



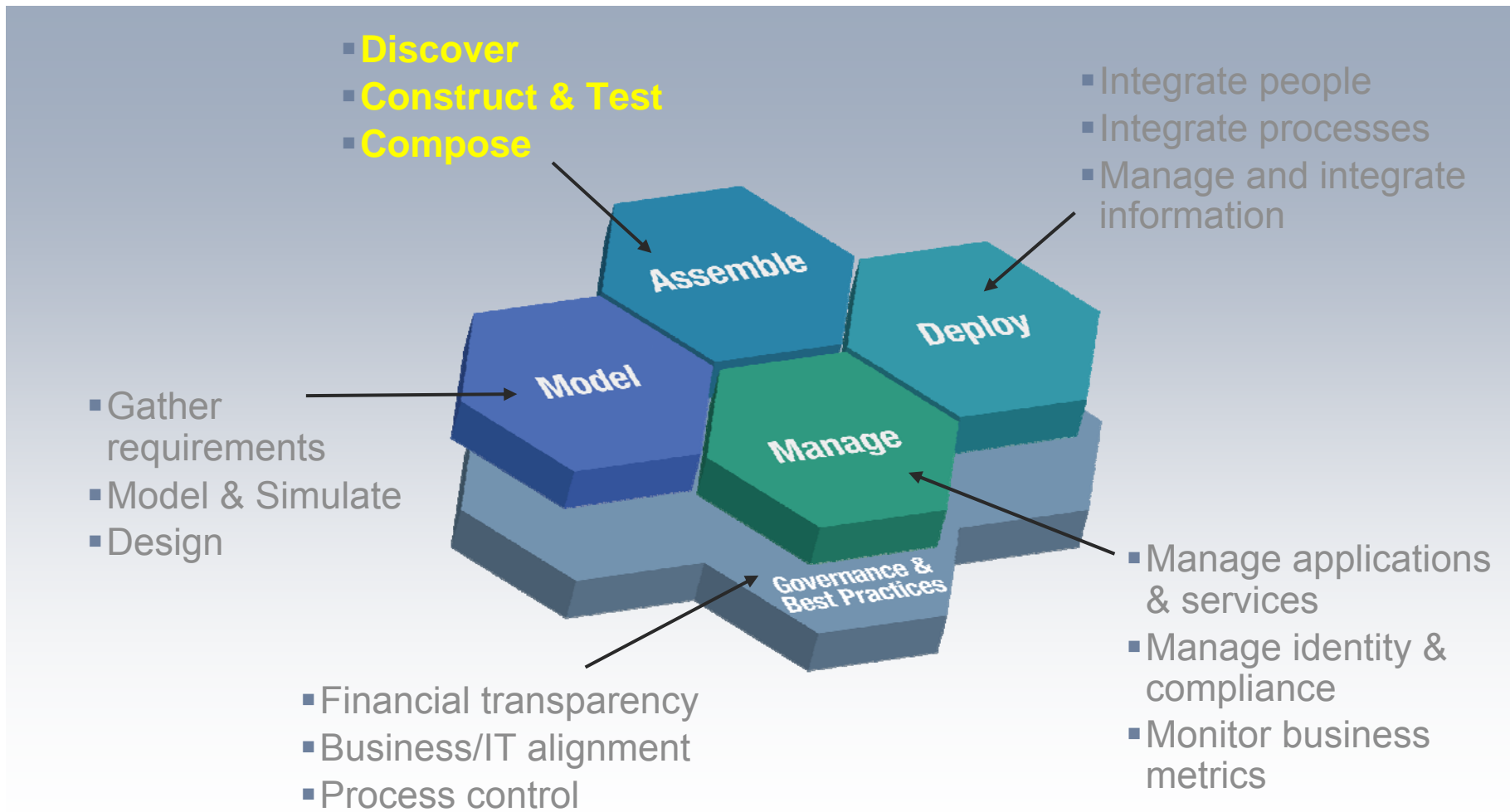
IBM SOA PoT

C1120
Process Assembly and Deployment

C1150
Lab Introduction



Where we are in SOA Lifecycle



Requirements for process assembly and execution

Common Data Model

- All data is represented consistently
- **Standard:** SDO (Service Data Objects)

Common Service Invocation Model

- All components are represented consistently and invoked identically
- **Standard:** SCA (Service Component Architecture)

