



IBM Software Group

2006 B2B Customer Conference

B2B – Catch the Next Wave

Build your SOA environment using
WebSphere Process Server and
WebSphere Enterprise Service Bus

WebSphere. software

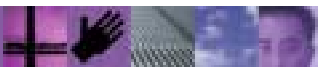
Ashutosh Arora

A decorative horizontal banner with a purple background, featuring a series of colorful squares (yellow, green, red) and various abstract icons (a globe, a person's face, a cross, a grid of circles).

ON DEMAND BUSINESS™

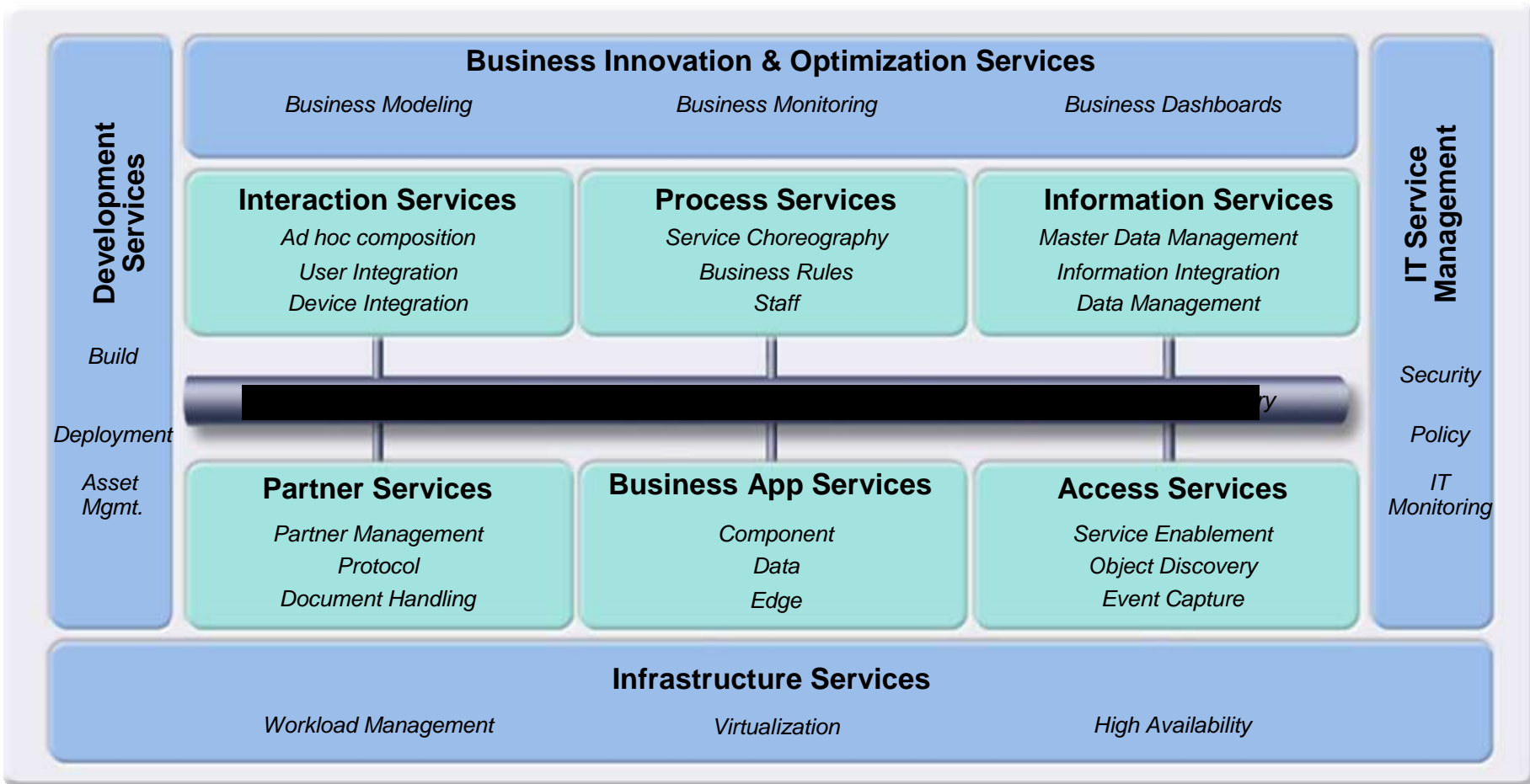
Objectives

- Understand SOA programming model
- Describe components of WebSphere Process Server and WebSphere ESB
- Understand how componentry provided by WebSphere Process Server and WebSphere ESB can be leveraged to build integration solutions



SOA Reference Architecture

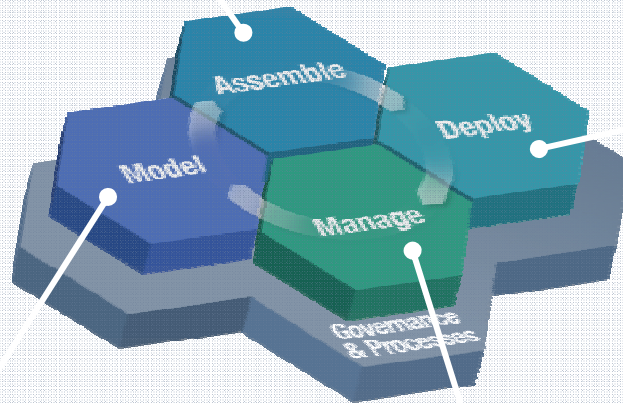
Comprehensive services in support of your SOA



IBM SOA Foundation

Part of a broader portfolio to meet your every need

WebSphere Integration Developer
Rational Application Developer



Process:

- WebSphere Process Server
- WebSphere ESB & Message Broker
- WebSphere Partner Gateway & Adapters
- WebSphere Data Interchange

People:

- WebSphere Portal
- WebSphere Everyplace Deployment
- Workplace Collaboration Services

Information:

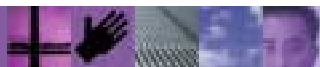
- WebSphere Information Integrator

Application Infrastructure:

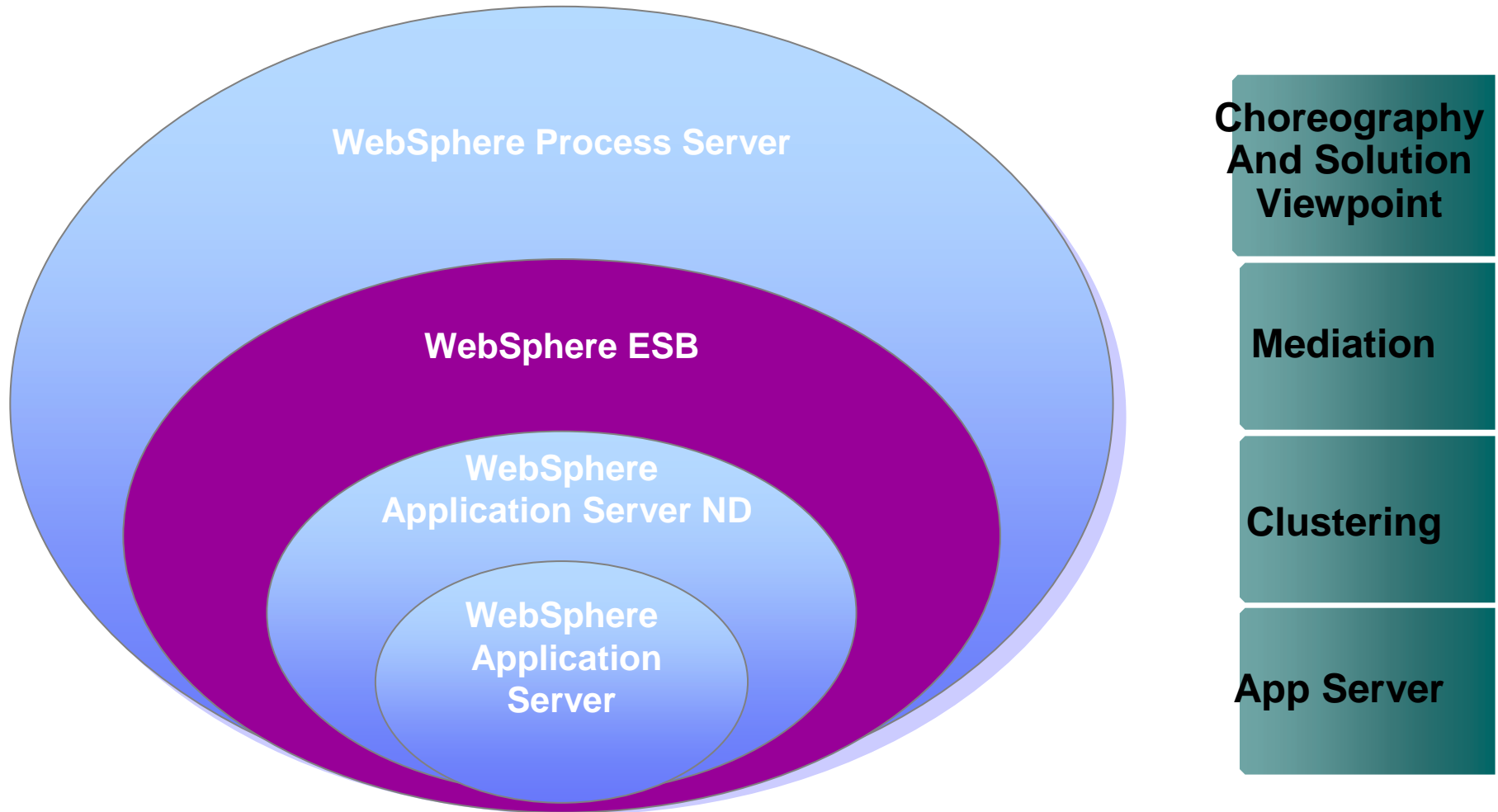
- WebSphere Application Server & XD

WebSphere Business Modeler
Rational Software Architect

WebSphere Business Monitor
Tivoli Composite Application Manager
Tivoli Federated Identity Manager
Tivoli Access Manager for e-business

