



# Using WebSphere Enterprise Service Bus for z/OS



# Trademarks

The following are trademarks of the International Business Machines Corporation in the United States, other countries, or both. For a complete list of IBM trademarks please visit [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml)

CICS	IBM Logo	S/390
DB2	IMS	Tivoli
E-business logo	iSeries	VM/ESA
ESCON	MVS	VSE/ESA
eServer	OS/390	WebSphere
FICON	pSeries	z/OS
IBM	Rational	zSeries
	RS/6000	System z

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 trademarks of Microsoft Corporation in the United States, other countries, or both.  
Microsoft trademark guidelines

Intel is a registered trademark of Intel Corporation in the United States and other countries.

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

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

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



## ***Agenda***

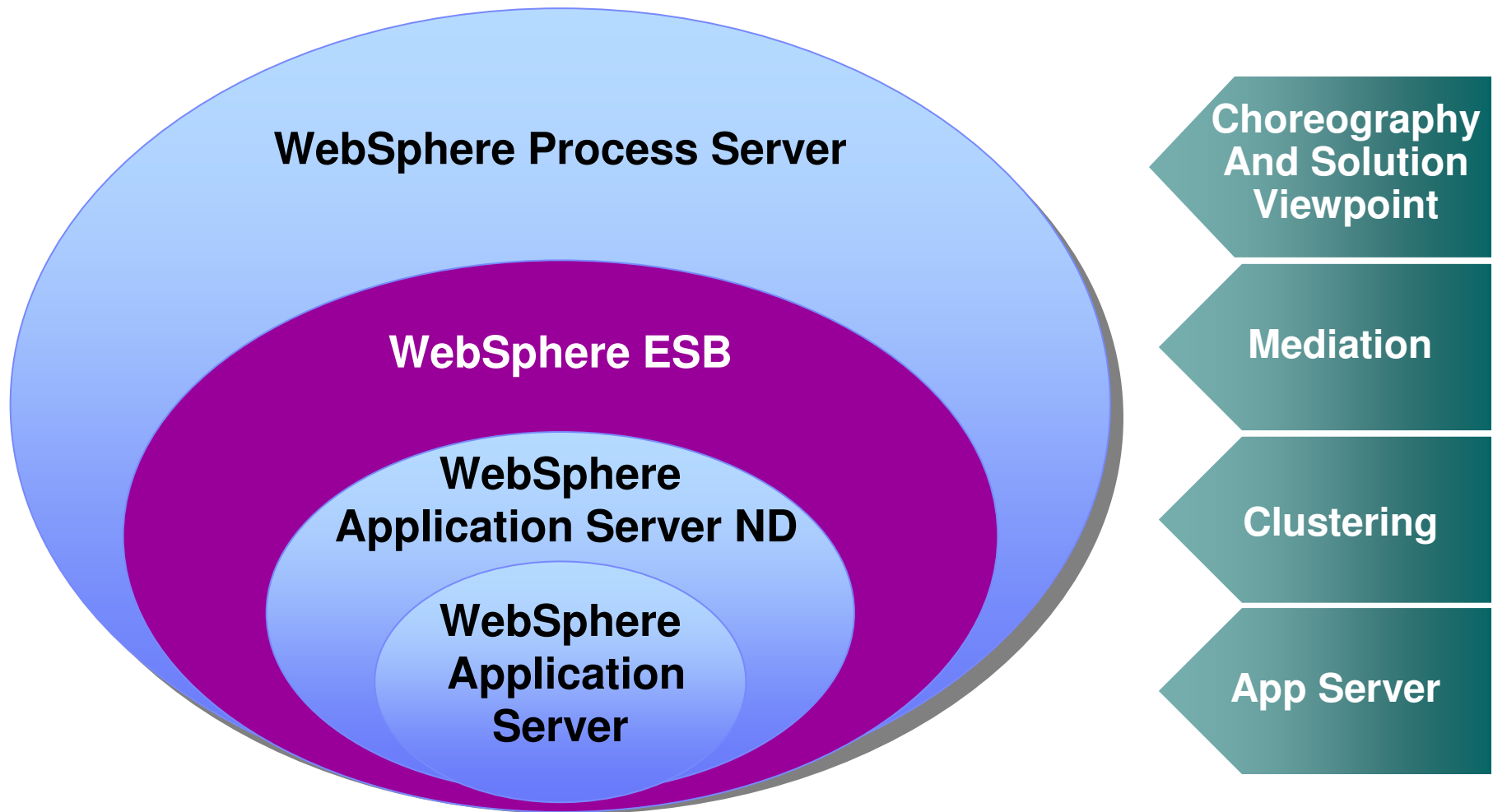
- What is WebSphere Enterprise Service Bus
- What's new with V6.1
- A System z customer scenario



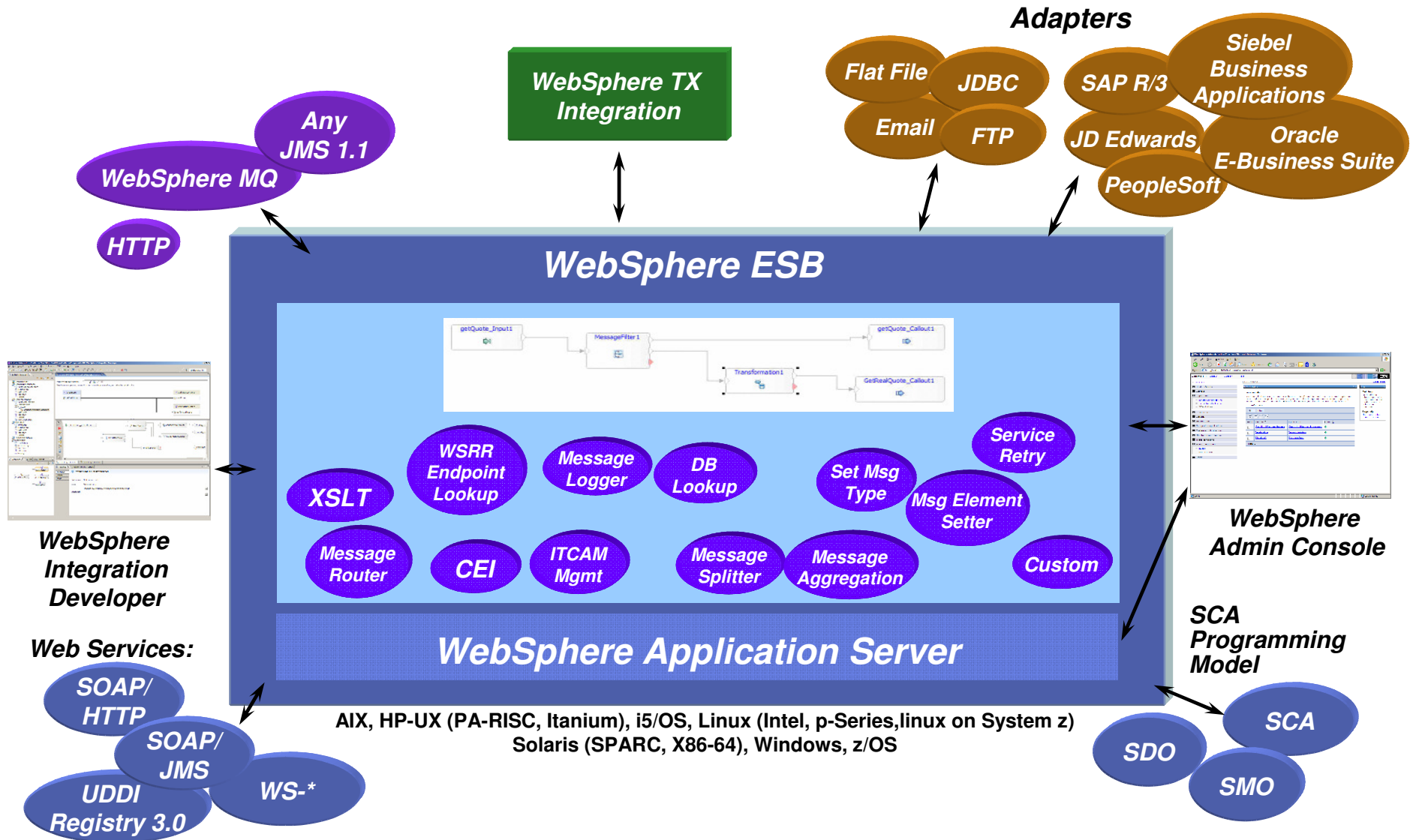
SOA: Unlock business value.  
→ New software and services.



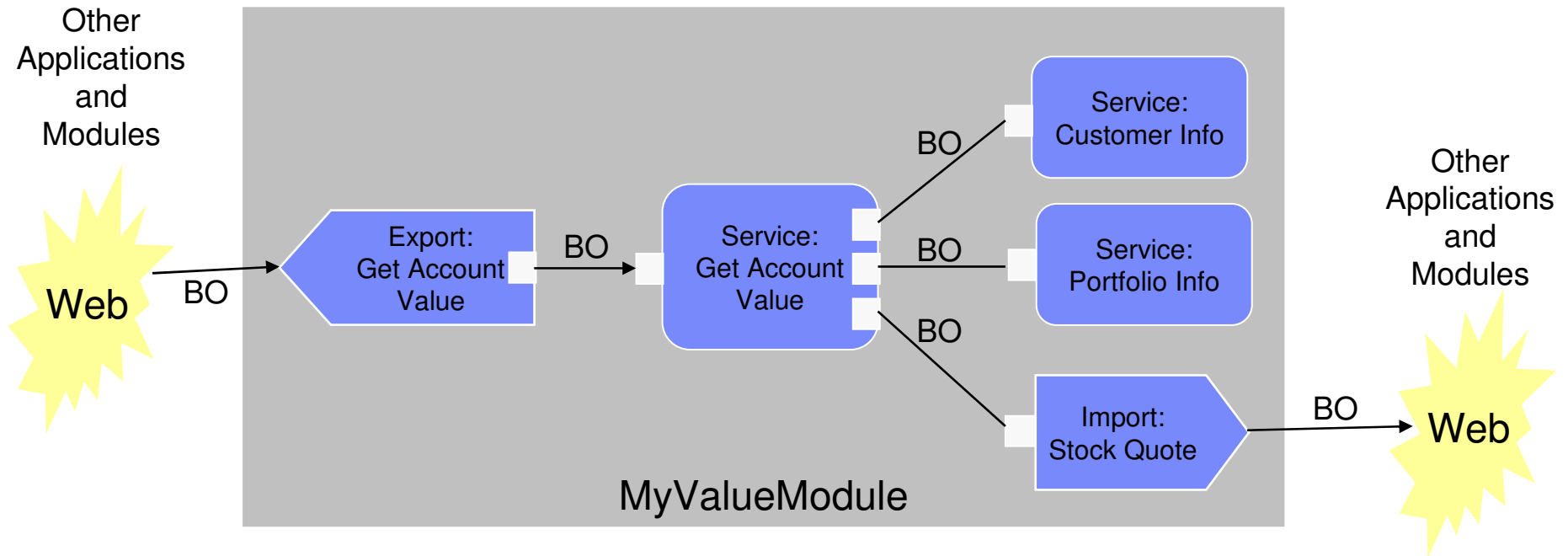
## *WebSphere Application Server, ESB, and Process Server*



# WebSphere ESB



## SCA and SDO/BO – Conceptual View

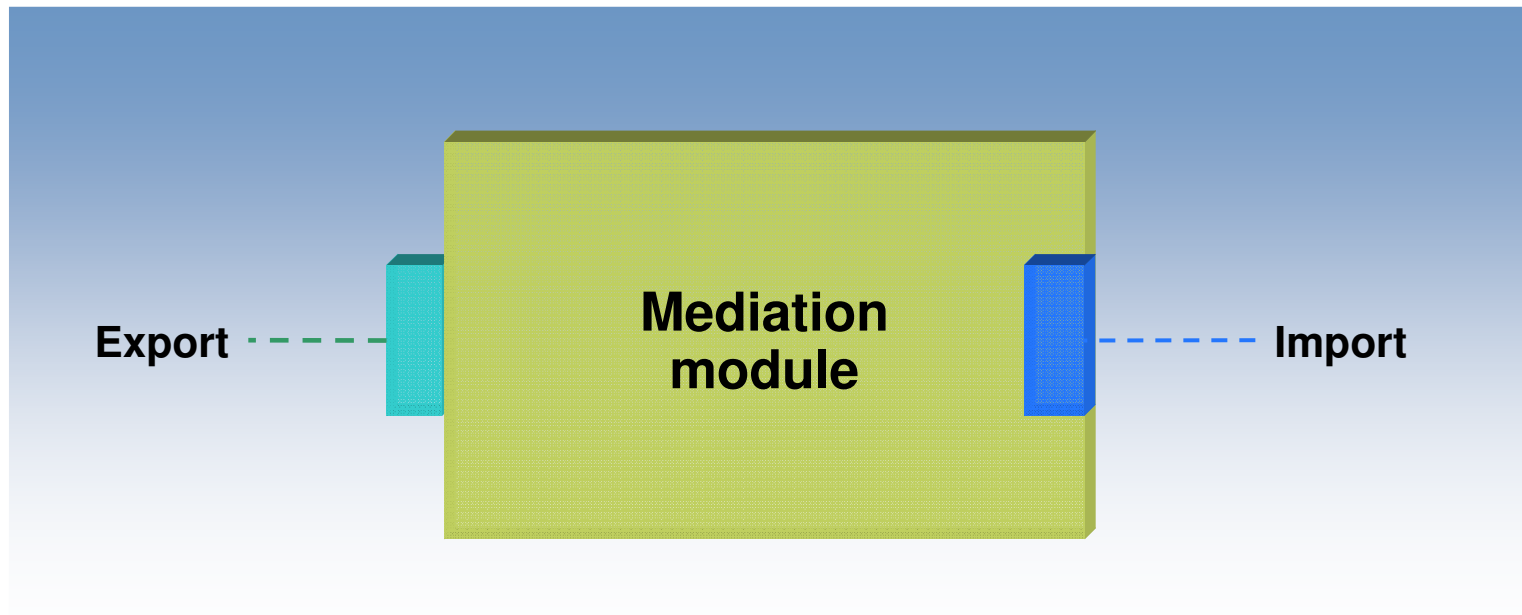


- SCA is the component model
- Components may be wired together
- Business Objects are the data flowing on wires between Components