Common Connectivity

- Enterprise Service Bus
- **Standards:** WS-*/Web Services standards

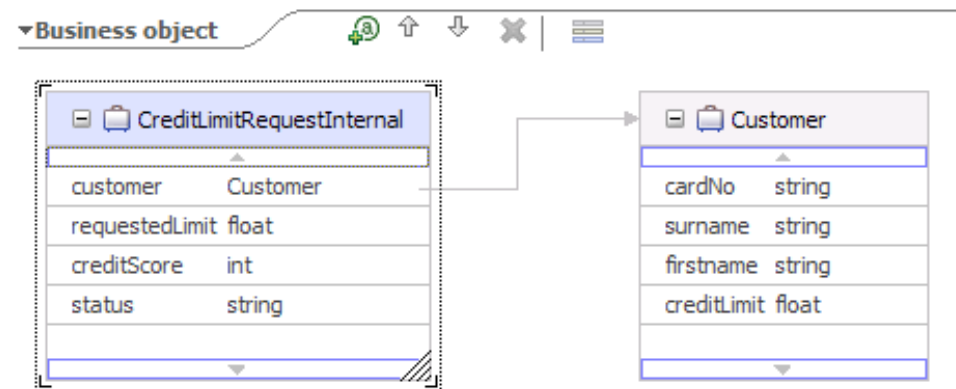
Common Service Choreography

- Components can be choreographed independently of their implementation
- Human tasks as services
- **Standard:** WS-BPEL (Business Process Execution Language)

Common Data Model

Business Objects

Common data
representation in
Process layer



Supports

- Inheritance
- Aggregation

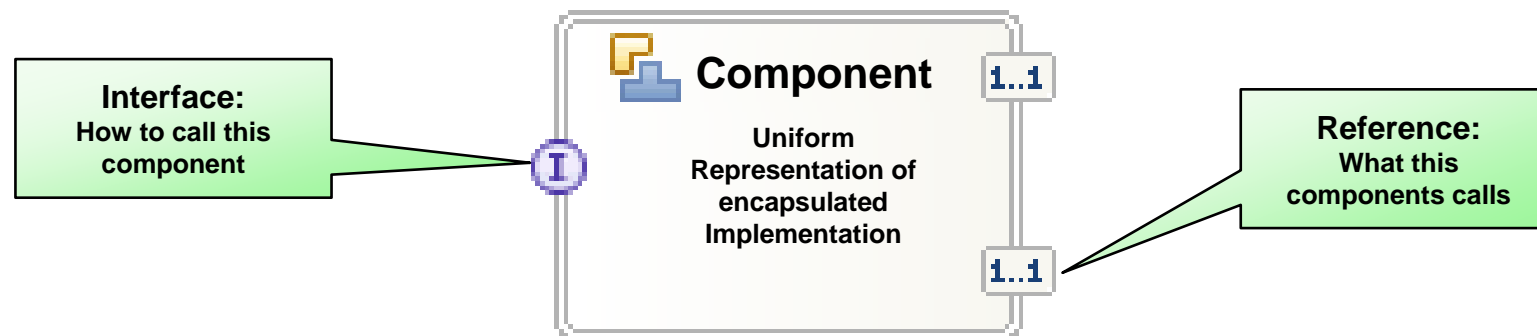
Benefit:

- **Business Objects form the basis for service orientation by decoupling data definitions from actual implementations**
- **Common model for representing data within WebSphere Process Server**
 - Consistent logical representation, independent of data source or wire format
 - Based upon SDO standard
- **Reduces effort, reduces project times, simplifies integration work**

