Heterogeneous Synchronization Layer for Web Services

Sakthi Saravanakumar P, Mehanathen N, Rajagopalan M R

Centre for Development of Advanced Computing,

Chennai, India.

{sakthis, mehanathenn, mrr}@cdac.in

Abstract— **The web services model is used for interoperability of Heterogeneous and distributed platforms. In this paper we present how different orchestrated and choreography based web services collide each other and exchange information. Exchanging of information takes place by means of integrating choreography Web services with push based web service architecture. In global scenario application are built with web services for their own scenario. Each services follow different set of mechanism for e.g. Web services A follows push based service and service B follows poll based services and so on. In order to exchange the information between these two web-services we introduce a layer which acts as a common platform for service synchronization. We call this layer as Heterogeneous Synchronization layer (HSL). This layer acts as a bridge between global web services and our subscribed web services.**

Keywords— **Web services, choreography, heterogeneous interaction model, push based service architecture, interoperability, Heterogeneous synchronization layer.**

1. INTRODUCTION

The web services model is used for dynamic and flexible interoperability of heterogeneous and distributed web based platforms. Web service choreography is usually referred to web services specification which describes collaboration protocol of co-operating web services participant from the global point of view. An orchestration describes collaboration of the Web services in a predefined pattern based on local decision about their interactions with one another at the message/execution level [1].There are three types of web services architecture traditional, pull and pushing. In the traditional architecture the requested service finds the UDDI registry and invokes the required services. In the pull based web service architecture service provider sends notification to the broker. The broker sends the notification messages to all the subscribed services, which in turn make the client services, invoke the provider. In the push based architecture client subscribes the registration to the server. The server stores information about the clients whenever changes happen in the server it thrusts the message to the subscribed client [6].

An idea of our proposed architecture is integrating the choreography web services and the traditional web services for synchronization of message exchanges. For implementation of choreography web services we have followed LCC standards (Light weight Co-ordination Calculus), which is discussed in section 4. In order to achieve internal architecture of HSL we have used Java api's and PostgreSQL for Data Storage.

The rest of the paper is organized as follows. In section 2, related work regarding various service communication methodologies. In section 3 we discussed about proposed architecture and its components. Section 4 describes implementation details. Section 5 Results and discussion of the HSL layer. Section 6 contains conclusion and further we conclude with the future work.

2. RELATED WORKS

Web Services are a network of application that interacts using standard protocols over well-defined interfaces. The integration of web services can be static or dynamic. The main difference between these two web services interaction is the finding of the related services, in static the related services are predefined while in dynamic it is at runtime identification [4].The web services communication has the following characteristics:

- Sending a sequence of messages to other participating web services, without any response
- Sending a sequence of messages to other participating web services, with a response variable

The communication between two web services is made through a communication channel. This communication will be either point to point or session communication [2]. In point-point communication all the necessary variable are available in the WSDL file. In session communication state is maintained, which ensures the correctness of interactions taking place between communicating processes. A sender and receiver of web services has to make a communication channel, both the services should have the same name for all the communicating variables. The implementation of web services is based on the behaviour of local variable which are described in the global scenario. Translating the global

description to its local end point is called End point Projection (EPP). Global communication of web services is implemented using choreography web services. Each choreography web services have its own EPP for communicating with local behaviours.

In this choreography implementation we define the global description and down the line implementation of the local behaviours. For these communications, two parties have to establish a private connection and do the series of interaction with these services [2]. For defining this sequence of service description there are several approaches available such as BPEL, Activiti, WS-CDL, calculus etc.

For implementing HSL we require the other scenario of web service implementation known as push based mechanism. In this approach the subscriber client registers the subscription details to the web service provider, whenever any changes occur in the participating web service provider the result message is thrusted to the subscribed clients [3]. This approach reduces the amount of un-necessary data send over the network, response time is shorter and bottle neck problems are eliminated.

Synchronization of a new web service participant in the current scenario leads to rework of existing code as well as the code of new participating web service. The HSL layer addresses this issue by integrating the global (Choreography) web services with push based web services. The HSL layer is developed by various component and these components are integrated with each other, which provides a high degree of reusability and increases the efficiency [5].

For example a citizen wants to change his address in one department, and it has to be automatically notified to all other related departments where his address is registered. Here in the current scenario services provided by these departments are assumed to be predefined. Any department which needs to avail these services have to redefine the existing global services.

3. PROPOSED ARCHITECTURE

The proposed architecture for Heterogeneous synchronization layer is depicted in Figure 1.