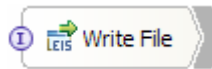
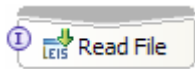


## Concepts: Mediation Module

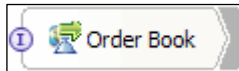
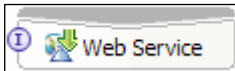
- **Interactions with external service requesters and providers defined by imports and exports**
  - ▶ Interfaces are defined using the Web Services Description Language (WSDL)
    - Which may contain several service *operations*
  - ▶ Different kinds of requester and provider are made available via different *bindings* for the imports and exports



# Export/Import Bindings



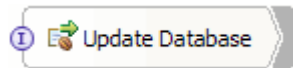
**WebSphere Adapters (e.g. CICS, IMS, SAP...)**



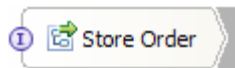
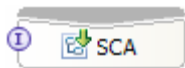
**Web services (SOAP/HTTP, SOAP/JMS)**



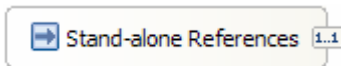
**JMS / WebSphere MQ JMS / WebSphere MQ**



**EJB**



**Native (SCA)**



**Java invocation**





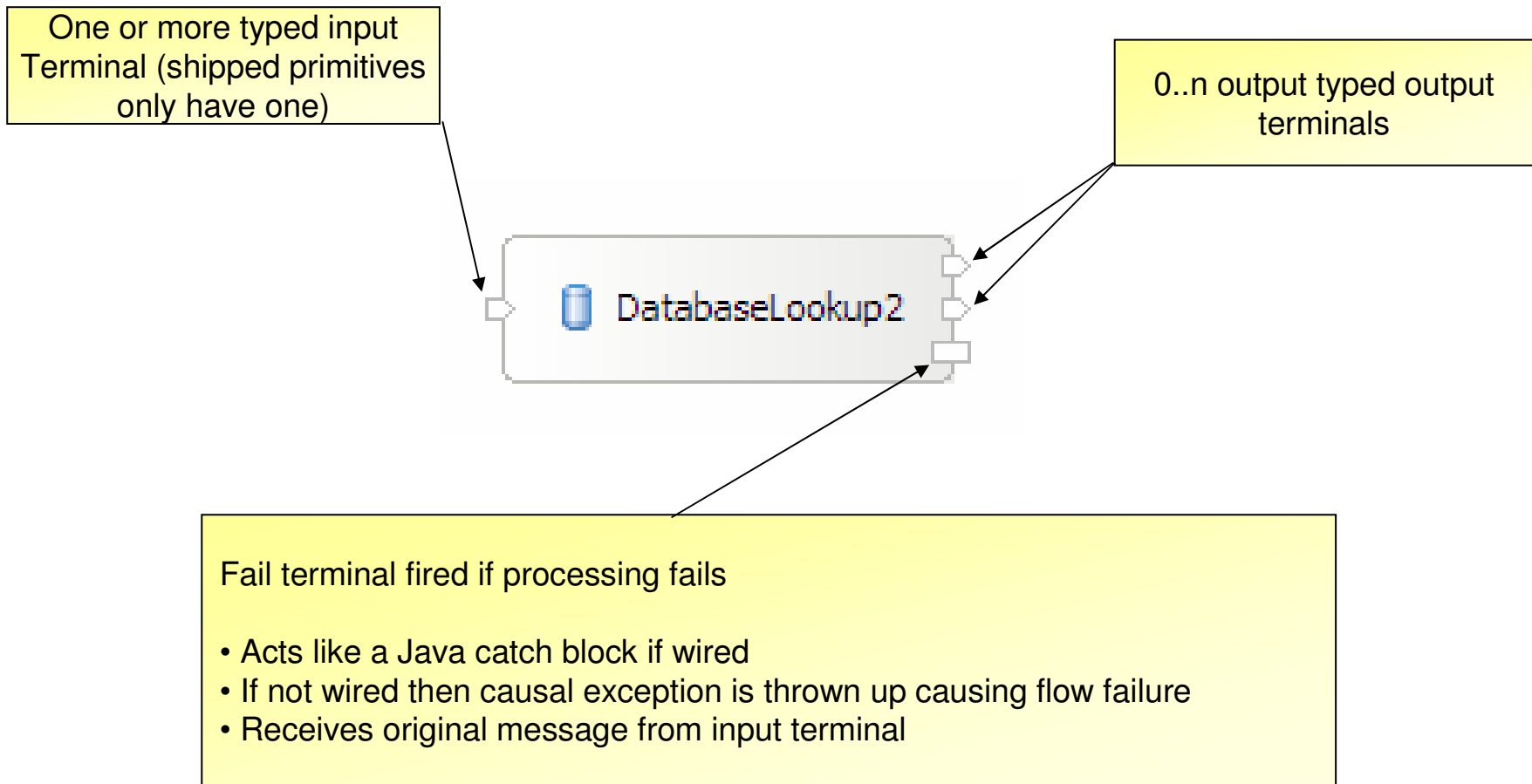
# Concepts: Integrating request-response interactions using WebSphere ESB Mediation Flow Components

- **Processing is performed by one or more mediation *flows*, comprising instances of mediation *primitives***
- **Processing of requests is separated from processing of responses**
- **WebSphere ESB Mediation Flow Components can act as ‘service intermediaries’**
  - ▶ Allowing the mediation flow component to
    - pass a (potentially modified) request from a service requester to a service provider
    - pass a (potentially modified) response from a service provider to a service requester
- **Request processing within a mediation flow component can send a response back to the requester without necessarily needing to contact a service provider**

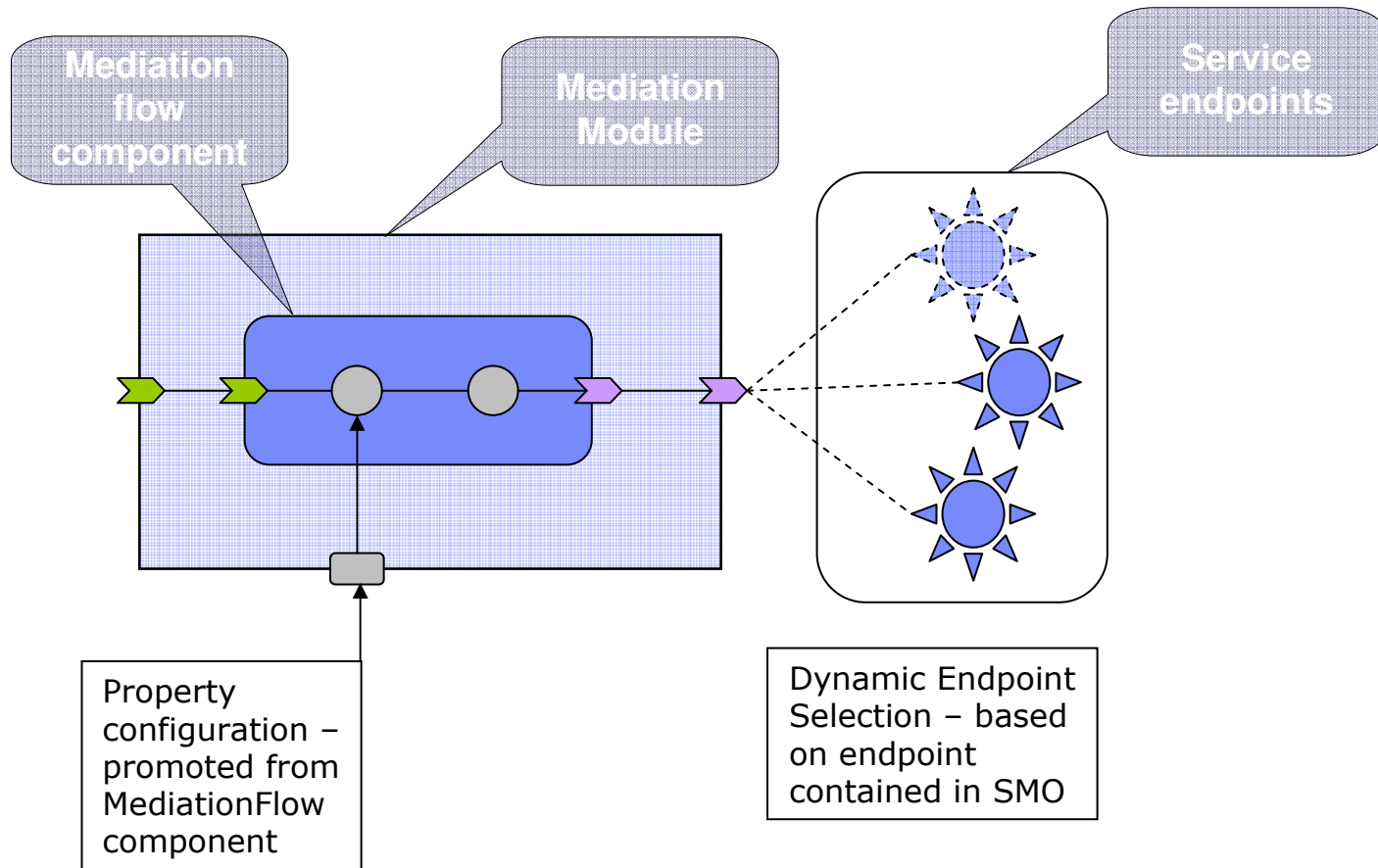




# Anatomy of a Mediation Primitive



# Dynamic Service Invocation



# User Roles and Tasks

**WebSphere ESB**

**ESB  
Solution**

**WebSphere Integration Developer**

**Solution  
Administrator**



- Operates in “administration space”
- Understands basic interaction patterns of business processes
- Focuses on deployed mediation modules and their external interactions

**Integration  
Developer**



- Operates in “development space”
- Understands semantics of business processes to be integrated and enables realization of business goals of underlying processes
- Focuses on Mediation Module construction



# WebSphere Integration Developer

- WESB module development
  - ▶ Using provided components for external interactions (common with WPS), and mediation flow
  
  - ▶ Protocol conversion
  - ▶ Matching and routing
  - ▶ Data conversion
  - ▶ Service aggregation
  - ▶ Registry lookup and dynamic invocation
  - ▶ Etc...
  
- Deployment and testing



# Converting (between transport protocols)

The screenshot shows the IBM WebSphere Integration Developer 6.1 interface. The main window is titled "Business Integration - ProtocolConversion - Assembly Diagram". The left sidebar shows a project tree for "ProtocolConversion" with sub-items like "Assembly Diagram", "Export1", "Dependencies", "Mediation Logic", "Data Types", "Account", "Interfaces", "AccountInterface", and "Mapping". The central workspace displays an assembly diagram with two components: "Export1" and "Import1", connected by a line. A yellow callout box points to the "Business Inte..." window title bar with the text: "The business integration perspective is used for developing WebSphere ESB modules". The bottom right pane shows a console window with the following table:

Description	Resource	Path	Location
<b>Warnings (1 item)</b>			
⚠ CWSCA8010E: The Import1 import has no binding.	Import1.import	ProtocolConversion	line 2

# Converting (between transport protocols)

The assembly diagram depicts the imports and exports (or externals) of the current module

Description	Resource	Path	Location
<b>Warnings (1 item)</b>			
⚠️ CWSCA8010E: The Import1 import has no binding.	Import1.import	ProtocolConversion	line 2



# Converting (between transport protocols)

The screenshot displays the IBM WebSphere Integration Developer 6.1 interface. The main canvas shows an Assembly Diagram with two components, 'Export1' and 'Import1', connected by a line. The 'Export1' component is on the left, and the 'Import1' component is on the right. A yellow callout box points to this connection with the text: "Here we can see that one import and one export have been dragged from the palette onto the canvas and linked together".

The 'Palette' on the left side of the canvas shows various components, with 'Import' and 'Export' highlighted in a pink box. The 'References' view at the bottom left shows a diagram of the 'ProtocolConve...' component.

The 'Problems' view at the bottom right shows a warning message: "CWSCA8010E: The Import1 import has no binding." This message is highlighted in a pink box, and a yellow callout box points to it with the text: "A warning appears to inform us that the import has no binding".

Description	Resource	Path	Location
<b>Warnings (1 item)</b>			
CWSCA8010E: The Import1 import has no binding.	Import1.import	ProtocolConversion	line 2



# Converting (between transport protocols)

Business Integration - ProtocolConversion - Assembly Diagram - IBM WebSphere Integration Developer 6.1 - C:\Stephen\work\workspaces\WID61GA\_...

