



IBM Software Group

IBM Enterprise Service Bus WebSphere® Enterprise Service Bus V6.0.1 WebSphere Process Server V6.0.1

ESB: Overview



@business on demand.

© 2006 IBM Corporation
Updated January 30, 2006

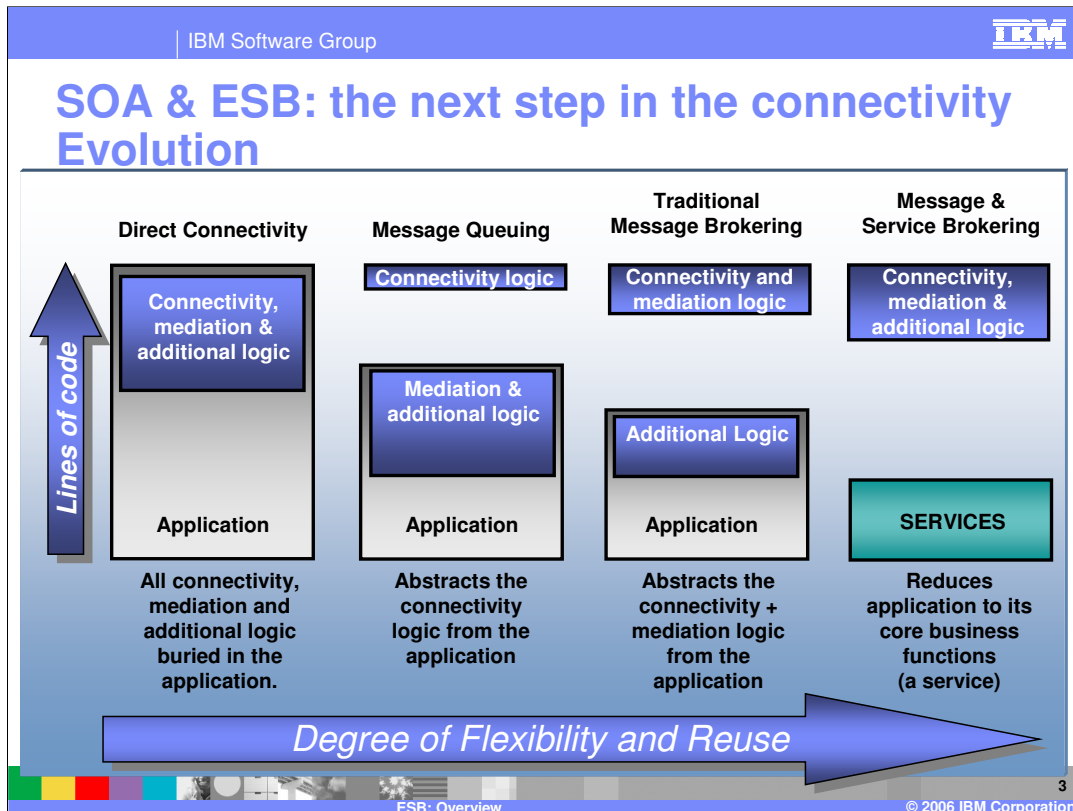
This presentation covers the overview of the products that use the IBM Enterprise Service Bus (ESB), including WebSphere Process Server V6.0.1 and WebSphere Enterprise Service Bus V6.0.1.

WebSphere ESB

- **Strategy and the role of ESB**
- IBM ESB Products
- WebSphere ESB and the WebSphere Family



The agenda starts with what ESB is and the how ESB can benefit integration connectivity. Next, this presentation will cover the WebSphere ESB V6.0.1 and WebSphere Message Broker V6. Finally the focus shifts to how WebSphere ESB relates to the other WebSphere family products.



SOA is not a new concept. It is the next step of a connectivity evolution that has been going on for some time, and indeed, WebSphere has already solved many of the problems with other technologies in use today.

The concept of message queuing is a powerful one that you may be using if you use IBM products like WebSphere MQ. WebSphere MQ decouples applications by abstracting connections through the use of queues. Instead of the application talking directly to another application, it talks to an intermediate queue which can then be directed to any application. Message queuing also eliminates the need for the application to deal with the intricacies of different platforms or handle the other application being busy or even off-line. It also assures that messages are delivered and not duplicated. All of this capability removes the need for interface code in your applications.

But message-queuing itself does nothing to help deliver information in the right format. Nor does it help you direct information to different targets based on the message content. You still have to build interface logic directly into your applications.

To eliminate this second level of interface logic, you need a different type of intermediary – a traditional broker. Brokers enable you to remove the transformation and routing logic from the application interfaces. Brokers can augment content and reroute based on message content. They can also translate between different protocols and programming models.

Nevertheless, even with a “traditional” broker, some logic may remain in your applications. To reduce this logic further – down to a bare-bones “service” – some additional standards are needed. First is the use of a Web Services Description Language (WSDL) interface. Second is the programming model for mediating between different programming models such as queue names (used with queuing), topics (used with publish/subscribe), and services (used by Web services). You may also perform other functions for the application such as the automation of request-reply calls. Finally, locating and binding to existing services -- or to the various transformations that may be required -- is another aspect that you may not want to handle from within your application program.

The objective of SOA is to reduce the service down to the bare business logic.

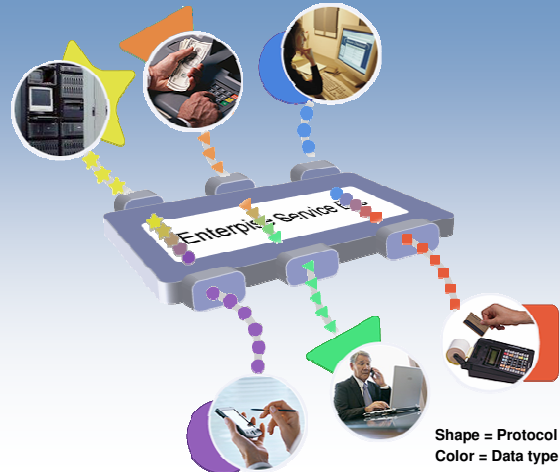
What is an Enterprise Service Bus?

An Enterprise Service Bus (ESB) is a flexible connectivity infrastructure for integrating applications and services

An ESB powers your SOA by reducing the number, size, and complexity of interfaces.

An ESB performs the following between requestor and service

- **ROUTING** messages between services
- **CONVERTING** transport protocols between requestor and service
- **TRANSFORMING** message formats between requestor and service
- **HANDLING** business events from disparate sources

