for example, instructions that generate no new state information would not need to have a next URI since they don't trigger communication with a remote server. More preferably, the telephony actions result in the repetition of step S3 with the new URI resulting from Step S11, but may alternatively initiate a repetition of one or more steps (Steps S5, S7, S9, or S11) of the method. Step S3 is preferably repeated using all new phone session state information resulting from execution of a telephony action, such as digits pressed, a recorded audio file, or the success or failure of any telephony action requested. Repetition also includes all state information that remains relevant during the course of the session, such as Caller, Called, unique Call ID, and call status. The state information may also be represented in the form of a URI Template. For example, if the server response specifies that the call router should collect DTMF digits, and specifies that the next URL is the URI Template http://demo.twilio.com/foo.php?digits={Digits}, and the caller presses 1234, the resulting URI is http://demo.twilio.com/foo.php?digits=1234. Similarly, if the server response specifies the URI Template: http://demo.twilio.com/myapp/{Digits}.mp3, the resulting HTTP Request could be to a static mp3 file located at: http://demo.twilio.com/myapp/1234.mp3. Thus, a call may be controlled by one server that issued the telephony instruction and a second server that processes the response, as shown in FIGURES 7 and 8. Such call control hand-offs constitute the transfer of state information between servers in the form of a URI and accompanying request data, such as GET, POST, and/or request body. Preferably, all state communications conform to a syntax established by the call router to facilitate integration between multiple servers. For example, digits pressed on the keypad are preferably communicated to application servers in an identical fashion, thus minimizing the need for coordination between a multiple application servers with regard to how state is transferred. Alternatively, call router instructions may dictate the method of communicating new state information, such as the names and types of variables to send representing new state.

1C. Creating Resources Accessible by a Call Router API

[0035] The step of creating call router resources accessible through an Application Programming Interface (API) S130 preferably functions to expose information and/or functionality of the call router. The interaction from outside parties is preferably performed via the API (call router API). The Call Router API may additionally cooperate with the use of telephony instructions to function as a storage and retrieval format for data generated or required by the call router's operation. The Call Router API is preferably an application programming interface (API) such as a REST API (Representational State Transfer) as is known in the art, but the Call Router API may alternatively be a SOAP (Simple Object Access Protocol) API or any suitable programmatic communication interface. The Call Router API preferably may be used by an application asynchronously to the execution of a call (such as to later query the call records or retrieve recordings). Alternatively, the Call Router API may be used synchronously during the course of a call (such as to alter the state of the call, hanging up a call, initiating call recording, etc.). The Call Router API preferably stores state information in a persistent URI for a resource. The persistent URI preferably contains all the necessary state information, and this preferably makes data persistent, queryable, and recoverable. The Call Router API is preferably used for modifying resources to alter state of call router and for interacting with media of the call router. An application server can use the Call Router API to preferably query meta-data of call records, caller identification, call media (such as recordings, text transcripts, etc.), account information, transfer or interact with in-progress communications in the call router, and/or any suitable data generated by or required to operate the call router. The Call Router API preferably involves communication between an application server and a call router, but may alternatively be communication from any suitable device to the call router. The Call Router API preferably resides on the same hardware as the call router, but may alternatively reside on remote hardware or on any suitable hardware environment. The communication is preferably HTTP, but alternatively HTTPS or any suitable communication protocol may be used. The Call Router API may additionally be compatible with any HTTP client. The telephony system of the preferred embodiment preferably implements a Call Router API that includes a Call Router API request format, a Call Router API response format, and a plurality of API Resources representing types of data generated by or used by the Call Router.

[0036] The Call Router API request of the preferred embodiment functions as a communication message sent from an application server to an API resource of the call router. The Call Router API request is preferably sent from an application server to a call router, but may be sent from any suitable device to the call router. The Call Router

API request is preferably similar to a REST API request, but the Call Router API request may alternatively conform to any suitable programming principle, such as SOAP. The Call Router API request preferably uses HTTP to interface with a resource, but HTTPS or any suitable communication protocol may be used. Preferably the HTTP or HTTPS method of GET is used to retrieve a resource or resource information, and the HTTP or HTTPS method of PUT or POST is used to create or update a resource. In some cases, PUT or POST may be used to affect the functionality of the call router by modifying the state of a resource. Alternatively, a method parameter may be included in the URI of the resource to identify a requested action for the resource, or any suitable commands or methods may be used to interface with an API resource. The Call Router API request preferably includes authentication such as basic HTTP or HTTPS authentication, by including message authentication information in the URI, such as a cryptographic hashing of the request content using a shared key, or by any suitable method.

[0037] The Call Router API response of the preferred embodiment functions as a communication sent in response to a method performed on an API resource. The Call Router API response is preferably sent from the call router to an application server, or any suitable device. The Call Router API response is preferably sent in response to a Call Router API request, and the response is preferably sent to the originating device. The Call Router API response is preferably similar to a REST API response, where the response is a representation of the requested resource. The Call Router API response may alternatively conform to any suitable programming principle such as SOAP. The Call Router API response is preferably returned as formatted XML with information corresponding to the HTTP status code, a message, error codes, and/or any suitable information related to the resource. The Call router API response may alternatively be represented as Comma-separated values list (CSVs), HTML, JSON, or any suitable format. In one variation, the response format is determined by a portion of the requested URI, such as a file extension. In one variation, an API resource may be a binary data resource, and the Call Router API response is preferably formatted in a native binary format (e.g., a wav or mp3 audio file), an XML meta-data description, and or any suitable format.

[0038] The API resource of the preferred embodiment functions as an addressable representation of call router meta-data, internal call router state, or the state of a given resource used by the call router. An API resource is preferably addressed by a persistent URI. Preferably, the API resource responds to at least one HTTP action of POST, PUT, GET, or DELETE. The API resource may alternatively respond to multiple HTTP actions. The API resource may alternatively respond to any suitable method(s) that are preferably included in the Call Router API request. Consistent with the RESTful conventions, a GET request of a resource may return the current state of a resource, while PUT may update the state, PUT or POST may be used to create a new resource, and DELETE may be used to destroy a resource. The call router API may alternatively be used to affect the functionality of an in-progress call in addition to modifying data. The API resources of the preferred embodiment include an account resource, caller ID resource, incoming address resource, call resource, media resource, and/or any suitable resource of the call router. The API resources may alternatively be any suitable combination of the listed resources or other suitable resources. An API resource is preferably a preconfigured (or "static") resource, such as account information, or a resource actively in use by the call router, such as a phone call. Modifying the state of a resource via the API may additionally affect the operation of the call router in real-time, affect the state or capabilities of the call router in the future, and/or have any suitable effect.

[0039] The account resource of the preferred embodiment functions to allow an application to retrieve and/or modify account information. An account is preferably created by a telephony service provider, such as the operator of the call router. Information such as account name, usage information, contact information, initial URI, setup parameters, or any suitable account information may be retrieved or edited by an application using the account resource.

[0040] The caller ID resource of the preferred embodiment functions to allow an application to retrieve, modify, register new caller ID's (phone numbers), and/or delete caller identification information. The caller identification information is preferably for the phone number associated with out-going calls made by an application and/or user (i.e. where the application appears to be calling from). The numbers for outgoing calls are preferably assigned or verified prior to being used as a caller ID. As an alternative, to prevent fraudulent use of caller ID phone numbers in applications, a verification step may be used by the API before adding a new caller ID resource. A request to add a caller ID may be initiated via a request to the API, wherein a random validation code is generated and returned in the API response. The validation code is preferably provided to an end user. A phone call is placed to the given phone number (caller ID), requesting that the validation code be entered via keypad digits or spoken. Entry of the validation code verifies possession of the phone number, or the device associated with the phone number, at the time of the request. Use of the caller ID resource may additionally be presented in a user interface, such as a web browser, by displaying the verification code. User interface may be provided by the operator of the call router, or may be provided by any suitable application using the API. Any suitable method may also be used for verification of a caller ID. In another alternative, where multiple parties are involved in a call, the caller ID of one of the existing party members may be assigned for additional outgoing calls during that call session.

[0041] The incoming address resource of the preferred embodiment functions to allow an application to get, modify, or provision new inbound DID phone numbers, SMS short codes, SIP Addresses, etc. for use with applications. PUT or POST may be used to set the initial URI associated with the inbound address. DELETE may be used to release the resource. The incoming address resource may be used for real-time provisioning of phone numbers or other addressable inbound identifiers.

[0042] The call resource of the preferred embodiment functions to allow an application to get or modify the state of a telephony session in the call router. A telephony session or call may be in-progress, completed, failed, not yet initiated, and/or in any suitable call status. A call resource can preferably change the state or connection of an in-progress call. State changes preferably include: hanging up or terminating existing telephony sessions, transferring one or more existing telephony sessions from one contextual group of sessions to another, merging or splitting an existing group telephony sessions, transferring one or more telephony sessions from one communications medium to another (such as from one URI to a second URI), injecting an event or notification into a existing session or group of sessions, recording or ceasing to record the audio from one or more parties on a call, and/or any suitable call action. Call information or call log data can preferably be retrieved by sending a GET to the call resource or by alternatively sending any suitable method. Outgoing calls may also be initiated by using a POST or any suitable method that preferably indicates that a new call resource is to be created. When using the call resource to initiate a call, information may be provided as required to place a phone call, such as a caller ID to present, a phone number to call, and/or a URI to handle the call, but alternatively any suitable information may be provided. A call instruction XML document may alternatively be provided to the API instead of a URI, which is to be used for call instructions. The Call Router API may additionally respond with the status of a call such as if the call is answered, if a machine answered the phone, busy signal, no answer, call failure, and/or any suitable call status. The response may alternatively indicate that the new call request was accepted, but has not yet been initiated. In the example shown in FIGURE 6, caller information and caller ID are included in a POST request to the call resource. This step would initiate an outgoing call to the phone number designated in the caller information. The Call Router API response includes available state information regarding the call, such as whether the call has commenced yet, the call start time, end time, price, caller info, and the Call Router API response could alternatively include any suitable information. Additionally, information about the call returned at any point by the API may depend on the status of the call. For example, a call start time would not be given if the call has not yet begun, or the call end time, duration or price would not be given if the call had not yet ended.

[0043] Additionally or alternatively, the call resource of the preferred embodiment may be used to transfer a call to a new URI by a single call resource receiving a POST, PUT, and/or any suitable method. In this alternative, a call is preferably transferred to the new URI for new call instructions. The API may preferably be used to issue asynchronous changes in call state, unlike the synchronous communication between the call router and application server for synchronous URI requests and responses. The call resource, in this alternative, functions to allow a call to be asynchronously directed to URIs. Examples of various applications of the call resource include initiating a new telephony session, terminating an existing telephony session, call waiting, call holding, call queuing, call parking, private call sessions within a conference, carry on multiple call sessions, and/or any suitable application. Any situation where asynchronous events affect the call status, such as a call agent becoming available, or a person returning to the phone after placing a caller on hold. The currently executing call router instruction may be allowed to complete, or may be immediately terminated, before requesting the provided URI. New call state resulting from the last call instruction executed by the call router, such as digits pressed on the keypad or audio recorded from the caller, may be provided to the new URI in a form POST or GET parameters, or may alternatively be discarded by the call router and not provided. As shown in FIGURE 9, call waiting may be implemented by an application sending a Call Router API request to the call resource that POSTs a new URI for the call. The caller is then directed to the new URI for instructions. A second Call Router API request is sent to the call resource that POSTs the original URI for the call, and thus brings the caller back to the first call session. The call resource may alternatively be used in any suitable application.

[0044] As an alternative embodiment of the call resource, a calls resource may implement a plurality of individual calls as distinct subresources. For example, a URI ending in "/Calls" may be a list of many calls performed by the account, and a URI ending in "/Calls/12345" may represent one specific call, uniquely identified by the key "12345". The calls resource preferably allows retrieval of many call records and/or creating new calls, while a single-call resource represents a single call. The calls resource preferably accepts a request to create a new call resource, as is common in RESTful architectures, which in the Call Router API, preferably serves to initiate one or more new calls. A calls resource may be used to both list current and previous calls using the GET method, as well as initiate a new outbound call using the POST method. Using RESTful principles such as POST or PUT to alter the state of an individual call resource can preferably change the state of an in-progress call, affecting the realtime activities of the call, such as by hanging up, transferring control to a new URI, joining the call with another call, or any suitable telephony action.

[0045] The media resource of the preferred embodiment functions to allow an application to retrieve and/or access information of media stored, cached, created, and/or used during a call. In one variation, the media resource is preferably a recording resource to access information and recordings made during a call via recording call instructions, or asynchronously via the Call Router API. In another variation, the media resource may alternatively include call transcripts, text messages, key press logs, faxes, a binary-coded resource, and/or any suitable media. The media resource may alternatively include a URI of the binary-coded file (such as a wav, mp3 audio file or PDF document file). In one variation, the media resources may additionally be integrated with the telephony instructions (or markup language) such that a telephony instruction may instruct the call router to perform an action that creates a media resource. The call router preferably sends a response to the application server with the URI of the created media resource. For example, when the call router is instructed to record a message, the call router preferably sends a response to the application server with a unique URI of the recorded message within the API. The media URI preferably responds to GET requests to return the media in a number of formats, such as binary or XML meta-data representations. The media resource may accept requests to delete a media resource. In one variation, the media resource preferably requires authentication to access the resource. In another variation, the media resource may not require authentication to enable URI embedding in a variety of applications, without exposing authentication credentials. In yet another variation, authentication is preferably performed via cryptographic hashing, such that credentials are not exposed to client applications that consume the media resources. In another variation, the media resource allows the initiation of transcription of audio resources to text using transcription technology. The audio resource used for transcription is preferably generated during telephony sessions (such as by using the record instruction) and hosted on the Call Router API. The media resource preferably allows retrieving or deletion of audio transcriptions generated from recorded media. The media resource may additionally allow centralized hosting of media files, and the resource URIs are preferably exchanged between the call router and the application server, instead of the large media files themselves. The media resource may alternatively be used for any suitable media.

[0046] Additionally or alternatively, a join resource of the preferred embodiment may be used to join one or calls into a shared session that allows the parties to communicate (i.e., a conference) by a single call resource receiving a POST, PUT, and/or any suitable method. In this alternative, one or more calls are preferably join together such that they are in a conference. The join resource may alternatively be a subresource or part of the call resource.

[0047] Additionally or alternatively, a split resource of the preferred embodiment may be used to split shared sessions (e.g., a conference) into individual call sessions by a single call resource receiving a POST, PUT, and/or any suitable method. In this alternative, one or more shared sessions involving two or more calls are preferably split such that one or more calls are split into separate calls or into on or more separate conferences. The split resource may alternatively be a subresource or part of the call resource.

2. Method of Caching Media for use in a Telephony Session

[0048] As shown in FIGURES 12-14, the method 20 of the preferred embodiment for processing media includes mapping a telephony media request to a resource address S220, sending the request to a cache server S230, forwarding the request to a media processing server S240, and caching responses S250. The method functions to generate telephony compatible media resources for improved efficiency. In one application, the method is preferably implemented during a telephony session, and is more preferably implemented during a telephony session established using the method 10 for processing telephony sessions. In this variation, the method 20 preferably includes the additional steps of initiating a telephony session S210, and after caching the response, sending the response to a call router S260, where the call router preferably executes the media resource. In another variation, the method is implemented with media layer API interaction. The media layer API preferably provides media layer control beyond normal capabilities (such as those possible by HTTP caching directives). Third party servers or applications preferably use the media layer API, but alternatively the call router may use the media layer API. This variation may be implemented outside of a telephony session or during a telephony session. The method 20 functions to handle call flows from a call router and uses a media layer to streamline network traffic and transparently handle aspects of application specific processing. The method preferably uses less bandwidth and stores the results of computationally intensive operations and/or large network transmissions to improve the perceived responsiveness of telephony systems to callers. The method additionally lowers the cost of service by enabling a larger volume of calls to be handled using the same CPU and network resources. The method 20 preferably utilizes a HTTP media transport protocol for the services of the media layer. HTTP is preferably used for external communication and for internal communication of the media layer. This allows the physical systems of the media layer (e.g., transcoding proxy servers, text-to-speech servers, encryption servers, etc.) to be distributed across different network-isolated systems and to be scaled independently. The use of the HTTP protocol additional enables the dynamic and automatic scaling of resources within the media layer. Systems of the media layer such as caches, text-to-speech servers, transcoding proxy servers, or other media processing servers may all be automatically load balanced independently of other systems.

[0049] Step S210, which recites initiating a telephony session, functions to accept an incoming call. The call preferably originates from PSTN-connected (Public Switched Telephone Network) or Internet addressable devices, such as landline phones, cellular phones, satellite phones, Voice-Over-Internet-Protocol (VOIP) phones, SIP devices, Skype, Gtalk, or any other suitable PSTN-connected or Internet addressable voice device. The originating phone number of the PSTN device is preferably captured using caller ID, but any other suitable ID may be captured, such as a VOIP provider ID. The dialed phone number and/or the date and time of the call are also preferably included in the session information. An authentication ID may additionally or alternatively be included in the session information. Step S210 is preferably substantially similar to Step S1 of method 10 for processing telephony sessions.

[0050] In one preferred variation of the invention, Step S210 functions to initiate a telephony session (such as a phone call) via an HTTP or other request sent to a call router from an application running on a third-party server. In this variation, the application running on the server preferably specifies an initial URI of an application server for the call router to use for the telephony session as well as the phone number (or other addressable destination) to dial, geographic information and the source phone number (caller id).

[0051] Step S220, which recites mapping a telephony media request to a resource address functions to convert a telephony session into a format that may be handled with standard web servers and web applications. The telephony media request is preferably received during a telephony session, but may alternatively be received from the media layer API when a telephony session is not established. A call router may preferably receive the media request or alternatively the media layer API may receive the request. Additionally a call router may initiate the media request and use the media layer API to interface with the media layer. The telephony session is preferably mapped to a Universal Resource Identifier (URI), but any suitable resource addressing protocol may be used. Step S220 is preferably substantially similar to Step S3 of the method 10 for processing telephony sessions. Preferably, the mapping and/or conversion are performed using a call router. The initial address or URI is preferably pre-specified at the call router by a web application (which may be running on a third party server) or call router account owner. More preferably, the initial URI is assigned to the call via a unique identifier for the call destination, such as a DID (Direct Inbound Dial) phone number, or a VOIP SIP address. In another preferred embodiment, the URI is specified by a remote server. Alternatively, the media layer API may specify the resource address using any suitable interface, but the media layer API is preferably a REST API. The URI may encapsulate at least a portion of the state information from the initiated telephony session, such as the originating phone number, the dialed phone number, the date and time of the call, geographic location of the caller (e.g. country, city, and/or state, zip), and/or the unique call ID. The URI is preferably associated with a media resource such as a media file location or a location of a media processor or generator. The URI may additionally include media parameters. The media parameters are preferably used in the processing or generation of a media file. The parameters may additionally or alternatively be embedded in the header or body of an HTTP message. The information included in the URI may be included in the form of a URI template. For example the URI default template could be:

http://demo.twilio.com/myapp/{dialed phone number}/{originating phone number}

or

http://demo.twilio.com/myapp/foo.php?dialed_number={dialed phone number}&
originating_number={originating phone number}

[0052] In one variation, the request is preferably made via a secure protocol, such as HTTPS. The HTTP header containing the request preferably includes an SSL header, indicating the final forwarding of the request to the URI is to be performed using SSL (HTTPS authenticated requests). This adds an additional layer of security to the application server, protecting valuable content from being accessible, protecting privacy of all communicating parties, and protecting the application server from malicious activity while allowing internal components, such as cache servers or transcoding proxies, to process request and response data.

[0053] In another variation of Step S220, plain text or XML of Text-To-Speech instructions are converted into a Text-To-Speech (i.e., speech audio) via an HTTP request sent to a URI of a Text-To-Speech web service. The call router preferably constructs a URI consistent with the Text-To-Speech web service when the call router determines a Text-To-Speech process is needed. The call router preferably sends the URI request to the Text-To-Speech web service, and more preferably sends it via the cache server. The call router preferably makes the request based on program instructions, media received from an application server, or any suitable event. The HTTP request preferably includes the desired Text-To-Speech conversion (text, voice type, speech speed, and/or any suitable setting) and the full text to be converted. The URI preferably includes the full text to be converted. The URI may alternatively include a fixed length cryptographic hash of the desired conversion including the full text. The full text is preferably included in the HTTP headers. This alternative functions to provide a unique URI for a specified conversion but having the URI limited to a certain length. The Text-To-Speech conversion request also preferably includes a voice selection (e.g. female, old man, child, etc.), but alternatively if no voice is selected, a default voice may be used for the converted speech. In one variation, the Text-To-Speech conversion request also includes a language specification to specify the language to be used for the conversion.

[0054] Step S230 functions to send the request to a cache server. Preferably, the request for a URI is sent to a cache server over a network. Step S230 preferably includes the cache server checking if the requested URI is already cached. The cache preferably stores telephony compatible media files. Telephony compatible media is preferably media in a suitable format for use with a telephony device. The media file may have previously been processed within the media layer prior to being cached or have been created by an application operator in a telephony compatible state. The telephony compatible media file and the original media file (the requested media) do not necessarily share a common media description and could differ in sampling frequency, bit rate, media type, or any suitable characteristics. For example, a video file is preferably stored as an audio file when a telephony compatible media file is cached. The URI of a telephony compatible media file is preferable cached with a persistent URI (or persistent address). The persistent URI functions to allow media to be requested which does not necessarily correspond to the media returned. This is an aspect of the transparent description of the media layer where the processing and caching operations of the media layer are carried out without the knowledge of the call router. When a URI specifying a video media file is requested, a telephony compatible audio file that has been cached and associated with that persistent URI is preferably returned. The media parameters embedded in a URI can additionally be used to identify cached media. For example, audio of a Text-To-Speech conversion is preferably cached with a media parameter describing the contents of the media, such as a cryptographic hash or the actual text voice settings, or any other Text-To-Speech variables. If the URI has been cached and the cache is still valid (based on an HTTP expires tag, a HEAD request to the URI resulting in a 304 “Not Modified”, or any other suitable cache maintenance algorithm), the cached content is returned to the sender, and Steps S240 and S250 are preferably skipped. However, if the URI has not been cached, or the cache is determined to be invalid (e.g. due to expiry, URI updates, etc.), then the HTTP request is preferably forwarded to another media layer server (e.g. a transcoding proxy server and/or a Text-To-Speech conversion server, or any other suitable server in the media layer) or the application server for processing. Using a dedicated hardware for process specific tasks functions to increase processing time and improve time response. In one variation, the request is sent to a local cache server on a local area network. In another variation, the request is sent to a server running locally on the device originating the call. In yet another variation, the request may be sent to multiple servers. In another variation, the request may be sent to another cache server if the cache is partitioned or hierarchical. The state information from the initiated telephony session is preferably sent via HTTP POST, HTTP GET or HTTP header parameters to mimic the data flow of a web browser. Communication between the cache and other media layer servers (e.g., media processing servers) is preferably operated in a controlled or trusted environment (e.g., an intranet) and a non-secure communication protocol such as HTTP is preferably used. Alternatively, the cache may use third party or external servers for storing media. In the case where external networks or servers are accessed, a secure communication protocol such as HTTPS may alternatively be used.

[0055] Step S240 functions to forward the request to a media processing server. The media request is preferably processed within the media processing server. The media processing server is preferably an audio processing server but may be any suitable signal processing server. In a first variation, the media processing server is a Text-To-Speech web service. In a second variation, the media processing server is a transcoding proxy server. In a third variation, the audio processing server includes both the Text-To-Speech web service of the first variation and the transcoding proxy server of the second variation. The media processing server preferably generates a telephony compatible audio file, but may alternatively perform any suitable task. In the case where the telephony device is an SMS or MMS device the telephony compatible media generate may be text or images compatible with the messaging service. The processing server is preferably capable of streaming media content to a destination.

[0056] As shown in FIGURE 13, the first variation of Step S240 includes the step of converting Text-To-Speech. The text to be converted is preferably included in the URI or in an HTTP header as described above, and also preferably includes a specification of a Text-To-Speech conversion (text, voice type, speech speed, and/or any suitable setting) as media parameters. The Text-To-Speech audio is preferably generated based on the voice selected and any other suitable Text-To-Speech parameters such as language, emotion, talking speed, G-rated, etc. The audio is preferably generated as 8-bit PCM mono with 8 kHz bandwidth (the standard for telephony voice communication), but may alternatively be generated as a binary audio file such as an MP3 file, a WAV file, or any other suitable binary audio file. The audio file is preferably transmitted back to the cache server, but may alternatively be transmitted directly to the call router, or to a transcoding proxy server for audio transcoding. The transmitted audio file may additionally be streamed to a destination location (e.g., streamed to the cache server).

[0057] In one further variation of the first variation of Step S240, the step of converting Text-To-Speech may be reversible (e.g. speech to text) and the forwarded request may include a binary audio file to be converted to text. This further variation may include transmitting the converted text to a caller via SMS, email, TTY or any other suitable transmission medium.

[0058] One variation of the first variation of Step S240 preferably includes the step of automatically selecting and professionally recording frequently used words and phrases. This variation of Step S240, preferably includes the step of determining frequently used words and phrases, where the frequently used words and phrases are preferably defined to be words and phrases used more than a pre-specified frequency over a period of time. The frequently used words and phrases may alternatively be specified by an application operator. A Text-To-Speech telephony instruction may include an API flag or any suitable mechanism indicating that the phrase should be professionally recorded. The frequently used words and phrases are preferably transmitted to a professional recording studio where the words and phrases are professionally recorded by a voice actor in a studio. The transmission is preferably email, but may alternatively be an SMS message, a fax, a mailed document, or any other suitable transmission. The professional recording may be recorded word by word, phrase by phrase, with multiple intonations, or any other suitable recording methodology. The professional recording is preferably transmitted electronically to an application server, but may alternatively be transmitted to a Text-To-Speech conversion web service, a cache server, a transcoding proxy server, a call router or any other suitable web server. The professional recordings are preferably transmitted in 8-bit PCM format, but may alternatively be transmitted in WAV, MP3, or any other suitable audio format. The return transmission of the professionally recorded audio files is preferably an HTTP post request, but may alternatively be an FTP transfer, an email, a mailed data storage device (e.g. CD ROM, memory card or Flash Disk) or any other suitable transmission.

[0059] As shown in FIGURE 14, the second variation of Step S240 includes the step of transcoding an audio file of a URI resource on an application server. In this variation, the HTTP request is forwarded from the HTTP cache server to the transcoding proxy server that requests a URI from an application server. The transcoding proxy server additionally may use a secure protocol during communication with the application server. The secure protocol is preferably HTTPS, though any suitable secure protocol may alternatively be used. The use of the secure protocol is preferably indicated by an instruction included in the headers of the HTTP request from the call router and alternatively via the HTTP cache. The second variation of Step S240 also includes retrieving the resource located at the URI from the application server. Upon receipt of the resource, Step S240 includes the step of determining if the media type requires conversion. The transcoding proxy server may additionally determine if conversion is required or desirable for a media file by analyzing the MIME-type of a media file. The transcoding proxy server preferably uses a preconfigured MIME-type to transcode audio to another MIME-type (e.g. from 128 Kbps MP3 audio to 8-bit PCM mono with 8kHz bandwidth for telephony applications). Media conversion instructions, such as which MIME-types should be converted into which other formats, may be pre-configured on the transcoding proxy server, or may be passed at the time of the HTTP request from the call router and/or HTTP cache server. If the media type does not require conversion, the media resource is sent forwarded without modification. The application server response is then sent to the HTTP cache server for possible caching, but alternatively may be transmitted directly to a call router if needed. The transcoding proxy server may alternatively stream the resource located at the URI from the application server, sending response data to the cache server or call router as it is downloaded and/or transcoded, without waiting for the complete operation to finish.

[0060] As another variation the Step S250 may include encrypting media. This step functions to cache media in a secure format. The media encryption preferably happens to the media file being cached. Media that is needed for a telephony session is preferably not sent to the call router encrypted. Either a non-encrypted telephony compatible version of the media file is sent to the call router, or alternatively, the media file is decrypted prior to transmitting to the call router. Step S250 may include decrypting media. When decrypting media a shared key is preferably shared between a client (application operator) and the system administer. However, an encrypted media file may be passed back to the client (using the media layer API) in situations where the client desires to not transmit decrypted media.

[0061] Step S250, which recites caching possible server responses functions to reduce redundant data transmissions and streamline network traffic. This reduces bandwidth and processing resource usage and lowers the cost of service by enabling a larger volume of calls to be handled per computing resource unit, while improving the perceived response time for callers. Preferably, Step S250 includes caching all server responses received from application servers and servers in a media layer, such as a transcoding proxy server response, a Text-To-Speech web service response, an application server response, or any other suitable server response. Step S250 preferably checks if the response is cacheable, for example caching a common audio response delivered either as an audio file directly from an application server, a transcoded audio file, an audio file containing converted Text-To-Speech. Customized content (e.g. personal voice mailbox recorded messages) and private personal data files (e.g. bank account balances) are preferably not cached. Alternatively, the HTTP response of the application server may include cache-related directives. The cache-related directives are preferably included in the HTTP headers, and include instructions for handling the URI resource. More preferably, the instruction includes whether or not a HTTP response is cacheable and/or the expiration of the URI resource (how long the URI resource remains cached on the cache server). The cache preferably deletes the URI resource when the expiration time has been reached. Alternatively the media layer API may alternatively be used for cache control. Though, any suitable cache-related directive may alternatively be used. The cache may additionally store media on an external server. The external server may be operated by a third party or be located in a remote location. The media is preferably encrypted in this variation such media stored in uncontrolled servers does not have compromised security. When retrieving media from the external server, the media layer may handle the decryption or alternatively pass the encrypted media onto the client such that the media layer never observes unencrypted media.

[0062] Step S260 functions to send a response. The response is preferably an audio file or XML call flow instruction file for playback and processing. The response is preferably sent to the call router if the call router originated the media request. Alternatively, the response may be sent to a destination specified by the media layer API, preferably the server using the media layer API. The response is preferably sent as XML, audio binary, or raw text, but may alternatively be any sort of messaging format, including HTML, delimited text, key/value text or binary encoded format. The response is preferably sent to the call router independently of whether or not the response is cached. This response is preferably an HTTP response. The HTTP response preferably includes directions to perform at least one telephony action (e.g. play this file, connect to this number, access this URI). The telephony compatible media file is preferably played by the call router for the telephony device. In one variation, the cache server streams the response to the call router, sending response data as it becomes available without waiting for the entire response. The call router consumes the media in a number of variations. The call router may consume a HTTP media stream and render the audio over the PSTN, VoIP network, or any suitable network to the telephony devices. The call router may append the data from the media layer to a file as the data becomes available and begin rendering the media from the file immediately (e.g., using a first in first out FIFO abstraction). The call router may append from the media layer to a file as the data becomes available and start rendering the media when the stream is finished.

3. System for Handling Telephony Sessions

[0063] As shown in FIGURES 2A, 2B, 3A, 3B, 11 and 15 a system 20 and 30 of the preferred embodiment for handling telephony sessions includes a call router 22, a resource address 23 for an application server, a telephony instruction 27, and a call router resource 29. As shown in FIGURES 2A and 2B, a first configuration 20 is initiated by a telephony device (such as a telephone call, fax or SMS message). As shown in FIGURES 3A and 3B, a second configuration 30 is initiated by an application developer side (i.e., server 26 calling out). The telephony system of the preferred embodiment preferably additionally implements a Call Router API 28 that includes a Call Router API request format, a Call Router API response format and a plurality of resources substantially similar to those described above. The system of the preferred embodiment additionally includes a media layer 40 that functions as an intermediary hardware/software layer for application media processing as shown in FIGURE 11. The media layer 40 preferably includes a cache server 42 and a media processing server 42. The media processing server 42 may include a transcoding proxy server 43, a Text-To-Speech web service 44, and/or any suitable media processing device.

[0064] The call router 22 functions to initiate or receive calls from the telephony device and connect to a web-application server. The call router 22 is preferably connected to a PSTN device over the PSTN network, such that it can receive and make calls from PSTN-connected devices 21, such as landlines, cellular phones, satellite phones, or any other suitable PSTN-connected devices, as well as non-PSTN devices, such as Voice-Over-Internet-Protocol (VOIP) phones, SIP devices, Skype, Gtalk, or other Internet addressable voice devices. The call router 22 may alternatively or additionally function as or include a message router for use with SMS messages. The call router 22 can preferably connect to an SMS network, such that it can receive and send messages from SMS network devices 21, cellular phones, computers, smart phones, or any suitable SMS network devices. The call router 22 may also send or receive text messages, multimedia messages, emails, faxes and other suitable PSTN-compatible communication messages. The call router 22 preferably communicates with the application server 26 using an application layer protocol, more preferably using the HTTP, or secure HTTPS, protocol. The call router 22 preferably communicates with the application server 26 through a media layer 40 using the HTTP protocol or a secure protocol such as HTTPS. HTTP is preferably used for communication for devices networked through an intranet such as between the call router 22 and the media layer 40 and within the media layer 40, and a HTTPS is preferably used for communicating with external servers or devices. The communication between the application server 26 and the call router 22 is preferably stateless and any state information (e.g., call state) or data is preferably located in a URI or the request parameters, such as HTTP headers, GET URI parameters, POST request body parameters, or HTTP cookies. Available state information is preferably transmitted by call router requests to the application server for stateless processing, and the application server preferably stores no state. Alternatively, the application server preferably stores local state information, such as databases or sessions, as is common in web development. The call router 22 preferably stores state information in call router resources 29. The call router resources 29 are preferably accessible by the application server 26 and other devices through the call router API 28. The call router resources 29 are preferably similar to those described above. The call router 22 preferably associates each incoming phone number with a starting resource address (or more specifically a URI) 23, more preferably the URI 23 is provided by the application server 26, still more preferably the URI 23 is provided by the application developer before a call is received at the call router 22 by associating the initial URI with the incoming call address (such as DID, SIP address, etc.) or by the application upon initiation of an outgoing call. The call router 22 preferably sends call data such as the caller number (obtained via Caller ID), caller geographic data (country, city, and/or state, zip) the number dialed, the time of the call, or any other suitable information or parameter. When an HTTP communication is associated with a media request (e.g., a file request or a media processing instruction), the associated URI 23 is preferably a persistent URI. A persistent URI functions to allow telephony compatible media stored in the cache to be returned in place of the requested URI. The call data is preferably digitally signed with a secret key 25 stored on the call router 22. A cryptographic hash of the information is preferably included along with the information as a digital signature.

The call router 22 may also encrypt sensitive information (either before or after the cryptographic hash is computed) using the secret key to allow sensitive information to be sent across the network. The call data is preferably sent as an HTTP POST request to the application server 26. Call data may also be sent in URL (GET) variables, or encapsulated in HTTP headers. An example HTTP request containing the information in the header is shown in FIGURE 4A and 4D. As shown in FIGURE 4B, further inputs (such as voice recording or DTMF button pressing) from the PSTN-device may be subsequently submitted to the application server 26 as HTTP requests (GET or POST). As shown in FIGURE 4C, the inputs from a phone keypad may be included in an HTTP GET request. As shown in FIGURE 4E, the content of an SMS message received by the call router may be sent to the application server 26 as an HTTP request. As shown in FIGURE 4F, the inputs from the text message are included in an HTTP GET request. The request data may alternatively be simultaneously sent in the URI (query string), message body (POST) and message headers, or any combination of the above. The call router 22 is preferably capable of handling media streams received by the media layer 40. The call router 22 consumes the media in a number of variations. The call router 22 may consume a HTTP media stream and render the audio over the PSTN, VoIP network, or any suitable network to the telephony devices. The call router 22 may append the data from the media layer 40 to a file as the data becomes available and begin rendering the media from the file immediately (e.g., using a first in first out FIFO abstraction). The call router 22 may append from the media layer 40 to a file as the data becomes available and start rendering the media when the stream is finished.

[0065] The preferred embodiment may additionally include a media layer Application Programming Interface (API) 50 that functions to allow programmatic access to the media layer 40 and in particular to the cache 41. The media layer API 50 is preferably RESTful in nature but any suitable protocol may be used. The media layer API 50 may be used for retrieving the status of a single cached resource, or alternatively a plurality of cached resources (what is cached, the date the media was cached, the file size, etc.). The media layer API 50 may additionally retrieve such status by specifying a partial or full URL to the canonical resource (i.e., show the cache status for the file located at: http://demo.twilio.com/foo.mp3). The media layer API 50 may additionally remove media from the cache. The media layer API 50 may request that one or more remote files be cached, specified as one or more URLs (ex: http://demo.twilio.com/foo.mp3) or request that that one or more remote URL resource that contains sub-resources be cached. For example, by requesting to cache http://demo.twilio.com/media/, the cache would preferably "crawl" that directory for sub-resources linked to, and cache those. Caching instructions such as media type, file size restrictions, modification date, or any suitable parameter may additionally be used for the crawling procedure. The media layer API 50 may alternatively or additionally be used by applications for any suitable purpose. One exemplary use of the media layer API 50 would be to preemptively cache media to the media layer. Such as if an application has media that will be commonly used or that changes on a periodic basis. Another exemplary use of the media layer API would be for accessing media generated during a telephony session after the telephony session has ended.

