IBM SOA ARCHITECT SUMMIT
LE 22 MAI 2008

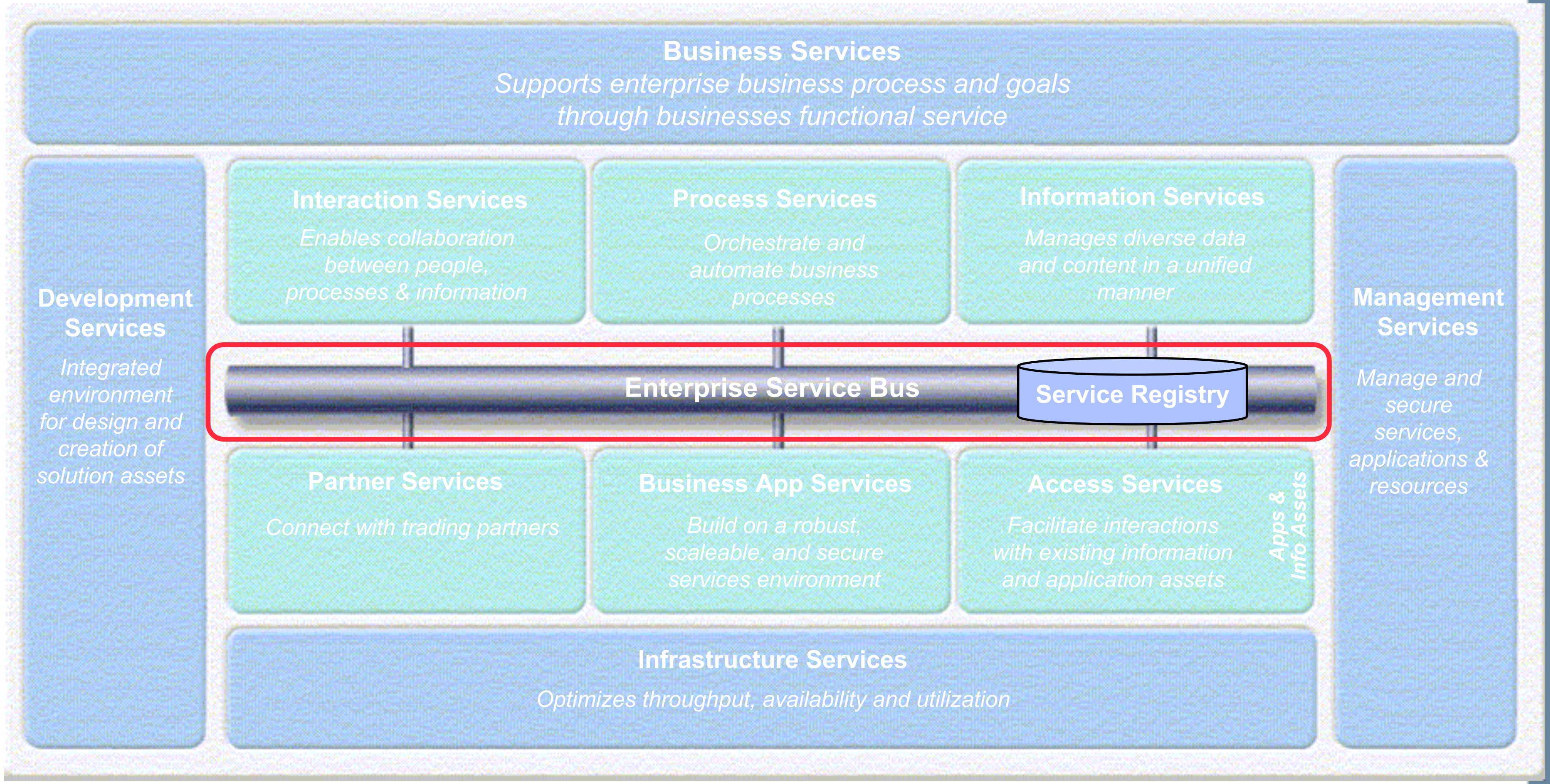
WebSphere ESB 6.1
Introduction

Rodolphe Lezennec

WebSphere Integration Solution Architect

IBM Software France
rodolphe.lezennec@fr.ibm.com

ESB in the SOA Foundation Reference Architecture


Business Services
Supports enterprise business process and goals through businesses functional service
Development Services
Integrated environment for design and creation of solution assets
Interaction Services
Enables collaboration between people, processes & information
Process Services
Orchestrate and automate business processes
Information Services
Manages diverse data and content in a unified manner
Management Services
Manage and secure services, applications & resources
Enterprise Service Bus
Service Registry
Partner Services
Connect with trading partners
Business App Services
Build on a robust, scaleable, and secure services environment
Access Services
Facilitate interactions with existing information and application assets
Apps & Info Assets
Infrastructure Services
Optimizes throughput, availability and utilization

Service Connectivity 1: Internal Connectivity

Business challenge

- Make real time stock information available between stores and headquarters
- Integrate disparate store systems

Solution

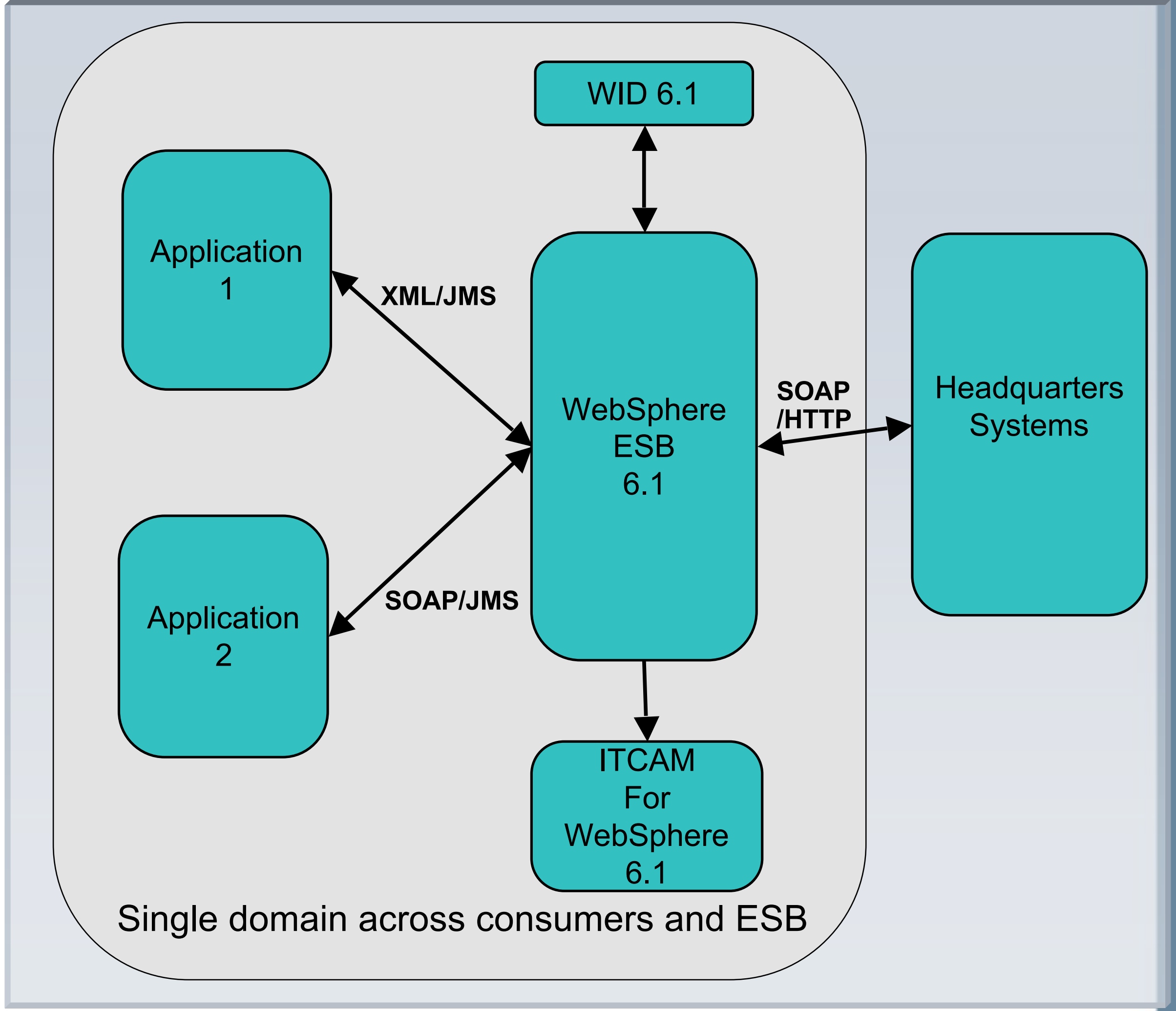
- Create standards based services and connect using an ESB
- Cost sensitive wrt to store systems

