



pointing the way

White Papers

JBoss ESB

Reliable SOA Infrastructure



Introduction

This paper describes basic concepts of JBoss ESB. It shows main features and common use of ESB mechanisms.

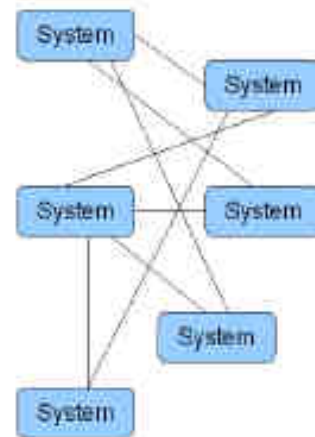
Background

Enterprise Service Bus (ESB) is middleware software that provides reliable channel of message interchange between services. ESB is tightly related to SOA concept. Even if SOA can be realized without ESB, in most cases ESB constitutes the key part of SOA.

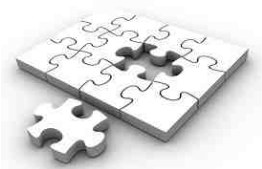
What is SOA

Service Oriented Architecture (SOA) is neither technology nor specific product. It is rather the design style for information systems. It describes how the system architecture should be designed, and influences organizational and methodological decisions in organization.

Enterprise information networks commonly consist of significant number of applications, subsystems, and interfaces to external services. Usually, mutual dependencies between those components raise the costs of further development and maintenance.

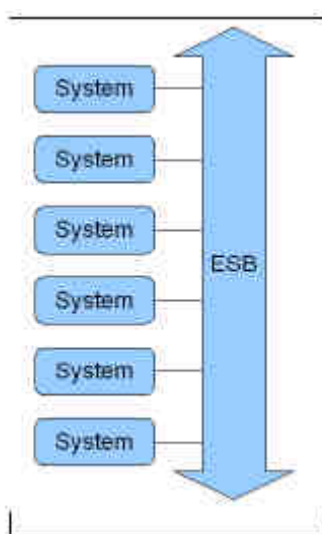


Mutual dependencies are not the only challenge. Different systems usually use different data formats, different communication protocols, different security or transactional rules. It relates also to links with systems external to the enterprise.



Earlier attempt to solve these problems was Enterprise Application Integration (EAI). However, it involved mostly technologies that were proprietary, expensive, and time-consuming to implement. These solutions range from expensive vendor solutions (high cost, vendor lock-in) to home-grown custom solutions (high maintenance, high cost). The overwhelming disadvantages of these solutions are high cost and low flexibility due to non-standard implementations.

SOA is different, because it is based on open formats, like web services (SOAP, WS-*) for communication protocols, and XML-based (XPath, XQuery) expressions for content routing. SOA simplifies the enterprise subsystems mix by introducing common bus for communication, with unified data format. Each subsystem is treated as a service, with clearly-defined interface. Each service can communicate with other services, but not directly, but through the common bus. In fact, the service is not interested in actual location or implementation of other services, only in their interfaces. Actual binding of interfaces with implementations, and routing, is done by the bus. The bus is what we usually call ESB: Enterprise Service Bus.



Enterprise Service Bus is a flexible middleware for systems communication.

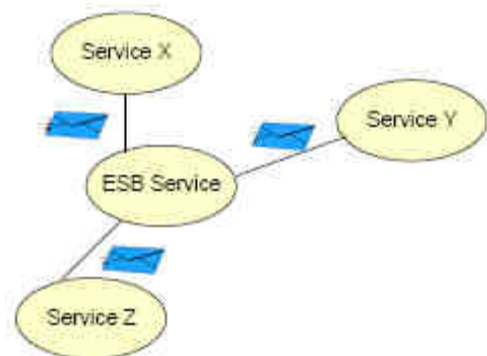
The role of ESB

As it was described in previous paragraph, ESB acts as a central point of intercommunication between services. Using ESB as a message broker between applications reduces the tight point-to-point coupling between them, thus reducing costs of maintenance.