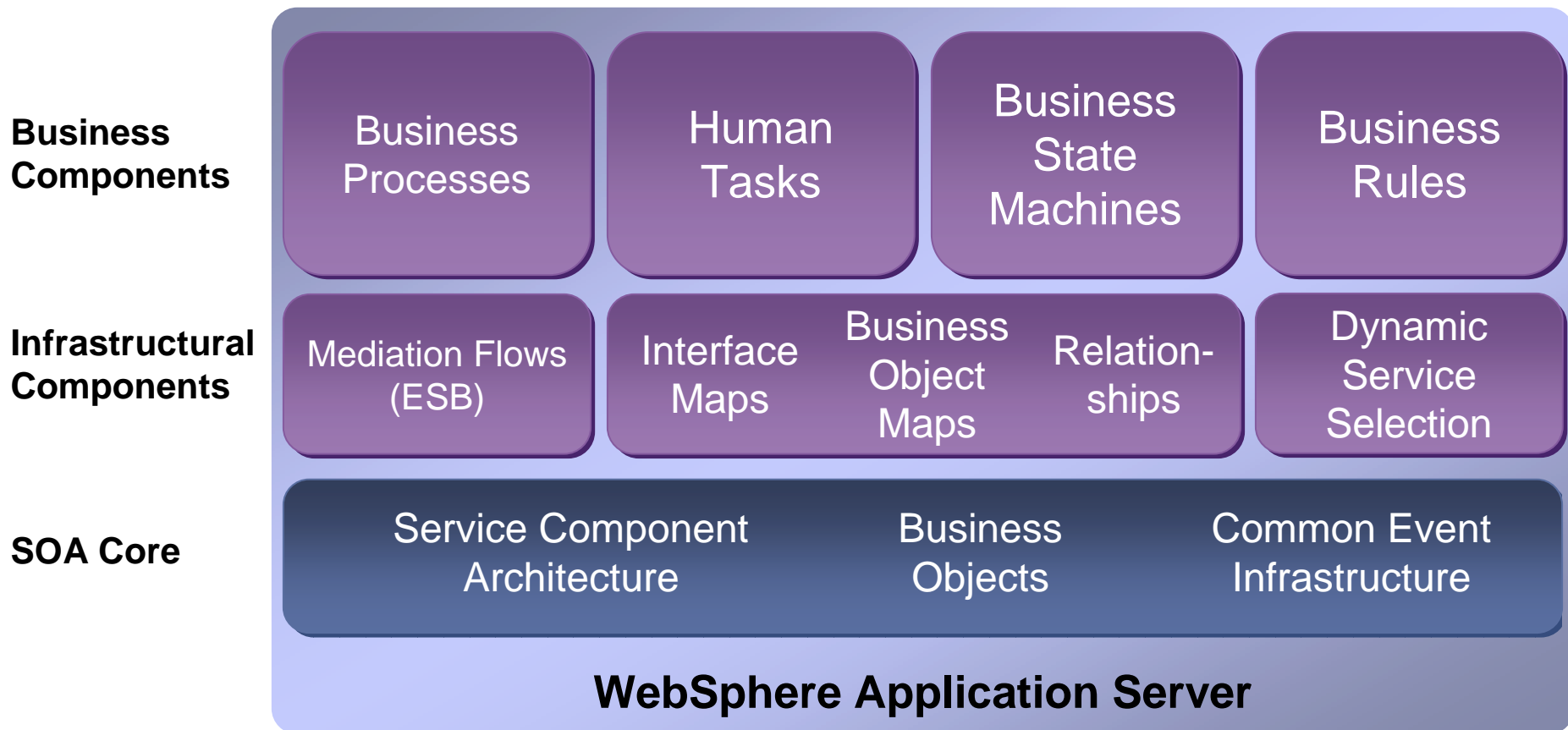


WebSphere Application Server, ESB, and Process Server



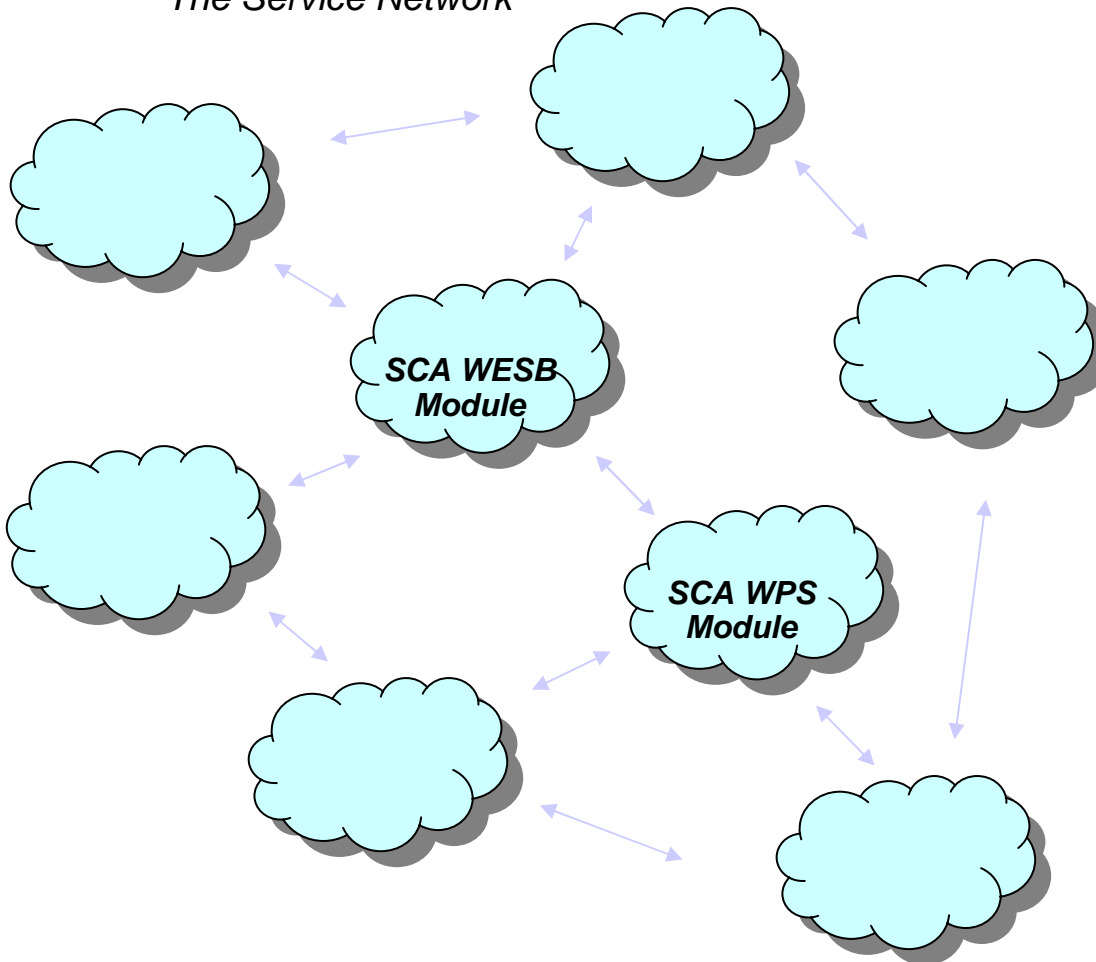
WPS/WESB 6.0x

Components



SOA & SCA

The Service Network

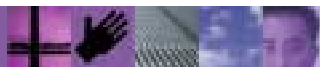


Service Oriented Architecture

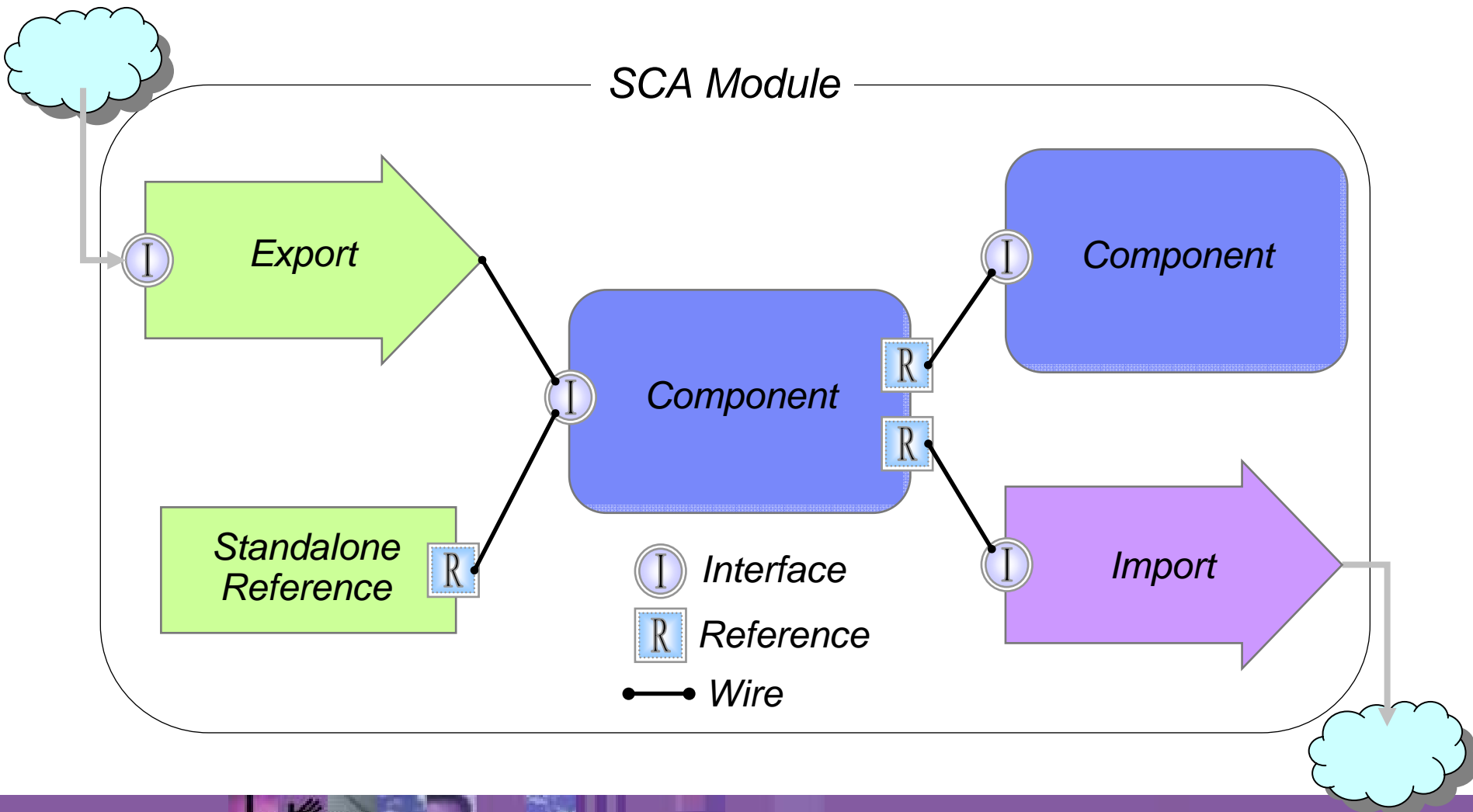
- Course grained services
- Disparate implementation programming models
- Disparate protocols
- Disparate formats
- Fabric / Service Bus
- Governance

