A speech processing system can include a client, a speech for Web 2.0 system, and a speech processing system. The client can access a speech-enabled application using at least one Web 2.0 communication protocol. For example, a standard browser of the client can use a standard protocol to communicate with the speech-enabled application executing on the speech for Web 2.0 system. The speech for Web 2.0 system can access a data store within which user-specific speech parameters are included, wherein the user of the client is able to configure the specific speech parameters of the data store. Suitable ones of these speech parameters are utilized whenever the user interacts with the Web 2.0 system. The speech processing system can include one or more speech processing engines. The speech processing system can interact with the speech for Web 2.0 system to handle speech processing tasks associated with the speech-enabled application.
FIG. 1
FIG. 2
FIG. 3

Speech for Web 2.0 System 310

REST/ATOM 302

WIKI Server 330

APP Format

Transformer 340

AJAX

Editor 350

WIKI data

Speech-enabled Application 334

WIKI App 360

Introspection Doc

WIKI App 360

Resource Collection 364

ASR Resource

TTS Resource

Collection 366

Collection 367

Collection 368

Collection 369

Entry Collection 362

WIKI pg 1

WIKI pg n

WIKI App 360

* Entries Link

* Resources Link

* Entries Link

* Resources Link

* Entries Link

* Resources Link

* Entries Link

* Resources Link

* Entries Link

* Resources Link
SPEECH PROCESSING SYSTEM BASED UPON A REPRESENTATIONAL STATE TRANSFER (REST) ARCHITECTURE THAT USES WEB 2.0 CONCEPTS FOR SPEECH RESOURCE INTERFACES

BACKGROUND

[0001] 1. Field of the Invention

[0002] The present invention relates to the field of speech processing technologies and, more particularly, to a speech processing system based upon Representational State Transfer (REST) architecture that uses Web 2.0 concepts for speech resource interfaces.

[0003] 2. Description of the Related Art

[0004] In the past, companies having a Web presence thrived by providing as many people broad access to as much information as possible. Information flow was unidirectional, from a company to information consumers. As time has progressed, users have become inundated with too much information from too many sources. Successful Web sites began to provide user-facing information management and information filtration mechanisms designed to aid users in identifying information of interest. Even these Web sites were somewhat flawed in a sense that information still flowed in a unidirectional manner. A user was limited to information gathered and groomed by a particular information provider.

[0005] A new type of Web application began to emerge which emphasized user interactions and two-way information exchange. These new Web applications operated more as information marketplaces where people shared information and not as information depositories where users accessed a semi-static reservoir of information. This new Web and set of Web applications can be referred to as Web 2.0, where Web 2.0 signifies a second generation of Web based services and applications that emphasize online collaboration and information sharing among users. In other words, a Web 1.0 application would be one that was effectively read-only from a user perspective, whereas a Web 2.0 application would provide read, write, and update access to end-users. Web 2.0 users can fundamentally change a Web 2.0 application.

[0006] Specific examples of Web 2.0 instances include WIKIs, BLOGs, social networking sites, FOLKSONOMIES, MASHUPS, and the like. All of these Web 2.0 instances allow end-users to add content which other users are able to access. A value of a Web 2.0 Web site is enhanced by the user provided content and may even be completely dependent upon it.

[0007] For example, WIKIPEDIA (e.g., one Web 2.0 application) is a WIKI based encyclopedia where each end-user is able to view, add, and edit content. No content would exist without end-user contributions. Information accuracy results from an end-user population constantly updating erroneous entries which other users provide. As new innovations emerge, customers update and add WIKIPEDIA entries that describe these new innovations. Other examples of Web 2.0 applications include MYSPACE.com, YOUTUBE.com, DELICIO.US.com, CRAIGSLIST.com, and the like.

[0008] Currently, a schism exists between speech processing technologies and Web 2.0 applications, meaning that Web 2.0 instances do not generally incorporate speech processing technologies. One reason for this is that conventional interfaces to speech resources are too complex for an average end-user to utilize. For this reason, speech technologies are typically only available from Web sites/services that provide a unidirectional flow of information. For example, speech technologies are commonly used by enterprises to handle routine customer interactions via a telephone interface, such as providing bank balances and the like.

