AMCIS 2002 Proceedings

Americas Conference on Information Systems (AMCIS)

December 2002

GOAL-ORIENTED ANALYSIS FOR WEB SERVICES APPLICATION DEVELOPMENT

Sooyong Park
Sogang University

Vijayan Sugumaran
Oakland University

GOAL-ORIENTED ANALYSIS FOR WEB SERVICES APPLICATION DEVELOPMENT

Sooyong Park
Computer Science Department
Sogang University
sypark@ccs.sogang.ac.kr

Vijayan Sugumaran
School of Business Administration
Oakland University
sugumara@oakland.edu

Abstract

The World Wide Web not only provides users with an enormous amount of information, but also a wide variety of services. The web is beginning to transform the way applications are being developed and provides the infrastructure needed for quickly assembling applications in real time by integrating available web services. Software industry analysts predict that the Web Services technology will cause a paradigm shift that will forever change the computing environment. The standardization group (IBM, Sun, Microsoft, HP and Oracle) for Web Services has provided the initial standards for enabling technologies that facilitate web services. In this paper, we provide some background on web services, followed by the current technologies utilized in supporting Web Services. We also present an approach for developing applications using Goal-Oriented analysis.

Introduction

The domain of Web Services is receiving a lot of attention these days (Cuberbera 2001), and it primarily focuses on developing web-based applications using distributed component technology. Using Web Services technology, developers can design components independently and share these components with others for developing new services (Tosic et al 2001). The difference between Web Services and component technology is that Web Service relies on the architecture of the web itself, whereas, a distributed component application doesn't necessarily use the web.

Until now, components have been heterogeneous, with varying functionalities and formats. These components are executed within the context of a particular component model. Component models like EJB (Enterprise Java Beans) of Sun, COM+ of Microsoft, and CCM (CORBA Component Model) of OMG etc, have their own architecture, predefined defined set of components, implementation language, naming conventions, object messaging protocols, etc. Because of these differences, it is difficult to use components from different models in one application. To solve this problem, developers have to find their individual solution in terms of translators and wrappers.

Recently, leading companies within the software industry such as IBM, MS, SUN, HP, and Oracle have made a big push for developing Web Services. One of the primary goals of Web Service technology is the development of standard protocols that would allow a heterogeneous set of components to interoperate. These protocols are used for interface definition, method invocation, component repository and implementation of the overall application. These protocols also improve compatibility between components. The principles of component based development are expanded over the Internet. In this environment, developers can share their own components with other departments (Bellifemine, 1999), companies or other developers on the web and develop new applications without the limitation of the underlying component model, language and the distributed object model.

The advancement of Web Service technology will facilitate the improvement in the productivity of application development. Because of the fact that the Web Service environment supports interoperability of all the components, it will promote integration between businesses (B2B). Also, Web Services components technology could start a new market, which dynamically supports renting and trading of Web Services.

The remainder of this research-in-progress paper is organized as follows. The next section discusses the current Web Services technology and recent standards that have been developed. Following that, we present the research on goal-oriented analysis and agent technology and describe how they can be applied to Web Services domain. The last section outlines our future work and provides some conclusions.

Web Services Technology

Web Services technology and the underlying standards are being rapidly developed by organizations that are part of the World Wide Web consortium (W3C). The recent introduction of ebXML for Web Service has generated lot of interest in the B2B market. In this section, we explain the Web Services architecture published in the W3C standard. First, we show the operation model and then each element within this model is discussed in a little bit more detail

In the Web Services Model proposed by W3C, to provide services, the interface of the service should be described with a standard language. The contents of services should be registered with a registry, shared on the Web. A service repository is a distributed directory of services and has functions for searching and registering.

Table 1. Web Services Protocols

Protocol	Function
SOAP	Remote Procedure Call
WSDL	Interface Definition Language of Web Services
UDDI	Registry of Web Services

In the service registry, information about service's location and usage are included. A particular user could use one or more of the services at run time to accomplish a particular task. Table 1 shows the Standards needed to perform the integration function.