Common Invocation Model

Service Component Architecture

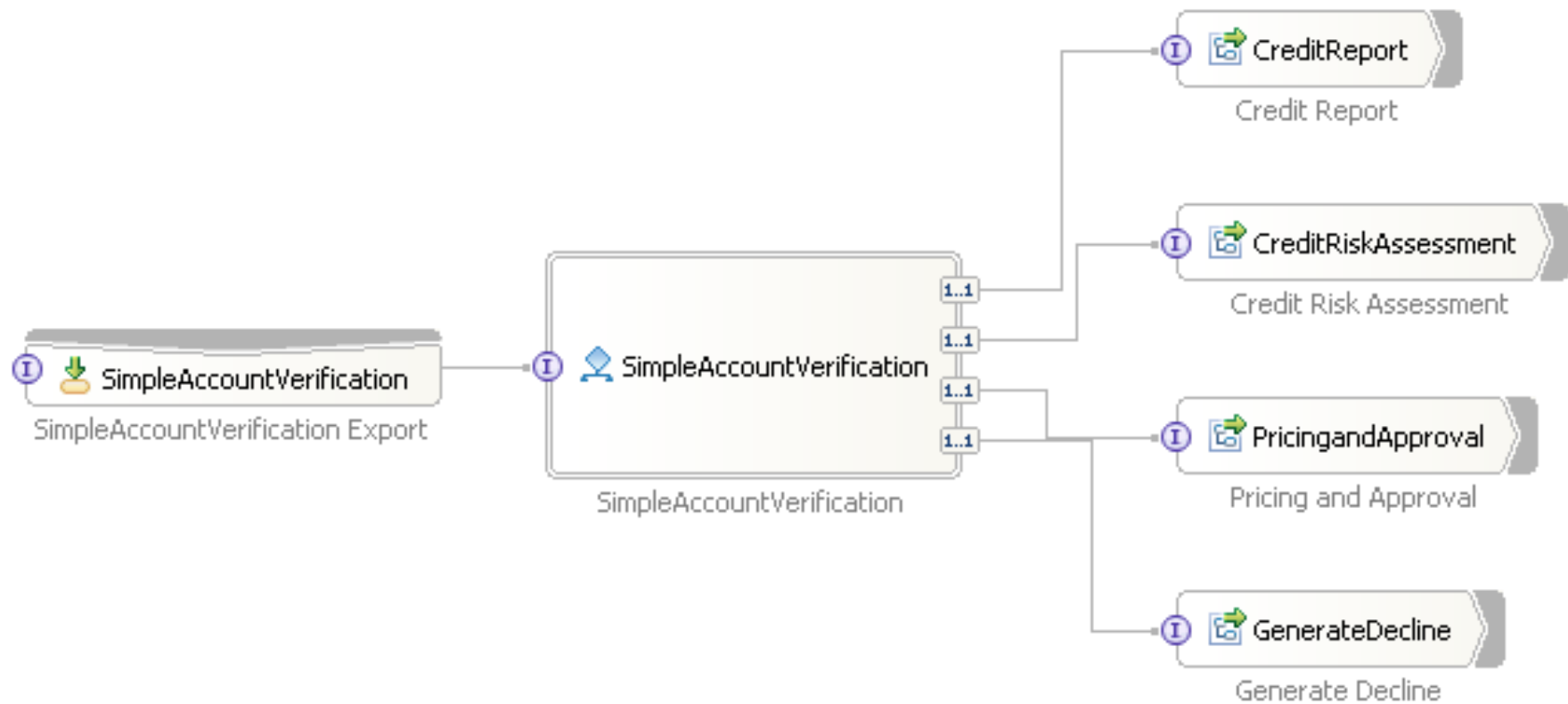
Service Component Architecture (SCA)
supported by IBM, BEA, Oracle, SAP,
IONA, Siebel and Sybase



Benefit:

