

[0039] FIG. 17 is a diagram showing a supply chain of producer-consumer relationships between a client application and the lowest level web services;

[0040] FIG. 18 is a diagram showing bundling on a per-client application basis in accordance with an embodiment of the present invention; and

[0041] FIG. 19 is a flowchart showing a method for billing and authorization of web services in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0042] FIG. 2 shows a web services infrastructure 201 in accordance with an embodiment of the present invention. The web services infrastructure 201 comprises client applications 15, web service providers 20, a gateway module 300, client application connections 31 to the gateway module 300, and web service connections 32 to the gateway module 300. More or fewer client applications 15, web services 25, and their respective connections 31, 32, may exist in the web services infrastructure 201.

[0043] The client applications 15 may be end user applications 10, other web service providers 20, or any application which requests access to a web service 25. Some client applications 15 may be used by client application developers to obtain a web service application programming interface (API) contract to develop other client applications 15 that will consume or use that particular web service 25. Each client application 15 has a client application connection 31. Client application connections 31 may be any suitable connection that allows the transfer of information between the client application 15 and the gateway module 300. The web service providers 20 have web services WS1 to WSx, WS1 to WSy, and WS1 to WSz, where x, y, and z are integers greater than zero. The web services 25 may be different for each web service provider 20. Each web service 25 has a web service connection 32 to the gateway module 300. Web service connections 32 may be any suitable connection that allows the transfer of information between the web services 25 and the gateway module 300. The gateway module 300 is an application that adds common and additional functionality, as will be described below, to any web service 25 registered with the gateway module 300. The gateway module 300 remains transparent to the client application 15 and the web service 25. Web services 25 are registered with the gateway module 300 as described below. In this description, clients may sometimes be referred to as users, consumers, and/or end users.

[0044] FIG. 3 shows an example of a gateway module 300 in accordance with an embodiment of the present invention. The gateway module 300 comprises a client application interface unit 310, a communication processor 311 and a web services interface unit 312. These components of the gateway module 300 comprise code that may be executed as software or embedded in hardware. The client application interface unit 310 receives web services requests coming from a client application connection 31 (shown in FIG. 2) connecting a client application 15 (shown in FIG. 2) with the gateway module 300. The client application interface unit 310 sends web services requests to the communication processor 311. The communication processor 311 determines which web service 25 is being requested and sends the

web service request to the web services interface unit 312. The web services interface unit 312 sends the web service request to the appropriate web service 25 through the web service connection 32 (shown in FIG. 2) connecting that web service 25 with the gateway module 300. Other components may be added to the gateway module 300 as described below.

[0045] Alternatively, the gateway module 300 may not include a client application interface unit 310, whereby the function of the client application interface unit 310 is performed by either the communication processor 311 or an external module. Parts of the remainder of this application will refer to a gateway module 300 containing a client application interface unit 310. However, the client application interface unit 310 may be removed, and the communication processor 311 modified, as described above.

[0046] As is described above, the gateway module 300 may be used by both client application developers and client application users. When developing client applications 15 that may use a web service 25, client application developers may use the gateway module 300 to obtain a web service API contract from a web service 25. When a client application 15 is used by a client application user, the client application 15 may use the gateway module 300 to send a method call to a web service. Some method calls instruct a web service to perform a task. Some method calls instruct a web service to return an item or response. The web service 25 may use the gateway module 300 to return an item or response to the client application 15.

[0047] FIG. 4 shows a method for managing functionality for one or more web services 25 (400) in accordance with an embodiment of the present invention. The method (400) begins with receiving a web service request from a client application 15 (401). This step may be performed by the client application interface unit 310, or the communication processor 311 as described above. Next, the web service request is processed (402). This step may be performed by the communication processor 311. Finally, the web service request is delegated to the appropriate web service 25 (403). This step may be performed by the web services interface unit 312. Once the appropriate web service 25 has the web service request (403), the method is done (404). Further steps may be added to this method (400) as described below.

[0048] An aspect of an embodiment of the gateway module 300 pertains to the field of distributed computing, where software running on a client system interacts with software running on remote server systems. More specifically, where the software running on a server has been developed such that its capabilities can be discovered programmatically using the tools based on a standard network protocol based on a standard language. An example of a standard network protocol is the Internet protocol known as simple object access protocol (SOAP), which is itself based on the standard extensible markup language (XML). The actual data transmitted between the client application 15 used by client application user and the server code may be transmitted via SOAP. The server functionality is typically referred to as web services. A typical client application 15 may be either a web browser or an Internet-aware application. The following description refers mainly to SOAP communication, but the gateway module 300 may be used with other standard protocols based on a standard language.