[0009] One problem contributing to the schism is that speech processing technologies are currently implemented using a non-uniform interface and the Web 2.0 is generally based upon a uniform interface. That is, speech processing operations are accessed via function calls, method invocations, remote procedure calls (RPC), and other techniques that are only understood by a specific server or a small subset of components. A specific invocation mechanism and required parameters must be known by a client and must be integrated into an interface. A non-uniform interface is characteristic of RPC based techniques, which includes Simple Object Access Protocol (SOAP), Common Object Request Broker Architecture (CORBA), Distributed Component Object Model (DCOM), JINI, and the like. Without deliberate integration efforts, however, the chances that two software objects designed from an unconstrained architecture are near nil. At best, an ad hoc collection of software objects having vastly different interface requirements results from the RPC style architecture. The lack of uniform interfaces makes integrating speech processing capabilities for each RPC based application a unique endeavor fraught with application specific challenges, which usually require significant speech processing design skills to overcome.

[0010] In contrast, a uniform interface exists that includes a few basic primitive commands (e.g., GET, PUT, POST, DELETE) that act upon targets, which in a Web 2.0 context are generally able to be referenced by Uniform Resource Identifiers (URIs). A term used for this type of architecture is Representational State Transfer (REST). REST based solutions simplify component implementation, reduce the complexity of connector semantics, improve the effectiveness of performance tuning, and increase the scalability of pure server components. The Web (e.g., hypertext technologies) in general is founded upon REST principles. Web 2.0 expands these REST principles to permit end users to add (HTTP PUT), update (HTTP POST), and remove (HTTP DELETE) content. Thus, WIKIs, BLOGs, FOLKSONOMIES, MASHUPS, and the like are all considered RESTful, since each generally follows REST principles.

[0011] What is needed to bridge the gap between speech processing resources and conventional Web 2.0 applications is a new paradigm for interfacing with speech processing resources, which makes speech processing resources more available to end-users. In this contemplated paradigm, end-users would optimally be able to cooperatively and dynamically develop speech-enabled solutions, which the end-users would then be able to integrate into Web 2.0 content. Thus, a more robust Web 2.0 environment that incorporates speech processing technologies will be allowed to evolve. This is a stark contrast with a conventional paradigm for interfacing with speech processing resources, which is decisively non-RESTful in nature.

SUMMARY OF THE INVENTION

[0012] The present invention discloses a RESTful speech processing system that uses Web 2.0 concepts for interfacing with server-side speech resources. The RESTful speech processing system can be used to add customizable speech processing capabilities to Web 2.0 instances, such as WIKIs, BLOGs, social networking sites, FOLKSONOMIES,
MASHUPs, and the like. The invention can access speech-enabled applications via introspection documents. Each speech-enabled application can contain a collection of entries and resources. The entries can include Web 2.0 entries, such as WIKI entries and the resources can include speech resources, such as speech recognition, speech synthesis, speech identification, and voice interpreter resources. Each entry and resource can be further decomposed into sub-components specified at a lower granularity level. Each application resource/entry can be introspected, customized, replaced, added, re-ordered, and/or removed by end users.

[0013] The present invention can be implemented in accordance with numerous aspects consistent with the material presented herein. For example, one aspect of the present invention can include a speech processing system that includes a client, a speech for Web 2.0 system, and a speech processing system. The client can access a speech-enabled application using at least one Web 2.0 communication protocol. For example, a standard browser of the client can use a HyperText Transfer Protocol (HTTP) to communicate with the speech-enabled application executing on the speech for Web 2.0 system. The speech for Web 2.0 system can access a data store within which user specific speech parameters are included, wherein a user of the client is able to configure the specific speech parameters of the data store. For example, a user can configure which speech resources are available (e.g., TTS, ASR, STT, VoiceXML, interpreter, and the like), resource characteristics (language, grammar, voice gender, speaking rate, and the like), delivery characteristics (real-time or not, synchronous or not, delivery protocol, delivery codec, delivery fidelity, and the like), and other such characteristics. Suitable ones of these speech parameters are utilized whenever the user interacts with the Web 2.0 system. The speech processing system can include one or more speech processing engines. The speech processing system can interact with the speech for Web 2.0 system to handle speech processing tasks associated with the speech-enabled application.