Implementation Details

- Applications communicate using standards based services
- ESB provides protocol and message transformation and routing

Products

• WebSphere Integration Developer 6.0.2
•WebSphere Enterprise Service Bus 6.0.2
•Tivoli Composite Application Monitor for WebSphere V6.1



Service Connectivity 2 - Adapting enterprise applications to Web services

Business challenge

- Provide web service access to functionality in SAP R/3 and in the future other EIS systems.

Solution

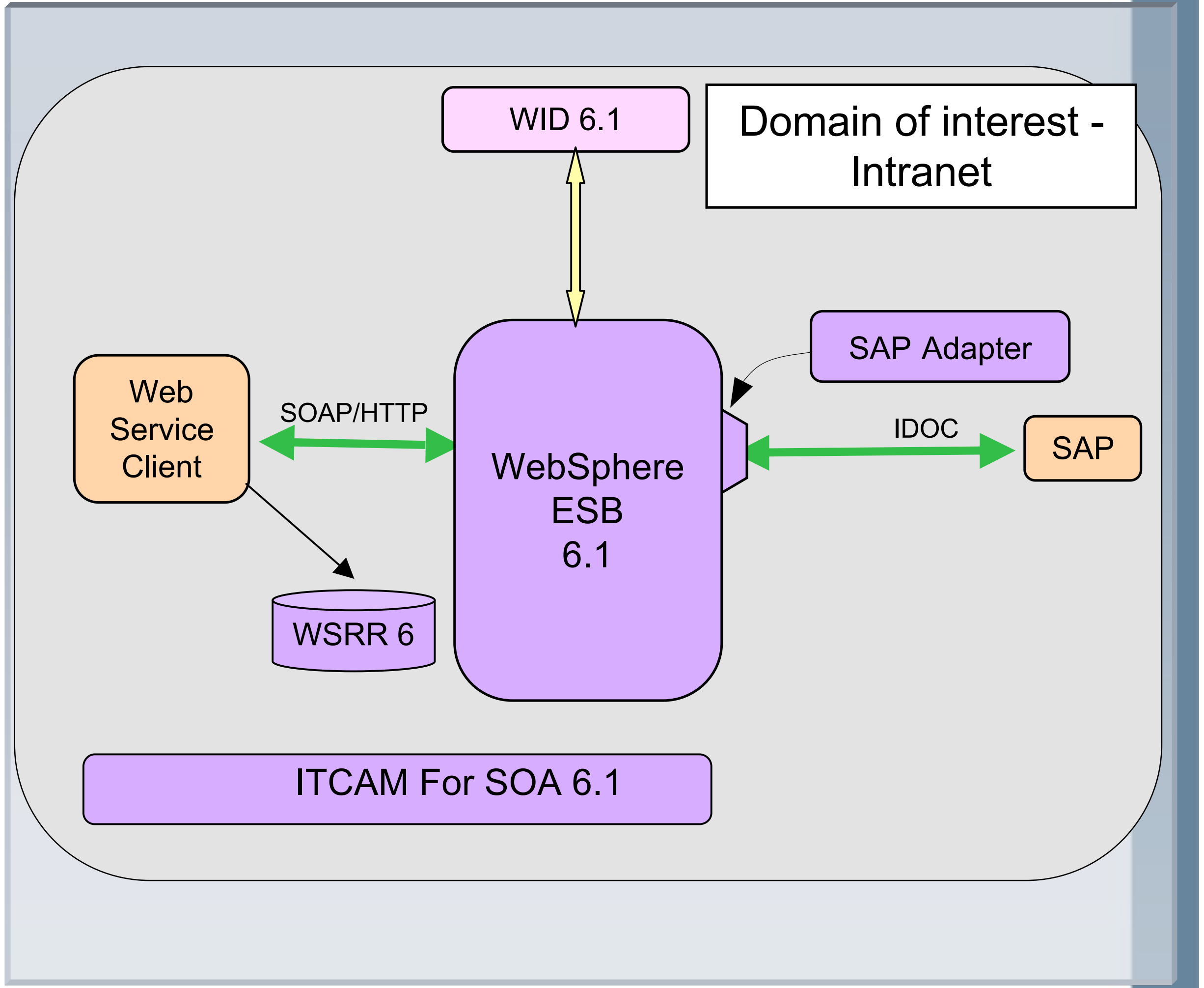
- Adapt between SAP system and web services using an ESB.

Implementation Details

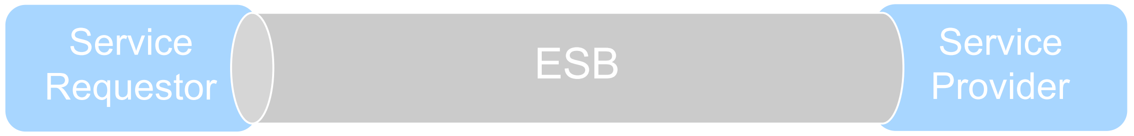
- The SAP adapter provides access to SAP as a BO. WebSphere ESB converts the message format and exposes services as SOAP/HTTP.
- Clients lookup the service endpoints of the ESB in WSRR.

Products

- WebSphere Integration Developer 6.0.2
- WebSphere Enterprise Service Bus 6.0.2
- Tivoli Composite Application Monitor for WebSphere V6.1
- WebSphere Service Registry and Repository 6.0



Core Principles of the ESB Architectural Pattern

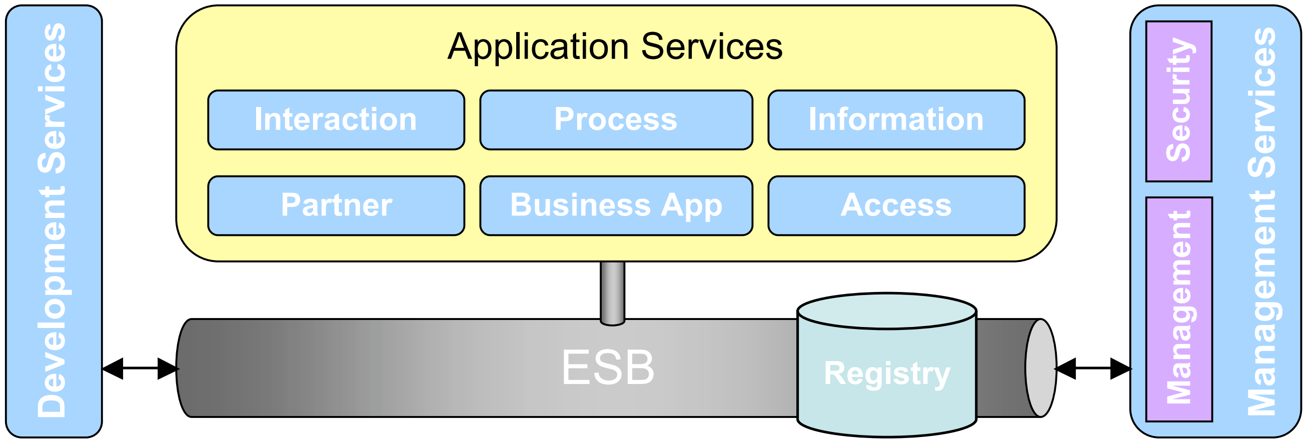


- ESB inter-connects requestor and provider
  - Interactions are *decoupled*
  - Supports key SOA principle – *separation of concerns*