The HSL introduces a layer which acts as a common platform for service Synchronization/Composition. This layer consists of four major components:

- Service sequence definition
- Service integrator
- Thrusted services.
- Synchronization analyser

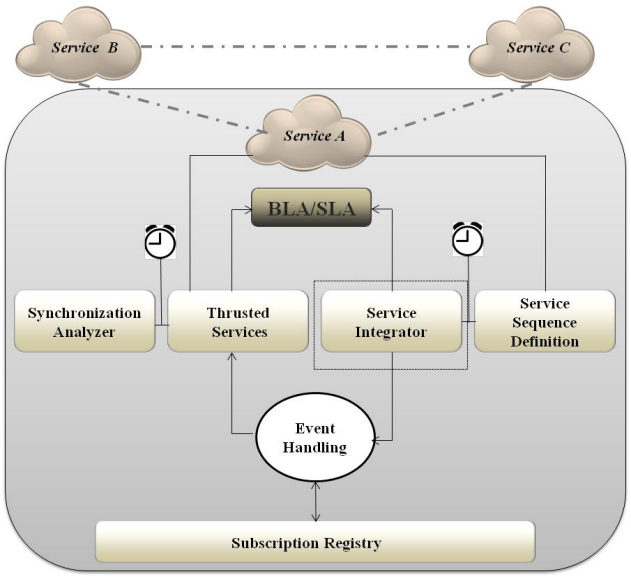

Figure 1. HETEROGENOUS SYNCHRONIZATION LAYER (HSL)

Service sequence definition:

All the global service definitions reside with the Service sequence definition component. This component communicates with other global services (Choreography services) .A sequence of messages are sent to other participating service in certain time intervals. It checks whether any changes have occurred in the participating global services. If any response is received from the participating global services, it invokes the Service Integrator component.

Service integrator:

Service integrator acts as a bridge between Choreography service and thrusted services. Whenever changes occur in the choreography services the data returned from the choreography services is pushed to the thrusted service.

Thrusted services:

This component implements the Business rules and policies between consumer and subscriber of our services. This component stores the subscription information in subscription registry. The registry contains information such as communicating variables, interface names of the subscriber, status flags, etc. This component helps to synchronize the message between global services and the subscriber services.

Synchronization analyser:

In synchronization analyser, the message between our services and the participating web services are checked in regular intervals if the messages are not synchronized a report is generated and notified to the administrator. The administrator triggers the sync event, thus the message is re-pushed to the subscriber.

4. IMPLEMENTATION

To achieve global communication of collaboration services (Choreography services) there are several tools and approaches available. Some of the implementation examples are available in SAVARA, LCC, Choreos, BPEL and Pi4soa [8] [9] [10] [11] .But these implementation states that all the services should be predefined within the participating services. If an external service needs to interact with these services, existing tools does not have a readily support mechanism.

The implementation of Push-Based Web Services component is divided into two components, one component to handle incoming events and subscriptions, and the other component for event processing and output [3].

The above two implementation choreography services and push based services are already implemented separately.

The service sequence definition (Choreography) is implemented by Light weight Co-ordination Calculus (LCC), which represents the co-ordination of different web services [7][8][9]. LCC has achieved Completeness, Compositionality, Parallelism, Resources compared to others [9]. It sends sequences of messages to the participating global services at certain intervals.

The message communication will be of two types one will be of requester and the other will be collaborator. The message flow is illustrated below:

Requester message definition:

```
a(requester,variable1)::
        get(X)=>a(collaborator,variable2)
        send(X)<=a(collaborator,variable2)
```

Collaborator message definition:

```
a(collaborator,variable2)::
        get(X)<=a(requester,variable1)
        send(X)=>a(requester,variable1)
```

The interpretation of LCC message constraints depends on a particular implementation. The interpretation of LCC is implemented by any programming language such as JAVA.

The Service sequence definition (Choreography services) can be implemented by the various approaches like work flow implementation and service composition using tools like switch yard, petals ESB, etc.

The other scenario of web services implementation is known as push-based implementation of services. Here the subscriber client needs to subscribe the services from our services i.e. the request is to be sending to our services in turn our service sends the list of available services with subscriptionID to the client. This subscriptionID is stored in the subscription registry, which helps us to identify the subscribed clients. Now the client can choose services from the list of defined services as per its needs and sends back the messages with the following parameters

- SubscriptionID
- Service consumption parameter
- Service acceptance parameter
- Communication parameters

SubscriptionID:

SubscriptionID is a unique identifier for each participating subscribed services. This is used to find out list of services subscribed by the clients.