[0014] Another aspect of the present invention can include a system for using Web 2.0 as an interface to speech engines. The system can include a Web 2.0 server and a server-side speech processing system. The Web 2.0 server can serve at least one speech-enabled application to at least one remotely located client. The server-side speech processing system can handle speech processing operations for the speech-enabled applications. Communications with the server-side speech processing system can occur via a set of RESTful commands, such as GET, PUT, POST, and DELETE.

[0015] Still another aspect of the present invention can include a speech for Web 2.0 system that includes a Web 2.0 server. The Web 2.0 server can serve at least one speech-enabled application to remotely located clients. The speech-enabled application can include an introspection document, a collection of entries, and a collection of resources. At least one of the resources can be a speech resource associated with a speech engine, which adds a speech processing capability to the speech-enabled application.

[0016] It should be noted that various aspects of the invention can be implemented as a program for controlling computing equipment to implement the functions described herein, or a program for enabling computing equipment to perform processes corresponding to the steps disclosed herein. This program may be provided by storing the program in a magnetic disk, an optical disk, a semiconductor memory, or any other recording medium. The program can also be provided as a digitally encoded signal conveyed via a carrier wave. The described program can be a single program or can be implemented as multiple subprograms, each of which interact within a single computing device or internet in a distributed fashion across a network space.

[0017] It should also be noted that the methods detailed herein can also be methods performed at least in part by a service agent and/or a machine manipulated by a service agent in response to a service request.

BRIEF DESCRIPTION OF THE DRAWINGS

[0018] There are shown in the drawings, embodiments which are presently preferred, it being understood, however, that the invention is not limited to the precise arrangements and instrumentalities shown.

[0019] FIG. 1 is a schematic diagram of a system that utilizes Web 2.0 concepts for speech processing operations in accordance with an embodiment of the inventive arrangements disclosed herein.

[0020] FIG. 2 is a schematic diagram of a system for a Web 2.0 voice system in accordance with an embodiment of the inventive arrangements disclosed herein.

[0021] FIG. 3 is a schematic diagram showing a WIKI server adapted for communications with a Web 2.0 for voice system in accordance with an embodiment of the inventive arrangements disclosed herein.

DETAILED DESCRIPTION OF THE INVENTION

[0022] FIG. 1 is a schematic diagram of a system 100 that utilizes Web 2.0 concepts for speech processing operations in accordance with an embodiment of the inventive arrangements disclosed herein. In system 100, a user 110 can use an interface 114 of client 112 to communicate with the speech for Web 2.0 system 120, which can include a Web 2.0 server 122 and/or a RESTful server 130. When the client 112 is a basic computing device (e.g., a telephone), a middleware server 116 can provide an interface 118 to system 120. Interface 114 and/or 118 can be a Web or voice browser, which communicates directly with system 120 using Web 2.0 conventions. Applications 126, which the client 112 accesses, can be voice-enabled applications stored in data store 124. A type of browser (e.g., interface 114 and/or 118) used to access the applications 126 can be transparent to the system 120, or can be transparent at least to RESTful server 130 of system 120.

[0023] The RESTful server 130 can provide speech processing operations for applications 126 by interfacing with speech processing system 150. Communications between the Web 2.0 server 122 and the RESTful server 130 can be REST based communications, such as those conducted using the ATOM PUBLISHING PROTOCOL (APP). In one embodiment, servers 122 and 130 can be functionally integrated into a single server of speech for Web 2.0 system 120.

