

[0056] An issued authentication ID may be passed as a parameter with subsequent web service method calls invoked by the client application 15. The client application interface unit 15 receives a web service method call (624) and passes the method call to the communication processor 311. The communication processor 311 determines which web service 25 is associated with the method call. The method call is then passed to the authorization module 525 to determine if the client application 15 associated with the authentication ID is authorized to use the method call in the web service 25 (625). For example, authorized client applications 15 for methods in web services 25 may be listed in a repository. If the client application 15 is not authorized to use the method call (625), then the method call is rejected (632) and the method is done (633).

[0057] If the client application 15 is authorized to use the method call (625), then the method call is passed to the communication processor 311 to be processed (626), i.e., prepared to be delegated to the web service 25. The authentication ID parameter is removed from the method call and the modified method call is passed to the web service interface unit 312. The web services interface unit 312 searches the web service registry repository 530 to determine the location of the requested web service 25, as described above. The method call is delegated to the web service 25 via the corresponding web service connection 32 (627).

[0058] If the method call does not have a corresponding response (628), then the method is done (633). Otherwise, a response is received from the web service 25 (629) and sent to the client application 15 (630), as described above. The method is done (633).

[0059] FIG. 6C shows another example of a method for managing functionality for one or more web services 25 (650) in accordance with an embodiment of the present invention. This example relates to a client application developer. To request an API of a web service 25, a client application 15 sends an API request method call (API request) through the client application connection 31. The client application interface unit 310 receives the API request and passes the API request to the communication processor 311 (651). The communication processor 311 determines which web service 25 is associated with the API request. The API request is then passed to the web services interface unit 312. The web services interface unit 312 searches the web service registry repository 530 to determine the location of the requested web service API (652). The web service registry repository 530 provides a mapping from the web service 25 URI to the physical location of the web service 25 and any other attributes of the web service 25 that are desirable to assist the gateway module 500 to interpret, process, and make actual requests or method calls of said web service 25. Once the location of the web service 25 is determined (652), the API request is delegated to the web service 25 via the corresponding web service connection 32 (653).

[0060] The web services interface unit 312 receives the requested web service API contract from the web service 25 (654) in response to the API request. The web services interface unit 312 passes the API contract to the communication processor 311 to be passed back to the client application interface unit 310. The client application interface

unit 310 sends the response through the appropriate client application 31 to the client application 15 from which the API request originated (655). The method is done (656). Alternatively, the method may include access restriction measures such as requirements for authentication and/or authorization, as described above with respect to web service method calls.

[0061] The gateway module 500 is transparent to the client application 15 and the web services 25. The gateway module 500 processes communication between client application 15 and web service 25. In the case of a client application developer, a single entry point, i.e., a single address, is exposed for client application developers to retrieve a web service API contract document, such as WSDL. WSDL contains a port, which is a single address in which to bind a client application 15. A WSDL returned by normal web services 25 contains its own address. Hence if a client application 15 has ten web services, the client application 15 will require ten addresses to connect to these web services 25 (as depicted in FIG. 1). An API contract processor in the communication processor 311, such as a WSDL processor, replaces this port address with the address of the gateway module 500 before returning the WSDL back to the client application developer. This ensures that the client application 15 that will be created by the client application developer will send its SOAP requests to the address that is associated with the client application interface unit 310. Subsequent SOAP requests sent by the client application 15 are received by the client application interface unit 310 and delegated to a web method call processor, such as a SOAP processing module, in the communication processor 311.

[0062] Two scenarios are described above: A) requests from a client application user (method calls over SOAP); and B) requests from a client application developer for the API contract (WSDL). In scenario A, the gateway module 500 acts as a SOAP processor. Scenario A occurs over the SOAP protocol whenever a method of the web service 25 is invoked. In scenario B, the gateway module acts as an API contract (WSDL) processor. Scenario B does not occur over SOAP and occurs during the design/development time of the client application 15 consuming the web service 25. In both scenarios, there may be two areas of processing: i) the processing of the request before the request is forwarded to the web service 25; and ii) if there is a response, the processing of the response before it is returned to the client application 15.

[0063] This description contains references to login and logon procedures. The embodiments of the inventions described in this specification apply to both login and logon procedures. A login reference is intended to include a logon reference and vice versa.

[0064] A client application user or client application developer may call a Login method, passing his/her credentials, through the use of a client application 15. The credentials may be authenticated and a list of web services 25 to which the client application 15 is authorized to access may be established and stored in a repository. After authenticating the credentials, the gateway module 500 returns an authentication ID to be passed with every method call. Advantageously, this simplifies the authentication and authorization checking steps in the management of web service 25 functionality. The use of authentication IDs will be described further below.