- ESB provides ***Service Virtualization*** of
  - *Identity* via routing
  - *Protocol* via conversion
  - *Interface* via transformation

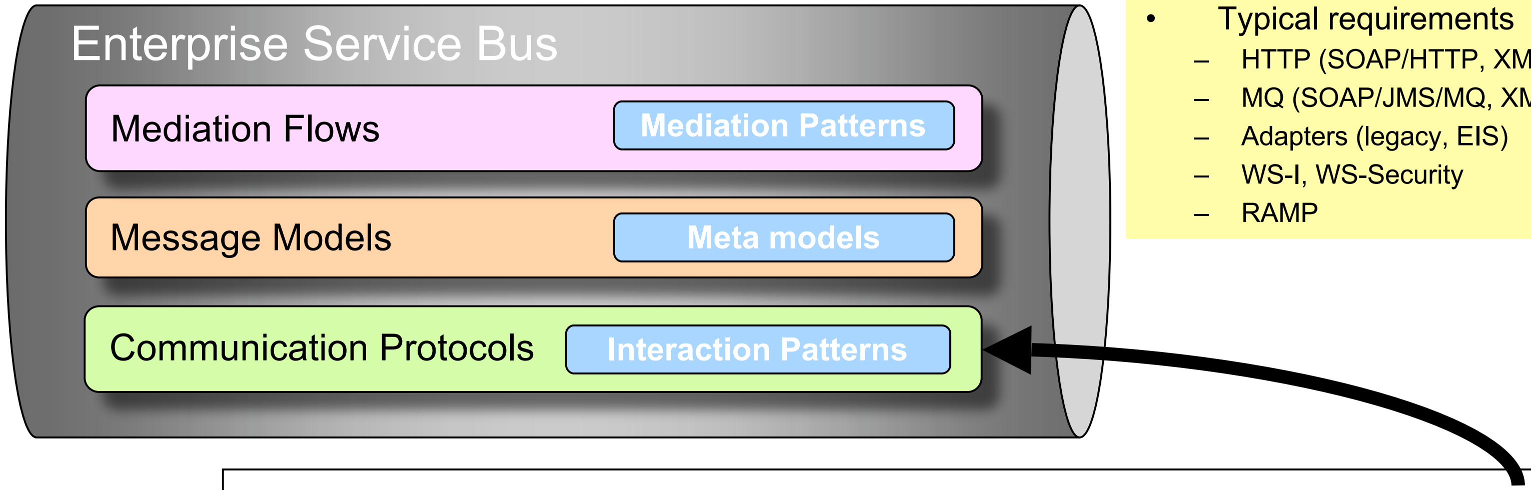
- ESB also enables ***Aspect Oriented Connectivity***
  - Security
  - Management
  - Logging
  - Auditing
  - …

An ESB-centric view of the Logical Model



- Outside ESB
  - Business Logic (Application Services)
    - ESB ***does*** contain integration logic or connectivity logic
    - Criteria: semantics versus syntax; aspects
- Loosely coupled to ESB
  - Security and Management
    - Policy Decision Point outside the ESB
    - ESB can be Policy Enforcement Point
- Tightly coupled to ESB
  - Service Registry
    - Registry a Policy Decision Point for ESB
    - ESB a Policy Enforcement Point for Registry
    - But, Registry has a broader scope in SOA
- Tooling required for ESB
  - Development
  - Administration
  - Configures ESB via Service Registry

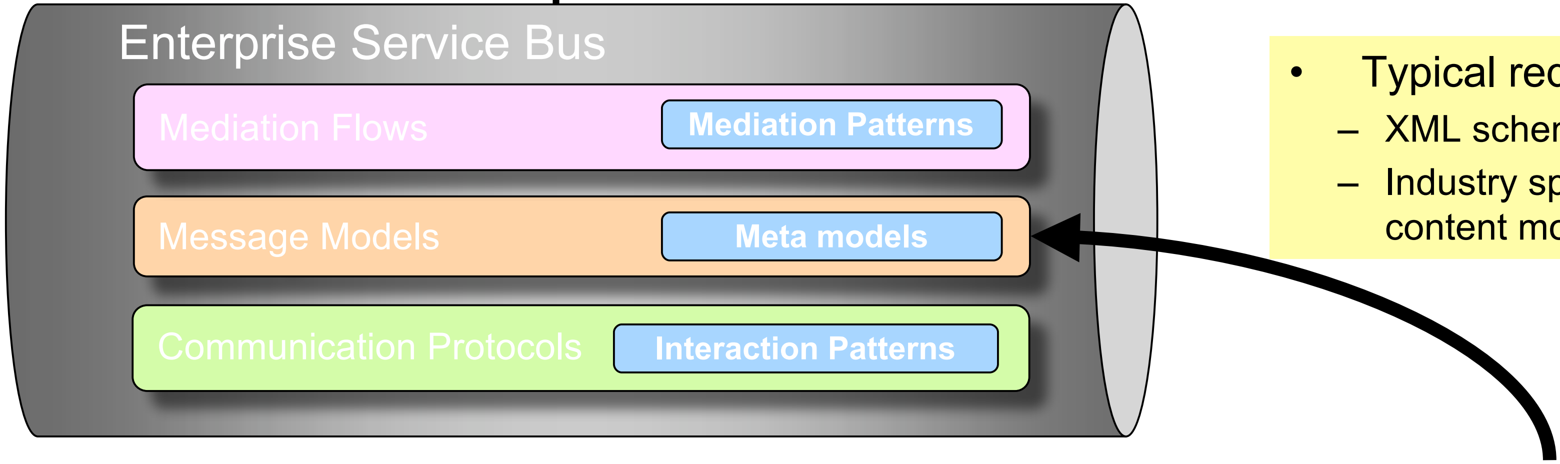
Expanded view of the ESB



- Typical requirements
  - HTTP (SOAP/HTTP, XML/HTTP)
  - MQ (SOAP/JMS/MQ, XML/MQ, text/MQ, …)
  - Adapters (legacy, EIS)
  - WS-I, WS-Security
  - RAMP

- Communication Protocols
  - Supply basic connectivity to requesters and providers
    - Impact QoS (e.g., reliable delivery, transactions)
  - Supply inherent *Interaction Patterns* (e.g., request/reply, one-way, pub/sub)
- An ESB leverages underlying communication fabrics of SOA infrastructure
  - ESB provides *on-ramps* and *off-ramps*
- Standards are important