[0024] The RESTful server 130 can utilize a set of basic commands enabling the command engine 132 to conduct speech processing operations. The commands can be REST commands that include an HTTP GET, an HTTP POST, and HTTP PUT, and an HTTP DELETE command. The RESTful server 130 can also include an introspection/discovery engine 134 and/or a media engine 136 as well as data store 138.

[0025] Data store 138 can include a set of documents 140, such as introspection documents 142, entry collection documents 144, and resource collection documents 146. The
documents 140 together can link the RESTful server 130 to speech processing engines 156 of speech processing server 150 and can control behavior of speech processing server 150. The documents 140 and resulting behavior of the speech processing server 150 can be configured by user 110 in a user-specific manner. That is different users 110 can inject their own voice characteristics, markup, behavior, and/or other features, which the speech processing system 150 utilizes.

The Web 2.0 system 120 can be communicatively linked to one or more enterprise servers 158 having an associated data store 160. Thus, the Web 2.0 system 120 can be a communication intermediary which provides user 110 with access to information and services of the enterprise server and data store 160.

Web 2.0 system 120 can further be communicatively linked to one or more additional RESTful servers 162, each associated with a data store 164, within which a set of documents, approximately equivalent to documents 140, are stored. Communications between Web 2.0 system 120 and speech processing system 150 or RESTful server 162 can be based on a RESTful protocol, such as APP.

It should be appreciated that RESTful servers 130 and 162 are able to operate in a stateless fashion which permits RESTful server 162 to seamlessly replace functionality of server 130. That is, state information does not have to be transferred when control is transferred from one server 130 to another 162. Thus, system 100 provides a highly scalable solution (i.e., when under a heavy load, server 130 can transfer load to server 162) and can provide fault tolerance and recovery capabilities (i.e., when server 130 experiences runtime problems, a different operational server 162 can immediately perform operations previously handled by server 130).

Another point about system 100 that should be emphasized is that client 112 is able to interact with the speech-enabled application 126 using Web 2.0 communication protocols only. No special client-side speech interface is required. At the same time, the user 110 is able to customize/personalize/configure speech processing behavior at low-levels.

As used herein, Web 2.0 is a concept that refers to a cooperative Web in which end-users 110 add value by providing content, as opposed to Web systems that unidirectionally provide information from an information provider to an information consumer. In other words, Web 2.0 refers to a readable, writable, and updateable Web. While a myriad of types of Web 2.0 instances exist, some currently popular ones include WIKIs, BLOGs, MASHUPS, FOLKSONOMIES, social networking sites, and the like.

REST refers to a Representational State Transfer architecture. A REST approach focuses on utilizing a constrained operation set, such as GET, PUT, POST, and DELETE, to act against a set of structured targets which can be URL addressable. A REST architecture is a client/server architecture which is stateless, cachable, and layered by nature. REST replaces a paradigm of do-something with a make-something-so concept. That is, instead of attempting to execute a kind of state transition for a software object, the REST concept changes a state of a software object to a user designated state. A RESTful object (e.g., a RESTful server 130, 162) is one which primarily conforms to REST concepts. A RESTful interface can be a simple interface that transmits domain-specific data using an HTTP-based protocol without utilizing an additional messaging layer, such as SOAP, and without reliance of session tracking HTTP cookies.

The client 112 can be any computing device capable of communicating with either the system 120 or middleware server 116. In one embodiment, client 112 can include a web browser 114, which operates as an interface between the user 110 and the system 120. In another embodiment, the client 112 can be a voice communication device that communicates with the middleware server 116, which can include a voice browser 118. In these embodiments, specific instances of the client 112 can include a computer, a web station, a media player, a telephone, a smart phone, and the like.

Web 2.0 server 120 can be a server 120 that provides Web content to interface 114 and/or 118 and which permits a user 110 to provide additional Web content, which is made available to other users. The Web 2.0 server can be a WIKI server, a BLOG server, a social networking server, a MASHUP server, a FOLKSONOMY server, and the like. In one embodiment, the Web 2.0 server 120 can be a RESTful server, in which case functionality shown for server 130 can be incorporated within server 120. Alternatively, a transformer can be included in Web 2.0 server 120 which converts content between a server-specific format (e.g., a WIKI format) and a RESTful format, such as a format adhering to an APP-based protocol.

