Fiorano
Enabling change at the speed of thought

www.fiorano.com

Message Driven SOA -- Enterprise Service Oriented Architecture

Demystifying ESB Technology

The creation of a true standard enterprise backbone

--- Atul Saini

AMERICA'S
Fiorano Software, Inc.
718 University Avenue Suite 212, Los Gatos,
CA 95032 USA
Tel: +1 408 354 3210
Fax: +1 408 354 0846
Toll-Free: +1 800 663 3621
Email: info@fiorano.com

EMEA
Fiorano Software Ltd.
3000 Hillswood Drive Hillswood Business Park Chertsey Surrey
KT16 0RS UK
Tel: +44 (0) 1932 895005
Fax: +44 (0) 1932 325413
Email: info_uk@fiorano.com

APAC
Fiorano Software Pte. Ltd.
Level 42, Suntec Tower Three 8
Temasek Boulevard 038988
Singapore
Tel: +65 68292234
Fax: +65 68292235
Email: info_asiapac@fiorano.com

DEMYSTIFYING ESB TECHNOLOGY

The creation of a true standard enterprise backbone

Executive Summary

Emerging standards for enterprise communication, connectivity, transformation, portability, and security have tried to simplify the enterprise integration and middleware problem. The Enterprise Service Bus (ESB), a new variation of software infrastructure, has added to the range of standards based technologies enterprises can use as the Enterprise Nervous System (ENS) backbone. This article clarifies ESB technology and the subtle but important differences in vendor implementations.

Today's enterprise networks typically deploy hundreds of applications from different vendors. There's little standardization of communication protocols between individual enterprise systems, and exchanging data between applications from different vendors is surprisingly hard. The lack of a standard platform for distributed enterprise applications increases the cost and complexity of developing and deploying business solutions. The ESB is a new breed of enterprise middleware designed to alleviate these and other problems.

The ESB isn't a revolutionary concept; it has evolved with the gradual emergence of enterprise standards for communication, connectivity, transformation, service-oriented application construction, portability, and security. The ESB promises, for the first time, the creation of a true standard enterprise backbone for deploying business processes, collaborative systems, and distributed business solutions.

Enterprise Solution Requirements

Requirements for enterprise solutions have been intact for 20 years. Enterprise solutions typically consist of one or more distinct applications (also known as services or components), combined into a single distributed solution with these characteristics:

- **Communication**
  Services need to communicate reliably with each other over the network. A reliable, scalable, robust, and location-independent communications system dramatically reduces the development time for distributed systems while increasing reliability.
- **Connectivity**
  To extract data from a service, one first needs to be able to easily connect to that service. Absent any standards, this has been difficult.
- **Transformation**
  Data produced by a given service is typically not easily understood by another service; to make the data digestible by another service, it first needs to be suitably transformed.
- **Service-Oriented Application Architecture**
  Distributed enterprise systems span multiple nodes and operating systems, and typically consist of a set of independently running services loosely bound to each other via event-driven messages. This SOA for application composition allows incremental, dynamic extensibility and greatly reduces costs of maintenance and Total Cost of Ownership (TCO).
- **Portability**
  Most enterprises have a variety of computer systems, ranging from thin-clients and Windows desktop PCs to higher-end UNIX servers and mainframe systems. Portability and ease of communication between different operating environments remain concerns for enterprise solutions.
- **Security**
  Finally, all connectivity to and communication between enterprise services need to be secure at levels satisfactory to the enterprise. Since distributed applications span different departments and locations within and outside the firewall, security is highly important.

Early Integration Solutions

Several companies developed Enterprise Application Integration (EAI) solutions during the mid-'90s with the above design patterns in mind. Unfortunately, all these solutions were proprietary.

WebSphere MQ and TIBCO Rendezvous are good examples of proprietary communications buses. Several companies developed proprietary connectors to many packaged enterprise applications (including SAP, PeopleSoft, and Oracle Applications, among others). Yet, lacking any standards for connectivity, brand new adapters had to be developed for different versions of each enterprise system or application. This resulted in tremendous maintenance problems.

Without standards for transformation, each EAI software vendor also developed customized transformation engines, leading to even more proprietary middleware. Most solutions were developed in C/C++ and other proprietary, non-portable languages, resulting in vastly increased porting and maintenance costs. Finally, security was typically added as an afterthought to most EAI and enterprise middleware platforms, resulting in a fragmented security model.

It's no wonder CIOs have felt they had a difficult set of problems on their hands.