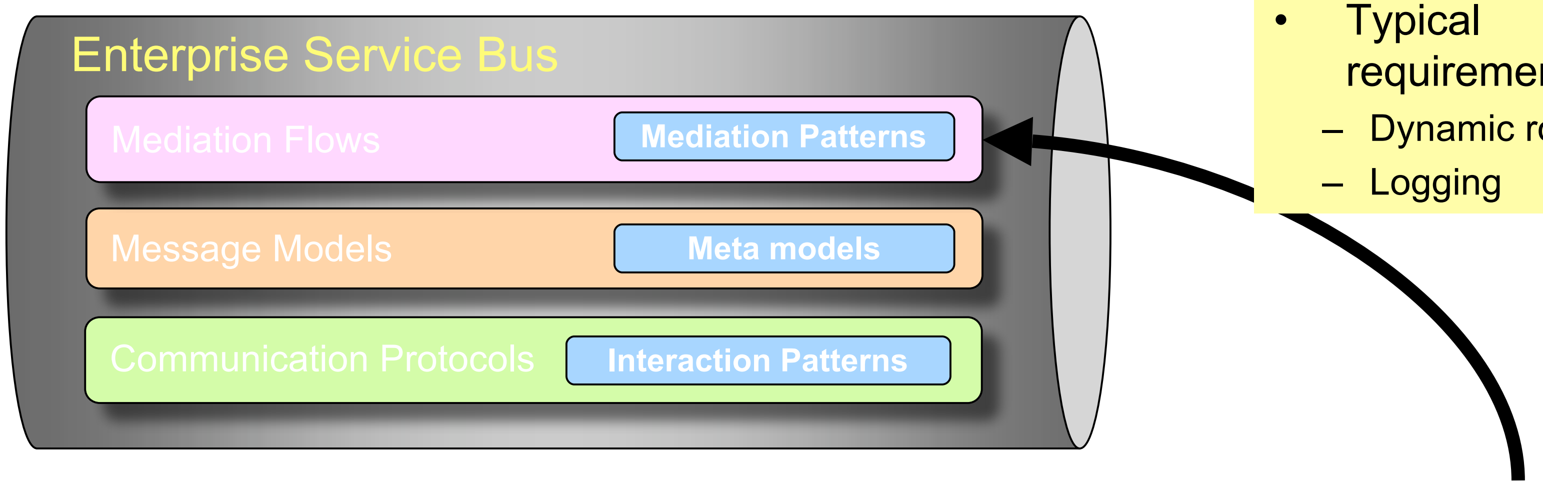
Expanded view of the ESB



- Typical requirements
  - XML schema definition
  - Industry specific content models

- Message Models
  - Describe message content exchanged with requesters and providers
    - For example, XML schema
  - Based on Meta-models
    - Fundamental means of describing messages
    - For example, XML Schema language
- An ESB supports one or more message meta-models
- An ESB supports multiple message content models
  - Can include industry standard models as well as enterprise specific models
  - Can include weakly-typed models

Expanded view of the ESB

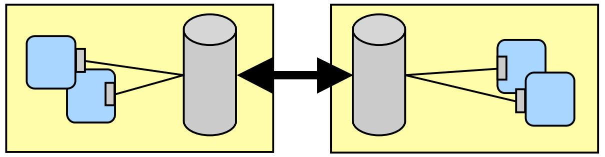


- Typical requirements
  - Dynamic routing
  - Logging

- Mediation Flows
  - Process messages exchanged between requester and provider via ESB
    - Large grained
    - Moderately reusable
    - Constructed from *Mediation Patterns*
  - Mediation Patterns define processing “steps” of a mediation flow
    - Small to middle grained
    - Highly reusable
    - ESB products include pre-built “mediation primitives”

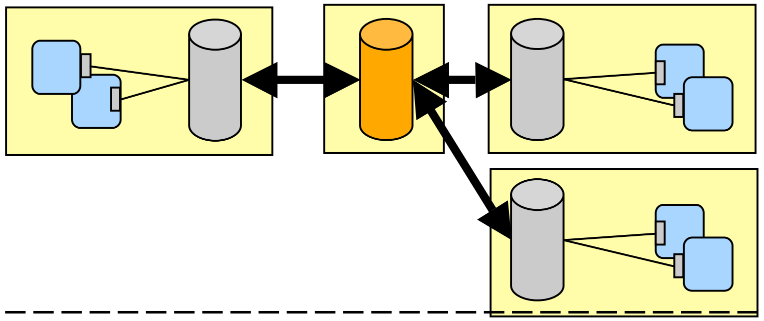
ESB Roles – ESB Integration Topology Patterns

Direct



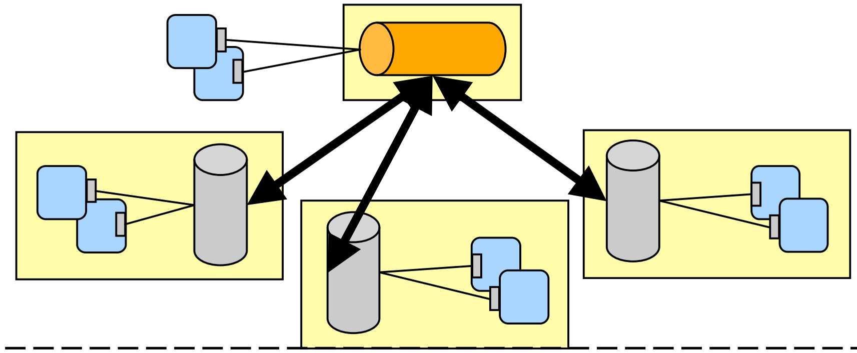
Multiple namespaces, administration domains; namespace mapping in each ESB; services are likely to be applicable throughout the enterprise

Brokered



Multiple namespaces, administration domains; namespace mapping in Brokered ESB Gateway facilitates service interaction; subset of services applicable throughout the enterprise

Federated

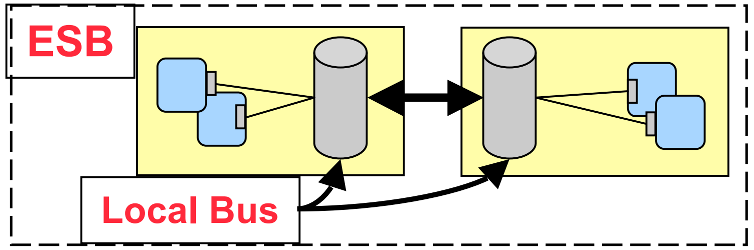


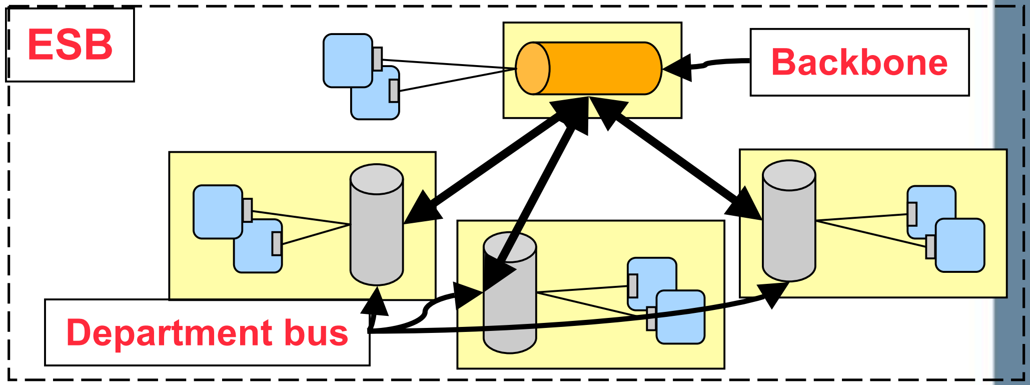
Multiple namespaces, administration domains; namespace mapping in Federated ESB facilitates service interaction with multiple implementations; subset of services applicable throughout the enterprise

NOTE: Adapted from Patterns for eBusiness

Topology patterns – Emerging view

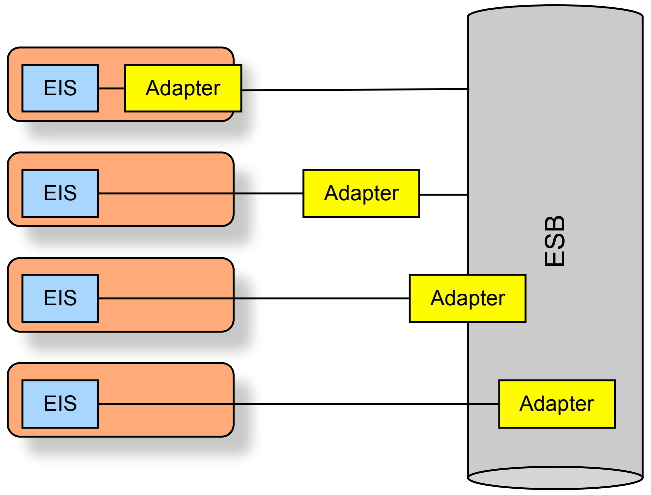
- “All” customers use some “topology pattern” variant (i.e., more than one “ESB role”)
  - For compartmentalization of “domains,” e.g.
    - Geographic locations
    - Departments
    - Stores
    - Business function
- Many think of the topology pattern itself as the ESB
  - New adjectives used for specific roles, e.g.,
    - Local (service) bus
    - Departmental (service) bus
    - Backbone (service bus)