File Edit Navigate Search Project Data Run Window Help

Business Integr... Physical Resour... ProtocolConversion - Assembly Diagram

Palette

- Favorites
- Components
- Mediation Flow
- Java
- Import
- Export
- References

Export1

- Undo Layout Contents
- Redo
- Add Interface
- Generate Binding...**
  - HTTP Binding
  - Messaging Binding...
  - SCA Binding
  - Web Service Binding**
- Remove Binding
- Refactor
- Copy
- Paste
- Delete
- Rename
- Select All
- Wire References to New
  - Wire to Existing
  - Wire (Advanced) ...
- Test Component
- Test Component in Isolation
- Show in Properties

Adding a binding is a simple right click and menu selection

Path	Location
protocolConversion	line 2

# Converting (between transport protocols)

The screenshot displays the IBM WebSphere Integration Developer 6.1 interface. The main window shows a 'ProtocolConversion - Assembly Diagram' with two components: 'Export1' and 'Jms Import1'. A callout box on the left states: 'Binding properties are easy to configure. The relevant options can be overridden at runtime via the admin console'. A callout box on the right states: 'The export has a Web service binding. The import has a JMS binding. Protocol conversion complete!'. The bottom right pane shows the 'Properties' view for 'Export: Export1 (Web Service Binding)' with the following configuration:

Property	Value	Action
Address	http://localhost:9080/ProtocolConversionWeb/sca/Export1	Configure...
Port	Export1_AccountInterfaceHttpPort	Browse...
Service	Export1_AccountInterfaceHttpService	
Namespace	http://ProtocolConversion/AccountInterface/Binding	

# Data Conversion

The screenshot displays the IBM WebSphere Integration Developer 6.1 interface. The main window shows an Assembly Diagram for 'DataConversion'. The diagram consists of two components: 'DataConversionExport' and 'DataConversion', connected by a line representing a mediation flow. A yellow callout box points to this flow with the text: "An export and a mediation flow in the assembly diagram. Double clicking on the mediation flow takes you into the mediation flow editor".

The Properties window at the bottom right is open to the 'Component: DataConversion (Mediation Flow)' tab. It shows the following details:

Section	Field	Value
Details	Name:	DataConversion
	Display name:	DataConversion <input checked="" type="checkbox"/> Synchronize with the name field
Implementation	Folder:	<input type="text"/> Refactor...
	Description:	<input type="text"/>

# Data Conversion

The screenshot displays the IBM Business Integration Mediation Flow Editor interface. The main workspace shows a mediation flow diagram with three primitives: 'Input', 'XSLTransformation', and 'Input Response', connected in a linear sequence. The 'Input' primitive is labeled 'getDetails : GetBa...', the 'XSLTransformation' primitive is labeled 'XSLTransformation', and the 'Input Response' primitive is labeled 'getDetails : GetBa...'. A yellow callout bubble points to this flow with the text: 'The mediation flow has Input and Response primitives. They have been connected with an XSLT primitive.'

The interface includes a menu bar (File, Edit, Navigate, Search, Project, Data, Run, Window, Help), a toolbar, and a left-hand tree view showing the project structure. The right-hand side features a 'Mediation Flow' panel with various configuration options like References, Correlation Co..., Transient Con..., and Shared Context. At the bottom, a 'Properties' panel for the selected 'XSLTransformation' primitive is visible, showing fields for Display name, Name, and Description.

Property	Value
Display name:	XSLTransformation
Name:	XSLTransformation1
Description:	This mediation primitive enables users to transform messages according to



# Data Conversion

The screenshot displays the IBM WebSphere Integration Developer 6.1 interface. The main window shows a mapping diagram for an XSLT transformation named 'XSLTransformation1\_req\_1'. The diagram is organized into two main sections: 'Input' and 'Output', connected by a central processing area.

**Input Section (Left):**

- Root: XSLTransformation1\_req\_1
- Element: body (expanded)
- Element: getDetails (type: GetDetailsType)
- Element: input (type: BankDetails)
- Fields: FirstName (string), LastName (string), SortCode (int), AccountNumber (int)

**Processing Area (Center):**

- Concat (receives input from FirstName and LastName)
- Move (receives input from SortCode)
- Move (receives input from AccountNumber)
- Assign (receives input from the Concat element)
- Assign (receives input from the Move element)

**Output Section (Right):**

- Root: body (expanded)
- Element: getDetailsResponse (type: GetDetailsResponseType)
- Element: output (type: PartnerBankDetails)
- Fields: Name (string), SortCode (int), AccountNumber (int)
- Element: PartnerInfo (type: PartnerDetails)
- Fields: Name (string), ID (string)

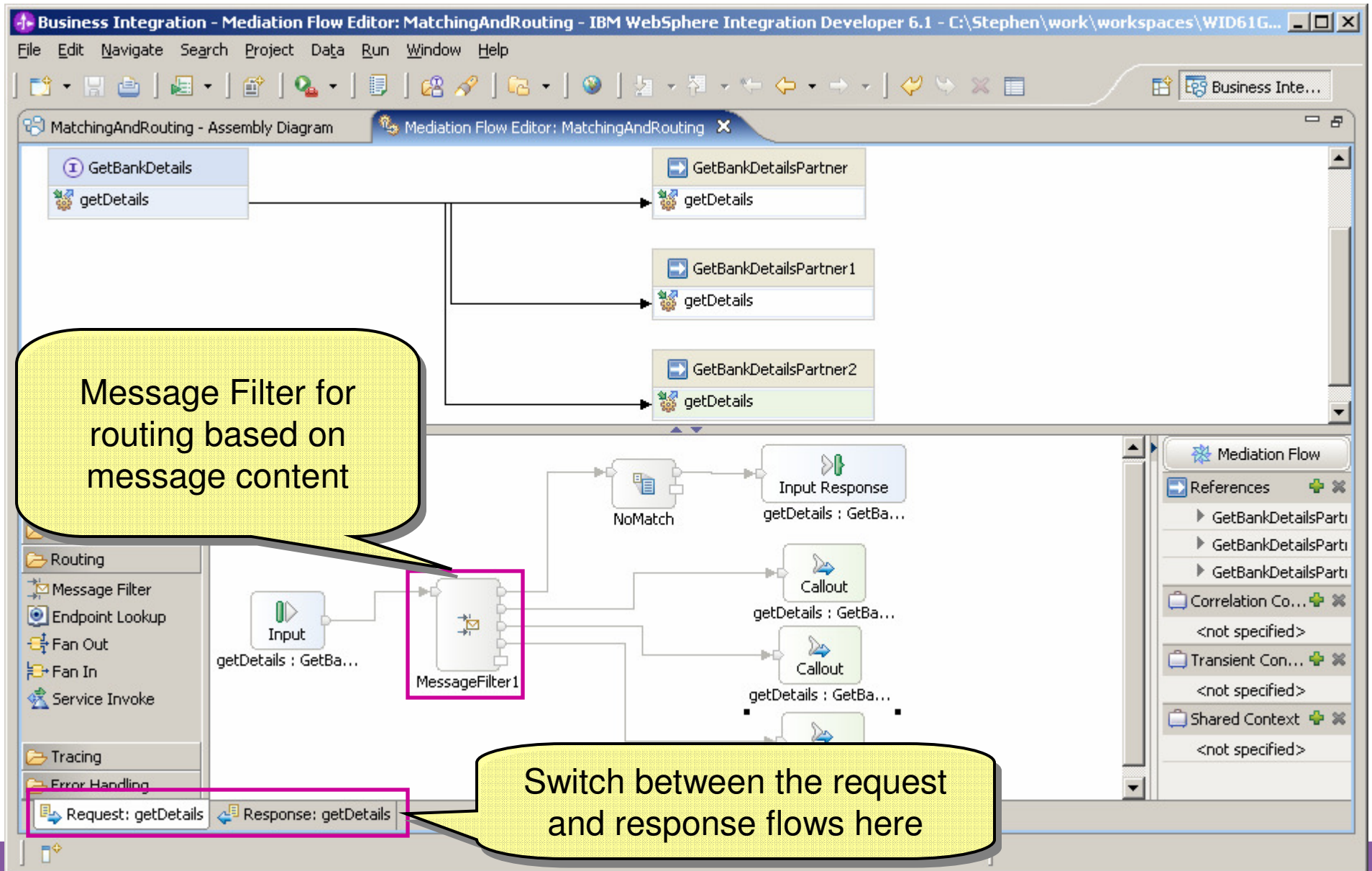
The diagram illustrates the flow of data from the input structure through various transformation operations (Concat, Move, Assign) to the output structure. A yellow callout box highlights the interface, stating: "A crisp, clean drag and drop interface for data conversion. In addition to the basics there are submaps, comprehensive array support, Java callouts. The XSLT primitive can access and manipulate headers as well as message data".

# Matching and Routing

The assembly diagram shows one web service export, one mediation flow, and three imports. The mediation flow will route to the one of the three services based on the content of the input message.

⚠ CWSCA8010E: The Bank1 import has no binding.	Bank1.import	MatchingAndRouting	lir
⚠ CWSCA8010E: The Bank2 import has no binding.	Bank2.import	MatchingAndRouting	lir
⚠ CWSCA8010E: The Bank3 import has no binding.	Bank3.import	MatchingAndRouting	lir

# Matching and Routing



# Matching and Routing

The screenshot displays the IBM WebSphere Integration Developer 6.1 Mediation Flow Editor. The main workspace shows an assembly diagram with an 'Input' component connected to a 'MessageFilter1' component. The 'MessageFilter1' component is configured with a 'Distribution mode' of 'First' and a list of filters. A yellow callout box highlights the text: 'Filtering is based on XPath expressions. In business terms we are taking data from the input request and selecting a service based on that data.'

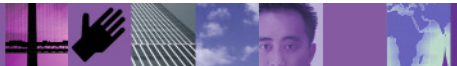
The 'Message Filter : M' properties window is open, showing the following filters:

Pattern	Terminal name
/body/getDetails/input/SortCode='998877'	bank1
/body/getDetails/input/SortCode='332211'	bank3
/body/getDetails/input/SortCode='445566'	bank2