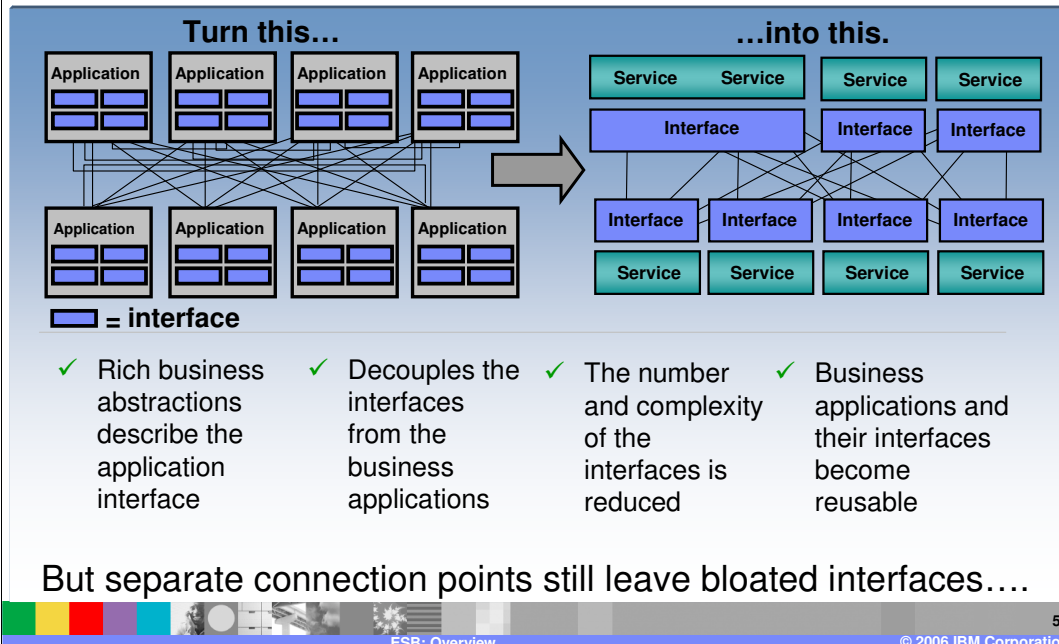


The Enterprise Service Bus (ESB) can help you achieve the goal of SOA. It is a flexible connectivity infrastructure for integrating applications and services. It is at the heart of an SOA, powering it by reducing the number, size, and complexity of interfaces.

The ESB does four things:

1. It ROUTES messages between services
2. It CONVERTS transport protocols between requestor and service
3. It TRANSFORMS message formats between requestor and service
4. It HANDLES business events from disparate sources

SOA without ESB decouples interfaces from Applications

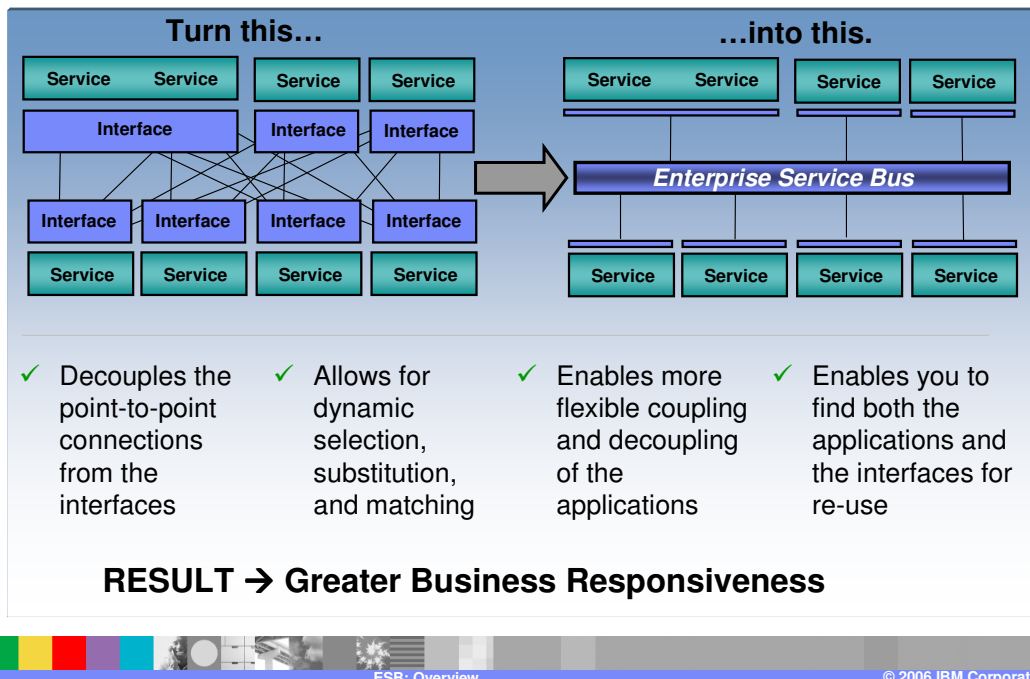


So how does basic SOA decouple the interfaces from their applications?

- 1) SOA uses a programming model that allows a rich abstraction of both the business application and the interface.
- 2) By abstracting that, the interfaces can be clearly separated from the business applications.
- 3) This enables you to reduce the number and complexity of those interfaces and,
- 4) It allows you to reuse both the interfaces and the business applications.

The problem is that you still have to build, find, and manage all of those interfaces somewhere.

SOA with ESB shrinks the interfaces further



The ESB shrinks the interfaces further.

- 1) It virtualizes the interface, or in other words, it decouples the point-to-point connections from the interfaces themselves.

The interfaces are put into a third party broker which helps you manage the interfaces better.

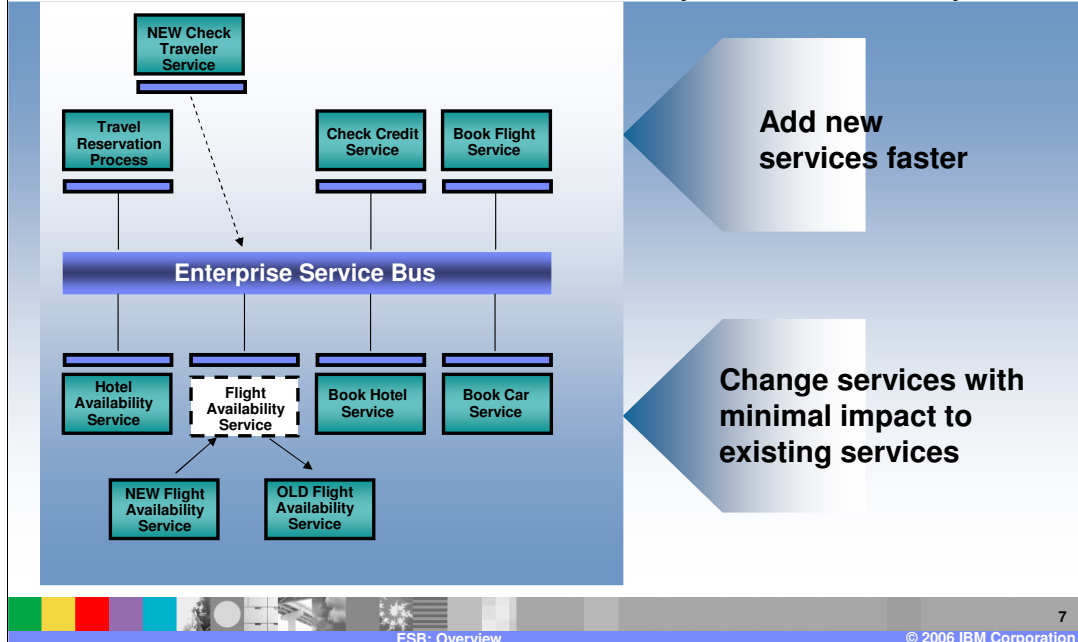
It enables faster and more flexible coupling and decoupling of applications.

Because you can find all of the applications and the interfaces, you can then reuse both.

