Category: Concepts, Trends, and Challenges

Service-Oriented Architecture

Kwan-Ming Wan
International Business Machine, UK

Pouwan Lei
University of Bradford, UK

Chris Chatwin
University of Sussex, UK

Rupert Young
University of Sussex, UK

INTRODUCTION

The established global business environment is under intense pressure from Asian countries such as Korea, China, and India. This forces businesses to concentrate on their core competencies and adopt leaner management structures. The coordination of activities both within companies and with suppliers and customers has become a crucial competitive advantage. At the same time, the Internet has transformed the way in which businesses run. As the Internet becomes a cheap and effective communication channel, businesses are quick to adopt the Web for integrating their systems together and linking them with their suppliers and customers. Current enterprise computing using J2EE (Java 2 Platform, Enterprise Edition) has yielded systems in which the coupling between various components in them are too tight to be effective for ubiquitous B2B (business-to-business) and B2C (business-to-consumer) e-business over the Internet. This approach requires too much agreement and shared context between business systems from different organizations. There is a need to move away from tightly coupled, monolithic systems and toward systems of loosely coupled, dynamically bound components. The emerging technology, Web services, provides the tools to accomplish this integration, but this approach presents many new challenges and problems that must be overcome. In this article, we will discuss the current approaches in enterprise application integration (EAI) and the limitations. There is also a need for service-oriented applications, that is, Web services. Finally, the challenges in implementing Web services are outlined.

BACKGROUND

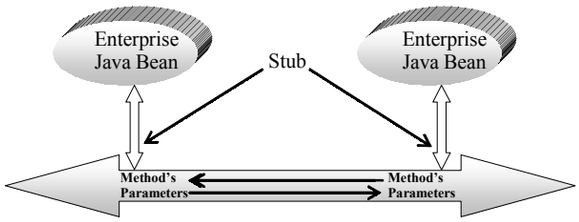
The advancement of computing in the past decade results in enterprise computing being distributed in a heterogeneous environment: computers ranging from mainframes to PCs (personal computers; including top-of-the-line 64-bit processors and outdated 386s) and running two or three different operating systems, database software from different vendors, and a variety of servers. On the other hand, the Internet changes the way that businesses run. The converging mobile and wireless technologies enable mobile users to access the Internet anywhere with broadband. The Internet becomes an indispensable and cheap communication medium and the backbone for enterprise application integration.

Developers have to put a lot of effort and resources in integrating new applications with existing ones, as well as getting existing applications to communicate with one another while keeping the number of changes to these applications to a minimum. Typically, this kind of integration is called EAI. The Java 2 Platform, Enterprise Edition and Microsoft .NET both address the need for making existing applications and business processes available on the Web in a robust, secure, and distributed transactional way. J2EE is platform independent while .NET is limited to Windows-based platforms. In addition, J2EE provides a connector, Java 2 Connector, to integrate with legacy systems such as IBM CICS (Customer Information Control System) for mission-critical applications. However, the choice really depends on the existing infrastructure of a company: Windows-based or multiple platforms. Here, J2EE is used to illustrate the operations and the limitations.

J2EE has a number of applications models (Graham et al., 2005).

- Thin client-browser-based applications use servlets and Java server pages in the Internet environment.
- Thick managed application clients use RMI-IIOP (remote method invocation run over Internet inter-orb protocol) to communicate with server-based

Figure 1. Java RMI architecture



EJBs (Enterprise Java Beans) components. This is applied in the intranet, which provides the extra infrastructure that allows Java programs to directly access EJBs within the intranet domain.

- Messaging applications use the Java message services to act on messages in queues or from subscriptions.

The first two models require a tight coupling between applications. Coupling is the strength of interconnection between two software modules: The higher the strength of interconnection, the higher the coupling (Wijegunaratne & Fernandez, 1998). Loosely coupled systems make it easier for the developer to modify as he or she can maintain one module without having to know very much about any other module in the system. If two modules are highly coupled, then there is a high probability that a programmer trying to modify one of them will have to change the other. The tight coupling in J2EE is due to the stubs. Figure 1 shows the Java RMI middleware. A Java application is composed of a number of components, called Java Beans, distributed across a network. The stub is used to connect to the middleware and transfer activation parameters from the component to the network. In such a way, components can be exchanged for other components, and this allows code mobility. However, any change in the application will be done in the stub in each component due to the tight coupling.

The third model is based on messaging called message-oriented middleware (MOM) and is asynchronous by nature (Serian, 2002). MOM sends messages from one application to another using a queue as a temporary storage area. The queues reside in the middleware layer and do not require a direct connection to the retrieving applications. Furthermore, applications can retrieve the message from the message queue at anytime and in any order. Figure 2 shows message-oriented middleware's architecture. The implementation involves the communication code only and reduces the involvement with the *complexity* of the distributed mechanism. Since the application code is not involved, each application code is independent of each other; that is, there is a decoupling. Conclusively, J2EE is not applicable for dynamic e-business environments where services are called on demand.

THE NEED FOR A SERVICE-ORIENTED ARCHITECTURE

Businesses are under intense pressure. First, customers are like butterflies. As they are well informed in the digital era, they are smart in shopping around and their preferences are changing. Customers are no longer loyal to the same business. Second, the emerging industrial nations in Asia such as Korea, China, and India are posing a threat.

Figure 2. The message-oriented middleware's architecture