[0066] The media layer 40 of the preferred embodiment functions to streamline and reduce bandwidth usage. The media layer 40 preferably assigns CPU or network intensive tasks to dedicated hardware to improve the perceived response time for callers. The media layer 40 further functions to separate media processing from a core router and enable improved allocation of resources and scaling of infrastructure. The media processing is preferably transparent to the call router or other applications accessing the media layer 40 through the media layer API 50. Here transparent indicates that the call router 22 only needs to be aware of how to handle expected media (telephony compatible media). The media layer 40 preferably takes care of converting the requested media to a telephony compatible media type. Additionally, new media types can be implemented without modifying the operation of the call router 22. The presence of the media layer 40 may also lower the cost of serving each caller by reducing the bandwidth and computational resources needed, effectively increasing the number of simultaneous call flows handled by the call router 22 and the application server 26. The media layer 40 preferably includes a cache server 41 and a media processing server 42. The media processing server 42 preferably includes both a transcoding proxy server 43 and a Text-To-Speech web service 44, but alternatively, either a transcoding proxy server 43 or a Text-To-Speech web service 44 may be solely included in the media layer. Alternative servers or media processing devices may additionally be used. Each server is preferably run on an independent device, but alternatively, some or all of the servers 241, 242, and 243 in the media layer 40 may run on the same device. The media layer preferably includes the ability to stream data through each component of the media layer 40 (e.g., the cache, Text-To-Speech servers, and transcoding proxy servers). Streaming functions to minimize the delay when media is requested through the media layer 40. Each component of the media layer 40 is preferably capable of accepting a block of data, performing an operation, and writing that block out. Streaming is preferably implemented using HTTP 1.1 chunked-encoding, which allows data to be added, removed, or modified by intermediate nodes (e.g., transcoding proxy server). A load balancer may additionally automatically allocate or deallocate resources of the media layer. The load balancer (or a plurality of load balancers) preferably independently scales components based on independent usage and independent performance profiles (CPU-bound, disk-bound, etc.). Using the media layer, the stateful system components (e.g., the call router) are separate from the stateless components (e.g., the transcoding proxy servers, Text-To-Speech servers, etc.). Thus the components of the media layer are easily scaled for more capacity. Furthermore, because the components of the media layer communicate using HTTP, the components may be scaled independently. Additionally, since the media layer may be distributed, there is greater flexibility in the physical (hardware or software) implementation. Media layer 40 components may be operated on different kinds of hardware or virtualized resources. The components are preferably scaled automatically using the load balancer and may additionally use predictive techniques to anticipate capacity requirements. As one example, a client with a large Text-To-Speech demand may be doing high volume work. The Text-To-Speech services may be scaled up by allocating additional Text-To-Speech servers, without the need to alter the call router 22, transcoding proxy servers, or any other system components. When anticipating/predicting capacity requirements, the system may use usage history as an indicator of times of day when particular capacity requirements must be met.

[0067] The cache server 41 of the media layer 40 functions to improve the response time to the call router 22 and improve the quality of each call flow, while reducing usage of processing and bandwidth resources. The cache server 41 preferably accomplishes these goals by storing and re-transmitting the content of repeatedly accessed and/or processed URIs. The cache server 41 is preferably an HTTP cache server running HTTP cache server software such as the Squid proxy cache as is well known in the art, but alternatively any suitable cache server software may be used. The cache server 41 preferably facilitates communication between the call router 22 and the application server 26 and enables retrieval of a URI resource. The URI resource is preferably a telephony compatible media file, and is more preferably referenced by a persistent URI. A cached URI resource is preferably not an exact copy of a media file, but a media file that has been previously processed within the media layer to be telephony compatible. In the case where the original media resource was in telephony compatible format, then the cached version may be identical to the original version. The URI of a resource may additionally include embedded media parameters. The media parameters function to uniquely distinguish cached media. For example, a cached Text-To-Speech media file can be identified by an embedded cryptographic hash of the text. A HTTP request is preferably sent to the cache server 41. The HTTP request preferably includes HTTP request details including HTTP headers, POST, or query string parameters, but alternatively any suitable communication scheme may be used for communication. The cache server 41 preferably checks for a valid copy of the URI resource (a non-expired copy previously retrieved during a URI request or during any suitable time). If a valid copy is found within the cache server 41, the cache server 41 preferably responds to the call router 22 with the valid copy of the URI resource. If a valid copy is not found within the cache server 41, the cache server 41 preferably sends the HTTP request to the application server 26. The cache server may alternatively send the HTTP request to an intermediate server, such as the transcoding proxy server 43, Text-To-Speech web service 44, or any suitable server or service. The cache server 41 may additionally or alternatively use external servers 41' for storing media content. Media stored on external or third party servers 41' is preferably encrypted. When returning encrypted data the cache may decrypt the media file within the media layer prior to transmitting. In variation where encryption and decryption operations are performed, the media layer preferably includes an encryption/decryption server 46 to handle the encryption and/or decryption of a media file. To avoid additional processing and transmission time for the first initiated telephony session, the system preferably includes an application-testing program that primes the cache server 41 with the complete set of cacheable telephony responses, or a subset of responses such as large static media files, before a telephony session and/or the first telephony application use. This may alternatively, by programmatically implemented by the media layer API 50. Additionally, the HTTP response of the application server 26 and/or the transcoding proxy server 43 preferably includes cache-related directives. Preferably, the cache-related directives are included in the HTTP headers, and preferably indicate whether the response may be cached, and for how long a URI resource remains cached on the cache server 41. Such directives may alternatively be indicated in media layer API messages. When the URI resource reaches an expiration time, the cache server 26 preferably deletes or updates the URI resource. Alternatively, any suitable cache-related directive may be used, such as those defined in IETF or W3C Standards including W3C RFCs 2616, 2186, 2187. In one variation of the preferred embodiment, when the application server content changes on a daily basis (e.g. stock prices, weather conditions, scheduled appointments), the cache server 41 updates the complete set of URI resources at least one time per day, preferably during off-peak hours (e.g., updating schedules for the next day at 3 AM).

[0068] The transcoding proxy server 43 of the media layer 40 functions to convert audio files received from the application server 26 and optimize the audio files for telephony applications. The transcoding proxy server preferably acts as an intermediary between the cache server 41 and the application server 26. In one variation of the preferred embodiment, the cache server 41 sends the HTTP request to the transcoding proxy server 43, and the transcoding proxy server 43 preferably requests the URI resource from the application server 26. The transcoding proxy server 43 preferably receives the URI resource and decides if any CPU-intensive tasks are required before returning the content to the cache server 41. In one variation, the CPU-intensive task is preferably transcoding audio from one format to another format (e.g., a telephony compatible format). The CPU-intensive task may alternatively be using preconfigured

MIME-types to transcode audio to another MIME-type. The MIME-type of the URI resource along with introspection of the media is preferably used as criterion for deciding if to transcode. The CPU-intensive task alternatively may use instructions in the HTTP request, preferably the HTTP header, to transcode audio into a specified format such as 11Khz, 8bit mono PCM audio. Configuration parameters, HTTP instructions, or any other suitable information may additionally be used to determine if transcoding is required or desirable. The transcoding proxy server 43 may alternatively convert video files to audio, perform signal processing on audio, or perform any suitable media translation tasks. After any mime-type conversion, the transcoding proxy server 43 preferably updates the mime-type in the response.

[0069] The audio files received from the application server 26 are preferably MP3 files, but may alternatively be WAV, AAC, RA, MP2, WMA, or any other suitable audio format or encoding. The transcoding proxy server 43 may also receive XML instructions and/or plain text information from the application server 26, which preferably passes the instructions and/or information through to the cache server 41 unmodified. The cache server 41 preferably connects to the transcoding proxy server 43 using an application layer protocol, such as the HTTP protocol. Preferably, the transcoding proxy server 43 connects directly to the application server 26 using an application layer protocol, such as the HTTP protocol. Alternatively, the transcoding proxy server 43 uses a secure protocol that functions to provide security during communication with the application server 26. The secure protocol is preferably HTTPS though any suitable protocol may be used. The use of a secure protocol is preferably indicated in the HTTP header of the HTTP request from the call router 22, and is passed through the cache server 41 to the transcoding proxy server 43. This preferably enables HTTP requests to be used in intranet connections (internal/controlled communication, such as between call router 22, cache server 41, and transcoding proxy server 43, where interim processing may be required) and HTTPS requests for Internet connections (external/uncontrolled communications). The use of a secure protocol further functions to allow an application developer or system administrator running the application server 26 to provide an additional level of security with respect to their web application. As another alternative, the transcoding proxy server 43 and cache server 242 stream URI resource responses as they are downloaded, transcoded, and/or cached, without waiting for the download, transcode, and/or cache operation to finish. This functions to allow large URI resources (such as large mp3 files) to begin playing before downloaded, transcoded, and/or cached entirely.

[0070] The Text-To-Speech web service 44 of the media layer 40 functions to convert textual input into audio speech. The Text-To-Speech web service 44 is preferably an independently running server, but may alternatively be located on the same device as the other media layer devices, or may alternatively be remotely accessible over the Internet. The Text-To-Speech web service 44 preferably receives the text to convert in an HTTP request from the call router 22 via the cache server 41. The HTTP request preferably includes a URI representing the desired conversion (text, voice type, speech speed, language, and/or any suitable setting) and the full text to be converted. The URI preferably encapsulates the full text to be converted. In another preferred variation, the URI may have a length limitation, and the URI preferably includes a fixed length cryptographic hash of the desired conversion including the full text. The full text is preferably included in the HTTP headers. The cryptographic hash functions as a unique persistent URI for each combination of text and conversion. The text is preferably the text that is to be converted into speech. The conversion information preferably includes voice type, speech speed, language, and/or any suitable parameters for the Text-To-Speech process. The cryptographic hash URI further functions to enable the cache server 41 to effectively cache the results of a Text-To-Speech conversion. In another variation, the cryptographic hash preferably encapsulates all conversion parameters, with their cleartext values provided in the HTTP headers of the request. The Text-To-Speech web service 44 generates the audio speech based on the conversion parameters from the call router 22, or more preferably from the web cache server 41 (with HTTP headers permitting caching). The cache server 41 preferably caches the audio for future reuse. In one variation, the audio resulting from the Text-To-Speech conversion is preferably streamed from the Text-To-Speech web service 44 back to the call router 22 through the cache server 41. Long audio files are preferably cached on the cache server 41 after being streamed the first time.

[0071] In one variation, the Text-To-Speech web service 44 preferably automates the selection and professional recording of frequently used words and phrases. Frequently used words and phrases are preferably defined to be words and phrases used more than a pre-specified frequency over a period of time (e.g. "Transferring your call" used one-hundred times per day). The frequently used words and phrases may be determined algorithmically (e.g., based on application history or system history) or alternatively may be specified by an application operator. Alternatively, high value words or phrases (e.g. phrases for marketing a brand) may be additionally or alternatively professionally recorded. The selected frequently used words and phrases are preferably transmitted to a remote site where they are professionally recorded by a voice actor in a studio. The transmission is preferably email, but may alternatively be an SMS message, a fax, a mailed document, or any other suitable transmission. The professional recording may be recorded word by word, phrase by phrase, with multiple intonations, or any other suitable recording methodology. The professional recording is preferably transmitted electronically to an application server 26, but may alternatively be transmitted to a Text-To-Speech conversion web service 44, a cache server 41, a transcoding proxy server 43, a call router 22 or any other suitable web server. The professional recordings are preferably transmitted in 8-bit PCM format, but may alternatively be transmitted in WAV, MP3, or any other suitable audio format. The return transmission of the professionally recorded audio files is preferably an HTTP post request, but may alternatively be an FTP transfer, an email, a mailed data storage device (e.g. CD ROM, memory card or Flash Disk) or any other suitable transmission.

[0072] The application server 26 functions to provide data processing logic for requests received from the call router 22. The application server 26 is preferably connected to the call router 22 via a network 24, more preferably via the Internet. The application server 26 is preferably a third party server operated outside of the system, but the system may alternatively include the application server 26. The URI 23 is preferably associated with an application server 26 or an application on an application server 26. The application server 26 preferably communicates with the call router 22 using an application layer protocol, more preferably using the HTTP protocol, or more secure HTTPS protocol. The application server 26 preferably receives HTTP requests from and sends HTTP responses to the call router 22. The application server 26 preferably runs on a standard stack of programming languages, hosting providers, operating systems and databases to handle HTTP requests, as if the caller were a website visitor in a web browser. The application server 26 also preferably verifies the digital signatures of the call data received in the requests using the secret key to compute a cryptographic hash from the received information and the hash received. If the computed hash and the received hash do not match, or no hash is received with the request, then the application server 26 preferably determines the request is fraudulent, and the request is preferably discarded. If the computed hash and received hash match, the application server 26 preferably determines that the request is authentic and proceeds further with the processing of the request. The application server may alternatively choose to ignore the hash if security is not important. The application server preferably uses call state data communicated by the call router request to determine the next call router instructions, without requiring call state stored on the application server. The application server may alternatively use call state data sent by the call router, such as the caller ID of the caller or the unique ID of the call, to reference additional or external state data, such as rows in a database or session data stored on the application server.

[0073] The application server 26 preferably responds to HTTP requests received from the call router 22 by generating telephony instructions 27 for the call router 22. The application server preferably replies to the call router in XML, however, any suitable machine-readable message format may be used, including HTML, key/value pair text, delimited text or binary encoding. The XML preferably includes the telephony instructions 27 for the call router 22 such as connecting to another number, playing a recorded greeting, reading text, and/or requesting DTMF digit entry from the caller. The telephony instruction 27 may alternatively be related to SMS messaging, Multimedia Messaging Service (MMS) messaging, email, or any suitable messaging task. The telephony instruction 27 may additionally be used to send an outgoing SMS message, arrange a phone call from a specific phone number, arranging for a callback, setting up a conference call (connecting multiple numbers), sending an email, interfacing with a calendar or scheduling system, purchasing goods, or services, or any other suitable instruction. The XML instructions are preferably a set of commands to be executed in order, one at a time (i.e., sequentially). An example XML response is shown in FIGURES 5A and 5B. In single telephony session (e.g. one initiated by a PSTN-device or an SMS device) a response from an application server can initiate an outgoing telephony call and/or a SMS message. That is, a single XML response preferably provides the ability to interact with both the SMS network and the voice telephony network (PSTN, SIP/VoIP, etc) sequentially or simultaneously. Media files may alternatively be sent from the application server 26. The application server 26 may respond to a request with an audio file, transmitting the audio file to the transcoding proxy server 242 for conversion into 8-bit PCM at 8kHz bandwidth suitable for telephony before sending the converted audio file, with its new mime-type header. The media file is then preferably sent to the cache server 241 for caching and forwarded to the call router 22. Preferably, the cache server 241 caches the most frequently used or all of the responses of the application server 26 to reduce the number of customized responses. This enables a more efficient use of both the computation and the transmission bandwidth to the applications server 26, effectively allowing more concurrent users to be served by a single application server. In one variation, the application server 26 may prime (or push) updated sound files to the media layer for caching. This priming is preferably done at off peak hours after new content is being generated or for periodic changes of an application (e.g. a new weather report). In addition, audio or video files sent to the call router 22 can be converted to text by an automatic speech-to-text engine, human or other technique, and sent back in text form as an SMS message or an attachment to an MMS. In one variation, an application running on a server may be a simple static XML page and static sound files, deployed on basic web servers where no development or scripting environment is available. This variation preferably uses URI Templates (a current IETF proposal for HTML5), which essentially includes URLs with placeholders for variable data, like this: http://www.twilio.com/audio/{Digit}.mp3 where the call router 22 would substitute the digits pressed for the {Digit} placeholder in the URI Template, GET the file at the resulting URI, and play the static sound file in response. This allows an entire application to be authored offline in a What-You-See-Is-What-You-Get (WYSIWYG) html editor. For example, if the server response specifies the URI

Template: http://demo.twilio.com/myapp/{Digits}.mp3, and the caller presses digits 1234, the call router 22 would GET the static mp3 file located at: http://demo.twilio.com/myapp/1234.mp3 and play it to the caller. The variables used for substitution in the URI Templates preferably correspond to the names of variables defined for state submission in HTTP GET, POST and/or header requests from the call router. From the previous example, {Digits} would be associated with a parameter named "Digits" that is preferably generated as a result of a "gather" telephony instruction (collection of DTMF digits). In the preferred embodiment for the second configuration, the call is initiated by the application server 26 (through the call router 22), and the second configuration 30 is substantially similar to the first configuration 20, such that the call routing is preferably handled identically to an incoming call, namely via URI requests from call router 22 to the server 26 upon call state changes. The application server preferably additionally is able to make calls to the Call Router API as described above.

[0074] As a person skilled in the art will recognize from the previous detailed description and from the figures and claims, modifications and changes can be made to the preferred embodiments of the invention without departing from the scope of this invention defined in the following claims. It is possible, and indeed hoped, that additional applications will be designed and built upon this technology platform (the preferred method and/or system of the invention) that would not otherwise be possible using conventional telephony platforms.

CLAIMS

We Claim:

1. A method of caching media for use in a telephony applications comprising:
    - receiving a media request;
    - sending the media request to a media layer using HTTP;
    - where the media layer performs the following steps:
        - o checking in a cache for a telephony compatible media resource specified by the media request;
        - o processing the media request within a media processing server to form a telephony compatible media resource; and
        - o storing the processed media in the cache as a telephony compatible media resource specified by persistent address.
2. The method of Claim 1, further comprising automatically allocating cache resources and media processing server resources of the media layer.
3. The method of Claim 1, further comprising mapping the media request to a persistent Universal Resource Identifier(URI), wherein a call router handling a telephony session between a caller and an application maps the media request to the URI and the telephony session initiates the media request, and wherein the persistent URI has embedded media parameters that uniquely identify contents of the media resource.
4. The method of Claim 1, wherein the media request is made through a media layer application programming interface (API).

5. The method of Claim 4, wherein media request is caching media for an application prior to a telephony session requesting the resource.

6. The method of Claim 1, where the step of processing the media request includes encrypting the telephony compatible media resource, and wherein the step of storing the processed media in the cache includes storing the encrypted media resource in a third party server.

7. The method of Claim 1, further comprising checking for a directive indicating to not cache the media and following the directive.

8. The method of Claim 1, wherein the step of processing the media request includes retrieving media from an external server and converting the retrieved media to a telephony compatible media format with a transcoding proxy server.

9. The method of Claim 8, wherein HTTPS is used for network communication between the media layer and the external server and HTTP is used for network communication within the media layer.

10. The method of Claim 1, wherein the step of processing media includes converting Text-To-Speech according to parameters of the media request.

11. The method of Claim 10, wherein the media parameters include voice parameters for the Text-To-Speech conversion.

12. The method of Claim 1, further comprising streaming the media from an external server through the media processor, through the cache, and to the call router.

13. The method of Claim 12, further comprising the call router writing the media resource to disk prior to using the resource.

14. The method of Claim 3, further comprising:

- using HTTP communication protocol for sending the media request wherein the HTTP message includes the text to be converted to speech;
- if a telephony compatible media resource is identified in the cache by the persistent URI, forwarding the cached media resource to the call router;
- if a telephony compatible media resource is not identified in the cache by the persistent URI, forwarding the media request to a Text-To-Speech server;
- caching speech audio generated by the Text-To-Speech server according to cache directives; and
- returning the speech audio to the call router.

15. The method of Claim 14, further comprising forwarding the speech audio to a transcoding proxy server and converting the speech audio to a telephony compatible media format prior to caching the speech audio.

16. The method of Claim 3, further comprising:

- if a telephony compatible media resource is identified in the cache by the persistent URI, forwarding the cached media resource to the call router;
- if a telephony compatible media resource is not identified in the cache by the persistent URI, retrieving the media from the external server, wherein HTTPS is used for network communication between the media layer and the external server and HTTP is used for network communication within the media layer and between the call router and the media layer;

- if the media requires conversion, converting the media to a telephony compatible media format with the transcoding proxy server;
- caching the telephony compatible media according to cache directives; and
- returning the telephony compatible media to the call router.

17. A system for caching media used in a telephony applications comprising:
    - a call router that receives media requests, and handles application communication between a telephony device and an application server;
    - a media layer that is an intermediary layer between the call router and application resources, that uses an HTTP messaging protocol, the media layer including:
        - a cache server that stores telephony compatible media; and
        - a media processing server that processes the media request to create telephony compatible media.

18. The system of Claim 17, further comprising a Media Layer API that provides programmatic access of the media layer from an external service.

19. The system of Claim 17, wherein the media processing server is a Text-To-Speech server that converts text to speech audio.

20. The system of Claim 17, wherein the media processing server is a transcoding proxy server that converts between media types.

21. The system of Claim 17, wherein the media processing server includes a Text-To-Speech server that converts text to an audio recording and a transcoding proxy server that converts media to a telephony compatible format.

22. The system of Claim 21, wherein the media layer independently scales cache servers, Text-To-Speech servers, and transcoding proxy servers, and the media layer includes a load balancer that automatically manages the capacity of the servers of the media layer.

23. The system of Claim 22, wherein HTTPS is used for network communication between the media layer and the application server and HTTP is used for network communication within the media layer and between the call router and the media layer.

24. The system of Claim 23, wherein the cache stores media in an external server.

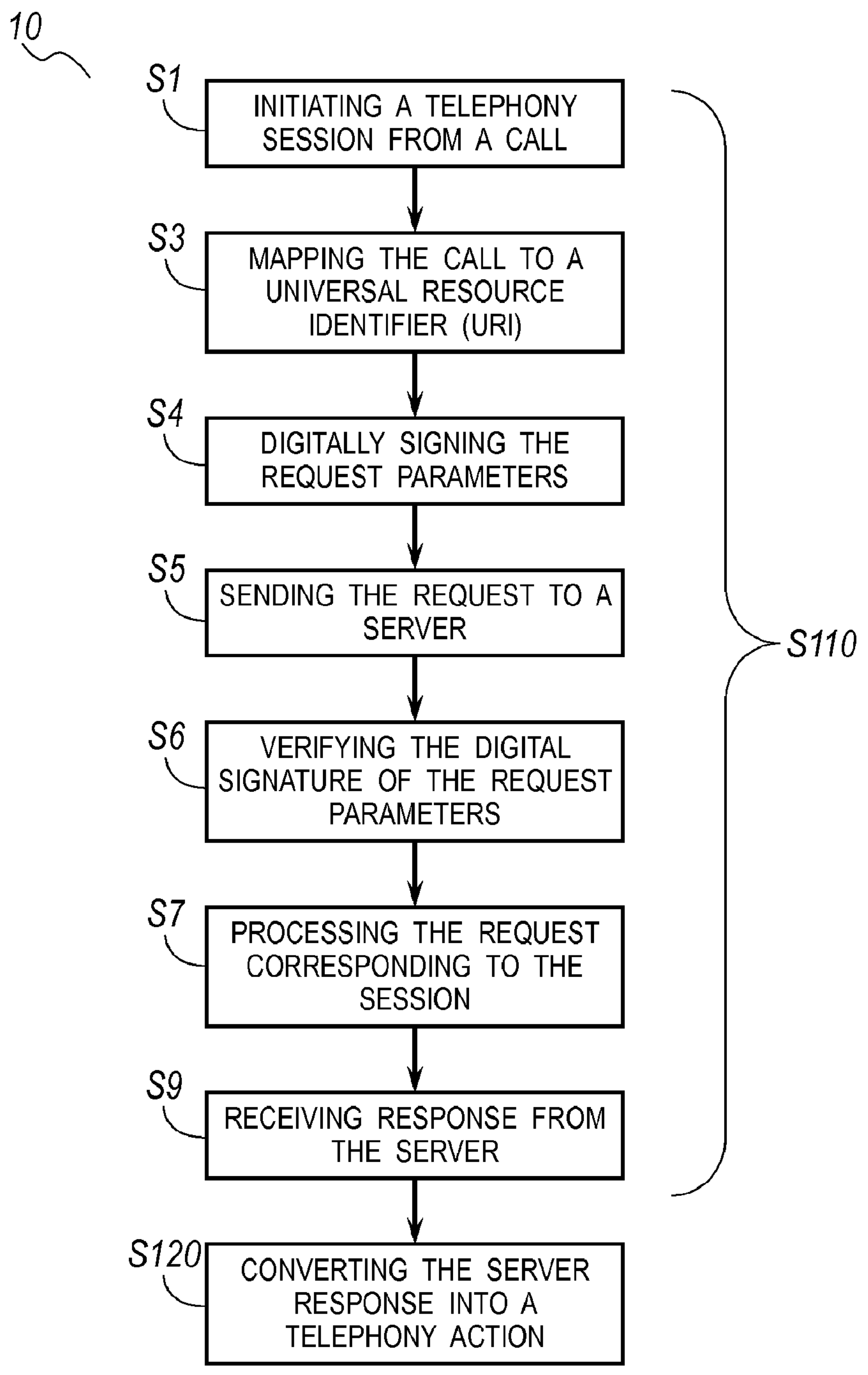
10
S1
INITIATING A TELEPHONY SESSION FROM A CALL
S3
MAPPING THE CALL TO A UNIVERSAL RESOURCE IDENTIFIER (URI)
S4
DIGITALLY SIGNING THE REQUEST PARAMETERS
S5
SENDING THE REQUEST TO A SERVER
S6
VERIFYING THE DIGITAL SIGNATURE OF THE REQUEST PARAMETERS
S7
PROCESSING THE REQUEST CORRESPONDING TO THE SESSION
S9
RECEIVING RESPONSE FROM THE SERVER
S110
S120
CONVERTING THE SERVER RESPONSE INTO A TELEPHONY ACTION

FIG. 1

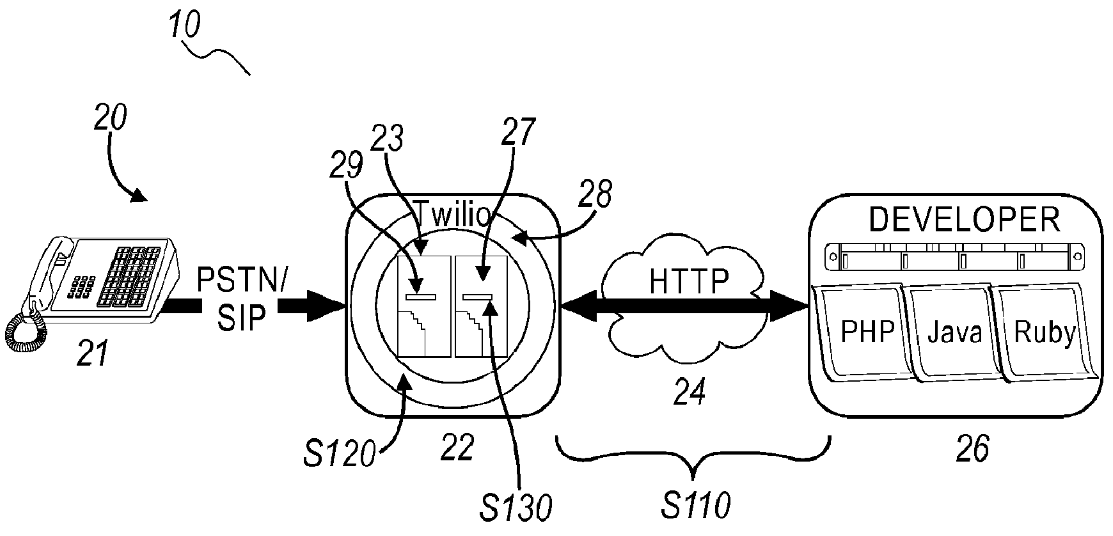
10
20
21
PSTN/
SIP
29
23
27
28
Twilio
S120
22
S130
HTTP
24
S110
DEVELOPER
PHP
Java
Ruby
26

FIG. 2A

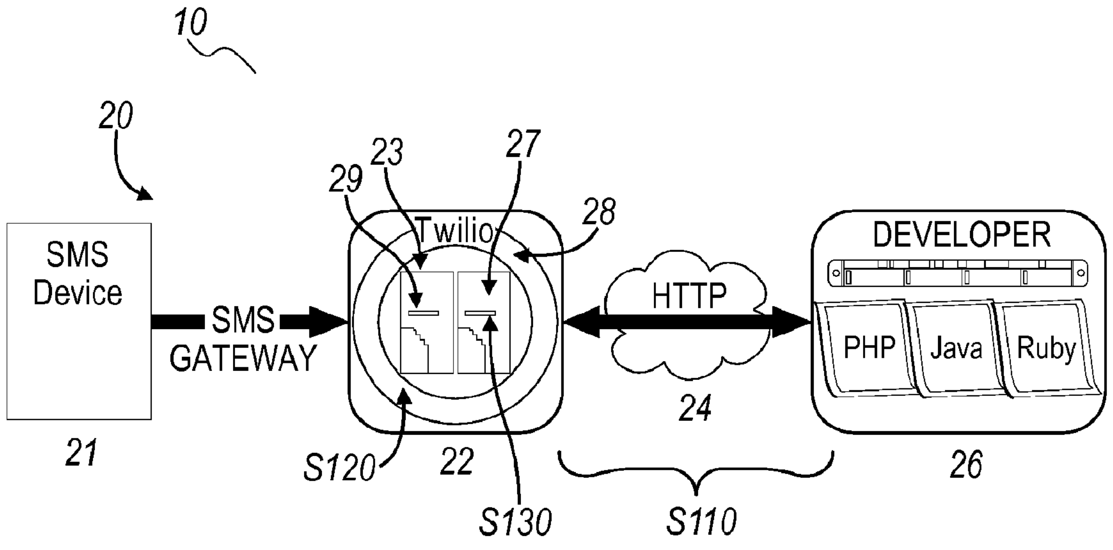
10
20
SMS
Device
21
SMS
GATEWAY
29
23
27
28
Twilio
S120
22
S130
HTTP
24
S110
DEVELOPER
PHP
Java
Ruby
26

FIG. 2B

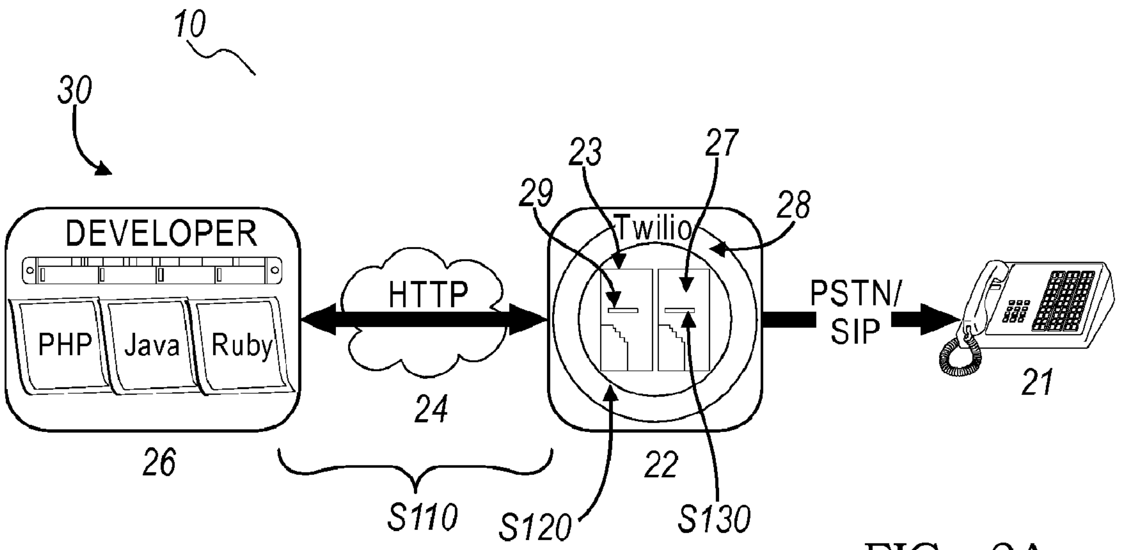
10
30
DEVELOPER
PHP
Java
Ruby
26
HTTP
24
S110
23
29
27
28
Twilio
22
S120
S130
PSTN/
SIP
21

FIG. 3A

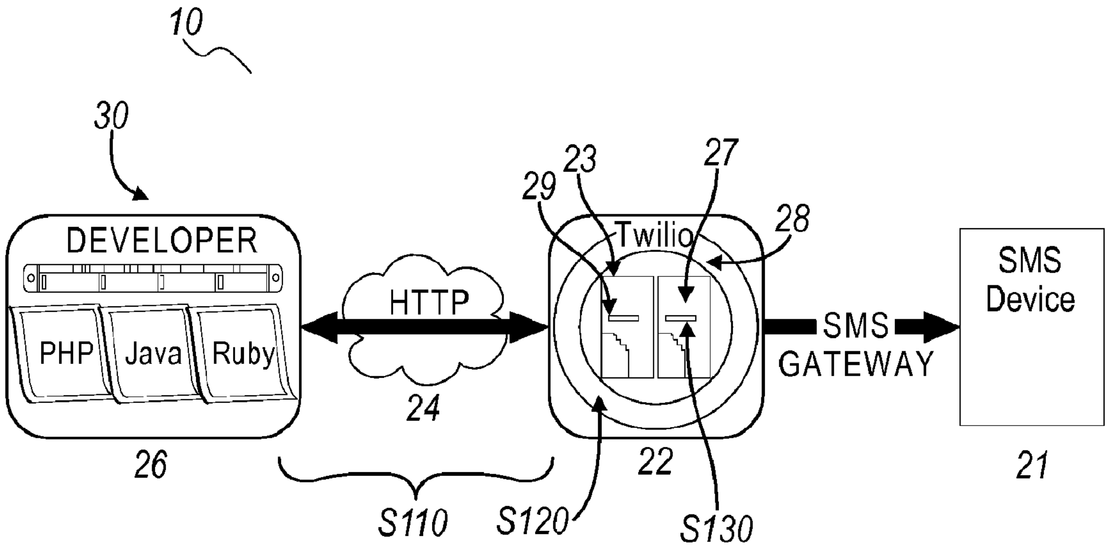
10
30
DEVELOPER
PHP
Java
Ruby
26
HTTP
24
S110
23
29
27
28
Twilio
22
S120
S130
SMS
GATEWAY
SMS
Device
21

FIG. 3B

```
GET /foo.php  HTTP/1.1
Host:  demo.twilio.com
X-Twilio-CallGuid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-CallerId=415-555-1212
X-Twilio-NumberCalled=415-867-5309
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length:  0
```

FIG. 4A

```
POST /foo.php  HTTP/1.1
Host:  demo.twilio.com
Content-Type:  application/x-www-form-urlencoded
X-Twilio-CallGuid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-CallerId=415-555-1212
X-Twilio-NumberCalled=415-867-5309
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length:  11

Digits=1234
```

FIG. 4B

```
GET /foo.php?digits=1234  HTTP/1.1
Host:  demo.twilio.com
X-Twilio-CallGuid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-CallerId=415-555-1212
X-Twilio-NumberCalled=415-867-5309
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length:  0
```

FIG. 4C

```
GET /foo.php  HTTP/1.1
Host:  demo.twilio.com
X-Twilio-SMSid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-SMSSenderID=415-555-1234
X-Twilio-SMSShortCode=11111
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length:  0
```

FIG. 4D

```
GET /foo.php  HTTP/1.1
Host:  demo.twilio.com
X-Twilio-SMSid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-SMSSenderId=415-555-1234
X-Twilio-SMSShortCode=11111
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length:  21

Message=statusrequest
```

FIG. 4E

```
GET /foo.php?message=statusrequest  HTTP/1.1
Host:  demo.twilio.com
X-Twilio-SMSid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-SMSSenderId=415-555-1234
X-Twilio-SMSShortCode=11111
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length:  0
```

FIG. 4F

XML Response

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
          <Collect
                    successUrl="http://www.example.com/phonetree.php"
                    numDigits="1"
                    timeout=20
          >
                    <Say voice="female">
For sales press one. For support press two. For the operator, press three.
                    </Say>
          </Collect>
</Response>
```

FIG. 5A

XML Response

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
          <sms address=415-555-555
                    thanks for the your text, will call at 5 PM.
          </sms>
          <CallAtTime="17:00PST">
                    Today
                    <Dial>415-555-5309</Dial>

          </CallAtTime>
</Response>
```

FIG. 5B

```
POST /2008-08-01/Accounts/AC309475e5fede1b49e100272a8640f438/Calls  HTTP/1.1
        Caller=4155555309&Called=4155551212&Url=http://www.myapp.com/myhandler.php

<TwilioResponse>
        <Call>
                <Sid>CA42ed11f93dc08b952027ffbc406d0868</Sid>
                <CallSegmentSid/>
                <AccountSid>AC309475e5fede1b49e100272a8640f438</AccountSid>
                <Called>4155551212</Called>
                <Caller>4155555309</Caller>
                <PhoneNumberSid>PN01234567890123456789 00<PhoneNumberSid>
                <Status>0</Status>
                <StartTime>Thu, 03 Apr 2008 04:36:33  -0400</StartTime>
                <EndTime/>
                <Price/>
                <Flags>1</Flags>
        </Call>
</TwilioResponse>
```

FIG. 6

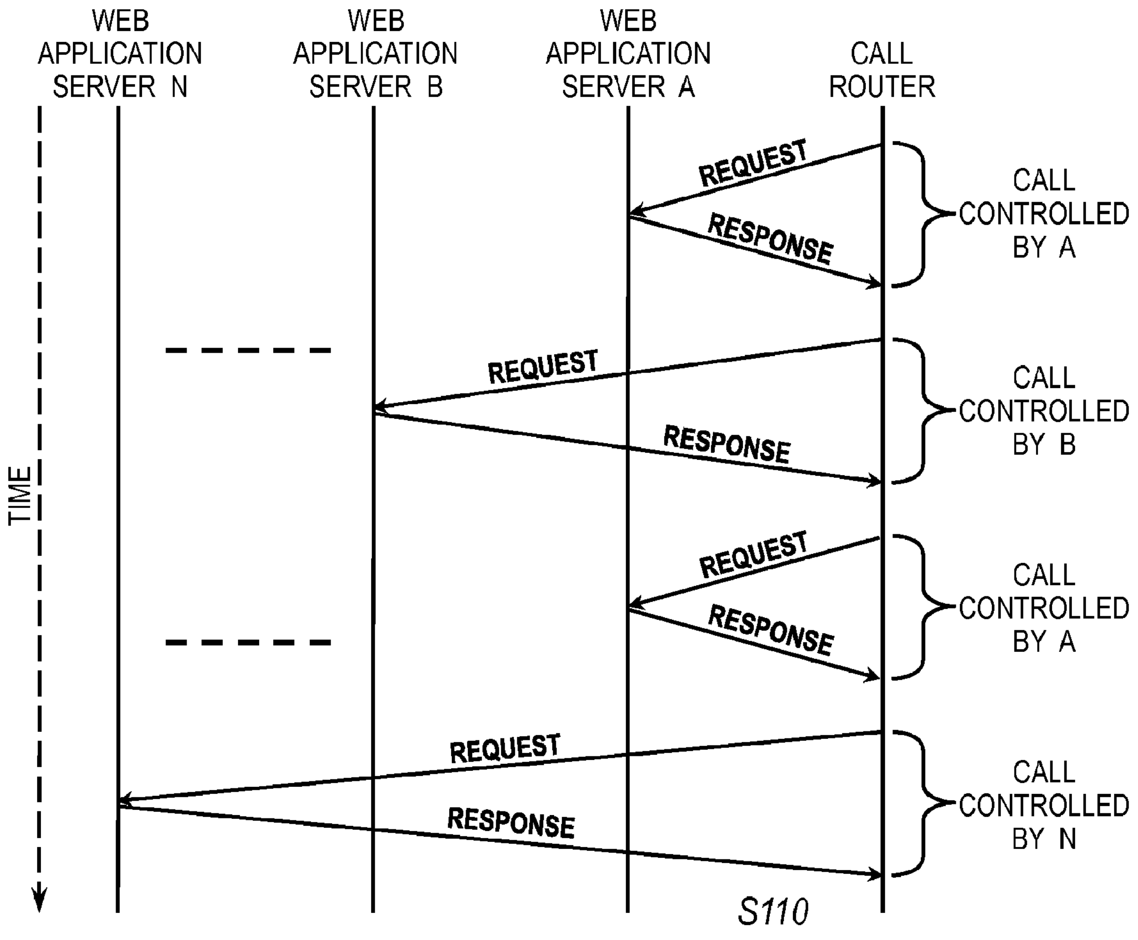
WEB APPLICATION SERVER N
WEB APPLICATION SERVER B
WEB APPLICATION SERVER A
CALL ROUTER
TIME
REQUEST
RESPONSE
CALL CONTROLLED BY A
REQUEST
RESPONSE
CALL CONTROLLED BY B
REQUEST
RESPONSE
CALL CONTROLLED BY A
REQUEST
RESPONSE
CALL CONTROLLED BY N
S110