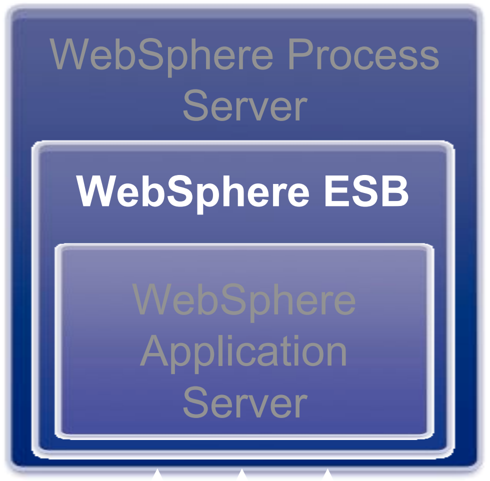
Adapters for Integrating Existing Systems

- An ESB must allow access to existing Enterprise Information Systems
- Adapters typically used, and may or may not be part of the ESB
  - Technology
  - Application
  - Legacy
- The following are the placement options for adapters, based on domain where adapter configuration managed:
  - Outside of the ESB, and inside the EIS domain
  - Outside the ESB, and the EIS domain
  - On the boundary of the ESB
  - Inside the ESB
- Two aspects to adapters
  - Communication protocol
  - Message format



WebSphere ESB

Leverages WebSphere Application Server for an integrated SOA platform



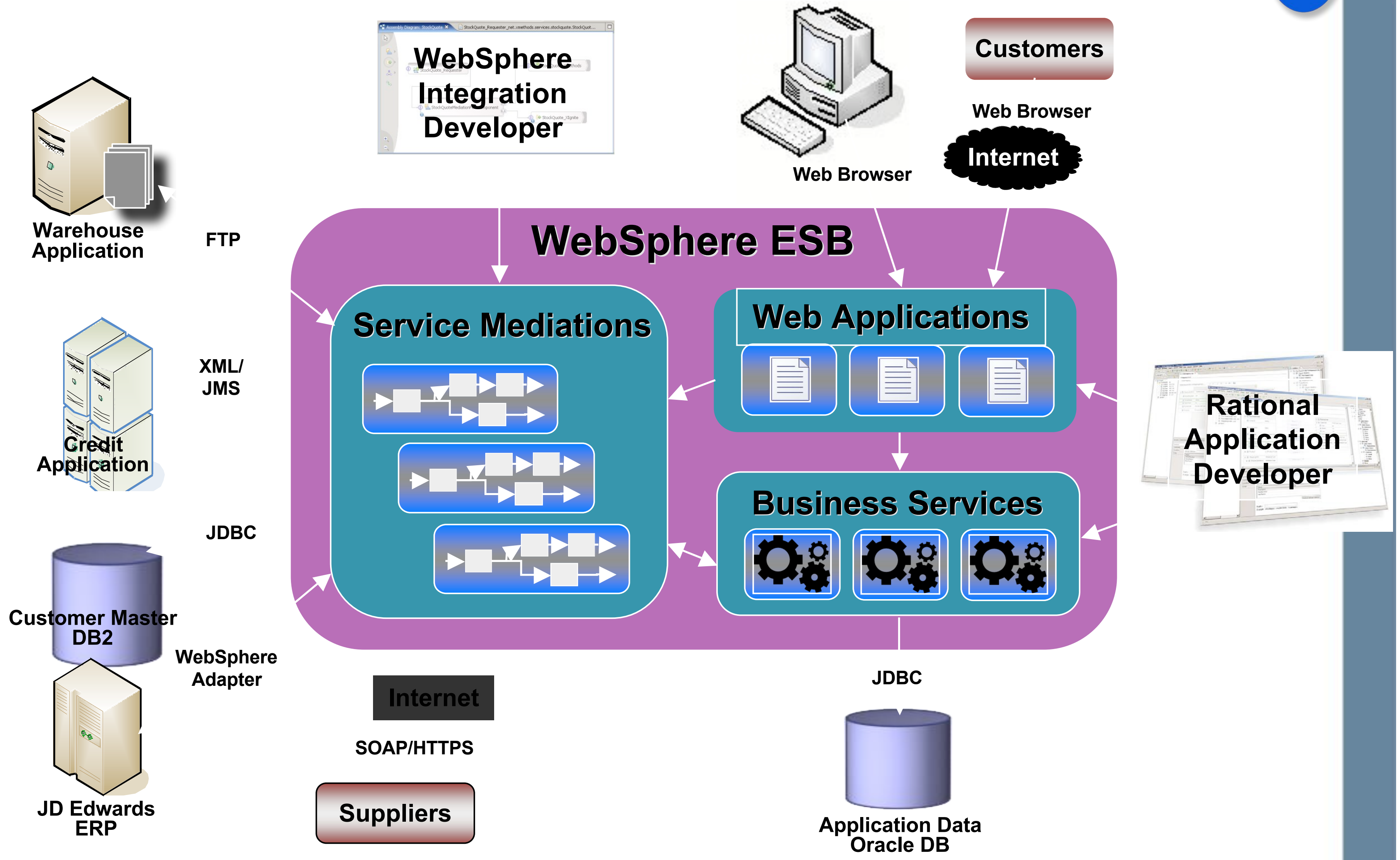
- Leverages the industry-leading WebSphere Application Server
- Provides service-oriented integration with first-class web services connectivity, JMS messaging, and pre-built mediation function
- Built on proven Java Enterprise standards, and providing leadership in SOA standards
- WebSphere Integration Developer provides an easy to use, visual integrated development environment

Java Enterprise/SOA standards
J2EE, JMS, HTTP, SOAP, UDDI, XML, WSDL, BPEL, SCA. SDO

- ➢ *Integrated solution for both service mediation and service hosting*
- ➢ *Integrates seamlessly with WebSphere platform and is easily extended to WebSphere Process Server for process orchestration and BPM*
- ➢ *Delivers business-critical qualities of service of WebSphere Application Server Network Deployment*