Service Component Model

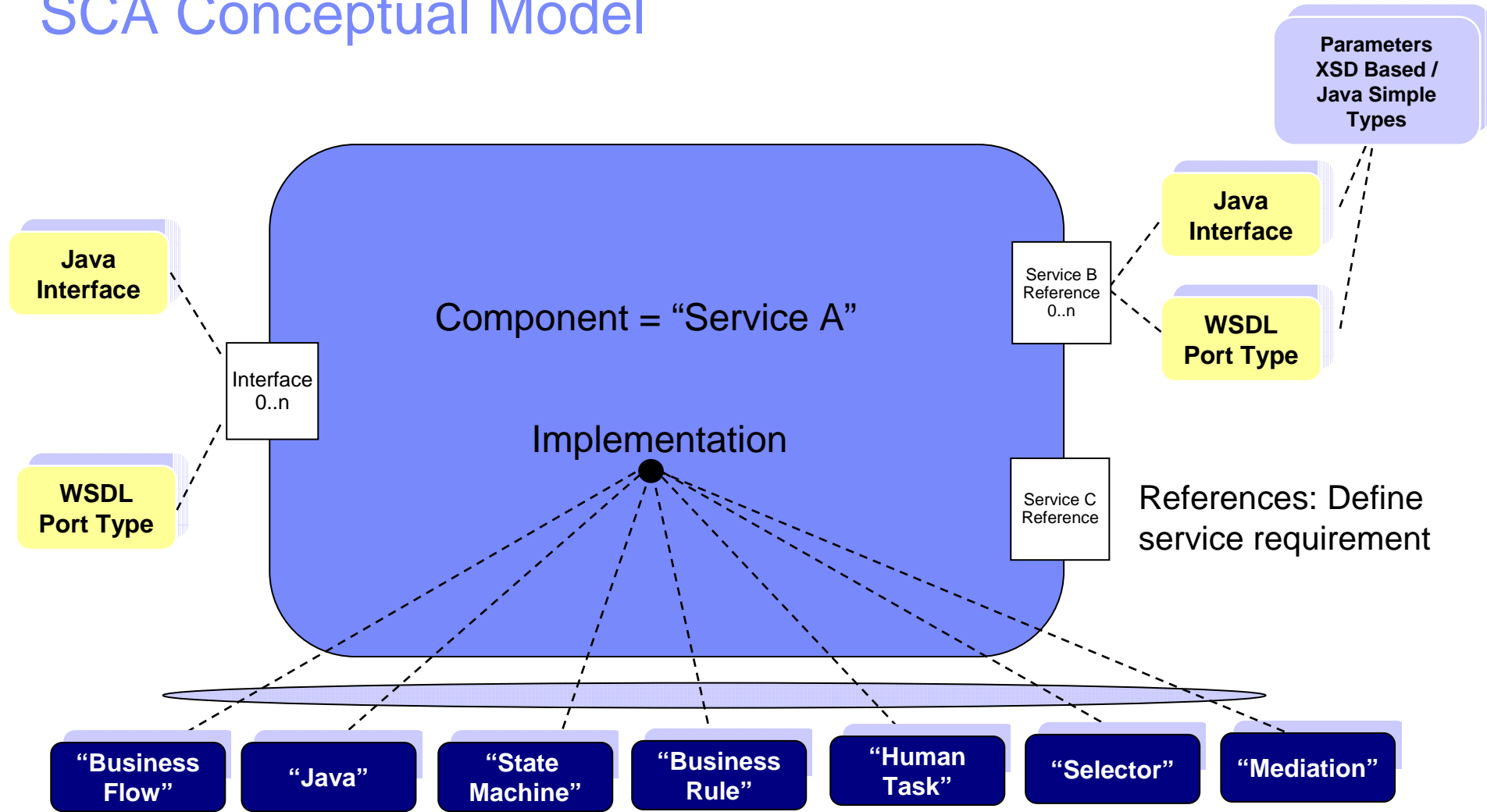
- Meta programming model for developing services
- Supports SOA notions



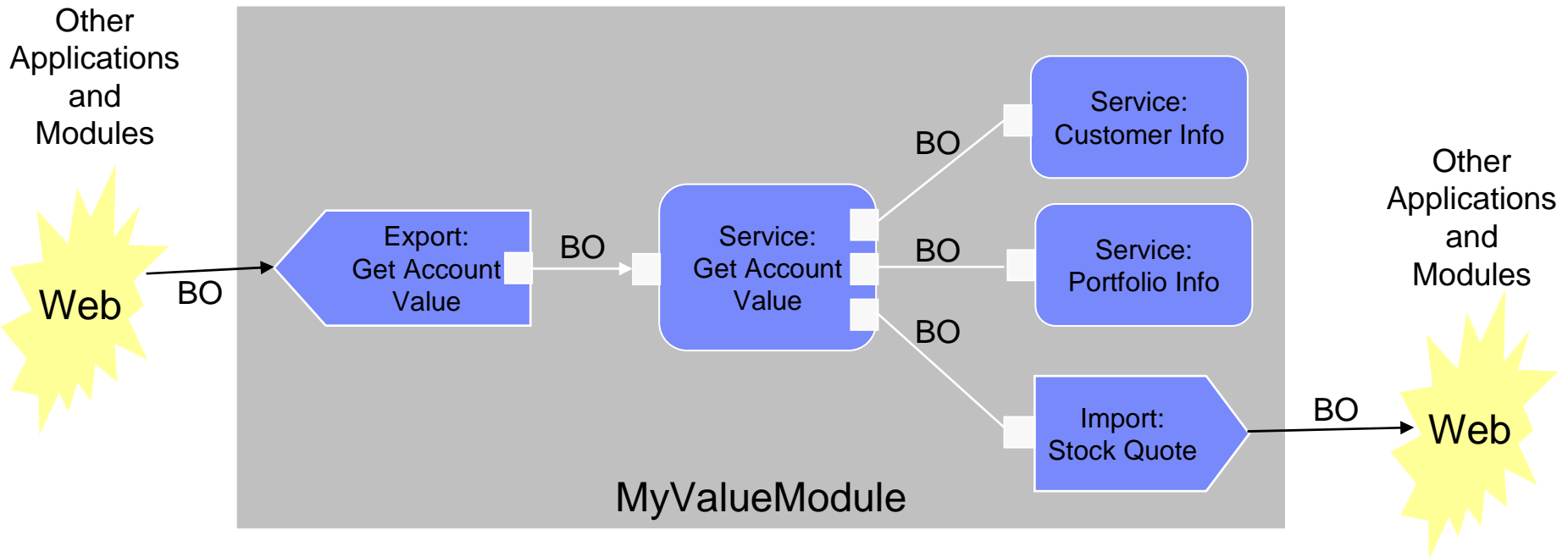
SCA Concepts Overview



SCA Conceptual Model



SCA and SDO/BO – Conceptual View

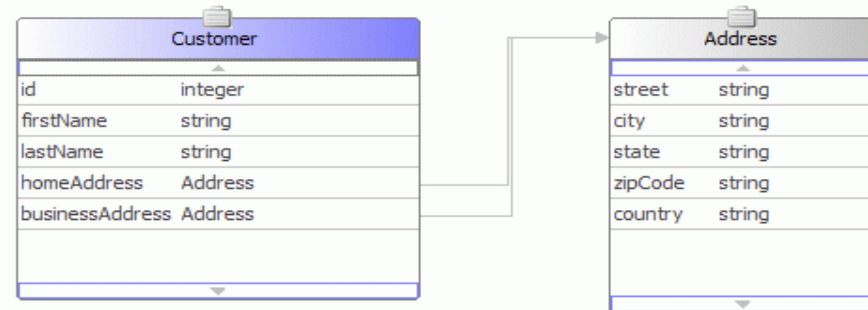


- SCA is the component model
- Components may be wired together
- Business Objects are the data flowing on wires between Components



Business Objects and Business Object Framework

- Enhanced Service Data Object
 - Provides some function not available in base SDO DO/DGs (close to SDO 1.1/2.0)
 - Provides functional equivalence to existing ICS Business Objects
 - Enables import of 'standard' XSD