RESTful server 130 and 162 can be a server adhering to REST concepts, which links the server 120 to speech processing server 150. In one embodiment, the RESTful server 130 can be an APP server. RESTful commands can be issued by command engine 132, which are received and processed by command interpreter 154. A media interface 136 of the RESTful server 130 can control caching, delivery, fidelity, and formatting of delivered media, which includes delivered speech. Delivery can be in accordance with a streaming protocol, a file-based protocol, a real-time protocol, and the like.

Speech processing server 150 can be any networked server or speech processing system which is able to process speech requests using one or more speech engines 156. In one embodiment, the speech processing server 150 can be a turn-based and/or clustered system capable of handling multiple requests in real-time. For example, speech processing server 150 can be implemented as a WEBSPHERE VOICE SERVER or other such commercially available product. Management tasks of the server 150 can be handled by the management processor 152. The various speech engines 156 can include ASR, TTS, SIV, voice markup interpreters, and the like.

Data stores 124, 138, 160, and 164 can be a physical or virtual storage space configured to store digital information. Data stores 124, 138, 160, and 164 can be physically implemented within any type of hardware including, but not limited to, a magnetic disk, an optical disk, a semiconductor memory, a digitally encoded plastic memory, a holographic memory, or any other recording medium. Each of the data stores 124, 138, 160, and 164 can be a stand-alone storage unit as well as a storage unit formed from a plurality of physical devices. Additionally, information can be stored within data stores 124, 138, 160, and 164 in a variety of manners. For example, information can be stored within a database structure or can be stored within one or more files of a file storage system, where each file may or may not be indexed for information searching purposes. Further, data stores 124, 138, 160, and 164 can utilize one or more encryption mechanisms to protect stored information from unauthorized access.
The components of system 100 can be communicatively linked to each other via a network (not shown). The network can include any hardware/software and firmware necessary to convey data encoded within carrier waves. Data can be contained within analog or digital signals and conveyed through data or voice channels. The network can include local components and data pathways necessary for communications to be exchanged among computing device components and between integrated device components and peripheral devices. The network can also include network equipment, such as routers, data lines, hubs, and intermediary servers which together form a data network, such as the Internet. The network can also include circuit-based communication components and mobile communication components, such as telephony switches, modems, cellular communication towers, and the like. The network can include line based and/or wireless communication pathways.

In system 200, Web 2.0 clients 240 can communicate with Web 2.0 servers 210-214 utilizing a REST/ATOM 250 protocol. The Web 2.0 servers 210-214 can serve one or more speech-enabled applications 220-224, where speech resources are provided by a Web 2.0 voice system 230. One or more of the applications 220-224 can include AJAX 256 or other JavaScript code. In one embodiment, the AJAX 256 code can be automatically converted from WIKI or other syntax by a transformer of a server 210-214.

Communications between the Web 2.0 servers 210-214 and system 230 can be in accordance with REST/ATOM 256 protocols. Each speech-enabled application 220-224 can be associated with an ATOM container 231, which specifies Web 2.0 items 232, resources 233, and media 234. One or more resource 233 can correspond to a speech engine 238.

The Web 2.0 clients 240 can be any client capable of interfacing with a Web 2.0 server 210-214. For example, the clients 240 can include a Web or voice browser 241 as well as any other type of interface 244, which executes upon a computing device. The computing device can include a mobile telephone 242, a mobile computer 243, a laptop, a media player, a desktop computer, a two-way radio, a line-based phone, and the like. Unlike conventional speech clients, the clients 240 need not have a speech-specific interface and instead only require a standard Web 2.0 interface. That is, there are no assumptions regarding the client 240 other than an ability to communicate with a Web 2.0 server 210-214 using Web 2.0 conventions.