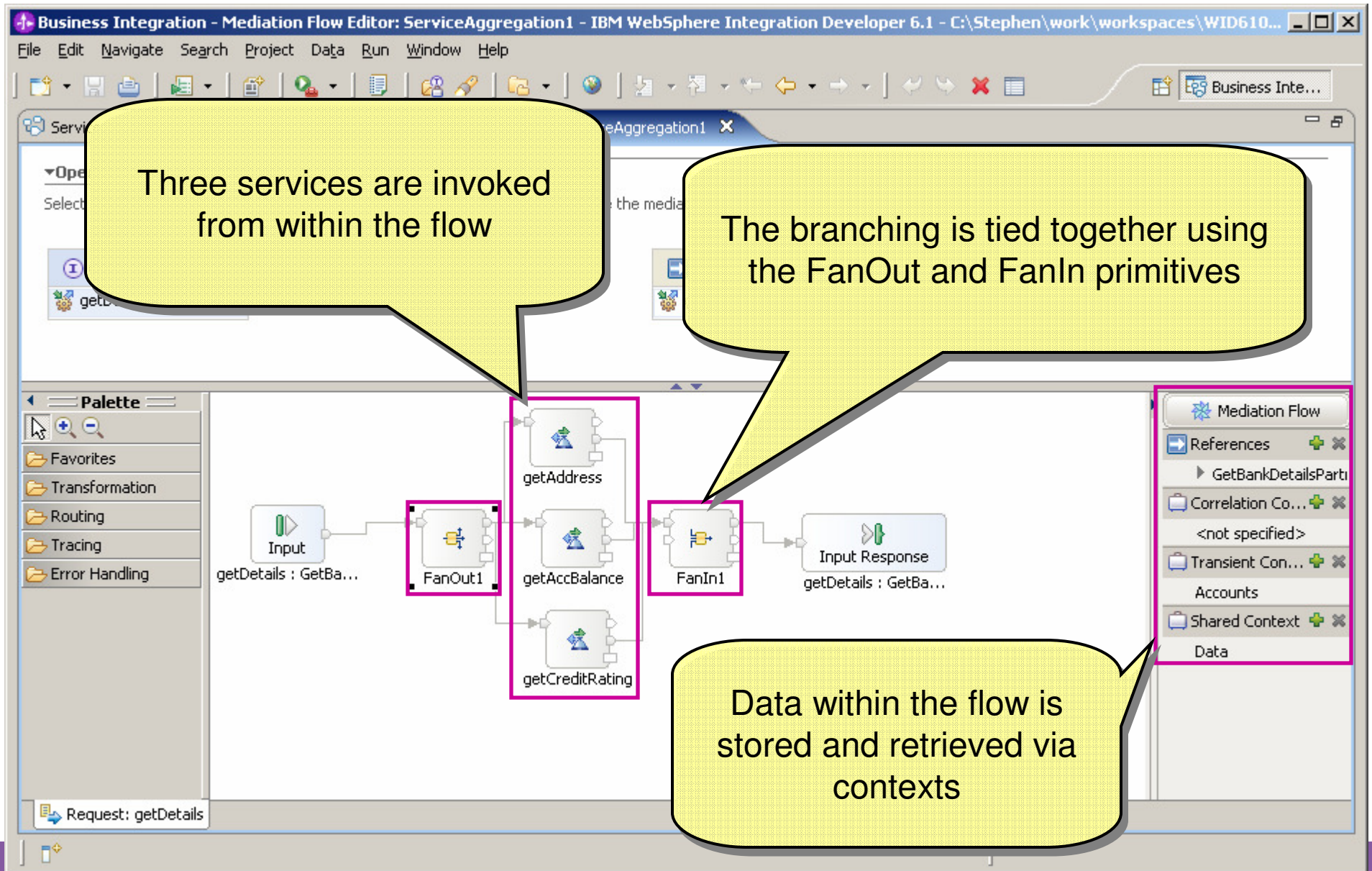


## Service Aggregation

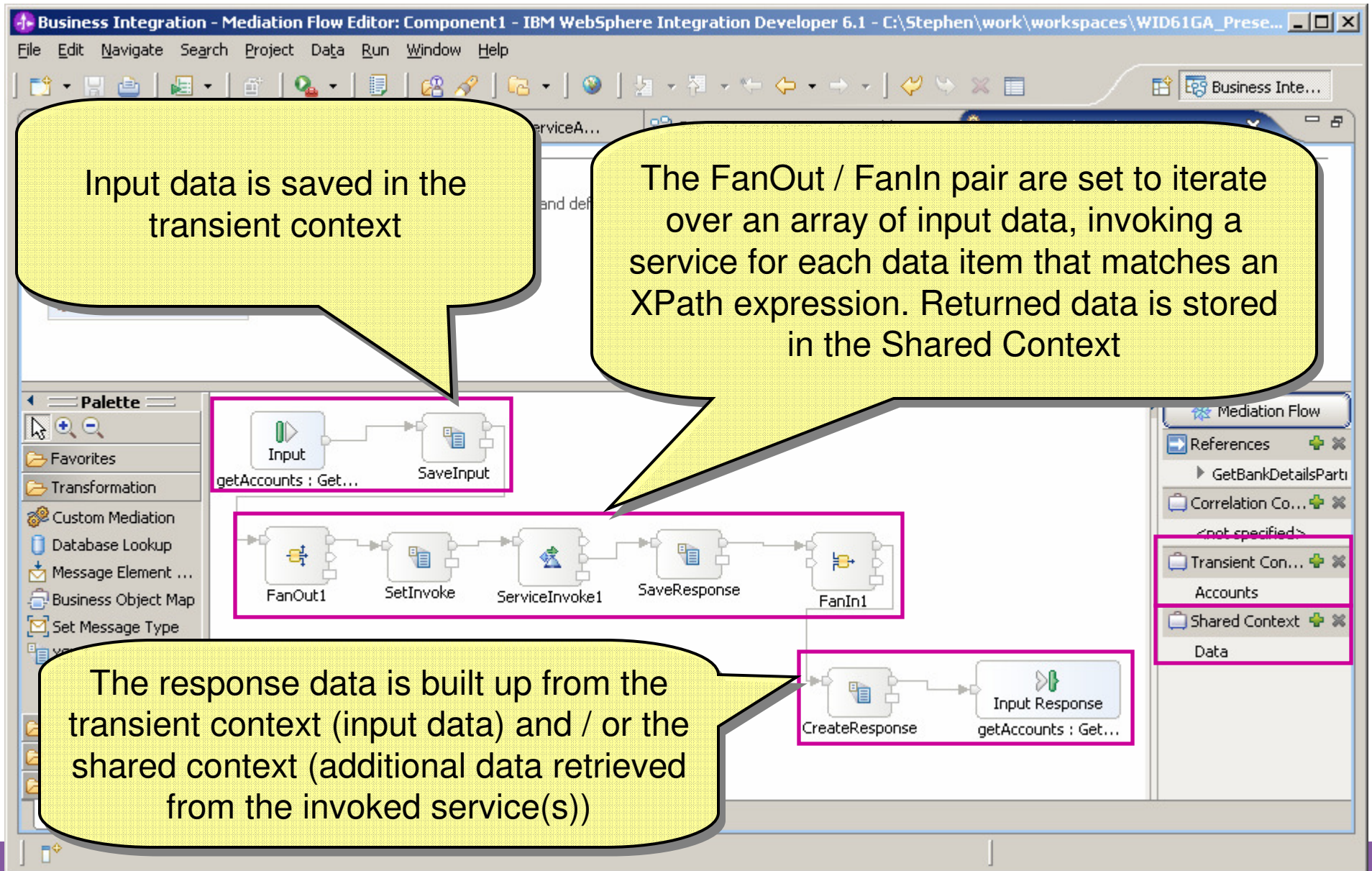
- Split messages for separate processing
- Augment messages from multiple services
- Amalgamate data from multiple sources
- Provide alternate services for failover



# Service Aggregation



# Service Aggregation

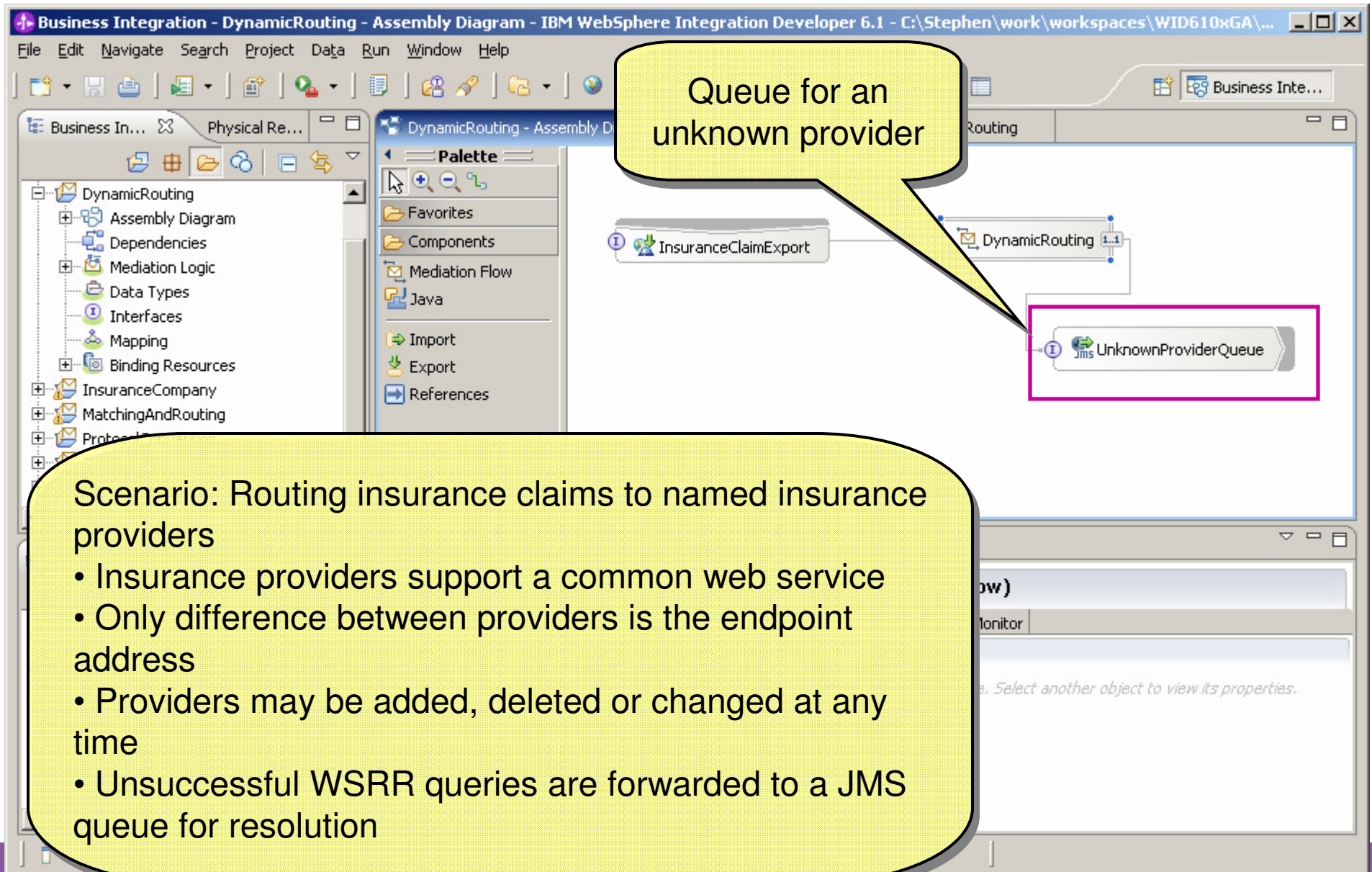


Input data is saved in the transient context

The FanOut / FanIn pair are set to iterate over an array of input data, invoking a service for each data item that matches an XPath expression. Returned data is stored in the Shared Context

The response data is built up from the transient context (input data) and / or the shared context (additional data retrieved from the invoked service(s))

# WSRR & Dynamic Endpoint Lookup



Queue for an unknown provider

Scenario: Routing insurance claims to named insurance providers

- Insurance providers support a common web service
- Only difference between providers is the endpoint address
- Providers may be added, deleted or changed at any time
- Unsuccessful WSRR queries are forwarded to a JMS queue for resolution



# WSRR & Dynamic Endpoint Lookup

## WebSphere Service Registry & Repository

- Enables your ESB to be more dynamic, flexible and adaptable
- Promotes reuse
- Enables governance

The screenshot shows the Mediation Flow Editor interface. At the top, a menu bar includes File, Edit, Navigate, Search, Project, Data, Run, Window, and Help. The main workspace displays a mediation flow diagram. A box labeled 'InsuranceClaimPartner' contains a 'makeClaim' activity. Below it, an 'EndpointLookup' activity is connected to a 'Callout' activity labeled 'makeClaim : Insur...'. A yellow callout bubble points to the 'EndpointLookup' activity, stating: 'The invoked endpoint is decided at runtime based on the response from WSRR'. The bottom of the interface features a 'Properties' pane for the selected 'Endpoint Lookup : EndpointL' activity, showing a 'Classifications' field with the value 'http://www.ibm.com/wsrr/governance=#Operational'. Other panes include 'References', 'Outline', and 'Visual...'. The title bar of the application window reads 'Business Integration - Mediation Flow Editor: DynamicRouting - IBM WebSphere Integration Developer 6.1 - C:\Stephen\work\workspaces\WID610xGA...'.

The invoked endpoint is decided at runtime based on the response from WSRR

# Deployment and Testing

The screenshot displays the IBM WebSphere Integration Developer 6.1 interface. The main workspace shows a mediation flow diagram with operations like 'getAddress' and 'getAccBalance'. A 'Servers' tab is open at the bottom, showing a table of server instances. A callout box points to the 'Servers' tab, and another callout box points to the 'Start the server in debug mode' button in the toolbar.

**Complete WebSphere Process Server and WebSphere ESB server installation profiles (with a limited usage licence)**

**Servers Tab**

**Server control options, including start the server in debug mode**

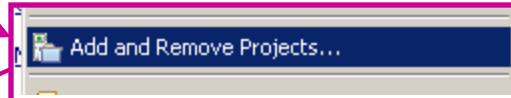
Server	Status	State
WebSphere ESB Server v6.1	Stopped	Republish
WebSphere Process Server v6.1	Stopped	Republish

Start the server in debug mode

# Deployment and Testing

Server	Status	State
WebSphere ESB Server v6.1	Debugging	Synchronized

Server started in debug mode



Right click on the server and select Add and Remove Projects...

**Add and Remove Projects**

Modify the projects that are configured on the server

Move projects to the right to configure them on the server

Available projects:

- DataConversionApp
- MatchingAndRoutingApp
- ProtocolConversionApp

Configured projects:

- ServiceAggregation1App
- ServiceAggregation2App

Add >

< Remove

Add All >>

<< Remove All

< Back   Next >   Finish   Cancel

Add and remove the developed mediations using the panel

# Deployment and Testing

Control of the component test happens here. The getDetails method of ServiceAggregation1 has been invoked using test data

The deployed applications now appear under the server

Test data has been entered here

The screenshot displays the IBM WebSphere Integration Developer 6.1 interface. On the left, a project tree shows 'ServiceAggregation1' with sub-items like 'Assembly Diagram', 'Dependencies', 'Mediation Logic', 'Data Types', 'Interfaces', and 'Mapping'. The main workspace is divided into several panes:

- Events:** A list of events for the 'Invoke (ServiceAggregation1:getDetails)' operation, including 'Invoke started' and 'Invoke (ServiceAggregation1:getDetails)'. This pane is highlighted with a pink box.
- Properties:** Configuration for the component test, including:
  - Component: ServiceAggregation1
  - Interface: GetBankDetails
  - Operation: getDetails
  - Invoke export using binding
- Initial request parameters:** A table showing the test data used for the invocation.
 

Name	Type	Value
input	BankDetails	✓
First	string	✓
Last	string	✓
Sort	int	✓ 112233
Account	int	✓ 98765432

 The values '112233' and '98765432' are highlighted with a pink box.
- Servers:** A table showing the status of various servers.
 

Server	Status
WebSphere ESB Server v6.1	Debugging
ServiceAggregation1App	Started
ServiceAggregation2App	Started
WebSphere Process Server v6.1	Stopped

 This table is also highlighted with a pink box.



# Deployment and Testing

The debug perspective was opened because a break point was reached while the flow was executing

Continued flow execution is controlled here

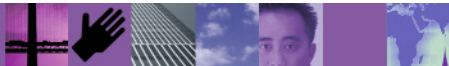
Name	Value
context	org.eclipse.emf.ecore.sdo.impl.DynamicED...
headers	org.eclipse.emf.ecore.sdo.impl.DynamicED...
body	org.eclipse.emf.ecore.sdo.impl.DynamicED...
getDetailsParameters	org.eclipse.emf.ecore.sdo.impl.DynamicED...