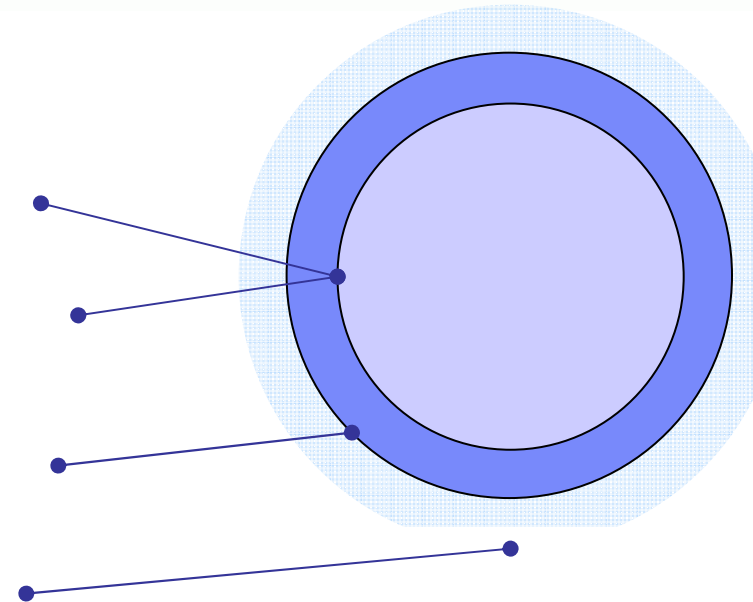


- Business Object Framework consists of:
 - Business Object definition

Business Object Metadata definition

Business Graph definition

Business Object Services

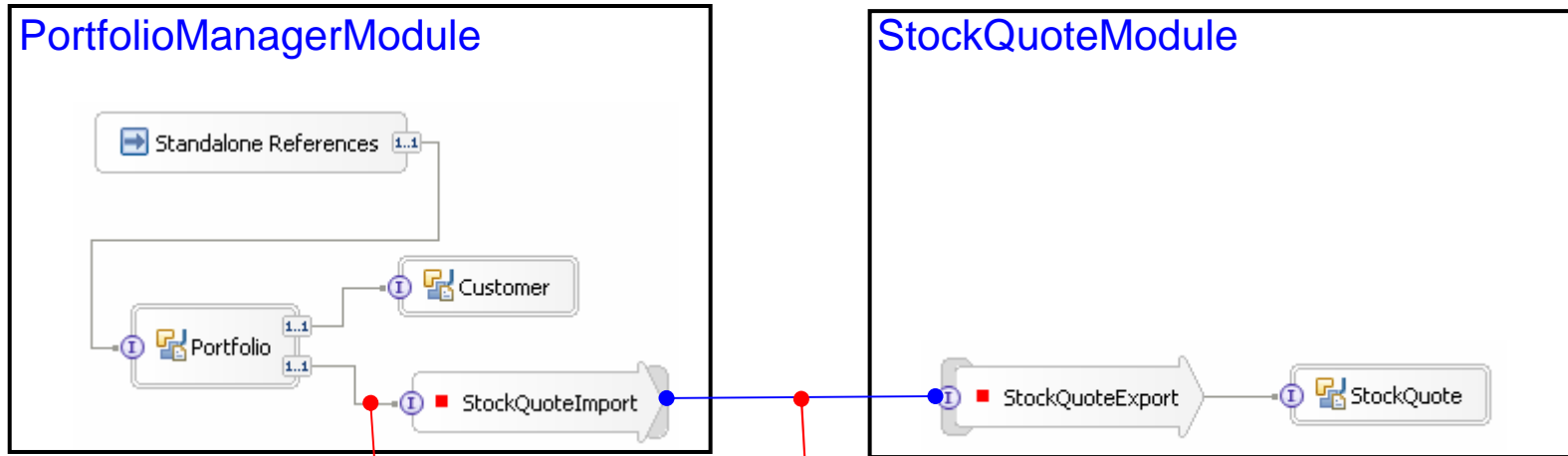


Provides support for disconnected use (change history), relationship integrity, dynamic interfaces, validation, rich meta-data

Provides support for both "Delta" & "After Image"



SCA Invocation Models



Invocation Models

Synchronous (**by value**)

Asynchronous – One Way (by value)

Asynchronous – Deferred Response (by value)

Asynchronous – Response with Callback (by value)

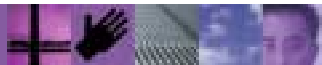
Invocation Models

Synchronous (**by ref**)

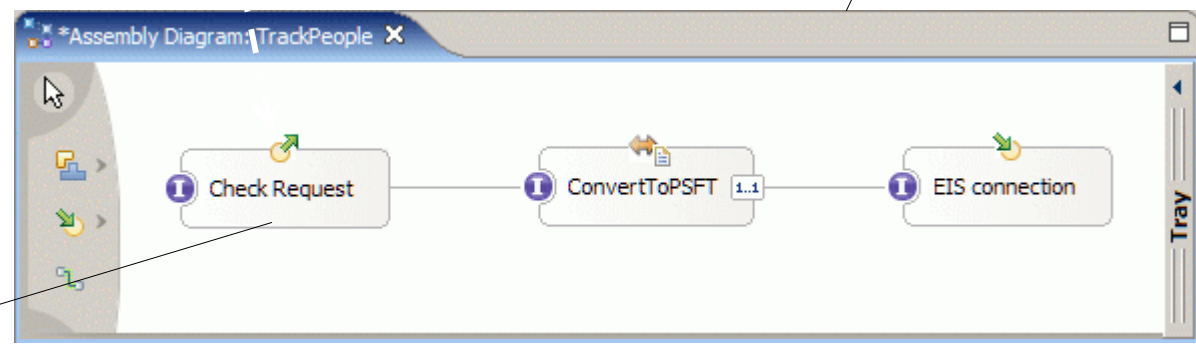
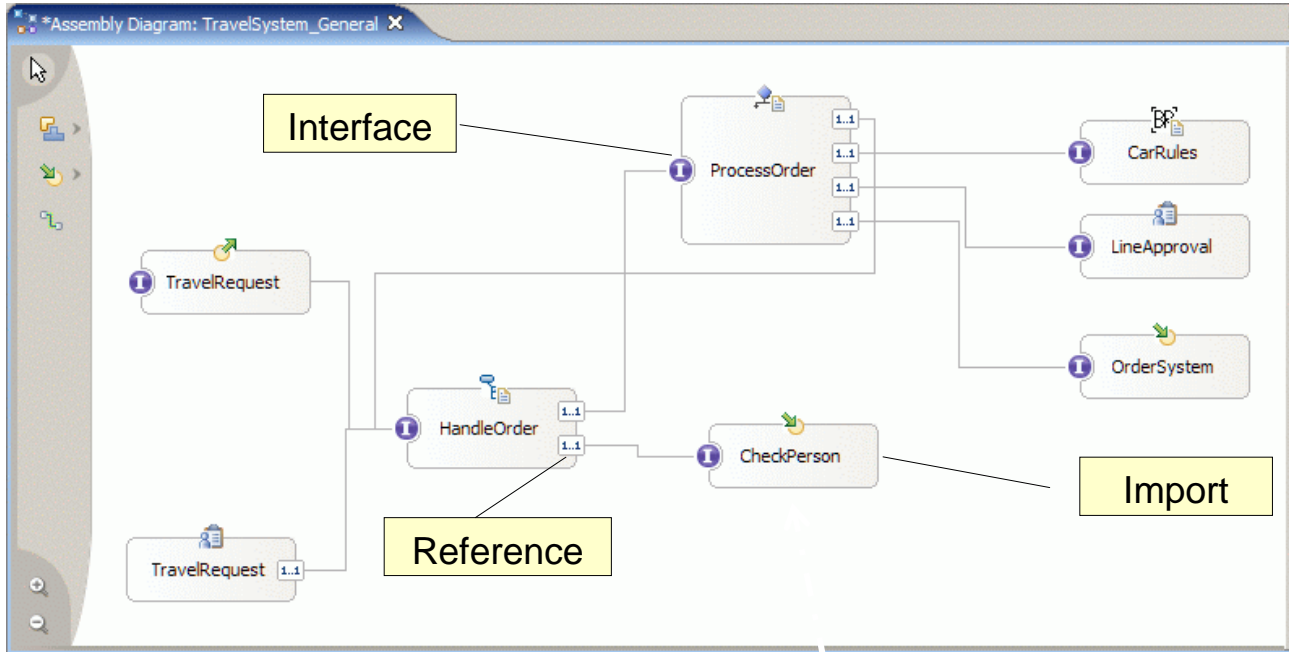
Asynchronous – One Way (by value)

Asynchronous – Deferred Response (by value)

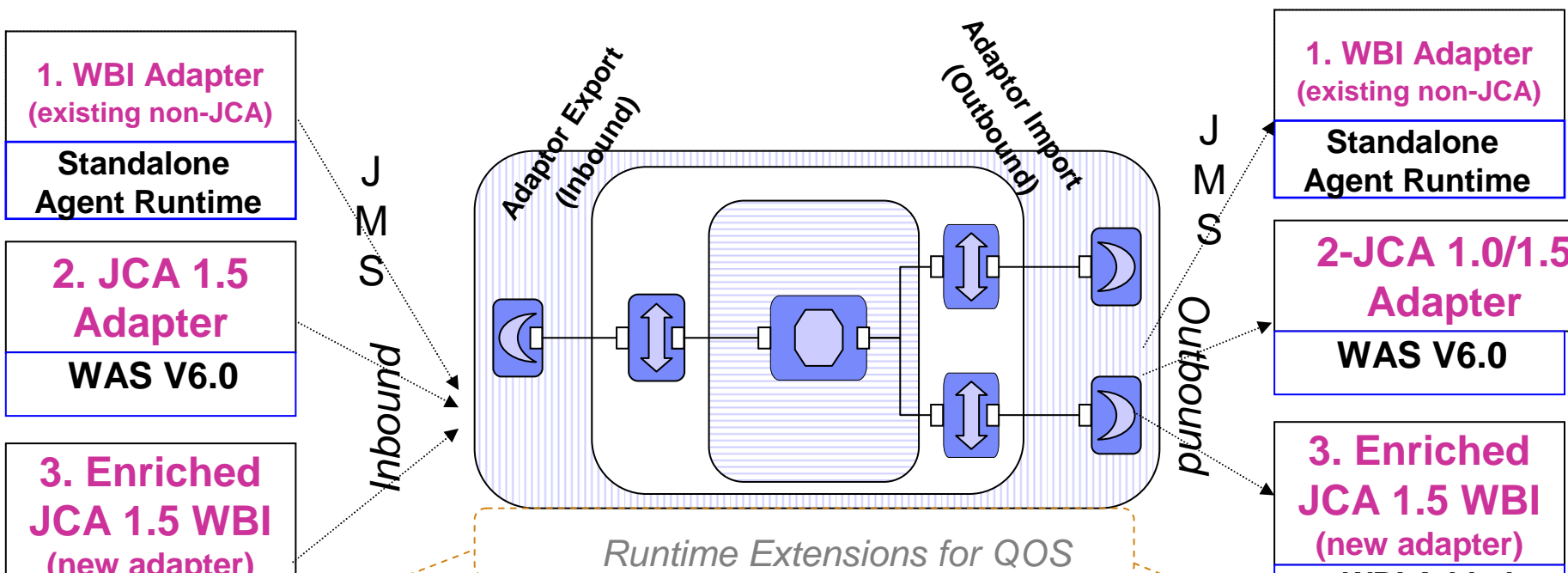
Asynchronous – Response with Callback (by value)



