Enhancing the Interactions in an Engineering Job Outsourcing Environment (B2B System) with Composite Web Services

E Kirubakaran, PhD.
AGM, Outsourcing Department,
BHEL,
Tiruchirappalli, India

D Ravindran
Associate Professor of Computer Science,
St Joseph's College,
Tiruchirappalli, India.

D I George, PhD.
Amalarathinam
Director, MCA,
Jamal Mohamed College,
Tiruchirappalli, India.

ABSTRACT
Interoperability among the various departments of an organization and the ancillary units has been geographically distributed and the coordination has become more complicated. Web-based services with service-oriented paradigm enable flexible and dynamic interoperation of autonomous software units. Service oriented architecture can be realized with web services as the low level building blocks and higher level services can be composed with the Business Process Execution Language. In this paper, effective interaction among the various departments of an organization and the ancillary units is studied and a solution based on composite web services is proposed. Intricacies involved, issues connected with this type of composition will be studied with reference to *engineering job outsourcing environment*. This study provides orchestration mechanism which leads to composition of new higher level abstract services.

General Terms
Composite Web Services

Keywords
Service oriented architecture, web services, orchestration, service composition, BPEL, BPMN, and UML.

1. INTRODUCTION
In a distributed environment, applications can achieve interaction with a help of different techniques like socket programming, Remote Method Invocation, CORBA based interaction, and XML based interaction. They are considered to be interaction techniques. They can also be viewed as techniques to access a service, offered by a remote server. Socket program based interaction is considered to be low level mode of interaction. Remote Method Invocation is a step above the previous technique, as the packet construction, sending, receiving, interpretation etc are not involved with the developer. But, it is a language dependant technique which prevents the linking of legacy systems. Language independency may be achieved with CORBA based interaction. But, that also suffers from the vendor dependency because of its association with Interface Definition Language (IDL). XML based interaction eliminates the problems faced in CORBA as it is considered to be a standardized language for data representation. XML acts as a bridge connecting different applications together to work as a unit. In other words, XML acts as glue that binds different applications together to work as a single unit.

This XML based interaction leads the way to a new computing paradigm called Service Oriented Architecture (SOA). SOA is an architecture in which various services, maintained by different service providers, are linked together to work as a system. These services are available always and can be accessed anytime, anywhere depending on the need. These services are nothing but application components; they are also totally independent to one another and self contained. These characteristics will lead to a situation in which a change in one service will not affect the other services and as a result loose coupling is said to exist among the components or services, involved with the system. This SOA concept and web services may be used effectively to implement the interactions among the different units of any organization with reference to engineering job outsourcing.

There are many departments involved in an outsourcing environment, along with the ancillary units to which the jobs are outsourced. There are so many processes involved and they are all executed in different geographical locations and they operate on different islands of data. Ancillary units are free to use their own storage format and one needs to interact with such a situation. This is possible only if the ancillary units provide methods in the form of a web service. Other departments of the organization also will have their own web services and all of them work in unity to achieve better coordination. This results in reduction of cycle time, as the business partners interact with the help of web services.

Rest of the paper is organized as follows: Section 2 will discuss the related work whereas the Section 3 discusses the modeling of the outsourcing industry. Section 4 discusses the material residue calculation as an example. Section 5 concludes by summarizing the work and opens up the area for further study.

2. RELATED WORK
A distributed system may be designed with client-server architecture and object oriented concepts. The same can be improved if component based development is included in the process. The system designed in such a way will comprise of presentation techniques, interaction techniques, component characteristics, storage aspects etc. This will lead to a situation in which the concept of Service Oriented Architecture is included in the development process to have total independence among the applications, deployed in different application servers. How SOA is applied in a business environment and how it differs from the previous technologies available for interaction, is discussed in [1]. Service oriented computing enables platform, system and language independency. Protocols used and the extensible Markup Language (XML) encoded textual messages pass firewalls, thus enabling service sharing [2]. Communication