The Web 2.0 servers 210-214 can be any server that provides Web 2.0 content to clients 240 and that provides speech processing capabilities through the Web 2.0 for voice system 230. The Web 2.0 servers can include a WIKI server 210, a BLOG server 212, a MASHUP server, a FOLKSONOMY server, a social networking server, and any other Web 2.0 server 214.

The Web 2.0 for voice system 230 can utilize Web 2.0 concepts to provide speech capabilities. A server-side interface is established between the voice system 230 and a set of Web 2.0 servers 210-214. Available speech resources can be introspected and discovered via introspection documents, which are one of the Web 2.0 items 232. Introspection can be in accordance with the APP specification or a similar protocol. The ability for dynamic configuration and installation is exposed to the servers 210-214 via the introspection document.

That is, access to Web 2.0 for voice system 230 can be through a Web 2.0 server that lets users (e.g., clients 240) provide their own customizations/personalizations. Appreciably, use of the APP 256 opens up the application interface to speech resources using Web 2.0, JAVA 2 ENTERPRISE EDITION (J2EE), WEBSPHERE APPLICATION SERVER (WAS), and other conventions, rather than being restricted to protocols, such as media resource control protocol (MRCP), real time streaming protocol (RTSP), or real time protocol (RTP).

A constrained set of RESTful commands can be used to interface with the Web 2.0 for voice system 230. RESTful commands can include a GET command, a POST command, a PUT command, and a DELETE command, each of which is able to be implemented as an HTTP command. As applied to speech, GET (e.g., HTTP GET) can return capabilities and elements that are modifiable. The GET command can also be used for submitting simplistic speech queries and for receiving query results.

The POST command can create media-related resources using speech engines 238. For example, the POST command can create an audio "file" from input text using a text-to-speech (TTS) resource 233 which is linked to a TTS engine 238. The POST command can create a text representation given an audio input, using an automatic speech recognition (ASR) resource 233 which is linked to an ASR engine 238. The POST command can create a score given an audio input, using a Speaker Identification and Verification (SIV) resource which is linked to a SIV engine 238. Any type of speech processing resource can be similarly accessed using the POST command.

The PUT command can be used to update configuration of speech resources (e.g., default voice-name, ASR or TTS language, TTS voice, media destination, media delivery type, etc.) The PUT command can also be used to add a resource or capability to a Web 2.0 server 210-214 (e.g., installing an ASR component). The DELETE command can remove a speech resource from a configuration. For example, the DELETE command can be used to uninstall a previously installed speech component.

The Web 2.0 for Voice system 230 is an extremely flexible solution that permits users (of clients 240) to customize numerous speech processing elements. Customizable speech processing elements can include speech resource availability, request characteristics, result characteristics, media characteristics, and the like. Speech resource availability can indicate whether a specific type of resource (e.g., ASR, TTS, SIV, Voice XML interpreter) is available. Request characteristics can refer to characteristics such as language, grammar, voice attributes, gender, rate of speech, and the like. The result characteristics can specify whether results are to be delivered synchronously or asynchronously. Result characteristics can alternatively indicate whether a listener for callback is to be supplied with results. Media characteristics can include input and output characteristics, which can vary from a URL reference to an RTP stream. The media characteristics can specify a codec (e.g., G.711), a sample rate (e.g., 8 KHz to 22 KHz), and the like. In one configuration, the speech
engines 238 can be provided from a J2EE environment 236, such as a WAS environment. This environment 236 can conform to a J2EE Connector Architecture (JCA) 237.

In one embodiment, a set of additional facades 260 can be utilized on top of Web 2.0 protocols to provide additional interface and protocol 262 options (e.g., MRCP, RTSP, RTP, Session Initiation Protocol (SIP), etc.) to the Web 2.0 for voice system 230. Use of facades 260 can enable legacy access/use of the Web 2.0 for voice system 230. The facades 260 can be designed to segment the protocol 262 from underlying details so that characteristics of the facade do not bleed through to speech implementation details. Functions, such as the WAS 6.1 channel framework or a JCA container, can be used to plug-in a protocol, which is not native to the J2EE environment 236. The media component 234 of the container 231 can be used to handle media storage, delivery, and format conversions as necessary. Facades 260 can be used for asynchronous or synchronous protocols 262.