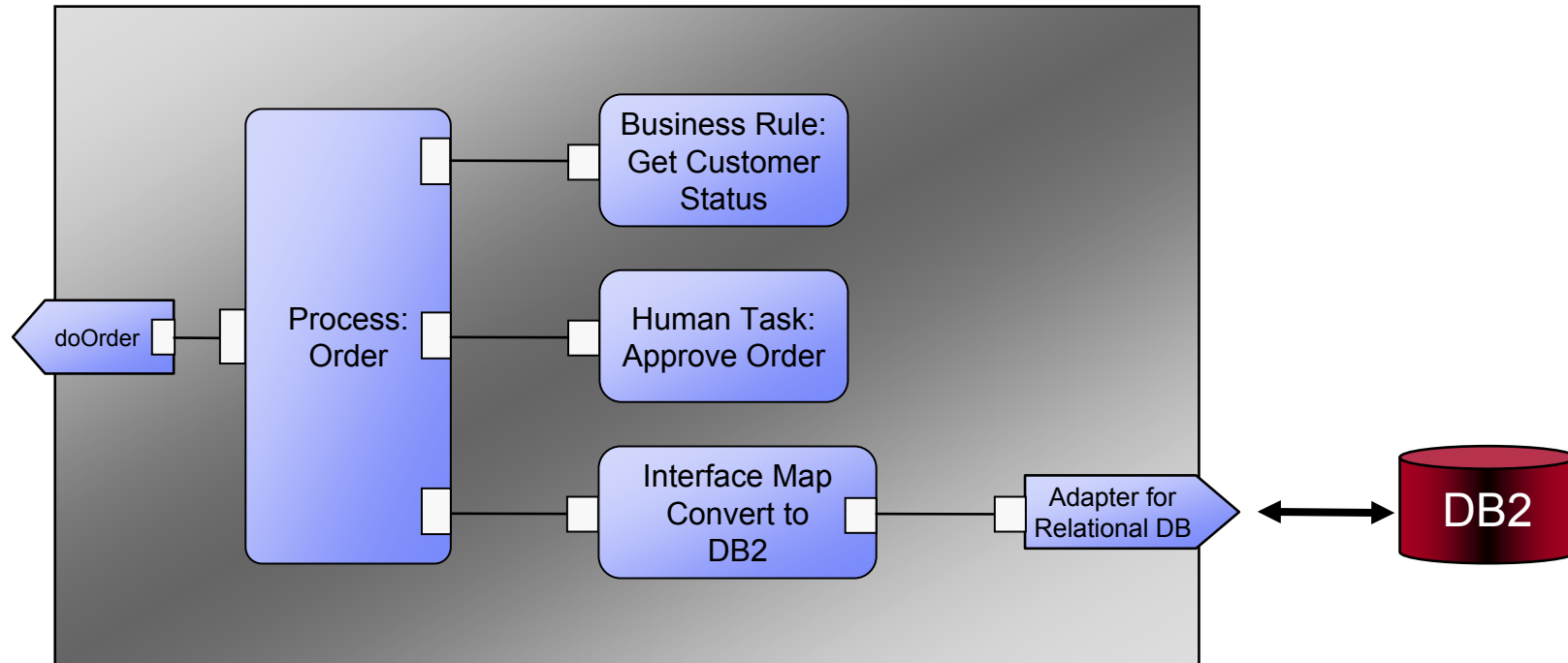
- Encapsulate components for reuse
 - ▶ Service Components are wired together to form deployable solutions
 - ▶ Business Objects are the data flowing between Service Components
- All components (e.g., services, rules, human interactions) are represented consistently and invoked identically - encapsulation and reuse will reduce development cost
- Increased productivity, reduced cost