between customers and the businesses will be enhanced and customized products and services are provided according to customers' preferences [3]. A system/application designed with SOA, can be realized with the help of web services. Web services are computational entities that are autonomous and heterogeneous [4]. Web service is an application component available in the web and can be accessed anytime, anywhere with the help of Hyper Text Transfer Protocol (HTTP) and XML, where XML is used for encoding and HTTP for the transfer of information. These low level web services are primitive services, which may be collected and organized to work in a particular sequence to represent a high level, abstract business process [5]. Quality of Service (QoS) parameters may be added by the service providers to the services and the reliability of these services are discussed by Paolo Bocciarelli et al [6]. Testing of stateful web services is carried out in [7].

The Web Services standard provides primitives for communication, messaging, and naming of software applications (or services) available over the Internet. Once software applications are available as a service, a composite service can be created by accessing a set of services programmatically using some scripting language or XML based language like Business Process Execution Language (BPEL). A composite service expresses a business process, which is a composition of many primitive services, maintained by different service providers. This paper is concerned with efficient collaboration of such enterprise applications.

Currently, composition of web services is done by *Orchestration*. An orchestration is a workflow that combines invocations of individual operations of the web services involved. It is therefore a composition of individual operations, rather than a composition of entire web services. In this paper, the authors try to propose the composition of abstract services from individual web services in the engineering job outsourcing environment. Service Oriented Computing requires the specification of data exchanges so that the final result can be produced by exploiting the results of the cooperating services. Design and development of model for the human, semi-automatic and automatic activities are discussed in, with an example [8].

Managing business applications that are executed in a distributed environment creates special challenges to an IT organization. The focus on management issues such as availability, performance, quality of service and security for that solution grows rapidly. Managing business process solutions requires knowledge of both business domains and Information Technology domains. The increasing system complexity of Business Process Management (BPM) infrastructure is reaching a level beyond human ability to manage and secure. This increasing complexity points towards the need to automate many of the managerial functions associated with the business system.

Processes being built today need the business agility to adapt to customer needs and market conditions. This would include the interaction among the customers, partners, suppliers, sub-contractors etc. involved in the business scenario. A single standard is desired that can manage both Enterprise Application Interaction (EAI) and Business-to-Business (B2B) interactions involving web services. Web services orchestration is about providing an open, standards-based approach for connecting web services together to create higher-level business processes. Standards such as BPEL4W, WSCI, and BPML are designed to reduce the complexity required to orchestrate web services, thereby reducing time to market, cycle time and costs, and increasing the overall efficiency and accuracy of business processes. As different units are involved, as shown in ***Figure 1***, without a common set of standards, each organization is left to build their own set of business methods. This situation paves the way for the use of web services to implement the business processes, so that the interaction among the web services achieved with XML based protocol called Simple Object Access Protocol (SOAP).

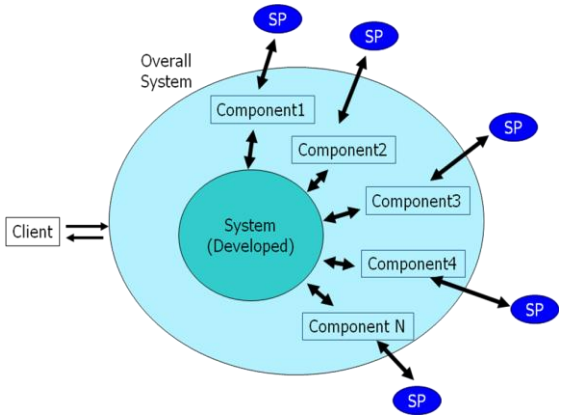

Figure 1: Component based development

In figure 1, each component can be implemented as a web service and they all will be maintained by different service providers, viz the different units involved in the business environment. These services will be available 24/7 basis and any other new higher level abstract service can be constructed by combining the primitive services in a particular sequence, which is referred as *service orchestration.* A web service cluster represents a collection of available web services provided by multiple service providers to perform a specific function [9].