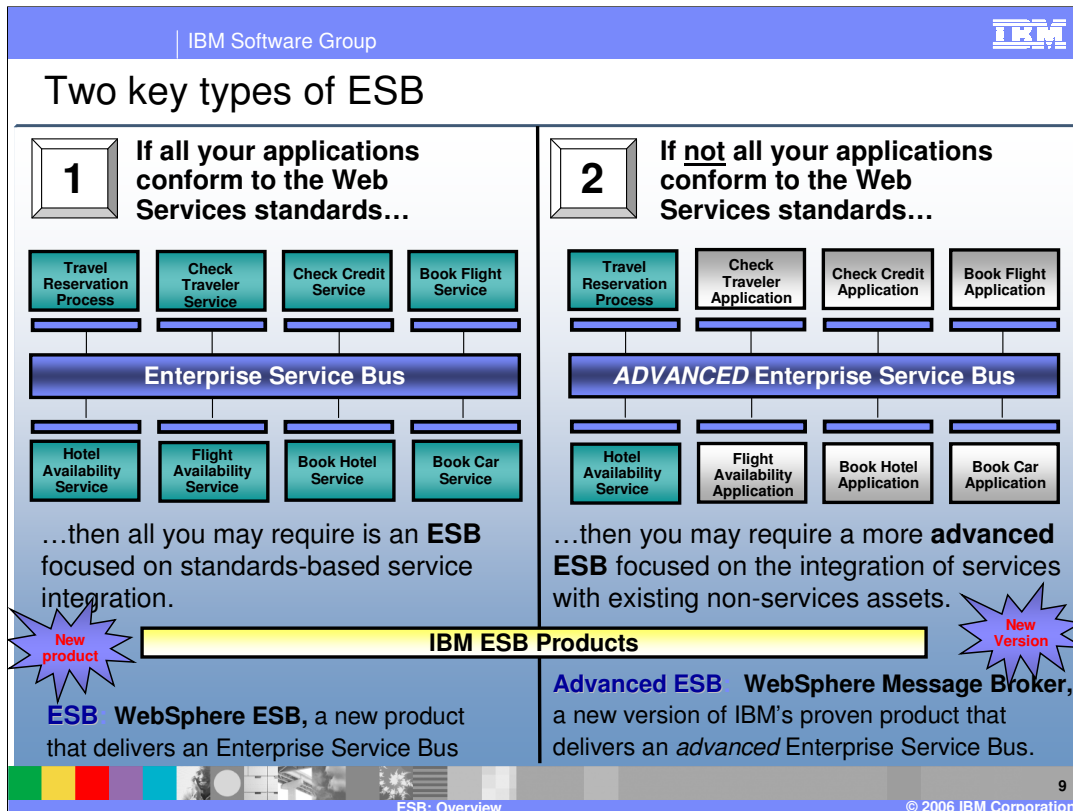
This results in achieving greater business flexibility and responsiveness.

The ESB allows you to focus on your core business *Rather than your IT*

Key Business Value Proposition



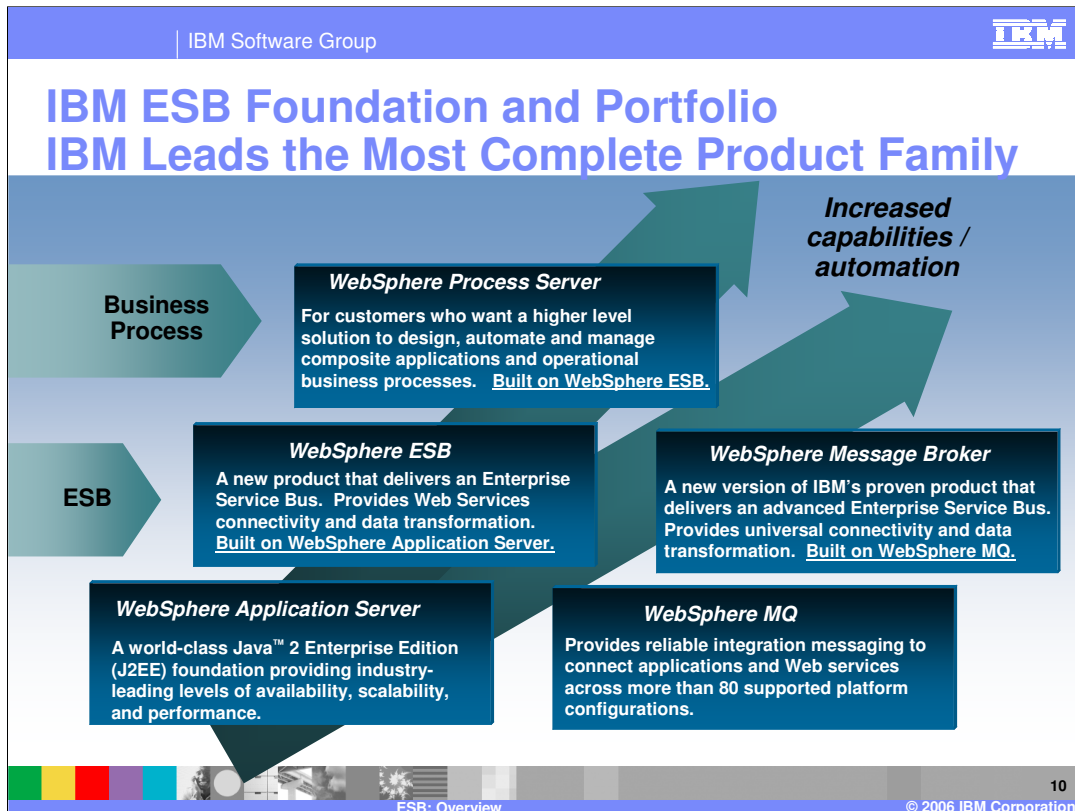
The ESB allows you to focus on your core business. It enables you to add new services faster and to change services with minimal impact to existing services.



If all your applications conform to the Web Services standards, you may only need an ESB focused on the integration of these standards-based interfaces.

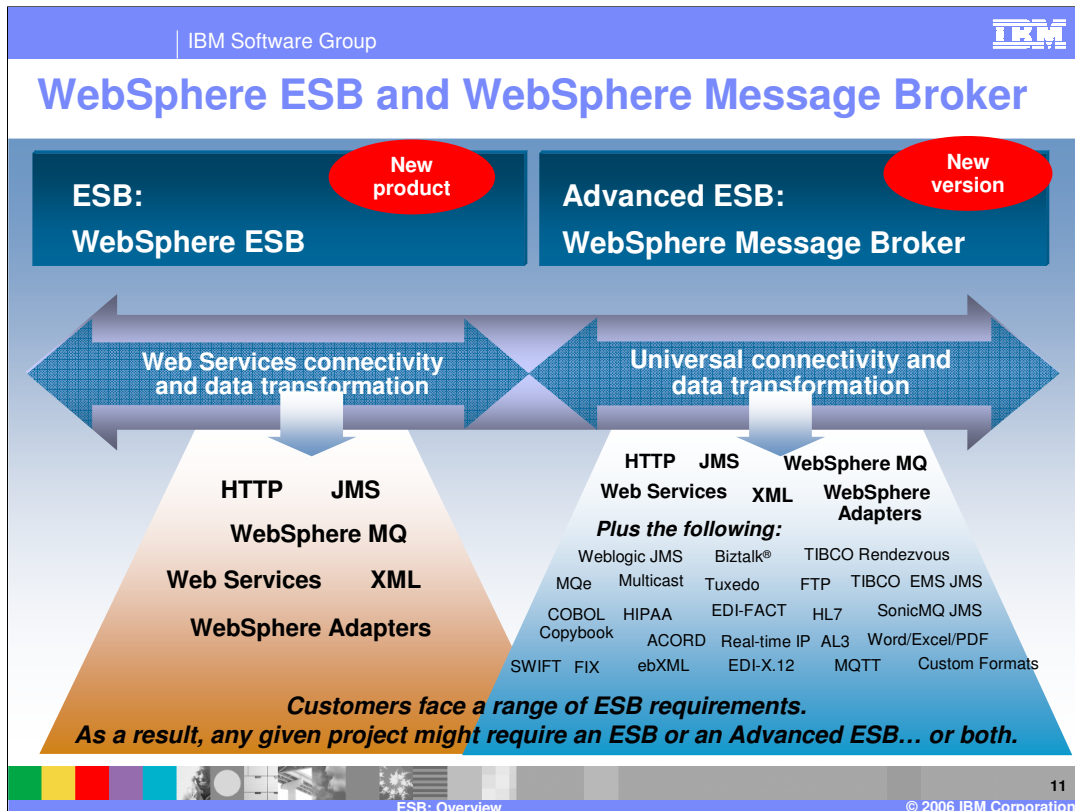
However, if not all your applications conform to the SOA standards, you may need something more advanced that can mediate between the SOA standards and everything else.