Standards Emerge

Emerging standards for enterprise communication, connectivity, transformation, portability, and security have greatly simplified the enterprise middleware problem. Enterprises should, at all costs, avoid dependence on a single vendor. This is possible now because of the opportunity to apply truly open industry standards in a heterogeneous, best-of-breed environment. Standards based technology coordination expands choices and is less costly. Specific standards include:

- **Communication Standards**
  Since 1998, the Java Message Service (JMS) has emerged as the dominant industry standard for enterprise communication, implemented by thousands of companies. JMS has gained such a strong mindshare that other Message- Oriented Middleware (MOM), such as IBM's WebSphere MQ, has had to offer a JMS implementation. The secret to the popularity of JMS is that it combines a level of functionality suitable for almost all market needs with an attractive price point and the benefits of being an industry standard.

- **Connectivity Standards**
  Web services standards allow any enterprise system or application to efficiently expose interfaces to the external world, dramatically simplifying the problem of connectivity to enterprise systems. These standards include Simple Object Access Protocol (SOAP), Universal Description, Discovery, and Integration (UDDI), and Web Services Description Language (WSDL). SOAP provides Remote Procedure Call (RPC) capabilities for XML. UDDI provides a searchable registry of XML Web services. WSDL is an XML based Interface Description Language (IDL) for describing XML Web services. XML is the standard format for Web data, and is beginning to be used as a common data format at all levels of the architecture.

- **Transformation Standards**
  Over the years, extensible Style sheet Language Transformation (XSLT) and X-query have emerged as enterprise standards for transformation, with support from most vendors and widespread acceptance from customers. XSLT transforms XML documents from one schema into another; it's used for data interchange between systems using different XML schema, or mapping XML to different output devices.

- **Service-Oriented Architecture (SOA)**
  The emergence of SOAs lets you compose complex distributed applications (spanning multiple platforms and operating systems) as a set of services with a defined form of invocation, both asynchronous and synchronous. This approach lets you compose or assemble applications from prebuilt, pre-tested services; the composed application is easy to modify or extend via service addition or replacement. This directly promotes reuse within the enterprise, decreasing time-to-market and system TCO.

- **Portability**

The Java programming language (used today by more than three million programmers worldwide and adopted by all Fortune 1000 companies) is the standard for building portable enterprise applications and middleware. Modern middleware technologies implemented in Java run unchanged across multiple hardware and operating systems, including Windows, UNIX, mainframes, and mid-size systems.

- **Security**
  The widespread acceptance of J2EE- and LDAP-compliant security gives administrators fine-grained control over the execution of applications and services on machines across the network. Additionally, SSL-based transport level security based on Secure Sockets Layer (SSL) provides robust security mechanisms for privacy and integrity checking. Lightweight Directory Access Protocol (LDAP) is a subset of X.500 designed to run directly over the TCP/IP stack. LDAP is, like X.500, both an information model and a protocol for querying and manipulating it. LDAPv3 is an update developed in the Internet Engineering Task Force (IETF), which addresses the limitations found during deployment of the previous version of LDAP. SSL is an open, non-proprietary protocol for securing data communications across computer networks. SSL is sandwiched between the application protocol and connection protocol. SSL provides server authentication, message integrity, data encryption, and optional client authentication for TCP/IP connections.

The presence of these foundation standards has spurred the evolution of a standards-based enterprise backbone: the ESB.

ESB Defined

A new Standards-based category for software infrastructure emerges.

The ESB is an enterprise platform that implements standardized interfaces for communication, connectivity, transformation, portability, and security. Emerging ESB implementations typically implement:

- Standards-based communication infrastructure (e.g., JMS)
- Standards-based connectivity such as Web services, Java 2 Enterprise Edition (J2EE), and NET adapters. (Sun's J2EE and Microsoft's .NET are the two dominant distributed computing architecture frameworks. J2EE provides portability of a single language [Java] over multiple operating systems and hardware platforms. .NET supports a wide range of languages but is primarily tied to the Microsoft Windows operating system and Intel hardware.).
- Standards-based transformation engines (e.g., XSLT and
- SOA for application
- Standards-based security (e.g., LDAP and Modern ESB implementations (see Figure 1) typically support development in multiple programming languages. This, coupled with the inherent portability of the ESB infrastructure, makes the ESB a true multi-language, multi-platform enterprise backbone.