Orchestration describes how web services can interact with each other at the message level, including the business logic and execution order of the interactions. These interactions may span applications and/or organizations, and result in a long lived, transactional, multi-step process model [10][4]. Choreography tracks the sequence of messages that may involve multiple parties and multiple sources, including customers, suppliers, and partners [10][4]. Choreography is typically associated with the public message exchanges that occur between multiple web services, rather than a specific business process that is executed by a single party. Computation and coordination are to be separated in order to increase re-usability, improve comprehension and for easy maintenance and evolution of software [11]. Service coordination does not require any executable logic whereas support for execution of process or flow definition is required for orchestration. Daniel and Pernici [12] discussed the rules governing the data exchanges and relationship between input and output messages. Marcos Lopez-Sanz et al [13] discussed the platform-independent model, an architecture and coordination strategies. Software agent based collaboration is discussed in [14].

WS-BPEL is a standard for orchestration and is used to compose new web services from existing web services. It enables a user to define new service from existing services and present the result as a service. That is why it is often described as a language for recursive composition. BPEL based dynamic collaboration mechanism for distributed

product design is discussed in [15]. Many of the standards that will be discussed in this paper focus on SOA and service orchestration. Here, the term web services orchestration will be used to describe the creation of business processes, either executable or collaborative, that utilize web services.

BPEL is a layer on top of WSDL. The WSDL interface defines the specific operations allowed, and BPEL defines how to sequence them. WSDL describes the public entry and exit points for every BPEL process, and WSDL data types describe the information that passes between process requests. Additionally, WSDL can refer external services that the BPEL process requires.

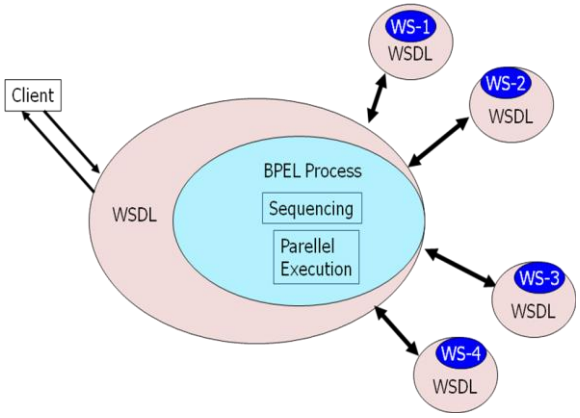

Figure 2: Web Service Orchestrations

This is depicted in Figure 2. BPEL provides support for both executable and abstract business processes. An executable process models the behavior of participants in a specific business interaction, essentially modeling a private workflow. An abstract process or business protocol specifies the public message exchanges between parties. Essentially, executable processes model orchestration while abstract processes model the choreography of services. BPEL4WS is a specification that provides an XML-based grammar for describing the control logic required to coordinate web services participating in a process flow. This grammar can then be interpreted and executed by an orchestration engine, which is controlled by one of the participating parties. The engine coordinates the various activities in the process, and compensates the system when errors occur

Every BPEL process is exposed as a web service using WSDL. The WSDL describes the public entry and exit points for the process. WSDL data types are used within a BPEL process to describe the information that passes between requests. WSDL might be used to refer external services required by the process.

3. ARCHITECTURAL MODELING

Interaction among the services and the service composition will be explored with reference to engineering job outsourcing environment by generating models to cover all aspects of collaborative processes. This environment includes departments like Marketing, Production, Engineering / design, Manufacturing, Quality assurance and Quality control. The model comprises of different views, each showing a particular view of the system that is analyzed. As it is designed with UML, it is a platform independent model. Information, getting exchanged among the units, comprises (not only) of Purchase Order Information, Materials Information, Engineering Information, Testing Information, Inspection Call Booking System, Delivery Challan System, Material Accounting Information, Billing Information, Vendor Corner, Recovery Posting to Vendor, Plan & Commitment Information, RSV Status as shown in figure 3.