Based on the needs, IBM has announced two products focussed on providing ESB solutions. The first, a new product called WebSphere Enterprise Service Bus (ESB) V6, which is devoted to the integration of standards-based interfaces. The second, WebSphere Message Broker V6, which is a new version of IBM's proven Message Broker product but with additional features to deliver an advanced Enterprise Service Bus functionality.



WebSphere ESB is only one piece of the complete solution that IBM offers for transport, mediation, process, and transformation needs. On the left hand side of this picture, you can see how WebSphere ESB and WebSphere Process Server build on top of one another, WebSphere ESB to deliver a mediation and connectivity layer, and WebSphere Process Server to deliver a business process management layer.


In a similar way, WebSphere Message Broker builds on top of the word class transport within WebSphere MQ to provide transformation and universal connectivity. The stacks interoperate to provide a comprehensive solution.



As mentioned earlier, IBM offers two key ESB products: IBM WebSphere ESB and IBM WebSphere Message Broker. Selecting an ESB to power your SOA depends on your requirements. WebSphere ESB provides Web services connectivity and service oriented integration. WebSphere Message Broker offers advanced integration with universal connectivity and any to any transformation. Customers face a wide range of ESB requirements from the simple to the complex. As a result, any given project might require an ESB or an ESB Advanced or both.

IBM Software Group IBM

Selecting Your ESB based on Requirements



Full J2EE, JMS, and Web Services focus

↓

WebSphere ESB

Integration of Services with non-Services Applications

↓

WebSphere Message Broker

	WebSphere ESB	WebSphere Message Broker
Web Services Support	●	●
Message Transport & Protocol Switching	●	●
Intelligent Routing & Message Logging	●	●
Event Driven Processing	●	●
Transformation of XML Data Formats	●	●
Transformation of non-XML Data Formats		●
Complex Event Processing		●
Sensor & Device Integration		●
Native Integration with CICS® & VSAM		●
Third party JMS integration		●

ESB: Overview
12 © 2006 IBM Corporation

Selecting the right ESB product depends on your requirements. The table shows the capabilities of the WebSphere ESB and the WebSphere Message Broker products. The advanced ESB can support everything that you can do with the basic ESB as well as a few additions.

The first advantage is the ability to natively transform non-XML formats without the use of adapters. You can go directly from one format to another. This is very useful if you have data in your enterprise today that does not conform to the XML standard.

The second is complex event processing. Complex events are those that are formed by identifying several earlier ones. By identifying complex business events your business can act and react much earlier that you might otherwise.

Currently sensors and mobile devices don't support the SOA standards, but they can be integrated into an SOA architecture using the Message Broker.

The Broker also provides native integration of CICS and VSAM.

Finally, many companies use a range of JMS services from different vendors within their organizations. JMS is an open standard API but implementations from different vendors do not talk to one another. The Broker can handle inputs from virtually all these 3rd-party JMS systems and work as an integration hub for different vendors. Unlike other vendors, whose JMS implementations can works as transactional clients but not transaction managers, WebSphere Message Broker can act as a transaction manager. This means that it is the only product on the market that can manage a transaction involving multiple messaging products.

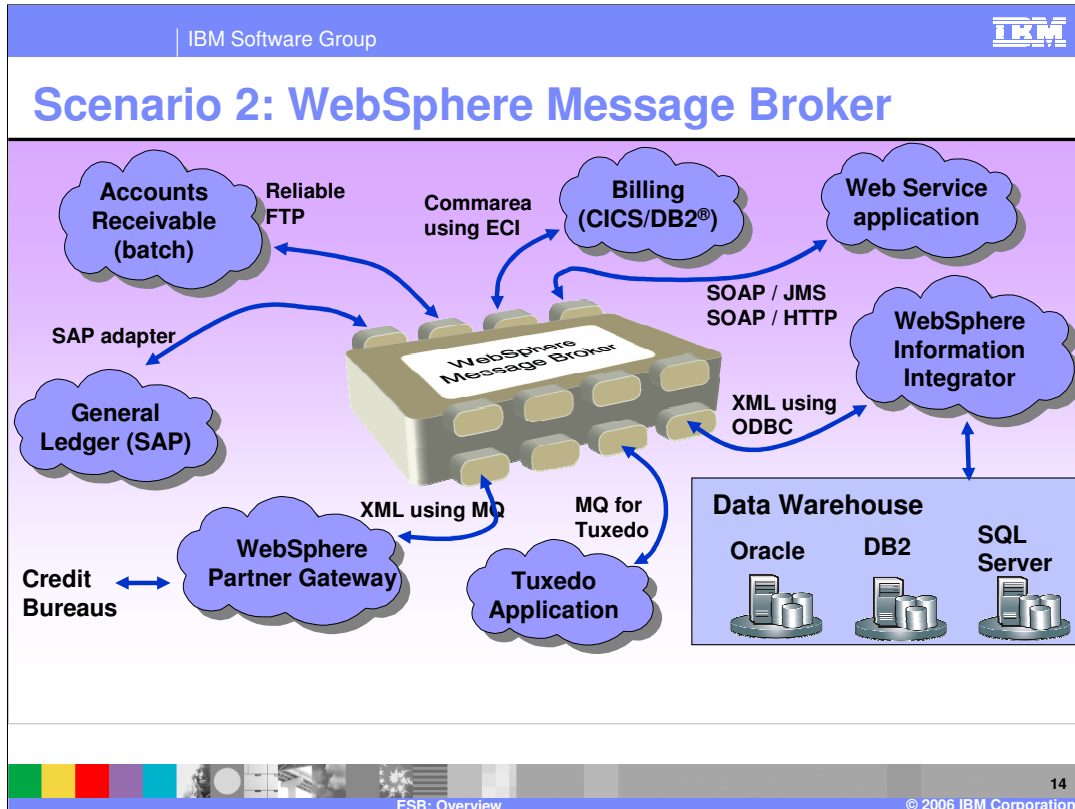
Scenario 1: WebSphere ESB



Scenario 1 demonstrates the capabilities of WebSphere ESB when used alone. It assumes that almost all of the interfaces are services-based (either SOAP/JMS or SOAP/HTTP). It can also talk to WebSphere MQ or WebSphere Message Broker or an adapter if it needs to talk to any system that is not using the Web Services or JMS model.

