

[0065] In order to achieve the transparency described above, client application developers may use parameters added to and/or modified from existing parameters to method calls of web services 25 API contract. These extra and/or modified parameters are created by an administrator or developer of the gateway module 500 and assist the gateway module 500 to distinguish method calls from one client application 15 from method calls of another client application 15. Furthermore, the extra and/or modified parameters assist in other administrative functions such as classification and storage of the web service 25 method calls, and storage of authentication IDs. The API contract request method may also include extra parameters. The areas of processing referred to above relate to i) transformations between the method call received from the client application 15 and the method call passed to the web service; and ii) transformations between the response, if any, received from the web service and the response passed to the client application 15.

[0066] FIG. 7A shows a depiction 701 of the web services infrastructure 501 as a SOAP processor in accordance with an embodiment of the present invention. A client application 15 (such as an end user application used by a client application user) invokes a SOAP method call (client call) 701. The client call 701 is received by the gateway module 500, as described above. The client call 701 is modified by a SOAP processor of the communication processor 311 of the gateway module 500. The modification here may include the removal of an extra parameter which was added to the web service SOAP method call by the client application developer. Thus the modification translates the client call 701 into the actual web service SOAP method call (WS call) 702 understood by the web service 25. The WS call 702 is sent by the gateway module 500 to the web service 25, as described above.

[0067] The gateway module 500 receives a web service SOAP response (WS response) 703 from the web service 25, as described above. The SOAP processor of the communication processor 311 translates the WS response 703 into a SOAP based response (client response) 704 that the client application 15 will understand. The translation may be the addition or modification of a parameter in the WS response. The gateway module 500 then sends the client response 704 to the client application 15, as described above.

[0068] The ability to act as a SOAP processor allows the functionality of the web services 25 to be better managed. Since the web services infrastructure 501 is listening to SOAP communication between client 15 and web service 25, the web services infrastructure 501 has the information to determine what methods 701 are being called, under what conditions and even by whom, provided that identification information was given by the caller. Furthermore, the web services infrastructure 501 is able to check authentication, authorization, and/or billing information and determine if the method call should be allowed to proceed to the service. When the response from the web service 25 returns, the web services infrastructure 501 is then able to update any relevant billing or audit information, as described below. In general the web services infrastructure 501 can perform infrastructure functions common to the related web services 25 because the web services infrastructure 501 is privy to the information passed from client application 15 to web service 25.

[0069] FIG. 7B shows a depiction 750 of the web services infrastructure 501 as an API contract processor in accordance with an embodiment of the present invention. A client application 15 (such as a programmer application used by a client application developer) may make a method call (developer call) 751 requesting the API contract of a web service 25. The developer call 751 is received by the gateway module 500, as described above. The developer call 751 is modified by an API contract processor of the communication processor 311 of the gateway module 500. The modification here may include the removal of an extra parameter which was added to the API contract method call. Thus the modification translates the developer call 751 into the actual API contract method call (API call) 752 understood by the web service. For example, an HTTP GET request method call may be used to obtain a web service API contract. The API call 752 is sent by the gateway module 500 to the web service 25, as described above.

[0070] In response to the API call 752, the gateway module 500 receives an API contract (for example, WSDL) 753 from the web service 25 as described above. The API contract processor of the communication processor 311 translates the WSDL 753 into a modified WSDL 754 that the client application developer will use. The translation may be the addition or modification of a parameter in the WSDL 753, such as the modification of an address, as described above. The gateway module 500 then sends the modified WSDL 754 to the client application 15, as described above.

[0071] By modifying the API contract (WSDL) 753 as it is delivered from the web service 25 to the client application 15, the web services infrastructure 501 is able to enforce the requirement of the consuming client application developer to insert parameters into any or all method calls in that web service 25. This allows the newly developed client application 15 to believe the added or amended parameters are part of the web service 25 method being invoked while the web service 25 is not even aware that parameters exist. When the new client application 15 calls the method 701, the SOAP processor in the gateway module 500 strips out the extra parameters, performs any processing required, and passes the SOAP message (minus the extra parameters) 702 onto the web service 25. For example, the web services infrastructure 500 can receive and process a user identifier parameter which allows for the association of method calls 701, 702 to a given client application 15 and all of the authorization and billing information appropriate for that client application 15. The web services 25 are not concerned with the identifier and the identifier need not be a parameter for any of their methods 702. From the client application 15 point of view, however, the methods 701 use this parameter.

[0072] FIG. 8 shows an example of a modification to an API contract through the gateway module 500 in accordance with an embodiment of the present invention. A web service API contract 753 may contain many method calls, i.e., WS calls 702. As an example, FIG. 8 shows a web service WS1 API contract 753 as containing 50 WS calls 702. Three examples of WS calls 702 given in FIG. 8 include:

[0073] method1(string param1, int param2, MyType param3);

[0074] method2(char param1, MyType param2); and

[0075] method3( ).