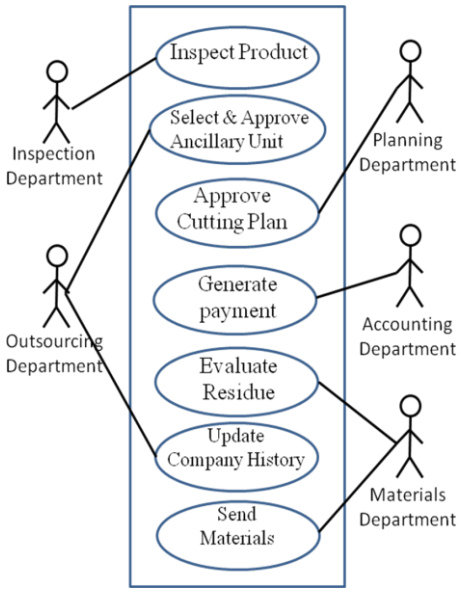

Figure 3: Use case model

Usecase diagram in figure 3, captures the interactions between the exernal entities and the software System.

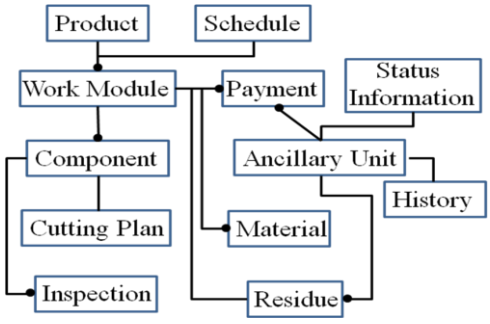

Figure 4: Class Diagram

Class diagram in figure 4 represents the relationships among the entities, sequence diagram in figure 5 depicts the data exchanges for a given sample task and the BPMN diagram in figure 6 explains the activities performed in different entities, deployed within various application servers distributed geographically.

This facilitates the construction of web services. In our context, web services are synonyms of the actvities at different locations. Usecase diagram and class digram are drawn for the entire system whereas the sequence diagram and the activity/BPMN diagram explains the specific task.

The sequence diagram shows the data exchanges for the task of calculating the materials residue, executed immediately after completion report by the outsourced company. Based on the outsourcing department, this task interacts with materials department, Planning, outsourced company and accounts department.

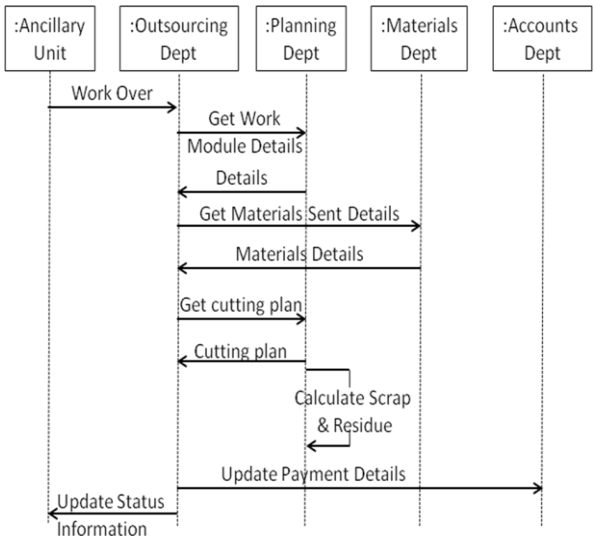

Figure 5: Sequence of message exchanges

The BPMN diagram in figure 6 explains the same task with respect to the activities performed in each department to achieve the goal of finding out the material residue, which may be used to evaluate the payment.