FIG. 7

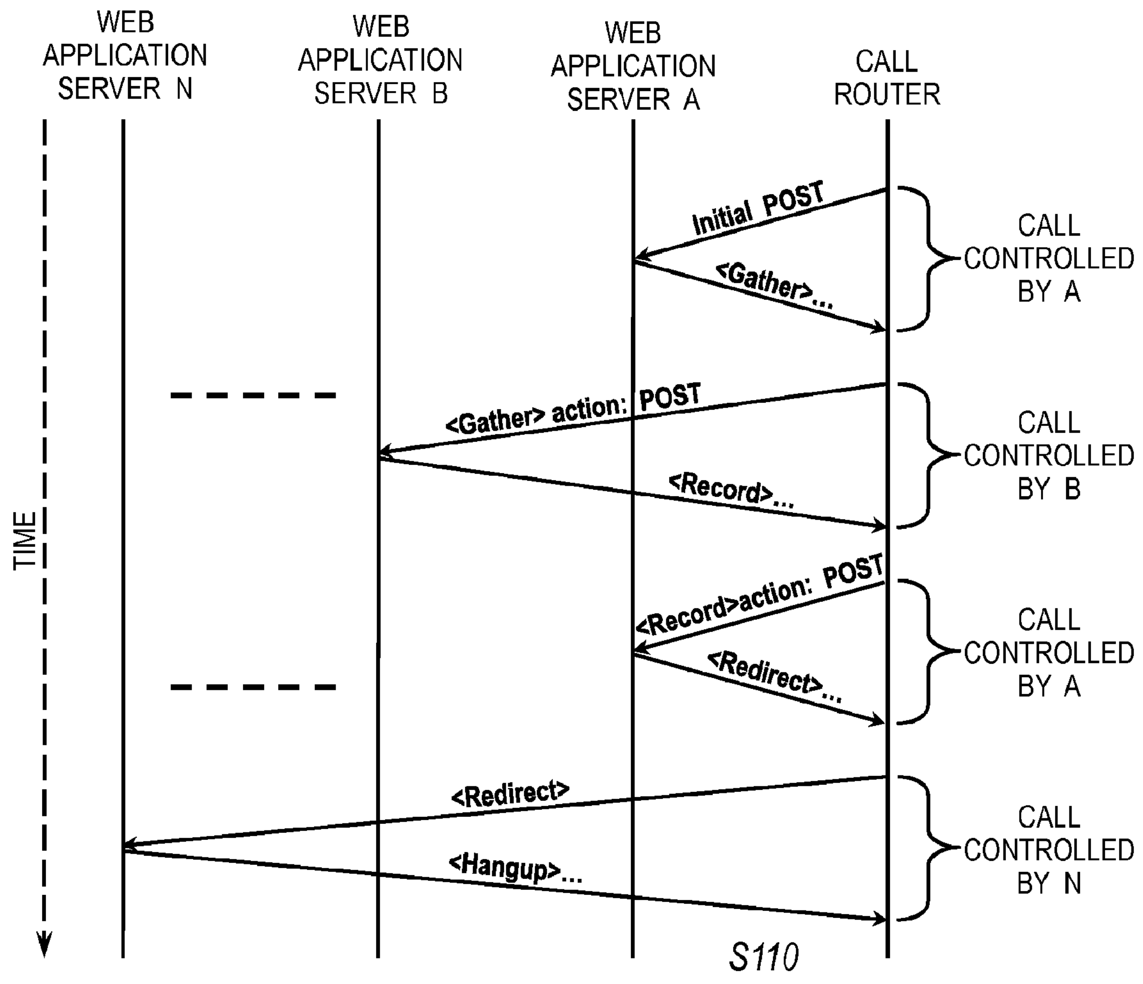
WEB APPLICATION SERVER N
WEB APPLICATION SERVER B
WEB APPLICATION SERVER A
CALL ROUTER
TIME
Initial POST
<Gather>...
CALL CONTROLLED BY A
<Gather> action: POST
<Record>...
CALL CONTROLLED BY B
<Record>action: POST
<Redirect>...
CALL CONTROLLED BY A
<Redirect>
<Hangup>...
CALL CONTROLLED BY N
S110

FIG. 8

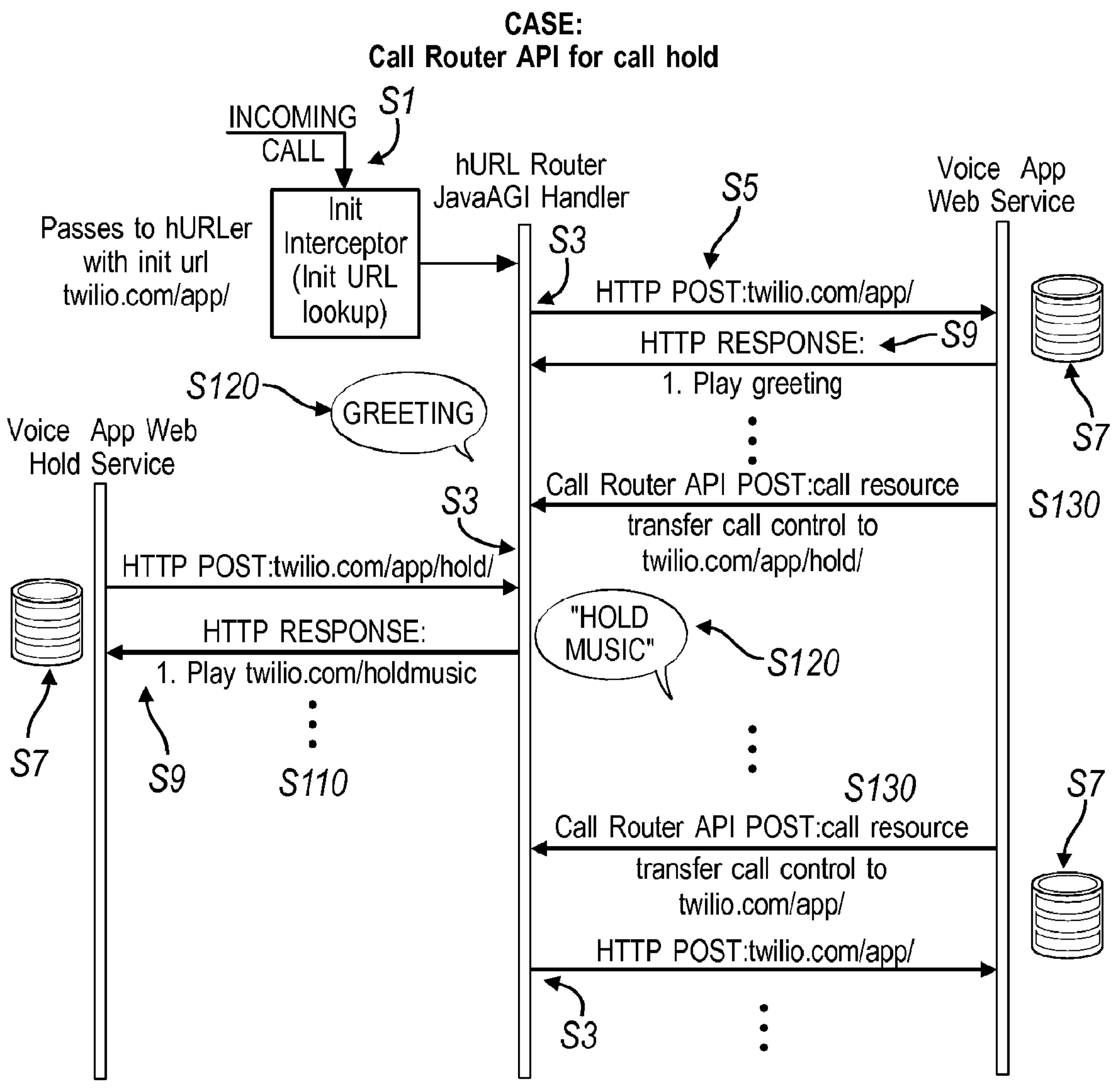
CASE:
Call Router API for call hold
INCOMING CALL
S1
Init Interceptor (Init URL lookup)
Passes to hURLer with init url twilio.com/app/
hURL Router JavaAGI Handler
Voice App Web Service
S5
S3
HTTP POST:twilio.com/app/
HTTP RESPONSE:
S9
1. Play greeting
S7
S120
GREETING
Voice App Web Hold Service
Call Router API POST:call resource
S130
transfer call control to twilio.com/app/hold/
S3
HTTP POST:twilio.com/app/hold/
HTTP RESPONSE:
1. Play twilio.com/holdmusic
"HOLD MUSIC"
S120
S7
S9
S110
S130
S7
Call Router API POST:call resource
transfer call control to twilio.com/app/
HTTP POST:twilio.com/app/
S3

FIG. 9

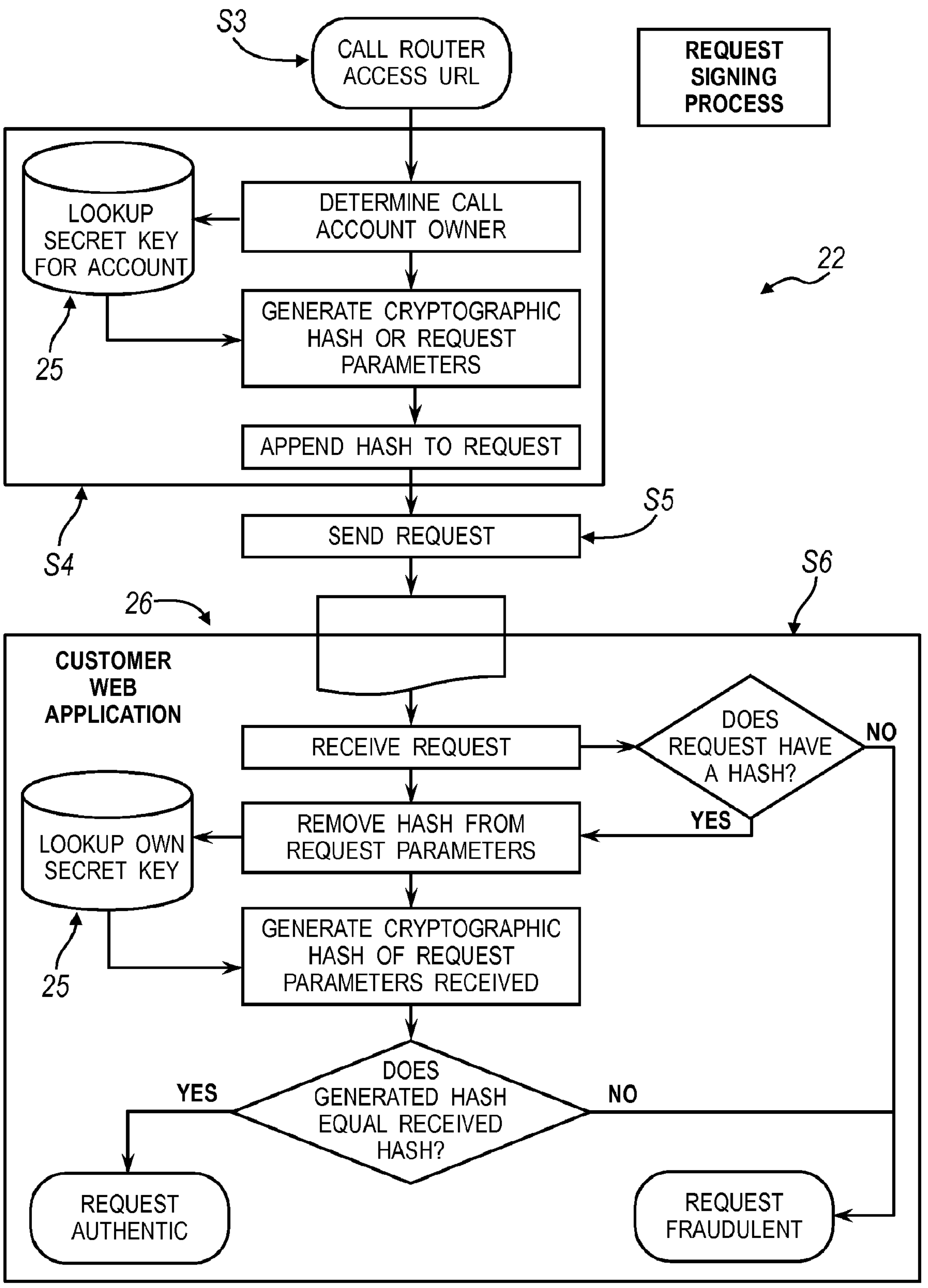
S3
CALL ROUTER ACCESS URL
REQUEST SIGNING PROCESS
LOOKUP SECRET KEY FOR ACCOUNT
DETERMINE CALL ACCOUNT OWNER
22
GENERATE CRYPTOGRAPHIC HASH OR REQUEST PARAMETERS
25
APPEND HASH TO REQUEST
S5
SEND REQUEST
S4
26
S6
CUSTOMER WEB APPLICATION
RECEIVE REQUEST
DOES REQUEST HAVE A HASH?
NO
YES
LOOKUP OWN SECRET KEY
REMOVE HASH FROM REQUEST PARAMETERS
GENERATE CRYPTOGRAPHIC HASH OF REQUEST PARAMETERS RECEIVED
25
YES
DOES GENERATED HASH EQUAL RECEIVED HASH?
NO
REQUEST AUTHENTIC
REQUEST FRAUDULENT

FIG. 10

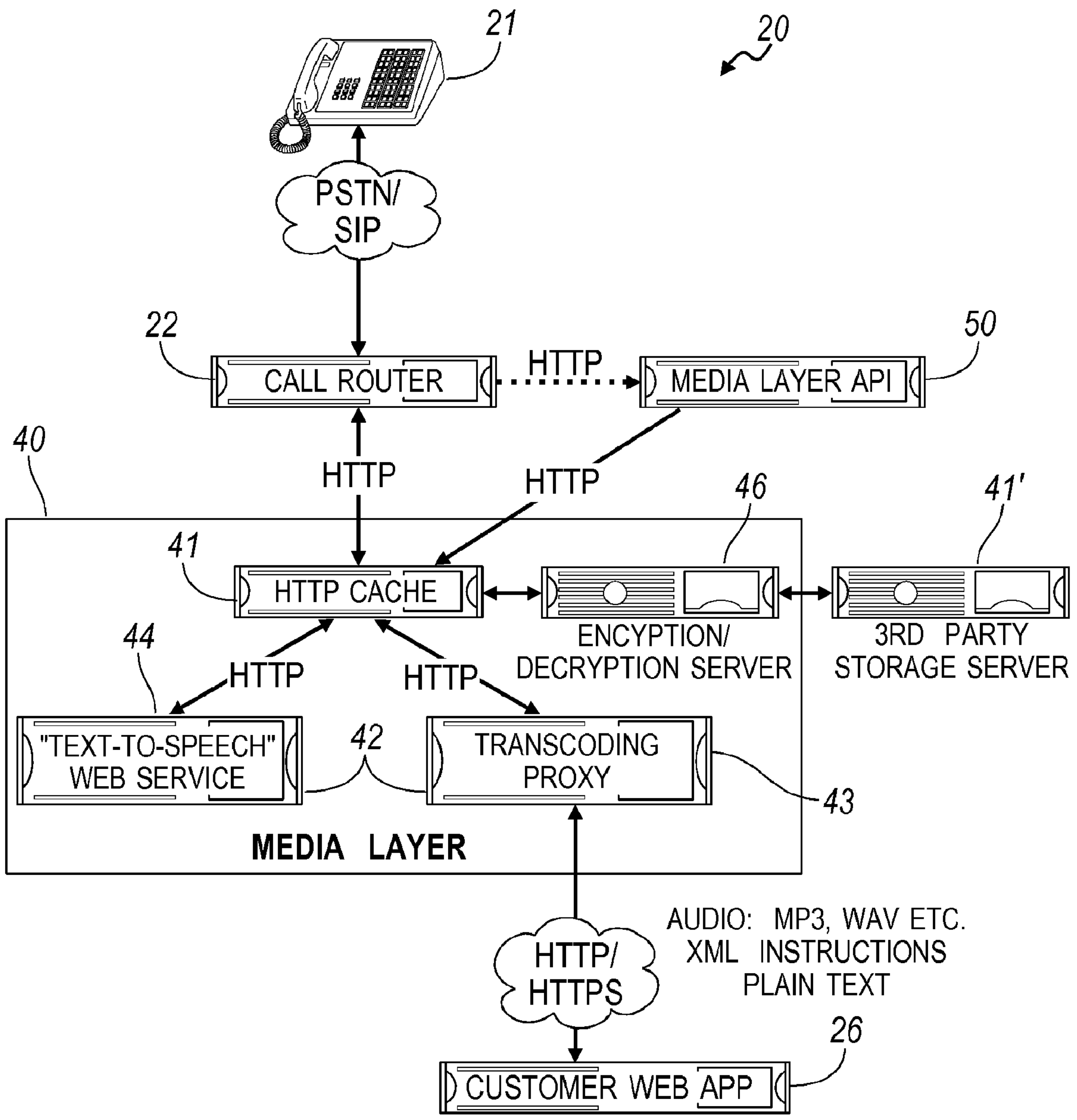
21
20
PSTN/ SIP
22
CALL ROUTER
HTTP
MEDIA LAYER API
50
40
HTTP
HTTP
46
41'
41
HTTP CACHE
ENCYPTION/ DECRYPTION SERVER
3RD PARTY STORAGE SERVER
44
HTTP
HTTP
"TEXT-TO-SPEECH" WEB SERVICE
42
TRANSCODING PROXY
43
MEDIA LAYER
HTTP/ HTTPS
AUDIO: MP3, WAV ETC.
XML INSTRUCTIONS
PLAIN TEXT
26
CUSTOMER WEB APP

FIG. 11

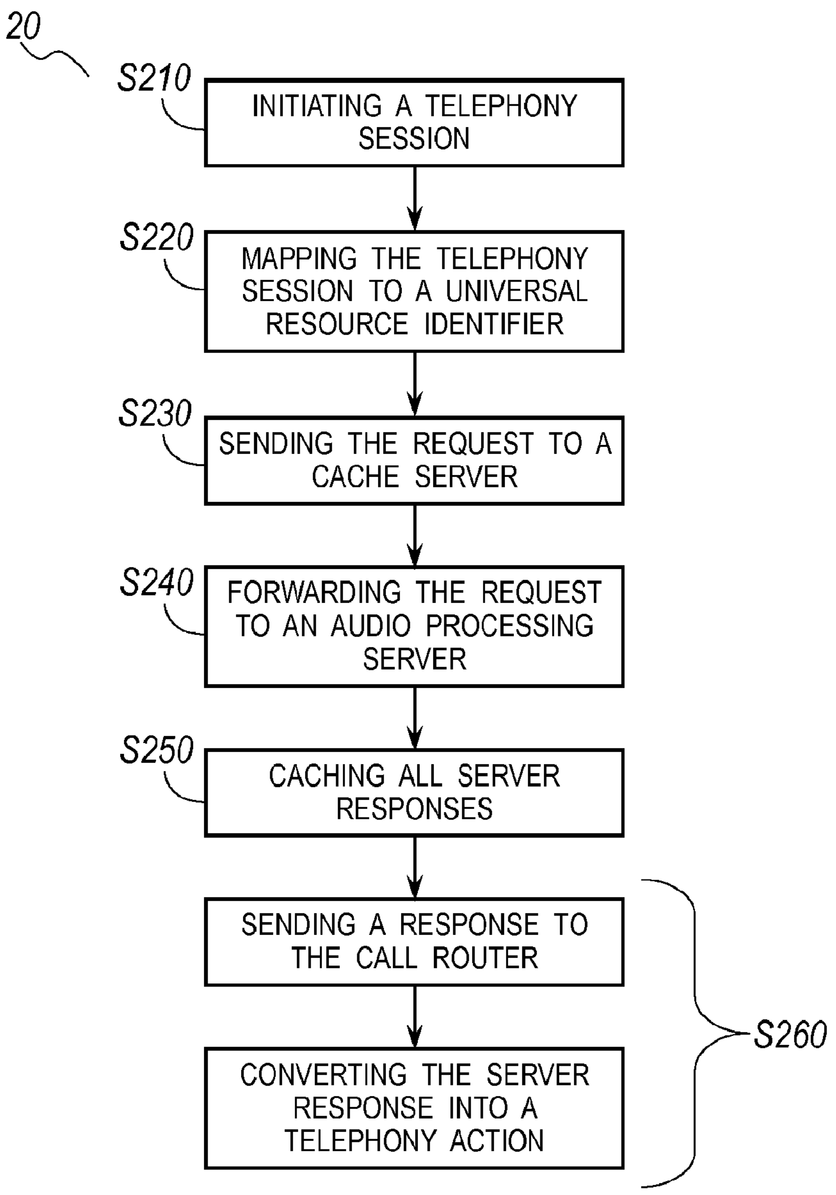
20
S210
INITIATING A TELEPHONY SESSION
S220
MAPPING THE TELEPHONY SESSION TO A UNIVERSAL RESOURCE IDENTIFIER
S230
SENDING THE REQUEST TO A CACHE SERVER
S240
FORWARDING THE REQUEST TO AN AUDIO PROCESSING SERVER
S250
CACHING ALL SERVER RESPONSES
SENDING A RESPONSE TO THE CALL ROUTER
CONVERTING THE SERVER RESPONSE INTO A TELEPHONY ACTION
S260

FIG. 12

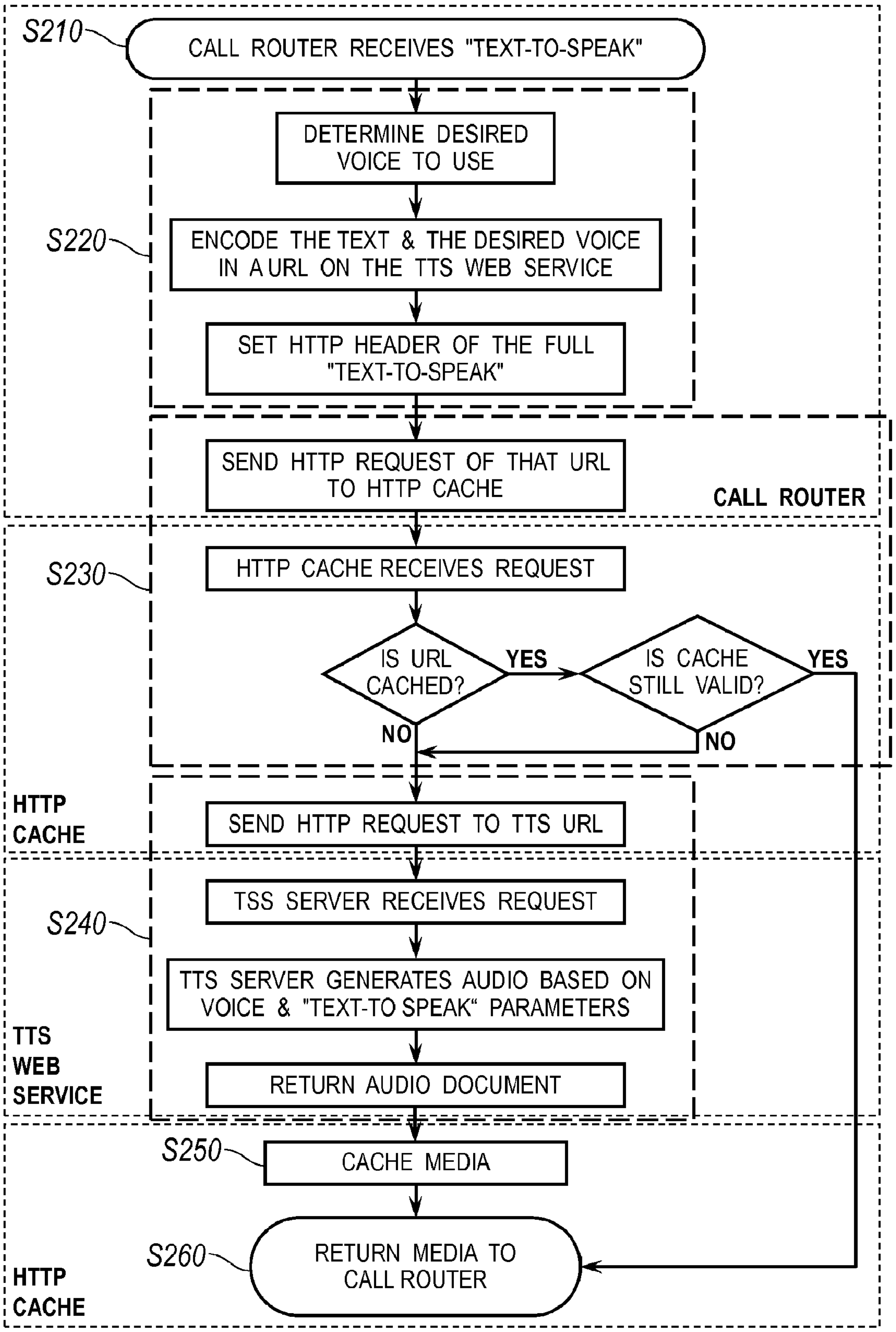
S210
CALL ROUTER RECEIVES "TEXT-TO-SPEAK"
DETERMINE DESIRED VOICE TO USE
S220
ENCODE THE TEXT & THE DESIRED VOICE IN A URL ON THE TTS WEB SERVICE
SET HTTP HEADER OF THE FULL "TEXT-TO-SPEAK"
SEND HTTP REQUEST OF THAT URL TO HTTP CACHE
CALL ROUTER
S230
HTTP CACHE RECEIVES REQUEST
IS URL CACHED?
YES
IS CACHE STILL VALID?
YES
NO
NO
HTTP CACHE
SEND HTTP REQUEST TO TTS URL
TSS SERVER RECEIVES REQUEST
S240
TTS SERVER GENERATES AUDIO BASED ON VOICE & "TEXT-TO SPEAK" PARAMETERS
TTS WEB SERVICE
RETURN AUDIO DOCUMENT
S250
CACHE MEDIA
S260
RETURN MEDIA TO CALL ROUTER
HTTP CACHE

FIG. 13

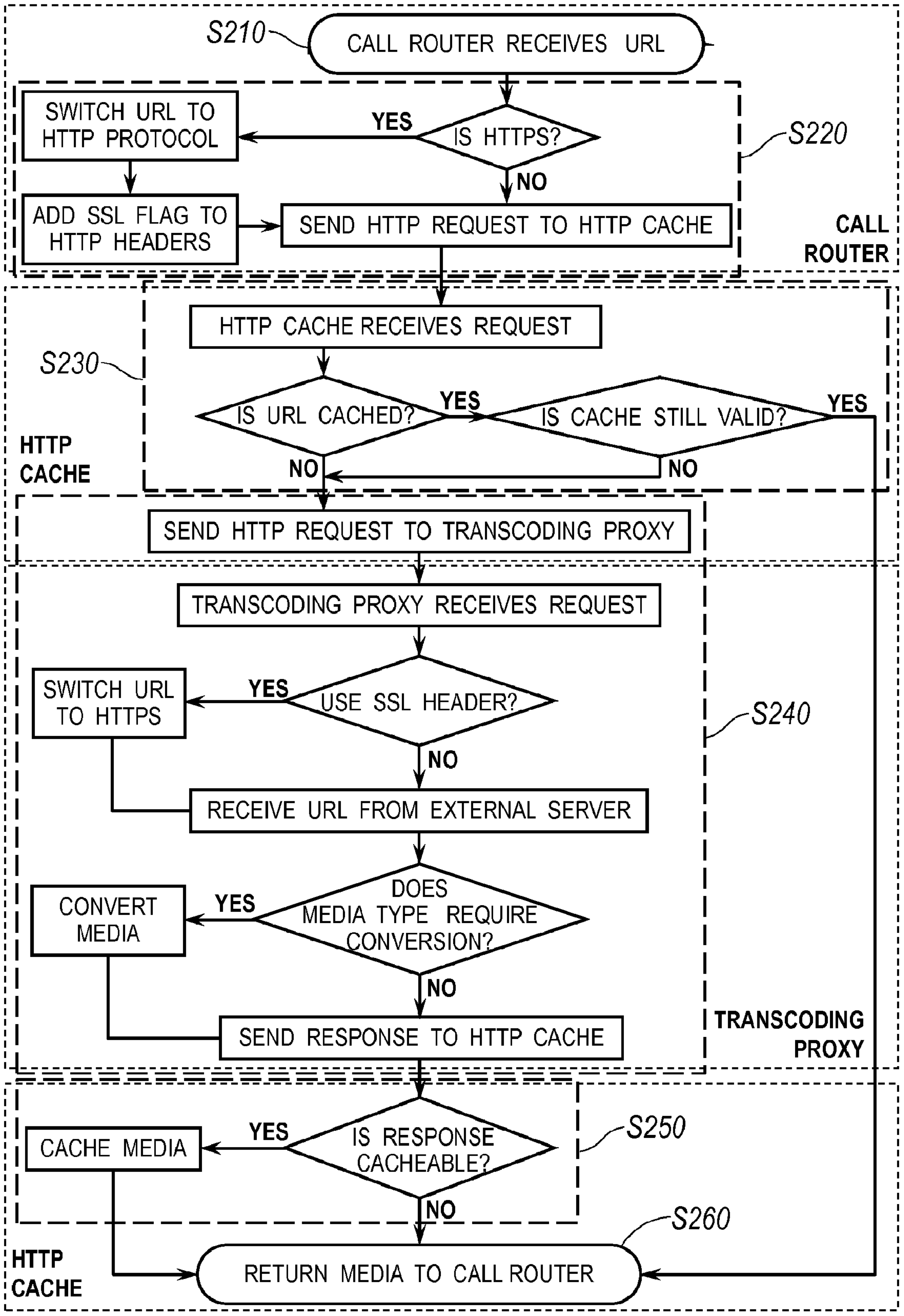
S210
CALL ROUTER RECEIVES URL
IS HTTPS?
YES
SWITCH URL TO HTTP PROTOCOL
NO
ADD SSL FLAG TO HTTP HEADERS
SEND HTTP REQUEST TO HTTP CACHE
S220
CALL ROUTER
HTTP CACHE RECEIVES REQUEST
S230
IS URL CACHED?
YES
IS CACHE STILL VALID?
YES
NO
NO
HTTP CACHE
SEND HTTP REQUEST TO TRANSCODING PROXY
TRANSCODING PROXY RECEIVES REQUEST
USE SSL HEADER?
YES
SWITCH URL TO HTTPS
NO
S240
RECEIVE URL FROM EXTERNAL SERVER
DOES MEDIA TYPE REQUIRE CONVERSION?
YES
CONVERT MEDIA
NO
SEND RESPONSE TO HTTP CACHE
TRANSCODING PROXY
IS RESPONSE CACHEABLE?
YES
CACHE MEDIA
NO
S250
S260
RETURN MEDIA TO CALL ROUTER
HTTP CACHE

FIG. 14

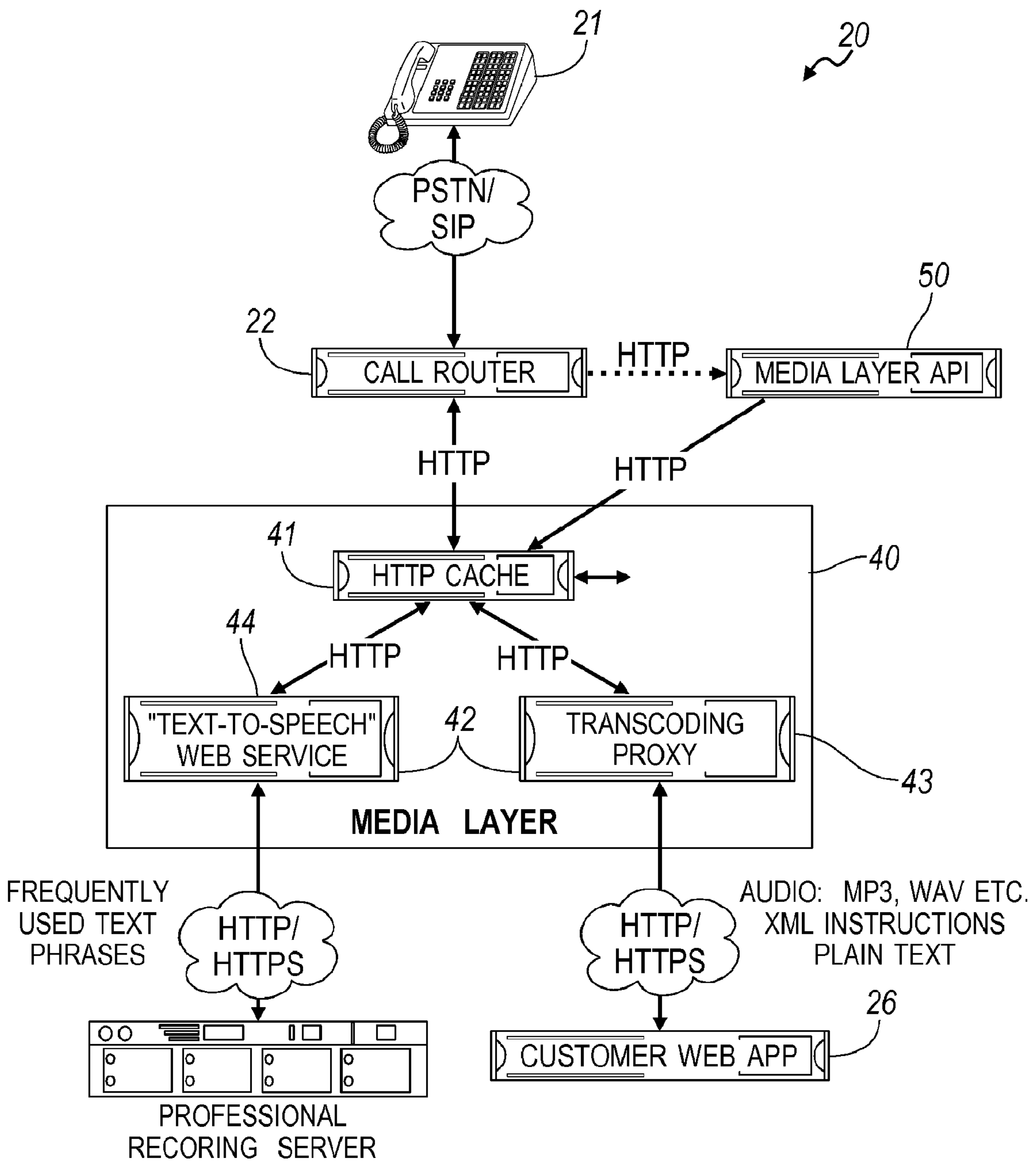
21
20
PSTN/ SIP
50
22
CALL ROUTER
HTTP
MEDIA LAYER API
HTTP
HTTP
41
HTTP CACHE
40
44
HTTP
HTTP
"TEXT-TO-SPEECH" WEB SERVICE
42
TRANSCODING PROXY
43
MEDIA LAYER
FREQUENTLY USED TEXT PHRASES
HTTP/ HTTPS
HTTP/ HTTPS
AUDIO: MP3, WAV ETC. XML INSTRUCTIONS PLAIN TEXT
26
CUSTOMER WEB APP
PROFESSIONAL RECORING SERVER

FIG. 15

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 09/58671

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - H04M 7/00 (2009.01)
USPC - 379/221.06
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)
USPC: 379/221.06

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
USPC: 379/219, 221.06

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
PubWEST(PGPB, USPT, EPAB, JPAB); Google
Search Terms Used: telephony, media, cache, process, request, application, multimedia, cellular, mobile, phone, text, speech, voice, http, URI, universal, resouce, identifier, convert

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2005/0240659 A1 (TAYLOR) 27 October 2005 (27.10.2005), entire document, especially para [0018], [0043], [0050]-[0058], [0062]-[0065], [0069]-[0070]	1-24
A	US 2003/0059020 A1 (MEYERSON et al.) 27 March 2003 (27.03.2003), entire document	1-24
A	US 2007/0121651 A1 (CASEY et al.) 31 May 2007 (31.05.2007), entire document	1-24
A	US 2004/0101122 A1 (DA PALMA et al.) 27 May 2004 (27.05.2004), entire document	1-24

☐ Further documents are listed in the continuation of Box C. ☐

* Special categories of cited documents:	"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
"A" document defining the general state of the art which is not considered to be of particular relevance	"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
"E" earlier application or patent but published on or after the international filing date	"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
"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)	"&" document member of the same patent family
"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	

Date of the actual completion of the international search 18 December 2009 (18.12.2009)	Date of mailing of the international search report 30 DEC 2009
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Authorized officer: Lee W. Young PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (July 2009)

WO2010040010

Publication Title:

TELEPHONY WEB EVENT SYSTEM AND METHOD

Abstract:

Abstract of WO 2010040010

(A1) Translate this text An embodiment of the system for publishing events of a telephony application to a client includes a call router that generates events from the telephony application and an event router that manages the publication of events generated by the call router and that manages the subscription to events by clients. The system can be used with a telephony application that interfaces with a telephony device and an application server

----
Courtesy of http://v3.espacenet.com

This Patent PDF Generated by Patent Fetcher(R), a service of Stroke of Color, Inc.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 April 2010 (08.04.2010)

PCT

(10) International Publication Number
WO 2010/040010 A1

(51) International Patent Classification:
H04M 3/42 (2006.01)

(21) International Application Number:
PCT/US2009/059300

(22) International Filing Date:
1 October 2009 (01.10.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/102,007 1 October 2008 (01.10.2008) US

(71) Applicant *(for all designated States except US)*: **TWILIO INC** [US/US]; 548 Market St# 14510, San Francisco, CA 94101 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **LAWSON, Jeffrey** [US/US]; 548 Market St #14510, San Francisco, CA 94104 (US). **WOLTHUIS, John** [US/—]; 548 Market St #14510, San Francisco, CA 94104 (US). **COOKE, Evan** [US/US]; 548 Market St #14510, San Francisco, CA 94104 (US).