Lab Example of Assembly Diagram



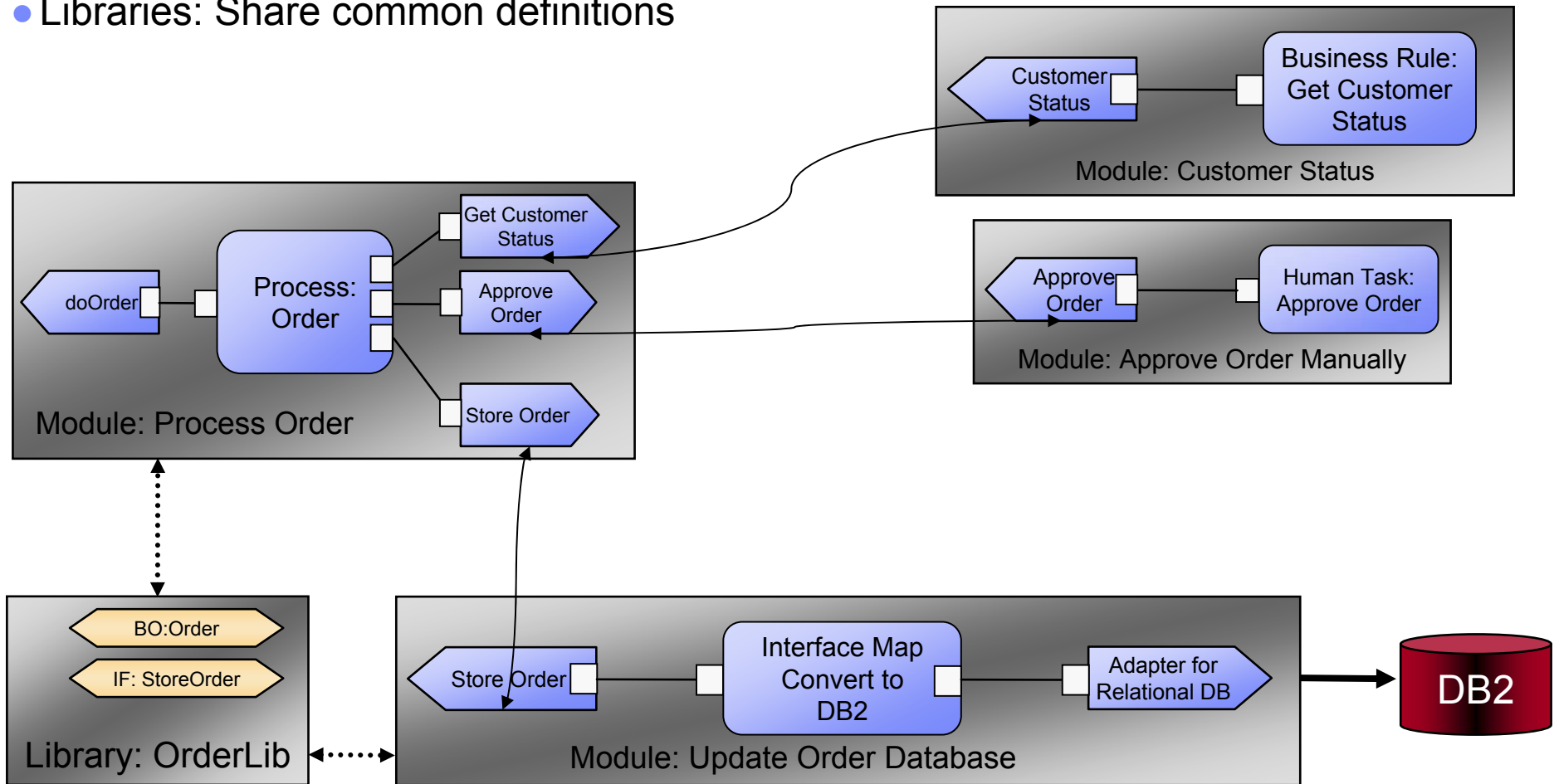
Component Assembly

Using Modules for Building Applications



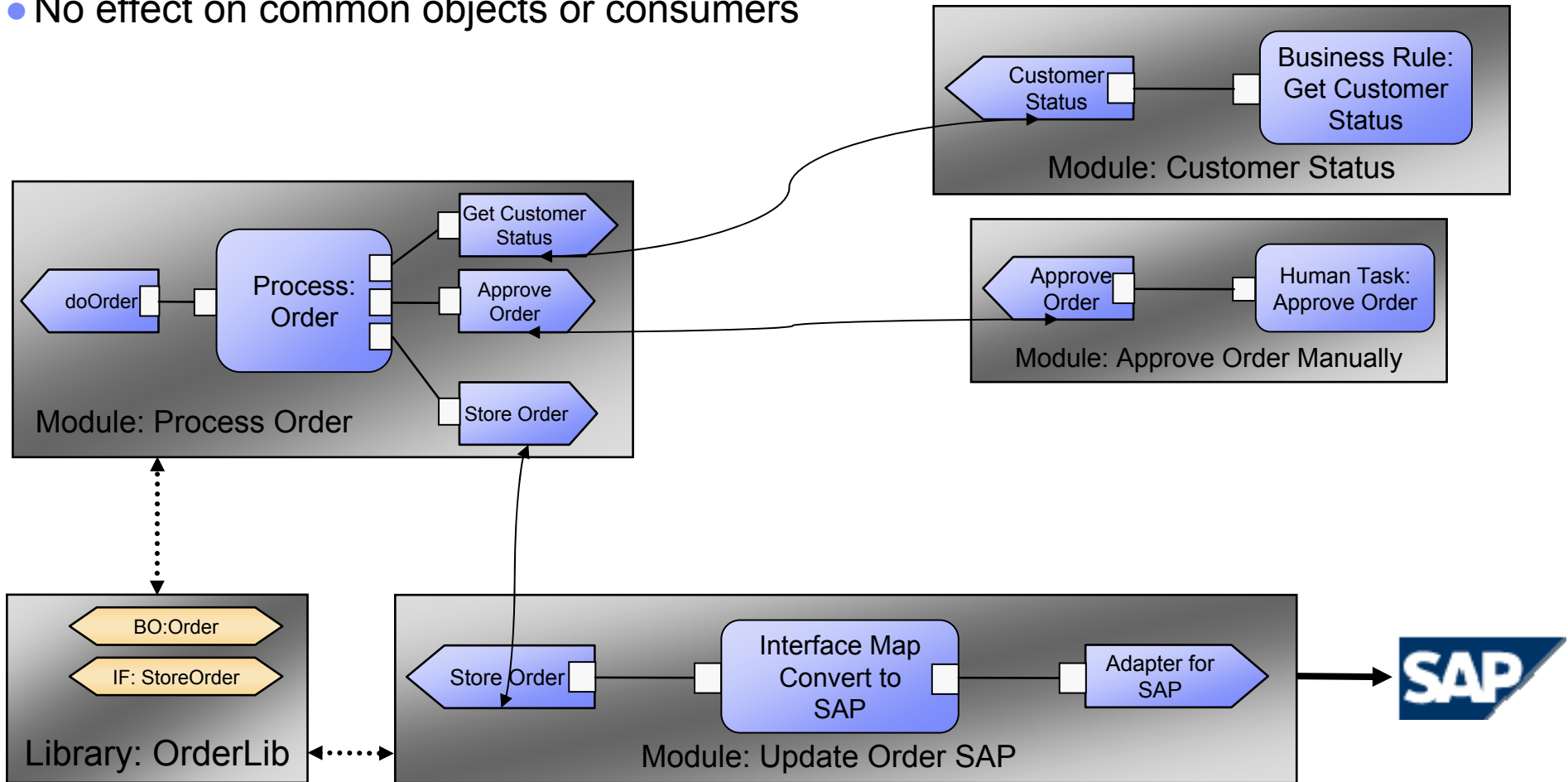
SCA: Using Modules for Encapsulation and Reuse