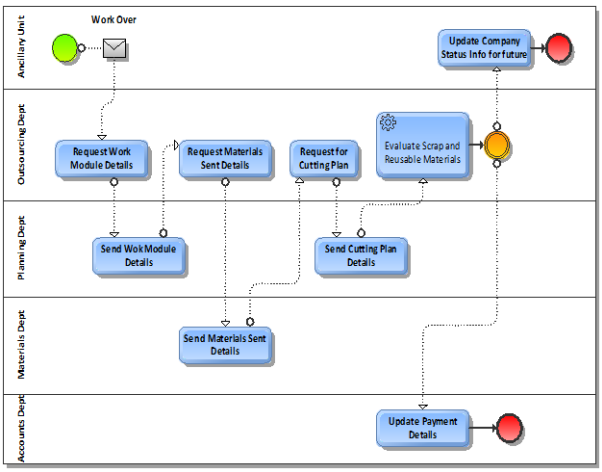

Figure 6: BPMN

Various activities are available as web services i.e, leaf node services. Using these services, higher level service like Material Residue Calculation is constructed as a composite web service. Composite service can access many low level web services with the corresponding WSDL files and it has its own WSDL file for making it accessible by other services.

4. IMPLEMENTATION

After selecting the ancillary unit for the job to be outsourced, contract to be established, materials sent, cutting plan to be approved and inspections are to be carried out regularly. After completing the job, the external outsourced unit informs the parent company of the completion. Once if the completion message is received, the parent company's outsourcing department initiates the material residue calculation, which may be used for payment. For this residue calculation, a composite web service is invoked which in turn invokes the following web services, viz

t1 = get_materials_sent(job_no, au_id)

p1 =get_wastage_percentage_allowed(job_no)

t2 = get_materials_leftover(job_no)

t3 = get_scrap(job_no, au_id)

t4 = get_product_weight(job_no)

After invoking these web services and getting the materials in tonnes namely t1 to t4s and the wastage-percentage allowed, the composite web service evaluates the wasted material as

Wastage = t1 – t2 + t3 + t4 + t1 * p1/100

Based on the wastage, the payment is decided and all these values are updated in the corresponding locations with the help of web services. This composite service can also report errors if the inspection report is not conforming to the standard set by the parent company.

Composite web service is invoked with BPEL, a non-proprietary, XML-based language to describe how Web Services interact with one another to form workflow systems that implement business processes, and the screen shot is shown in figure7

5. CONCLUSION AND FUTURE SCOPE

The System envisaged, a loosely coupled system, has many primitive web services and composite web services. Composite services are constructed with the help of BPEL. All of them together improve the interaction among different units and also helps to bring down the cycle time. It also helps the units to use their own storage structure and different executing environments. After the deployment of services, the departments can change their storage format and internal logic of the service without affecting the existing system. Users of the service need not worry about the implementation details, for accessing the service. The SOA infrastructure will provide standardized access mechanisms with service-level agreements.

The factors that are to be considered to improve the performance of this system are (i) availability: Services should be always available. There is every possibility that a web service, a machine or the network may become non-functional. (ii) Performance: QoS factors such as response time, throughput and security aspects are to be taken care and (iii) cost: Cost of the Web service is often related to its quality.