(74) Agent: **SCHOX, Jeffrey**; 500 3rd Street #515, San Fransisco, CA 94107 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report (Art. 21(3))*

(54) Title: TELEPHONY WEB EVENT SYSTEM AND METHOD

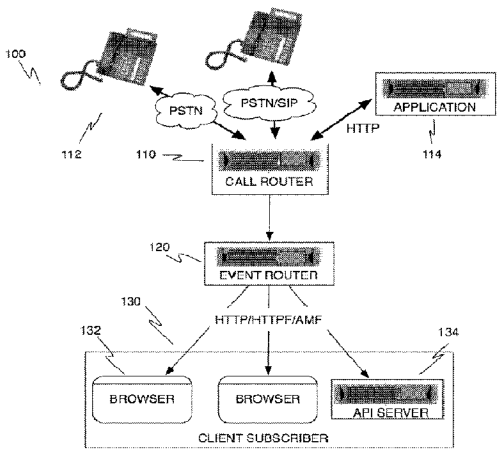

FIGURE 1

(57) Abstract: An embodiment of the system for publishing events of a telephony application to a client includes a call router that generates events from the telephony application and an event router that manages the publication of events generated by the call router and that manages the subscription to events by clients. The system can be used with a telephony application that interfaces with a telephony device and an application server

WO 2010/040010 A1

TELEPHONY WEB EVENT SYSTEM AND METHOD

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application also claims the benefit of the following: US Provisional Application number 61/102,007 filed on 01 October 2008 and entitled "Telephony Web Event System and Method", which is hereby incorporated in its entirety by this reference.

TECHNICAL FIELD

[0002] This invention relates generally to the event notification field, and more specifically to an new and useful system and method in the telephony web event field.

BACKGROUND

[0003] In recent years, there has been a growing trend of both internet-enabled phones and "websites as software". These two markets thrive on the transfer of information, instantaneous communication, and interaction between remote devices. However, current systems do not provide a seamless integration of telephony actions with remotely hosted applications, with server-side components, or with front-end websites. For instance, it is currently extremely difficult (if not impossible) to relay events that happen during a telephony call to or from a website securely in real-time during the telephony call. In addition, separation of business logic from telephony components complicates the transfer of realtime events from the call infrastructure to other remote services. Thus, there is a need in the telephony field to create a new and useful telephony web event system and method. This invention provides such a new and useful system and method.

BRIEF DESCRIPTION OF THE FIGURES

[0004] FIGURE 1 is a schematic diagram of the preferred embodiment of the invention.

[0005] FIGURE 2 is a detailed schematic diagram of the preferred embodiment of the invention.

[0006] FIGURE 3 is a flowchart diagram of a preferred method of event subscription for telephony applications.

[0007] FIGURE 4 is a flowchart diagram of a preferred method of distributing events.

[0008] FIGURE 5 is a flowchart diagram of a preferred method of subscribing to events.

[0009] FIGURE 6 is a flowchart diagram of a preferred method of subscribing and publishing events.

[0010] FIGURE 7 is a flowchart diagram of a preferred method full duplex publishing and subscribing of events.

[0011] FIGURES 8A – 8C are examples of a HTTP GET request, a HTTP POST request, and a HTTP GET request, respectively.

[0012] FIGURES 8D – 8F are examples of a HTTP requests.

[0013] FIGURES 9A and 9B are examples of XML responses.

[0014] FIGURE 10 an example of subscription aggregation.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0015] The following description of the preferred embodiments of the invention is not intended to limit the invention to these preferred embodiments, but rather to enable any person skilled in the art to make and use this invention.

1. Telephony web event system

[0016] As shown in FIGURES 1 and 2, the telephony web event system 100 of the preferred embodiment includes a call router 110 and an event router 120. The system functions to enable real-time telephony event interaction. In the preferred system, telephony events (e.g., a dialing sequence, a speech command, and a phone call termination) are sent out by a publisher (a device that prepares and electronically announces the occurrence of an event, preferably a call router) and received by a subscriber. A subscriber is preferably any client 130, such as a web browser 132 or Application Programming Interface (API) server 134 that has permission to receive information concerning a particular event. The system 100 is preferably implemented on a multitennant system where a plurality of applications and users are handled on the same software or hardware system. In particular, the call router can preferably simultaneously manage a plurality of telephony application and the event router can publish a plurality of events and manage a plurality of subscriptions for a plurality of clients. The components of the system are preferably scalable with respect to the number of accounts, in-progress calls, events on those calls, and the number of client subscriptions. Call routers 110, event routers 120, or any sub-elements (such as event proxy servers 122 or message brokers 128) may be allocated or deallocated to automatically adjust for capacity requirements. A load balancer 121 may additionally be included to optimize the operation of the system components.

[0017] The call router 110 of the preferred embodiment functions to initiate the publication of events occurring during a telephony application. The telephony application is preferably a program controlling the interaction between a telephony based device 112 and an internet based web application server 114. The call router 110 preferably controls the call and program logic that enables telephony devices 112 to interface with the application server 114, as discussed in more detail below. The call router 110 preferably detects events occurring from the telephony application 114 and publishes these events to the event router 120. The call router may additionally include an event distributor 116 that determines to which event router to deliver the event. In one variation, the event router 120 includes a plurality of message brokers 128 that manage the publication of events. The message brokers 128 may be sharded or arranged according to event type or any suitable arrangement. The event distributor 116 preferably has control logic to know the correct message broker 128 to send an event. The control logic is preferably updated or synched with the scaling of hardware or software resources of the event router. As an example, an event may have varying attributes, such as account ID, a call ID, or event type. The message brokers 128 may be sharded based on any of these attributes or any combination of attributes. For example, if there are 3 event routers (shards), then the call router could convert the call ID into an integer and that number modulo 3 to determine which event router to contact. An event proxy server 122 may additionally share this control logic such that the event proxy server 122 knows which message broker(s) 128 to subscribe to. The event proxy server 122 preferably uses a similar technique to determine which event router 120 to contact based on what attributes where subscribed to. Additionally an event may be distributed (published) to multiple message brokers 128, such as in the situation where an event has attributes that are managed by multiple message brokers 128. For example, the event may be published by one message broker 128 that manages publication of events for a particular account and the event may additionally be published by a second message broker 128 that manages publication of events of a particular event type. Additionally, multiple event distributor 116 may be allocated or deallocated. The call router and the event router preferably communicate using HTTP or alternatively HTTPS, though any suitable communication system may be used. The published event preferably includes the account to which the event belongs, the type of event, and optionally a set of event data related to the event.

[0018] The call router 110 of the preferred embodiment additionally functions to initiate or receive calls from the telephony device 112 and connect to a web-application server 114. The call router 110 is substantially similar to the call router disclosed in application number 12/417,630 filed on 02 April 2009 and entitled "System and Method for Processing Telephony Sessions" which is hereby incorporated in its entirety by this reference. The call router 110 is preferably connected to a PSTN device over the PSTN network, such that it can receive and make calls from PSTN-connected devices 112, such as landlines, cellular phones, satellite phones, or any other suitable PSTN-connected devices, as well as non-PSTN devices, such as Voice-Over-Internet-Protocol (VOIP) phones, SIP devices, Skype, Gtalk, or other Internet addressable voice devices. The call router 110 may alternatively or additionally function as or include a message router for use with message-based networks such as SMS, email, faxes, instant messaging, or micro-blogging networks. The call router 110 can preferably connect to an SMS network, such that it can receive and send messages from SMS network devices 112, cellular phones, computers, smart phones, or any suitable SMS network devices. The call router 110 may also send or receive text messages, multimedia messages, emails, faxes and other suitable PSTN-compatible communication messages. The call router 110 can preferably connect to an instance messaging network, such that the call router 110 can receive and send messages and receive and transmit presence information from different instance messages networks such as those based on protocols like Jabber, AIM, or fax. The call router 110 can preferably connect to micro-blogging networks such as Twitter such that it can receive and send messages to and from micro-blogging networks via APIs exposed by those networks. The call router 110 may alternatively send and receive message from any suitable system with exposed APIs. The call router 110 preferably communicates with the application server 114 using an application layer protocol, more preferably using the HTTP, or secure HTTPS, protocol. The communication between the application server 114 and the call router 110 is preferably stateless and any state information (e.g., call state) or data is preferably located in a URI or the request parameters, such as HTTP headers, GET URI parameters, POST request body parameters, or HTTP cookies. Available state information is preferably transmitted by call router requests to the application server for stateless processing, and the application server preferably stores no state. Alternatively, the application server preferably stores local state information, such as databases or sessions, as is common in web development. The call router 110 preferably stores state information in call router resources 29. The call router resources are preferably accessible by the application server 114 and other devices through a call router API. The call router 110 preferably associates each incoming phone number with a starting resource address (or more specifically a URI), more preferably the URI is provided by the application server 114, still more preferably the URI is provided by the application developer before a call is received at the call router 110 by associating the initial URI with the incoming call address (such as DID, SIP address, etc.) or by the application upon initiation of an outgoing call. The call router 110 preferably sends call data such as the caller number (obtained via Caller ID), caller geographic data (country, city, and/or state, zip), the number dialed, the time of the call, or any other suitable information or parameter. The call data is preferably digitally signed with a secret key stored on the call router 110. A cryptographic hash of the information is preferably included along with the information as a digital signature. The call router 110 may also encrypt sensitive information (either before or after the cryptographic hash is computed) using the secret key to allow sensitive information to be sent across the network. The call data is preferably sent as an

HTTP POST request to the application server 114. Call data may also be sent in URL (GET) variables, or encapsulated in HTTP headers. An example HTTP request containing the information in the header as shown in FIGURE 8A and 8D. As shown in FIGURE 8B, further inputs (such as voice recording or DTMF button pressing) from the PSTN-device may be subsequently submitted to the application server 114 as HTTP requests (GET or POST). As shown in FIGURE 8C, the inputs from a phone keypad may be included in an HTTP GET request. As shown in FIGURE 8E, the content of an SMS message received by the call router may be sent to the application server 114 as an HTTP request. As shown in FIGURE 8F, the inputs from the text message are included in an HTTP GET request. The request data may alternatively be simultaneously sent in the URI (query string), message body (POST) and message headers, or any combination of the above.

[0019] Any suitable action or parameter, either initiated by the telephony device 112 or by the application server 114, may constitute an event generated by the call router 110. The call router 110 preferably automatically detects such events through any suitable program logic. Events may relate to call related events such as starting or ending a call, starting or ending dialing a number, or any call based occurrence. The events may additionally or alternatively be related to telephony actions such as starting or stopping recording audio, starting or stopping a Text-To-Speech (TTS) conversion, starting or stopping the playing of an audio file, beginning or stopping the gathering of telephony input, redirecting the call to another phone number, or any telephony based instruction. Events may additionally be adjusted for particular applications such as conference calls. Conference call events might include a participant joining a call, a participant leaving a call, gathering telephony input, participant muted, participant unmated, or any suitable conference based event. Furthermore, events may be generated for actions that occur on the message-based protocols used by the call router 110. For example, an messaging events might include a message sent, message received, and message error for SMS, email, faxes, instant messaging, or micro-blogging messages.

[0020] The event router 120 of the preferred embodiment functions to connect published events with subscribers of the event. Preferably, the published event is pushed to subscribers through an open HTTP connection (a single continuous HTTP connection). The open HTTP connection functions to simplify software of a web application using the system. Alternatively, the connection may push data with HTTPS, intermittent HTTP/HTTPS connections, AMF Channel, TCP connection, UDP connection, chat protocols such as jabber, or any suitable messaging framework. The event router is preferably a server, which may be partitioned and scaled for greater capacity. The event router 120 may alternatively be a monolithic system or any suitable software or hardware device(s) for routing events between any number of event publishers and authorized subscribers. In one preferred embodiment, the event router includes an event proxy server 122 and/or a message broker 128. The event proxy server 122 preferably manages the subscriptions from a client (i.e., subscriber) and/or performs more computational intensive processes like filtering events and security. The message broker 128 preferably manages the publications and more preferably manages a subset of event publications from the call router. The event proxy server 122 is preferably part of a cluster of event proxy servers 122 that can be dynamically scaled to satisfy capacity requirements. The message broker 128 is preferably part of a cluster of message brokers 128 that can similarly be dynamically scaled to satisfy capacity requirements. A load balancer may additionally be included within the event router 120 to manage the capacity load of the various components (e.g., the event proxy servers 122 and the message brokers 128). A plurality of load balancers may be individually implemented for each component type, or a single load balancer may manage the event router 120.

[0021] The message broker 128 of the event router 120 functions to manage the publication of events. The message broker 116 (or core message distributor) preferably handles routing of events to be published. A message broker is preferably any message broker as is known in the art such as RabbitMQ or other Advanced Message Queuing Protocol (AMQP) based broker. Preferably, a plurality of message brokers 128 is used to manage the events. More preferably message brokers 128 are sharded (i.e., partitioned) according to dedicated event types (or group of event types). Events are preferably distributed to the appropriate message broker 128 based on the event type. The sharding of message brokers may alternatively be according to any suitable rule. The message brokers 128 (i.e., the shards) may additionally be hosted on different hardware or software platforms, and event type responsibilities may be adjusted when additional message brokers 128 are allocated or deallocated from the cluster of message brokers 128. The message broker 128 may alternatively be a single device for all published events or hosted on a single system. A message broker 128 preferably sends events to an event proxy server 122 that is subscribed to a particular event. A message broker 128 may additionally send an event for any number of subscriptions, where the subscriptions are managed by any suitable number of event proxy servers 122.

[0022] The event proxy server 122 of the event router 120 functions to manage the subscriptions of clients. The client preferably connects to the event proxy server 122 for establishing a subscription to an event and to receive notification of events being published through the event router 120. Events published by a message broker 128 are preferably distributed to a subscribed event proxy server 122, and the event proxy server 122 preferably sends the event to a client. The event proxy server 122 is preferably part of a plurality of event proxy servers 122 that can be automatically scaled. Additionally, an event proxy server preferably manages multiple subscriptions, and may subscribe to multiple message brokers 128. Additionally, a plurality (or series) of event proxy servers 122 may be connected to a single message broker 128 or any suitable device publishing the event for the event router 120. The plurality of event proxy servers 122 functions to increase the volume of subscriptions the event router 120 can manage. A plurality of event proxy servers 122 may alternatively be used with a partitioned message broker 128, multiple message brokers 128, or any suitable configuration. As another variation, an event proxy may have multiple subscriptions (e.g., for different clients) to a message broker 128. This multiple subscriptions may have redundancies. The event proxy server 122 may aggregate the subscriptions for improved system efficiency, as shown in FIGURE 10. The event distributor 116, the message brokers 128, and the proxy servers

122 cooperate to increase the subscription capabilities of the event router. The event proxy server 122 additionally functions to perform resource intensive processes such as running an event filter or security policy engine. The event proxy server 122 is preferably a dedicated server for handling event filtering, operating the security policy engine, and/or any suitable CPU intensive tasks.

[0023] The event router 120 of the preferred embodiment may additionally include an event filter 124 that functions to selectively pass events to a client. The event filter 124 is preferably operated on an event proxy server 122. Event filters 124 selectively filter the number of events published to a particular subscriber based on account security, event type, contents, and/or any suitable parameter of the event. When a subscriber issues a subscription request, the request preferably contains a set of credentials and more preferably a set of event filters. The event router 120, more preferably the event proxy server 122, first verifies the account ownership via the credentials to determine if the subscriber is authorized to view events for the given account. Once a subscriber's identity is determined, the event router only passes events that are associated with authorized accounts. Preferably, this account-level security preferably limits visibility of events to only the relevant account or accounts, and the event filters are preferably applied after the account-level security. The event proxy server 122 preferably applies the event filters 124 to determine if the event router should publish an event to a given subscriber. The event filter 124 is preferably a type filter or a parameter filter. A type filter preferably filters event details by the type of event such as 'call start', 'call end', 'call error', 'call warning', 'record start', 'record end', 'gather start', 'gather end', 'dial start', 'dial end' and/or any suitable type of event. A parameter filter preferably filters events based on characteristics of the event such as by caller ID, contents of call, time of call, duration of call, area code of call, and/or any suitable characteristic of a call. A parameter filter may additionally filter based on the event (such as which digit was pressed by the caller, the warning message issued by the call router, or the length of a recording). Event filters 124 may be used in a variety of ways. As one example, filters may be configured to subscribe to a particular call. As another example, the filters may be configured to subscribe to all calls to and/or from a given phone number. As yet another example, the filters may be configured to subscribe to a particular telephony application action such as "call start" across an account (which may involve multiple simultaneous calls for various phone numbers). The event filter 124 may additionally function to provide a level of security. The event filter 124 preferably prevents inspection, observation, receiving, and/or gathering of any useful information concerning a subset of events. A publisher may implement a security event filter 124 for events the publisher does not want a subscriber to see or alternatively, any suitable entity may implement a security event filter 124.

[0024] The event router 120 of the preferred embodiment may additionally include a security policy engine 126 that functions to enforce a security policy governing which subscribers are allowed to subscribe to a particular event. The event proxy server 122 preferably operates the security policy engine 126. Preferably, the security policy engine 126 includes a signed URL. The signed URL is preferably composed of a subscription message and a verification token. The verification token functions to be validation of the authenticity of the subscription request. A private key shared between the client application developer and the event router is preferably used to generate the verification token and is preferably a code, password, or any suitable identifier. The verification token is preferably implemented as an HMAC (Hash Message Authentication Code) hash using the key, but may alternatively be implemented by any suitable cryptographic message authentication technique. The verification token preferably includes the subscription request including a subscription URL, subscription filters, subscription expiration time, and/or any other suitable subscription metadata or parameter. The verification token is preferably attached to the subscription request. In one preferred version, the verification token is appended to the end of a URL of the subscription message to form the signed URL. The signed URL allows the subscription request to be passed to unknown devices, such as a remote web browser, and allowing the browser to issue subscription requests without knowing the key or other information. Alternatively, other security systems such as OAuth URL signing or any suitable security method may be implemented.

[0025] The system of the preferred embodiment may also include a connection to the client device 130, which functions to be a channel through which published events that a client subscribes to can be delivered. The connection 130 is preferably any suitable network connection either wired or wireless. The connection 130 is preferably between an event router 120 and the client, and more preferably, the connection 130 is between an event proxy server 122 and the client. The connection to the client is preferably an HTTP connection but may be any suitable signaling protocol. The client device of the preferred embodiment functions to provide interaction with subscribed events. The client device may be a front-end interface of the web application. The client device preferably reacts to events and provides an interface for user interaction. The client device may be a website, a computer program, a web enabled consumer product, a server, or any suitable device. However, the client device may alternatively be a backend system such as a data collection system. A browser 132 is one common type of client. A connection with a browser is preferably implemented via Ajax in javascript (e.g., XMLHttpRequest) or via a flash plugin (e.g., XMLSocket or an AMF or SecureAMF channel). An Application Programming Interface (API) server 134 is a second common type of client. A client preferably initiates the creation of connection 130 to the event router 122. However, in the situation of an API server 134, the event router 120 or, more preferably, an event proxy server 122 may initiate the creation of a connection 130 to the client 130. There may additionally be a control channel and a publication channel. The control channel is a connection through which a client submits a subscription request. The client can manage subscriptions through the control channel. Management of subscription includes modifying an existing subscription (e.g., updating filters or changing expiration time), adding a subscription, deleting a subscription and/or any suitable subscription change. The subscription channel is the connection through which events are sent to the client. In one variation, a client may setup multiple subscriptions and/or modify subscriptions through multiple control channels or alternatively through one control channel at different times. One publication channel is preferably used regardless of the number of subscriptions of a client. This functions to reduce the number of open connection 130 between the client and the event proxy server 122. The connection 130 may be any suitable connection as mentioned above. In the case where the connection is a long poling type connection, a client preferably connects to the event proxy server 122, gets an event, and closes a connection. When the client is not connected, events may be missed by the client. The event proxy server 122 preferably includes a cache to store events (up to an expiration time) and delivers the cached events to the client during the next connection with the client.

[0026] The application server 114 of the preferred embodiment functions to provide a web developer with an improved development environment for interfacing and designing interactions for communications applications. The call router 110 preferably manages the interaction between the telephony device(s) 112 and the application server 114. Events are preferably generated by this interaction. The web application (application server) is preferably a website, but may alternatively be a computer program, a web enabled consumer product, or any suitable method or device capable of event subscription tasks. The application server 114 preferably combines telephony actions and a website to form a powerful user experience. In one example of an application server 114, a user may enter a phone number and leave audio comments for individual photos as the photos cycle through a slideshow. In a second example, a conference call can be managed through a web interface. In a third example, a customer service phone call may be managed and annotated using a web interface. In a fourth example, a business or sales phone call may be managed and supported by using a web interface. Any suitable application server utilizing telephony interaction may alternatively be used. The web application is preferably programmed in a manner similar to regular websites, but integration into the system allows for new user experiences relying on telephony actions.

[0027] The application server 114 functions to provide data processing logic for requests received from the call router 110. The application server 114 is preferably connected to the call router 110 via a network 24, more preferably via the Internet. The application server 114 is preferably a third party server operated outside of the system, but the system may alternatively include the application server 114. A URI is preferably associated with an application server 114 or an application on an application server 114. The application server 114 preferably communicates with the call router 110 using an application layer protocol, more preferably using the HTTP protocol, or more secure HTTPS protocol. The application server 114 preferably receives HTTP requests from and sends HTTP responses to the call router 110. The application server 114 preferably runs on a standard stack of programming languages, hosting providers, operating systems and databases to handle HTTP requests, as if the caller were a website visitor in a web browser. The application server 114 also preferably verifies the digital signatures of the call data received in the requests using the secret key to compute a cryptographic hash from the received information and the hash received. If the computed hash and the received hash do not match, or no hash is received with the request, then the application server 114 preferably determines the request is fraudulent, and the request is preferably discarded. If the computed hash and received hash match, the application server 114 preferably determines that the request is authentic and proceeds further with the processing of the request. The application server may alternatively choose to ignore the hash if security is not important. The application server preferably uses call state data communicated by the call router request to determine the next call router instructions, without requiring call state stored on the application server. The application server may alternatively use call state data sent by the call router, such as the caller ID of the caller or the unique ID of the call, to reference additional or external state data, such as rows in a database or session data stored on the application server.

[0028] The application server 114 preferably responds to HTTP requests received from the call router 110 by generating telephony instructions for the call router 110. Telephony instructions when executed by the call router preferably cause an event to be detected by the call router 110. The application server preferably replies to the call router in XML, however, any suitable machine-readable message format may be used, including HTML, key/value pair text, delimited text or binary encoding. The XML preferably includes the telephony instructions for the call router 110 such as connecting to another number, playing a recorded greeting, reading text, and/or requesting DTMF digit entry from the caller. The telephony instruction may alternatively be related to SMS messaging, Multimedia Messaging Service (MMS) messaging, faxes, instant messages, email, micro-blogging, or any suitable messaging task. The telephony instruction may additionally be used to send an outgoing SMS message, arrange a phone call from a specific phone number, arranging for a callback, setting up a conference call (connecting multiple numbers), sending an email, interfacing with a calendar or scheduling system, purchasing goods, or services, or any other suitable instruction. The

XML instructions are preferably a set of commands to be executed in order, one at a time (i.e., sequentially). An example XML response is shown in FIGURES 9A and 9B. In single telephony session (e.g. one initiated by a PSTN-device or an SMS device), a response from an application server can initiate an outgoing telephony call and/or a SMS message. That is, a single XML response preferably provides the ability to interact with both the SMS network and the voice telephony network (PSTN, SIP/VoIP, etc) sequentially or simultaneously. In addition, audio or video files sent to the call router 110 can be converted to text by an automatic speech-to-text engine, human or other technique, and sent back in text form as an SMS message or an attachment to an MMS. In one variation, an application running on a server may be a simple static XML page and static sound files, deployed on basic web servers where no development or scripting environment is available. This variation preferably uses URI Templates (a current IETF proposal for HTML5), which essentially includes URLs with placeholders for variable data, like this: http://www.twilio.com/audio/{Digit}.mp3 where the call router 110 would substitute the digits pressed for the {Digit} placeholder in the URI Template, GET the file at the resulting URI, and play the static sound file in response. This allows an entire application to be authored offline in a What-You-See-Is-What-You-Get (WYSIWYG) html editor. For example, if the server response specifies the URI Template: http://demo.twilio.com/myapp/{Digits}.mp3, and the caller presses digits 1234, the call router 110 would GET the static mp3 file located at: http://demo.twilio.com/myapp/1234.mp3 and play it to the caller. The variables used for substitution in the URI Templates preferably correspond to the names of variables defined for state submission in HTTP GET, POST and/or header requests from the call router. From the previous example, {Digits} would be associated with a parameter named "Digits" that is preferably generated as a result of a "gather" telephony instruction (collection of DTMF digits). In the preferred embodiment for the second configuration, the call is initiated by the application server 114 (through the call router 110), and the second configuration is substantially similar to the first configuration, such that the call routing is preferably handled identically to an incoming call, namely via URI requests from call router 110 to the server 114 upon call state changes.

[0029] As an additional alternative, the system may be implemented to be full duplex where events (client events) may additionally be published from a client and the call router can subscribe to the client events as shown in FIGURE 7. In this alternative, the event router additionally manages the publication of client events and manages call router subscriptions to client events. The system is preferably implemented in substantially the same way as above, but with the client additionally generating events and the call router subscribing to events. The client event system is preferably integrated with the eventing system described above.

2. Telephony web event method

[0030] As shown in FIGURES 3 - 6, the method of event subscription for telephony applications includes distributing events S100 and subscribing to events S200. Distributing events preferably includes the sub-steps publishing an event to a router S110, identifying subscribers to an event S120, and sending an event to a subscriber S130. Subscribing to events includes the sub-steps of generating a signed URL for an event subscription S210, sending an event subscription request to an event router S220, verifying an event subscription S230, and allowing an event subscription S240. The method may additionally include allocating new resources to the event router. In particular event proxy servers and message brokers may be allocated or deallocated. Additionally, call routers, event distributors, and/or any suitable part device of the system may be allocated or scaled to accommodate capacity needs. A load balancer may additionally distribute processing across the plurality of components.

[0031] As an alternative, the method may additionally include receiving a subscriber generated client event, publishing the client event to the event router and identifying a call router subscribed to a client event, and sending the client event to the call router. This functions to make the eventing method full duplex for two way event publication and subscription. The duplex eventing system is substantially similar to the eventing system described, but where the client generates the events and the call router is subscribed to the events.

2A. Method of Publishing an Event

[0032] Step S110, which includes publishing an event to a router, functions to initiate the announcement of an event. Event details are preferably sent to an event router at a URL or other suitable resource or connection identifier. Event details preferably include account identification, event type, any event data associated with the event, and any other suitable parameters relating to the event. Publishing to a router preferably occurs after a new event occurs, but alternatively, a batch of events may be published periodically, a batch of events may be published when an event count is satisfied, an event may be published when an event type is satisfied, or any suitable event publishing rule may be applied. An event is preferably generated from a telephony application operating on a call router. The telephony application is preferably substantially similar in functionality to the one described above. A call router preferably publishes an event to an event router. The event is preferably published over HTTP, but any suitable protocol may be used. An event distributor may additionally select a message broker to send the event. A plurality of message brokers may be sharded according to event types and the event distributor preferably is capable of mapping the event to the appropriate message broker.

[0033] Step S120, which includes identifying subscribers to an event, functions to identify all authorized subscribers that should be notified that the event occurred. The subscribers are preferably associated with a subscription URL. Any suitable number of subscribers is preferably identified, and subscribers are preferably identified by inspecting a list of subscribers. The subscribers may alternatively be associated with a group of other subscribers, and a group (or groups) may be identified as a subscriber. Identifying subscribers is preferably performed by an event router, and more preferably an event proxy server in cooperation with a message broker, though any suitable device may be used. Preferably, an event proxy server performs the steps of managing a subscription of a client (subscriber) and subscribing to an event publication of the event router. The event proxy server preferably subscribes on behalf of a client, so that subscription processing can be delegated to the event proxy server. More preferably a message broker performs the steps of publishing the event. So that the event proxy server subscribes to a message broker. Identifying subscribers of an event may include a sub-step of verifying event filters. An account-level security may additionally be enforced by the event router to limit the visibility of events to only relevant accounts. The event is compared to filters of a subscriber to ensure the subscriber should be sent the event. The filters are preferably type filters or parameter filters as discussed above.

[0034] Step S130, which includes publishing an event to a subscriber, functions to notify a subscriber of the occurrence of an event. The event is preferably published by an event router over an open HTTP connection, but alternatively, a periodic HTTP connection, messaging framework such as Jabber, or any suitable communication protocol may be used. In the situation where a client has previously broken a connection with the event router (i.e., is not connected to the event router at the time an event occurs), the event proxy server preferably establishes a connection to the subscriber. The event proxy server preferably establishes the connection by accessing the stored address of the client and connecting through any suitable protocol. In the case the subscriber is an API server, an API command may be used to connect or notify the API server of the event.

2B. A Method of Subscribing to an Event

[0035] Step S210, which includes generating a signed URL for an event subscription, functions to generate a URL encoding any subscription URL, filter, and/or identification information. An unsigned URL is preferably generated including account identification, subscription URL, subscription filters, subscription expiration time, and/or any other suitable subscription metadata or parameter. Preferably, a key is looked up based on the account identification. The key is used to create a verification token. The verification token is preferably implemented as an HMAC-SHA1 (Hash Message Authentication Code) hash using the key or any suitable another cryptographic message authentication technique may be used. The verification token additionally includes the subscription request including a subscription URL, subscription filters, subscription expiration time, and/or any other suitable subscription metadata or parameter. The verification token is preferably appended to the unsigned URL to form a signed URL. The signed URL is preferably integrated into a subscription request. The subscription request is preferably a request to receive particular events of a telephony application.

[0036] Step S220 includes sending a subscription request to an event router. The subscription request is preferably sent to an event router by HTTP protocol, but any suitable protocol may alternatively be used. The subscription request is preferably received by an event router, and more preferably is received by an event proxy server. The event proxy server preferably manages subscriptions.

[0037] Step S230, which includes verifying an event subscription, functions to verify the identity of a subscriber. The signed URL of the event is preferably deconstructed to identify account identification, subscription URL, subscription filters, subscription expiration time, and/or any other suitable subscription metadata or parameter. The event router preferably verifies that the account identification is included in the signed URL or other authentication credentials. If no account identification is found, the subscription request is discarded and an error is returned. If the identification information is included a key for the account is looked up (i.e. found in a database). The key is preferably a shared key that is identical to the key of Step S210. The key is then used to form a verification token. The verification token is preferably a HMAC hash, or alternatively any suitable cryptographic message or identifier. The verification token is compared to the verification token from the signed URL to verify the match.

[0038] Step S240, which includes allowing an event subscription, functions to allow a client to subscribe to an event. A subscription preferably allows a client to receive events in real time (approximately a few milliseconds to a few seconds). An event subscription also only allows events that are authorized to be viewed by the account, such as events generated by calls on that account. More preferably, the events preferably pass any filters of a subscriber. As an additional alternative, subscriptions may expire after a given time. As part of Step S240, the method includes the event proxy server (or event router) configuring filters for the subscription. This step functions to setup processing operations of the subscription. Additionally, any suitable subscription setup that must be performed is additionally performed. In the variation where the subscriber previously has a subscription, the previous subscription may be modified to include the new subscription details. Events from multiple subscriptions of one subscriber are preferably sent to the subscriber through a single connection as described above. In the situation where a subscriber is not connected to the event router when an event occurs the event proxy server or any suitable device may queue or cache the events for delivery when the subscriber next establishes a connection.

[0039] As a person skilled in the art will recognize from the previous detailed description and from the figures and claims, modifications and changes can be made to the preferred embodiments of the invention without departing from the scope of this invention defined in the following claims.

CLAIMS

We Claim:

1. A system for publishing events of a telephony application to a client, wherein the telephony application interfaces with a telephony device and an application server, the system comprising:
   - a call router that generates events from the telephony application; and
   - an event router that manages the publication of events generated by the call router and that manages the subscription to events by clients.
2. The system of Claim 1, wherein the event router includes a plurality of event proxy servers that manage the subscriptions for the clients of the event router and that subscribe to the events of the event router.
3. The system of Claim 2, wherein the client is an API server.
4. The system of Claim 2, wherein the event router further includes a load balancer that manages event proxy loads and scales the event proxy system.
5. The system of Claim 2, wherein the event proxy servers include event filters that selectively pass events to a client according to an event attribute.
6. The system of Claim 5, wherein the event filters selectively pass events from a single call to a client.
7. The system of Claim 5, wherein the event filter filters selectively pass events associated with a set phone number.
8. The system of Claim 5, wherein at least one of the event filters is implemented by an event publisher to prevent a client from receiving an event type.

9. The system of Claim 5, wherein the event proxy servers additionally include a security policy engine that authorizes clients to subscribe to an event.

10. The system of Claim 5, wherein the event router includes a plurality of message brokers that manage the publication of events for the event router, and wherein the event proxy servers connect to a message broker to subscribe to an event.

11. The system of Claim 10, wherein the message brokers are sharded according to an event attribute.

12. The system of Claim 1, wherein the call router simultaneously manages a plurality of telephony applications and the event router publishes a plurality of events and manages a plurality of subscriptions.

13. The system of Claim 1, wherein the client generates client events, and the event router additionally manages the publication of client events and manages call router subscriptions to client events.

14. A method for publishing events of a telephony application comprising:

- generating an event from the telephony application;
- publishing an event to an event router;
- identifying a subscriber to an event; and
- sending the event from the event router to the subscriber.

15. The method of Claim 14, wherein the step of identifying a subscriber includes an event proxy server managing a subscription and subscribing to an event publication of the event router.

16. The method of Claim 15, further comprising publishing the event within the event router on a message broker.

17. The method of Claim 16, wherein the event is published on at least a second message broker, wherein the message brokers manage the publication of events based on different attributes.

18. The method of Claim 16, further comprising allocating additional event proxy servers to increase subscription capacity and allocating additional message brokers to increase event publication capacity.

19. The method of Claim 16, further comprising the steps of: receiving a subscription request from a subscriber for an event; verifying a subscriber is authorized to subscribe to the event; and initiating management of the subscription.

20. The method of Claim 19, wherein the step of sending the event from the event router to a subscriber includes detecting if the client is not connected and establishing a connection to the subscriber if a connection is not detected.

21. The method of Claim 19, further comprising receiving a subscriber generated client event; publishing the client event to the event router; identifying a call router subscribed to the client event; and sending the client event to the call router.

22. The method of Claim 14, further comprising sending the event from the event router to a plurality of subscribers.

23. A method for forming a subscription to an event of a telephony application comprising:

- receiving a request to subscribe to an event publication from a subscriber;
- verifying the subscriber is authorized to subscribe to the event publication;
- subscribing to the event publication published by a event router, wherein the event router manages subscriptions of a telephony application; and
- returning events of the event publication to the subscriber through an event connection.

24. The method of Claim 23, further comprising aggregating subscriptions to the event publication with a plurality of subscriptions from an event proxy server to the event router.

25. The method of Claim 23, further comprising configuring filters for the subscription and filtering events prior to returning to the subscriber, wherein the filters are specified in the subscription request.

26. The method of Claim 25, further comprising queuing events to be returned to a subscriber, and returning the events to the subscriber through an event connection after the subscriber establishes the event connection.

27. The method of Claim 23, further comprising receiving a request to subscribe to a second request from the subscriber; verifying the subscriber is authorized to subscribe to the second event; subscribing to the second event published by a event router; and returning the second event through the event connection of the first event.