An example of this Solution is a Share Trader utilizing WebSphere ESB to implement an Enterprise Service Bus that does the following:

1. Routes incoming requests from the Web site to the appropriate service based on the customer's status using content based routing
2. Transforms the request to the appropriate format for the target destination using XSLT and protocol transformation and routes the request back to the source
3. Enables Share Trader to seamlessly transition service end points. Share Trader recently moved from an internally provided stock quote service to a more dependable external provider without any impact to the rest of the offering.

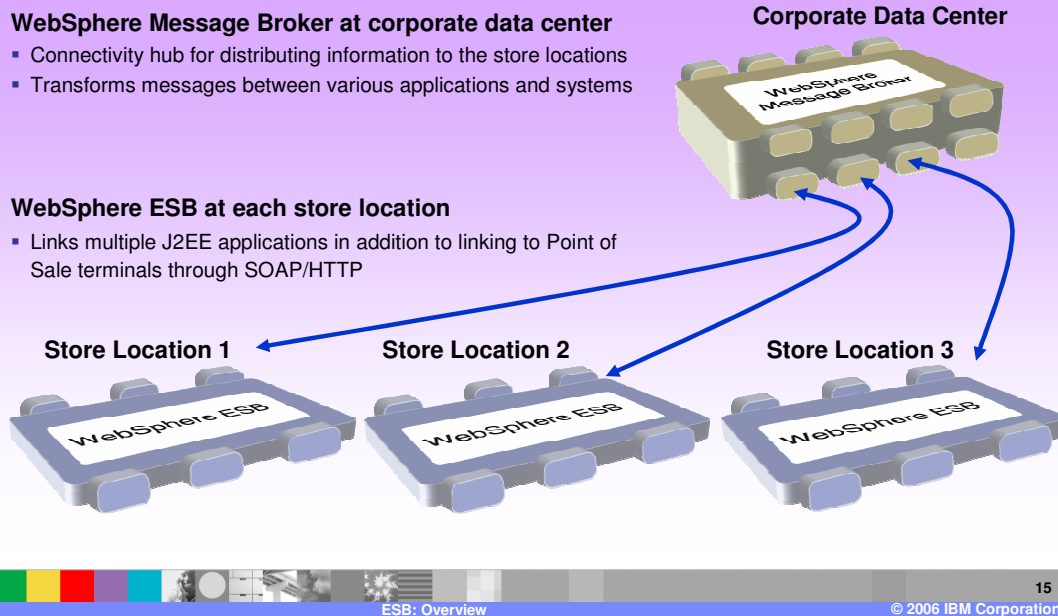


Scenario 2 is focused around WebSphere Message Broker. It assumes you have to connect to a wide range of business applications including FTP, SAP (through a non-services interface), Business to Business connections, CICS, BEA Tuxedo, and ODBC links to products such as WebSphere Information Integrator (which provides the capability to do a single SQL call across a range of disparate data sources). It also allows you to do Web Services calls using either SOAP/JMS or SOAP/HTTP.

An example of this solution is an enterprise implementing WebSphere Message Broker to establish an extensible enterprise integration backbone to integrate existing applications. The advantages are:

1. Reduces risk by insulating applications from the complexity of highly heterogeneous environments.
2. Increases consistency across various systems in the enterprise by integrating several back end applications using a variety of formats and connection methods.
3. Correlates individual messages to detect situations mandated by the business – such as fraud detection.

Scenario 3: WebSphere ESB and WebSphere Message Broker



Scenario 3 uses both WebSphere ESB and WebSphere Message Broker.

In this scenario, Retail Stores utilize WebSphere ESB and WebSphere Message Broker together to implement an Enterprise Service Bus.

At each store location, WebSphere ESB links multiple J2EE applications in addition to linking to Point of Sale terminals through SOAP/HTTP.

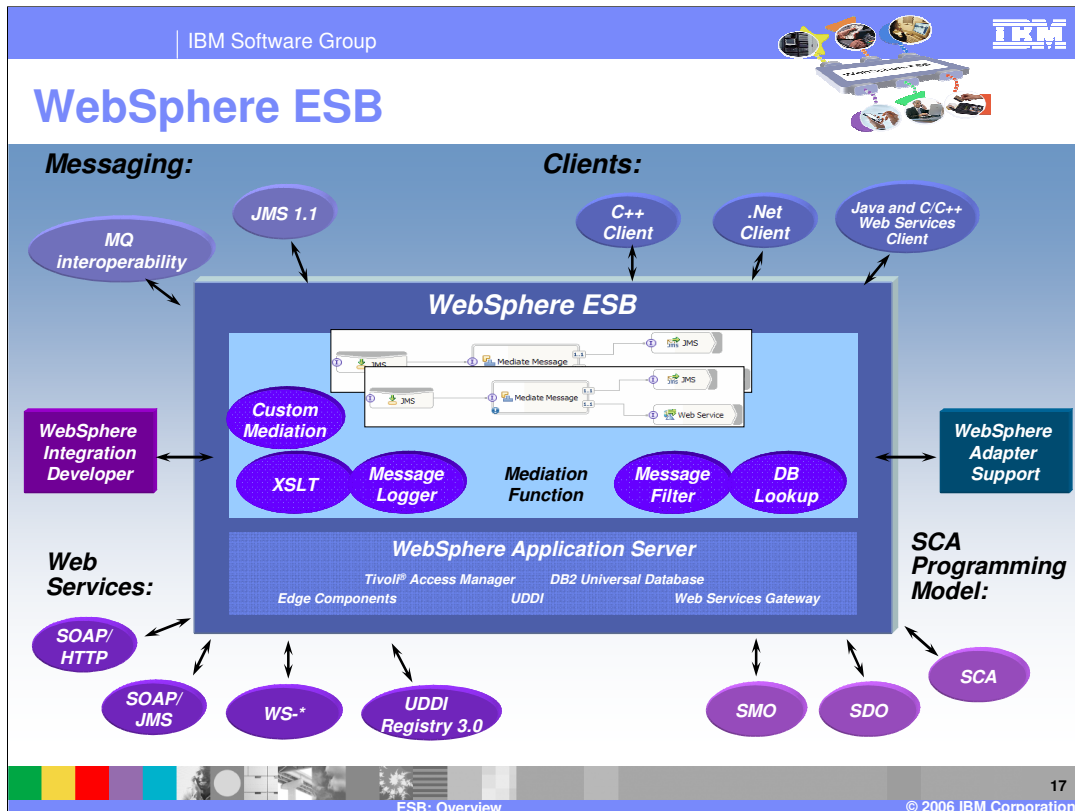
At the corporate data center, WebSphere Message Broker serves as the connectivity hub for distributing information to the store locations as needed. WebSphere Message Broker also transforms messages between the various packaged applications and mainframe systems as well as masking that back end complexity for easy store integration.

Section

- Strategy and the role of ESB
- IBM ESB Products
- **WebSphere ESB and the WebSphere Family**



The next section covers the WebSphere ESB product and its relation with the WebSphere product family.



This picture provides the most complete overview of what WebSphere ESB is - what is in the box and the features and functions associated with the product.

At the center of the picture, you can see that WebSphere ESB is built on top of WebSphere Application Server Network Deployment. WebSphere ESB, then, inherits all the WebSphere Application Server qualities of service, with its world class clustering, failover, scalability, and security. Along with these features, WebSphere ESB also includes a number of key WebSphere Application Server Network Deployment features, including UDDI as a services registry, the Web services Gateway, Tivoli Access Manager, DB2, and Edge Components.