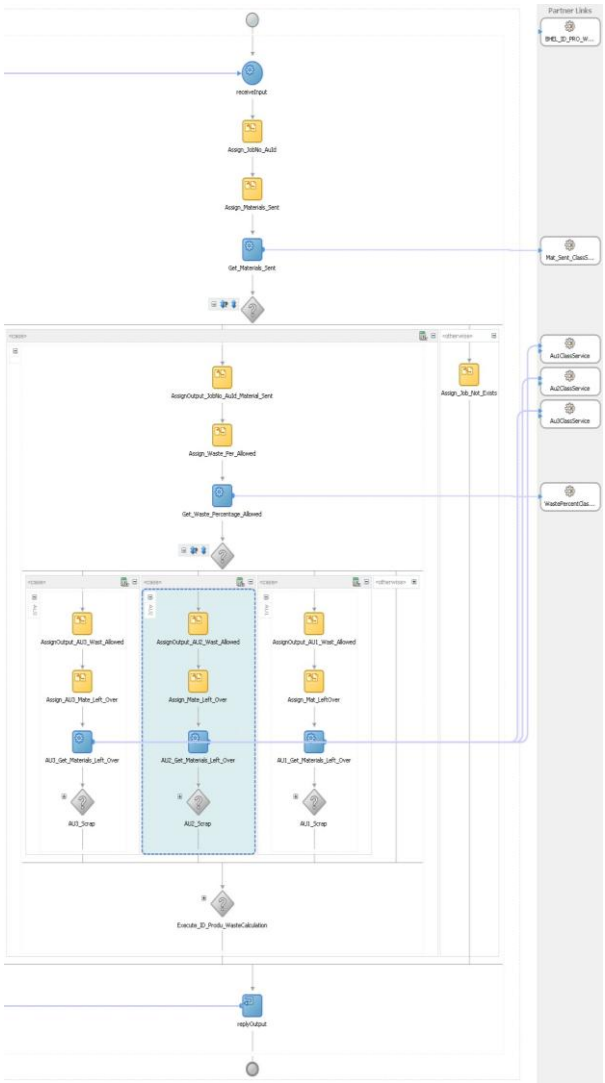

Figure 7: Screenshot of invoking Web Services

Speed, reliability and security features increases cost, but there could also be penalties associated with not meeting certain QoS goals or service-level agreements (SLAs). Excluding cost factors, candidate services, in the local server and also in hired servers, may be incorporated to make it available. This will increase the service selection problems among the candidate services. Mobile interaction may be another feature that will definitely achieve the objectives.

6. REFERENCES

[1] Jan Lohe and Christine Legner, 2010, SOA adoption in business networks: do service-oriented architectures really advance inter-organizational integration?, Electron Markets

[2] Bernd J. Kramer, 2008, Component meets Service: What does the mongrel look like? - Innovations in Systems and Software Engineering

[3] Reiner Alt, Witold Abramowicz and Haluk Demirkan, Service-orientation in electronic markets, Electron Markets.

[4] Nadia Busi, Roberto Gorrieri, Claudio Guidi, Roberto Lucchi and Gianluigi zavattaro, 2005, Towards a formal framework for choreography, In Proc. of International Workshop on Distributed and Mobile Collaboration (DMC 2005), IEEE Computer Society Press. WETICE

[5] Michele Barletta, Silvio Ranise and Luca Vigano, 2009, A declarative two-level framework to specify and verify workflow and authorization policies in service-oriented architectures, International Conference on Computational Science and Engineering

[6] Paolo Bocciarelli and Andrea D' Ambrogio, 2010, A model-driven method for describing and predicting a reliability of Composite Service, Software and Systems Modeling

[7] Fevzi Belli and Michael Linschulte, 2008, Event-driven modeling and testing of real-time web services, 32nd Annual IEEE International Computer Software and Applications Conference

[8] Boukhedouma Saida and Alimazhighi Zaia, 2010, A process meta-model based approach for development of collaborative applications bilt on workflow and SOA, European and Mediterranean Conference on Information Systems

[9] Liang-Jie Zhang and Bing Li, 2004, Requirements driven dynamic services composition for web services and grid solutions, Journal of Grid Computing

[10] Chris Peltz, web services orchestration and choreography, Hewlett-Packard Company

[11] Corchuelo, J.A. Perez and A Ruiz, Multi-party Coordination in the context of MOWS, Programming and computer software

[12] Florian Daniel and Barbara Pernici, 2006, Insights into web service Orchestration and Choreography, International Journal of E-Business Research, 2(1), 58-77

[13] Marcos Lopez-sanz, Carloas E. Cuesta, Esperanza Marcos and Jesus Dominguez, Developing Coordination Strategies using a Service-Oriented Model-driven approach, International Journal of Web Services Practices

[14] [Mohammed Zahiri, Mohammed R. khayyambashi, An agent-oriented executive model for service choreography, Journal of Theoritical and applied information Technology

[15] Yi hu, Xionghui and Congxing Li, A BPEL-based Service-oriented dynamic collaborative mechanism for distributed product design, The International Journal of Advanced Manufacturing Technology.