FIG. 3 is a schematic diagram showing a WIKI server 330 adapted for communications with a Web 2.0 for voice system 310 in accordance with an embodiment of the inventive arrangements disclosed herein. Although a WIKI server 330 is illustrated, server 330 can be any WEB 2.0 server (e.g., server 120 of system 100 or server 210-214 of system 200) including, but not limited to, a BLOG server, a MASHUP server, a FOLKSONOMY server, a social networking server, and the like.

In the system 300, a browser 320 can communicate with Web 2.0 server 330 via Representational State Transfer (REST) architecture / ATOM 304 based protocol. The Web 2.0 server 330 can communicate with a speech for Web 2.0 system 310 via a REST/ATOM 302 based protocol. Protocols 302, 304 can include HTTP and similar protocols that are RESTful by nature as well as an Atom Publishing Protocol (APP) or other protocol that is specifically designed to conform to REST principles.

The Web 2.0 server 330 can include a data store 332 in which applications 334, which can be speech-enabled, are stored. In one embodiment, the applications 332 can be written in a WIKI or other Web 2.0 syntax and can be stored in an APP format.

The contents of the application 332 can be accessed and modified using editor 350. The editor 350 can be a standard WIKI or other Web 2.0 editor having a voice plug-in or extensions 352. In one implementation, user-specific modifications made to the speech-enabled application 334 via the editor 350 can be stored in customization data store as a customization profile and/or a state definition. The customization profile and state definition can contain customization settings that can override entries contained within the original application 332. Customizations can be related to a particular user or set of users.

The transformer 340 can convert WIKI or other Web 2.0 syntax into standard markup for browsers. In one embodiment, the transformer 340 can be an extension of a conventional transformer that supports HTML and XML. The extended transformer 340 can be enhanced to handle JAVA SCRIPT, such as AJAX. For example, resource links of application 332 can be converted into AJAX functions by the transformer 340 having an AJAX plug-in 342. The transformer 340 can also include a VoiceXML plug-in 344, which generates VoiceXML markup for voice-only clients.

The present invention may be realized in a centralized fashion in one computer system or in a distributed fashion where different elements are spread across several interconnected computer systems. Any kind of computer system or other apparatus adapted for carrying out the methods described herein is suited. A typical combination of hardware and software may be a general purpose computer system with a computer program that, when being loaded and executed, controls the computer system such that it carries out the methods described herein.

The present invention may also be embodied in a computer program product, which comprises all the features enabling the implementation of the methods described herein, and which when loaded in a computer system is able to carry out these methods. Computer program in the present context means any expression, in any language, code or notation, of a set of instructions intended to cause a system having an information processing capability to perform a particular function either directly or after either both of the following: a) conversion to another language, code or notation; b) reproduction in a different material form.

This invention may be embodied in other forms without departing from the spirit or essential attributes thereof. Accordingly, reference should be made to the following claims, rather than to the foregoing specification, as indicating the scope of the invention.

What is claimed is:

1. A speech processing system comprising:
a client configured to access a speech-enabled application using at least one Web 2.0 communication protocol;
a speech for Web 2.0 system within which the speech-enabled application executes, said speech for Web 2.0 system accessing a data store within which user-specific speech parameters are included, wherein a user of the client is able to configure the specific speech parameters of the data store associated with the user, and wherein the speech-enabled application executes in accordance with the specific speech parameters corresponding to the user of the client; and
a speech processing system comprising a plurality of speech processing engines, wherein the speech processing system interacts with the speech for Web 2.0 system to handle speech processing tasks associated with the speech-enabled application.

2. The system of claim 1, wherein the specific speech parameters specify at least one of speech resource availability, speech resource characteristics, and speech delivery characteristics.

