The Study and Application of System Integration Based on ESB for Enterprise Group

Xing MA, Quan-bing YAO and Shuo SHAN
Centre of Information Technology, CGN Power Co., Ltd., Shenzhen City, China

Keywords: ESB, SOA, System integration.

Abstract. Most enterprises usually face the integration problem of heterogeneous systems in the process of information construction, while the traditional way of interface lacks flexibility and expandability. Based on the basic principles and architecture of ESB, this paper proposed a lightweight ESB integration solution for enterprises and establishes a system integration platform based on ESB. The centralized management and control of the service has been realized through the ESB platform, thus provides a powerful guarantee for the enterprise system integration.

Introduction

With a rapid development of enterprise informatization in the past few years, many enterprises have plenty of information systems with different types. These systems differ in their business functions, system architecture, and databases to meet the diverse needs. As time goes by, these separate systems became "data islands". The traditional way of integration is to develop interfaces between systems to achieve data access, which also causes the problem of excessive coupling between systems.

Service Oriented Architecture (SOA) is a strategy of enterprise business integration, which emphasis on reuse and loose coupling. As the core of SOA, the Enterprise Service Bus (ESB) is designed to integrate heterogeneous platform applications, provide interactive, collaborative, and combined web-based distributed bus services for SOA. This article explores a cross-platform, cross-system application integration solution based on ESB to deal with the issues facing the enterprise informatization under the traditional architectures.

ESB’s Technical Principle

The ESB is a flexible, interoperable architecture that consolidates applications and services. It reduces the number and complexity of interfaces in SOA architecture, improves business processes, integrates heterogeneous applications, and enhances the value of data assets as well. It is a loosely coupled, distributed, event-driven enterprise-class SOA. It is also a pre-packaged implementation of SOA that contains the basic functional components necessary for a SOA tiered goal.

ESB is a combination of traditional middleware technology, including XML and Web services. It provides an open, standards-based messaging mechanism that providing coarse-grained application services and other components through standard adapters and interfaces to meet the application integration needs for large enterprises. The messages that ESB delivered need to be translated, subscribed, published, delivered, and dispatched. The message types also contain event parsing, event notification, event registration, and event storage etc. The principle is: through the standard integration technology, SOA, Web Service and XML technologies are fused into a unified distributed architecture, which are easy to deploy and manageable. It can integrate new application services. By decomposing and packaging legacy systems, the existing applications can also be integrated. ESB provides existing software application functionality that connects internal and cross-enterprise businesses, serves as a bridge between service consumers and service providers, thus the loosely coupled, platform-independent service interfaces can be integrated, as is shown in Figure 1.
Integration Ideas about the Enterprise Information System

In order to meet the increasing enterprises’ needs, a variety of different software applications are developed and applied, such as OA, ERP and so on. The data between such systems are isolated from each other, the goal of data interconnection is unable to achieve. Enterprises come up with a variety of technical means to link the data, such as HTTP protocol, JMS (Java Message Service) protocol, Web Services and so on. Enterprise groups may have a large number of member companies, the approval of different companies need to connect with the group or external entities. So more interfaces have been developed for inter-system data connection. The topological relationship between systems caused by this traditional architecture is very complicated.

Due to the complex topological relations between the original systems, the manual management is difficult and costly. When making changes, the whole system is affected. Besides, the system coupling degree is very high, which makes it difficult to meet the current requirements for the informatization. To this end, people put forward a new idea of software integration methods that is SOA. Under this circumstance, all software are transformed into separate services, the connection between the software is also used to describe and complete by the service. However, the time and cost of transformation work is unacceptable to the enterprise. ESB has become the core infrastructure of SOA architecture as a comprehensive platform for transformation and processing.

The difference of ESB solution is that the ESB publishes the service address by performing the necessary modifications without changing the technical means in the data-related part of the software. This address may conform to a uniform standard naming convention. The service consumer obtains the service through the address released by the ESB. Through the standardized management of the interface, the decoupling between the service consumer and the service provider can be realized.

As shown in Figure 2, an enterprise rebuilds its existing business system. Systems achieve data interoperability through the interface. The systems’ high-coupling problem was solved. Similarly, the dependence between services, the requirements of the original system and the transformation cost was reduced a lot. After all, the diverse needs of the business are satisfied.

ESB-Based System Integration

Take a construction management system (CMS) as an example to introduce an ESB-based solution. CMS and other management systems overlap in some functions. There are many interfaces for interaction between these systems. Without a unified standard to manage this complex system and interface, operation and maintenance cannot be carried out.
The main content of the integration between construction system, SAP and other systems is as follows: project management, construction information management, personnel management, workflow management, decision support, and information exchange. If the construction management information has changed, then the consistency of the relevant system data needs to be maintained. If the system is not one manufacturer's product, the systems can no longer be integrated at the database level. Here's how to solve the problem.

**The Hierarchical Structure of the ESB**

ESB commercial solutions are already mature, IBM, JBOSS, BEA have launched its own ESB software, these software are powerful enough, but the complexity, price and transformation cost is unacceptable.

The most basic realization of ESB is to adjust the mesh service mode to the bus service mode. The service is released for consumer call after ESB registration.

Figure 3 shows that the ESB has two ends. One end is an access end, which is connected to a specific service provider; the other end is a calling end, that is, an outgoing end, whose purpose is published the service on the ESB platform for the service consumer or the third party to call in. The service released by the ESB can be exactly the same as the original service, and also can be changed according to the service needs.

According to the idea of ESB, we adopted a lightweight ESB bus design to achieve system integration. Its hierarchical structure is shown in Figure 3.