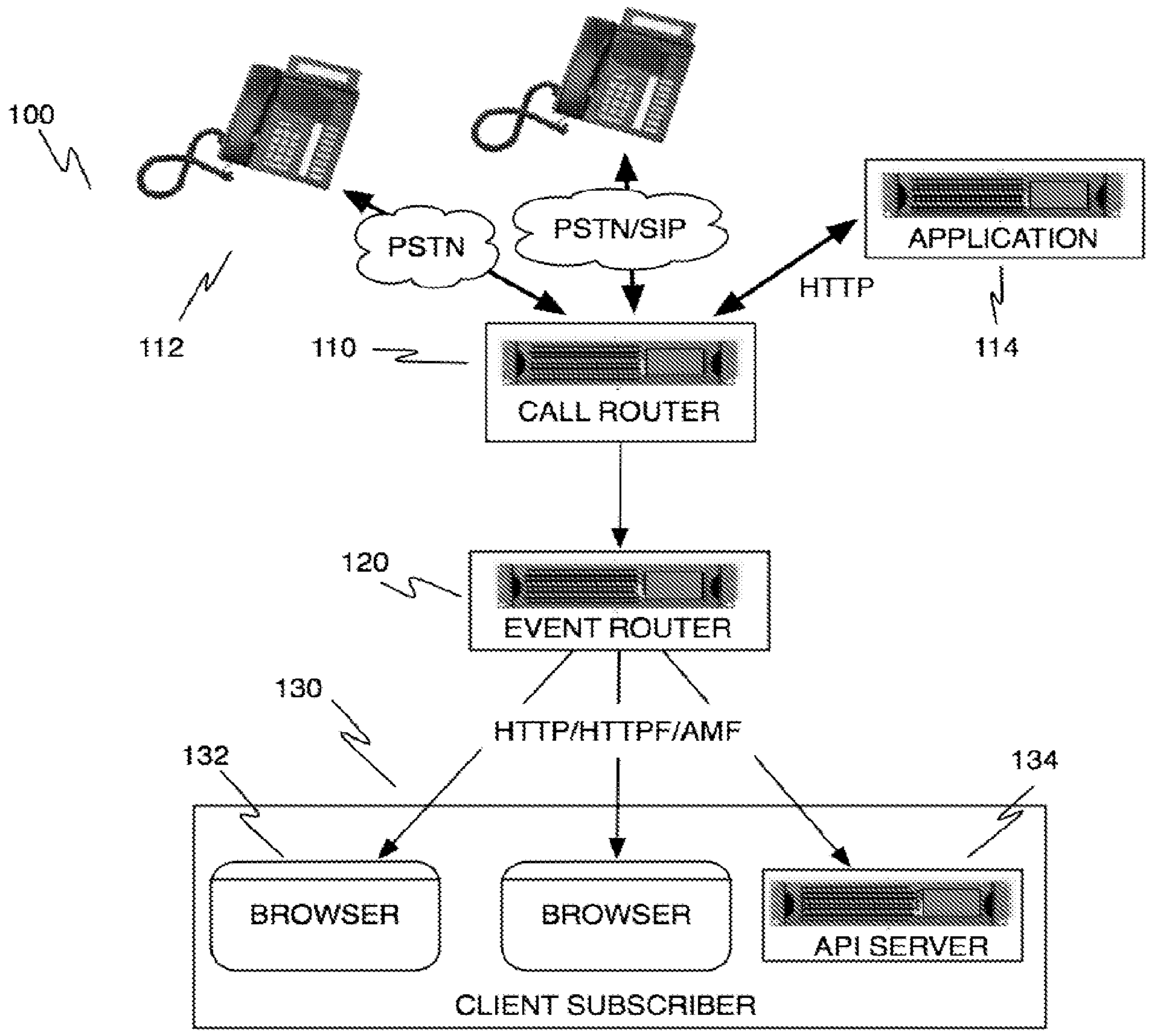
100
PSTN
PSTN/SIP
APPLICATION
HTTP
112
110
114
CALL ROUTER
120
EVENT ROUTER
130
HTTP/HTTPF/AMF
132
134
BROWSER
BROWSER
API SERVER
CLIENT SUBSCRIBER

FIGURE 1

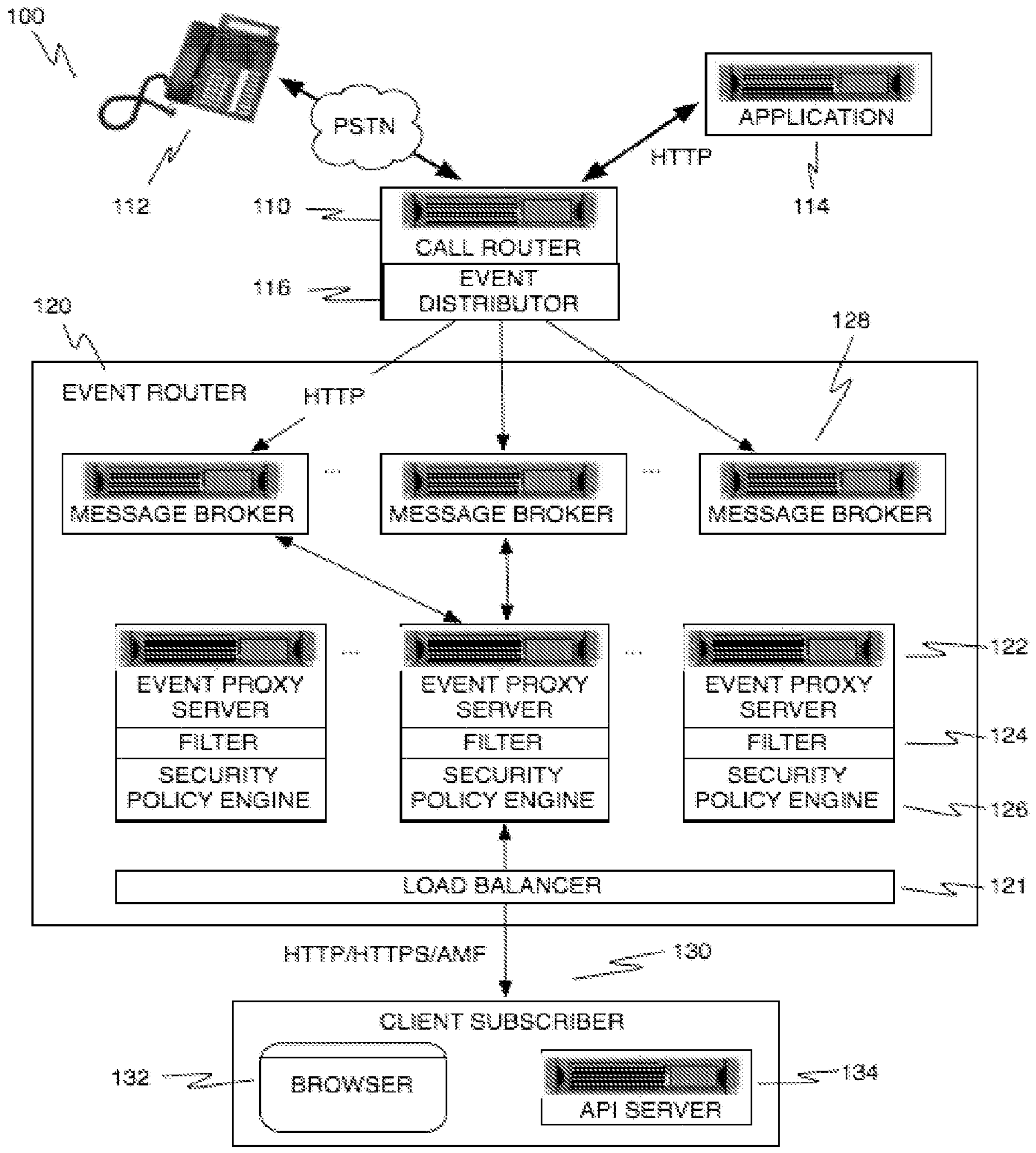
100
PSTN
112
110
CALL ROUTER
EVENT DISTRIBUTOR
116
APPLICATION
HTTP
114
120
EVENT ROUTER
HTTP
128
MESSAGE BROKER
MESSAGE BROKER
MESSAGE BROKER
EVENT PROXY SERVER
FILTER
SECURITY POLICY ENGINE
122
124
126
LOAD BALANCER
121
HTTP/HTTPS/AMF
130
CLIENT SUBSCRIBER
BROWSER
API SERVER
132
134

FIGURE 2

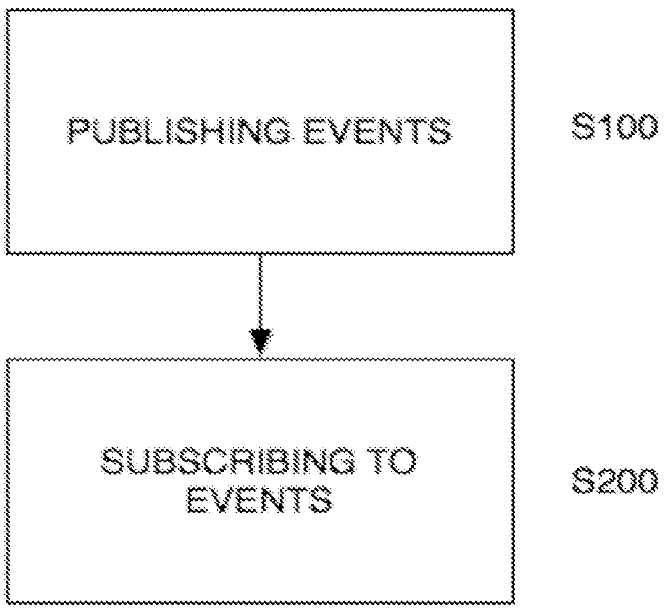

FIGURE 3

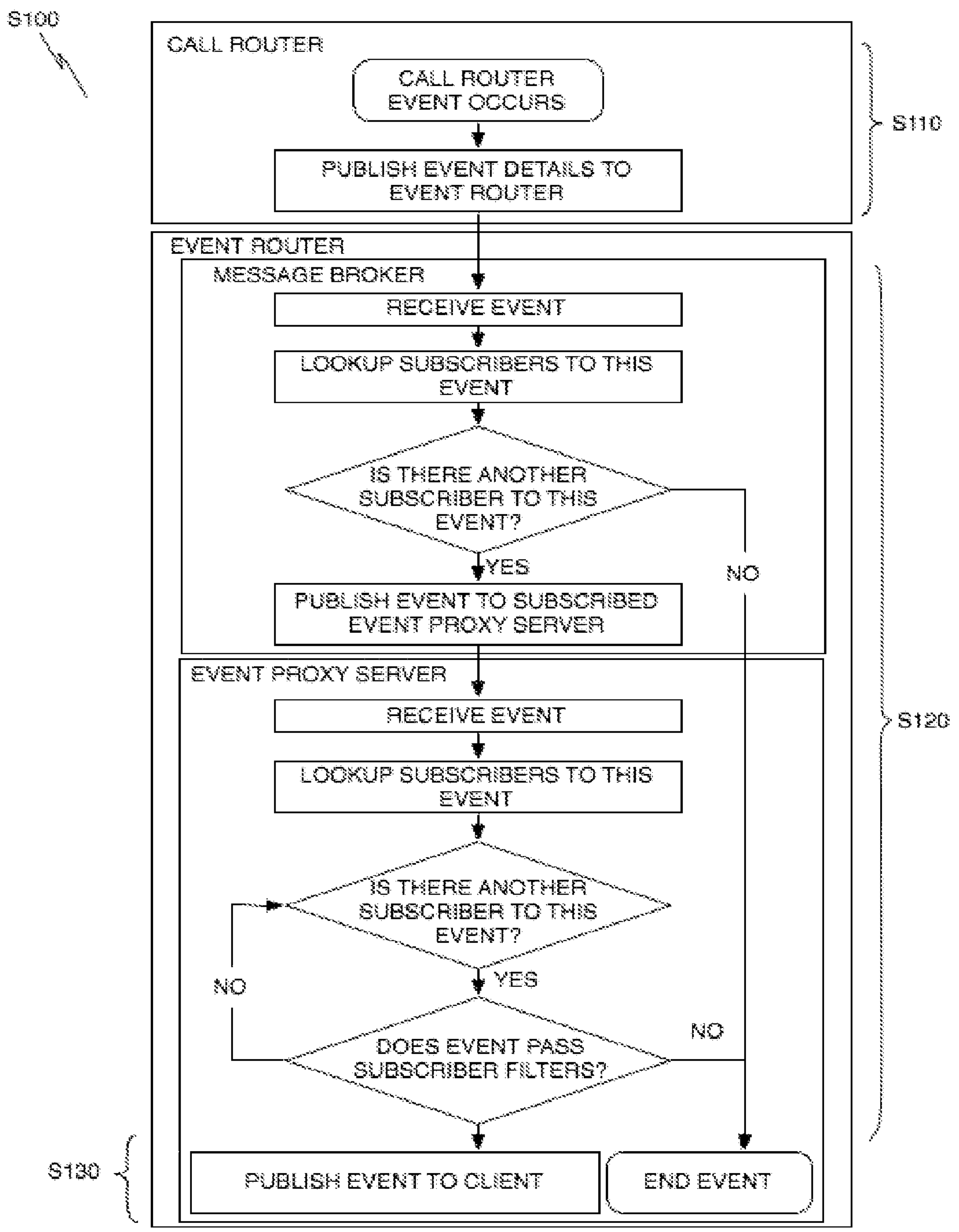
S100
CALL ROUTER
CALL ROUTER EVENT OCCURS
PUBLISH EVENT DETAILS TO EVENT ROUTER
S110
EVENT ROUTER
MESSAGE BROKER
RECEIVE EVENT
LOOKUP SUBSCRIBERS TO THIS EVENT
IS THERE ANOTHER SUBSCRIBER TO THIS EVENT?
YES
NO
PUBLISH EVENT TO SUBSCRIBED EVENT PROXY SERVER
EVENT PROXY SERVER
RECEIVE EVENT
LOOKUP SUBSCRIBERS TO THIS EVENT
S120
IS THERE ANOTHER SUBSCRIBER TO THIS EVENT?
YES
NO
DOES EVENT PASS SUBSCRIBER FILTERS?
NO
S130
PUBLISH EVENT TO CLIENT
END EVENT

FIGURE 4

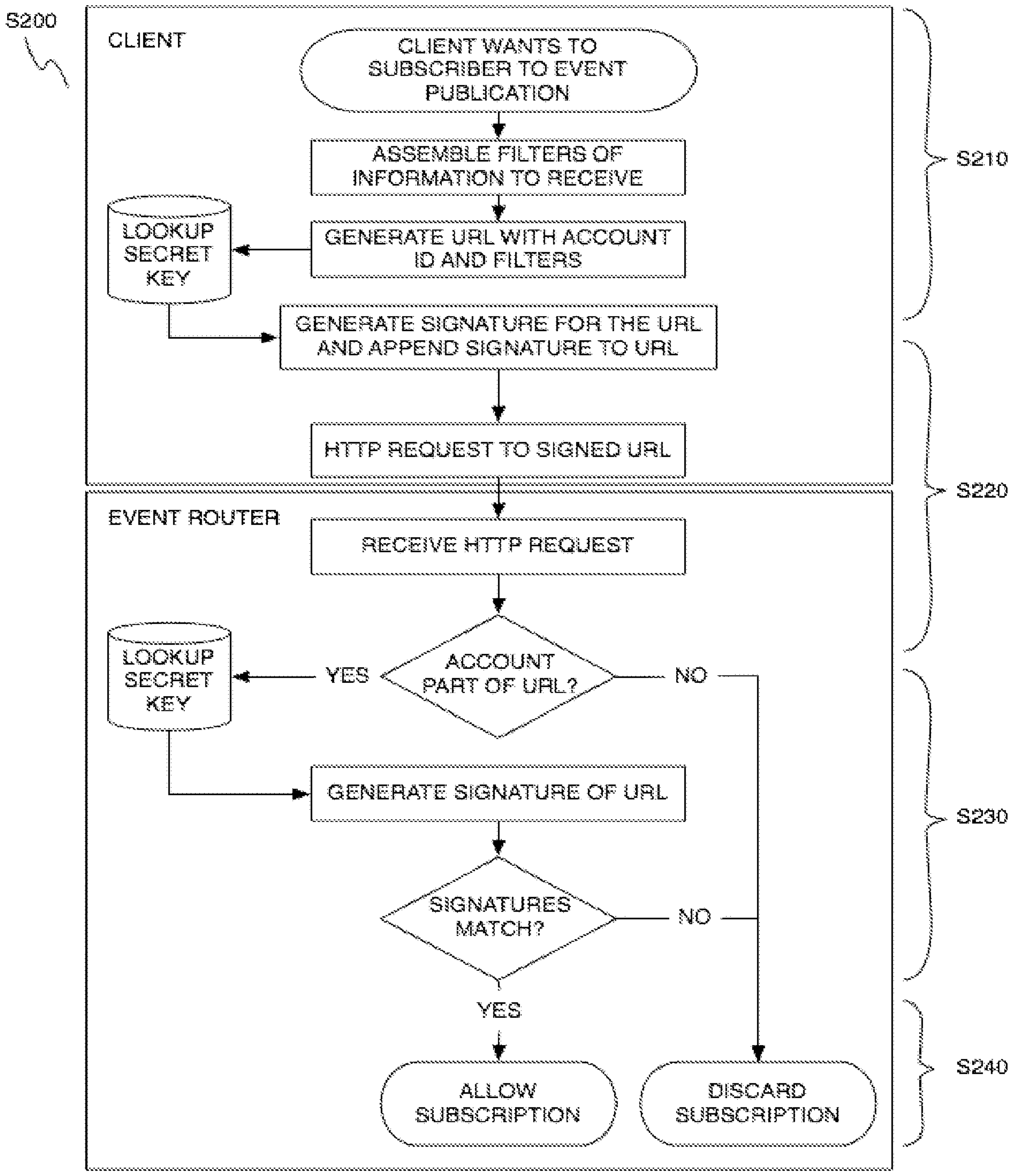
S200
CLIENT
CLIENT WANTS TO SUBSCRIBER TO EVENT PUBLICATION
ASSEMBLE FILTERS OF INFORMATION TO RECEIVE
LOOKUP SECRET KEY
GENERATE URL WITH ACCOUNT ID AND FILTERS
GENERATE SIGNATURE FOR THE URL AND APPEND SIGNATURE TO URL
HTTP REQUEST TO SIGNED URL
EVENT ROUTER
RECEIVE HTTP REQUEST
LOOKUP SECRET KEY
YES
ACCOUNT PART OF URL?
NO
GENERATE SIGNATURE OF URL
SIGNATURES MATCH?
NO
YES
ALLOW SUBSCRIPTION
DISCARD SUBSCRIPTION
S210
S220
S230
S240

FIGURE 5

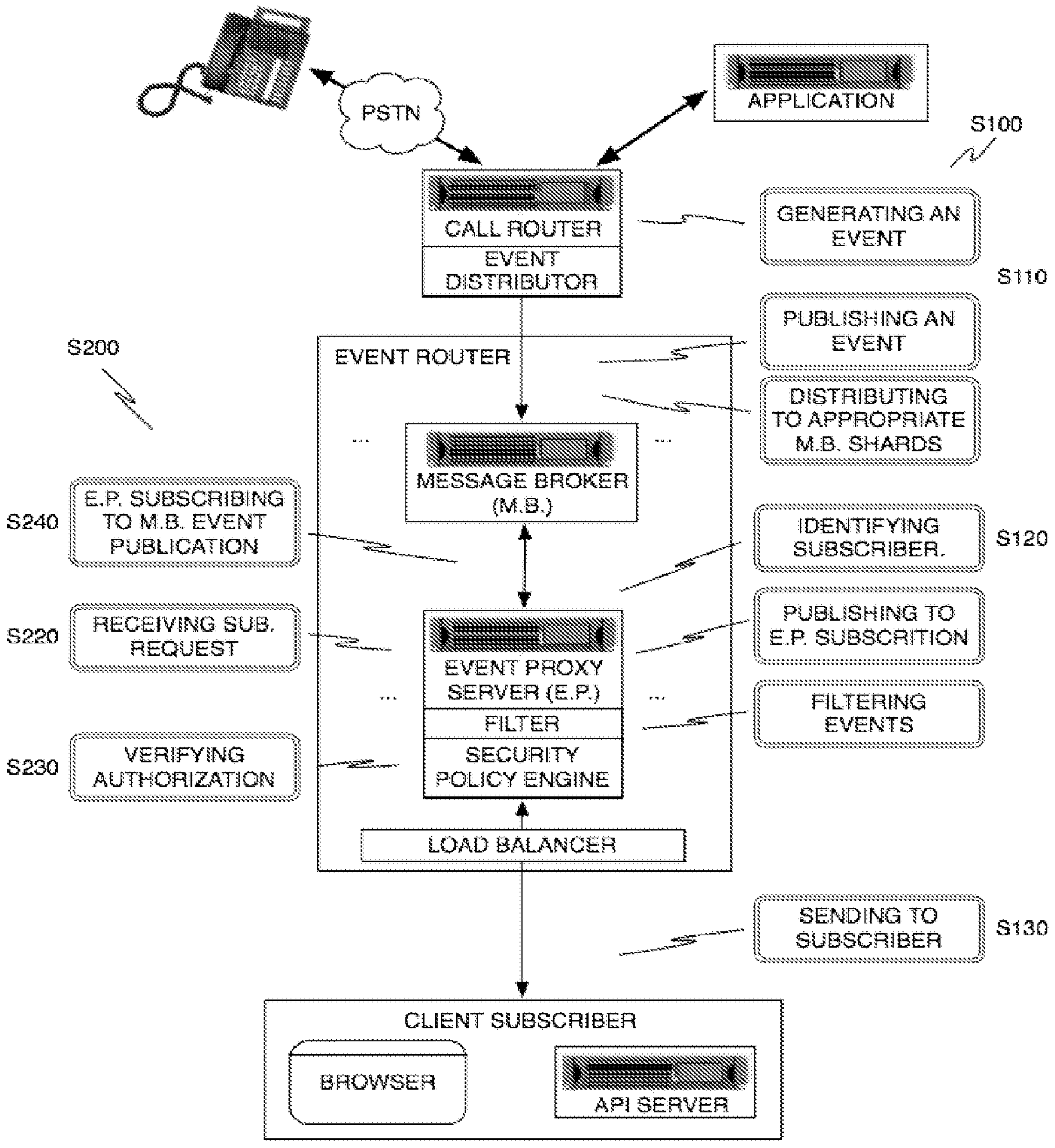
PSTN
APPLICATION
S100
CALL ROUTER
EVENT DISTRIBUTOR
GENERATING AN EVENT
S110
PUBLISHING AN EVENT
S200
EVENT ROUTER
DISTRIBUTING TO APPROPRIATE M.B. SHARDS
MESSAGE BROKER (M.B.)
S240
E.P. SUBSCRIBING TO M.B. EVENT PUBLICATION
IDENTIFYING SUBSCRIBER.
S120
S220
RECEIVING SUB. REQUEST
EVENT PROXY SERVER (E.P.)
PUBLISHING TO E.P. SUBSCRITION
FILTER
FILTERING EVENTS
S230
VERIFYING AUTHORIZATION
SECURITY POLICY ENGINE
LOAD BALANCER
SENDING TO SUBSCRIBER
S130
CLIENT SUBSCRIBER
BROWSER
API SERVER

FIGURE 6

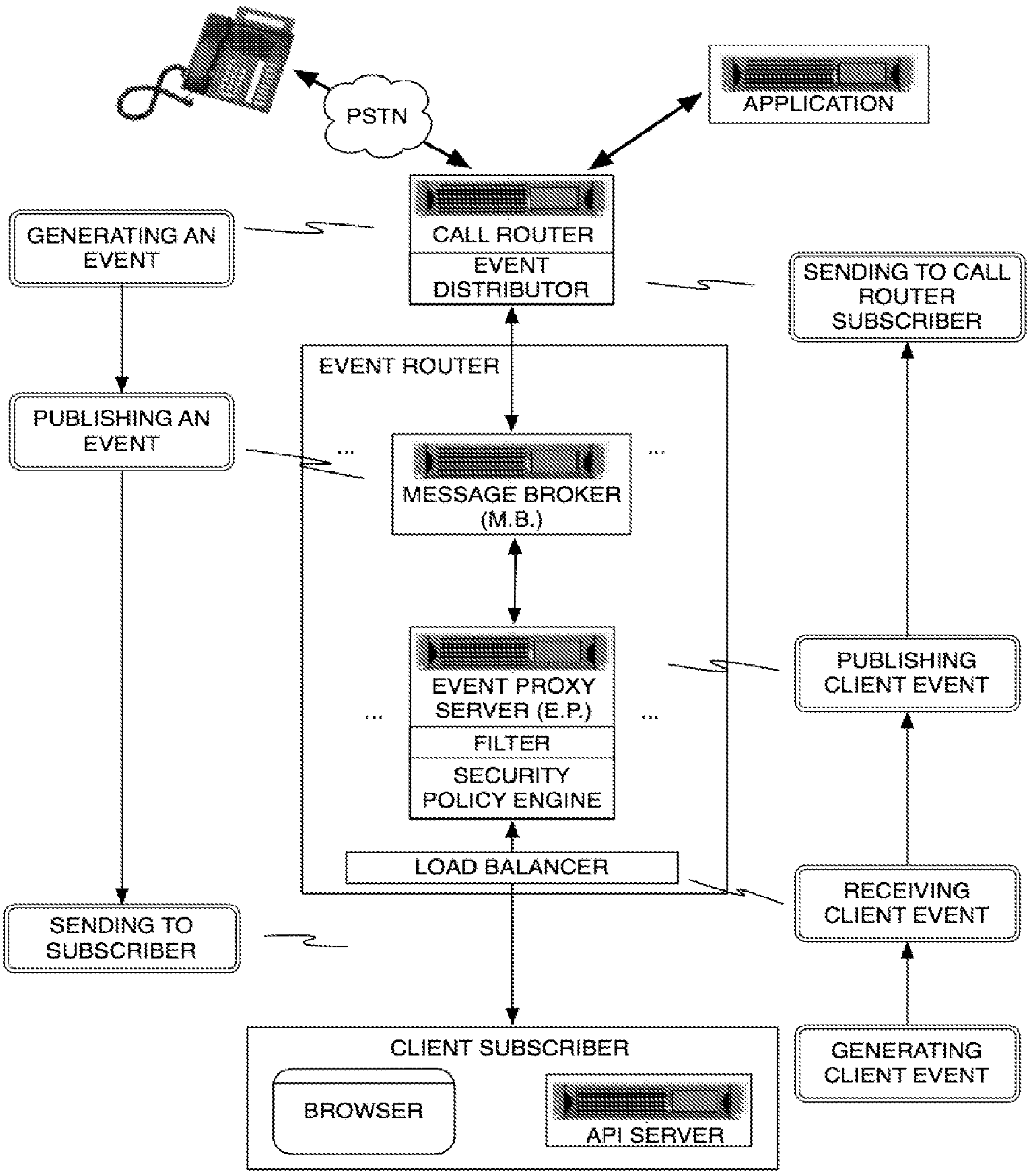
PSTN
APPLICATION
GENERATING AN EVENT
CALL ROUTER
EVENT DISTRIBUTOR
SENDING TO CALL ROUTER SUBSCRIBER
EVENT ROUTER
PUBLISHING AN EVENT
...
MESSAGE BROKER (M.B.)
...
EVENT PROXY SERVER (E.P.)
...
...
FILTER
SECURITY POLICY ENGINE
PUBLISHING CLIENT EVENT
LOAD BALANCER
RECEIVING CLIENT EVENT
SENDING TO SUBSCRIBER
CLIENT SUBSCRIBER
BROWSER
API SERVER
GENERATING CLIENT EVENT

FIGURE 7

```
GET /foo.php HTTP/1.1
Host: demo.twilio.com
X-Twilio-CallGuid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-CallerId=415-555-1212
X-Twilio-NumberCalled=415-867-5309
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length: 0
```

FIGURE 8A

```
POST /foo.php HTTP/1.1
Host: demo.twilio.com
Content-Type: application/x-www-form-urlencoded
X-Twilio-CallGuid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-CallerId=415-555-1212
X-Twilio-NumberCalled=415-867-5309
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length: 11

  Digits=1234
```

FIGURE 8B

```
GET /foo.php?digits=1234 HTTP/1.1
Host: demo.twilio.com
X-Twilio-CallGuid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-CallerId=415-555-1212
X-Twilio-NumberCalled=415-867-5309
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length: 0
```

FIGURE 8C

```
GET foo.php HTTP/1.1
Host: demo.twilio.com
X-Twilio-SMSid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-SMSSenderId=415-555-1234
X-Twilio-SMSShortCode=11111
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length: 0
```

FIGURE 8D

```
GET foo.php HTTP/1.1
Host: demo.twilio.com
X-Twilio-SMSid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-SMSSenderId=415-555-1234
X-Twilio-SMSShortCode=11111
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length: 21

message=statusrequest
```

FIGURE 8E

```
GET foo.php?message=statusrequest HTTP/1.1
Host: demo.twilio.com
X-Twilio-SMSid=DE870AD708ED70AE87D0AE7DAD7
X-Twilio-SMSSenderId=415-555-1234
X-Twilio-SMSShortCode=11111
X-Twilio-AccountId=AAF4AF5AF8A9A885449F7A647AF84
Content-Length: 0
```

FIGURE 8F

XML Response

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
        <Collect
                successUrl="http://www.example.com/phonetree.php"
                numDigits="1"
                timeout=20
        >
                <Say voice="female">
For sales press one. For support press two. For the operator, press three.
                </Say>
        </Collect>
</Response>
```

FIGURE 9A

XML Response

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
        <sms address=415-555-5555>
                thanks for the your text, will call at 5 PM.
        </sms>
        <CallAtTime="17:00PST">
                Today
                <Dial>415-555-5309</Dial>

        </CallAtTime>
</Response>
```

FIGURE 9B

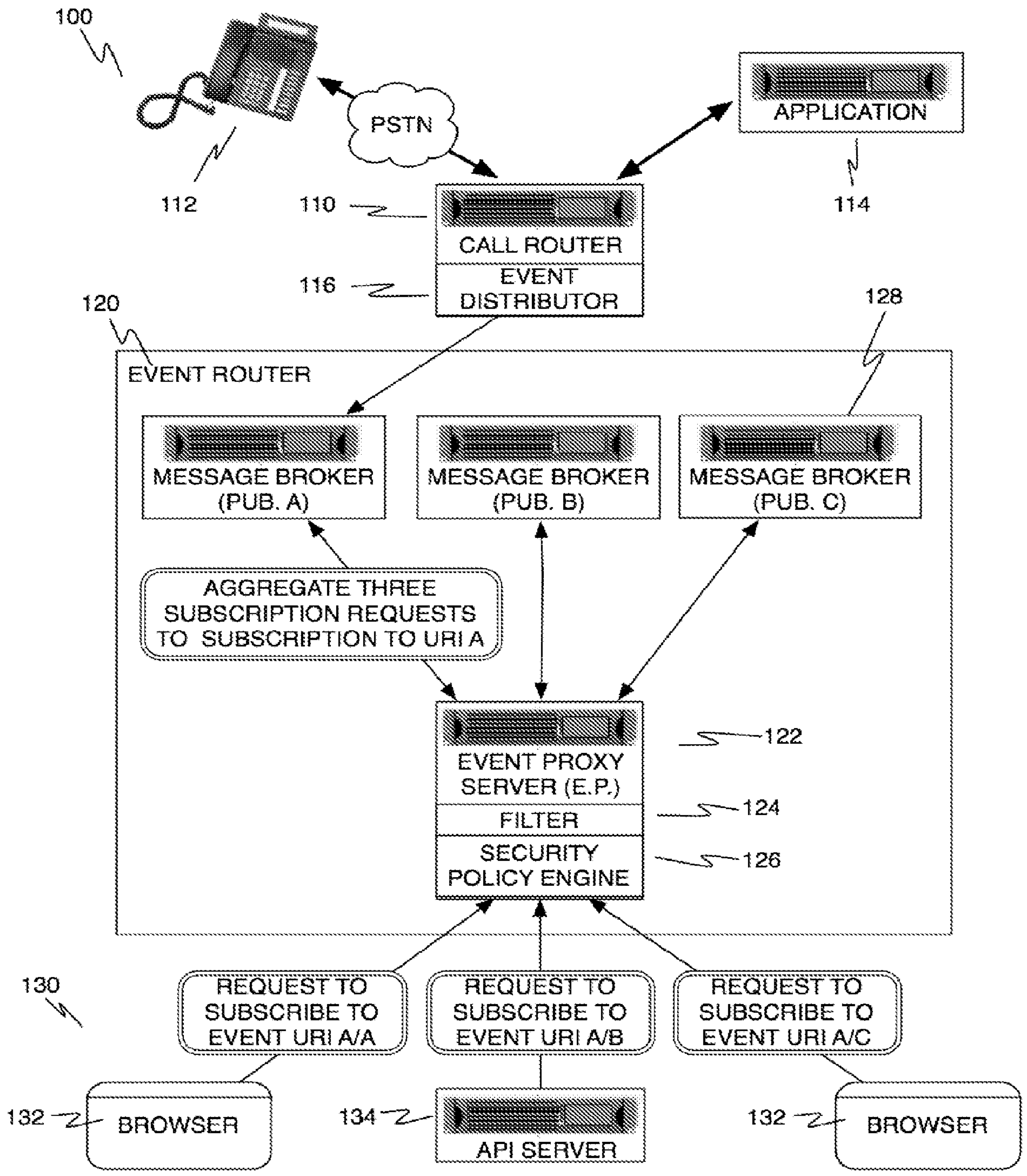
100
PSTN
APPLICATION
112
110
CALL ROUTER
114
116
EVENT DISTRIBUTOR
120
128
EVENT ROUTER
MESSAGE BROKER (PUB. A)
MESSAGE BROKER (PUB. B)
MESSAGE BROKER (PUB. C)
AGGREGATE THREE SUBSCRIPTION REQUESTS TO SUBSCRIPTION TO URI A
EVENT PROXY SERVER (E.P.)
122
FILTER
124
SECURITY POLICY ENGINE
126
130
REQUEST TO SUBSCRIBE TO EVENT URI A/A
REQUEST TO SUBSCRIBE TO EVENT URI A/B
REQUEST TO SUBSCRIBE TO EVENT URI A/C
132
BROWSER
134
API SERVER
132
BROWSER

FIGURE 10

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2009/059300

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - H04M 3/42 (2010.01)
USPC - 370/259
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(8) - H04M 3/42 (2010.01)
USPC - 370/259

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 practicable, search terms used)
Patbase

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2006/0262915 A1 (MARASCIO et al) 23 November 2006 (23.11.2006) entire document	1-27
Y	US 6,445,776 B1 (SHANK et al) 03 September 2002 (03.09.2002) entire document	1-27
Y	US 6,507,875 B1 (MELLEN-GARNETT et al) 14 January 2003 (14.01.2003) entire document	5-11, 25-26

☐ Further documents are listed in the continuation of Box C. ☐

*	Special categories of cited documents:	"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
"A"	document defining the general state of the art which is not considered to be of particular relevance	"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
"E"	earlier application or patent but published on or after the international filing date	"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
"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)	"&"	document member of the same patent family
"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		

Date of the actual completion of the international search	Date of mailing of the international search report
30 December 2009	11 JAN 2010

Name and mailing address of the ISA/US	Authorized officer:
Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Blaine R. Copenheaver PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (July 2009)

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
10 September 2010 (10.09.2010)

PCT

(10) International Publication Number
WO 2010/101935 A1

(51) International Patent Classification:
H04L 12/66 (2006.01)

(21) International Application Number:
PCT/US2010/025943

(22) International Filing Date:
2 March 2010 (02.03.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

61/156,758	2 March 2009 (02.03.2009)	US
61/249,493	7 October 2009 (07.10.2009)	US
61/296,270	19 January 2010 (19.01.2010)	US

(71) Applicant (*for all designated States except US*): **TWILIO INC.** [US/US]; Pier 38, The Embarcadero, San Francisco, CA 94107 (US).

(72) Inventors; and

(75) Inventors/Applicants (*for US only*): **LAWSON, Jeffrey** [US/US]; Pier 38, The Embarcadero, San Francisco, CA 94107 (US). **WOLTHUIS, John** [US/US]; Pier 38, The Embarcadero, San Francisco, CA 94107 (US). **COOKE, Evan** [US/US]; Pier 38, The Embarcadero, San Francisco, CA 94107 (US).

(74) Agent: **SCHOX, Jeffrey**; 500 3rd Street #515, San Francisco, CA 94107 (US).

(81) Designated States (*unless otherwise indicated, for every kind of national protection available*): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: METHOD AND SYSTEM FOR A MULTITENANCY TELEPHONE NETWORK

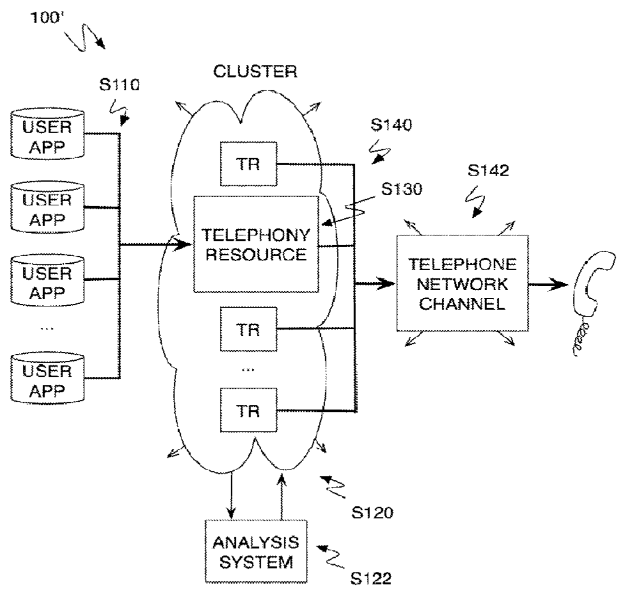

FIGURE 1

(57) Abstract: A method and system for operating a multitenancy telephony system including a call queue that stores call requests received from a plurality of users; an expandable and contractible telephony resource cluster that establishes call sessions for call requests; a analysis system that calculates capacity requirements of the system; a resource allocator that manages the scaling and operation of the telephony resource cluster; and a plurality of telephony network channels that are used as telephony communication channels for call sessions.

WO 2010/101935 A1

Published:

— *with international search report (Art. 21(3))*

METHOD AND SYSTEM FOR A MULTITENANCY TELEPHONE NETWORK

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Application number 61/156,758, filed 02 March 2009, entitled "A Method of Providing a Telephony Network for a Plurality of Users", U.S. Provisional Application number 61/249,493, filed 07 October 2009, entitled "Method and System for a Multitenancy Telephone Network", and U.S. Provisional Application number 61/296,270, filed 19 January 2010, entitled "Method and System for a Multitenancy Telephone Network", which are all incorporated in their entirety by this reference.

[0002] This application is related to prior application number 12/417,630, filed 02 April 2009, entitled " System and Method for Processing Telephony Sessions", which is incorporated in its entirety by this reference.

TECHNICAL FIELD

[0003] This invention relates generally to the telephony field, and more specifically to a new and useful multitenancy telephone network in the telephony field.

BACKGROUND

[0004] A telephone network has historically used a channel architecture for a telephone session or connection. This channel architecture has a foundation in the history of telephony; a physical wired connection or channel needed to be physically connected to make a telephone call. The concept of channels is still used today. Subscribers to a telephone network are conventionally required to pay on a per channel basis. Users that wish to have a public branch exchange ("PBX"), call center, or similar telephony application typically subscribe to a service and have a fixed number of channels that are available to them and only them. As the number of channels is part of their contract, they cannot exceed that number of channels (or else the call or telephone session will fail). Since most applications only see full capacity usage on rare occasions, the user often pays for more channels than are typically used.