SOAP (Simple Object Access Protocol) (Box et al. 2000), one of XML based RPC, is used for calling Web Services. SOAP utilizes XML and defines message type needed in Remote Procedure call and response. A SOAP message is sent-through HTTP, which is the basic protocol of the Web. Because SOAP uses the default port (port 80) on the web, it could access all the servers that support Web Services.

WSDL (Web Service Description Language) (Christensen et al. 2000) is a language used to describe Web Service Component Interface. Using WSDL, a component developer can provide his or her own components to the general users. The contents of abstract interfaces, protocol bindings, deployment etc, could be described as Web Services component interface. WSDL also uses XML language. It is defined as extensions of XML tags, similar to SOAP.

UDDI (Universal Description Discovery, and Integration) (UDDI Project 2001) is a Web Services API for publishing and discovering Web services. A service provider opens the service to the public so that users can access the service. Service information is included in UDDI registry. Information about available services and how to use these services is published. In UDDI specification, there are functions for publishing and discovering services using an API. This API is defined as a set of SOAP messages.

Problems in Application Development

Even though Web Services technology can extend application development over the Internet, current development technologies do not take advantage of this standardized environment. How to develop software applications using the available Web Services is rarely discussed. One of the research problems in this area is how to organize various Web Services to satisfy user's requirement.

In simple cases, user's requirement could be satisfied by connecting one or two Web Services. However, in reality, various Web Services should be organized, in other words, assembled, in different ways to meet all the requirements of the user. This paper proposes goal analysis based approach for identifying appropriate web services to meet user's requirements.

Goal-Oriented Analysis

To develop the Web Services based systems that users desire, first the necessary web services have to be identified and then, determine how they can be combined. The users' needs are treated as the system goals. In order to achieve these goals, an agent-oriented system can be developed (Woolridge et al. 1999, Jennings et al. 1998, Sycara et al. 1996). Therefore, the Web Services application system can be designed through goal-directed analysis and agent-oriented systems development method.

A goal is defined as "a non-operational objective to be achieved by the composite system" (Rolland et al. 1998). From the definition, we could also think of goal-directed analysis as a process that successively decomposes higher-level goals into sub-goals and defines the relationships between them. This process continues until there are no more sub-goals to be further decomposed.

Figure 1 illustrates a system root goal, which the Web Services component system must achieve ultimately. It may require many sub-goals, which are generated from the system root goal. These goals could be categorized into three types: system external goals which are viewed from outside; user goals which are the lowest level of goals to be perceived by users; and system internal goals viewed from inside. The goals can also be classified by their properties: Achieve and Maintain (Park et al. 2000).

After the goal analysis stage, we set up the relationships between the Web Services and the goals as a one-to-one correspondence. The proposed goal analysis process is presented in Figure 2.