![Figure 3. ESB’s hierarchical structure.](image)

Transport is responsible for different protocol access, such as JMS or HTTP types.

The proxy initially translates the incoming data and extracts the data from the concrete protocol.

The intermediary services provide services such as choreography, data conversion, protocol conversion and other functions.

The business service is responsible for converting the data into protocol related data and sending it to the endpoint.

The endpoint is responsible for sending the message to a third-party service.

**ESB Functional Architecture**

The functional architecture of ESB supports such functions as protocol conversion, message conversion, message routing, service orchestration, service registration, service discovery and service monitoring. The architecture is shown in Figure 4:

ESB Server: provides multi-protocol support and provides a reliable operating environment for service operation. The convenient extension mechanism provides effective support and control measures for the ESB to integrate into the enterprise IT environment, and greatly improves the performance without relying on J2EE container’s performance;

ESB Console: Provides management and update of ESB Server in cluster environment and system-level configurations such as various levels of logging control and service thread configuration;

ESB SSM (Service State Monitor): Provides functions of enhanced customer perception, such as: early warning in various dimensions, alarm in events, post-event statistical analysis and other functions;

ESB SAM (Software Assert Manager): Provides service metadata management functions such as service registration, service discovery and SLA management.
External Services Access ESB

External service access is illustrated in Figure 5. Before introducing the access, two concepts are introduced: public components and business components. The common component contains the configuration of two parameters, the transport and the endpoint. The transport is mainly for the access port information configured when the service consumer calls the service through the ESB. The endpoint is mainly the IP address and its port corresponding to the service provider. Business components mainly implement two aspects of configuration, one is to configure specific service provider information, the configuration information associated with the endpoint; the second is proxy the service out through the establishment of ESB proxy service. The proxy service is associated with the business service. Intermediary service is designed to deal with circumstances such as: the service is called upon by multiple protocols, or the data format needs to be converted, or the services need to choreograph. The data exchange between a CMS of an enterprise and its related systems is realized through an interface, which is published in the form of a web service. According to the hierarchical structure of Figure 5, the access process is described as Figure 5.

The CMS of an enterprise accesses the data of the project management system through the service published by the ESB and obtains the data of the project management system. The information of the relevant ports no longer relies solely on the access service itself, two systems achieve the purpose of decoupling and flexibility.
ESB Application

Based on the above ESB architecture, an enterprise group’s information department set up a system integration platform based on ESB, carried out the pilot work, and realized service registration and centralized monitoring functions. Figure 6 and Figure 7 reflect the operational monitoring information of the commissioned interface. So far, a total of 30 interfaces have been put into operation. Under the 300,000 transaction scale, a total of 213 alarm messages have been processed to ensure the stable operation of the system to the maximum extent. During the pilot phase, the ESB platform received nearly 5000 trades a day, reaching a peak value of 8,000. The platform enabled instant alert notification of abnormal interface messages. The detailed information of each transaction can be queried based on user defined conditions; the validity of the architecture is validated.

![Figure 6. Comprehensive statistics of ESB interface operation.](image1)

![Figure 7. Detail query results diagram of the construction management system.](image2)

Performance Comparison

Due to the inaccessibility of core index data of commercial ESB, we compared the throughput and response time of the service before and after accessing the ESB. Based on a single PC Server (Intel Core 2 Duo Q8300), the ESB processes 3600 messages of 5K per second and processes about 12 million messages per hour with a CPU utilization of 41%. Instead of using ESB, the background web service is called directly. The processing efficiency is 3700 pens per second, and about 13 million pens per hour. The relationship between the number of clients, throughput, and response duration is shown in the Fig.8. As the number of clients increases, the throughput and response time of the service increase. The response time of the ESB is less than 10 milliseconds. The performance difference between before and after ESB access becomes smaller and smaller. When the number of clients increases to 24, the performance loss is almost negligible. Such performance can easily cope with the business needs of a typical enterprise.
Summary
The enterprise ESB integration platform reduces the cost of system operation and maintenance, the system coupling degree is also greatly reduced. Standardized management of service access to service termination is enabled. System administrators can get exception and alarm information in time and 10% of services are optimized. In addition, the ESB platform turns services into quantitatively manageable software assets, increasing the reuse of services and reducing the cost of system development. Based on the initial application of the situation, we will continue to improve and optimize the platform according to the actual needs. Eventually, centralized management and control of the enterprise group’s services can be realized.

Acknowledgement
This research was financially supported by the CGN’s SNP (Smart Nuclear Power) research fund.

References
[1] Huang W, Sun B C, Xu M D, et al. Enterprise Service Bus Based on OSGi[M]// Proceedings of the 22nd International Conference on Industrial Engineering and Engineering Management 2015. Atlantis Press, 2016.
[2] Serrano N, Hernantes J, Gallardo G. Service-oriented architecture and legacy systems[J]. IEEE software, 2014, 31(5): 15-19
[3] Chaudhari N, Bhadoria R S, Prasad S. Information Handling and Processing Using Enterprise Service Bus in Service-Oriented Architecture System[C]//Computational Intelligence and Communication Networks (CICN), 2016 8th International Conference on. IEEE, 2016: 418-421.
[4] Jayashree K, Ramalingam C. Service Cluster Approach in Enterprise Service Bus[M]//Exploring Enterprise Service Bus in the Service-Oriented Architecture Paradigm. IGI Global, 2017: 128-141.
[5] Zeng Wen-ying, Zhao Yue-long, Qi De-yu. ESB principle,architecture,implementation and application. Computer Engineering and Applications, 2008,4 4(25):225-228.