[0005] In contrast to the channel based architecture of the telephone network, packet based network innovations have seen a rise in recent years, such as voice over internet protocol (VOIP), internet based applications, and internet-based telephony applications. With newer technology coming to the telephony field there are unique challenges arriving for handling the hardware and software capacity demands. Dedicated hardware and software often perform tasks during a telephone call session or even act as an intermediary system for connecting a caller to an internet based application. Telephone systems generally have higher performance expectations than a website based application. While a user of a website expects a website and software to take time to load and process information, a caller experiences frustration in delays or unresponsive interactions while on the phone. Additionally, the telephony applications are still dependent on the channel based telephone system, which adds yet another barrier to scalability. The telephone network and existing telephone application software and hardware architecture limit the growing capabilities of the telephony application field. Thus, there is a need in the telephony field to create a new and useful multitenancy telephone network. This invention provides such a new and useful system and method.

OBJECTS OF THE INVENTION

[0006] The present invention provides a system and method for providing a multitenancy telephone network for telephony applications. One objective of the present invention is to manage shared resource usage in a multi-user environment and to dynamically scale resources to satisfy capacity requirements. A related effect of this objective is that the sum total of the apparent number of resources available to each user is greater than the actual number of resources used to implement the multi-tenant telephone network. Another objective of the present invention is to efficiently use resources of a telephony platform by provisioning the processing and storage resources to satisfy capacity requirements, effectively leaving other unused resources for alternative applications, powered off for power saving, or any suitable functions. Another objective of the present invention is to make the use of a cluster of telephony resources transparent to an application of a user. This transparency is preferably preserved despite situations where operation of an application is distributed between a plurality of telephone service resources and may involve a plurality of telephone sessions on different channels. These and other objects of the invention are accomplished by the preferred embodiments of the invention, including a system for multitenancy telephone network, a method for operating a multitenant telephone network, a method of operating a dynamic telephone network, and a method of distributing calls between telephone hardware, each described in the following sections.

BRIEF DESCRIPTION OF THE FIGURES

[0007] FIGURE 1 is a flowchart representation of a preferred embodiment for the method of operating a multitenant telephone network;

[0008] FIGURES 2-4 are schematic representations of preferred embodiments of a system for a multitenancy telephone network;

[0009] FIGURE 5 is a schematic representation of a preferred embodiment of the invention using a cluster of call transcribers;

[0010] FIGURE 6 is a flowchart of a preferred embodiment for the method for operating a dynamic telephone network;

[0011] FIGURE 7 is a flowchart of a preferred embodiment of the invention implementing a conference call; and

[0012] FIGURE 8 is a flowchart of preferred embodiment of the invention receiving an incoming call.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0013] The following description of the preferred embodiments of the invention is not intended to limit the invention to these preferred embodiments, but rather to enable any person skilled in the art to make and use this invention.

1. System for a Multitenancy Telephone Network

[0014] As shown in FIGURES 2-4, the system 100 of the preferred embodiment includes a telephony resource cluster 110, a call queue 120, an analysis system 130, a resource allocator 140, and a plurality of telephony network channels 150. The telephony resource 110 cluster preferably includes a plurality of allocated telephony network channels 152 and/or a plurality of telephony resources 112 such as a plurality of call routers, a load balancer, and may additionally include a service application. The system functions to distribute the use of the network and system resources and dynamically adjust the system based on capacity requirements.

[0015] The telephony resource cluster 110 (or "cluster") functions as a scalable (expandable and/or contractible) collection of resources, where at least one resource is used to create a phone call session requested by a user. The cluster 110 is preferably a collection of hardware and/or software components that can dynamically adjust to satisfy processing and/or storage requirements. The cluster 110 preferably appears as a hardware and/or software cloud to outside devices, such that management of hardware allocation and usage is handled internally by the system. In one variation shown in FIGURE 2, the telephony resource cluster 110, is preferably a plurality of telephony resources 112 which functions to provide intermediary processing tasks for a call request or call session, such as establishing a call session, converting telephony instructions into call actions, transcribing a call, or directing a call. In another variation shown in FIGURE 3, the telephony resource cluster 110 is preferably a plurality of connections to allocated telephony network channels 152, where an allocated telephony network channel 152 is a channel of the allocated telephony network channels 152 that has been activated or designated as a channel available for a call session.

[0016] The telephony resources 112 are preferably software or hardware resources that are provisioned for a particular telephony processing task. There are preferably a plurality of telephony resources 112, and there may be a plurality of types of telephony resources that perform different dedicated tasks. A telephony resource 112 preferably includes a computer processors and/or computer storage devices. The telephony resource 112 may be a physical hardware device, virtual machine, a software program/routine, and/or any suitable combination to provide the processing and storage operations of a telephony resource 112. In some cases, a telephony resource 112 may include dedicated hardware or software. Since the telephony resources 112 share the basic functionally as either processing power or data storage, the core functionality of a telephony resource 112 may be reprovisioned such that the telephony resource 112 performs a different dedicated task. The resource allocator 140 (and more specifically the load balancer 142) preferably reprovisions telephony resources 112 to act as different parts of the resource cluster 110. For example, the cluster may include a number of text-to-speech servers and a number of call routers, but at some point in time there may be a low number of text-to-speech operations being performed and an increased number of telephony applications, and so a text-to-speech server is preferably reprovisioned as a call router. In one variation, the plurality of telephony resources 112 (i.e., the cluster 110) preferably includes a plurality of call routers 114. Additionally or alternatively, the cluster may include other hardware devices or software instances such as media processing systems, transcription systems, text-to-speech systems, call recorders, call data storage, or any suitable hardware (physical device or virtual machine) or software. The resource allocator 140 for the cluster preferably includes a load balancer 142 that manages the distribution of processing tasks and the operation of the plurality of telephony resources 112. Additionally, the cluster may include a service application and/or a call router network that can cooperatively resolve issues that result from using a plurality of resources.

[0017] The plurality of call routers 114 functions to initiate or receive calls from telephony devices and provide telephony application related processing. Preferably, the call routers connect to an application server, which is preferably the source of the call request. The plurality of call routers 114 is preferably a dynamic number of call routers 114 that can be adjusted according to capacity requirements. As stated above, in alternative embodiments the plurality of call routers 114 may be replaced by or combined with other suitable telephony hardware or software resources such as media processing systems, transcription systems, text-to-speech systems, or other specialized hardware or software resources that are used in a telephony application. In one example, a plurality of transcription hardware or virtualized resources may be used in place of call routers for transcribing phone calls, as shown in FIGURE 5. Additionally, a call router 114 may be reprovisioned as a media processing system, transcription system, text-to-speech system, or for any suitable process, and similarly any processor may be reprovisioned to serve as a call router. The number of hardware or software resources may additionally or alternatively be allocated or deallocated so that any desired number of resources in any suitable combination may be operated at any time. A hardware instance may be powered down, put into energy saving mode, or placed in any suitable state when deallocated. The telephony resources 112 may additionally or alternatively be operated as virtualized resources on a cloud computing platform (which may be operated by an outside party such as Elastic Compute Cloud operated by Amazon). When a telephony resources 112 such as a call router 114 is deallocated the virtualized resources may be returned to the vendor, given to other customers of the cloud computing platform, ending the virtualization of the resources, or any suitable process. A software instance may be quit or deleted when deallocated. The ratio of resources, such as the ratio of call routers to media processing systems, may be adjusted or maintained.

[0018] A call router 114 is preferably connected to a Public Switched Telephone Network (PSTN) device over the PSTN network, such that it can receive and make calls from PSTN-connected devices, such as landlines, cellular phones, satellite phones, or any other suitable PSTN-connected devices, as well as non-PSTN devices, such as Voice-Over-Internet-Protocol (VOIP) phones, SIP devices, Skype, Gtalk, or other Internet addressable voice devices. Thus the call routers 112 can preferably create connections to the telephone network of the distributed telephone controller. The call router 112 may alternatively or additionally function as or include a message router for use with telephony messaging such as SMS (Short Message Service) messages or MMS (Multi Media Messaging). The call router 112 can preferably connect to a messaging network, such that it can receive and send messages from SMS/MMS network devices, cellular phones, computers, smartphones, or any suitable SMS/MMS network devices. The call router 112 may also send or receive text messages, multimedia messages, emails, faxes and other suitable PSTN-compatible communication messages. The call router 112 preferably communicates with the application server using an application layer protocol, more preferably using the HTTP (Hypertext Transfer Protocol), or secure HTTPS (Hypertext Transfer Protocol Secure), protocol. The application server preferably hosts a telephony application, sound file, text file, a database, and/or any suitable media, resource or file that can be used by the call router for telephone interactions. The call router 112 may additionally generate call router resources. The call router resources are preferably accessible by the application server and other devices (such as other call routers) through a call router API. The call router resource functions as an addressable representation of call router meta-data, internal call router state, or the state of a given resource used by the call router. For example a call router 114 may record a call and save the recording as a call router resource.

[0019] Additionally, the telephony resource cluster 110 of the preferred embodiment may include a service application 116 that functions as a messaging component to coordinate functionality of an application that has been distributed across various call routers 114, hardware resources, and/or software resources. The service application 116 is preferably an internal resource that is used when normal operation of an application is prevented because the operation of an application is distributed amongst various hardware and software resources of the cluster 110. The service application 116 is preferably a messaging service that offers reliable messaging where a message is delivered to a particular destination (such as to another call router 114). The service application 116 may alternatively offer broadcasting messaging that announces a message without knowing who receives a message of if the message was received. As a first example, a hang-up service application 116 may be used to coordinate hanging up call sessions on different call routers 114. The hang-up service is preferably used to communicate to the appropriate call routers 114 to cancel outgoing calls when, for example, an application wants to dial a plurality of numbers but then hang up all unanswered calls once one of the calls is answered. As a second example, a multiple input service may gather and input commands from multiple telephone devices. So dual-tone multi-frequency (DTMF) input or voice commands may be issued by any caller and communicated to the application even if the calls are distributed over multiple call routers 114 within the cluster. This may be used in voting applications within a conference call. In this way, a telephone application does not need to actively account for processing and call handling being distributed within the cluster, and the hardware and software resources of the cluster preferably appear as a single entity to outside applications because of the internal service applications 116.

[0020] Additionally, the telephony resource cluster 110 of the preferred embodiment may include a call router network 118 that functions to allow a level of communication and synchronization between various call routers 114. The call router network 118 may additionally or alternatively be applied to other hardware or software resources. The call router network 118 is preferably used to access shared resources or as a communication channel. In one exemplary application, a voice over internet protocol

(VOIP) connection is established over the call router network 118 for mixing audio from various call routers. The VOIP connection is preferably used in implementing conference calls distributed over multiple call routers 114. As another example, the call router network 118 may additionally be used to stream audio from a call router to a realtime internet audio stream. As another example, the call router network 118 may be used to access data on another telephony resource 112 such as by using the call router API to access a call router resource. The service application 116 and the call router network 118 may additionally cooperate in synchronizing applications distributed within the cluster.

[0021] The call queue 120 of the preferred embodiment functions to manage a stack of call requests. The call queue 120 is preferably a list of outbound call requests that have not been serviced or been assigned necessary resources. The requests are preferably serviced at a rate suitable for the current capacity of the network 150 and telephony resource cluster 110. The servicing rate may alternatively be adjusted according to the capacity of the distributed telephony controller 144, the telephony resource cluster 110, and/or number of requests in the queue 120. A call request (such as one made by a telephony application) is preferably placed in the call queue 120 when capacity is exceeded or alternatively placed in the call queue 120 for every request or based on any suitable rule.

[0022] In one variation, an application preferably has associated user limits, in particular: an inter-call request rate limit (throttle) and a total limit (cap). The throttle and cap are preferably used to determine the positioning of requests in the call queue.

The limits may alternatively be assigned to an account, phone number, or any suitable entity. Telephony messages (e.e., SMS or MMS) are one variation of a call request that can additionally be placed in the call queue. Inbound and outbound telephony message can preferably be queued because inbound messages do not require immediate action unlike inbound calls. The SMS message is preferably sent after the request is serviced in the queue. SMS messages and/or MMS messages may alternatively be queued in a dedicated message queue. SMS message may have a rate limit (throttle) and total limit (caps) that varies from requests. Requests received at any rate from a user are preferably spaced in time in the call queue according to the throttle. There is preferably a latency enforced between call requests from an application. Requests of different users are preferably ordered in the queue in a staggered or alternating fashion as shown in FIGURE 6, but alternatively, users may have priority based on a service plan, first-come-first-serve policy, type of call request, and/or any suitable policy. The cap is preferably a limit on the total number of requests a user can make in a given amount of time. The user limits, handling, spacing, and/or ordering of the call queue 120 function to prevent one application from unfairly dominating the usage of the telephone network 150 or telephony resource cluster 110 at any one time. Additionally, applications may request access to telephony resources 112 as soon as possible or at some time in the future (e.g., a user schedules a call or calls for a later time). Additionally or alternatively, the user limits may be adjusted or set according to the needs of an application. An application may have particular requirements based on the nature or characteristics of the application of the user. The user limits are preferably set according to the contract and/or pricing model that a user selects or by any suitable means.

[0023] In another variation, the call queue 120 is dedicated to requests of a single user entity. In this variation, there is preferably a plurality of individually assigned call queues 120. Call requests are preferably organized into a call queue 120 for each user. Telephony message requests alternatively have a queue for each phone number. A user requests can preferably be added to the individually assigned queue 120 at any time. Each queue is preferably serviced (i.e., dequeued) on a schedule that considers the per-user limits (such as resource limits, system-wide limits, etc.). In other words the dequeuing occurs in an alternating fashion between the plurality of call queues 120. The individually assigned call queues may additionally be for particular resources, and the dequeuing preferably occurs according to the dequeuing rate of the particular resource. The dequeuing rate is preferably related to the capacity of the resource but may alternatively be based on any suitable criteria. As with the other queuing variations, queuing may alternatively occur according to any suitable queuing methodology such as randomly, in a round-robin fashion, with fair queuing, with weighted fair queuing, based on actual resource utilization, and/or any suitable methodology. As an alterantive to queuing based on account/phone number, call or message requests may be queued based on time, priority, usage history, or any suitable aspect. There may additionally be a control queue used to coordinate the dequeuing of individually assigned call queues (or message queues) 120.

[0024] As mentioned above, the call queue 120 may include an additional or alternative system for handling telephony messages (e.g., SMS or MMS messages). SMS messages preferably have additional limitations on their servicing rates and restrictions. SMS messages are preferably not only queued for sharing telephone network access with various users, but rates are also preferably implemented to prevent SMS messages from a single user from being rate limited, identified as spam. A call queue 120 for telephony messages may include at least two types of queues: a control queue and a phone number queue. The phone number queue preferably functions as a personal queue of a single user for telephone messages the user wants to send, and the control queue functions substantially similar to the multi-user queue described above for the call queue 120. The individually assigned call queue 120 may alternatively be used without the control queue, and the individually assigned call queue 120 may be based on account phone number or any suitable assignment. The control queue and phone number queue preferably functions to isolate the queuing of messages for a particular application and the messages of the plurality of messages. The content of the SMS message (the text) or MMS message (the multimedia) is preferably not stored in the call queue directly, and a reference to the SMS message content is preferably stored. This functions to reduce the load on the queue. The SMS/MMS content is preferably stored and accessed when the queued reference is serviced.

[0025] A queue popper 122 (i.e., a dequeuer) is preferably a software or hardware mechanism that functions to select call requests to service from the call queue The queue popper 122 preferably selects call requests at a preferred rate, but the queue popper 122 may alternatively select calls requests according to capacity or available resources, or a combination thereof. There may additionally be a plurality of queue poppers 122 that function to simultaneously select call requests from the call queue 120. The number of call poppers 122 may be variable. Additional or special queue poppers 122 may be used for the additional SMS call queues. The call queue(s) 120, the queue popper(s) 122, or any suitable combination are preferably used to control the throttling (or servicing rate) of the call requests. The throttling may be performed on a per-phone number, per-account (as in a multi-tenant application), and/or according to any call/message attribute.

[0026] The analysis system 130 of the preferred embodiment functions to analyze the system to predict resource requirements. The analysis system 130 preferably monitors a plurality of aspects of the system. The analysis system 130 may monitor the current capacity such as network or hardware operation levels or trends (increasing or decreasing); usage history such as logged data to find correlations in capacity (e.g., detecting patterns); queue length and queue entry latency; analysis of applications such as historical patterns from past usage of an application; and/or any suitable aspects. Patterns in capacity needs are preferably found related to the time of day, day of the week, yearly patterns, usage patterns (such as if an increase in capacity needs by one user indicates increase in capacity needs by other users), call location, call duration of calls, and/or any suitable indicator. The analysis system 130 preferably makes distinctions between inbound and outbound capacity for telephone network channels. The analysis system preferably generates data for the resource allocator 140, a distributed telephone controller 144, a load balancer 142, and/or additionally the call queue 120. The predictions or data from the analysis system may additionally be used for provisioning capacity of the distributed call controller, planning capacity requirements of the static capacity of the telephone network, the number of call routers, hardware or software resources within the cluster, and/or parameters of queue management. The analysis system 130 preferably compares expected and actual load, and provides data that is used to compensate for the variability in utilization of resources of the system.

[0027] The resource allocator 140 of the preferred embodiment functions to scale and manage the operation of the telephony cluster 110. The resource allocator 140 additionally preferably reprovisions telephony resources 112 of the cluster 110, allocates new telephony resources 112, deallocates telephony resources, and/or any other suitable allocation process. The resource allocator 140 may additionally control the provisioning of call queues and other devices of the system. The resource allocator 140 preferably uses data of the analysis system 130 in determining the provisioning and operation of resources. The resource allocator 140 preferably uses information from the analysis system 130 to predict required telephony resource 112 capacity. The resource allocator 140 preferably uses the predicted capacity requirements to determine how many hardware (physical or virtualized) or software resources need to be operating, and the resource allocator preferably allocates, deallocates, or reprovisions telephony resources 112 (e.g., call routers and/or other hardware or software resources) as necessary. The resource allocator 140 may additionally use startup time, operation cost, or other parameters of hardware and software resources when determining the number and ratio of resources to have allocated at a particular time. The resource allocator 140 also preferably keeps track of the quantity of resources currently available, and makes resource availability information available to other system components, including dequeuers, load balancers etc. Such resource availability information is preferably used by other system components to adjust operation of the system components. The resource allocator 150 preferably monitors the resources and reprovisions resources in real time.

[0028] The resource allocator 140 of the preferred embodiment preferably includes a load balancer 142 that functions to distribute processing tasks amongst the call routers and other hardware. The load balancer 142 of the preferred embodiment preferably optimizes the distribution of processing tasks such that the plurality of call routers 114 is operated at or near an optimal level. The operation of the call routers 114 may be optimized for performance, energy, cost, and/or any suitable conditions. The load balancer 142 preferably directs tasks (e.g., servicing of call requests/sessions) to appropriate call routers 142 (or telephony resource 112) as the tasks are created. A task is preferably an operation of a telephony application, but may alternatively be a call request or call session. In one example, one hundred call routers 114 may provide the call router tasks for one hundred telephony applications. In a second example, one hundred call routers 114 may each handle a single call session associated with one telephony application, such as for a conference call application with one hundred participants. The resource allocator 140 preferably sends notifications as to the current status of resources of the system (the load of resources, the number of resources, etc.) to the load balancer 142. The load balancer 142 distributes requests to currently available and running resources matching the requirements of the application being load balanced, based on data provided by the resource allocator 140.

[0029] The resource allocator 140 of the preferred embodiment may include a distributed call controller 144 that functions to controls usage and operation of the telephone network 150 by the system. The distributed call controller preferably manages the shared usage of the telephone network channels 150 by the plurality of telephony resources. The distributed call controller 144 may alternatively be a subset of multiple telephone networks if multiple network providers or carriers are used. The operation of the distributed call controller 144 preferably functions to operate an allocated number of channels for current capacity requirements of the telephone network 150. The allocated channels are preferably channels within the available static channel capacity that are in use or prepared for use. The distributed call controller preferably has less than or equal capacity as the static channel capacity at any given time. The capacity of the distributed call controller 150 can preferably be increased by allocating more resources of the telephone network to the call controller, and the capacity of the distributed call controller 144 can preferably be decreased by deallocating resources of the telephone network. As an example, a commodity hardware node may be added to the telephone network to run a telephony software stack during high capacity requirements. The distributed call controller 144 preferably uses the analysis system 130 to predict or respond to the desired capacity requirements. The telephone network 150 may additionally be divided into inbound channels, outbound channels, and bidirectional channels that can be used for receiving calls, making calls, and both, respectively. The telephone network 150 may further include SMS or MMS inbound and outbound channels. The distributed call controller 144 preferably manages the usage of the type of channels according to predicted usage. The bidirectional channels are preferably used for flexibility in capacity requirements. As one example, if inbound call load is expected to be high, then outbound calls are preferably directed to outbound channels to leave more capacity for inbound calls. The distributed call controller 144 may additionally manage the number and usage of allocated channels according to subscription or contracts from network providers. Channels may be used allocated or deallocated to ensure that volume pricing thresholds or other network conditions are satisfied.

[0030] A telephone network with a static number of channels 150 is preferably the base infrastructure for providing users with telephone network access. Telephony sessions are preferably communicated over the telephone network and the telephony sessions preferably include telephony voice sessions and/or text/media messaging (telephony messaging). The static number of channels is preferably the total number of concurrent telephony sessions or calls that can be supported at one time. The number of channels is typically limited by the number of interconnections available to a specific carrier or network. The telephone network 150 may alternatively be composed of multiple carriers or network providers or the Public Switched Telephone Network, but the plurality of carriers or networks is preferably managed or handled as one telephone network. The static number of channels is preferably a set number for a period of time (usually based on a contract with a telephone company), and the number is preferably large enough to provide sufficient capacity. The static number of channels preferably determines the capacity of a network and the ability of the telephone network to connect with other networks. The operation of the telephone network is preferably handled by providing applications access to a channel of the telephone network. The telephone network may have a given number of channels not being used at any given time. In one variation, the telephone network may alternatively operate unused channels in an unused-mode. The unused mode may be a full or partial hardware power down mode, a hardware sleep mode, a secondary use (such as for non-crucial uses that can preferably be interrupted with minimal adverse effects), and/or any suitable way. The unused mode would function to reduce operation cost and/or maximize the utility of unused capacity. The telephony network channels 150 are preferably Public Switched Telephone Network (PSTN) connections but may alternatively be Session Initiation Protocol (SIP) trunks or any suitable device to create a telephony network connection to a telephony device.

2. Method of Operating a Multitenant Telephone Network

[0031] As shown in FIGURE 1, the method 100' of operating a multitenant telephony network of the preferred embodiment includes the steps of multiplexing call requests of a plurality of users to a telephony resource S110, creating a first call session from the call request through the telephony resource S130, and multiplexing the call session with a plurality of additional call sessions to a telephony channel S140. The method 100' functions to create an efficient and scalable network system for resource intensive telephone applications. The telephony resource is preferably part of a telephony resource cluster. The telephony resource cluster preferably scales to satisfy immediate capacity demands which functions to reduce operation cost and allow a wide variety of applications to use the multitenant telephony network due to the ability to handle a wide spectrum of network loads. Additionally, the method 100' functions to allow the operation of a telephony application to be distributed between a variety of multi-user, shared resources (e.g., a telephony resource) such that the specific goals of telephone applications are not limited by the multitenant telephony network. The method 100' of the preferred embodiment is preferably implemented by a system described above, but may alternatively be implemented by any suitable system.

[0032] Step S110, which includes multiplexing call requests of a plurality of users to a telephony resource, functions to share the use of a telephony resource between a plurality of users. A single telephony resource is preferably shared between a plurality of users/applications. The multiplexing preferably occurs in a form of time division multiplexing in which call requests are sent to telephony resource in an alternating fashion. The time division multiplexing is preferably based on completion of complete call sessions or processes. In other words, users take turns using the telephony resource to create a call session and run an application. For example, a first customer preferably has a call request serviced by a telephony resource and upon completion of the call session of the call request, a second user may have a call request serviced by the same telephony resource. A call request is preferably received from a user or more specifically a telephony application residing on an external server, but the call request may alternatively be sent from any suitable source. The call request is preferably received over a packet-based communication channel, in other words a non-direct communication channel. In one variation, the call request is preferably received in a HTTP or HTTPS message but may alternatively be received through any suitable application communication protocol. Step S110 may additionally include queuing a call request of a user S112, which functions to gate or prioritize incoming call requests. The call queue is preferably used for outbound requests, while inbound calls are preferably handled immediately (or else the call session will most likely fail). Alternatively, an inbound call may be queued for full service, with a "ringing" audio played back while call is waiting in the queue to be fully serviced. A queue may, however, be used for inbound telephony messages because telephony messages such as SMS messages and MMS messages will be resent if not received on the first attempt. The call queue is preferably a list of pending call requests from a plurality of users. An additional queue may additionally or alternatively be used for telephony messages. The call requests are preferably ordered within the queue in a way that balances access to resources. Each user (e.g., account, application, or phone number) is preferably assigned an inter call request limit (a throttle) and a limit on the maximum number of call requests that can be made in a specified amount of time (a cap). Call requests are preferably selected for servicing at a specified rate or by a device (i.e., a queue popper), which may be selecting calls based on current load on the telephony resource cluster. The queue may alternatively be operated in any suitable variation such as those described above. A queue may be assigned to each user or phone number. Queuing may alternatively occur according to any suitable queuing methodology such as randomly, in a round-robin fashion, with fair queuing, with weighted fair queuing, based on actual resource utilization, and/or any suitable methodology. A load balancer preferably distributes call requests to a telephony resource that has the least capacity. The load balancer and the call request queue preferably cooperatively distribute the load as described above.

[0033] As an additional step, method 100' preferably includes provisioning resources of the telephony resource cluster S120, which functions to scale the capacity of the telephony resource cluster to adequately multiplex a call request to a telephony resource. Step S120 may include reprovisioning an existing telephony resource of the telephony resource cluster, allocating additional resources to the telephony resource cluster, and/or deallocating resources of the telephony resource cluster, and/or re-allocating resources from one type of resource to another in realtime. The telephony resource cluster preferably includes a plurality of telephony resources performing various functions or operations as described above. For example, the telephony resource cluster may include a plurality of call routers, transcription systems, media processing systems, and text-to-speech systems. A telephony resource preferably is composed of a computer processor and/or storage resources for a first purpose. As part of S120, a resource of the telephony resource cluster a processor and/or storage device of a telephony resource is preferably reprovisioned for a new second purpose. For example a text-to-speech may be reprovisioned to function as a call router at times when more calls need to be served. Additionally, more resources may be allocated or deallocated which may include adding new resources to the system and/or activating resources, or re-allocating resources from another customer of a shared resource environment. The resources are preferably those provided by a multitenant shared virtualized computing environment such as a cloud hosting provider (i.e., a web service that provides resizable compute capacity that allows a user to boot a machine image to create a virtual machine resource), but may alternatively be physical machines either co-located or distributed. For example, a number of resources may be operating in a powered down state. When the more capacity is required, the resources may be turned on/booted (i.e., allocated) to serve as a new resource of the telephony resource cluster. Similarly, when the telephony resource cluster has more capacity than is currently required a resource may be powered down, returned to a pool of resources for use by other companies (i.e., deallocated), or any suitable action to end current use of the resource.

[0034] Additionally, Step S120 may include analyzing resource capacity requirements S122 which functions to collect data on real-time or imminent capacity requirements. Data may be collected from the call request queue, from stored history on capacity requirements, current load of the telephony resource cluster, data from an analysis of applications, or any suitable source of predicting capacity requirements. Data from the call request queue may provide information such as number of pending call requests, the type or details of the call request, or any suitable queue related information. Stored capacity history preferably provides insight to capacity patterns such as temporal patterns throughout the day, week, or year. Current load of the telephony resource cluster preferably provides information such as the current number of resources of the telephony resource, number of available resources of the telephony resource, the division of type of resources, the number of deallocated resources, the number of telephone network channels, etc. Application analysis data preferably is data from the telephone applications of users on expected or predicted capacity requirements. An analysis is preferably performed on the operation of the application and or gathered from a user on the expected capacity requirements of the application such as number of calls, peak time for calls, what type of calls (e.g., conference calls, SMS messages etc.). The analysis information is preferably used to control the provisioning, allocation, and deallocation of resources of Step S120. Additionally, after analyzing the capacity requirements, other components of the system such as the telephony resource cluster, telephony resources, call queue, dequeuers, resource allocator are preferably notified of relevant analysis information. Particular analysis information may be specifically sent to a component. For example, the load balancers and the dequeuers are preferably informed about available resources and adjust operation according to the capacity information.

[0035] Step S130, which includes creating a first call session from the call request through the telephony resource, functions to convert the call request into a call session using the telephony resource. Step S130 preferably additionally includes additional processing and steps specific to a particular application. In a preferred variation, a call router preferably processes telephony instructions of a call request to identify the destination phone number and then establishes a connection to the destination phone number as part of Step S140. A transcription server may initiate recording or prepare to record a conversation of the call session.

[0036] Step S140, which includes multiplexing the call session with a plurality of additional call sessions to a telephony channel, functions to establish a telephone network connection to a telephony device. The telephony channel is preferably a PSTN (Public Switched Telephone Network) connection. This may be a physical wire or some interfacing infrastructure to connect to the PSTN. In some cases the concept of a channel is preferably subscribed to or rented from a telephone network. In one alternative, a SIP (Session Initiation Protocol) trunk may be used as an internet based gateway to a telephone network. The multiplexing preferably occurs in a form of time division multiplexing in which call sessions are connected to telephony channel in an alternating fashion. The time division multiplexing is preferably based on completion of complete call session. For example, a particular network channel may first be utilized for a call session of a first user, and upon completion of the call, a second call session may be established with the particular network channel for a second user. As part of Step S140, the telephony channels may additionally include provisioning telephony channels S142. This functions to adjust the number of available telephone network capacity of the system. By provisioning gateways to the telephone network (e.g., call routers or SIP trunks), channels or gateways to channels may be allocated or deallocated. Such scaling of telephony network channels preferably allows operation near the current telephone network capacity requirements. If such scalability was not in use then there would be a set limit on the number of channels that could be simultaneously used.

3. Method of Operating a Dynamic Telephone Network

[0037] As shown in FIGURE 6, the method 200 of providing a telephony network of the preferred embodiment includes the steps of operating a telephone network with a static number of channels S210, providing telephone network channel access to a plurality of users S220, and managing usage of channels to allow a user to access a number of channels that exceeds normal operation S230. The method functions to allow the operator of the telephone network to offer high capacity to a plurality of users, without a reduction in quality or reliability of services based on usage. This method is preferably implemented on a system substantially similar to the one described above, but any suitable system may alternatively be used. The method may additionally be used in combination with the methods herein described. The method 200 further functions to allow users to use the telephone network without a specific concern about the number of channels required for operation. The users of the telephone network are preferably operating telephony applications such as call centers, Private Branch Exchanges (PBX), phone trees, telephony phone applications, VOIP services, SMS or MMS services, and/or any suitable telephony application. The operators of the telephone network are preferably a telephone service provider such as a telephony platform provider (e.g., a internet-telephone platform provider), a telephone company (e.g., owners of a telephone network such as AT&T), and/or any suitable party. In a variation of the preferred embodiment, the method 200 may additionally include a distributed call controller, a call queue, and/or the step of assessing capacity requirements.

[0038] Step S210, which includes operating a telephone network with a static number of channels, functions to be the base infrastructure for providing users with telephone network access. The static number of channels is preferably the total number of concurrent telephony sessions or calls that can be supported at one time. The number of channels is conventionally limited by the number of interconnections available to a specific carrier or network. The telephone network may, however, be composed of multiple carriers or network providers or the Public Switched Telephone Network, but the plurality of carriers or networks is preferably managed or handled as one telephone network. The static number of channels is preferably a set number for a period of time (usually based on contract with a telephone company), and the number is preferably large enough to provide sufficient capacity. The static number of channels is preferably an indication of the capacity of a network and the ability of the telephone network to connect with other networks. The operation of the telephone network is preferably handled by providing users access to a channel of the telephone network. The telephone network may have a given number of channels not being used at any given time. In one variation, the telephone network may alternatively operate unused channels in an unused-mode. The unused mode may be a full or partial hardware power down mode, a hardware sleep mode, a secondary use (such as for non-crucial uses that can preferably be interrupted with minimal adverse effects), and/or any suitable way. The unused mode would function to reduce operation cost and/or maximize the utility of unused capacity.

[0039] As an additional alternative to the preferred embodiment, the method may include operating a distributed call controller as a subset of the telephone network S212. The distributed call controller may alternatively be a subset of multiple telephone networks if multiple network providers or carriers are used. The operation of the distributed call controller preferably functions to operate an allocated number of channels for current capacity requirements of the telephone network. The distributed call controller preferably has less than or equal capacity as the static channel capacity at any given time. The capacity of the distributed call controller can preferably be increased by allocating more resources of the telephone network to the call controller, and the capacity of the distributed call controller can preferably be decreased by deallocating resources of the telephone network. Access to the telephony network is preferably facilitated by virtualized hardware or software (such as call routers or SIP trunks). Allocation of more resources of the telephone network may additionally include a virtualization of a device to access a telephony network. For example a virtualization of a network access channel may be added to add further access capacity to the telephony netowork. As another example, a commodity hardware node may be added to the telephone network to run a telephony software stack during high capacity requirements.

[0040] Step S220, which includes providing telephone network channel access to a plurality of users, functions to allow a plurality of different parties to access the channels of the telephone network. The users preferably subscribe to a service of the operator of the telephone network. The users of the telephone network are preferably operating telephony applications such as call centers, Private Branch Exchanges (PBX), phone trees, Interactive Voice Response (IVR) applications, internet-telephony applications, VOIP services, and/or any suitable telephony application. The user preferably does not subscribe to the service based on any specific number of channels. From the viewpoint of the user, the number of channels is preferably infinite or an irrelevant point for the operation of an application of the user. The user is preferably presented a per usage or time perspective (e.g., pricing and/or application usage perspective), while the telephone network is being operated on a per channel basis. The operator of the telephone network preferably converts costs associated with the operation of the telephone network (e.g. fixed capital costs of leasing from a telephone company or operation cost) into variable costs for the users. The access to the telephone network is preferably operated, leased, and/or on contract from a telephone company (such as AT&T) by a per channel basis. A lease agreement or contract may alternatively be negotiated to minimize per-channel (capacity) cost and preferably emphasize per usage or per time costs, or alternatively, any suitable leasing agreement or contract may be used. Users preferably pay by usage, a flat rate for a time period, per minute, a combination of usage and time charges, and/or any suitable pricing model.

[0041] Step S230, which includes managing usage of channels to allow a user access to a number of channels that exceeds normal operation S130, functions to provide high capacity capabilities to users while ensuring that the quality and reliability of the telephone network is not aversely affected by the usage of other users. An individual user of the plurality of users is preferably allowed to use a number of channels greater than an equal division between the plurality of users of the static number of channels. The sum total of the maximum number of channels an individual user uses at given times may preferably be greater than the static number of channels. The given times where an individual user has access a maximum number of channels is preferably when demand on the telephone network by other users is low. Usage of the telephone network and the telephony resource cluster is preferably time based multiplex based on the completion of telephony sessions (i.e., users share the use of the resources and network). In a simplified example, a telephone network has 10 channels available and there are five users. When distributed uniformly, the users would each have 2 channels available for usage, but in one preferred embodiment all five users may have access of up to 10 channels each, assuming no other user is using the channels. During regular use of the telephony network, the user still has the ability to access the maximum number of telephone network channels but the call requests are preferably gated by user limits implemented by a call queue. In another example extending on the above example, analysis might indicate that 4 users may use 2 channels at a given point of time, thus 8 may be available to the 5th user, while keeping capacity available for the first 4 users. Managing the usage of the channels preferably includes managing the usage of resources such as by: managing a call queue, enforcing user limits, predicting and/or analyzing usage and capacity requirements, adjusting capacity based on the capacity of the distributed call controller, and/or any suitable steps of managing the resources of the telephone network. Capacity of the distributed call controller may additionally be controlled or affected by predictions and analysis and user limits may additionally be affected.

[0042] The method of the preferred embodiment may additionally include the step of managing a call queue of requests from the plurality of users S232. Step S232 functions to prioritize the handling of call requests from users. The call queue is preferably a program or hardware managed stack that is operated as part of a control architecture of the telephone network. The control architecture preferably manages the telephone network and usage by the plurality of users. The call queue is preferably a list of call requests awaiting service by the telephone network including telephony voice session requests and/or SMS/MMS message requests. The requests are preferably serviced at a rate suitable for the current capacity of the network and for each user. The servicing rate may alternatively be adjusted according to the capacity of the distributed call center or number of requests in the queue. A user request is preferably placed in the call queue when capacity is exceeded or alternatively placed in the call queue for every request or based on any suitable rule. A user preferably has associated user limits, in particular: a call rate limit (throttle) and a total limit (cap). The throttle and cap are preferably used to determine the positioning of requests in the call queue. Requests from a user are preferably spaced in time in the call queue according to the throttle. Requests of different users are preferably ordered in the queue in a staggered or alternating fashion as shown in FIGURE 6, but alternatively, users may have priority based on a service plan, first-come-first-serve policy and/or any suitable policy. The cap is preferably a limit on the total number of requests a user can make in a given amount of time. Subsequent requests are preferably schedule for a later time according to the cap, but requests exceeding the cap may be handled in any suitable manner. For example, if user can make one call per second, and the user requests 100 calls, they will be scheduled equally over the next 100 seconds. Note that this cap can be described as the number of calls / time frame (1/second), or the required latency between calls in the queue (1 second). The user limits, handling, spacing, and/or ordering of the call queue function to prevent one user from unfairly dominating the usage of the telephone network at any one time. In the variation of SMS/MMS message requests, the rate of individual users is considered to prevent message filtering by a network. For the SMS/MMS variation the requests may additionally be queued in a control queue and a phone number queue. The contents of the SMS/MMS messages are preferably stored and a reference to the contents of a message is queued which functions to reduce the load on the queue. A plurality of cache servicing ports or pointers are preferably used. The servicing ports are preferably a software and/or hardware control mechanism for operating a call request from the call queue. A servicing port preferably takes a request from the call queue and connects the corresponding user application or user to a telephone network channel. The servicing port may be a direct connection, but may alternatively be a hardware or software resource such as a call router in a cluster as described above. The servicing ports are preferably less than the static number of channels to allow capacity for incoming calls, but the servicing ports may alternatively be equal to the static number of channels. In one example where there are 1000 channels of the telephone network, there may be 500 service ports. This would leave 500 channels free for incoming calls. Additionally, users may request access to telephony resources as soon as possible or at some time in the future (e.g., a user schedules a call or calls for a later time). A queue popper is preferably a software or hardware mechanism responsible for selecting a call from the call queue to service. There may additionally be a plurality of queue poppers to select calls from a call queue. Additionally or alternatively, the user limits may be adjusted or set according to the needs of a user. A user may have particular requirements based on the nature or characteristics of the application of the user. The user limits are preferably set according to the contract and/or pricing model that a user selects or by any suitable means.

