Integrating Heterogeneous Web Services into a Seamless Application

Sanjeev Neupane, Rajendra Pandey, Shrabani Mallick, Shakti Mishra, D.S. Kushwaha
CSED, Motilal Nehru National Institute of Technology, Allahabad, India
{cs0923, cs0920, shrabani, shaktimishra, dsk}@mnnit.ac.in

Abstract—In the present scenario, Composition and Adaption of heterogeneous web services is a challenging issue faced by most of the business organizations. Microsoft’s VB.Net and C#.Net, Java’s Netbeans and Eclipse, PHP, etc. technologies are used to create web services. Visual Studio, Eclipse, Netbeans, etc. IDEs (Integrated Development Environment) are currently most used in industries. Legacy systems written in low level language such as C/C++ are still used in many organizations. In this paper, we propose an approach to convert legacy code to web services and composing heterogeneous services created in VB.net, Java and C into a single application. The proposed methodology offers tremendous scope of converting the legacy applications to the web service paradigm and compose the heterogeneous web services.

Index Terms—Web service Composition, Legacy Code, Heterogeneous web services, Service Oriented Paradigm

I. INTRODUCTION

The business model is changing day by day and adapting a legacy model with the new business model is not an easy task. A legacy system is an old method, technology, computer system, or application program that continues to be used, typically because it still functions for the users’ needs, even though newer technology or more efficient methods of performing a task are now available. A legacy system may include procedures or terminology which are no longer relevant in the current context, and may hinder or confuse understanding of the methods or technologies used. Most of the business organizations want to use legacy system because it is still working properly though it may lack latest technical functionality and may not be proper user friendly and it may cost lots of investment to convert them into new business model [9]. There are many legacy systems which are being used in the present scenario, those legacy systems must be integrate with new business model and might need to change to support the new requirements. The benefit of adding legacy system to business model is that it saves time and money of the business organization. Requirement Analysis for new system is not required if there is adoption of legacy code. With the rise of Service Oriented Architecture (SOA), these Business adoptions are possible with the help of web services. Web service is a standard used for exchanging data between applications. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet [1]. After creating a web services they are hosted on some server and can be invoked with the help of SOAP (Simple Object Access Protocol) message Request from the client side. A single organization can use different type of web services (Heterogeneous services) created on different frameworks like Java, Microsoft etc. Heterogeneous services, in software context, refer to services that have different aspects such as the interface, the implementation, the data, etc. Two services are heterogeneous if they are incompatible in some way. They may represent different information or may have different functionality or may have different security policies. If two services are the same and can interoperate, they are homogeneous. Federating or integrating homogeneous systems is presumably simpler than federating heterogeneous systems. So consider a scenario where a company is using heterogeneous web services while having a legacy system. In our proposed approach legacy system is first converted into web services and then a controller service is created to compose heterogeneous services. Two steps we require to fulfil this task, first legacy code is converted into web service and in second step composition of heterogeneous Web services and invoke them from Controller.

The remainder of the paper is organized as follows. Section 2, presents the related work. Motivational example is proposed in section 3. Section 4 discuss about approach used. The complete implementation aspects are highlighted in section 5. Results are given in section 6 followed by conclusion in section 7.

II. RELATED WORK

Web services are being implemented by most of the organization in the present time. Each Web service consists of WSDL (Web Service Description Language) which is just a XML (Extensible Markup Language) document. XML achieves a technical level which is cross-platform, cross-system, and cross-application without linking to language combining the practical requirement of current development of Web services. Goal of this paper is to achieve the Composition of Heterogeneous Web services. Various researchers have already proposed the many solutions for the same but a comprehensive solution is still missing out on various fronts.

For example Authors in [2] used a graph based method to compose the different homogeneous Web services, where the input and output parameter of all Web services are compared. On the basis of this comparison, a graph is created and Web services are composed. Different services may also be used to compose a single service. If the output of one
service acts as input for other and if the match occurs, the services are composed.

Similarly Zhang [3] propose a General Integration Model of Web Services (GIMWS) to solve heterogeneous Web Services integration provided by different sellers. By use of transforming and coordinating mechanisms, a uniform general service is proposed, which efficiently provides a general mechanism to integrate a large number of heterogeneous Web Services. GIMWS uses a multi-layers construction to implement Web services integration and finally a Composite Aggregation Service (CAS) is presented that is called by an application directly. A XML-based language, Web Service Description Language (WSDL) is used in GIMWS to describe the interfaces of services. GIMWS model encapsulates several Web Services, message format, message content and transaction action, offering services in different layers.