- Modules: Encapsulate and Reuse Functionality
- Libraries: Share common definitions



Using Modules for Encapsulation and Reuse...

- Store Order in SAP instead of DB2
- No effect on common objects or consumers



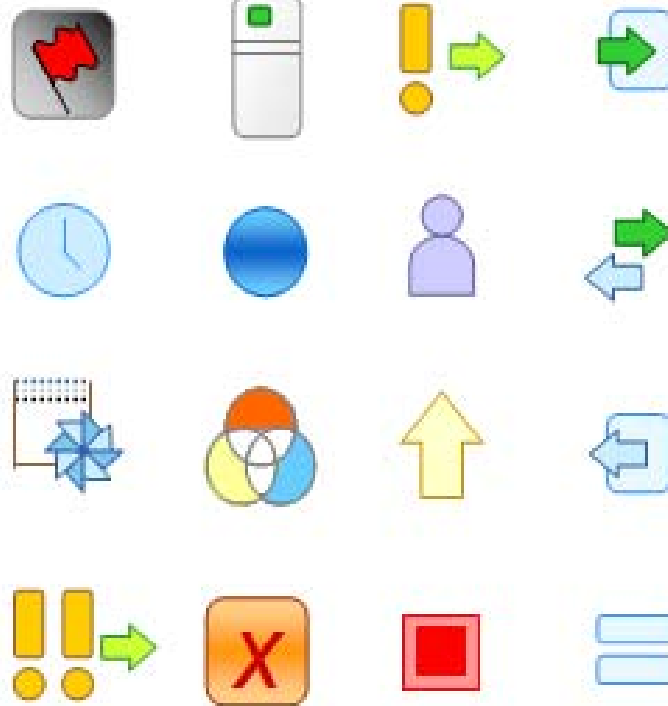
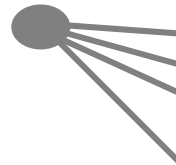
The Construct of a Business Process

Common Choreography enabled by BPEL

Capabilities for creating powerful business processes that address automation and dynamic change

Constructing a dynamic process using WS-BPEL

Choreographed Services



- ✓ Long-running
- ✓ Straight-through
- ✓ Compensation
- ✓ Transaction handling
 - ✓ Two-phase commit
- ✓ Event handling
- ✓ Fault handling
- ✓ Parallel paths
- ✓ Process Bundles