WebSphere ESB

Scenario – Composite application service hosting and mediation

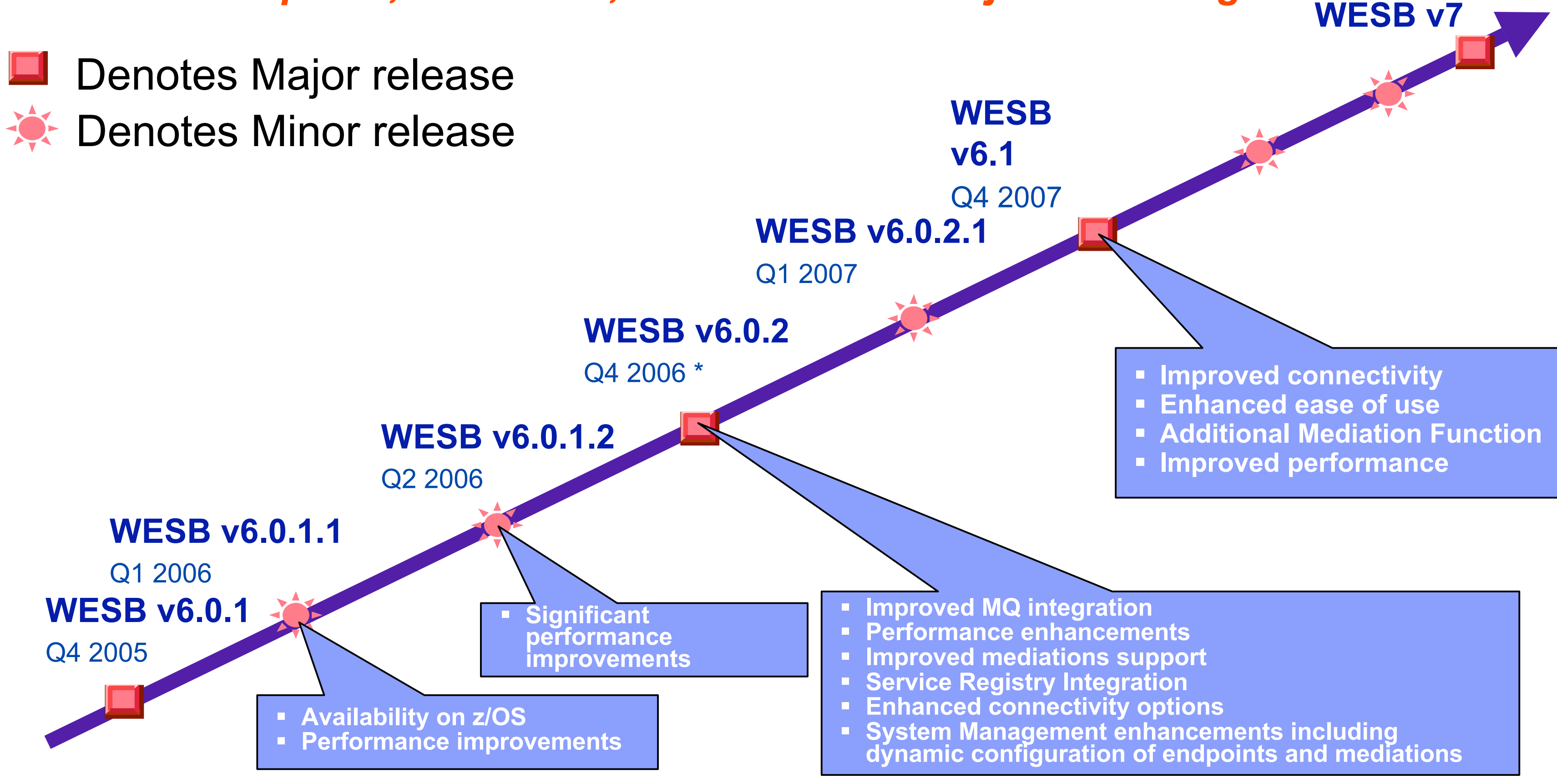


Examples for When to Use WebSphere ESB

- Adding ESB capabilities to a WAS environment
  - Adding support for mediated interactions to J2EE-based application environment, focussing on service-oriented interactions based on state-of-art SOA programming model centred around SCA, SDO
- Using ESB capabilities in context of BPM
  - Deep integration with WebSphere Process Server and BPM stack enabling separation of concerns between business process designers and service implementers
- Entry-level ESB for standards-based endpoints
  - ESB-in-a-box that has on-ramps and mediation capabilities needed to implement basic SOA connectivity-focussed scenarios

WebSphere ESB Product Roadmap

IBM's plans, directions, and intent are subject to change or withdrawal


🟥 Denotes Major release
✺ Denotes Minor release
WESB v7
WESB v6.1
Q4 2007
WESB v6.0.2.1
Q1 2007
WESB v6.0.2
Q4 2006 *
WESB v6.0.1.2
Q2 2006
WESB v6.0.1.1
Q1 2006
WESB v6.0.1
Q4 2005
▪ Improved connectivity
▪ Enhanced ease of use
▪ Additional Mediation Function
▪ Improved performance
▪ Significant performance improvements
▪ Improved MQ integration
▪ Performance enhancements
▪ Improved mediations support
▪ Service Registry Integration
▪ Enhanced connectivity options
▪ System Management enhancements including dynamic configuration of endpoints and mediations
▪ Availability on z/OS
▪ Performance improvements

** All releases have included updates to WebSphere Integration Developer*

WebSphere ESB v6.1 Overview

Consumability & Productivity

- Maintain user changes to the J2EE Deployment Descriptor
- Pattern-based configuration
- Improved deployment to the Unit Test Environment
- Common install/upgrade mechanism for WID and UTE
- Performance enhancements

Extended Connectivity and Interoperability

- HTTP SCA import/export binding
- WebSphere TX for Data binding

Enhanced Mediation and Transformation

- New BO Mapper primitive
- New primitives for splitting and aggregating messages
- Enhancements to Logger Primitive
- Support Retry in the flow programming model
- Enhance Custom Primitive

Continuing Support for Standards

- WS-Notification
- Java 5 support
- WS-I Basic Security Profile

Mission Critical QoS

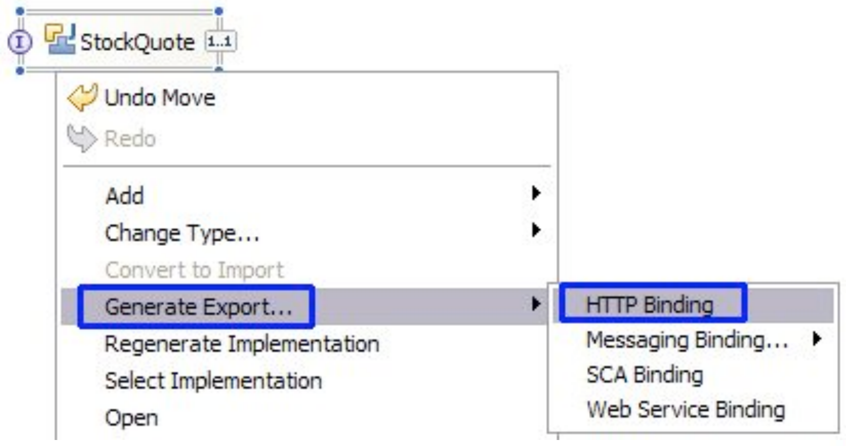
- WAS 6.1 based runtime
- Enhanced support for WAS XD
- z/OS 1.6+, exploiting WAS z/Os 64-bit
- Enhanced exploitation of ND features

Generic JMS binding: Overview

- **JMS 1.1 providers**
  - Oracle AQ (tested by IBM)
  - TIBCO, SonicMQ, WebMethods, BEA WebLogic (Not tested by IBM)
  - WebSphere MQ (tested by IBM)
    - Used to test because MQ JMS provider meets implementation requirements
    - Does not imply users should use in place of MQ JMS SCA bindings
- **Compatible behavior to JMS and MQ JMS bindings**
  - Supports point-to-point and pub/sub styles
  - Same data binding and function selector implementations
  - Expose JMS headers
  - Correlation schemes and event sequencing supported
  - Security using authentication aliases
  - Obeys SCA qualifiers and programming model

HTTP Bindings – Supported Functions

- HTTP 1.0 and 1.1
- SSL over HTTP
- Synchronous Request/Response invocation
- Supports Binary, XML and SOAP payloads
  - Plus custom data bindings
- Endpoint based routing in Export
- Ability to modify the HTTP binding attributes in the runtime server