Assembly Editor



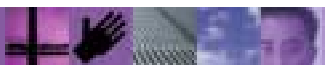
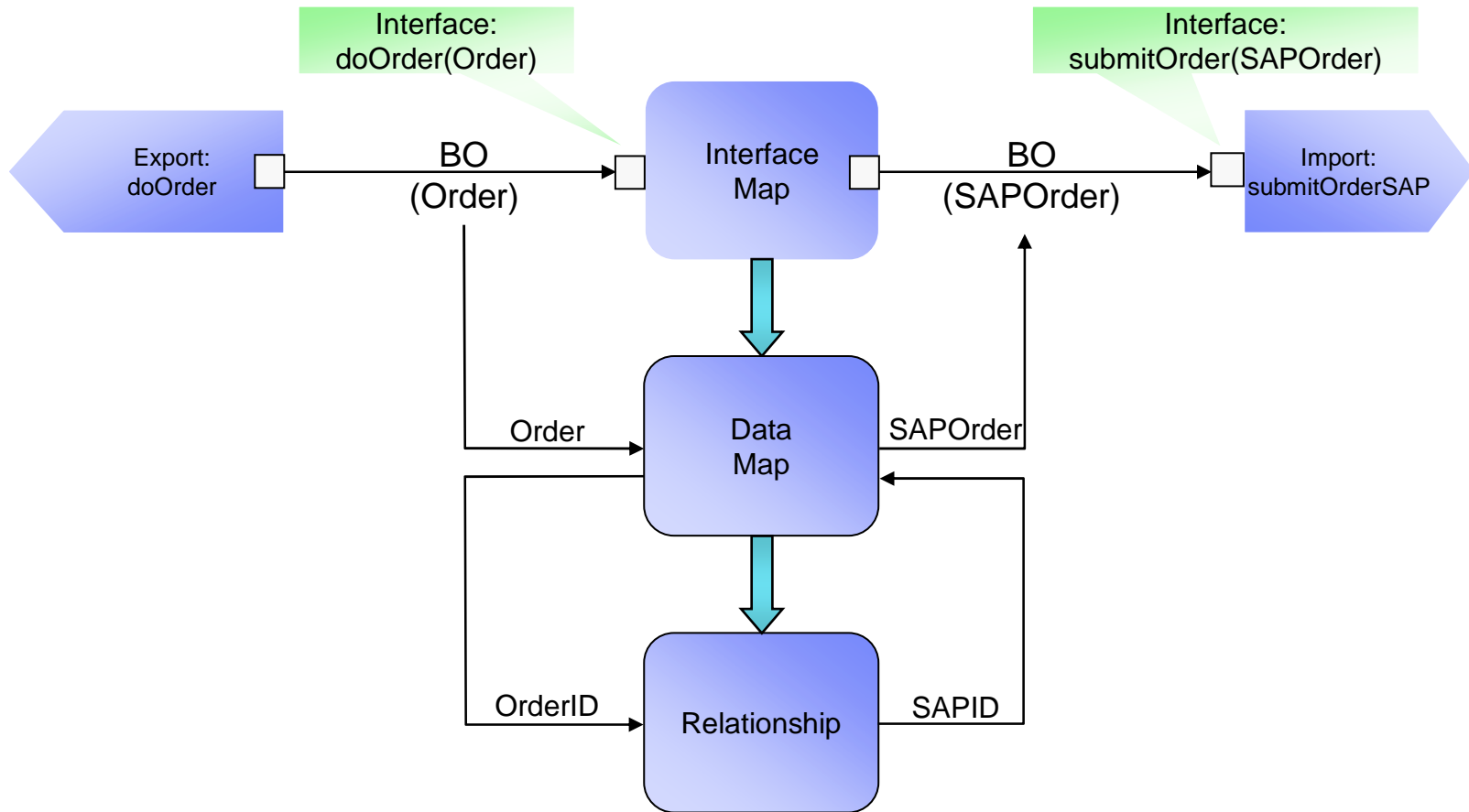
Adaptors and WebSphere Process Server 6.0



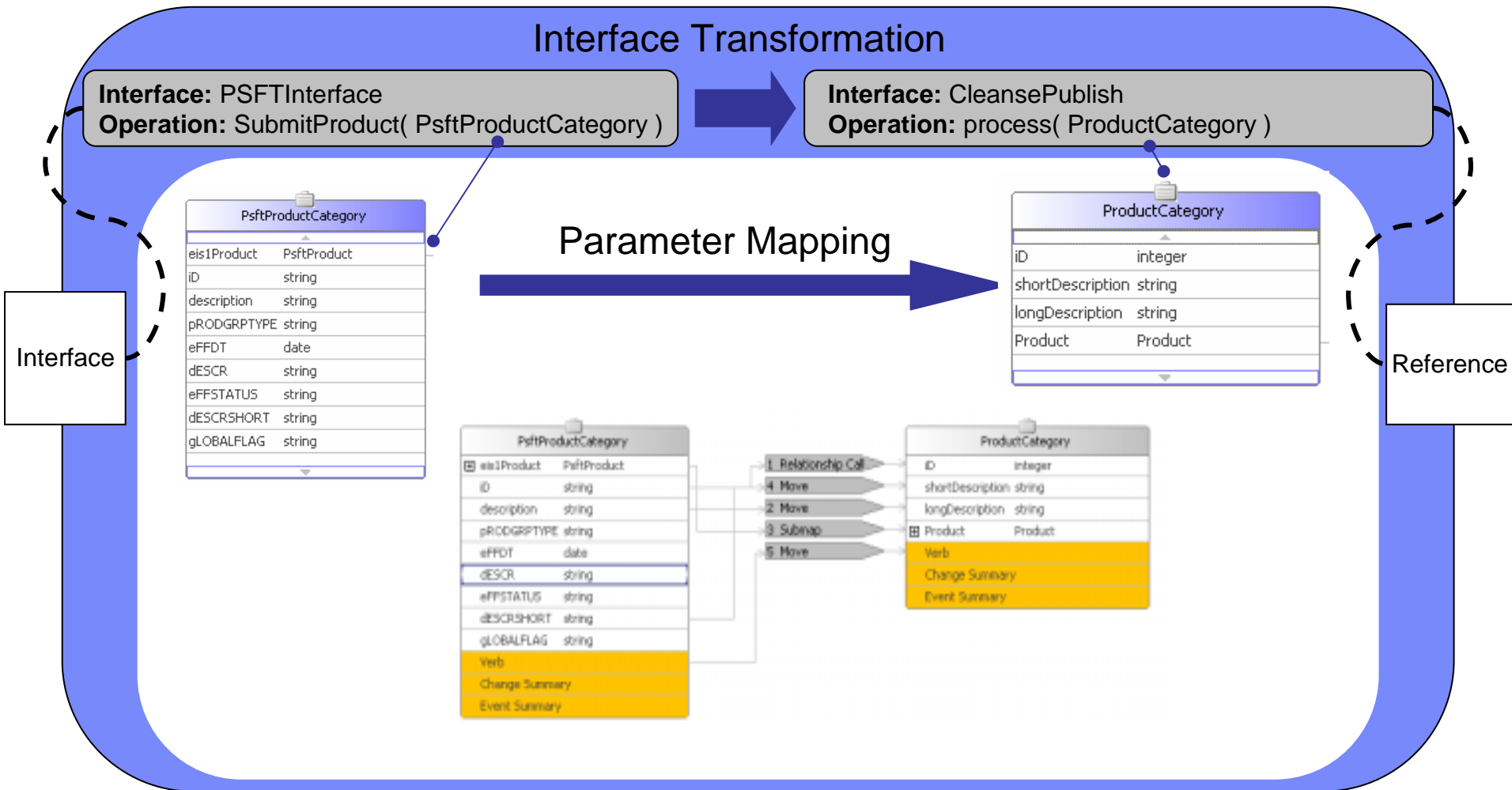
Separation of tasks performed:

1. The 'connector' – communications, QOS initiation, propagation, termination, etc.,
2. **Adaptor (SCA Import/Export)** – EIS inbound format to AsBO, AsBO to EIS specific outbound format
3. **WBI Mediation Component (SCA)** – convert AsBO to GBO, GBO to ASBO, Selection, etc..