3. The system of claim 1, wherein the Web 2.0 communication protocol is a Hypertext Transfer Protocol (HTTP) based protocol, and wherein the speech processing system interfaces with the speech for Web 2.0 system using an Atom Publication Protocol (APP) based protocol.

4. The system of claim 1, wherein interactions between the speech processing system and the speech for Web 2.0 system occur through one of four RESTful commands, said RESTful commands comprising a GET command, a POST command, a PUT command, and a DELETE command.

5. The system of claim 1, wherein said speech-enabled application comprises at least one introspection document, which is used to enable the client to configure the specific speech parameters.

6. The system of claim 1, wherein the speech-enabled application comprises two collections, one of these collec-
tions comprising at least one entry, each entry defining content that is presented to the client, the other one of the collections comprising a collection of resources that include speech processing resources, wherein a one-to-one relationship exists between the speech processing resources of the collection of resources and a type of speech comprising engine of the speech processing system to which the speech processing resource corresponds, said types of speech processing engines including at least two of a recognition engine, a text-to-speech engine, a speech identification and verification (SIV) engine, and a VoiceXML interpreter.

7. The system of claim 1, wherein the speech-enabled application is at least one of a WIKI, a BLOG, a MASHUP, a social networking application, and a FOLKSONOMY.

8. The system of claim 1, wherein the client comprises a standard Web browser through which the client interfaces with the speech for Web 2.0 system, wherein the Web 2.0 communication protocol is directly supported by the standard Web browser.

9. The system of claim 1, further comprising:
   a middleware server comprising a standard voice browser,
   wherein said client interacts with the middleware server over a real-time voice communication channel, wherein the standard voice browser interfaces with the speech for Web 2.0 system, wherein the Web 2.0 communication protocol is directly supported by the standard voice browser.

10. The system of claim 1, further comprising:
    an enterprise server comprising enterprise content,
    wherein the enterprise server interacts with the speech for Web 2.0 system to permit the client to access the enterprise content by interacting with the speech-enabled application.

11. A system for using Web 2.0 as an interface to speech engines comprising:
    a Web 2.0 server configured to serve at least one speech-enabled application to at least one remotely located client; and
    a server-side speech processing system configured to handle speech processing operations for the at least one speech-enabled application, wherein communications with the server-side speech processing system occur via a set of RESTful commands.

12. The system of claim 11, wherein Web 2.0 server utilizes at least one introspection document associated with the speech-enabled application for introspection and discovery of speech resources and to configure the speech resources.

13. The system of claim 12, wherein the introspection document and the RESTful commands conform to an Atom Publication Protocol (APP) based specification.

14. The system of claim 11, wherein the set of RESTful commands comprise an HTTP GET command, an HTTP POST command, an HTTP PUT command, and an HTTP DELETE command.

15. The system of claim 14, wherein said GET command selectively returns modifiable speech processing capabilities and elements, said GET command also selectively returning speech query results, wherein said POST command selectively provides input to a speech engine and returning output from the speech engine, said output being a processed result of the input, wherein said PUT command selectively updates speech resources for a configuration, said PUT command also selectively installing a speech resource for a configuration, and wherein said DELETE command selectively removes a speech resource from a configuration.

16. The system of claim 11, wherein the set of RESTful commands consist of an HTTP GET command, an HTTP POST command, an HTTP PUT command, and an HTTP DELETE command.

17. A speech for Web 2.0 system comprising:
    a Web 2.0 server configured to serve at least one speech-enabled application to remotely located clients, said speech-enabled application comprising an introspection document, a collection of entries, and a collection of resources, wherein at least one of the resources is a speech resource associated with a speech engine, which adds a speech processing capability to the speech-enabled application.

18. The system of claim 17, wherein the speech-enabled application conforms to an Atom Publication Protocol (APP) based specification.

19. The system of claim 17, wherein the speech engine is a turn-based speech processing engine executing within a JAVA 2 ENTERPRISE EDITION (J2EE) middleware environment.

20. The system of claim 17, wherein the Web 2.0 server is configured so that end-users are able to introspect, customize, replace, add, re-order, and remove entries and resources in the collections.