[0043] The method of the preferred embodiment may additionally include the step of predicting capacity requirements for the distributed call controller S234. Step S234 functions to assess indicators that correlate to the number of telephone network channels needed at a later point. The predicting of capacity is preferably accomplished by programmatically or mathematically (through pattern detection or any suitable algorithm) analyzing current and past information but any suitable method may alternatively be used. Patterns in capacity needs are preferably found related to the time of day, day of the week, yearly patterns, usage patterns (such as if an increase in capacity needs by one user indicates increase in capacity needs by other users), call location, call duration of calls, and/or any suitable indicator. The predictions of Step S234 may additionally be used for realtime provisioning, deprovisioning, and/or reprovisioning capacity of the distributed call controller or planning capacity requirements of the static capacity of the telephone network.

[0044] The method of the preferred embodiment may additionally include the step of reacting to capacity needs of the call queue S236. Step S236 functions to use the call queue and other current capacity indicators to adjust the distributed call controller for the current capacity requirements or anticipated near term requirements. The call queue is preferably assessed through software or alternatively by any suitable monitoring of the call queue. The number of calls currently in the queue, the total number of users currently using the telephone network, incoming calls (that may be not be queued), the frequency of user requests, and/or any suitable characteristic of the telephone network or the call queue preferably cause a reaction to the capacity needs. The reaction is preferably for current overall capacity needs but may alternatively be for current capacity needs of an individual user or any suitable party. The reactions may include adjusting the settings of the call queue (such as call queue service rate or ordering), modifying user limits, adjusting capacity of the distributed telephone controller, and/or any suitable action. In one example, a call queue may have many calls scheduled for 100 seconds after the current time, the distributed call controller may increase capacity to accommodate the anticipated capacity requirements.

[0045] The method of the preferred embodiment may additionally include the step of analyzing capacity needs of a user and predicting the telephone network capacity needs S238. Step S238 functions to detect individual capacity needs to determine total capacity requirements of the telephone network. Capacity needs of a user are preferably acquired by analyzing a telephony application of a user. Part of the analysis preferably includes detecting periodic events that indicate capacity needs of an individual application. An example of such an event might be an application associated with a weekly TV show where callers call in around the air time of the show. The analysis may alternatively or additionally include detecting typical call duration for an individual application. Some applications may only be used for a brief amount of time (such as when a short message is played), while other applications may require longer durations of use (such as when a user must navigate a long phone tree). Additionally, application history may be used to determine usage patterns such as by monitoring maximum, minimum, and/or average capacity requirements, frequency of requests, duration of requests, number of SMS messages sent in a particular time duration, and/or any suitable call characteristic. Usage characteristics of the individual applications of users are preferably combined with the usage characteristics of the other users to determine the total usage characteristics and capacity needs of the telephone network. Preferably, the code of the application is preferably analyzed to assess the functionality and usage patterns of the application. The application code or operation is preferably programmatically analyzed, but any suitable method may be used. Alternatively, the user and/or a second party may characterize the application and/or telephony service of the user. This characterization is preferably performed by the user while signing up, and preferably includes user expectations for the frequency of use, times of use, duration of calls, and/or any suitable characteristic of the application. The user may additionally prioritize when capacity should be highest for their application. Any suitable steps may be used to analyze an individual application.

[0046] As an additional alternative to the preferred embodiment, the method may include adjusting capacity of the distributed call controller S240. Step S240 functions to change the number of active channels of the telephone network to appropriately handle the capacity requirements. Step S240 is preferably used in combination with Step S212, which includes operating a distributed call controller. The adjustments to the distributed call controller adjust the capacity capability that the operator offers. The capacity is preferably adjusted based on the management of the usage of channels of the telephone network. The capacity is more preferably adjusted based on the predictions and analysis of Steps of S234 and/or S236, but may alternatively be adjusted in cooperation with Step S232, Step S238, and/or for any suitable reason. When more capacity is needed, more resources, such as CPU, RAM, DISK, etc., capable of handling simultaneous channels or providing more channels, are preferably allocated to the distributed telephone controller, and conversely when less capacity is required, resources are preferably deallocated from the distributed telephone controller. The adjustment of capacity is preferably made to handle the expected or predicted capacity. The static capacity of the telephone network may alternatively or additionally be adjusted. As the telephone network capacity is typically less flexible. Adjustments to the telephone network capacity are preferably made for long-term capacity needs (e.g., on a per month basis). Any suitable adjustment to the system for more or less capacity may alternatively be used.

4. Method of Distributing Calls Between Telephony Hardware

[0047] As shown in FIGURES 7-8, the method 300 of distributing calls between telephony hardware of the preferred embodiment includes the steps of queuing a call request S310, selecting a load balancing call router S320, and connecting a call with the selected call router S330. The method functions to balance usage of resources used in a telephony application. This method is preferably implemented on a system substantially similar to the one described above, but any suitable system may alternatively be used.

[0048] Step S310, which includes queuing a call request, functions to manage a call request until the necessary resources are available to service the call. A call request is preferably instantiated by a telephony application, a call router, a telephony device, and/or any suitable source of a call request. The call request may additionally be a SMS or MMS message request. The call request is preferably outgoing. An incoming call is preferably viewed as a more urgent call request than an outgoing request, and an incoming call may not be queued but alternatively may be passed directly to an available resource. Alternatively incoming call requests (call session initiations) may be queued, but since incoming calls have more immediacy they are preferably prioritized or the system must be have short queuing wait where a short wait is less than the time it would take for an incoming call to fail. The incoming call may alternatively be placed near the front of a queue or positioned in the queue according to separate rules appropriate for the higher priority of the call request. Similarly, a synchronous outgoing call request may be queued with high priority. A synchronous call is a call that another caller is relying on to proceed, as opposed to a new call initiated by an application in which a user will not notice a delay. Call requests are preferably ordered in the queue according to rules based on the throttle, caps, real-time urgency (priority) and/or any suitable factors.

[0049] Step S320, which includes selecting a load balancing call router, functions to identify a call router that should handle the call to preferably optimize the operation of a telephone resource cluster. The selected call router is typically the call router with the least load, but may alternatively be selected to optimize cost, energy usage, processing capability, and/or any suitable variable. Step S320 may additionally be applied to other hardware or software resources in addition to or alternatively to a call router. Call routers of a telephone resource cluster may have variable capacity and performance depending on hardware and/or software specs. The variability between the plurality call routers is preferably considered in selecting a call router. A load balancer substantially similar to the one described above is preferably the component that implements step S320, though step S320 may be performed by any suitable device. The load balancer is preferably capable of allocating and deallocating resources of the cluster, and so resources may be allocated and/or deallocated as a substep of S320. The resource allocator can preferably allocate and deallocate call routers, hardware resources, and/or software resources. The resources are preferably allocated or deallocated based on current or predicted utilization, but the resources may alternatively be allocated or deallocated as a function of other resources. For example, one media processing resource may be allocated (e.g., operating) for every five call routers. The selection of a load balancing call router preferably uses data from an analysis system. So that the step of selecting a load balancing call router may include selecting a call router that will balance load at a future time.

[0050] Step S330, which includes connecting a call with a selected call router, functions to pass control of the call to the specified resource. For an outgoing call, a call router preferably connects through a telephone network to the designated phone number. For an incoming call, the call router preferably connects to the specified telephony application; PSTN-connected device(s), such as landlines, cellular phones, satellite phones, or any other suitable PSTN-connected devices; non-PSTN devices, such as Voice-Over-Internet-Protocol (VOIP) phones, SIP devices, Skype, Gtalk, or other Internet addressable voice devices; and/or any suitable device associated with the number of the incoming call.

[0051] The method of the preferred embodiment may additionally include networking call routers that have a shared application S340. Step S340 functions to allow communication between multiple call routers. This is preferably useful in situations where functionality of an application is distributed over multiple resources (e.g., multiple call routers). The network preferably allows sharing of resources between call routers. Audio channels of call routers may additionally be mixed and shared between call routers. A VOIP channel is preferably formed over the network for bridging audio of different call routers. For example, a conference call may use the network to bridge audio of multiple call sessions from different call routers.

[0052] The method of the preferred embodiment may additionally include synchronizing applications with a service application S350. The service application functions to monitor an application distributed over a call router cluster and coordinate operation of the application. The service application may additionally be used to share state information between the call routers. The service application preferably provides a specific functionality such as a hang up service or a multiple input service as described above. Any suitable application may be implemented by the service application such as input-gathering, multi-dialing, call splitting, call merging, and any suitable feature. Any number of service applications may be used.

[0053] As a person skilled in the art will recognize from the previous detailed description and from the figures and claims, modifications and changes can be made to the preferred embodiments of the invention without departing from the scope of this invention defined in the following claims.

CLAIMS

We Claim:

1. A system for operating a multitenancy telephony system comprising:
   - a call queue that stores call requests received from a plurality of users;
   - a scalable telephony resource cluster that establishes call sessions for call requests;
   - an analysis system that calculates capacity requirements of the system;
   - a resource allocator that manages the scaling and operation of the telephony resource cluster; and
   - a plurality of telephony network channels that are used as telephony communication channels for call sessions.
2. The system of Claim 1, wherein the call requests are ordered in the call queue according to a user inter-request rate limit and a user cap on the maximum number of requests in a specified time for a user.
3. The system of Claim 2, wherein the analysis system receives call queue data to calculate capacity requirements and wherein the resource allocator additionally manages allocation and operation of a plurality of call queues.
4. The system of Claim 1, wherein a call request is received from a telephony application, and wherein the analysis system analyzes the telephony application for capacity requirements, accesses past capacity requirement data, and monitors current capacity loads to calculate the capacity requirements of the system.

5. The system of Claim 1, wherein the call queue includes an additional queue for telephony message requests.

6. The system of Claim 1, wherein the telephony resource cluster includes a plurality of allocated telephony network channels that are a subset of the plurality of telephony network channels; and the resource allocator includes a distributed call controller that scales the number of allocated telephony network channels and connects call requests to a channel of the allocated telephony network channels.

7. The system of Claim 1, wherein the telephony resource cluster includes a plurality of telephony resources.

8. The system of Claim 7, wherein the plurality of telephony resources includes a plurality of call routers, wherein a call router makes and receives telephony communications through the telephony network channels and communicates with an application server of the user using an application layer protocol

9. The system of Claim 8, further comprising a call router network that is a data channel between at least two call routers.

10. The system of Claim 8, further comprising a service application that coordinates the operation of a telephony application that includes at least two call sessions on at least two call routers.

11. The system of Claim 8, wherein the telephony resource cluster includes a plurality of allocated telephony network channels that are a subset of the plurality of telephony network channels; and the resource allocator includes a distributed

call controller that scales the number of allocated telephony network channels and connects call requests to a channel of the allocated telephony network channels.

12. A method for operating a telephony network comprising:

- multiplexing call requests of a plurality of users to a telephony resource;A
- creating a first call session from the call request through a telephony resource; and
- multiplexing the call session with a plurality of additional call sessions to a telephony channel.

13. A method for providing a telephony network comprising:

- operating a telephony network with a static number of channels;
- providing telephone network access to a plurality of users; and
- managing usage of channels to allow a user to access a number of channels that exceeds normal operation.

14. A method of distributing calls between telephony hardware comprising:

- queuing a call request;
- selecting a load balancing call router; and
- connecting a call with the selected call router.

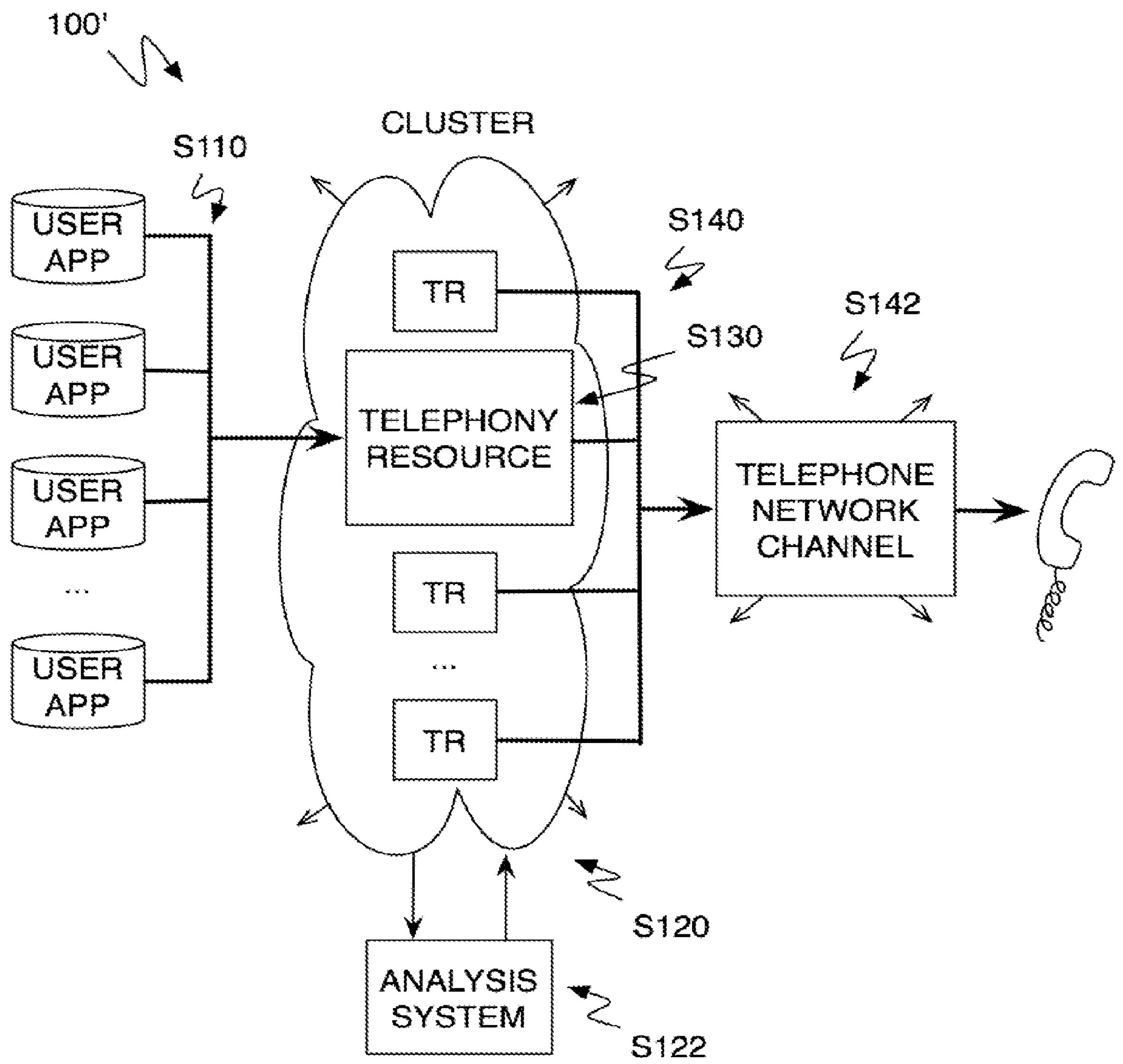
100'
CLUSTER
S110
USER APP
USER APP
USER APP
...
USER APP
TR
TELEPHONY RESOURCE
TR
...
TR
S140
S130
S142
TELEPHONE NETWORK CHANNEL
S120
ANALYSIS SYSTEM
S122

FIGURE 1

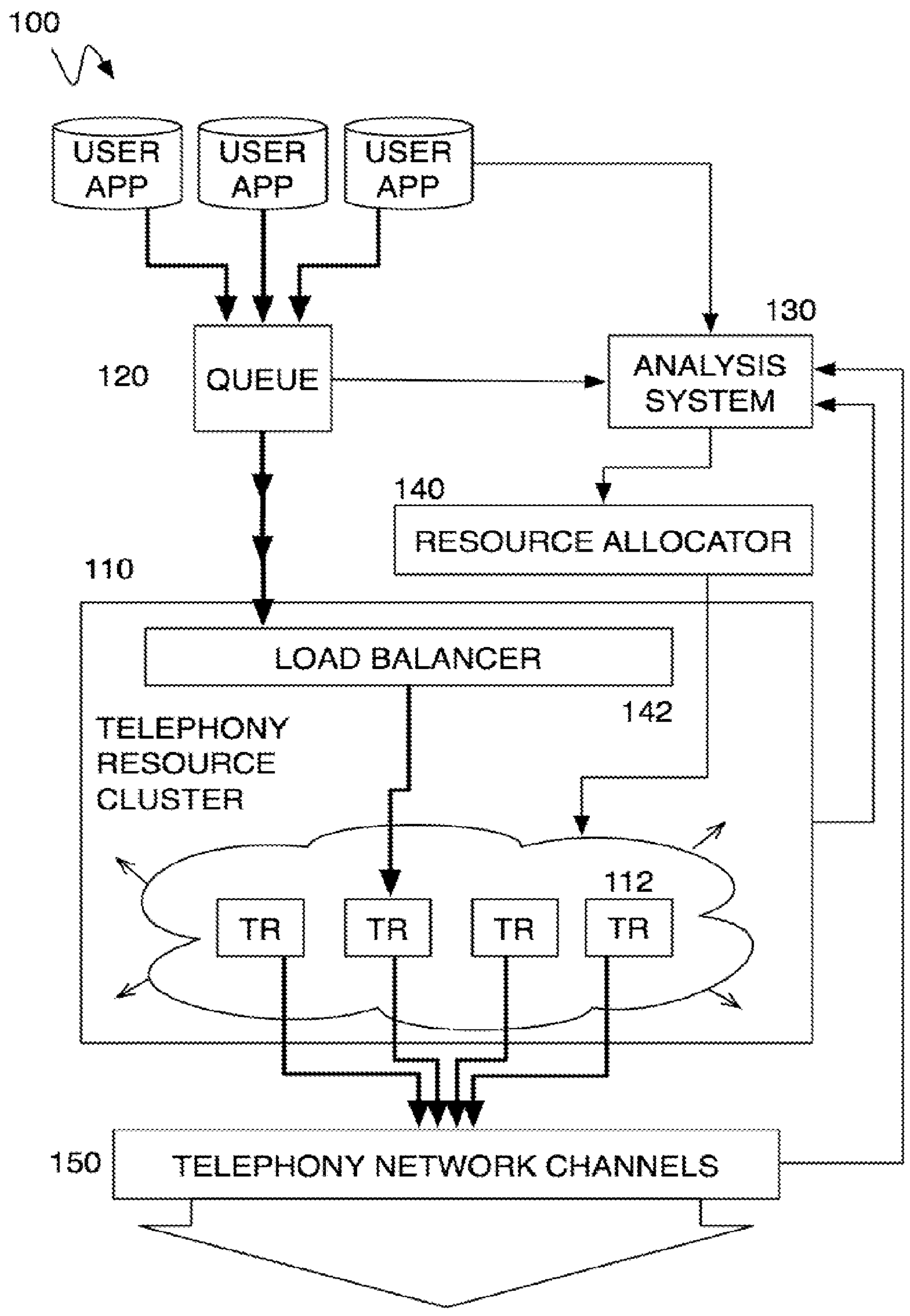
100
USER APP
USER APP
USER APP
130
120
QUEUE
ANALYSIS SYSTEM
140
RESOURCE ALLOCATOR
110
LOAD BALANCER
142
TELEPHONY RESOURCE CLUSTER
112
TR
TR
TR
TR
150
TELEPHONY NETWORK CHANNELS

FIGURE 2

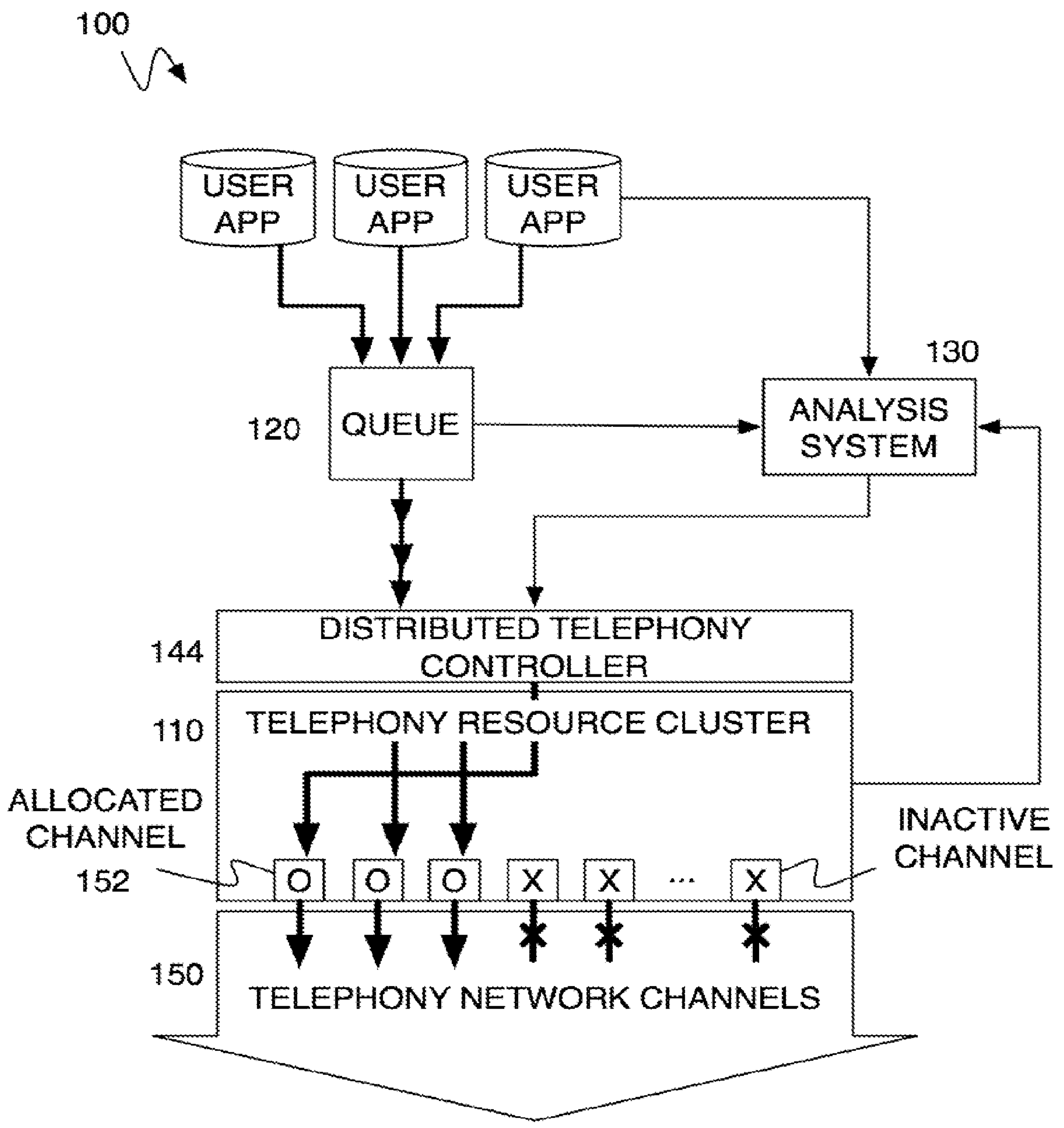
100
USER APP
USER APP
USER APP
130
120
QUEUE
ANALYSIS SYSTEM
144
DISTRIBUTED TELEPHONY CONTROLLER
110
TELEPHONY RESOURCE CLUSTER
ALLOCATED CHANNEL
152
INACTIVE CHANNEL
O
O
O
X
X
...
X
150
TELEPHONY NETWORK CHANNELS

FIGURE 3

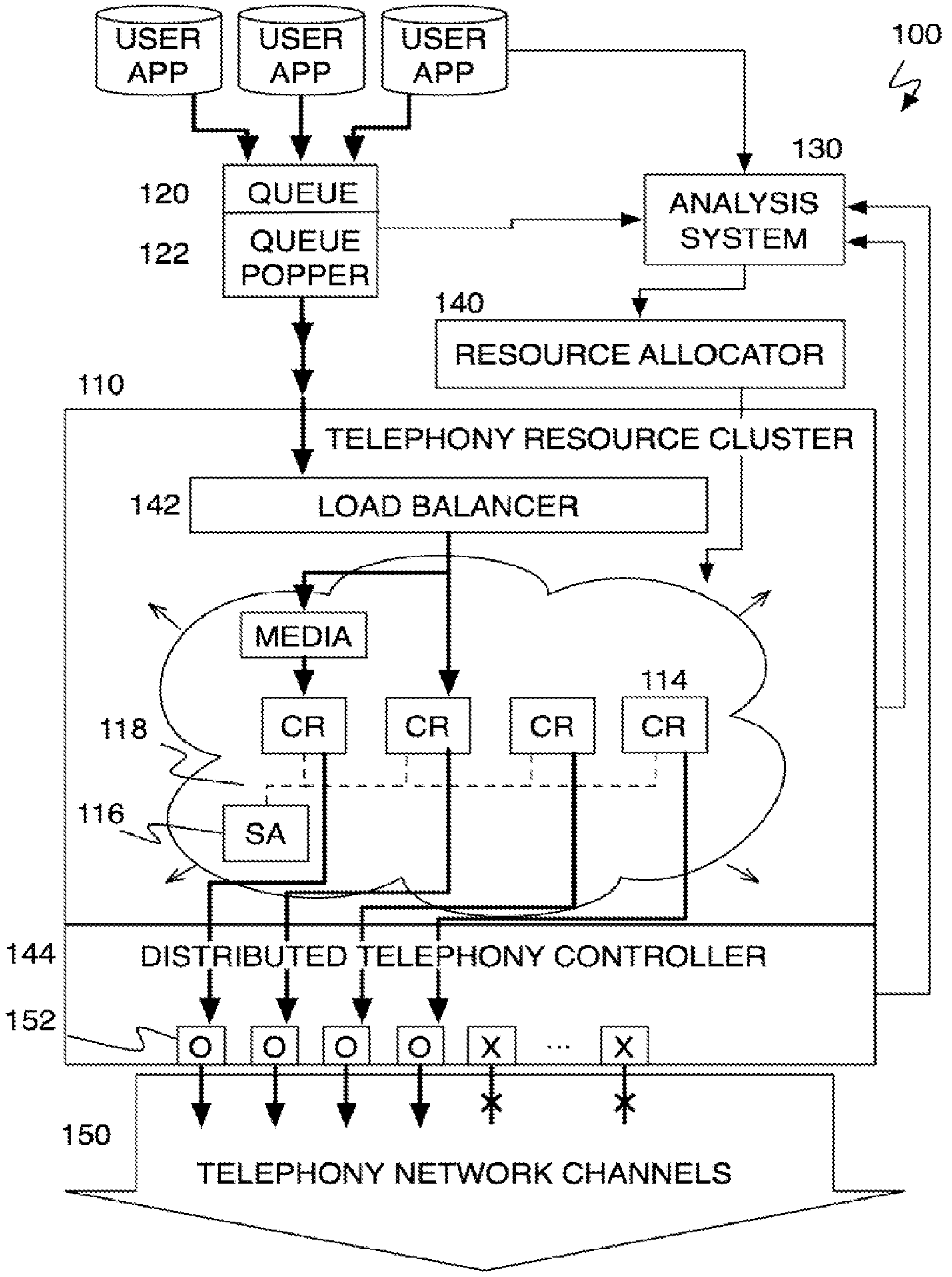
USER APP
USER APP
USER APP
100
130
120
QUEUE
122
QUEUE POPPER
ANALYSIS SYSTEM
140
RESOURCE ALLOCATOR
110
TELEPHONY RESOURCE CLUSTER
142
LOAD BALANCER
MEDIA
114
CR
CR
CR
CR
118
116
SA
144
DISTRIBUTED TELEPHONY CONTROLLER
152
O
O
O
O
X
...
X
150
TELEPHONY NETWORK CHANNELS

FIGURE 4

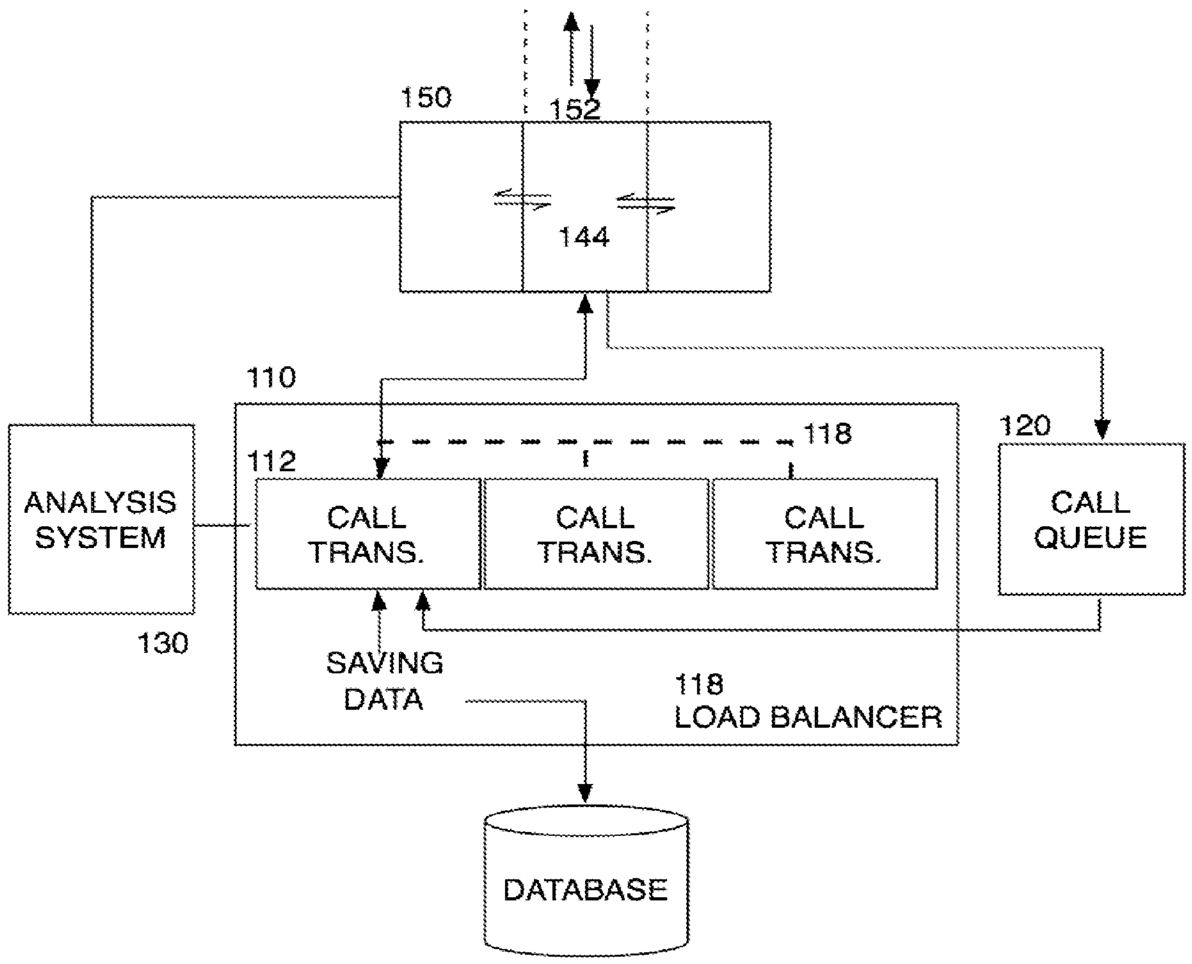
150
152
144
110
118
112
120
ANALYSIS SYSTEM
CALL TRANS.
CALL TRANS.
CALL TRANS.
CALL QUEUE
130
SAVING DATA
118
LOAD BALANCER
DATABASE

FIGURE 5

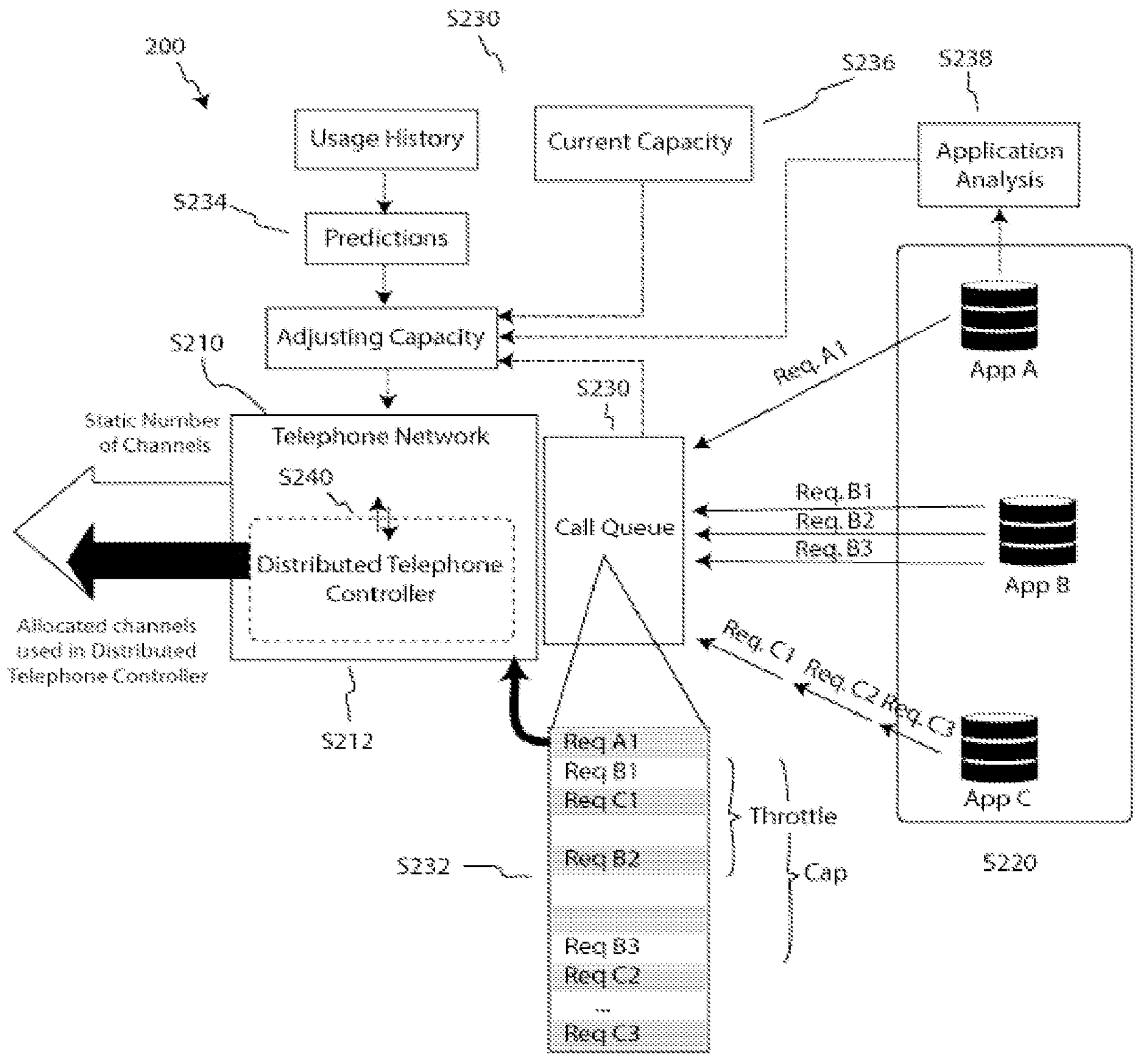
200
S230
Usage History
Current Capacity
S236
S238
Application Analysis
S234
Predictions
Adjusting Capacity
S210
S230
Req. A1
App A
Static Number of Channels
Telephone Network
S240
Distributed Telephone Controller
Call Queue
Req. B1
Req. B2
Req. B3
App B
Allocated channels used in Distributed Telephone Controller
Req. C1
Req. C2
Req. C3
App C
S212
Req A1
Req B1
Req C1
Req B2
Req B3
Req C2
...
Req C3
Throttle
Cap
S232
S220