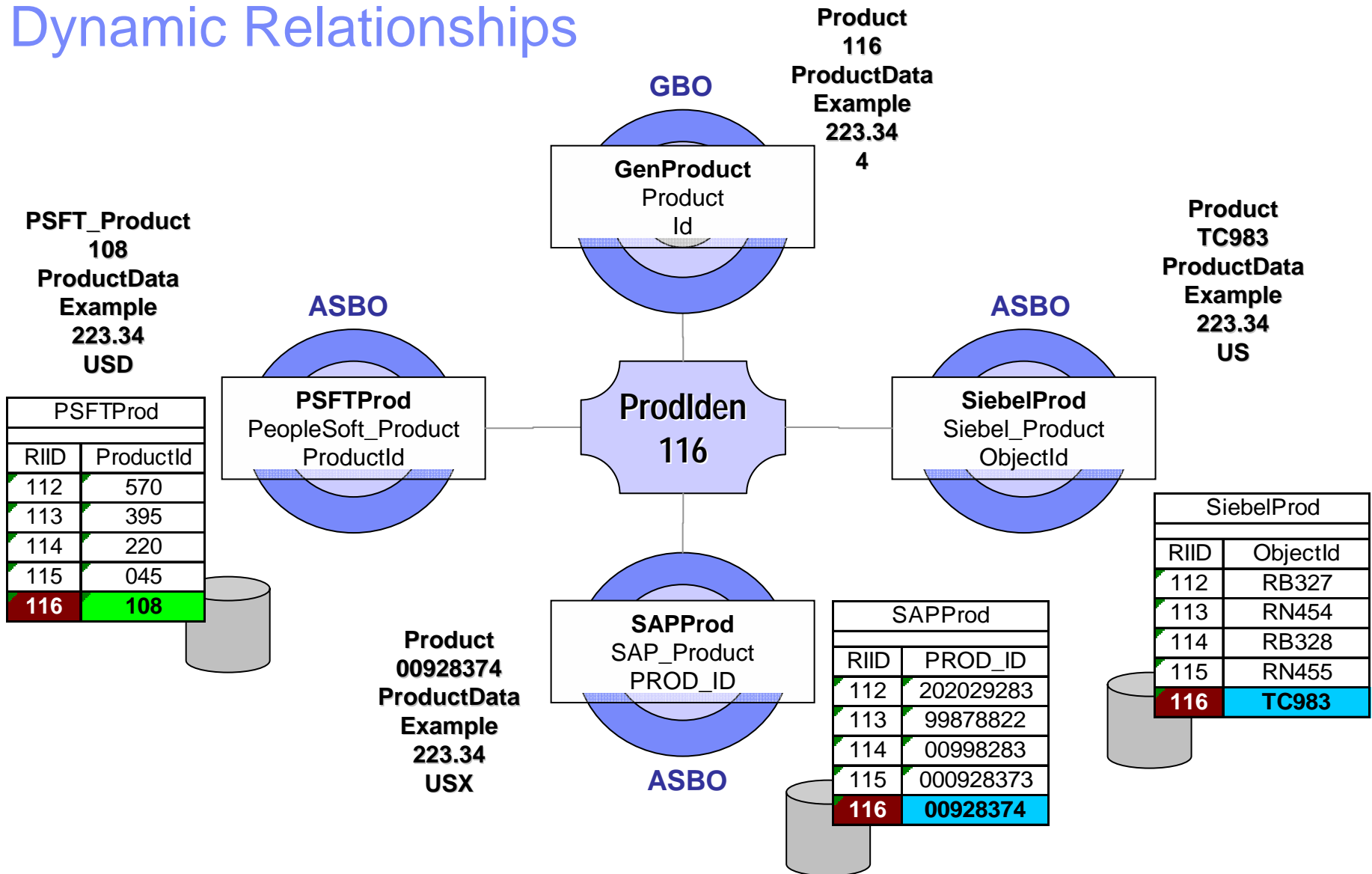
Transformation Components



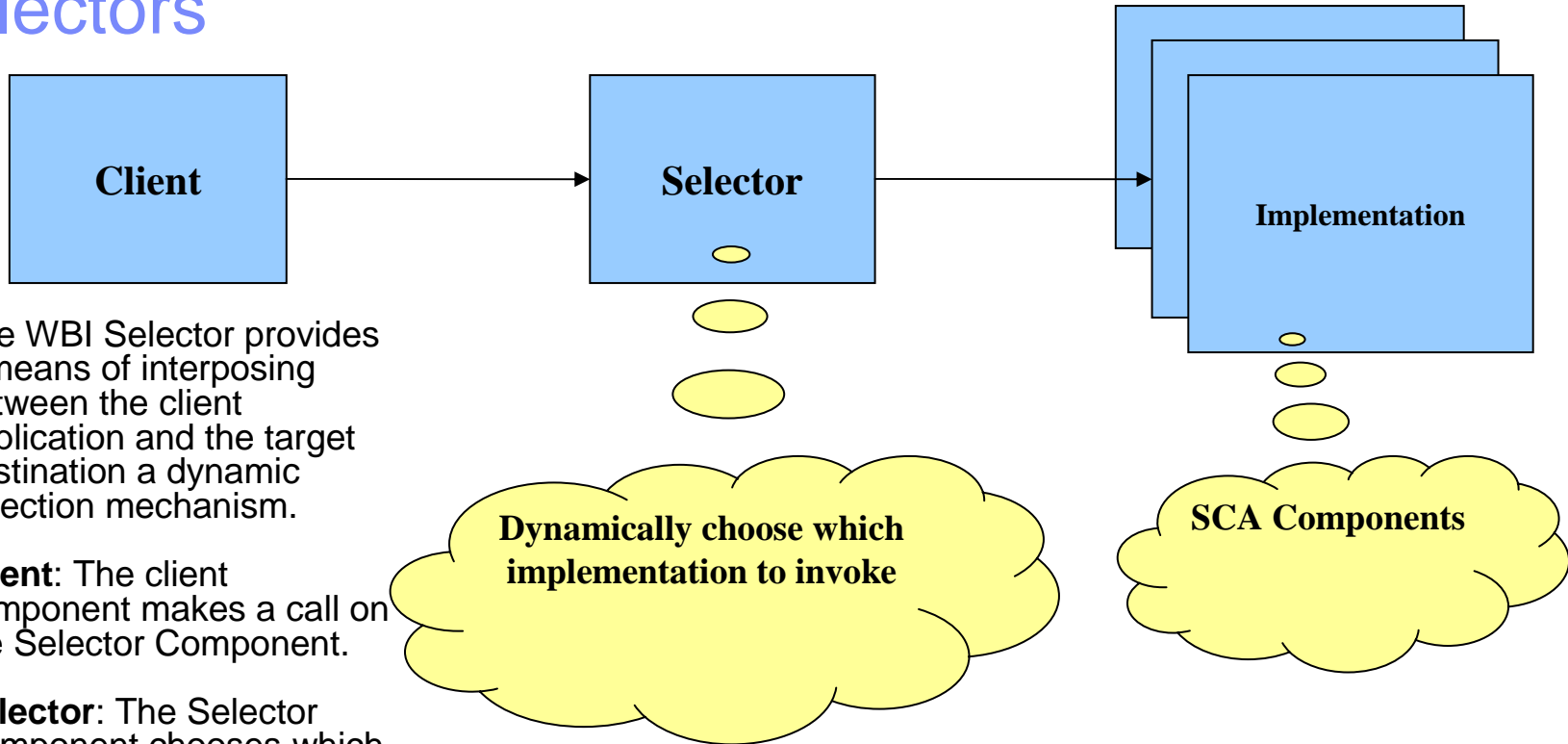
Interface Transformation as a SCA component



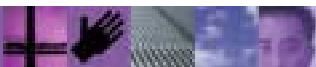
Dynamic Relationships



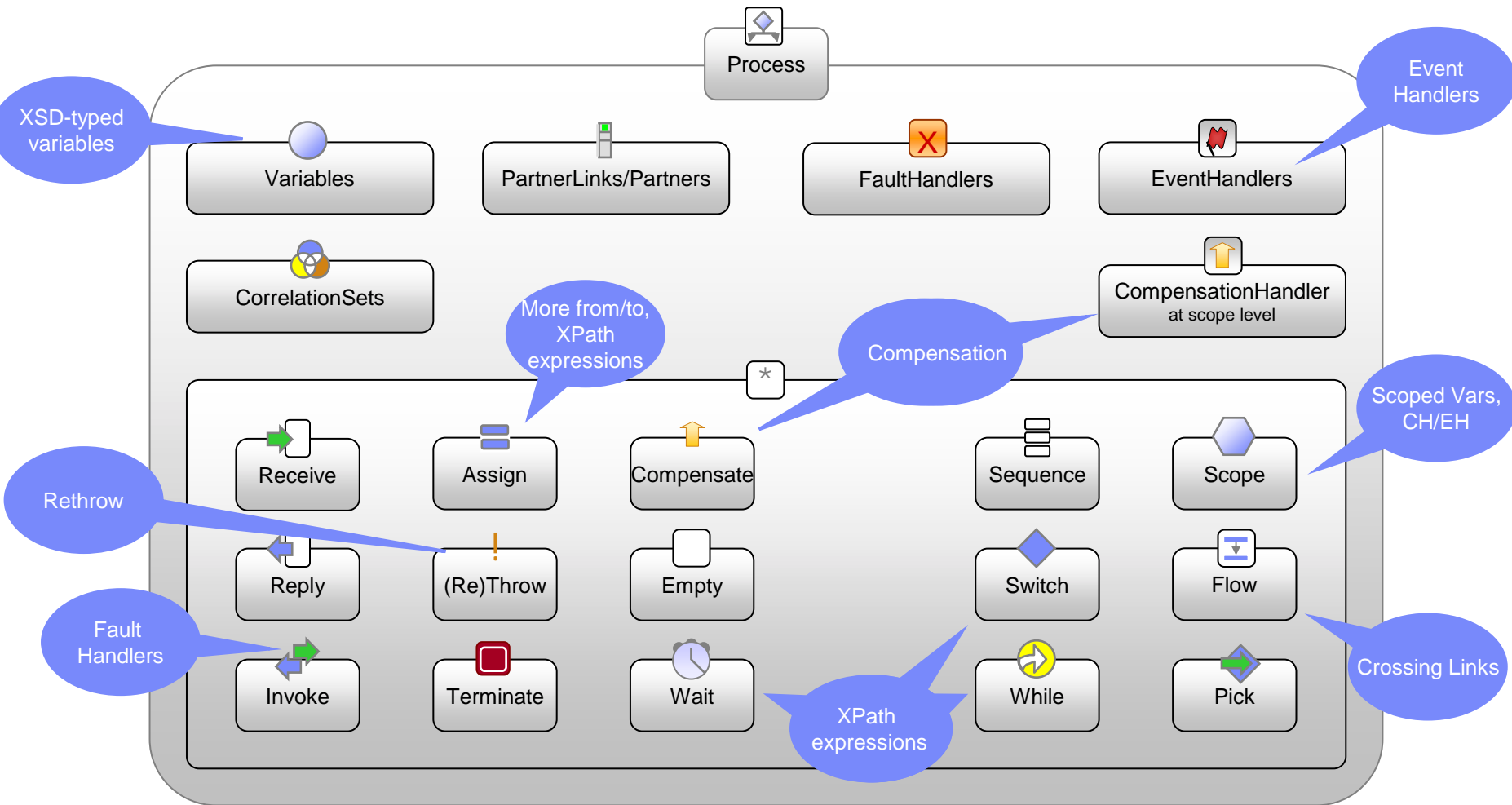
Selectors



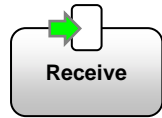
- The WBI Selector provides a means of interposing between the client application and the target destination a dynamic selection mechanism.
- **Client:** The client component makes a call on the Selector Component.
- **Selector:** The Selector Component chooses which target destination to invoke using a declared selection implementation.
 - **Determine dynamically which implementation** of a target destination to invoke based on some defined set of criteria, data and logic
 - **Decouple** the client application from a specific target destination implementation. Change of target does not require change of client.
 - Allow new SCA implementations of a target destination to be added to the Selector dynamically **without requiring a restart** of the application or server
 - Date-based selection available through tooling
 - Enter date range and destination (reference) name
- **Implementation:** The destinations for each operation on the Selector Component are associated with the Selector Component.