ESB Benefits

ESBs leverage recent integration technology enhancements into a standards- based, affordable package. ESBs offer several advantages over existing, proprietary integration solutions:

- **Extended, Standards-based Connectivity**
  ESBs incorporate a standards based messaging backbone, letting systems within and across the entire value chain easily exchange information via asynchronous or synchronous messaging. ESBs provide enhanced systems connectivity using Web services, J2EE, .NET, and other standards.

- **Flexible, Service-based Application Composition**
  Based on SOA, the ESB application model allows complex distributed systems, including integration solutions spanning multiple applications, systems, and firewalls, to be composed from pre-built, pretested services. This provides easy extensibility.

- **Reduced TCO via Enhanced Reuse**

The SOA approach to application construction directly promotes reuse, easing maintenance and further reducing system TCO.

- **Reduced Time-to-market and Increased Productivity**
  ESBs provide these benefits through the reuse of components and services, and the ease of application composition offered by an SOA, standards-based communication, transformation, and connectivity. All these benefits derive from the strong support for standards in each component of the ESB architecture: communications, connectivity, transformation, portability, and security.

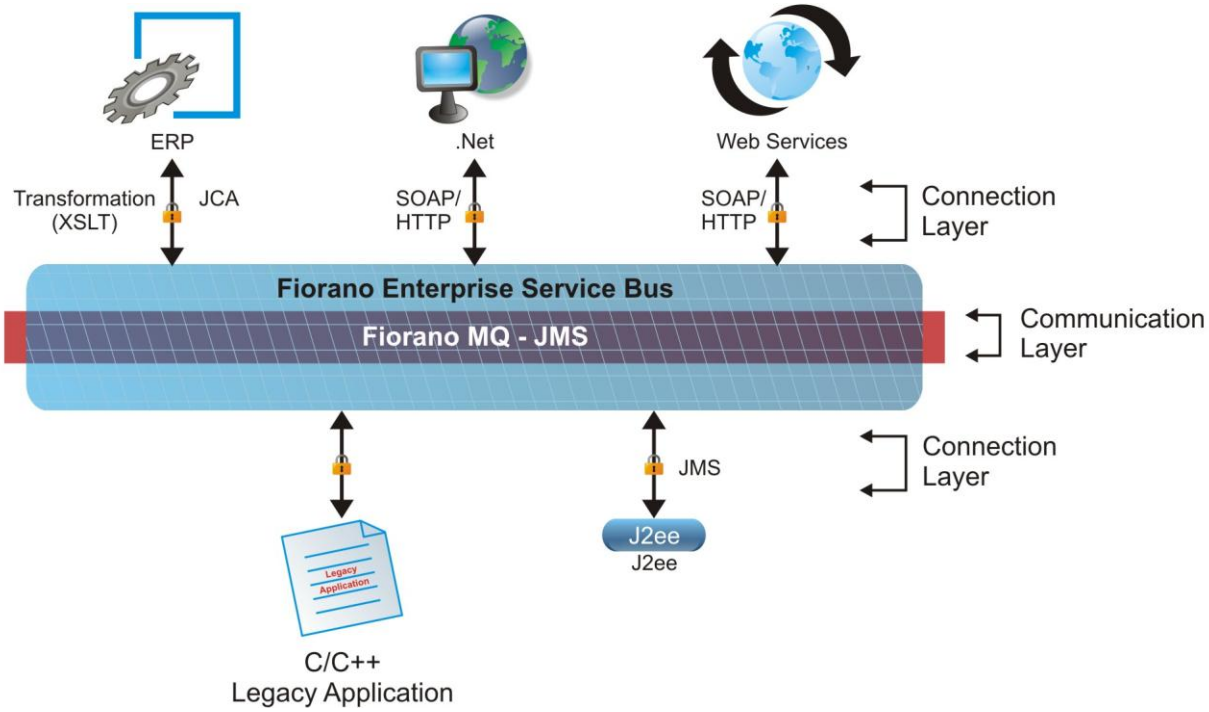

Figure 1: ESB Architecture

ESB Implementations

Different vendors choose different implementation strategies, often adding value in specific domains. While evaluating ESB implementations, consider these core characteristics.

Robustness, Scalability and Performance

While most ESBs support the JMS standard for communication, there are often vast differences in the quality of implementations. The most scalable implementations allow centralized control with fully distributed, parallel flows of data between participating services, with no single point of failure in the distributed infrastructure. Less scalable implementations support one or more centralized brokers, which tend to become data bottlenecks.