'In flight' data can be viewed and edited here

The mediation flow shows current progress. The golden sphere is where the flow has been halted. A purple tick mark shows the path along which execution has occurred to this stage

## ***WebSphere ESB for z/OS – Extends the value of WebSphere Application Server for z/OS***

- Clustering support for fault tolerance, scalability
- Java platform with zAAP offload
- Native application integration
  - WebSphere MQ for z/OS (including CICS)
  - WebSphere TX integration
  - CICS Transaction Gateway (JCA)
  - IMS Connect (JCA)
  - DB2 – DB2 Universal JDBC Provider, IBM Application Connectivity to DB2 for z/OS Feature
  - PDS files
  - WebSphere Classic Federation Server for z/OS integration
- IBM z/OS Security Server support



## ***New in Version 6.1.2***

- SubstitutionGroup, AttributeGroup and Group support
  - Business Object Framework Java API support
  - MessageElementSetter Mediation Primitive updated
    - Can specify which member of a SubstitutionGroup is expected
- Runtime Problem Determination Improvements
- Improved Support for MQ CICS Bridge Interaction
  - MQCIH support



## ***New in Version 6.1.2***

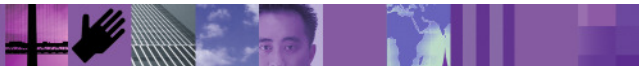
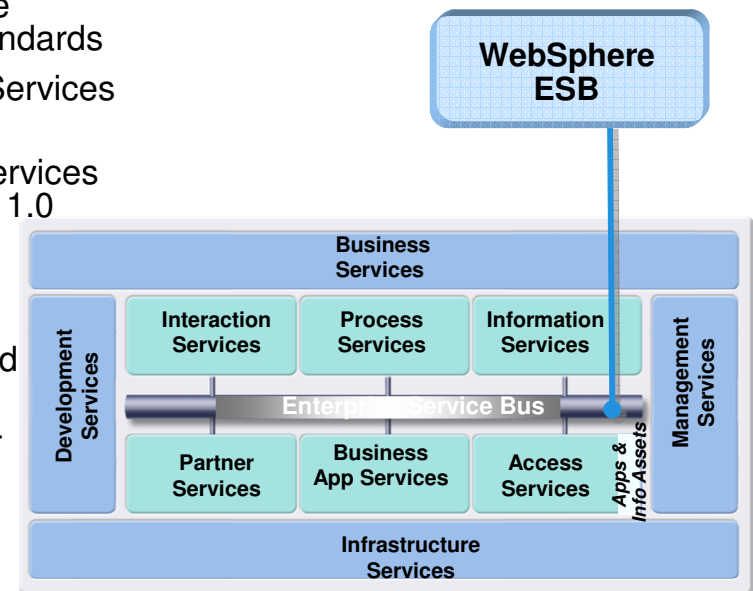
### ■ Data Handlers

- Convert between physical and logical format
- Can be used with any binding that supports Data Handlers
  
- Delimited Data Handler
  - used when you have serialized data in a delimited format (for example, comma-separated format) as input into an export or need to generate serialized data in a delimited format for an import.
- Fixed Width Data Handler
  - used when you have serialized data in a fixed-width format as input into an export or need to generate serialized data in a fixed-width format for an import.
- JSON Data Handler
  - used when you have serialized data in a JSON format as input into an export or need to generate serialized data in a JSON format for an import.
- WTX MapSelection Data Handler
  - lets you use WebSphere Transformation Extender, a universal validation and transformation engine, to convert business objects to many data formats and vice versa.
- XML Data Handler
  - used when you have serialized XML data as input into your export (for inbound processing at run time) or need to generate serialized XML data for an import (for outbound processing at run time).



## WebSphere Enterprise Service Bus v6.2 – Available now!

- Delivers new policy-driven connectivity
  - Increased flexibility to administratively configure service mediations through policies
  - Full set of mediation capabilities manageable via policies
  - Integration with WebSphere Service Registry and Repository for policy management and governance
- Enhances web services standards support
  - Improved interoperability with support for WS-I Reliable Secure Profile, SOAP 1.2, WS-Reliable Messaging and supporting standards
  - Simplified administration of Qualities of Service through Web Services policy sets
  - Simplified development through the provision of a new Web Services binding based on JAX-WS 2.0, JAXB 2.0, SAAJ 1.3, and StAX 1.0
- Enhances service mediation capabilities
  - Enhanced flexibility in service gateway scenarios
  - Enhanced performance with parallel processing for splitting and aggregation patterns
  - Streamlined integration to support WebSphere Process Server solutions
- New! IBM Media Extender for WebSphere ESB
  - Delivers “media-aware” service mediation facilities
  - Key element of IBM’s Media Hub Solution Framework to link business and content systems together for effective media management



**Industry:** Education  
**URL:** <http://cms.bsu.edu/>

**“SOA has been such a gift to us. It enables us to embrace a new technology that provides services at a level that we couldn’t even imagine before.”**

–Dr. O’Neal Smitherman



BALL STATE  
UNIVERSITY



## Ball State University

*Ball State University bridges disparate systems and solves key administrative issue with IBM SOA solution. See <http://soaflexibility.com/ibm/standalone/files/BallState.zip>*

### CHALLENGE

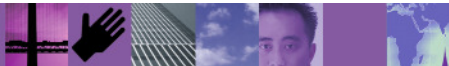
- Coordinate 40 name and address systems to streamline administrative processes and ensure information integrity for users

### SOLUTION

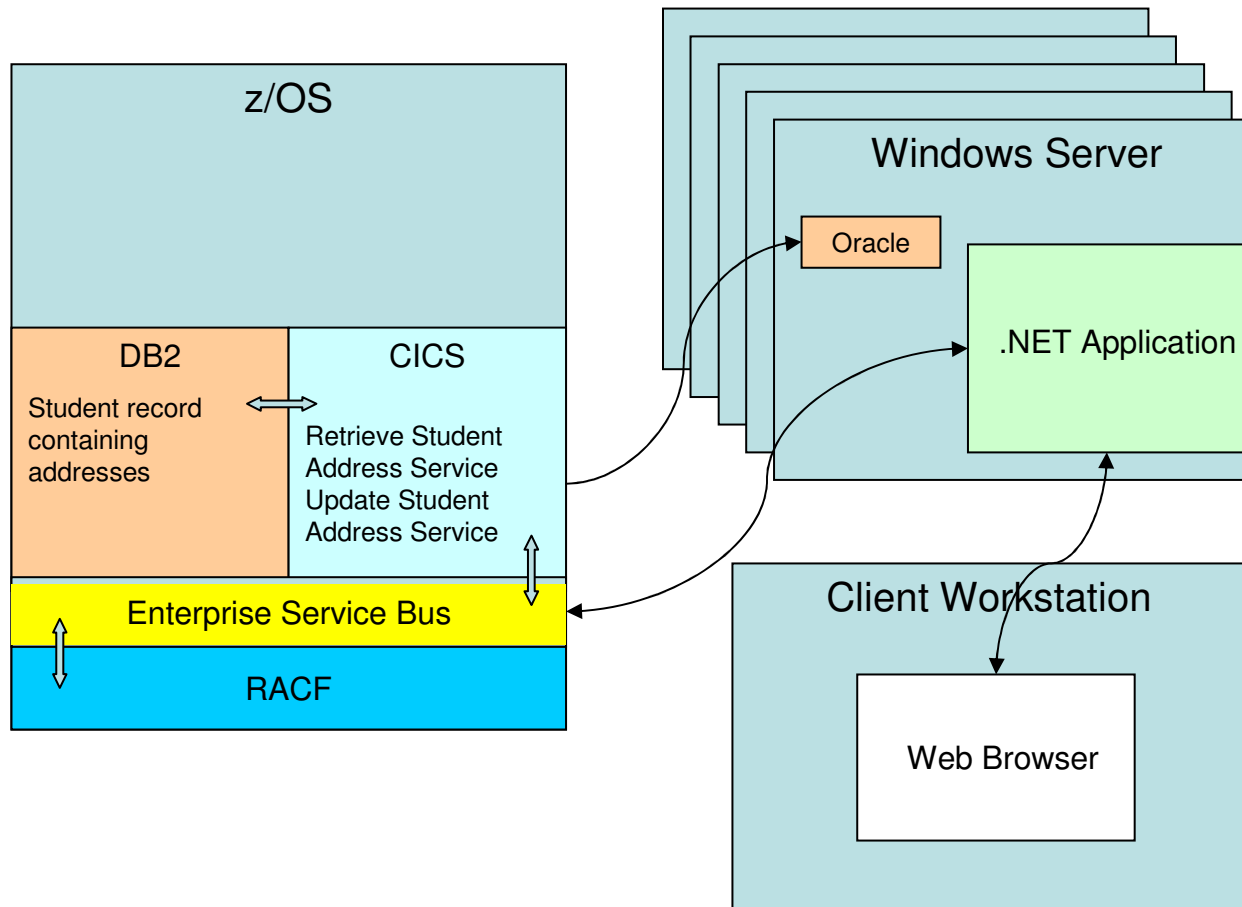
- SOA with Enterprise Service Bus to connect siloed applications without hand-coding individual API calls (WESB, CICS TS, System z)

### BENEFITS

- Ability to develop and implement services in an SOA environment for resolving name and address discrepancies in 10 months, as opposed to several years for hand-coding individual application connections
- Confidence that IBM solution can lead to wider use of SOA to further streamline administrative business processes
- Services created here can be reused in later SOA efforts



# BSU Application Architecture





## Reference Information

- Migration Information
  - Migration to WebSphere Process Server and WebSphere Enterprise Bus 6.2 <http://www-01.ibm.com/support/docview.wss?rs=2307&uid=swg21329733>
  - Announcement Letter 208-325 IBM WebSphere Process Server for z/OS, WebSphere Integration Developer, and WebSphere Enterprise Service Bus for z/OS V6.2 enable dynamic processes and flexible connectivity for agile business solutions <http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=an&subtype=ca&htmlfid=897/ENUS208-325>
  
- Redbooks
  - Redpaper: IBM Connectivity Reviewer's Guide <http://www.redbooks.ibm.com/redpapers/pdfs/redp4434.pdf>
  - RedBook: Implementing an ESB using IBM WebSphere Message Broker V6 and WebSphere ESB V6 on z/OS <http://www.redbooks.ibm.com/abstracts/sg247335.html?Open>
  - IBM WebSphere Business Process Management V6.1 Performance Tuning Redpaper <http://www.redbooks.ibm.com/abstracts/redp4431.html>
  
- Analyst paper
  - Choosing the right SOA platform on System z
  - [ftp://ftp.software.ibm.com/software/websphere/pdf/SOA\\_on\\_System\\_z.pdf](ftp://ftp.software.ibm.com/software/websphere/pdf/SOA_on_System_z.pdf)
  
- Other
  - ESB Portfolio Trifold  
[ftp://ftp.software.ibm.com/software/websphere/integration/wbimessagebroker/esb\\_trifold\\_0103A.pdf](ftp://ftp.software.ibm.com/software/websphere/integration/wbimessagebroker/esb_trifold_0103A.pdf)
  - Teleconference: Which ESB on System z? Selection Guidelines for WebSphere Message Broker, WESB and DataPower XI50 <http://www.ibm.com/software/os/systemz/telecon/30jul/>





Thank  
YOU





# WESB V6.2 Details

**WebSphere** software



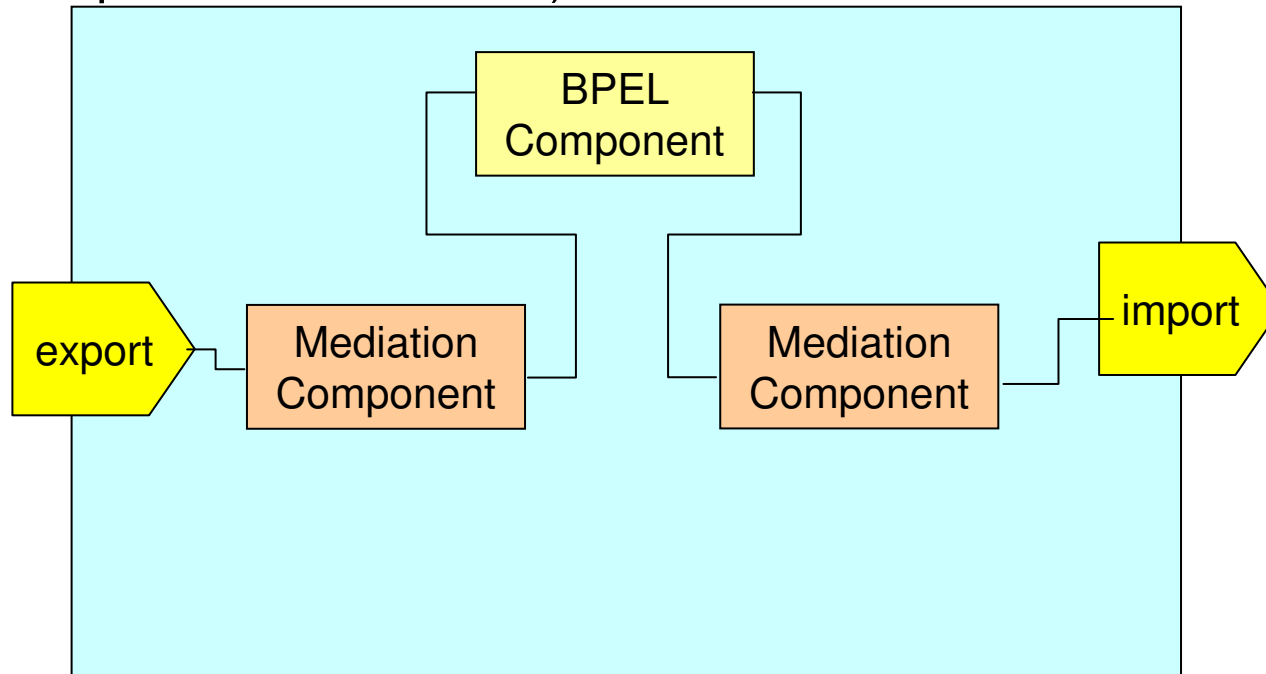
## ***New in Version 6.2***

- Base Connectivity Enhancements
  - Enhanced web service support
  - Data Handlers
  - Enhanced Dynamic Invocation Support
  