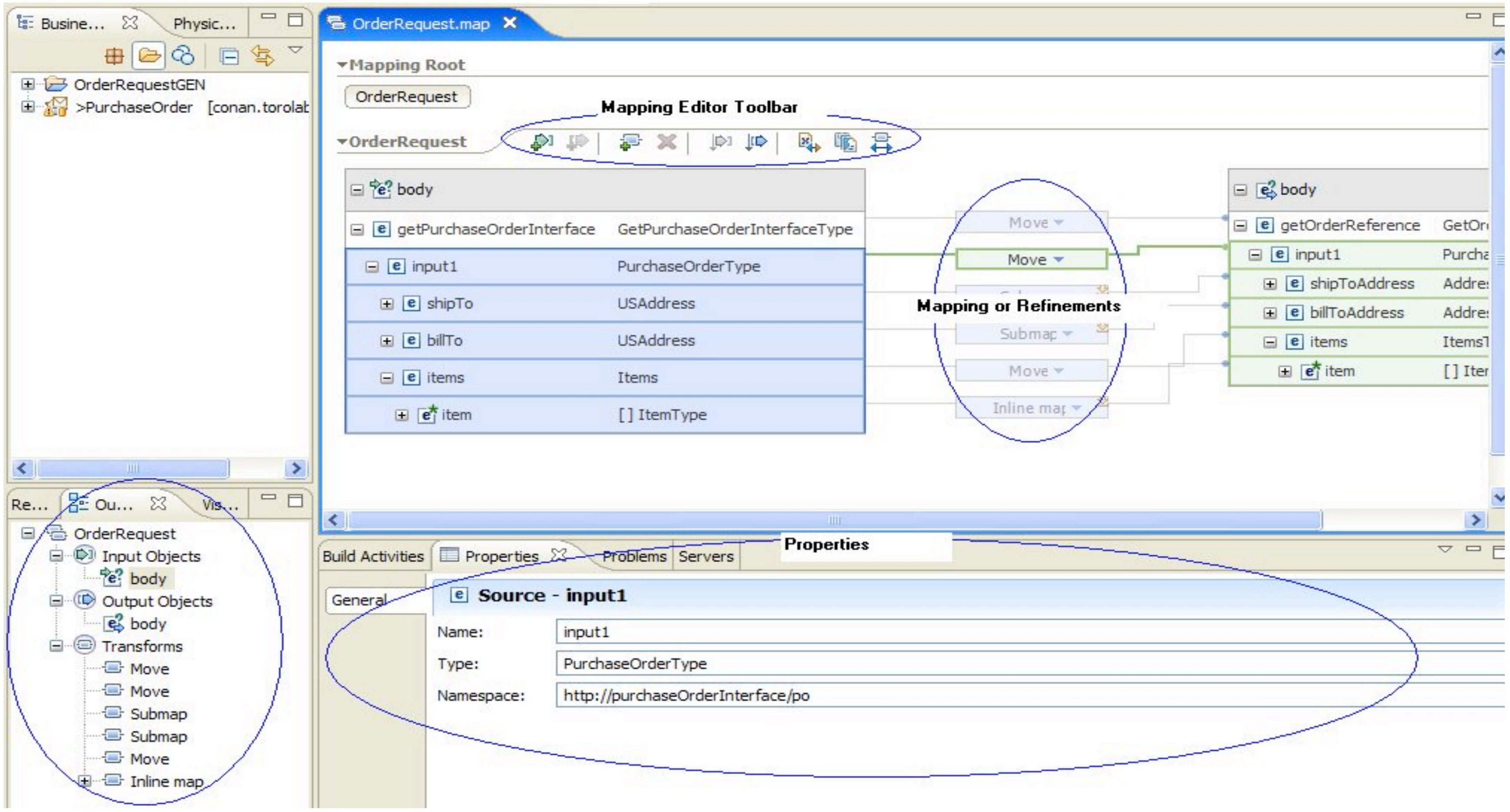
JMS binding enhancements

- **Update to JMS bindings**
  - More closely aligns with MQ JMS binding capabilities
    - Configurable correlation schemes for both Imports and Exports
      - Request message ID to correlation ID
      - Request correlation ID to correlation ID
    - Event sequencing for exports
      - Configurable setting for exports
      - Export delivers messages to SCA component in order received
      - Requires underlying JMS implementation to limit concurrency
        - » Set maximum concurrency to 1 on ActivationSpec for export's connection configuration
    - Configurable reply connection for imports
      - JMS bindings in V6.1 support reply connection configuration
        - » Exposes the previously hidden JCA 1.5 ActivationSpec
        - » Can be pre-configured or newly created

XSL transformation primitive enhancement

- **XSLT mapping editor in V6.0.2**
  - – Used old RAD mapping editor to define map between source and target SMO
  - – Several limitations
    - Worked with XML documents rather than schemas
    - Limited support for choice and repeating elements
    - Problems with complex XML schema structures
    - No support for anyType
    - No support for map reuse
- **XSLT mapping editor in V6.1**
  - – Uses new RAD7 XML mapping editor
    - Enhancements made to this editor to meet mediation requirements
  - – Resolves several of the limitations and enables map reuse
- **Maps from V6.0.x releases**
  - – XSLT from V6.0.x can continue to be used, including editing

XSLT - XML Mapping Editor



body	
getPurchaseOrderInterface	GetPurchaseOrderInterfaceType
input1	PurchaseOrderType
shipTo	USAddress
billTo	USAddress
items	Items
item	[] ItemType

body	
getOrderReference	GetOr
input1	Purcha
shipToAddress	Addre
billToAddress	Addre
items	ItemsT
item	[] Ite

```
OrderRequest
  Input Objects
    body
  Output Objects
    body
  Transforms
    Move
    Move
    Submap
    Submap
    Move
    Inline map
```

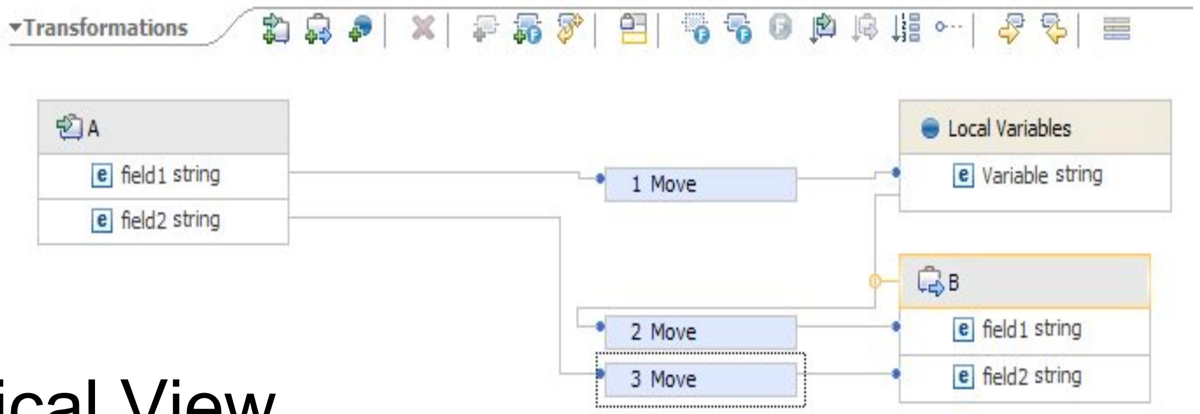
Source - input1

Name: input1
Type: PurchaseOrderType
Namespace: http://purchaseOrderInterface/po

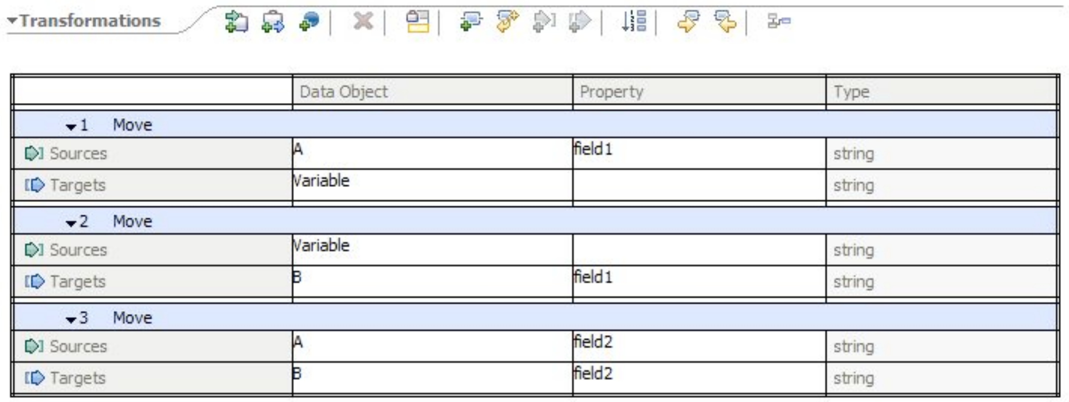
Business object map – Overview

- **Business object map and XSL transformation primitives**
  - Provide overlapping capabilities within a mediation flow
  - New XML editor for XSL transformations provides similar user interface and capabilities
- **Why use business object maps instead of XSL transformations?**
  - Mapping requires maintaining a relationship
  - Change summary needs to be maintained in a business graph
  - Configure event settings to raise CEI events
  - Utilize existing investment in business object maps
  - Business object map editor provides some unique capabilities
    - Variables
    - Fuzzy mapping