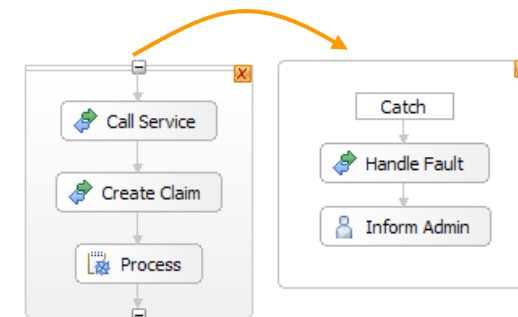
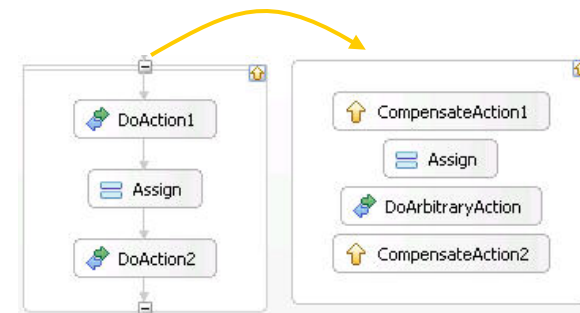
... while service implementation details stay hidden:

- *Human tasks*
- *Business rules*
- *ESB Mediations*
- *External services*
- *...*