- Enhanced Mediation Support
  - Component Level Updates
  - Mediation Subflow Support
  - Service Gateway Scenario Support
  - Enhanced Mediation Dynamic/Declarative Control
  
- Updated/new mediation primitives
  
- Plus
  - ...more Problem Determination enhancements
  - ...more consumability enhancements
  - ...more performance enhancements



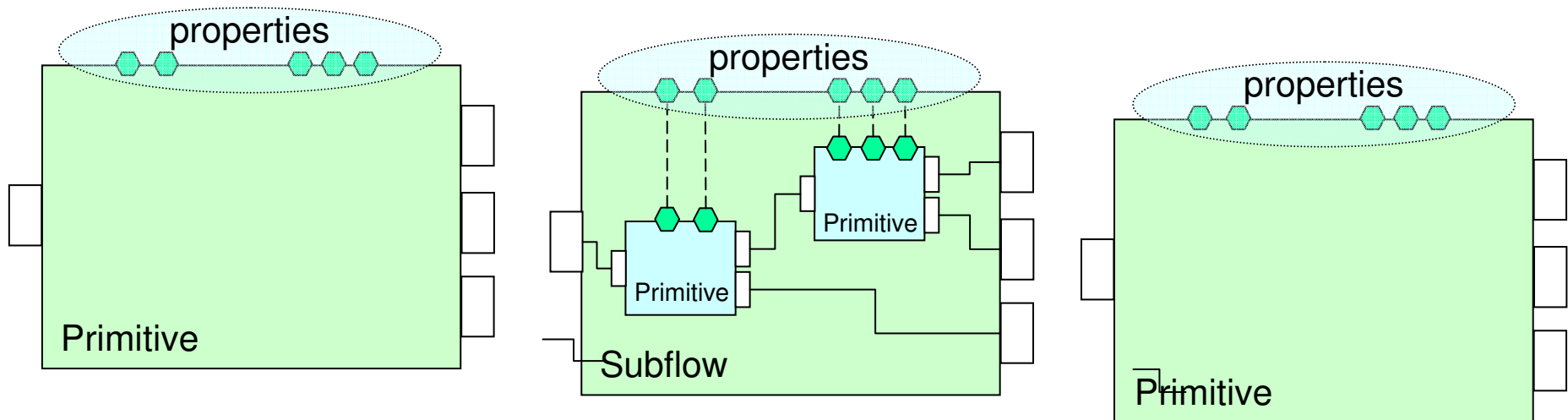
## ***Multiple Mediation Flow Components/in any module***

- Multiple Mediation Flow Components in a Module
- Mediation Flow Components allowed in a (Business) Module
- Header and other context propagated seamlessly (without mandating component intervention)



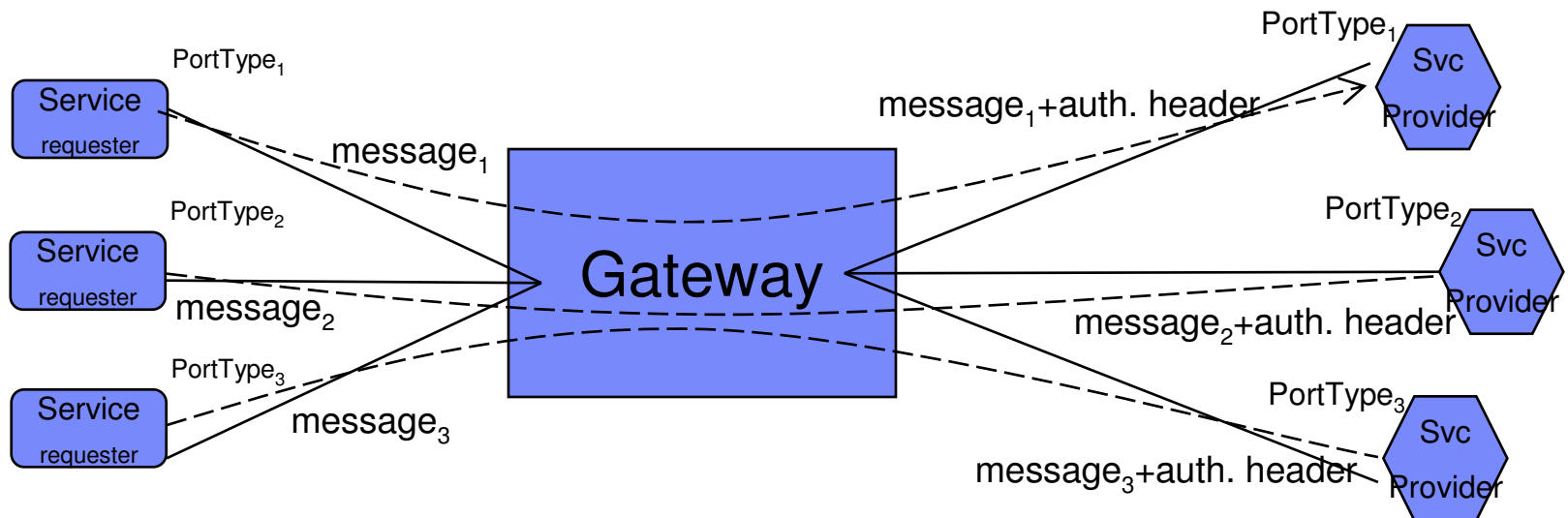
## Mediation Reuse: Mediation Sub-flows

- Subflows enable a snippet of mediation flow to be encapsulated and reused
- A subflow will act rather like a mediation primitive
  - Terminals of primitives within the flow can be exposed become terminals of the subflow
  - Properties of primitives within the flow can be promoted to become properties of the subflow
- The user of a subflow does not need to be aware of the contents of the subflow, just its externals (terminals and properties)



## Service Gateway Scenario: Introduction

- A Service Gateway acts as a proxy to a variety of different services
  - All of the service requesters interact with a single endpoint address
  - The gateway is responsible for
    - Performing a common operation on every message
    - Routing the request to the correct service provider
- Example: add a custom authentication SOAP header, common to all services implemented within a single company
  - Many services with different port types
  - All require an authentication header
  - No other change is made to the web service interaction





## ***Overview of Gateway Scenario Support in WPS/WESB***

- Support for a 'service gateway' interface
  - Support for a 'gateway' interface, which reduces all interactions to one of two abstract operations: one-way or in-out
  - Input and output messages are weakly typed, so that the messages can represent any required business content
- Mediation Primitives for Handling weak-to-strong type Assertions
  - SetMessageType
    - Makes a user-configured assertion about which type is being dealt with
  - QueryMessageType
    - Can discover which message subtype is being dealt with
- Mediation Primitive for performing an explicit data conversion
  - DataHandler mediation primitive
    - can apply a specific DataHandler to an opaque part of the message (such as a String or Byte array field) to parse it into a logical structure



## ***Enhanced declarative mediation control: Goals***

- Goal
  - Allow declarative control of mediation function by attaching policies that specify mediation points of variability dynamically
- Benefits
  - Reduce mediation construction complexity
  - Much more flexible mediation dynamic control
- Summary of new function
  - Dynamic Properties
  - Mediation Policies and the Policy Resolution Primitive



## ***New Function for Policy-Driven Mediation: Policy Processing***

- Mediation Policies
  - Are simple property sheets
  - Are attached to a mediation module
  - Can have *conditional* attachment
    - each policy only applies when its condition is met
    - The condition can be a query on the contents of the message
  - Are authored and stored in WSRR
- Policy Resolution Primitive
  - Interacts with WSRR to retrieve the policies that have been attached to the module
  - Evaluates the mediation policies that are attached to the current module (including any conditions) and resolves them down to a single 'effective policy'
  - Converts the effective policy into a set of dynamic property values, and propagates them along the mediation flow
  - This means that the effective policy determines the values of the dynamic properties





# Additional Information on WESB V6.1

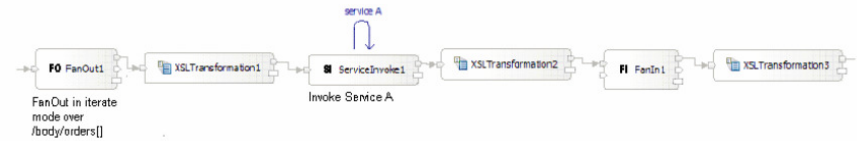


# What's New in WebSphere Enterprise Service Bus

## v6.1

- Supports a broader set of mediation patterns quickly and with reduced development effort

- ▶ New message splitting and aggregation patterns
- ▶ New service retry capability
- ▶ Updated Business Object Mapper and relationship support

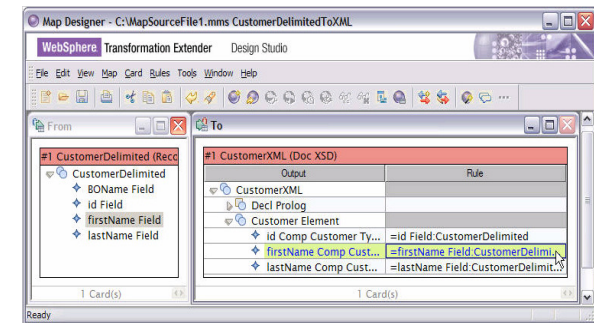


- Integrates a broader range of services with expanded connectivity

- ▶ New WebSphere TX integration
- ▶ New WS-Notification support
- ▶ New generic HTTP support
- ▶ Enhanced 3rd party JMS support

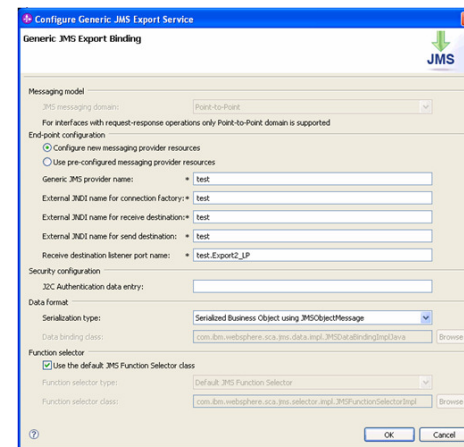
- Enables streamlined operational infrastructure with platform currency

- ▶ Updated WAS 6.1 based runtime
- ▶ New and expanded OS platforms, including 64-bit exploitation and i5/OS support



- Improves consumability and usability across the solution lifecycle

- ▶ Easier server installation and configuration
- ▶ Expanded XML and WSDL support



## WebSphere ESB 6.1.2 Available 1 Aug 2008

- **Enhanced support for Web Services Description Language (WSDL) XML Schema Definition (XSD), enabling the use of many industry-standard XML schemas.**
- **Improved error messages, logging, and First Failure Data Capture (FFDC) usage. FFDC can reduce defect resolution times and the number of times IBM Support asks a client to recreate a problem with different trace settings turned on.**
- **Adds support for manipulating MQCIH message headers for WebSphere MQ, which are exploited by the CICS bridge to enable interaction with CICS applications.**
- **Adds new format support for the following additional message formats:**
  - ▶ Delimited and full support for Comma Separated Values (CSV) Fixed-width format, and JavaScript Object Notation (JSON)
  - ▶ Enhances support for COBOL Copybook, C Struct, and PLI







# ***WebSphere ESB Core***

## **Fundamental Services and Connectivity**

**WebSphere** software



## HTTP Export / Import [New]

- HTTP 1.0 and 1.1
- SSL over HTTP
- Request/Response invocation
- Static header setting in WID and Admin
- Dynamic header setting via SMO in mediation modules
- Binary, XML and SOAP payloads
  - Plus custom data bindings
- Custom HTTP methods in Import
- Specifiable Content and Transfer encodings
- Endpoint based routing in Export

The screenshot displays the IBM WebSphere IDE interface. At the top, a context menu is open for the component 'CustomerOrderImport'. The menu items include: Undo Update Display Name, Redo, Add Interface, Generate Binding... (highlighted), Remove Binding, Copy, Paste, Delete, Rename, Select All, Wire References to New, Wire to Existing, Wire (Advanced) ..., Test Component, and Show in Properties. A sub-menu is open for 'Generate Binding...', listing: HTTP Binding (highlighted), SCA Binding, Stateless Session Bean Binding, and Web Service Binding.