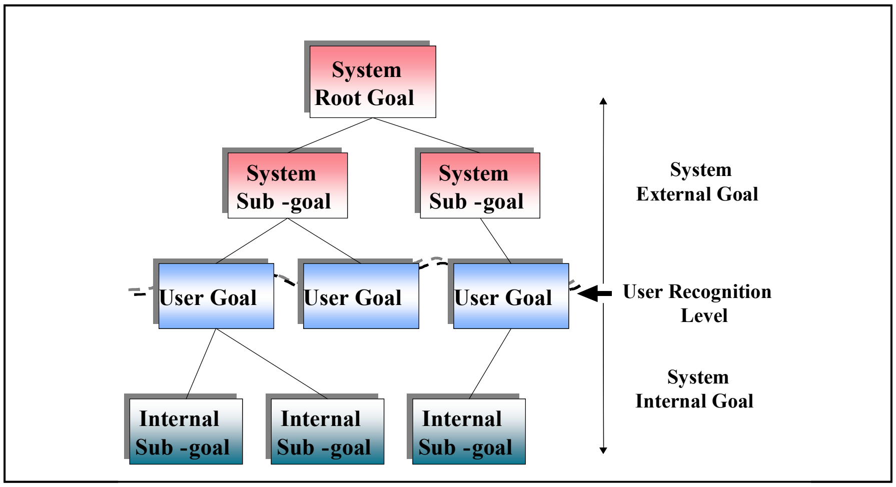

Figure 1. Goal Hierarchy

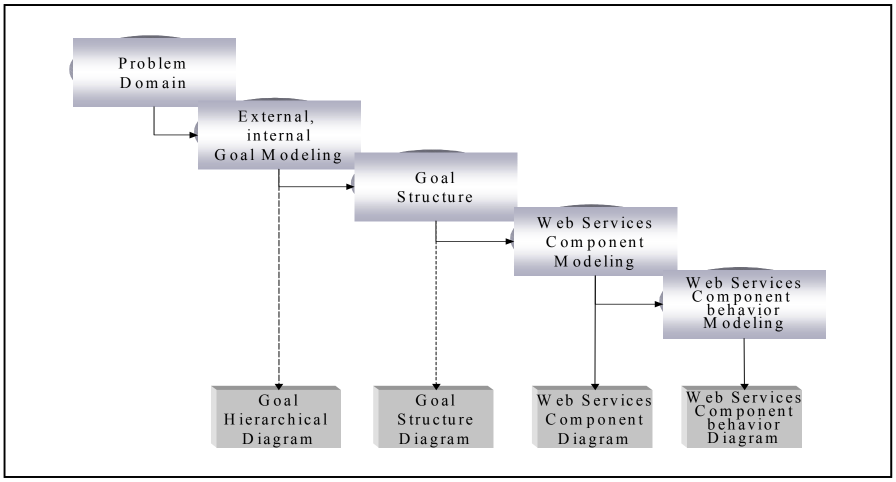
Problem Domain
External, internal Goal Modeling
Goal Structure
Web Services Component Modeling
Web Services Component behavior Modeling
Goal Hierarchical Diagram
Goal Structure Diagram
Web Services Component Diagram
Web Services Component behavior Diagram

Figure 2. Goal Oriented Analysis

External Goal Modeling

The ultimate goal of the Web Services component system is located at the root of the goal hierarchy. Depending upon the various types of goals to be accomplished (users, external systems, etc.), the system goal is divided into many sub-goals with the purpose of accomplishing the system root goal. The sub-division continues until there are user goals; a point where the external view cannot perceive a sub-goal further by sub-dividing.

Internal Goal Modeling

The user goal is the most detailed (lowest granularity) goal that cannot be further divided into sub-goals from the external view. However, the low-level user goals are mapped to internal system goals, which can be then further subdivided. These internal sub-goals, when aggregated, will eventually accomplish the user goals.

Goal Structure

In this research, the Web Services component system is composed of a set of goals and each goal could be expressed as a hierarchy. This hierarchy contains the goal at the root and is composed of constraints and plans.

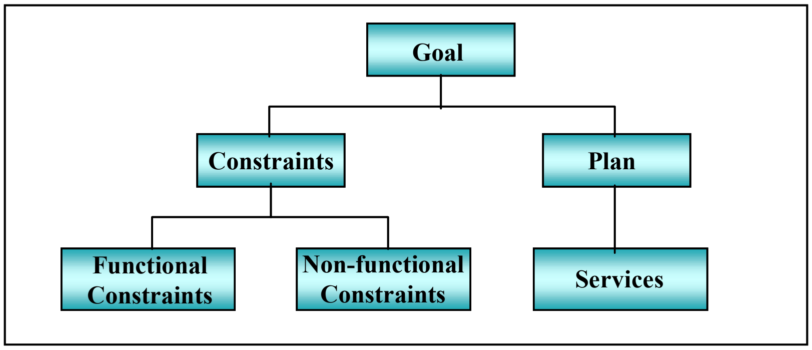

Figure 3. The Goal Structure Diagram

Figure 3 shows the relationship among Goal, Constraints, and Plan. From the internal goal hierarchy diagram, we create the goal structure diagram for every leaf goal node. We can represent each requirement to be a goal that is composed of constraints and plans using UDDI (Universal Description Discovery, and Integration) and DAML (Darpa Agent Markup Language).