Moving outward you can see the value that WebSphere ESB adds to the market-leading application server. First, there are the mediation functions, which can be used together to create integration logic for “smart” connectivity. WebSphere ESB offers JMS messaging and MQ interoperability for messaging, as well as a comprehensive clients package for connectivity. The SCA Programming Model enables ease of use, while Web Services support allows you to incorporate leading edge capabilities. On the sides of the picture you can see WebSphere Integration Developer, an easy to use tool that supports WebSphere ESB, as well as WebSphere Adapters, which includes new JCA adapters.

WebSphere ESB supports all the Web Service specifications, denoted by WS*, present in the WebSphere Application Server.



WebSphere ESB - Overview

- Provide tools and runtime support for Enterprise Service Bus with the new Mediation Service Application that intercept and modify messages between service requester and the service provider
- Simple to develop, build, test, deploy and manage Mediation Service application in WebSphere Integration Developer
 - ▶ Integrated, interactive and visual development experience requires minimal programming skills
- Seamless integration with WebSphere platform
 - ▶ Leverages WebSphere qualities of service:
 - ▶ Easily extends to leverage WebSphere Process Server as needs dictate
 - ▶ Integrates with IBM Tivoli security and systems management offerings
 - ▶ CEI events from WebSphere ESB monitored using WebSphere Monitor
 - ▶ Extensive WebSphere Adapter support, including new JCA-based adapters

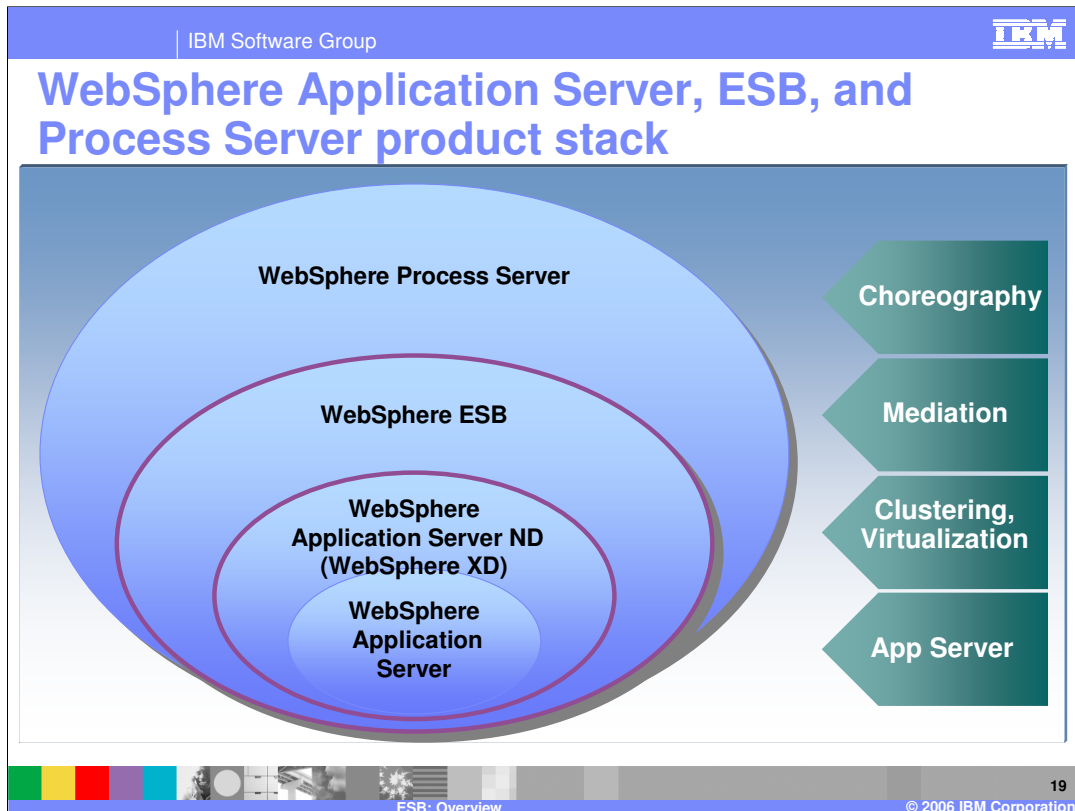


WebSphere ESB delivers an Enterprise Service Bus infrastructure to enable connecting applications that have standards based interfaces in order to power your SOA. WebSphere ESB provides the mechanism to process the request and the response message from the service requestors and providers connecting to the bus.

WebSphere Integration Developer is used to develop the ESB mediation applications. It is easy to use and require minimal programming skills. Knowledge of Java is not needed to develop the applications. -- it is integrated, interactive and provides a visual development experience. It is simple to develop, build, test, deploy and manage services components.

WebSphere ESB provides seamless integration with the WebSphere platform with the ability to easily move up the stack to solve more complex business problems with WebSphere Process Server, which is built on WebSphere ESB. This allows customers to easily extend to leverage WebSphere Process Server as needs dictate. Common tools and administration means the move from WebSphere ESB to WebSphere Process Server is painless. WebSphere ESB is built on the WebSphere Application Server: a world-class J2EE foundation providing industry-leading levels of availability, scalability, security and performance. Also, WebSphere ESB can tightly integrate with IBM Tivoli security and systems management offerings.

This cost effective solution further saves time and development costs by providing pre-built mediations such as XML transformation, content based routing and message logging.



The product stack of the WebSphere servers is shown here. The stack is built as follows: WebSphere Application Server provides the base J2EE hosting environment. WebSphere Application Server Network Deployment provides clustering support for scalability and fail over of the application servers. WebSphere ESB stacked on top of the WebSphere Application Server provides the ESB functionality to be able to mediate message flows between service requestors and providers. Stacked on top of WebSphere ESB is the WebSphere Process Server providing Choreography functionality for business process management applications.

WebSphere XD v.6.0.1 has been enhanced to be used to provide virtualization function with the WebSphere Process Server. WebSphere XD could be used in place of WebSphere Network Deployment to provide clustering and virtualization.

Built on this stackable architecture, each product makes use of all the functionality of the products on which they are built, allowing easy extension of capabilities as needed.

IBM Software Group IBM

WebSphere ESB and WebSphere Application Server

New Product

WebSphere ESB:
Mediation layer builds on WebSphere Application Server foundation to provide intelligent connectivity