Below the context menu, the 'Properties' view is visible, showing the configuration for 'Import: CustomerOrderImport (HTTP Binding)'. The configuration fields are:

Endpoint URL:	http://temp.url
Data Binding:	com.ibm.ws.sca.databinding.impl.DataBindingImplXML
HTTP Version:	1.1
HTTP Method:	GET
Binding description:	

## Generic JMS Export / Import Bindings [New]

- Simplify definition of JMS Export/Import binding
  - For 3<sup>rd</sup>-party JMS 1.1 ASF-compliant providers
    - Oracle AQ, TIBCO, SonicMQ, WebMethods, BEA WebLogic, etc.
  - Avoids Deployment Descriptor editing
  - Automatic setup of WAS Generic JMS resources
    - Manual setup of provider's JMS resources
- Admin visibility and modification of binding properties
  - Now available across all JMS and MQ bindings
- Dynamic JMS header updates via SMO
- Works with existing JMS Data Bindings

Configure Generic JMS Import Service

Generic JMS Import Binding

JMS

Messaging model

JMS messaging domain: Point-to-Point

For interfaces with request-response operations only Point-to-Point domain is supported

End-point configuration

Configure new messaging provider resources

Use pre-configured messaging provider resources

Generic JMS provider name: \*

External JNDI name for connection factory: \*

External JNDI name for send destination: \*

External JNDI name for receive destination: \*

Receive destination listener port name: test.Import3\_RESP\_LP

Security configuration

J2C Authentication data entry:

Data format

Serialization type: Serialized Business Object using JMSObjectMessage

Data binding class: com.ibm.websphere.sca.jms.data.impl.JMSDataBindingImplJava Browse

Function selector

Generate "TargetFunctionName" message header property for default JMS Function Selector

OK Cancel



## ***WTX Data Bindings [New]***

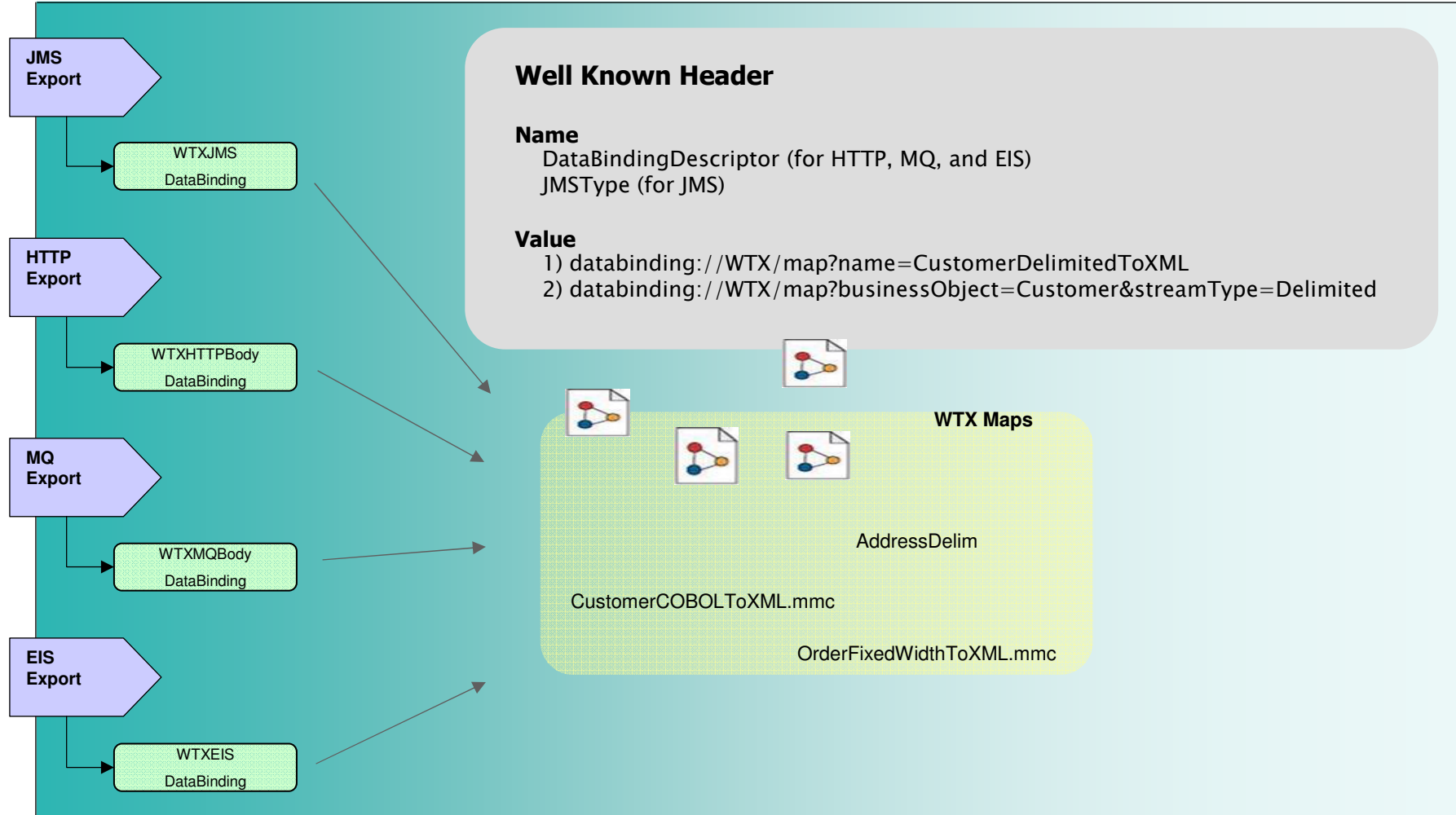
- WebSphere Transformation Extender Integration
  - Tooled Data Bindings
  - Semi-structured to Business Object Transformations
- WPS/WESB & WTX
  - WPS/WESB will provide the data bindings
  - WTX 8.2 must be installed in order to use the data binding
- Exports & Imports
  - JMS, MQ, HTTP, EIS
  - Flexible bindings (any data any format)



# WTX Data Bindings

## Any Business Object, Any Format

SCA Module





## *Mediation Updates*

New and changed mediation primitives  
Service Message Object Updates

**WebSphere** software

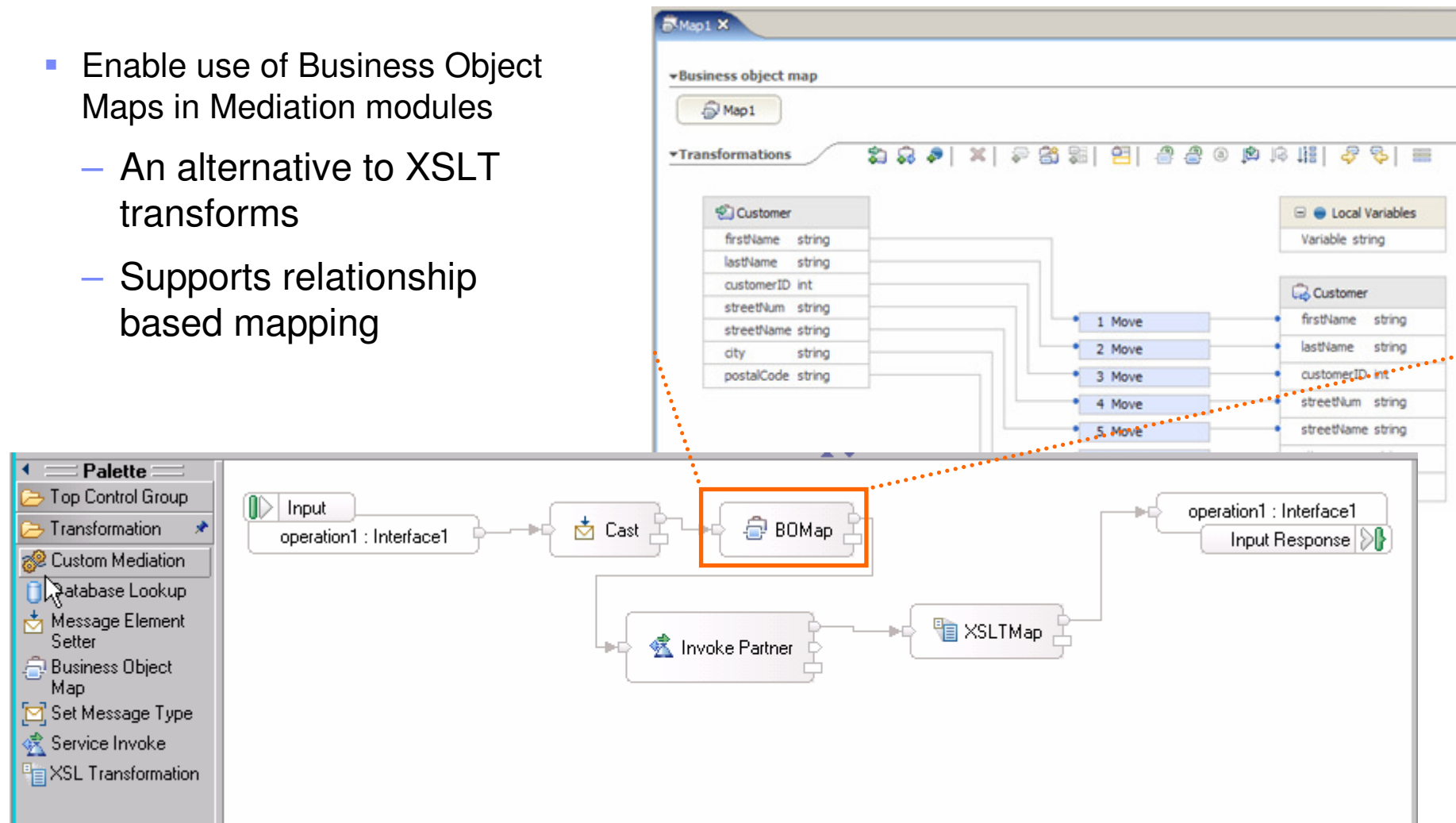




## WESB Enhancements

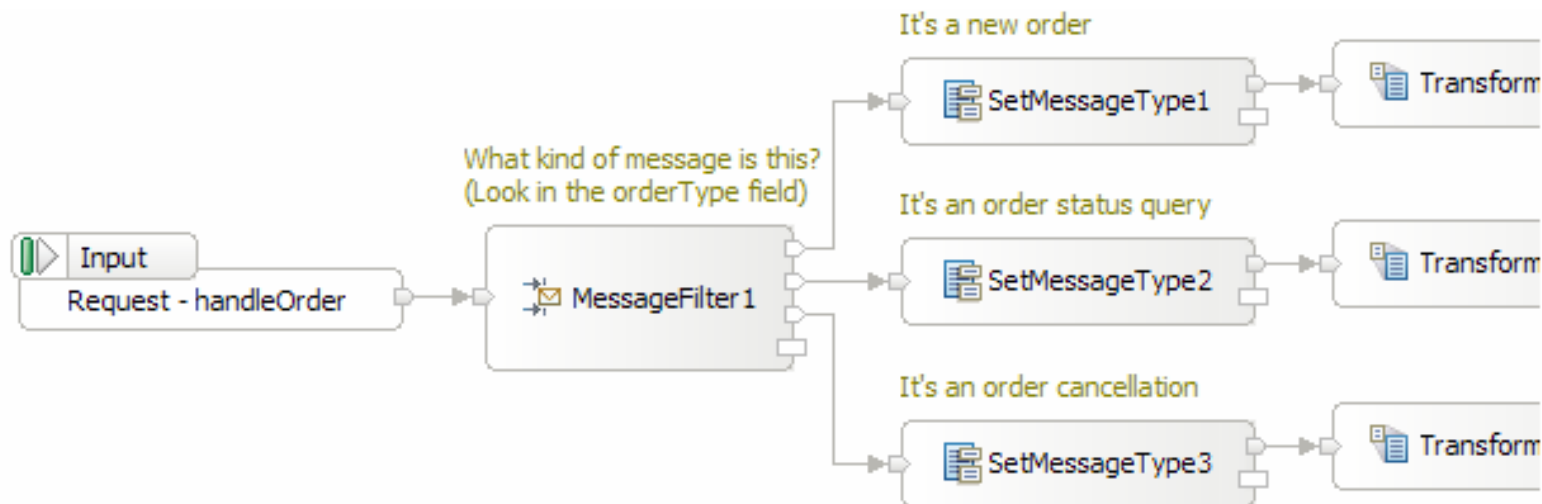
### BO Map Primitive

- Enable use of Business Object Maps in Mediation modules
  - An alternative to XSLT transforms
  - Supports relationship based mapping



## Mediation Flow Component Type Refinement [New]

- Ability to assert concrete types for message elements described by 'weak types' such as anyType, anySimpleType or any
- New SetMessageType mediation primitive
  - Conceptually performs a 'cast' operation on any part of the message
  - Works with other primitives and XPath support to provide full access to weakly typed message content in mediation flows
  - Weakly typed content can be visualized, mapped and manipulated as if its structure were fully defined in the original BO definition



# WESB Enhancements

## Service Invocation and Retry Primitive

- Invokes a target service from within a request or response flow
- Includes built-in retry capability; retries x times in event of failure
- Can try/retry a list of target endpoints in turn until success
  - Can exploit multiple endpoints returned from the Endpoint Lookup primitive for this purpose
- Also acts as a building block for aggregating content from more than one service
- Capability similar to existing Callout, but flow continues inline (no switch to response flow)
- New context area in the SMO contains invocation request/response body content
- Service A invocation can be asynchronous

**Service Invoke : Invoke Partner**

Reference Name:

Operation Name:

Retry On:

Retry Count:

