Research on the Model of Enterprise Application Integration with Web Services

XIN JIN
School of Information, Central University of Finance& Economics,
Beijing, 100081
China

Abstract: - In order to improve business by automating business processes that span these isolated systems, the enterprise needs to integrate the isolated systems and applications with Enterprise Application Integration (EAI), which would be linking diverse systems and applications of the enterprise to enable the enterprise to adapt to the dynamic business environment. Whereas, with traditional EAI solution, it is difficult to efficiently link the different proprietary applications and data sources, and difficult to enable the system to rapidly identify and respond to changes in the dynamic business environment. This paper discusses the traditional EAI technologies and the bottlenecks for applications integration, also elaborates the Web Services technology and the advantages suited for applications integration. Therefore, the model of EAI with Web Services (EAIWS) is proposed to efficiently implement the integration of diverse applications and systems within or between the enterprises. With EAIWS, this paper provides an example to explain how the application system works. This paper also discusses the advantages of EAIWS..

Key-Words: - EAI, Web Services, Integration

1 Introduction

Most of the enterprises might use some kinds of isolated IT systems, such as Enterprise Resource Planning (ERP) systems, Customer Relationship Management (CRM) systems, Supply Chain Management (SCM) systems, Office Automation (OA) suites and so on, to automate distinct business practices. These systems often coexist with a wide variety of in-house applications[1]. In order to improve business by automating business processes that span these isolated systems, the enterprise needs to integrate the isolated systems and applications with Enterprise Application Integration (EAI), which would be linking diverse systems and applications of the enterprise to enable the enterprise to adapt to the dynamic business environment. Traditional EAI solution generally uses some technologies which include Remote Procedure Call (RPC), Shared Data Files， Opening Database and other distributed technologies such as Common Object Request Broker Architecture (CORBA), Distributed Component Object Model (DCOM), Remote Method Invocation (RMI),etc. But using these technologies, the efficiency, integrality and maintainability of the integrated enterprise application system are not very well. When some errors have happened, it is difficult to check the reasons. That means the enterprise always faces the future running risk[2]. The development of Web Services technologies，which is based on open standards (such as XML, SOAP, UDDI, WDSL), easy to use and widely supported and deployed，provides a better integration solution In our approach, we consider to use Web Services technologies for EAI.

2 Issues the Traditional EAI

EAI, which involves linking diverse applications and systems, provides a structured approach to integration that results in flexible and scalable solutions to adapt to the dynamic business environment[3]. Generally, the EAI architecture pattern contains two types: Point-to-Point integration and Adapter integration[2]. Fig.1 describes these patterns. The EAI based on Point-to-Point pattern generally results in high application coupling and the inability to share common logic (e.g., the integrated product availability logic implemented in the ordering application is not available to other application), both of which reduce the flexibility of the business and system infrastructure. For example, if a new application is added to the enterprise that needs an integrated view of product availability (e.g., a web-based ordering application), then the integrated product availability logic must be repeated for the new application. Worse still, if one or both of the availability systems should change, then all the applications that repeat the integrated product availability logic must change. Therefore, the Adapter integration pattern is a better EAI architecture, which has been the basic pattern of EAI solution (The following discussion about EAI will

always be based on this pattern). Then, the EAI can connect each of the IT systems to a central pattern that control the interaction between different systems. Each time introducing a new technology or application, the EAI can link it to this pattern, enabling that technology or application to participate in automated business. All of systems communicate and exchange data through the integrated pattern.

