IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP5093

# Applications Development with Web Services

Vincent C. Yen
Department of Information Systems and Operations Management, Wright State University, Dayton, OH 45435

## ABSTRACT

*"Web Services" draws a lot of attention today. At least a dozen leading IT companies are actively engaged in this field along with standard setting organizations. Significant progresses, for example in Web service architecture, have been made over the past few years. Proprietary development tools are now available and have successfully demonstrated. In this paper, we discuss differences, from the systems analyst view, the development life cycle for applications development with Web services. We briefly outline our paper as follows: 1) introduction to Web services, 2) review of SDLC, 3) the generic SDLC with Web services, 4) the analysis activities, and 5) implications to systems analysts.*

## INTRODUCTION

Software integration and interoperability is a long standing issue in software industry. Since the basic Internet infrastructure is now well-established this paves the way for the fulfillment of the desire for total distributed computing. Among many important initiatives, systems integration and interoperability has been a major focus. Companies and end-users alike do not want to reinvent the wheels if needed "components" are available and at low cost. However, it has proven to be difficult of making integration and interoperability to work easily and economically. Past attempts to make heterogeneous systems interoperable are many, notably CORBA, DCOM, and Java/RMI initiatives. CORBA (*Common Object Request Broker Architecture*) is a specification defined by the Object Management Group, DCOM is Microsoft's distributed common object model, and Java/RMI is the remote method invocation mechanism. However, the success of these standards is marginal (Chung, Lin, & Mathieu, 2003).

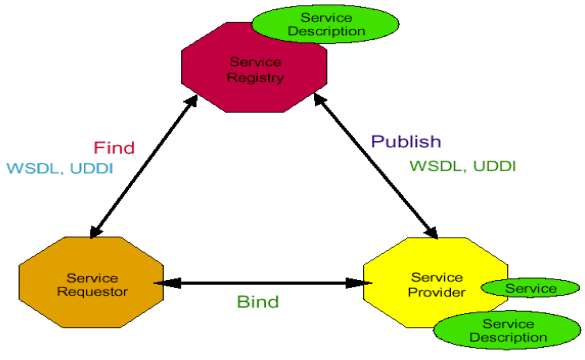
A new approach to solve the interoperability problem is XML web services, or simply web services. This approach uses Web standards of HTTP, URLs and XML as the lingua franca for information and data encoding for platform independence; therefore it is far more flexible and adaptable than earlier approaches.

## WHAT ARE "WEB SERVICES"?

The phrase "Web Services" has been defined in many different ways (Castro-Leon, 2002, Ambrosio, 2002). In the working draft of Web Services Architecture (W3C, 2003) it is defined as

*"A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."*

*Figure 1. A Simplified Web Services Architecture (W3C, 2003)*




A simplified Web service archetecture is conceptually depicted in figure 1.

Main features of web services are that services (Burner, 2003):

1. Expose programmable application logic.
2. Are accessed using standard Internet protocol
3. Communicate by passing messages.
4. Package messages according the SOAP specification.
5. Describe themselves using WSDL.
6. Support the discovery of Web services with UDDI.
7. Are loosely coupled.

## WEB SERVICES TECHNOLOGIES

Three XML-base protocols, one for communication, one for service description, and one for service discovery have become de facto standards. They are:

- SOAP (the simple Object Access Protocol) provides a message format for communication among Web services;
- WSDL (the Web Services Description Language) describes how to access Web services;
- UDDI (the Universal Description, Discovery, and Integration) provides a registry of Web services descriptions.

Another area of importance in Web services is the capability of constructing new composite Web services from existing Web services. Many standards in this area are being developed (Van der Aalst, 2003), for example, Business Process Execution Language for Web Services (BPEL4WS) by IBM and Microsoft (Fischer, 2002). The Web Services Choreography Interfaces (WSCI) is a specification from Sun, SAP, BEA, and Intalio, for Web services collaboration (Arkin, Assaf, et. al. 2002). It is not clear if there will be a common standard in the near future. However, regardless of the differences among vendor groups the composition of Web services uses the concept of business processes and workflow management (Allen, 2000).

## ASSUMPTIONS

Application developers traditionally relies the system development life cycle (SDLC) methodology for systems development. Basically, SDLC consists of planning, analysis, design, and implementation stages. Depending upon the software development environment, the development process utilizes varying degrees of 'program reuse'. In component based system a larger chunk of code called a component may be reused in different software. The interoperability of components is a serious problem even with the help of CORBA and DCOM.

Web services are units of low coupling and low dependency "components" on the Internet. To simplify our discussion, we assume that the Web services infrastructure is fully functional. So an important goal of application development is to take advantage of the available services on the Internet and use them in the new application. The new application under development involves, among other activities, analysis and design of business applications. In this paper we limit our

discussion to "differences between traditional SDLC and development with Web services". Specifically we devote our discussion to the "analysis" activities.

## APPLICATION DEVELOPMENT WITH WEB SERVICES (ADWS)

The basic framework of the SDLC of the application development with Web services is the same as in traditional systems development. However, detail activities of each stage in SDLC may not be the same due to the differences in infrastructure and development environment. We briefly highlight the tasks in each stage of SDLC – planning, analysis, design, and implementation - for the application development with Web services. In the planning stage, the tasks involve: the identification of the business problem or a business process that should be automated and the feasibility analysis. In the analysis stage, the tasks involve: identify business sub-processes, and the workflow analysis of the business process. The result of the analysis answers the question of what the application must do. In the design stage the tasks involve: identifying and structuring the parts of the application and making them fit together in an efficient and quality way with respect to the hardware, software and the operating environment. In the implementation stage the tasks involve: coding, testing, conversion. Coding requires knowledge skills of XML and protocol standards discussed earlier. This paper will only discuss tasks relevant to the analysis stage.