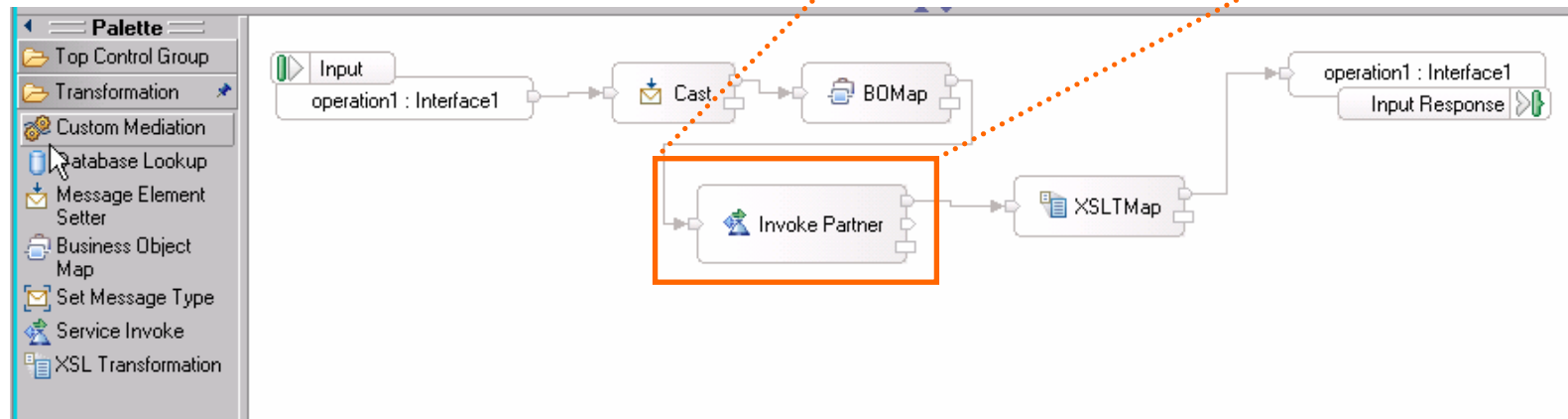
Retry Delay:

Use Dynamic Endpoint

Try Alternate Endpoints

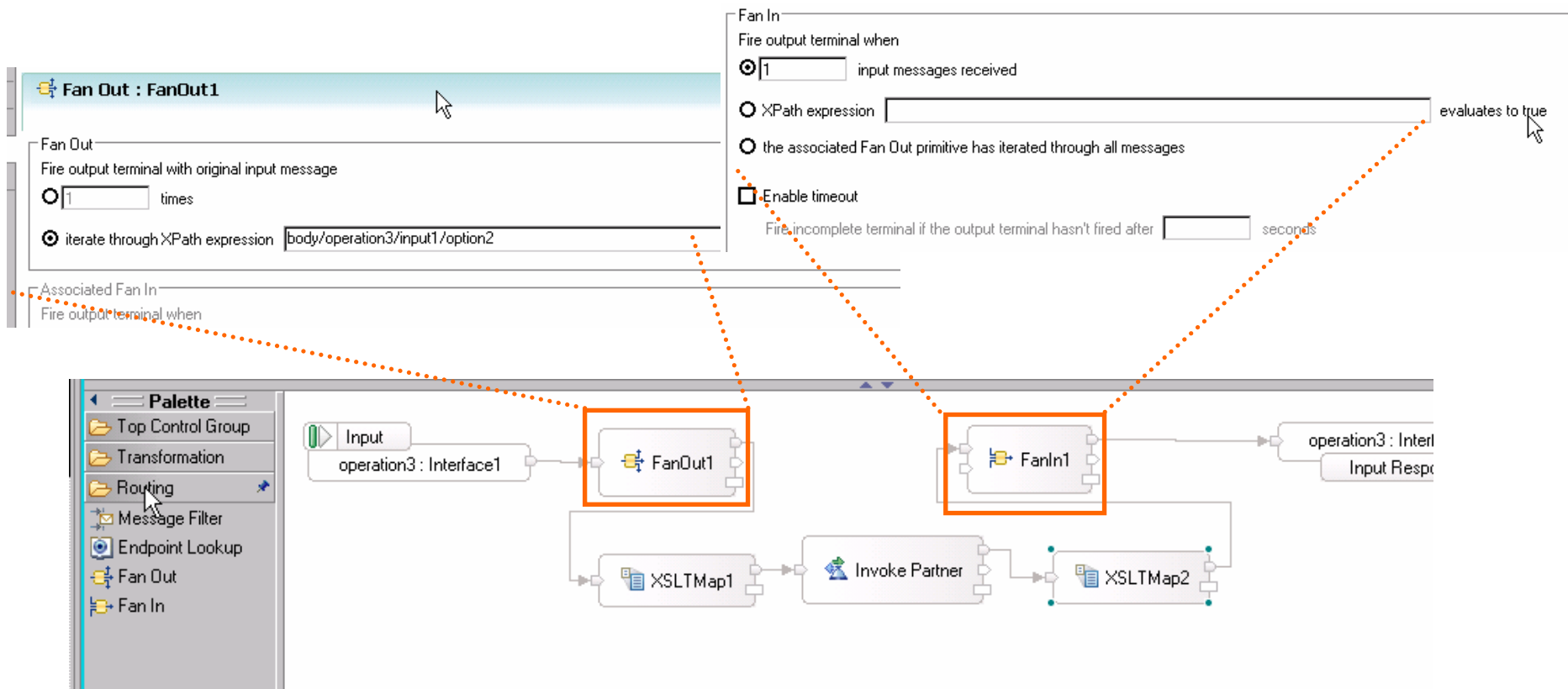
Async Timeout:

Force Sync



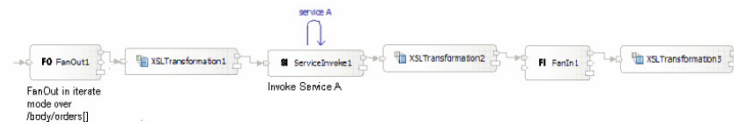
## New: FanOut and FanIn Mediation Primitive

- Messages can be split for further processing
- Results can be aggregated

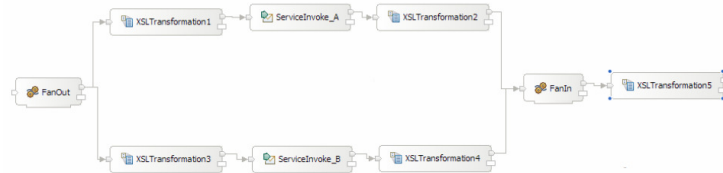


## Splitting and Aggregating - details

- New FanOut and FanIn primitives
  - FanOut splits an incoming message based upon a repeating element



- ...or sets up branching paths that are later joined by FanIn



- Allows composite messages to be split up for individual processing of the parts; and assembly of composite messages
- Aggregation supported using ServiceInvoke primitive
  - Invoke multiple services and combine the results



# WESB Enhancements

## Custom Mediation Primitive

- Multiple input and output terminals
- User-defined properties
- Easy access to ESB mediation primitive programming model

**Custom Mediation : CustomMediation1**

CustomPropertyGroup CustomUserProperties CustomJavaImports

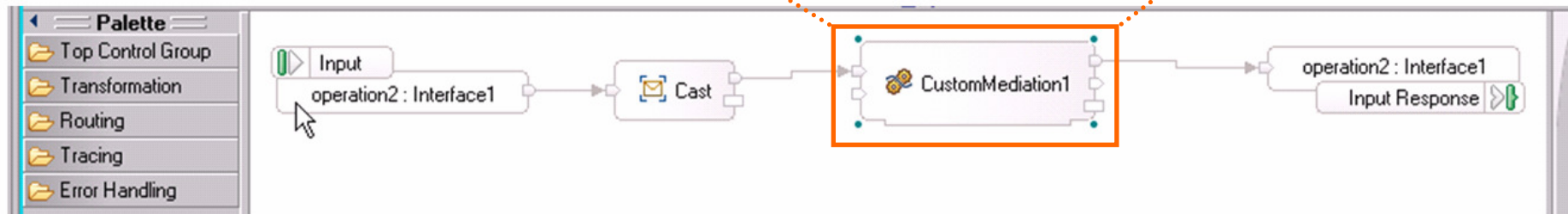
User Properties:

Name	Type	Value	Required
frequency	Integer	5	<input checked="" type="checkbox"/>

Implementation:  Visual  Java

```

/**
 * Variables: for output terminals - out1, out2 (com.ibm.wsspi.sib)
 *            for user properties - frequency (int)
 * Inputs:    inputTerminal (com.ibm.wsspi.sibx.mediation.InputTerm)
 * Exceptions: com.ibm.wsspi.sibx.mediation.MediationConfigurationE
 */
// Fire the output terminal(s)
out1.fire(smo);
out2.fire(smo);
    
```







## ***ND Topology Configuration***

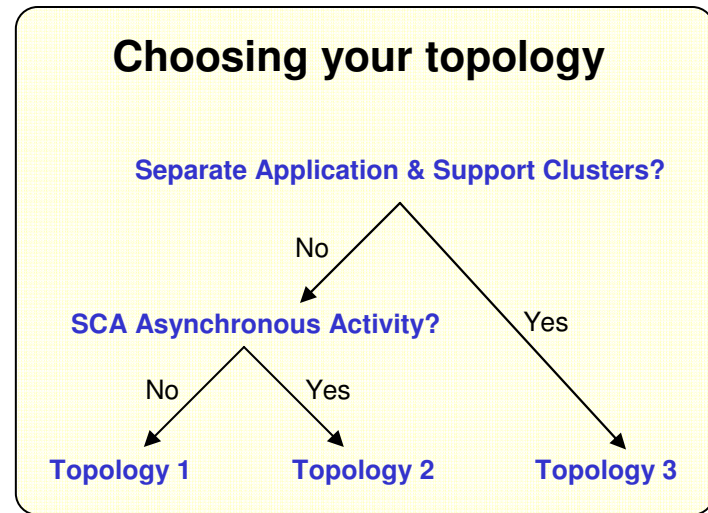
**Ease of Use Updates**

**WebSphere** software

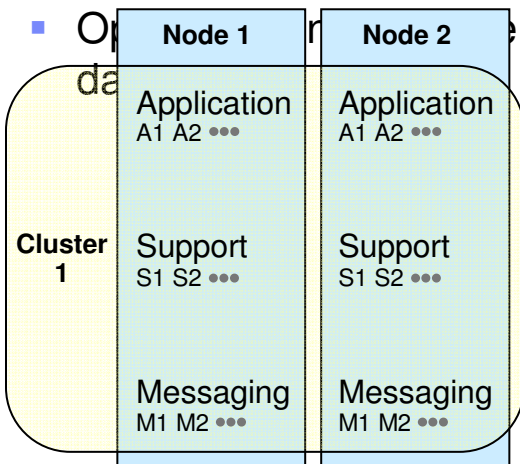


# Template-driven ND topology

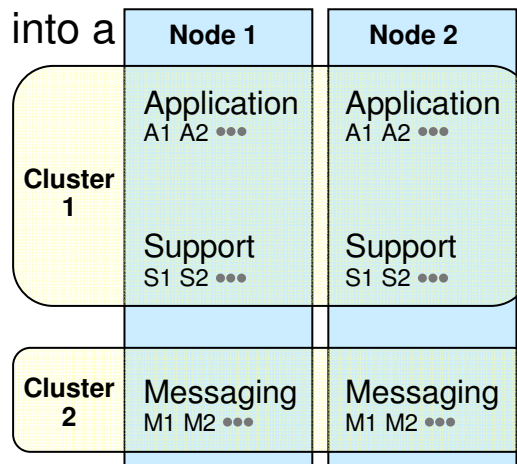
- Wizard driven approach to configuring your ND topology
  - Install support
  - Administration support
- Three primary roles nodes can play
  - Customer applications
  - WPS/WESB Support applications
  - Messaging (Destinations)



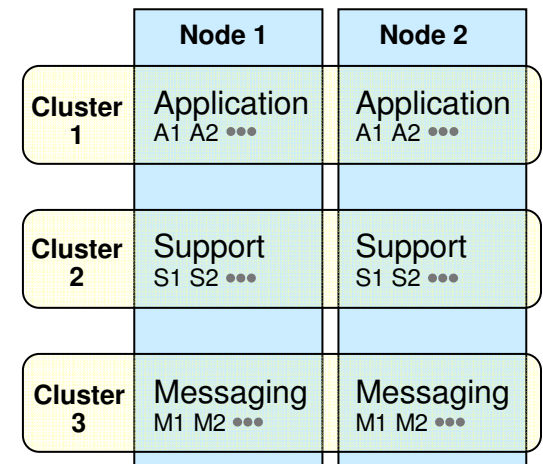
■ Or data everything into a



Topology 1: Single Cluster



Topology 2: Remote Messaging



Topology 3: Remote Messaging and Support



## *Summary and Conclusion*

**WebSphere** software



## ***Conclusion***

- Key evolution in 6.1, centering on
  - Consumability
    - both administration and development interfaces
  - Completeness of capability
    - Additional connectivity
    - Additional mediation function
- Continued maturation and evolution



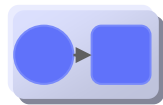
## **WebSphere ESB for z/OS**

**Built on WebSphere Application Server for an integrated SOA platform**

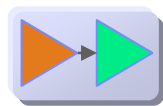
- Integrates seamlessly with WebSphere platform
- Delivers business-critical qualities of service
- Easily extended to WebSphere Process Server
- Integrated solution for service mediation and hosting



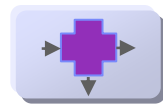
Delivers leadership in SOA standards for service composition, and leverages the embedded messaging and web services engines from WebSphere



Integrates everything with WebSphere Adapters for enterprise applications, the breadth of the WebSphere ecosystem, and support for standard protocols



Optimized for standard XML and web services formats, with basic support for other common formats



Provides business visibility with embedded event engine for Business Activity Monitoring solutions