WS-BPEL in WebSphere Process Server 6.0



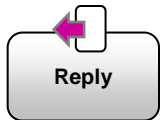
BPEL Basic Activities



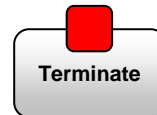
Do a blocking wait for a matching message to arrive



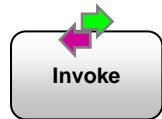
Generate a fault from inside the business process



Send a message in reply to a formerly received message



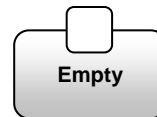
Immediately terminate execution of a business process instance



Invoke a one-way or request-response operation on a port type offered by a partner



Wait for a given time period or until a certain time has passed



A “no-op” instruction for a business process



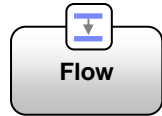
Update the values of variables or partner links with new data



Invoke compensation on an inner scope that has already completed



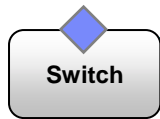
BPEL Structured Activities



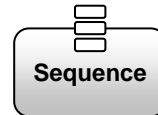
Contained activities are executed in parallel, partially ordered through control links



Block and wait for a suitable message to arrive (or time out)



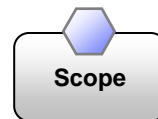
Select exactly one branch of activity from a set of choices



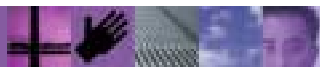
Contained activities are performed sequentially in lexical order



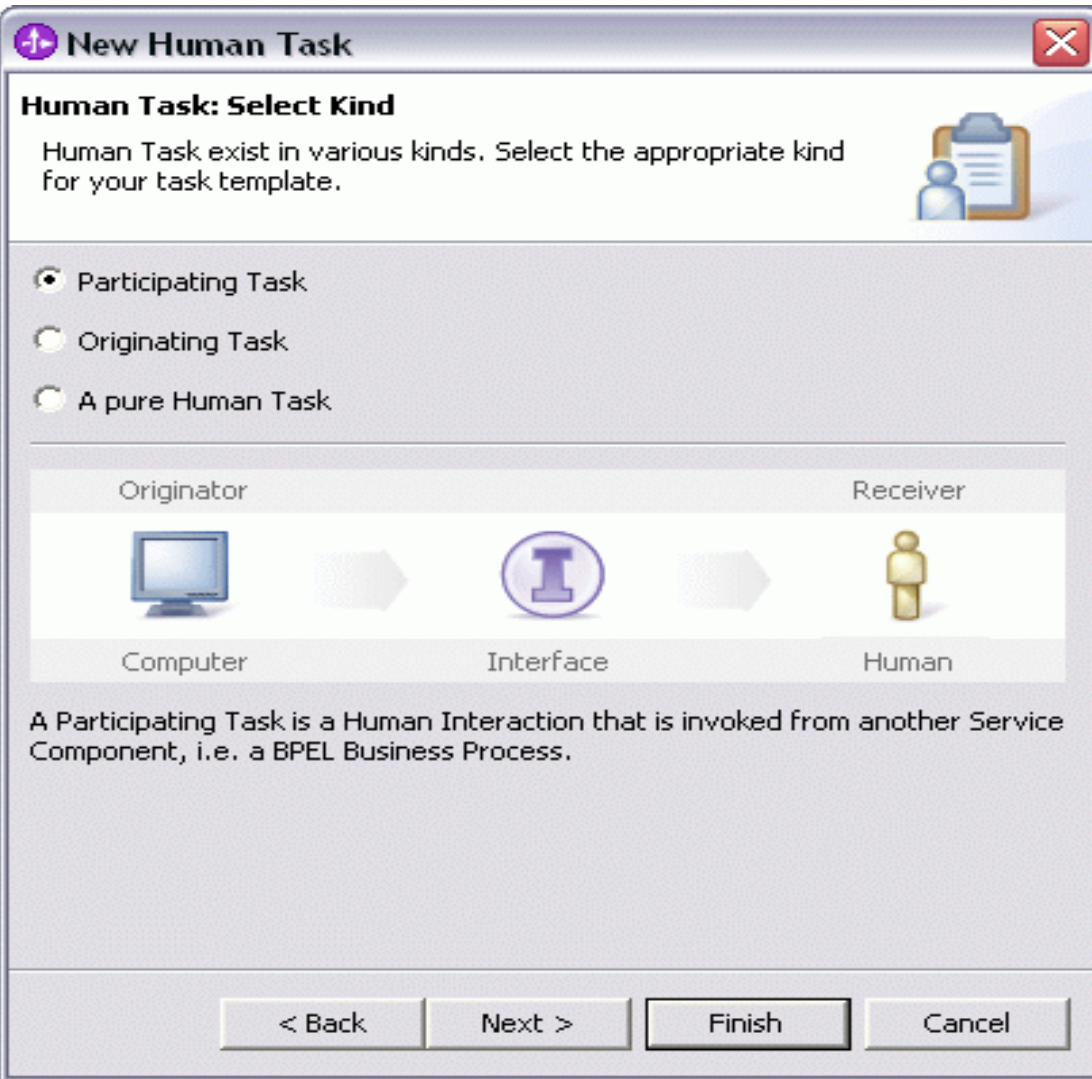
Contained activity is repeated while a predicate holds









Associate contained activity with its own local variables, fault handlers, compensation handler, and event handlers

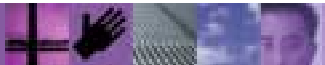
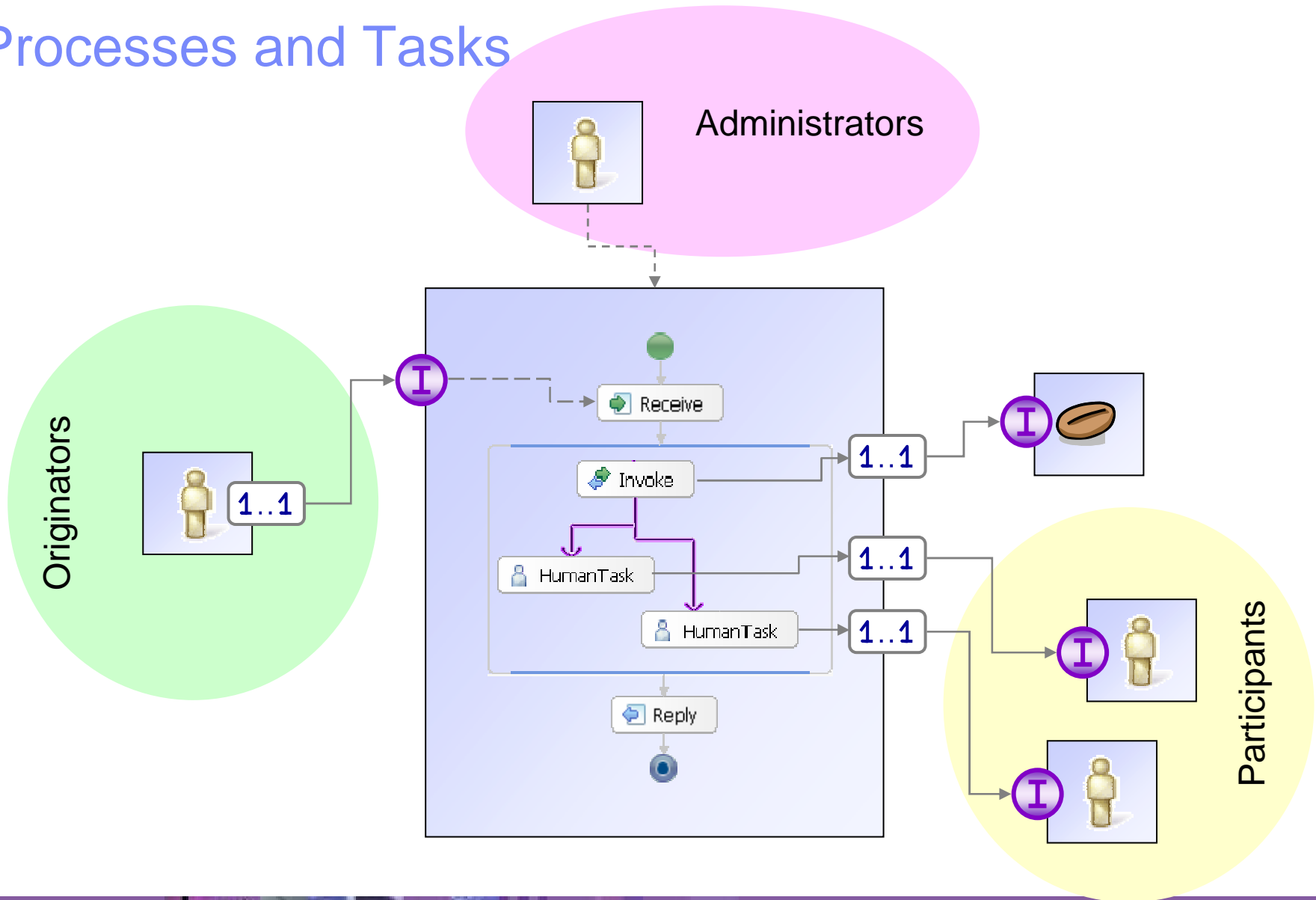


Human Task Manager – Human Tasks



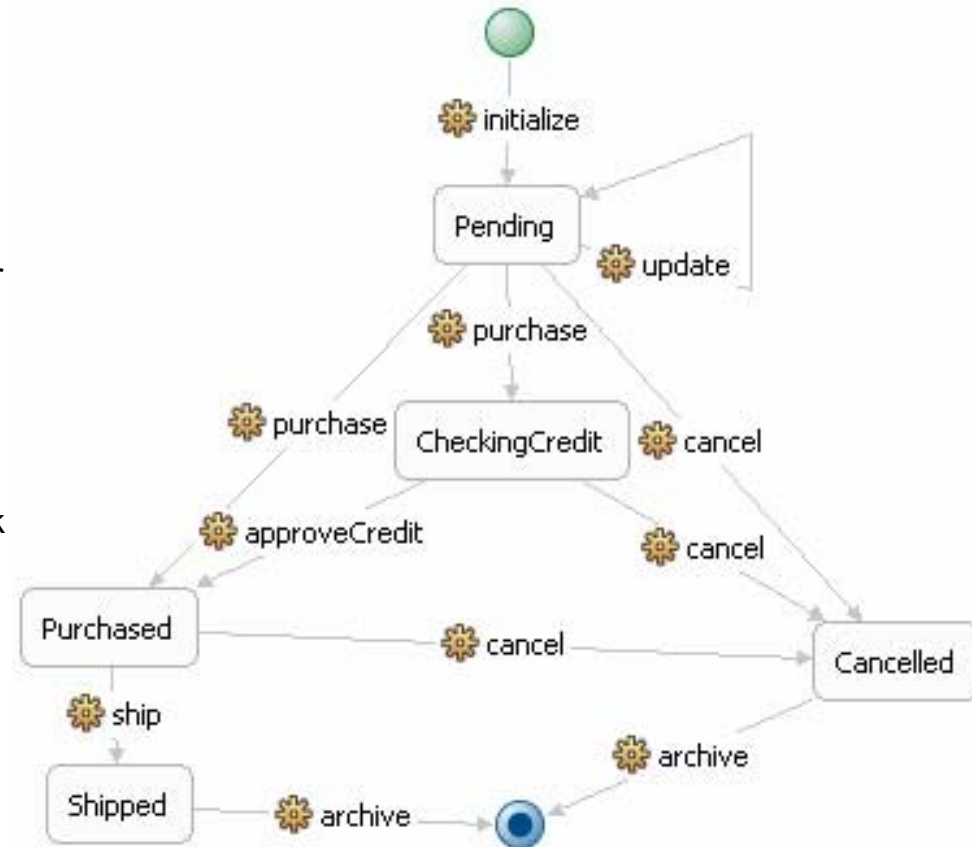
- A Standalone Component
 - Not restricted to just invocation from WS-BPEL Processes
- Three kinds of Human Tasks
 - Machine to Human  → 
Component creates a work item for Human interaction (WS-BPEL)
 - Human to Machine  → 
Human interaction invokes a Component (i.e. Business State Machine)
 - Human to Human  → 
Human interaction invokes a Component which creates a work item for another Human
- Human Task Components
 - Implement WSDL interfaces
 - Are implemented as SCA Components
 - Fit the overall SOA Model