FIGURE 6

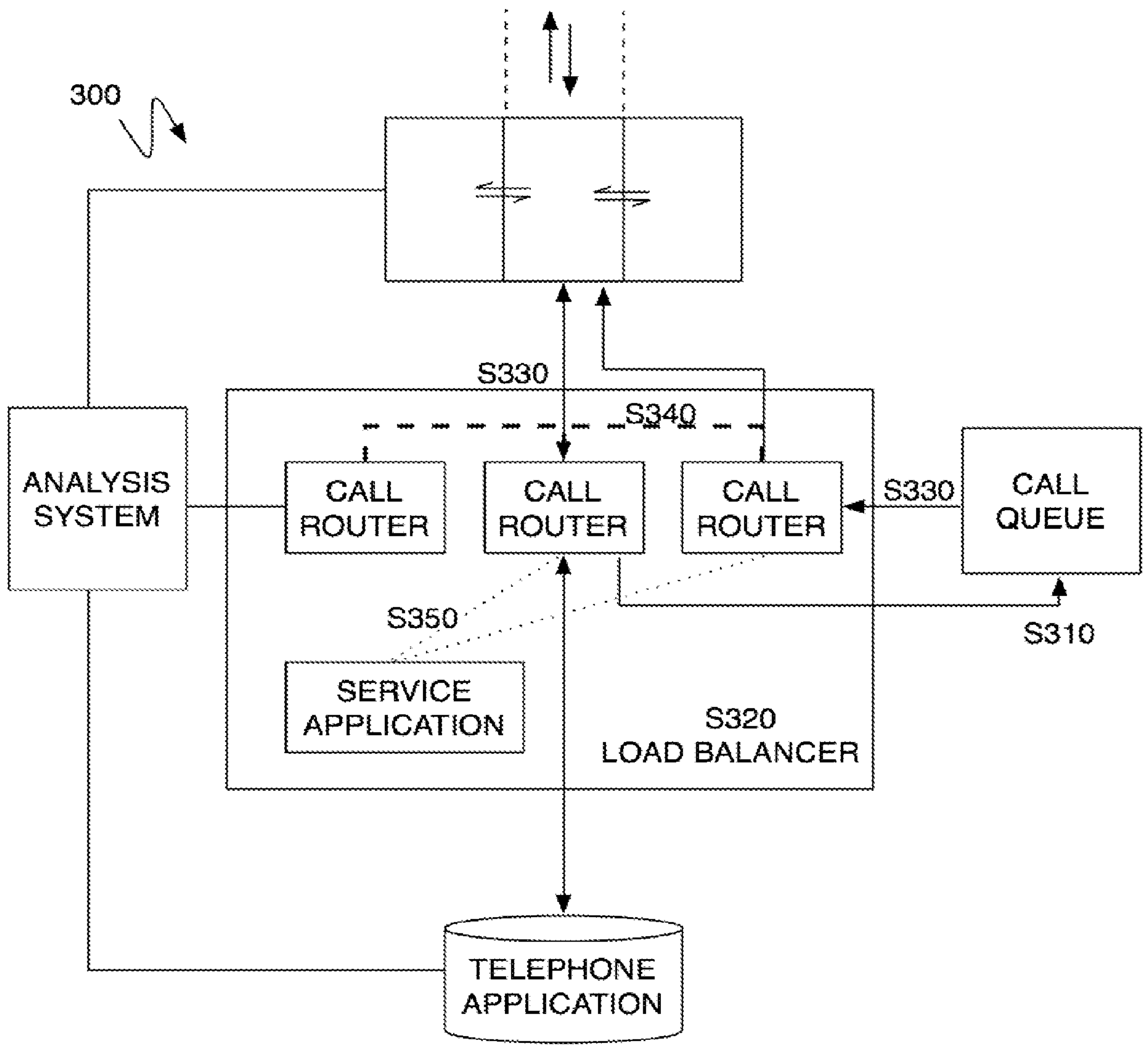
300
S330
S340
ANALYSIS SYSTEM
CALL ROUTER
CALL ROUTER
CALL ROUTER
S330
CALL QUEUE
S350
S310
SERVICE APPLICATION
S320 LOAD BALANCER
TELEPHONE APPLICATION

FIGURE 7

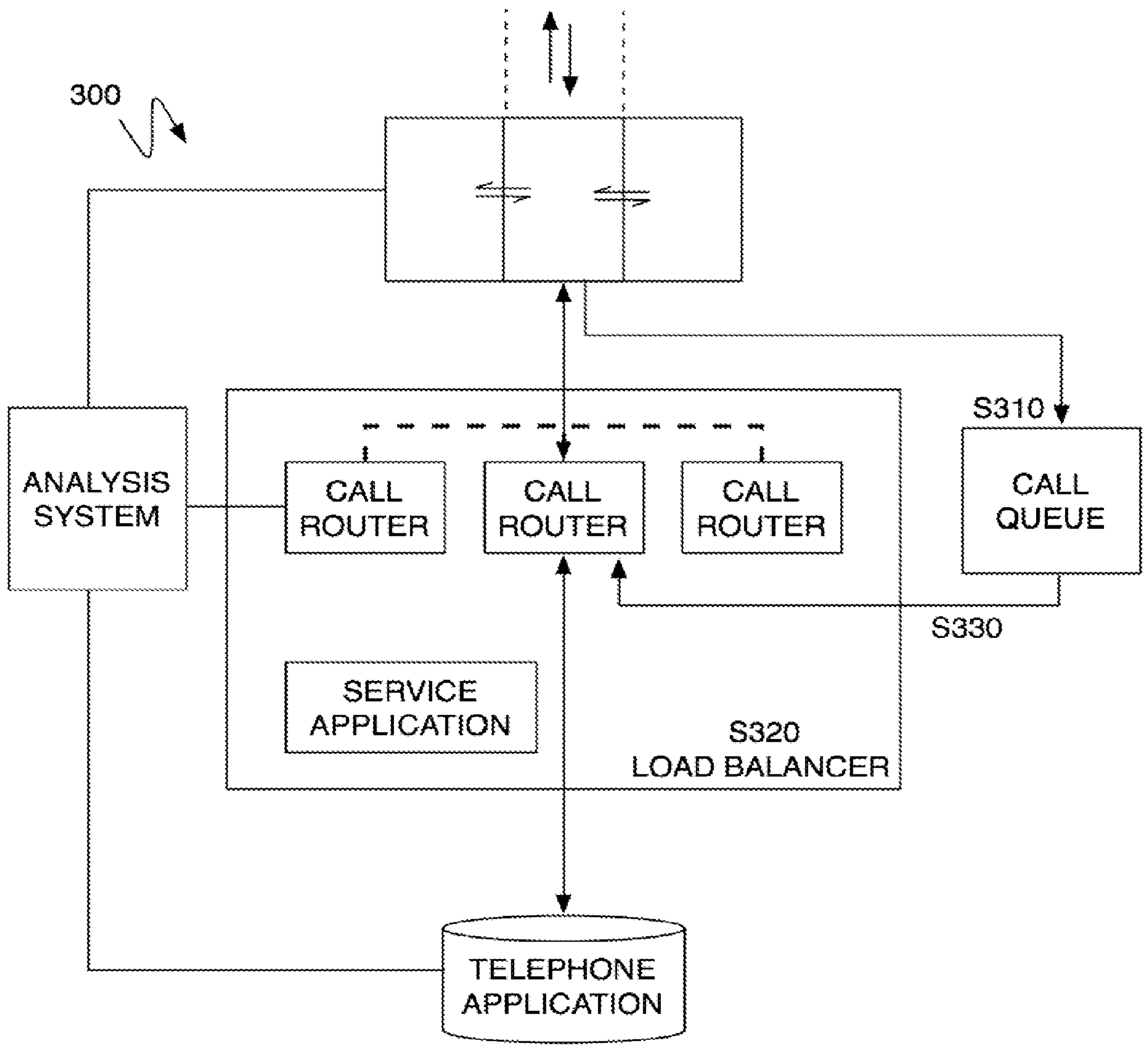
300
ANALYSIS SYSTEM
CALL ROUTER
CALL ROUTER
CALL ROUTER
S310
CALL QUEUE
S330
SERVICE APPLICATION
S320
LOAD BALANCER
TELEPHONE APPLICATION

FIGURE 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2010/025943

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - H04L 12/66 (2010.01)
USPC - 370/352
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(8) - H04L 12/66, 12/28, 12/56 (2010.01)
USPC - 370/352, 412, 537

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 practicable, search terms used)
PatBase

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X -- Y	US 2007/0070980 A1 (PHELPS et al) 29 March 2007 (29.03.2007) entire document	14 -------- 1-11, 13
Y	US 7,227,849 B1 (RASANEN) 05 June 2007 (05.06.2007) entire document	12 -------- 1-11, 13

☐ Further documents are listed in the continuation of Box C. ☐

* Special categories of cited documents:	
"A" document defining the general state of the art which is not considered to be of particular relevance	"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
"E" earlier application or patent but published on or after the international filing date	"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
"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)	"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
"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	"&" document member of the same patent family

Date of the actual completion of the international search 16 April 2010	Date of mailing of the international search report 03 MAY 2010
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Authorized officer: Blaine R. Copenheaver PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774

Form PCT/ISA/210 (second sheet) (July 2009)

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
28 July 2011 (28.07.2011)

PCT

(10) International Publication Number
WO 2011/091085 A1

(51) International Patent Classification:
H04L 12/42 (2006.01)

(21) International Application Number:
PCT/US2011/021774

(22) International Filing Date:
19 January 2011 (19.01.2011)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/296,301 19 January 2010 (19.01.2010) US
13/009,831 19 January 2011 (19.01.2011) US

(71) Applicant *(for all designated States except US)*: **TWILIO INC.** [US/US]; 501 Folsom St. Third Floor, San Francisco, CA 94105 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **LAWSON, Jeffrey** [US/US]; Twilio Inc., 501 Folsom St. Third Floor, San Francisco, CA 94105 (US). **WOLTHUIS, John** [US/US]; Twilio Inc., 501 Folsom St. Third Floor, San Francisco, CA 94105 (US). **COOKE, Evan** [US/US]; Twilio Inc., 501 Folsom St. Third Floor, San Francisco, CA 94105 (US).

(74) Agent: JEFFREY, Schox; 500 3rd Street #515, San Francosco, CA 94107 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: METHOD AND SYSTEM FOR PRESERVING TELEPHONY SESSION STATE

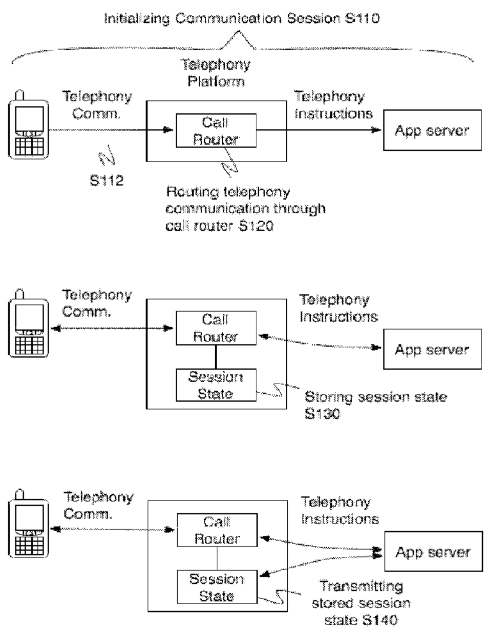

FIGURE.1

(57) Abstract: A method and system for preserving session state in telephony communication including initializing a communication session of telephony communication between a telephony device and an application server; routing the telephony communication through a call router; storing session state for the communication session of the telephony device and the application server; and transmitting the stored session state in communication between the application server and the call router.

WO 2011/091085 A1

Published:

— *with international search report (Art. 21(3))*

METHOD AND SYSTEM FOR PRESERVING TELEPHONY SESSION STATE

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Application Number 13/009,831, filed 19 January 2011, and entitled “METHOD AND SYSTEM FOR PRESERVING TELEPHONY SESSION STATE” and U.S. Provisional Application Number 61/296,301, filed 19 January 2010 and entitled “METHOD AND SYSTEM FOR PRESERVING TELEPHONY MESSAGE STATE”, which are incorporated in their entirety by this reference.

TECHNICAL FIELD

[0002] This invention relates generally to the telephony application field, and more specifically to a new and useful method and system for preserving telephony state in the telephony application field.

BACKGROUND

[0003] Innovations in the web application and Voice over Internet Protocol (VOIP) have brought about considerable changes to the capabilities offered through traditional phone services. New services and platforms have been introduced that integrate telephone voice conversations with website interaction. At the same time the use of SMS (Short Message Service) or MMS (Multimedia Messaging Service) messages, more generically known as text messaging or multimedia messaging, have also become leading forms of communication around the world. However, SMS messages have been limited in the amount of integration with internet applications due to the single message nature of the messaging system. SMS messages have a restricted character limit and correspondingly a limited amount of data that can be associated with a single message. The source of the problem, as discovered by the inventors, is there is no method or system for preserving the telephony message state, so that a single message can be associated with other messages. Additionally, information and application state built up during interactions over phone are not preserved when changing communication channels. Thus, there is a need in the telephony application field to create a new and useful method and system for preserving telephony session state. This invention provides such a new and useful method and system

BRIEF DESCRIPTION OF THE FIGURES

[0004] FIGURES 1 and 2 are schematic representations of methods of a first preferred embodiment;

[0005] FIGURE 3 is a schematic representation of a HTTP cookie variation of a preferred embodiment;

[0006] FIGURE 4 is a schematic representation of a API resource variation of a preferred embodiment;

[0007] FIGURE 5 is a schematic representation of regulating telephone messages of a preferred embodiment;

[0008] FIGURE 6 is a schematic representation of associating a communication session with a voice session of a preferred embodiment;

[0009] FIGURES 7 and 8 are schematic representations of methods of a second preferred embodiment;

[0010] FIGURE 9 is a schematic representation of a variation of the first and second preferred embodiments; and

[0011] FIGURE 10 is a schematic representation of a system of a preferred embodiment.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0012] The following description of the preferred embodiments of the invention is not intended to limit the invention to these preferred embodiments, but rather to enable any person skilled in the art to make and use this invention.

[0013] As shown in FIGURES 1 and 2, a method for preserving telephony state of a preferred embodiment includes initializing a communication session of telephony communication between a telephony device and an application server S110; routing the telephony communication through a call router of a telephony platform S120; storing session state for the communication session of the telephony device and the application server S130; and transmitting the stored session state in communication between the application server and the call router S140. The method functions to preserve the state of a communication session between different instances of telephony communication. The method preferably makes state information of past communication available to application servers and/or provides a mechanism for websites to store data about a particular "telephony visitor" (e.g., the device involved in the messaging or past communication information). In one preferred embodiment, the method employs HTTP cookies with telephony communication for preserving application state. The method is preferably implemented on a system capable of handling voice telephone based applications such as the telephony platform described in published U.S. Publication Number 2009/0252159, filed 02 April 2009 and entitled "SYSTEM AND METHOD FOR PROCESSING TELEPHONY SESSIONS", which is incorporated in its entirety by this reference. The method may alternatively be implemented by a system specifically for telephony messaging or be implemented through an application server communicating with a telephony platform or any suitable system. This method may be used with telephony messaging, which includes Short Message Service (SMS) messages, Multimedia Messaging Service (MMS), fax, or any suitable telephony messaging. A communication session preferably involves a plurality of telephony messages in which application state may be preserved. The method may alternatively or additionally be used with voice sessions (i.e, phone calls), video calls, or any suitable sustained communication protocol. The method may be used with a voice session to preserve state during a phone call and/or between multiple calls between the same caller and callee. While in this document telephony messaging and more specifically Short Message Service (SMS) is commonly used as the preferred telephony protocol, any suitable alternative form of telephony communication using any suitable protocol may be used in addition to or instead of telephony messaging or SMS, such as MMS (Multimedia Messaging Service), fax, or voice.

[0014] Step S110, which includes initializing a communication session of telephony communication between a telephony device and an application server, functions to send a first telephony message or call request to start a conversation over a telephony network. A communication session is preferably a period of communication between at least one device and an application server. A period of communication preferably includes an initial message or communication and preferably subsequent messages. During the communication session, requests are preferably passed between the application server and the call router which translate to telephony communication between the application server and the telephony device. For example, a communication session for SMS messaging occurring between a device and an application server may include each application server to call router request and SMS message transferred for facilitating the communication between an application server and a device until the session is ended. The communication session may include all communication that occurs between the device and application, but may alternatively be defined to be communication within a time period or any suitable rule for beginning and ending a communication session. The telephony communication is preferably telephony messaging and more preferably an SMS message but may be a MMS message voice or any suitable telephony communication. For voice the communication session may be within a call but more preferably includes a plurality of calls between the same caller and callee. A SMS gateway server preferably connects to a SMS network through a Short

Message Service Center ("SMS-C"), directly to the Signaling System #7 (SS7) telephony network, or by any other suitable SMS gateway provider, and the message is preferably received from the gateway by the call router. The call router preferably coordinates the exchange between a telephony device and a networked application server. The telephony device or the application server may initiate the call as exemplified in Steps S112 and S114 below.

[0015] As a first alternative, shown in FIGURE 1, initializing a conversation preferably includes receiving an incoming SMS message from telephony device S112. The telephony device may be a PSTN-connected (Public Switched Telephone Network) or Internet addressable devices, such as landline phones, cellular phones, satellite phones, Voice-Over-Internet-Protocol (VOIP) phones, SIP (session Initiation Protocol) devices, Skype, Gtalk, or any other suitable PSTN-connected or Internet addressable voice device. The incoming SMS message is preferably received from an originating address. The originating address is preferably a standard phone number, but may be any suitable originating address such as a VOIP provider ID, SMS device number, email address, or a short code. Additional information, such as location, may additionally be captured such as from the area code of the phone number. The SMS message is preferably addressed to a destination address (or an incoming address of an application), which is preferably a standard phone number but may alternatively be a toll free number, a short code number, a long code number, a phone number plus an inputted extension number, a phone number plus a tag included in the message, or any suitable destination address. The contents of the message may additionally impact the associated URI, and keyword or tag in addition to the destination number may impact what application server handles the message. After receiving an incoming SMS message from a telephony device, the call router preferably identifies a URI (Universal Resource Identifier) associated (or "mapped") with the destination address. The initial URI is preferably pre-specified at the call router by a web application (which may be running on a third party server) or call router account owner. More preferably, the initial URI is assigned to the incoming SMS message via a unique identifier for the call destination, such as a DID (Direct Inbound Dial) phone number, or a VOIP SIP address. In one variation the SMS system operates within a larger telephony application system, and multiple applications may be associated with a single incoming address. In this variation the URI is identified based on the destination address and the type of incoming message (SMS, MMS, voice call, fax etc.). In other words, a different URI may be identified depending on the source of the telephony session. For example, a voice call or a fax might have different initial URI's that would handle a telephony session initiated by these alternative devices.

[0016] As a second alternative, shown in FIGURE 2, initializing a conversation preferably includes receiving a message request from an application server S114. The SMS message request preferably includes the message context, the text if the request is an SMS message, the media if the message is a MMS, or other content for other forms of telephony communication. The message request may additionally include a response URI that includes the URI to direct messages received in reply to the message initiated by the application server, alternatively the initial URI assigned to the application server may be used. As a first variation, the instructions from the application server may be in the form of a telephony instruction or a command included in the HTTP communication between the call router and the application server. A call router preferably processes telephony instructions to convert a server response into telephony actions or executable operations during a telephony session. A server response is preferably received over HTTP and is preferably formatted as XML. The call router may additionally process the telephony instructions according to the mime-types associated with the server response. For example, if the response mime-type is XML, it is considered to be a set of call router instructions. If the response mime-type is MP3, JPEG, video file, or other media file, it is considered to be media that should be sent as a MMS. If the response type is plain text, it is considered to be text that should be sent as a SMS.

[0017] As a variation of Step S114, the SMS message request from the application server may alternatively be issued through a Call Router API. The Call Router API is preferably an application programming interface (API) such as a REST API (Representational State Transfer) as is known in the art, but the Call Router API may alternatively be a SOAP (Simple Object Access Protocol) API or any suitable programmatic communication interface. A message request is preferably communicated to the call router via the Call router API, and the call router then preferably proceeds to send the message during Step S120. An application server or any suitable HTTP enabled device may use the Call Router API to initiate sending a SMS message to a device that is preferably specified by a phone number, VOIP provider ID, SMS device number, email address, short code, or any suitable telephony device address.

[0018] Step S120, which includes routing the telephony communication through a call router of a telephony platform, functions to forward the telephony communication to a destination. In the variation where an incoming telephony message is received from a device, the telephony message is included in an HTTP or HTTPS message to the application server specified by the identified URI. In the variation where an application server initiates the sending of a telephony message, the application server preferably communicates with the call router, and the call router preferably sends the telephony message to a device specified by a telephony address. As mentioned above, the application server communicating with the call router is preferably achieved through a telephony instruction or the Call Router API, but any suitable framework may be used.

[0019] Step S130, which includes storing session state for the communication session of the telephony device and the application server, functions to provide a resource for preserving state of application interaction for a telephony device and an application. The session state preferably uniquely identifies the communication session created by the telephony device and the application communicating. The session state is preferably defined by a tuple including a "to" and "from" field that include the caller and the callee information of the telephony device and application server. The participants of the telephony communication may alternatively be defined through any suitable construct. Data of a session state is preferably stored in an HTTP Cookie, as shown in FIGURE 3. Alternatively, the session state may be stored using any suitable website session storage mechanism. The HTTP cookie is preferably managed by the telephony platform and may be stored by the call router. A cookie is preferably stored for each communication established through the telephony platform. The to-field and from-field defining the session state is preferably used to identify a corresponding cookie. An application server can preferably utilize the cookie for applications similar to those of browser based interactions. Each device accessing an application server preferably simulates or appears as a browser from the perspective of the application server, and a cookie can preferably be created for each device address accessing an application server. The cookie can be used by the application server to preserve user preferences, a browsing session, or other data for an application. telephony address (e.g., phone number) and the callee telephony address.

[0020] Additionally or alternatively, session state may be stored as a call router API resource that the application server may access through the call router API, as shown in FIGURE 4. The call router API preferably stores state information in a persistent URI for a resource. The API resource may function substantially to the HTTP cookie described above, but may additionally store other information such as communication history. The persistent URI preferably contains all the necessary state information, and this preferably makes data persistent, queryable, and recoverable. The application server or another device may later access the data of the persistent URI to determine the state of the conversation. Specifically, a session history resource is preferably created that is accessible through the call router API. The to-field and from-field is preferably included in the request to identify the data of the session state. A full transcript, individual messages, media files (original and MMS formatted messages), response URI's, meta data such as time stamps, and any suitable data on the conversation may be accessible through the call router API. Session history may include saving the content and actions associated with messages sent between a device and an application server. Often an application server will reply to a telephony message from a device. Session state of these replies are preferably preserved and communicated in substantially the same manner as an initial telephony message (e.g., by a telephony instruction or through a Call Router API). Though the reply is preferably sent to the originating address of the device (as opposed to some specified address of the application server). Additionally, the application server can preferably specify a response URI for future messages from a device. The call router will preferably pass the next message received from the device to the reply URI as opposed to the initial URI discussed above. In the course of executing a telephony message application, there may be a plurality of messages passed between a device and an application server. The storing of state of a conversation additionally functions to support these multi-message applications. A transcript or history of past messages are preferably stored as part of the state of the conversation. By preserving the state of the conversation, SMS messages become a full conversation as opposed to unrelated, single messages.

[0021] Additional abstractions of the history of conversations may additionally be made. As one abstraction, a concept of sessions may be introduced which can be used to group a plurality of related messages. Such sessions may be assigned based on the temporal spacing of the messages. For example, a session may be closed after an expiration time requirement is met and the next message marks the beginning of a new session. An application server may alternatively define the beginning and end of a session. For example, the application server may send a telephony instruction or call router API message to signal the beginning and the conclusion of a session. This may additionally be signaled through the session state.

[0022] Step S140, which includes transmitting the stored session state in communication between the application server and the call router, functions to communicate the session state to the application server. The session state may be used by the application server to set application state or for any suitable application. Preferably, the session state HTTP cookie is sent to the application server for all communication with the application server. As described above the call router preferably communicates with the application server through HTTP or HTTPS. The HTTP cookie is preferably transmitted through these techniques. Alternatively, if the session state is stored as an API resource, then the session state is transmitted in response to an API call. The session state may alternatively be stored on the application server, or communicated to the application server in any suitable manner.

[0023] Additionally, the method may include regulating within a communication session S150 as shown in FIGURE 5, which functions to place restrictions on forms of communication within a communication session. In particular this may include limiting the speed and number of telephony messages that may be sent. When sending telephony messages from the call router, the telephony messages may be queued to manage the load on the telephone network used by the call router. The queue functions to control network resources shared between a plurality of application servers sending telephony messages, and also to individually control the rate of telephony messages from an application server and avoid any network limitations such as SMS or MMS filtering, labeling as spam, or forced throttling. There are preferably a plurality of queues managed by the call router which may be allocated and deallocated according to resource demand. There is preferably a queue for each communication session. The queues may alternatively be shared amongst entities involved in various communication sessions. A dequeuer (or popper) preferably manages the selection of items from the queue to send from the call router. There may additionally be a plurality of dequerers for the plurality of queues, and each queue may have multiple dequerers selecting items from the queue. The dequeuers preferably dynamically control the rate and number of messages selected from the queue. Messages and/or calls may be rate limited for a particular communication session. Rate limiting may include limiting the number of messages/calls in a time period, time period between messages/calls, and/or any suitable rule for limiting communication. When sending a telephony message, the call router may additionally handle formatting of the contents of the telephony message. This substep functions to alleviate application servers from correctly formatting all messages before communicating with the call router. As one example, a message may have more than the allowed number of characters (e.g., 160 characters) in a SMS message. The call router preferably splits the message into multiple SMS message requests to satisfy the character limit. Each SMS message request is then preferably individually queued. The message may additionally be split based on the contents of the text to prevent splitting a word, phrase, or sentence between different messages. In other words, the message is preferably split to preserve the semantic meaning of the message by analyzing the grammar and written structure of the message. Additionally, content may be added to the message such as the name of the source of the text message or the page number (e.g., "1 of 3:") in the case of a split message. In the case of MMS messages, the media may be compressed, resized, converted to an appropriate format, or replaced with a web link to the media file (if the media is not compatible with MMS). Related to the split messages, the call router may combine split messages sent from a device into a single message. This is preferably implemented by delaying the passing of a message to an application server, and combining additional messages into a single message before passing the message to an application server. Any suitable pre-processing of application server messages and post-processing of device messages may alternatively be used.

[0024] Additionally, the method may include associating the communication session with a voice session S160 as shown in FIGURE 6, which functions to allow the session state of a communication session to be used by a voice session. For example, while performing SMS messaging with an application, a voice session may be initiated from the same device to the application, and the session state of the previous SMS communication session may be used within the voice session. Similarly, the step may include merging voice call session state with a SMS conversation. This step may be applied to enable merging session state of numerous forms of communication. This preferably involves sharing session state between communication sessions with different mediums of conversation (e.g., voice and SMS). This is particularly applicable on telephone networks capable of simultaneous voice and data transmission. Some applications may call for SMS messages or MMS messages to be sent during a telephone conversation with an application server. The session state is preferably associated with a second communication session by using the same HTTP cookie for communication between the call router and the application server or alternatively copying the HTTP cookie. As another alternative, the API resources storing the session state may be accessed for use with either communication session. The telephone conversation (i.e., voice session) can preferably be a standard two party call (between two telephony devices with the call router in between or between a telephony device and an application server) or a multi party conference call. In this alternative resources generated during a voice session such as recordings, transcriptions, DTMF (Dual-Tone Multi-Frequency) signal inputs, or any suitable resources created during a voice session are preferably associated with the session history resource. Preferably the resources of the voice session are included as part of the conversation resource but they may alternatively be referenced or a shared identification code may be used.

[0025] As shown in FIGURES 7 and 8, a second preferred embodiment the method may include the steps of initializing a communication session with a telephony message between a telephony device and an application server S210, and additionally assigning a unique tracking link to the communication session S270, communicating the tracking link S280, and associating resource access made through the tracking link with the communication session S290. Method functions to preserve session state between modes of communication. This is particularly useful for associating browser sessions with telephony sessions that involve the same parties. For example, data entered during a phone call can be imported into a web application accessed through a browser. Step S120 is preferably substantially similar to Step S110 described above. Additionally the Steps S120, S130, S140, S150, and/or S160 may all be used in combination with the additional steps of S270, S280, and/or S290 as shown in FIGURE 9. The tracking link preferably functions as a way for triggering an association with a communication session when accessing a resource. This is preferably used for associating telephony sessions with browser sessions, but may be used to associate a telephony session with any suitable session such as an application session.

[0026] Step S270, which includes assigning a unique tracking link to the communication session, functions to create a sharable link to a resource that can be used to identify the entity accessing the resource. The tracking link is preferably a URI that includes parameters that associate a communication session with the URI. The URI may alternatively route the user initially through an initial site for tracking, similar to a link shortening service. The tracking link preferably leads to a webpage, but may alternatively direct to other resources such as an application. The tracking link may alternatively open up an application on a device. The resource opened by the tracking link is preferably operated by the application server entity but may be operated by any suitable party, preferably one with access to information of the communication session. Thus resource access and communication session information are both available to the application server entity or outside entity.

[0027] Step S280, which includes communicating the tracking link, functions to deliver the tracking link for user access. The tracking link may be sent in a SMS message as shown in FIGURE 7, an email message as shown in FIGURE 8, fax, over audio, in an image or video, or communicated through any suitable means. The communication of the tracking link is preferably initiated by an application server during or following a telephony communication session.

[0028] Step S290, which includes associating resource access made through the tracking link with the communication session, functions to associate a communication session with the access of a resource. When the tracking link is used to access the resource, the communication session associated with that particular tracking link is preferably identified. Preferably, session state or other resources associated with the communication session can then be used within this newly accessed resource (e.g., webpage or application). For example, after being sent a link in an SMS message, a user may click the link and a browser of the phone preferably opens the link to a resource hosted by the application server. Since the link was uniquely assigned for that communication session, the application server knows that the user opening this browser session must have some connection to the user participating in SMS conversation. Information gathered during the SMS conversation can be used to impact the resource accessed by the user. The device opening the tracking link need not be the same device where the communication session occurred. Since the tracking link is unique and preferably shared with the user(s) of the communication session, an application server can assume that the user(s) is the same.

[0029] As shown in FIGURE 10, a system 300 for preserving telephony message state of the preferred embodiment preferably includes a call router 310 and a session state manager 320. The system functions to store and make session state accessible for telephony messages passed between a device and an application server. The call router 310 preferably includes a message router for sending and receiving SMS/MMS messages. The call router 310 can preferably connect to an SMS network SMS through a Short Message Service Center ("SMS-C"), directly to the Signaling System #7 (SS7) telephony network, or by any other suitable SMS gateway provider. The message router can preferably send and receive messages from SMS network devices, cellular phones, computers, smartphones, or any suitable SMS network devices. The call router 310 may additionally or alternatively send or receive text messages or multimedia messages different protocols, emails, faxes, make voice calls over PSTN (Public Switched Telephone Network) network, and other suitable PSTN-compatible communication messages. The communication between the application server and the call router 310 is preferably stateless and any state information (e.g., call state) or data is preferably located in a URI or the request parameters, such as HTTP headers, GET URI parameters, POST request body parameters, or HTTP cookies. The session state manager 320 preferably functions to store and communicate stored session state. Preferably, this includes infrastructure to store and transmit HTTP cookies used to store session state for communication sessions involving a device and telephony application. Session state is stored within an HTTP cookie unique for a communication session, and the HTTP cookie can be transmitted to application servers for application logic. The session state manager may alternatively or additionally include a call router API 322 and a session history resource 324. The session history resource 324 is preferably stored data related to the conversation created between a device and an application server (either a single message or a plurality of messages). The conversation resource 324 is preferably a resource of the call router API 322, but may alternatively be a cookie or any suitable device to store state information. The system may additionally include a plurality of queues, a plurality of dequeuers (i.e., queue poppers), which functions to balance resource usage of the messaging network. The queues and dequeuers can preferably be allocated and deallocated from the system to account for capacity requirements. The system may encourage a plurality of messages to be sent which may cause increased load on the network. The queue and the dynamic allocation of resources preferably provide a device to compensate for a large volume of messages. The queues and/or dequeuers preferably function to control the throttling (i.e., service rates) of message requests. The throttling may be performed on a per-phone number, per-account (as in a multi-tenant application), and/or according to any message attribute.

[0030] An alternative embodiment preferably implements the above methods in a computer-readable medium storing computer-readable instructions. The instructions are preferably executed by computer-executable components preferably integrated with a telephony platform and/or application server. The computer-readable medium may be stored on any suitable computer readable media such as RAMs, ROMs, flash memory, EEPROMs, optical devices (CD or DVD), hard drives, floppy drives, or any suitable device. The computer-executable component is preferably a processor but the instructions may alternatively or additionally be executed by any suitable dedicated hardware device.

[0031] As a person skilled in the art will recognize from the previous detailed description and from the figures and claims, modifications and changes can be made to the preferred embodiments of the invention without departing from the scope of this invention defined in the following claims.

CLAIMS

We Claim:

1. A method for preserving session state in telephony messaging comprising:
   - initializing a communication session with a telephony communication between a telephony device and an application server;
   - routing the telephony communication through a call router;
   - storing session state for the communication session of the telephony device and the application server; and
   - transmitting the stored session state in communication between the application server and the call router.
2. The method of Claim 1, wherein a telephony communication is an SMS message.
3. The method of Claim 1, wherein a telephony communication is a voice call.
4. The method of Claim 1, wherein the session state is defined by a to-field and from-field.
5. The method of Claim 4, wherein transmitting the stored session state includes transferring session state data through an HTTP Cookie identified from the to-field and from-field of the session state.
6. The method of Claim 1, wherein data of the session state is stored as an API resource accessible through an API of the telephony platform; and wherein transmitting the stored session state data includes transferring session state data through an API call identifying a session state by a to-field and from-field.

7. The method of Claim 1, further comprising regulating communication within a communication session.

8. The method of Claim 7 wherein regulating includes queuing telephony communication and rate limiting telephony communication within a communication session determined by the session state.

9. The method Claim 1, wherein the telephony communication is telephony messaging; and further comprising associating the communication session with a voice session.

10. The method of Claim 9, wherein the session state is stored as an HTTP cookie, and wherein associating the communication session with a voice session further includes sharing the HTTP cookie with the voice session between the telephony device and the application server.

11. The method of Claim 1, further comprising communicating a tracking link, wherein a tracking link is a link to a resource and the tracking link is unique to the communication session; and associating access of a resource through the tracking link with the communication session.

12. The method of Claim 11, wherein the tracking link is a universal resource identifier (URI) and resource access is made through a web browser; and wherein associating resource access includes associating the browser session with the communication session.

13. The method of Claim 11, wherein a telephony communication is an SMS message, and wherein the tracking link is communicated through an SMS message.

14. The method of Claim 11, wherein the tracking link is communicated in an email message.

15. A method for preserving session state between plurality of communication channels comprising:

    - initializing a communication session between a telephony device and an application server;
    - assigning a unique tracking link to the communication session
    - communicating the tracking link; and
    - associating resource access made through the tracking link with the communication session.

16. The method of Claim 15, wherein the communication session is a telephony voice session.

17. The method of Claim 15, wherein the tracking link is a URI and resource access is made through a web browser; and wherein associating resource access includes associating the browser session with the communication session

18. The method of Claim 15, wherein the resource access includes opening an application indicated through the tracking link.

19. The method of Claim 15, wherein the tracking link is communicated through an SMS message.

20. The method of Claim 15, wherein the tracking link is communicated through an email message.

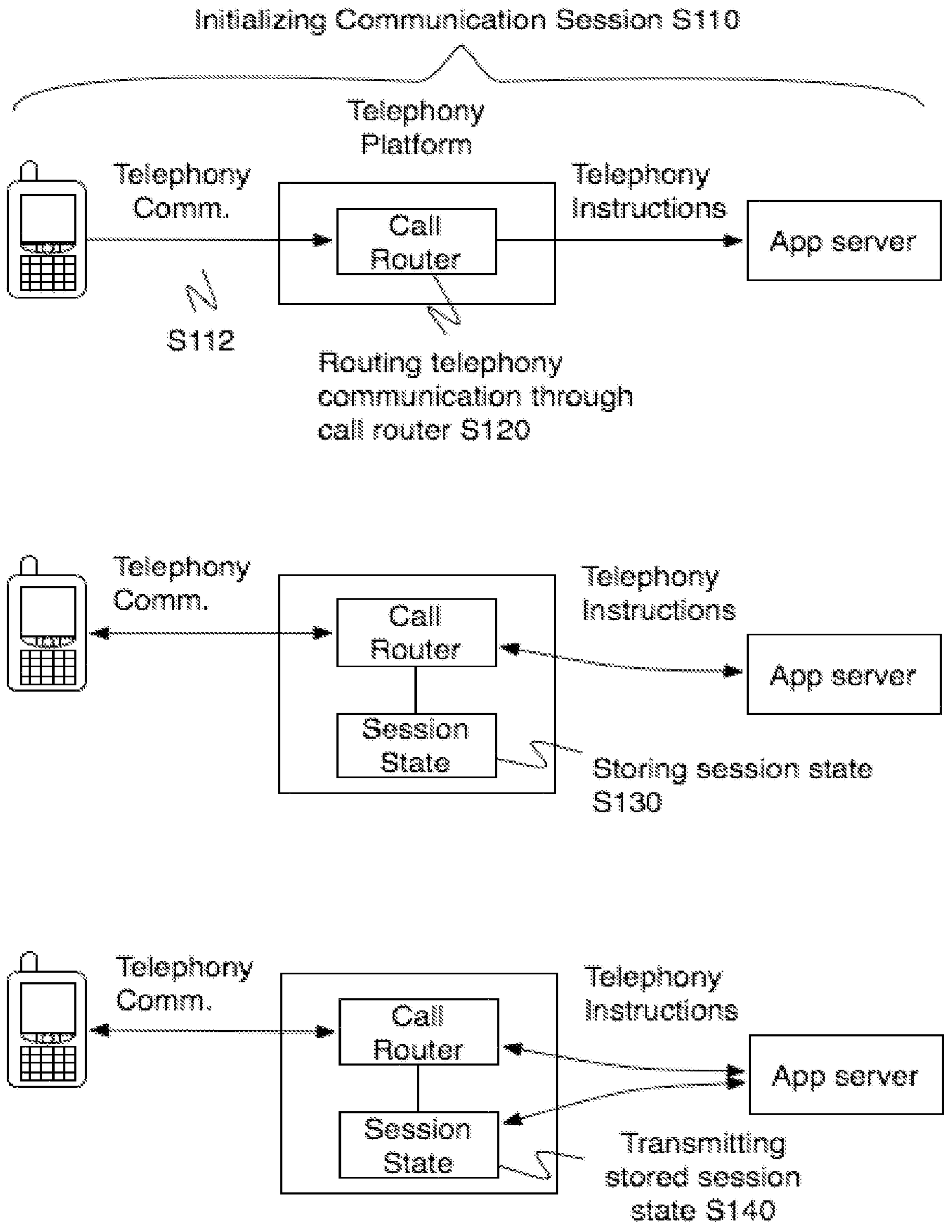
Initializing Communication Session S110
Telephony Platform
Telephony Comm.
Call Router
Telephony Instructions
App server
S112
Routing telephony communication through call router S120
Telephony Comm.
Call Router
Telephony Instructions
App server
Session State
Storing session state S130
Telephony Comm.
Call Router
Telephony Instructions
App server
Session State
Transmitting stored session state S140

FIGURE 1