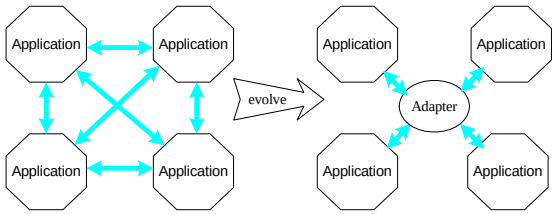

Fig.1 Point to Point Integration Pattern vs Adapter Integration Pattern

Since EAI can provide services to e-business, the contents of EAI mainly include: information integration and business process integration [3]. The technologies to implement the EAI mainly include RPC, Shared Data Files and Opening Database and other distributed technologies such as CORBA, DCOM, RMI, etc. The Shared Data Files technique uses the batch files to communicate between two applications, and the Opening Database shares the data source for other applications, which would hide the great security risk. These two technologies only fit for the monolithic system environment and have been outdated. The RPC also is an old distributed technology, which has been replaced with other distributed technologies such as CORBA, DCOM, RMI. The applications based on CORBA—a specification debuted in 1991 for achieving interoperability between distributed computing applications—never achieved this level of standards agreement, hence these applications could communicate with other applications only by using a common broker. Likewise, the applications based on the DCOM—Microsoft's proprietary distributed application architecture—could communicate only with applications running in a Microsoft Windows environment. The RMI is only applied to the communication between java-based applications. Normally, the traditional EAI uses these technologies to integrate the systems and specialized adapter for connection to data sources and legacy systems. Unfortunately this monolithic EAI solution is always static and complex integration, therefore it is difficult to efficiently link the different proprietary applications and data sources and difficult to enable the enterprise to rapidly identify and respond to changes in the dynamic business environment[4].

To realize the true benefits of integration, we need a new approach to EAI that reflects the dynamic business that we want to achieve. This approach should enable enterprise to link all existing systems with any potential new technologies in an ongoing process of integration and adaptation. At the same time, it should also provide a decentralized model of integration, in which autonomous systems can participate easily in any number of collaborative processes, both inside and outside the boundaries of the enterprise. The Web Services, because they are: based on open standards, easy to use, widely supported and deployed , offer a new approach to EAI that would overcome the limitations of traditional EAI and provide a better idea to integration. The following sections, we would introduce what is Web Services and how they benefits to EAI.

3 Issues Web Services

Web Services utilize some XML-based standards: Simple Object Access Protocol (SOAP), Universal Description, Discovery, and Integration (UDDI), and Web Services Description Language (WSDL), ebXML, RosettaNet, and several other protocols that allow the applications to communicate with others[5]. XML has already been a fully recognized standard, having received the official blessing of the World Wide Web Consortium (W3C), the Web's official standards authority. It has become the standard for defining data interchange formats on the internet. XML supports HTTP protocol and uses structural tags to define values for the information, also lets users create their own tags. SOAP is an XML-based protocol for exchanging information in a decentralized, distributed environment. It provides an envelope that defines a framework for describing what is in a message and how to process it, encoding rules for expressing application-defined datatypes, and a convention for representing remote procedure calls and responses. UDDI is an XML-based specification for a registry of businesses and the Web Services they offer. By providing the necessary translations, it enables software to automatically discover Web Services and integrate with them. WSDL lets developers expose the syntax of a Web Service. Using XML format, it describes network services as a set of endpoints operating on messages containing either document or procedure-oriented information, and the operations and messages are described abstractly and then bound to a concrete network protocol and message format to define the endpoints. The SOAP, WSDL and UDDI standards

have been submitted for ratification to the W3C. Web Services also support the main current e-business standards such as ebXML, RosettaNet. Several other proposed standards, covering encryption, digital signatures, and other support technologies, are at various levels of official approval. Therefore, Web Services can be nearly any type of applications that have the ability to define to other applications what they do and can perform that action for authorized applications[6-9]. Specifically, a Web Service fits the following criteria:

①It is able to expose and describe itself to other applications, allowing those applications to understand what the service does.

②It can be located by other applications via an online directory, if the service has been registered in the directory.

③It can be invoked by the origination application by using standard protocols.

A simple Web Service is characterized by the three standards SOAP, UDDI, and WSDL, which taken together provide a basic "request and response" functionality. Thus, the simple Web Services can be used to efficiently deliver information such as news, stock, and weather reports to certain web application like web site. The complex Web Services might involve multipart, long-running transactions, perhaps involving several trading partner or suppliers and based on the ebXML and RossettaNet set of standards.