Business object map editor – Mode of operation



- Graphical View



	Data Object	Property	Type
1 Move			
Sources	A	field1	string
Targets	Variable		string
2 Move			
Sources	Variable		string
Targets	B	field1	string
3 Move			
Sources	A	field2	string
Targets	B	field2	string

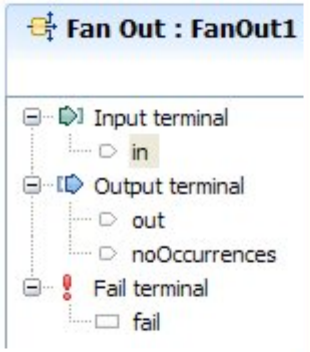
- Table View

New Mediation Capabilities

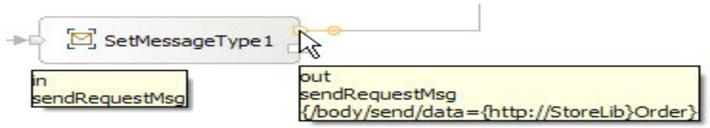
- **Invoking services from within a mediation flow**
  - –Service invoke primitive
  - –Services can be invoked from a request or response flow
  - –Synchronous and asynchronous invocation supported
- **Support retry when service returns a fault**
  - –Available with service invoke primitive and callout node
  - –Retry to:
    - Same service
    - Same service with a different endpoint
    - Different service

New Mediation Capabilities

- **Splitting and aggregating messages**
  - Fan out primitive - splitting
    - Iterate through repeating element
    - Same message on multiple flow paths
  - Fan in primitive – aggregating
    - Combine responses from multiple paths or iterations



- **Support for weak typing**
  - SMO supports use of weakly typed fields
  - Set message type primitive
    - Provides ability to downcast a weakly typed field to a more specific type



Enhanced custom mediation – Background

- **Ways to customize mediation primitives in V6.0.2**
  - Create a user-defined primitive
    - Need to deploy a plug-in to WID and jar to the server runtime
    - Fully customizable
      - Number of input and output terminals, processing logic, use of properties
    - Reusable – use in WID used like any of the built in primitives
    - Requires considerable work and knowledge to create
  - Specify code in a custom mediation primitive
    - Add custom code directly to primitive
      - Java or visual snippet code added to properties of primitive
      - Invocation of SCA reference to Java component or import on assembly diagram
    - Easy to implement and modify
    - Limited capabilities:
      - Exactly one input and one output terminal, output terminal only fired upon return from the code
      - Unable to specify properties
      - Not reusable

Enhanced custom mediation – Overview

- **Enhanced custom mediation for V6.1**
  - Enable several capabilities of user defined primitive in custom mediation
    - One or more input terminals
    - Zero or more output terminals
    - Control of when output terminals are fired
    - Use of user defined properties to configure processing
    - Reuse (via copy/paste)
  - Custom code options
    - Java or visual snippet code added to properties of primitive
    - Invoke option no longer supported (use service invoke primitive instead)
  - Migration from V6.0.2
    - Code from v6.0.2 will continue to work without migration
    - Conversion from invoke option to service invoke primitive requires manual migration
    - Quick fix provided to migrate java/visual implementation to V6.1

Easier installation and configuration

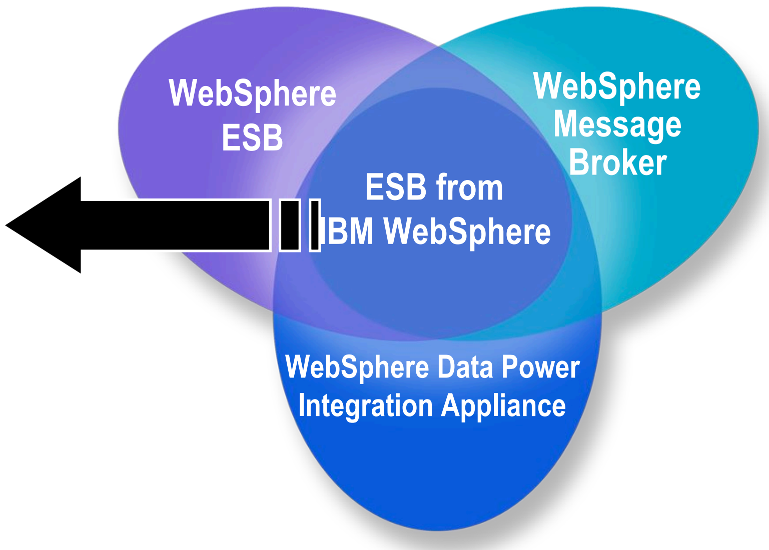
- **Guided installation**
  - –Typical installation using default selections and configurations
    - Stand-alone
    - Deployment manager
    - Custom
  - –Deployment environment installation
    - Single cluster
    - Remote Messaging
    - Remote Messaging and Support
    - Invoke option no longer supported (use service invoke primitive instead)

Performance: WebSphere Process Server, WebSphere Enterprise Service Bus and WebSphere Integration Developer v6.1

- **WebSphere Process Server and WebSphere Enterprise Service Bus v6.1 delivers substantial performance improvements relative to v6.0.2.0, including:**
  - Increased mediation throughput up to 100% across a variety of workloads and communications mechanisms
  - Increased long-running business process throughput by up to 40%
  - Support of objects and messages up to 80 MB in size
  - Time to install reduced by approximately 50%
  - Continued improvement in SMP and clustered scaling
- **WebSphere Integration Developer v6.1 offers a dramatically improved authoring experience with significant performance improvements relative to v6.0.2.0, including:**
  - Build memory use reduced by 50%
  - Build response time reduced by 45%
  - Application publish memory use reduced by 65%
  - Application publish response time reduced by 55%
- **Builds on significant improvements delivered in WAS 6.1**

Summary: ESB Trends and Directions
Common Patterns & Components across ESB Family

- Common terminology for ESB concepts
  - Mapping existing terminology to emerging Reference model
- Common patterns that are supported by all ESB runtimes
  - Support for templates in ESB tooling and enable mapping of templates implementing common patterns to different runtime implementations
- Common components & add-ons across the family
  - WSRR exploitation, WebSphere TX integration, Web Services support, Adapters, Event processing



All statements regarding IBM's plans are subject to change or withdrawal without notice.

Summary: ESB Trends and Directions

- Customer preferences in the ESB category continue to vary widely, and often differ between business units
  - Continue to invest in multiple ESB offerings
  - Continued focus on enhancing commonality and interoperability across ESB offerings
  - SOA hardware continues to gain momentum
- Increasingly, ESB decisions are not focused merely on feature/function of the ESB alone, but on the broader set of SOA and BPM capabilities for which the ESB is the foundation

All statements regarding IBM's plans are subject to change or withdrawal without notice.

Summary: ESB Trends and Directions

- As SOA adoption increases within the enterprise, ESB requirements and capabilities are trending towards unified management
  - Registry and repository for policy-based connectivity is becoming increasingly important to enable enhanced virtualization
  - Multiple ESBs in the enterprise is already becoming the norm – monitoring and managing across them and between them will become paramount
  - Registry and repository is becoming an essential tool to enable improved governance in ESB deployments
  - Interest increasing in Complex Event Processing for the ESB, limited to specific verticals

धन्यवाद
Hindi

多謝
Traditional Chinese

ขอบคุณ
Thai

Спасибо
Russian

Gracias
Spanish

Thank You
English

شكراً
Arabic

Merci
French

Obrigado
Brazilian Portuguese

Grazie
Italian

多谢
Simplified Chinese

Danke
German

நன்றி
Tamil

ありがとうございました
Japanese

감사합니다
Korean