A system to emulate SSO (single sign on) in a heterogeneous Web Services environment is developed in [4], freeing the end user from repeatedly entering service credentials. This was motivated by the use case of composing multiple services, each with distinct authentication schemes. It assumes that it is unreasonable to expect to change the end service to suit the authentication, requirements but imposes unnecessary load on the end user to submit multiple credentials for the same task. The paper introduces a uni_ed authentication framework which stores service credentials in XML-based formats, and associates those credentials with users, roles, or organizations. This is a practical approach to deal with authentications for service oriented systems. The key factors distinguishing their framework from other SSO implementations is that it is not limited to a single trust domain or authentication protocol, and that is capable of storing arbitrary types of credentials, instead of a constrained set.

A flexible script language called WebXcript [5] is used for integrating heterogeneous Web services and automating E-commerce activities. It automates dialogues with traditional web-based (HTML) services and information sources. In addition, WebXcript is also compatible with contemporary XML technologies. In particular, WebXcript is in XML and therefore development tools for the WebXcript engine are robust and widely available, as well as script authoring tools for end-users. The applicability of the approach in E-commerce environments is also demonstrated with detail examples as integrating heterogeneous information over the web, automating online ordering, and messaging with XML. Compared with other programming approaches, WebXcript, though simple, is more application-oriented, with tailor-made primitives for database access, web dialogue, and exception handling, and is therefore much easier to use, understand, debug, and maintain. Further noting that WebXcript may be plugged into most existing systems, or may even use as a stand-alone productivity tool.

In [6], authors propose a heterogeneous data integration framework which is based on Web service composition (HDIFBWSC). The author also focused on how to utilize Web service composition template to compose Web services dynamically. They also focus on how to search template intelligently to achieve high efficient query results and how to improve the performance of data integration based on granular computing. Their model optimizes Web services composition and improves the efficiency and quality of the heterogeneous by the granular computing. The experiments prove that the framework can increase the extensibility and maintainability of heterogeneous data integration system. They also intend to adopt other Web service composition techniques to the proposed framework. Service composition template is a framework with predefined sequence, structure and attributes, which can directly be applied to business processes. Through composition template, users can compose services conveniently. Services composition template includes the basic information, the business services set, the configuration interface, logical structure and the internal association.

In [7], authors propose an automatic wrapping tool to expose legacy program functionality written in legacy languages such as Pascal into Web services. The proposed approach consists of two stages of analysing and Web service wrapping of legacy code. In the analysis stage, reverse engineering techniques is applied to extract the functions within a given code. The extracted methods are then presented to the user to select a number of them. In the wrapping stage, a class comprising the selected methods is created. This class is then scanned by their Web service wrapping tool, to convert the class methods into Web services.

A WebTransact mediator method [8], integrates Web services through two XML-based languages: Web Service Description Language (WSDL), which is the current standard for describing Web service interfaces and Web Service Transaction Language (WSTL). The transaction support of heterogeneous Web services is defined using WSTL. WSTL is built on top of WSDL extending it with functionalities for enabling the transaction composition of Web services. Through WSDL, a remote service understands how to interact with a Web service. Through WSTL, a remote service knows the specific transaction support of a Web service. Besides the description of the transaction support of Web services, WSTL is also used to specify mediator related tasks such as: the specification of mapping information for resolving representation and content dissimilarities, the definition of mediator service interfaces, and the specification of transactional interaction patterns of Web service compositions.

III. MOTIVATIONAL EXAMPLE

A scenario can be considered in which an organization is using a legacy system which is written in C and Web services written in different platforms like Asp.net in Visual Studio, JSP in Netbeans. The requirement for this organization is to integrate these different services and must adapt the new business rule. ASP.net Web services have different specification to that of Web services developed in Java. So it is not an easy task to merge all of them together. Composing all these heterogeneous services becomes challenge for business organization. Beside that, reuse of existing legacy
code for this organization is also a challenge. So if an organization wants to use all services in a single application, Web service composition will make it easy and there is no need to design new application which works as a legacy system. The organization can proceed with its usual business task from the single application.