## REQUIREMENTS ANALYSIS IN ADWS

The input of the analysis stage is the business problem or the business process that should be automated. Let us use "borrowing a book from a library" as an example of a business process. A dissection of the process gives rise to a series of activities: 'check user identification', 'check out books', and 'generate a receipt'. In the Web services environment each activity of a business process is a Web service. Since BPEL4WS supports execution of composite Web services structured in the form of sequential, parallel, and conditional process flows of activities and handling data exchange between activities, a systems analyst needs only to specify the structured (as in structured programming) nature of activity flows. Suppose one would like to develop an application for "borrowing a book from a library", we first draw a "structured" workflow diagram based on the above activities of the business process. If any activity in the workflow requires interaction with external Web services, the diagram should show the interactions (i.e., messages).

In general, systems development with Web services requires the composition of Web services and that in turn needs modeling of a Web service from one party perspective and the interaction among each involved party (Peltz, 2003). This is accomplished by using business process and workflow modeling in two aspects: 1) specifying internal details of the composition to offer an executable business process, and 2) planning the message sequences between parties and sources. The former is called *orchestration* and the latter is called *choreography*. In other words, a systems analyst will be quite suitable to develop orchestration and choreography diagrams in the analysis stage. However, there exist variations of the above procedure due to multiple Web services standards for business process modeling and execution (Van der Aalst, 2003) other than the BPEL4WS.

## IMPLICATIONS TO SYSTEM ANALYSTS

Since Web services are at a higher level of abstraction and each Web service often represent a concrete business activity within a business process, analyzing business process at this concrete business level makes it intuitive to systems analysts than working with those concepts such as subroutines, procedures, and even components. Similar simplicity applies to the drawing of a workflow analysis diagram with these activities. However, each activity of the business process must be made as a Web service before it could be applied. New Web services are built upon the composition of existing Web services according to a Web services composition protocol. Systems analysts should be familiar with these protocols, allowable operations between Web services in particular. Their availability greatly simplifies system analysts' analysis, and design tasks. As to the coding, a good knowledge of XML, SOAP, WSDL, UDDI, and a composition language like BPEL4WS is necessary.

The composition of Web services requires system analysts to change modeling concept from fine grained to very coarse grained. They work with business activities (Web services) and structure them into a workflow diagram by using sequence, iteration, selection, and parallel operations. In addition, analysts should determine data elements exchanged between activities, and communication messages with external Web services. When the analysis completes the diagramming analysis it becomes straight forward to convert the workflow diagram to an execution language such as BPEL4WS for testing.

## CONCLUSION

Web Services is an important technology for system integration and interoperability. Although a very young field it has made tremendous progress in the last few years. Software vendors like Microsoft's .NET and Sun's J2EE have tools for Web services development. Many companies have been experimenting with the technology in-house hoping they could one day capitalize it. As in any systems development projects, developers must follow SDLC process in creating Web services. A critical step is systems analysis where the result will be used later for transforming into executable code. Systems analysts are good fit of doing the analysis because the work will be based more on the knowledge of business processes and activities of the processes. In order to become a Web services analyst, systems analyst shall receive minimal training in workflow modeling and the operating context of the composition of Web services. This paper limited its discussion on the analysis activities of SDLC with Web services. We find there are significant differences in approaches to analysis. Since the technology is increasingly becoming practical, it is worth investigating the differences in other stages of SDLC so as to result in a more complete comparison in systems development process.

## REFERENCES

Allen, R. (2000), Workflow: An Introduction, in The Workflow Handbook 2001, edited by Fisher, L., Workflow Management Coalition.

Ambrosio, J. (2002). Web Services: Report from the Field. Application Development Trends, Vol. 9, No. 6, June, 2002.

Arklin, Assaf, et. Al., (2002). Web Service Choreography Interface 1.0.

www.sun.com/software/xml/developers/wsci/wsci-spec-10.pdf.

Casati, F., Ceri, S., Pernici, B., and Pozzi, G., (1997), Advances in Object-Oriented Data Modeling, edited by Papaxoglou, M., Spaccapietra, S., and Tari, Z., The MIT Press.

Chung, J., Lin, K., & Mathieu, R. (2003). Web Services Computing: Advancing Software Interoperability. Computer, Vol. 36, No. 10..

Ferris, C. and Farrell, J. (2003). What Are Web Services? Communications of the ACM, Vol. 46, No. 6, p.31.

Marin, M., Norin, R., and Shapiro, R. Edited. (2002); *Workflow Process Definition Interface — XML Process Definition Language*. Document Number: WFMC-TC-1025. Document Status: 1.0 Final Draft.

Fischer, L. (2002). The WfMC Heralds BPEL4WS Standards for Business Process Management Industry. URL: http://xml.coverpages.org/WfMC-Heralds-BPEL4WS.html.

Ganesarajah, D. and Lupu, E. (2002). Workflow-Based Composition of Web-Services: A Business Model or A Programming Paradigm?. Proceedings of the 6th International Enterprise Distributed Object Computing Conference, Lausanne, Switzerland.

Kleijnen, S. and Raju, S. (2003). An Open Web Services Architecture. ACM Queue, 1/1.

Peltz, C. (2003). Web Services Orchestration and Choreography. IEEE Computer, 16/10.

Van der Aalst, W.M.P. (2003). Don't go with the flow: Web services composition standards exposed. IEEE Intelligent Systems. Jan/Feb.

W3C (2003), Web Services Architecture, W3C Working Draft 8 August. http://www.w3.org/TR/2003/WD-ws-arch-20030808/