ESB supports SOA by:

- ✓ Decoupling the consumer's view of a service from the actual implementation of the service
- ✓ Decoupling technical aspects of service interactions
- ✓ Integrating and managing services in the enterprise

ESB provides an infrastructure that removes any direct connection between service consumers and providers. Consumers connect to the bus and not the provider that actually implements the service. Such approach allows limiting the number of connections between these components and as a result the scale of interdependencies within the integrated system.



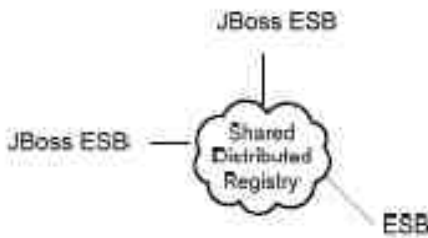
Basic communication concept in ESB architecture.

Basic ESB concept assumes that every service consumer and service provider including even bus is a service. All services interact via messages.



There are two levels of messages: first are seen and used by clients and services, second are seen and used by the core bus service.

ESB Service is a central communication point but in more complex architecture services can be plugged into multiple buses concurrently.



ESB should be standards-based and flexible, and support many types of communication channels (protocols). Most ESB solutions are based on Web Services Description Language (WSDL) technologies, and they use Extensible Markup Language (XML) formats for message translation and transformation.

XML

XML is a platform and programming language independent format for representing data. Because of these characteristics it is a common choice for integration solutions. XML format defines types and structure of the data elements.

Web Services

Web Services are technology build on XML and HTTP standards which allow remote procedure calls (RPC) or message-style communication independent of the underlying programming language.

Web Services support SOA by:

- ✓ providing an open-standard and machinereadable model for creating explicit, implementation-independent descriptions of service interfaces,
- ✓ providing communication mechanisms that are location-transparent and interoperable.

ESB Functionalities

ESB is usually responsible for tasks such as:

- ✓ Addressing and routing
- ✓ Synchronous and asynchronous style
- ✓ Multiple transport and protocol bindings
- ✓ Content transformation and translation
- ✓ Business process orchestration
- ✓ Event processing
- ✓ Providing adapters to multiple platforms
- ✓ Integration of design, implementation, and deployment tools
- ✓ QOS features like transactions, security, and persistence
- ✓ Audit, logging, and metering
- ✓ Service management

There are many implementations of ESB on the market, both commercial and open-source. JBoss ESB is the ESB implementation from the global leader in open source middleware solutions, JBoss.

JBoss ESB

JBoss ESB is an open implementation of ESB concept. It may be run as an application deployed to JBoss Application Server or Tomcat container. It may be also run standalone, as JBoss ESB Server. JBoss ESB Server is a lightweight, stripped-down version of the JBoss Application Server, with much quicker boot time than full-weight Application Server.

Key features of JBoss ESB include:

- ✓ A pluggable architecture enabling all JBoss ESB subsystems such as messaging and transformation to be swapped with other alternatives, which gives customers flexibility and choice.
- ✓ Support for a variety of messaging services, including secure FTP, HTTP, email and JMS (JBossMQ, JBoss



Messaging, IBM MQSeries, and ActiveMQ).

- ✓ Transformation engine that bridges data formats for seamless communication, supporting XSLT and Smooks.
- ✓ Service registry for service discovery and integration, using JAX-R and UDDI.
- ✓ Persisted event repository to support governance of the ESB environment.
- ✓ Notification service to allow the ESB to register events and signal subscribers.
- ✓ Content-based routing based on XPath and JBoss Rules for a more flexible and dynamic alternative to publish-subscribe.
- ✓ Gateways that allow non-ESB aware clients to interact with services deployed within the JBoss ESB environment.

JBossESB is packaged and shipped with base services. A service in JBoss EJB is a deployable service unit archive: simply a zip file with an .esb extension, following internal structure requirements.

Normally, there will be many ESB archives deployed to the JBoss ESB. JBoss EJB simplify also integration with other JBoss tools like:

- ✓ Drools (rules engine and business rule management system),
- ✓ jBPM (JBoss Business Process Management),
- ✓ Hibernate (ORM tool).

Conclusion

JBoss is a flexible, open source, production-ready platform for building service-oriented architecture (SOA). SOA and ESB simplify enterprises architecture and reduces costs of systems developments and maintenance.

Links

JBoss ESB Presentations

<http://www.jboss.org/jbossesb/presentations.html>