Since Web Services are entirely based on open standards which are mostly based on XML standard, they are easy to use and have been widely supported and deployed. The active supporters of Web Services include significant industry participants such as Microsoft, IBM, Intel, Sun Microsystems, Oracle, SAP, which can guarantee Web Services interoperability across platforms, operating systems, and programming languages. Thereby, it is easy to do integration between different applications by using Web Services. Previously, to do the integration, we had to build custom connectors between these applications. But, with Web Services, we basically work with XML[8-10].

4 EAI with Web Services

4.1 The Model of EAI with Web Services

Web Services offer a new model of integration, designed to overcome the constraints of traditional EAI solutions. The model of EAI with Web Services (shortly as EAIWS) is described in the Fig.2. In this model, each IT application system within the enterprise acts as a component in a loosely coupled architecture. Component interfaces, communications, data transformations, and directory information are all based on open, widely adopted standards. A Web Service is an accessible software component through the SOAP which is an XML-based protocol that runs on existing Internet technologies. Supporting for SOAP and other Web Services standards is a key element in changing the way to do integration. Using these standards, the enterprises can connect their disparate systems in a flexible, responsive IT infrastructure. For the application to be able to communicate with other applications, it must have a set of interfaces defined in WSDL. Defining those interfaces is called publishing WSDL. To make the application locatable, register it with a Web Service registry, such as a UDDI registry. Next, that is the process of invoking the Web Service. The application can invoke a Web Service by sending a request via SOAP and listening for the response. Within a private UDDI registry, an application could search for the appropriate Web Services at runtime. The management of the Web Service involves monitoring the Web service, ensuring that it remains secure, and measuring the performance of the service.

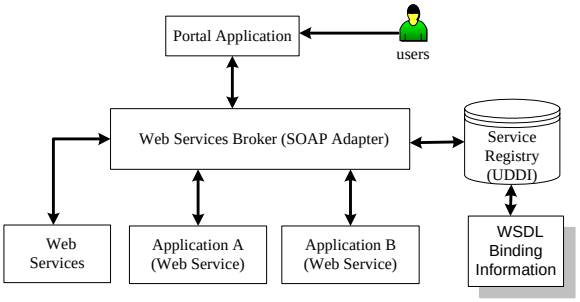

Fig. 2 The Model of EAI with Web Services

The EAIWS using Web Services can significantly change the traditional Point-to-Point approach. Using Web Services that loosely integrate applications, an application system achieves just a subsection of EAI. On the other hand, EAIWS takes a complete holistic approach of tightly integrating and connecting all applications and systems that support an enterprise's business. Otherwise, because Web Services loosely bind collections of services, the application services of EAIWS can be developed quickly and easily, published, discovered, and bound dynamically.

4.2 How to Work with EAIWS

In order to show how the integrated enterprise system works based on the above EAIWS model (Fig.2), we present a practical but simple example to describe. We assume there are some applications as Web Services within an enterprise. These applications

include Portal application, CRM application and ERP application. The Portal application running within an application server aggregates information from multiple internal applications, providing a single point of entry into business processes spread across those applications. The SOAP adapter server also is within that application server. The Portal application gets information about Web Services offered by internal applications using private UDDI registry and invokes these services over the intranet. WSDL binding information for frequently used Web Services can be cached by the application, to reduce the consumption of the resource and time during dynamic information binding. In this example, the Web Services loosely integrate Portal application with CRM and ERP applications. The steps of the procedure can be described as follows:

Step 1, after logging on to the enterpriser portal, users request information.

Step 2, the portal application gets information about Web Services made available by the CRM and ERP applications by doing a look up in the private UDDI registry.

Step 3, the location and WSDL binding information of Web Services is sent to the application server.