Web Services Component Modeling

The Web Services component modeling shows the characteristics of each Web Services component and the relationships between these characteristics. This component model can be represented using UML notations. The Web Services component behavior modeling shows how the Web Services components behave (Martin et al. 1999). The overall behavior of the Web Service application system is also modeled. This model is represented using SOAP, UDDI, WSDL and DAML.

Summary and Future Work

Web Services is a web based distributed component technology, using which developers can create independent components, share and use these components to develop new services. This technology is different from the current development environments in that Web Services use the Web itself as the development environment, whereas component based software to development and execution takes place in different environments.

Even though Web Services technology can extend application development over the Internet, the current focus is on technology standardization. Methodologies for developing Web Services application software are still evolving, and one of the research problems is how to organize various Web Services to satisfy user's requirements.

Goal-oriented analysis helps us to structure Web Services component system. A goal is a non-operational objective to be achieved by the composite system. A goal is composed of constraints and plans that are represented using UDDI, and DAML. This paper has discussed Web Services component modeling and Web Services component behavior modeling that are helpful in developing Web Service application systems.

Our future work will include refining the development method by incorporating goal-oriented approaches in the context of agent based Web Services. We will also formalize the development of the goal-oriented approach for Web Services using SOAP, UDDI, WSDL and DAML (Darpa Agent Markup Language).

References

Box, E., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H.F., Thatte, S., and Winer, D. "Simple Object Access Protocol (SOAP) 1.1," World Wide Web Consortium, May 2000.

Christensen, E., Curbera, F., Meredith, G., and Weerawarana, W. "Web Services Description Language (WSDL) 1.0," World Wide Web Consortium, September 2000.

UDDI Project, "UDDI Executive White Paper," November 14, 2001. Available at http://www.uddi.org

Rolland, C., Souveyet, C., and Ben Achour, C. "Guiding Goal Modeling Using Scenarios", IEEE Transactions on Software Engineering, 24(12), December 1998.

Park, K., Kim, J., and Park, S. "Goal Based Agent-Oriented Software Modeling," Seventh Asia pacific Software Engineering Conference (APSEC 2000), Pages 320-324, December 2000.

Cuberbera, F., Mukhi, N., and Weerawarana, W. "On the Emergence of a Web Services Component Model," Proc. of the WCOP 2001 Workshop at ECOOP 2001, Buapest, Hungary, June 2001.

Tosic, V., Pagurek, B., Esfandiari, B., and Patel, K. "On the Management of Compositions of Web Services," position paper at the Workshop on Object-Oriented Web Services at OOPSLA 2001, Tampa USA, October 2001.

Woolridge, M., Jennings, N., Kinny, D. "A Methodology for Agent-Oriented Analysis and Design," Proc. Of Autonomous Agents 99 Conf., Seattle, WA, 1999, pp. 69-76.

Martin, D. L., Cheyer, A. J., and Moran, D. B. "The open agent architecture: A framework for building distributed software systems," Applied Artificial Intelligence, January-March 1999, Vol. 13, pp. 91-128.

Sycara, K., Decker, K., Pannu, A., Williamson, M., and Zeng. D. "Distributed intelligent agents," IEEE Expert - Intelligent Systems and Their Applications, 1996, 11(6), pp. 36-45.

Bellifemine, F., Poggi, A., and Rimassa, G. "JADE - A FIPA-compliant agent framework," *Proceedings of PAAM'99*, London, April 1999, pages 97-108.

Jennings, N. R., Sycara, K. P., and Wooldridge, M. "A Roadmap of Agent Research and Development," Journal of Autonomous Agents and Multi-Agent Systems. 1(1), pp 7-36. July 1998.