IV. APPROACH USED

The proposed approach consists of two steps:

a. Transformation of legacy code to Web services.
b. Composition of heterogeneous services

a. Transformation of legacy code to Web services

gSoap tool is used to convert legacy code to Web services. After defining the method name and its parameter, this is compiled with gSoap tool to generate a CGI file which acts as Web service. This CGI file, on receiving SOAP request, responds with SOAP message. This Web service is hosted on Apache web server as shown in Fig. 1.

IV. APPROACH USED

The proposed approach consists of two steps:

a. Transformation of legacy code to Web services.
b. Composition of heterogeneous services

a. Transformation of legacy code to Web services

b. Composition of heterogeneous Web services.

Typical Web services are implemented in different software IDEs and then the services are hosted on the respective web server.

To consume the hosted Web services, a client application needs to be created for the respective platform. Every Web service methods will be invoked through SOAP Request and Reply messages. Though Web services are loosely coupled, these services are not transparent in context of heterogeneous Web services, so that the message transfer between heterogeneous Web services is not an easy task. In our proposed approach, users as well as organizations use their own existing legacy code and compose them. This provides an environment to invoke all the methods from single interface while not remembering the address of services to execute.

In Fig. 2, one can observe that there are many heterogeneous Web services hosted within an organization. Services are developed on Microsoft, Java and Legacy code on C/C++. A client, which is like a wrapper for every service, can invoke all the services. After getting all those wrapper client, these are merged and represented as a single services known as Controller service. This controller service will be used as composition of heterogeneous services.

Figure 2. Creating a C client to invoke any Web services

V. COMPLETE IMPLEMENTATION

Now, consider a scenario in Fig. 3, where a company is having legacy code, Web service in Asp.net and Java. The following steps are followed to make these heterogeneous applications to function as a complete Web service.

Step 1: The first step is to convert legacy code in C/C++ into Web services and host it on Apache server so that this service can be invoked from anywhere.

Step 2: The second step is to develop and deploy Web services developed in Asp.net and Java as well as Web service in C/C++ as shown in Fig. 3. Microsoft IIS application server is used to host web service developed in Asp.net and Glassfish ESB Server is used to deploy web service developed in Java. Similarly, Apache web server is used to deploy Web services developed in C/C++.

Figure 3. Composition of Heterogeneous Web services
Step 3: The third step is to develop client application for each of the Web services. These client applications are developed in C. gSoap tool is used to extract WSDL from Web services which were hosted on respective server in step 2. gSoap tool is used to get the structure of services such as SOAP Request parameter, SOAP Response parameter, number of inputs require to invoke the services or to get the function prototype. These client applications are developed on the basis of these prototypes.

Step 4: The last step is to develop Controller service that is used to call heterogeneous clients developed in step 3.

VI. RESULTS

The Controller service as shown in Fig. 4, comprises of different Web services. The different Web services can be invoked from this controller service.

![Image](call.png)

Figure 4. Calling any Web services from Controller service

In our implementation, the Web services are developed in different systems and are hosted at different locations. The heterogeneous service composition is dependent on XML messaging. A XML request file which is created from legacy C/C++ code is shown in Fig. 5.

![Image](xml_multiply_SOAP_request_file.png)

Figure 5. XML multiply SOAP request file

The XML file in Fig. 5 shows two input parameters a and b and their values respectively. Now, the legacy application which is hosted on Apache server gives the SOAP reply message. The SOAP reply message is shown in Fig. 6.

![Image](legacy_application_returning_SOAP_reply_message.png)

Figure 6. Legacy Application returning SOAP reply message

VII. CONCLUSIONS

Composition and adaption of heterogeneous Web services is a challenging issue for business organizations. This paper shows the composition of legacy systems with applications developed in Microsoft .Net, Java Netbeans, etc. According to the implementation discussed above, the composition of legacy systems with Web services is really tough job. Our implementation shows the possibility of conversion of legacy system into Web services and easily composes the heterogeneous Web services. Though we have developed and shown the approach with simple application, we claim our approach works for any business application.

REFERENCES

[6] Han Jian-min,Yong Jun,Li Xi-yu “An Adaptive Heterogeneous Database Integration Framework Based on Web Service Composition Techniques”