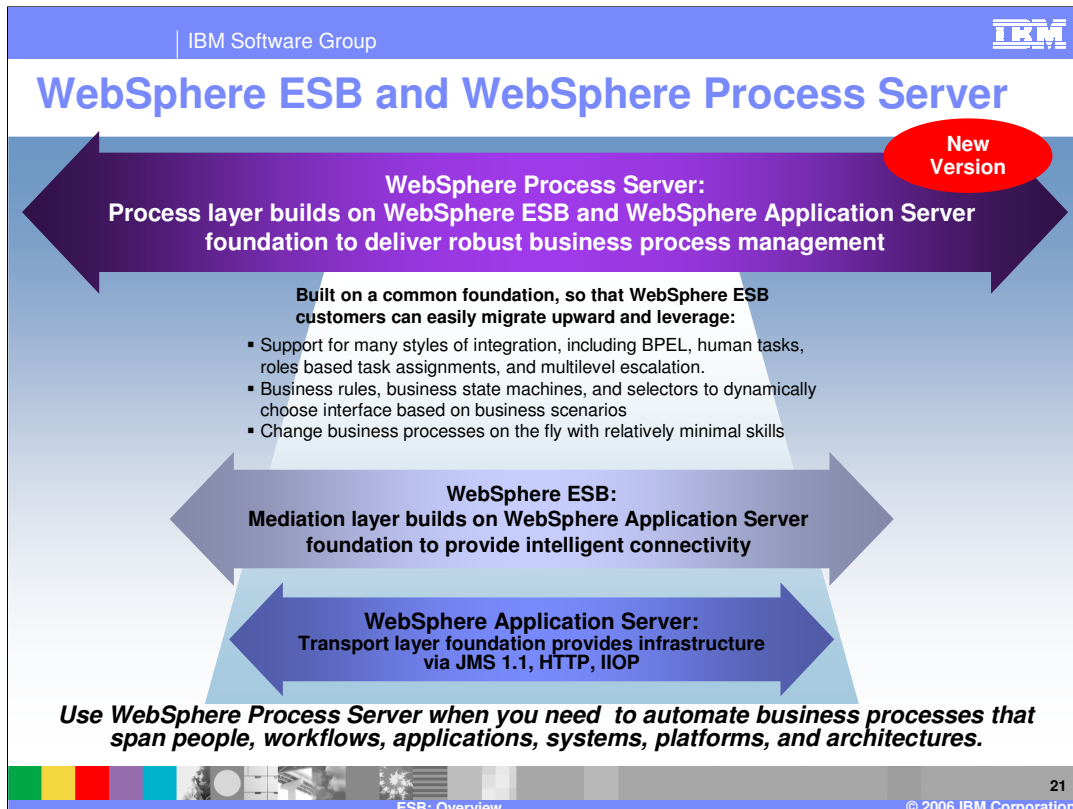
- Pre-built mediation functions and easy to use tools enable rapid construction and implementation of a Enterprise Service Bus
- Leverage visual tools to exploit supplied pre-built mediation functions

WebSphere Application Server:
Transport layer foundation provides infrastructure via JMS 1.1, HTTP, IIOP

Use WebSphere ESB for rapid time to integration value that doesn't require extensive Java or J2EE skills

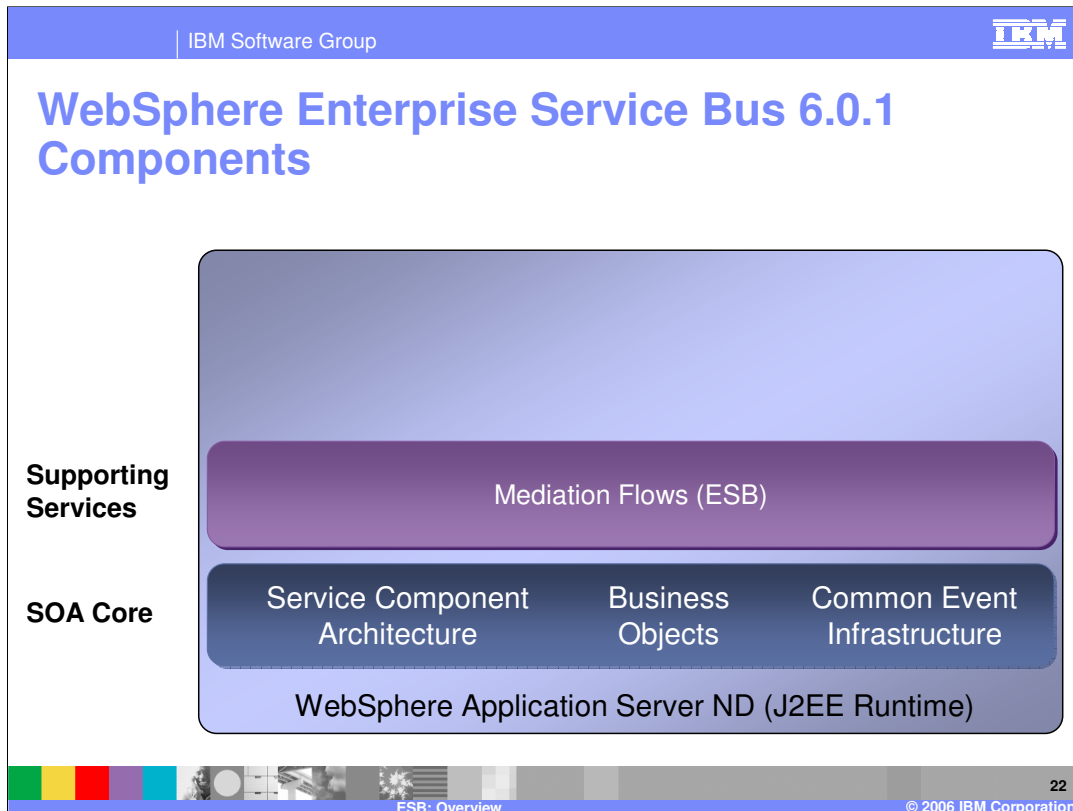
ESB: Overview
© 2006 IBM Corporation

WebSphere ESB is the mediation layer that runs on top of the transport layer within WebSphere Application Server. As such, WebSphere ESB provides the following value-add on top of WebSphere Application Server. It provides pre-built mediation functions and easy to use tools to enable rapid construction and implementation of an Enterprise Service Bus. It leverages visual tools in WebSphere Integration Developer to exploit supplied pre-built mediation functions.



Just as WebSphere ESB is built on top of WebSphere Application Server, WebSphere Process Server provides a process layer on top of the WebSphere ESB mediation layer. Because the two are built on a common foundation, you can easily migrate upward and leverage support for all styles of integration, including human tasks, roles based task assignments, and multilevel escalation.

WebSphere Process Server provides integration of business applications by supporting many styles of integration, including BPEL and human tasks. Additionally it supports Business rules, business state machines, and selectors to dynamically choose interfaces based on business scenarios. This allows changing business processes on the fly with relatively minimal skills.



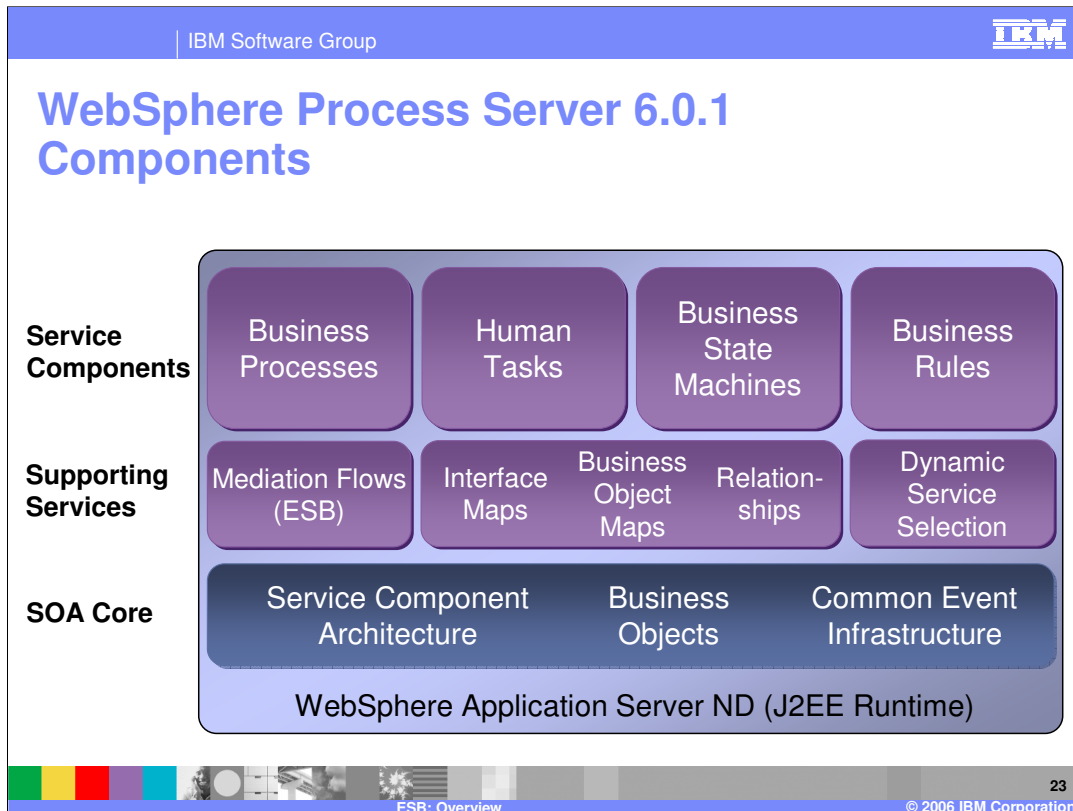
WebSphere ESB V6 is built on a robust J2EE Runtime provided by WebSphere Application Server. This runtime also offers Qualities of Service that WebSphere ESB exploits, namely clustering, failover, scalability, security, and a built-in messaging provider.