Step 4, the application invokes the Web Service published by the CRM application and retrieves the personal information, such as name, social security number, mail address and email address, of the user. The communication is based on SOAP.

Step 5, the application invokes the Web Service published by the ERP application and retrieves the account information, such as account number, balance and transaction history, of the user. The communication is based on SOAP.

Step6, the information is then formatted and sent to the user.

4.3 The Advantages of EAIWS

From the depiction of the above sections, we think the EAIWS provides many advantages as follows:

①Based on Open Standards: Unlike traditional EAI solutions, the EAIWS are built on existing and ubiquitous open standards and protocols such as UDDI, SOAP, WSDL, HTTP. Therefore, it is easy to add or link a new or legacy application.

② Simple and Flexible: There is no doubt that EAIWS using Web Services technologies is much simpler to design, develop, maintain, and use as compared to a traditional EAI solution which may involve distributed technology such as DCOM and CORBA. Once the framework of EAIWS is ready, it will be relatively easy to automate new business processes spanning across multiple applications. The EAIWS is quite flexible, as it is built on loose coupling between the application publishing the services and the application using those services.

③Efficient and fast: As mentioned in the previous point, Web Services allow applications to be broken down into smaller logical components, which makes the implement of EAIWS easier as it is done on a granular basis. This makes EAIWS solutions much more efficient and faster than traditional EAI solutions.

④ Dynamic: Web Services provide a dynamic approach to integration for EAI by offering dynamic interfaces, whereas traditional EAI solutions are pretty much static in nature.

⑤ Widely supported and deployed: The active supporters of Web Services include significant industry participants such as Microsoft, IBM, Intel, Sun Microsystems, Oracle, SAP, and so on. That means the EAIWS is easy to integrate applications across platforms, operating systems, and programming languages.

5 Conclusion

EAI would be linking diverse systems and applications across the enterprise to enable the enterprise information system to adapt to the dynamic business environment and neatly automate any business practices. But the traditional EAI solutions always be static and complex integration, therefore it is difficult to link efficiently the different proprietary applications and data sources, and to enable the system to rapidly identify and respond to changes in the dynamic business environment. The EAIWS solution overcomes these bottlenecks. It uses Web Service technologies which are Based on open standards (such as XML, SOAP, UDDI, WDSL), Easy to use and Widely supported and deployed, to seamlessly and loosely integrate or connect applications and systems that support the enterprises' e-business. In EAIWS, the applications would expose their public functions as services using technologies such as XML, SOAP, WSDL, and UDDI. Thus, the EAIWS solutions would have to provide a broad support for services integration beyond applications integration.

References:

[1] Manufacturers Parametric Technology Corporation, Exploring Enterprise Application Integration Solutions, http://www.ptc.com/WCMS/ files/ 048en_file1.pdf, 2002

[2] John S. Saling, The Role of the Enterprise Repository in Enterprise Application Integration, http://herbb.hanscom.af.mil/forums/aca-1/dispatch.exe/techsol/showFile/100060/d20011107153157/No/EP/Role/in/EAI.pdf, Nov.7,2004
[3] Jeff Pinkston, *eAI Journal*, 2001,8, 48-52
[4] Jeffrey .C, Web Services Architecture Usage Scenarios W3C Working Draft, http://www.w3.org/TR/2002/WD-ws-arch-scenarios-20020730/ , July.30 2002
[5] Rick Kuzyk, *eAI Journal*, 2002, 4, 23-25
[6] F.Leymann, *IBM System Journal*, 2002,41(2), 31-36
[7] Rosen and J.Boak, *eAI Journal*, 2002, 1, 43-46
[8] Scott Bluman, XML-enabled Enterprise Application Integration(XAI):Fundamental for B2B integration, http://xml.e-centre.org.uk/download/XML-enabled/EAI.pdf , 2003
[9] Rahim Adatia, *Professional EJB*, Wrox Press, 2001,890-936
[10] F.Cura, *IEEE Internet Computing*, 2002, 26(2), 86-93