Processes and Tasks



Business State Machine (formerly Adaptive Entity)

- States and state transitions frame the process
- Logic embedded in the transitions
- Based on UML 2.0 State Machine
- Basic Pattern/Execution:
 - Use input parameter for 'correlation' to get proper instance and current 'state'
 - If the **event** isn't supported by the current state, throw an exception.
 - For each transition that supports the event, check the **guards** (if specified) for a result of 'true'
 - Process the state exit **action** (if specified)
 - Process the **transition's** action (if specified)
 - Change the **state**
 - Process the state entry action (if specified)
 - Check for any automatic transitions out of the new state and repeat, or wait for next event



Partners represent external services (SCA components) that are called by the state machine and can be called from

- Actions, Entries, Exits Guards



Business Rules

- Used to capture algorithms in a component
 - Implementation is not exposed

- Natural representation of rules

- Business rules change over time
 - Integrated date/time configuration to support the agile business
 - Dynamically updating rules at runtime

If-then rule (Ruleset)

Rules	
Name	Action 1
Presentation	
Action	temporaryVar = NULL
Name	Rule 1
Presentation	
IF	inClipBG.Clip.color == red
Then	outClipBG.Clip.color = red#5

Decision Tables

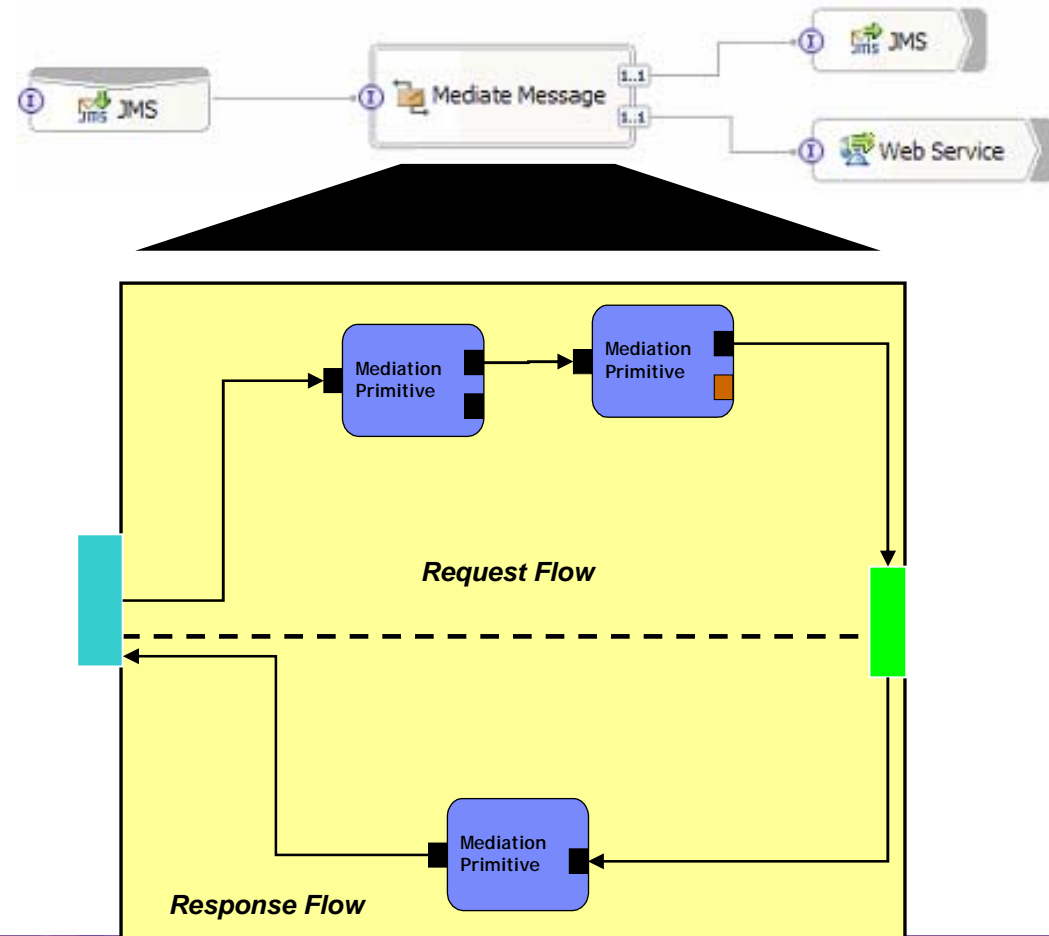
Define the decision table

Conditions				
deliveryMethod	== "AIR"		== "SURFACE"	
testCondTerm	== "CV1"	== "CV2"	== "CV1"	== "CV2"
weightInGrams	postalRate	postalRate	postalRate	postalRate
< 250	= 5.05	= 5.05	= 3.35	= 3.35
< 500	= 6.70	= 6.70	= 5.10	= 5.10
<1000	= 11.40	= 11.40	= 8.50	= 8.50



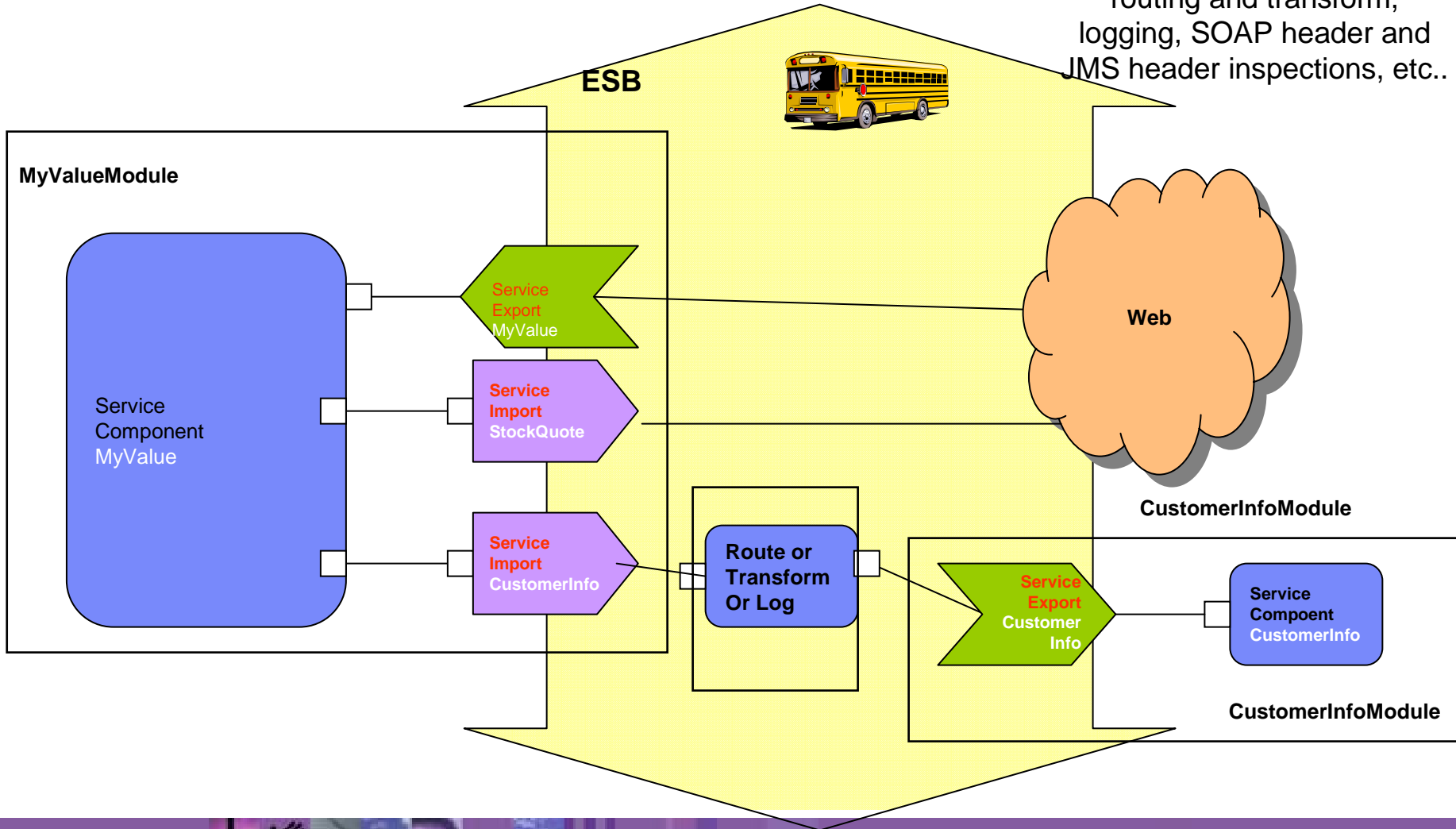
ESB Mediation Component

- Provide the Implementation of mediation “logic”
 - “flows” that operate on messages/events as they are processed by the system
 - Operate on both One-Way and Request-Response interactions
- Pre-Supplied primitives allow flows to be visually composed
 - XSLT Transformation
 - Message Logger
 - Message Filter
 - Fail
 - Stop
 - Database Lookup
 - Custom (Java) Component



WebSphere ESB

Programming Model, Authoring and Admin/Config tools and Primitives to enable message routing and transform, logging, SOAP header and JMS header inspections, etc..



Summary

- WebSphere Process Server and WebSphere ESB provide necessary SOA infrastructure to build next generation business integration solutions
- You can leverage the business integration componentry provided by WebSphere Process Server and WebSphere ESB to build your business integration solutions