On top of this infrastructure there is a layer called the SOA Core. IBM uses this layer in WebSphere ESB V6 and other software offerings. To do integration in an SOA properly it is necessary to have a single invocation model and a single data model. Service Component Architecture (SCA) is this invocation model – every integration component is described through an interface. These services can then be assembled in a Component Assembly editor thus enabling a very flexible and encapsulated solution. Business Objects are the universal data description. They are being used as data going in and out of services – and are based on the Service Data Object (SDO) standard.

Also in the infrastructure is the Common Event Infrastructure – which is the foundation for monitoring applications. IBM uses this infrastructure throughout its product portfolio, and Monitoring Products from Tivoli as well as WebSphere Business Integration Monitor exploit it. The event definition (Common Business Event) is being standardized through the OASIS standards body – so that other companies as well as customers can use the same infrastructure to monitor their environment.

On top of this SOA core, there are a number of components and services that can be used in an integration solution. WebSphere ESB contains the mediation flow services providing the ESB functionality used to process the message between different service requestors and providers.

This way, WebSphere ESB includes all the functions and services necessary to provide the ESB services in any given integration solution.



The WebSphere Process Server is built on the component stack of WebSphere ESB. All the components of WebSphere ESB are included in the WebSphere Process Server.

In addition, there are other supporting services and service components within WebSphere Process Server.

Transformation happens quite frequently in any given integration project. Besides the ESB mediation services, WebSphere Process Server supports the following kinds of mediations:

- Maps: These transform one kind of business object into another
- Relationships: Key management for same objects in different data stores (for example, a customer record has a different identifier in SAP than in Siebel)
- Interface Mediation: to translate between semantically identical but syntactically different services – for example, updateCustomer (Customer) to updateCustomerInSAP (SAPCustomer).
- Selector: This allows the invocation of a different component (with the same interface) based on business rules.

WebSphere Process Server includes the business process engine that WebSphere Business Integration Server Foundation included. It has been updated to support the full WSBPEL 1.1 specification as well as most of the WSBPEL 2.0 draft items.

Human Tasks have been greatly improved over WebSphere MQ Workflow and WebSphere Business Integration Server Foundation. There is now a separate component, the Human Task Manager, that deals with Human Tasks. As far as a business process is concerned, a human task is just another SCA component. The SCA architecture allows replacing a human task (for approval for example) with a business rule for example without changing anything else in the entire solution. This is an extremely powerful and valuable benefit.

Business State Machines are another way of modeling a business process. There are some processes that are highly event driven – for example, an order could be cancelled at any point during the order process. It has been very difficult to model these kinds of processes in a “traditional, sequential” business process. Therefore there is another service where UML state machine diagrams can be used to describe such event driven business processes. Note that under the covers these are still being implemented as WSBPEL flows.

WebSphere Process Server also includes Business Rules capabilities. This includes Rule Sets (If/Then rules) as well as Decision tables. There is also a web client where the parameters of these rules can be changed by a business user

WebSphere Process Server includes all the functions and services necessary in any given integration solution.

Mediation Components in WebSphere ESB vs. Mediation Handlers in WebSphere Application Server Service Integration Bus

- Mediation Components in WebSphere ESB: Service Intermediaries
 - ▶ Operate on interactions between service endpoints (requesters and providers)
 - ▶ Administered as part of the WebSphere ESB and WebSphere Process Server
 - ▶ Created using visual tools, exploiting supplied mediation functions
 - Even custom function can be created with visual tools assist
 - ▶ Access to a binding-specific header data
 - SOAP headers
 - JMS headers
- Mediation Handlers in WebSphere Application Server Service Integration Bus: Message Handlers
 - ▶ Operate on messages traversing the WebSphere Application Server Service Integration bus
 - ▶ Administered as part of the WebSphere Application Server Service Integration Bus
 - ▶ Created by implementing Java programs
 - ▶ Allow access to the full WebSphere messaging header information
 - ▶ Low level implementation compared to Mediation Services provided by WebSphere ESB and Process Server

You can run WebSphere Application Server V6 based mediations in WebSphere ESB unchanged. However, WebSphere ESB provides a new mediation framework based on SCA.

The new mediation framework uses the new ESB enhancements to the WebSphere Integration Developer tool and therefore cannot be used to make changes to these Service Integration Bus mediations message handlers. You can continue to use WebSphere Application Server V6 mediation assets alongside new mediation assets you create for WebSphere ESB.

Section

Summary and References

Next section covers the Summary and References.

The Benefits of using WebSphere for your ESB solution

- ✓ The most advanced, and complete, SOA platform
- ✓ Best-of-breed services AND best-of-breed universal connectivity & transformation
- ✓ Advanced capabilities
- ✓ Proven technology
- ✓ Easy to use tools



In summary, there are several benefits for using WebSphere for your ESB solution:

1. Enables IT to be more responsive and flexible to the changing demands of the business -- easily add, remove, and change applications as required.
2. Reduces the time, cost, and risk of integration projects with a single interface for each application
3. Simplifies moving information between applications.
4. Makes timely use of any relevant, accurate information or data from across the organization.
5. Provides reliable integration of your production applications, whether packaged, in-house, or newly developed.

References

- Information Center for WebSphere Integration Developer V6.0.1, WebSphere Enterprise Service Bus V6.0.1 and WebSphere Process Server V6.0
 - ▶ <http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/index.jsp>
- Supported platforms and system requirements:
 - ▶ WebSphere Integration Developer V6.0
 - <http://www.ibm.com/software/integration/wid/sysreqs/>
 - ▶ WebSphere Process Server V6.0
 - <http://www.ibm.com/software/integration/wps/sysreqs/>
 - ▶ WebSphere Enterprise Service Bus V6.0.1
 - <http://www.ibm.com/software/integration/wsesb/sysreqs>



Listed here are the links to the Information Centers and the supported platforms and system requirements.

Trademarks, Copyrights, and Disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	CICS	IMS	MQSeries	Tivoli
IBM (logo)	Cloudscape	Informix	OS/390	WebSphere
e(logo)/business	DB2	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	Lotus	pSeries	zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

© Copyright International Business Machines Corporation 2005,2006. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.