Data Routing Mechanisms

Participating services in a business process within the ESB framework can communicate using different methods for data routing. Most ESBs implement at least one of these methods, and some implement all:

- Traditional JMS routes, where data routing is a part of the business logic of the component(s)
- Content-based routing, where data is routed based on its content (typically XML-based)
- External routing, where participating services are completely oblivious to the routing of data between component instances. Such an approach allows easy reuse of components

across business processes, because the process of routing data is completely disconnected from the process of producing the data; developers are thus shielded from data routing and can focus exclusively on business logic creation.

- Service execution, which can be central or end-point

Compute-intensive Services

Compute-intensive services such as data transformation logic can either be performed at a centralized server, creating a potential performance bottleneck, or at the end-point of the network, allowing processing to be distributed for better scalability and performance. Orchestration and management tools: The base ESB infrastructure provides Application Program Interfaces (APIs) to route data between instances of executing services. Advanced ESB implementations, however, provide substantially added value by allowing the composition of business solutions from pre-built, pre-tested enterprise services. Such composite application platforms, deployed on top of the ESB, simplify the componentization of existing Web services, database applications, legacy systems, and J2EE and .NET software assets. This also enhances reuse within enterprise business processes, driving down development and management costs.

Orchestration and Management Tools

The base ESB infrastructure provides Application Program Interfaces (APIs) to route data between instances of executing services. Advanced ESB implementations, however, provide substantially added value by allowing the composition of business solutions from pre-built, pre-tested enterprise services. Such composite application platforms, deployed on top of the ESB, simplify the componentization of existing Web services, database applications, legacy systems, and J2EE and .NET software assets. This also enhances reuse within enterprise business processes, driving down development and management costs.

Summary

By leveraging emerging standards for communication, connectivity, transformation, and security, an ESB delivers a powerful, affordable, standards-based backbone throughout the enterprise and partner organizations. The ESB smoothes the operational path of the processes running a business and reduces the time, effort, and cost of integrating the different components supports these process steps. A powerful benefit of this type of approach is that it allows in-house development teams to build new applications that are already “integration-enabled” and can easily be incorporated into the ESB as required. Associated benefits include savings on investments of expensive skills, decreased time to market, and enhanced reusability of existing software assets.

About Fiorano Software

Fiorano Software (www.fiorano.com) is a leading provider of enterprise class business process integration and messaging infrastructure technology. Fiorano's network-centric solutions set a new paradigm in ROI, performance, interoperability and scalability. Global leaders including Fortune 500 companies such as Boeing, British Telecom, Credit Agricole Titres, Lockheed Martin, NASA, POSCO, Qwest Communications, Schlumberger and Vodafone among others have used Fiorano technology to deploy their enterprise nervous systems.

Note: This whitepaper has been reproduced from the original, which was published in Business Integration Journal.

Glossary Table:

Term	Brief Description
J2EE	Sun's J2EE and Microsoft's .Net are the two dominant distributed computing architecture frameworks. J2EE provides portability of a single language (Java™) over multiple operating systems and hardware platforms.
LDAP	Lightweight Directory Access Protocol (LDAP) is a subset of X.500 designed to run directly over the TCP/IP stack. LDAP is, like X.500, both an information model and a protocol for querying and manipulating it. LDAPv3 is an update developed in the IETF (Internet Engineering Task Force), which address the limitations found during deployment of the previous version of LDAP.
. NET	Microsoft's .Net and Sun's J2EE are the two dominant distributed computing architecture frameworks. . NET supports a wide range of languages but is primarily tied to the Microsoft Windows operating system and Intel hardware.
SSL	An open, non-proprietary protocol for securing data communications across computer networks. SSL is sandwiched between the application protocol (such as HTTP, Telnet, FTP, a NNTP) and the connection protocol (such as TCP/IP, UDP). SSL provides server authentication, message integrity, data encryption, and optional client authentication for TCP/ connections.
SOAP	Simple Object Access Protocol. SOAP provides HTTP/XML based remote procedure call capabilities for XML Web Services
UDDI	Universal Description Discovery and Integration UDDI provide a searchable registry of XML Web Services and their associated URLs and WSDL pages.
WSDL	Web Services Description Language. WSDL is an XML based Interface Description Language for describing XML Web Services and how to use them.
XML	XML has emerged as the standard format for web data, and is beginning to be used as a common data format at all levels of the architecture. Many specialized vocabularies of XML are being developed to support specific Government and Industry functions.
XSLT	Extensible Style sheet Language Transform. Transforms XML document from one schema into another. Used for data interchange between systems using different XML schema, or mapping XML to different output devices.