Service consumption parameter:

It describes which services the client needs to consume from the service layer. The value of this parameter should be same as the name of the interface defined in our services.

Service acceptance parameter:

Service acceptance parameter describes how our service layer communicates with the participating web services layer. The value of this parameter should be same as the name of the interface defined in the subscribed services.

Communication parameters:

Communication parameter is an optional parameter which describes the Business Level Agreement (BLA) / Service Level Agreement (SLA) attributes.

For Implementing the proposed HSL we should consider the following parameters:-

- Service parameter
- Service Interface
- Logging
- Exception Handling
- Failure Transactions

The Service parameter contains sending and receiving variables. Both sending and receiving variables should be equal in number. The service integrator component has many service interface method, which is used to communicate between global services and pushed services. This interface definition depends on particular implementation. All the transactions variables are logged for the future tracking. In the exception handling section it keeps on checking whether any service link has failed due to errors. These errors are notified to the service administrator. In a failure transaction the

transaction of service which has failed are logged in the failure transaction records.

The Service integrator checks local database as well as the response message received in local communicating variable. If any change occurs in the data received from global communication immediately the Service integrator gets activates the Thrusted Service component, which is handled by the events. If communication from the client service fails, this gets notified and stores in the storage repository. Our Service keeps on checking the client service at regular intervals and as soon the client service resume it pushes the already lost message. Hence the participating global services and the subscribed services are always synchronized.

5. RESULTS AND DISCUSSION

There are two major issues identified during web services integration

- Messaging format
- Atomicity in co-ordination of services

In Messaging format, the communication protocols used are SOAP or Data sets. For communication between two layers the message format should be of pre-defined. When the communication protocols are dynamic the message processing / Communication will be complex.

In the existing scenario we identify the service communication types in the service registration level. Whenever the subscriber wants to register the service with us, the mode of communication is to be selected by the subscribing web service. Our Heterogeneous Synchronization Layer (HSL) supports SOAP as well as JSON format.

In co-ordination of services we define the attributes and requirements of different services. Synchronization between two layers needs to be atomic. In some cases atomicity fails i.e. When a Service A pushes a message to another Service B sometimes due to communication link failure the message could not be reached to service B. If service B carries out its task without proper message from service A there will be a mismatch of messages between participating services.

6. CONCLUSIONS

Since applications are developed using their own standards way of communication our proposed HSL is very much needed in order to achieve synchronization between participating global service and local services. This layer is to be used for data synchronization as a future work we would like to make this layer standardized and used in all the applications.

For example in real-time scenario the currency exchange rates in banks in India differ. This is due to the access of web service by banks at different times, provided by the Reserve Bank of India (RBI). This happens due to the delay in access of services. Therefore synchronization on web services by one end point is needed, which is addressed by HSL. The HSL provide facility for RBI to have the global communication of web service and the result is pushed to the implementing banks.

7. FUTURE WORK

We would like to take communication link failure between participating web services as a future work as part of our research.

8. ACKNOWLEDGEMENT

We thank the management of Centre for Development of Advanced Computing, Chennai management for their support to this research. This research is part of National Resource Centre for Free and Open Source Software phase-II project.

9. REFERENCES

[1] Jun Sun, Yang Liu, Jin Song Dong, Geguang Pu, Tian Hut Tan "Model-Based Methods for Linking Web Service Choreography and Orchestration", Asia Pacific Software Engineering Conference, 2010.

[2] Marco Carbone, Kohei Honda Nobuko Yoshida, Robin Milner, Gary Brown Steve, Ross-Talbot, “A Theoretical Basis of Communication-Centred Concurrent Programming”, pp. 5-6.

[3] Lars Brenna, Dag Johansen, "Configuring push-based Web services", 2005.

[4] Zhile Zou, Zhenhua Duan and Jianli Wang, “A Comprehensive Framework for Dynamic Web Services Integration, Web Services”, Proceedings of the European Conference on Web Services (ECOWS'06).

[5] J. L. Herrero, F. Lucio, P. Carmona, "Web services and web components", 2011.

[6] Thanisa Numnonda and Rattakorn Poonsuph, "Inversion of Web Service Invocation using Publish/Subscribe Push-Based Architecture",International Journal of Computer Science Issues, Vol. 9, Issue 1, No 3, January 2012.

[7] http://groups.inf.ed.ac.uk/OK/drupal/resources

[8] http://ccsl.ime.usp.br/baile/

[9] http://ccsl.ime.usp.br/wiki/images/7/79/Choreographies.pdf

[10] http://www.jboss.org/savara

[11] http://www.choreos.eu/