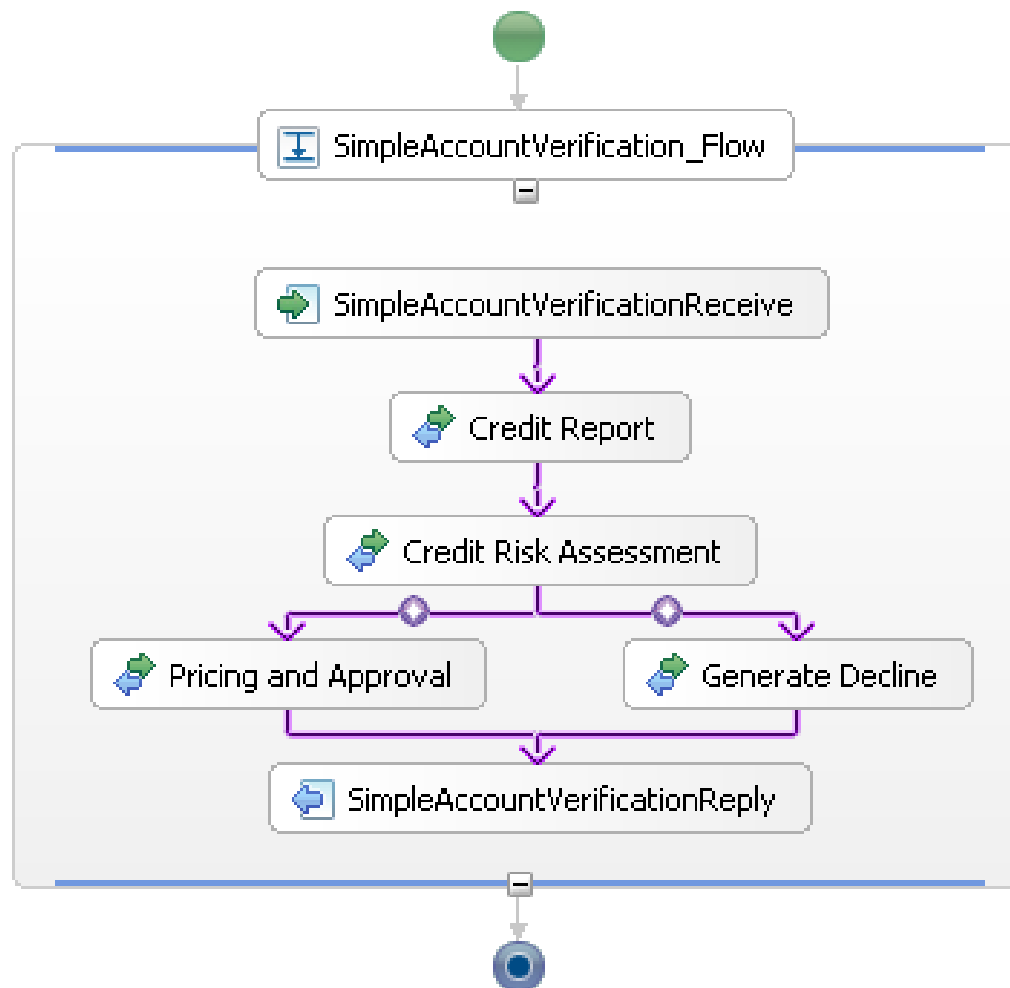
Process Integrity in BPM

Using native BPEL features

- Advanced Event handling
 - Process waits for event to arrive
 - arbitrary “event handler flows” can be assigned based on the event type received
 - Synchronize processes with external partners
- Process Compensation
 - Logical “undo” of already committed process steps allows to “roll-back” multi-transactional processes consistently
 - Keeps your process environment consistent
- Effective error handling
 - Assign different “fault handler flows” based on type of error received
 - Either stop/cancel process or handle fault as part of the process – based on business requirements
 - Flexible way to handle any type of error in your business processes



Lab Example of Process Diagram



Base Tools and Runtimes

Development Tools



WebSphere Integration
Developer
v6.1

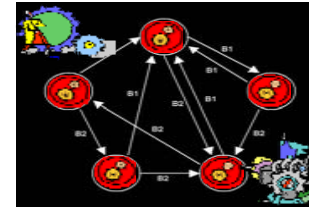
RAD v7.0.0.5

Eclipse v3.2

Deploy



Runtime



WebSphere Process
Server
V6.1

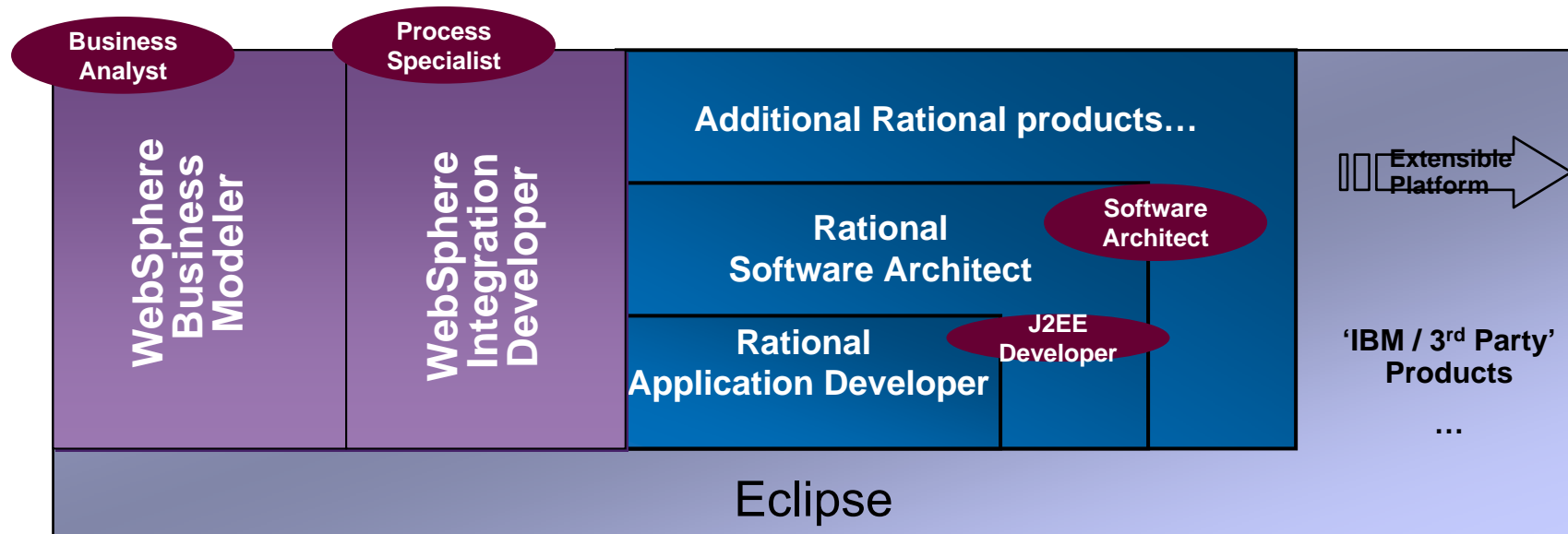
WebSphere ND v6.1.0.11

- **WebSphere Integration Developer is based on a subset of Rational Application Developer (RAD) v7.0.0.5**

- **WebSphere Process Server is based on WebSphere Application Server ND v6.1.0.11**

WebSphere Tooling Portfolio

WebSphere Integration Developer for WebSphere Process Server

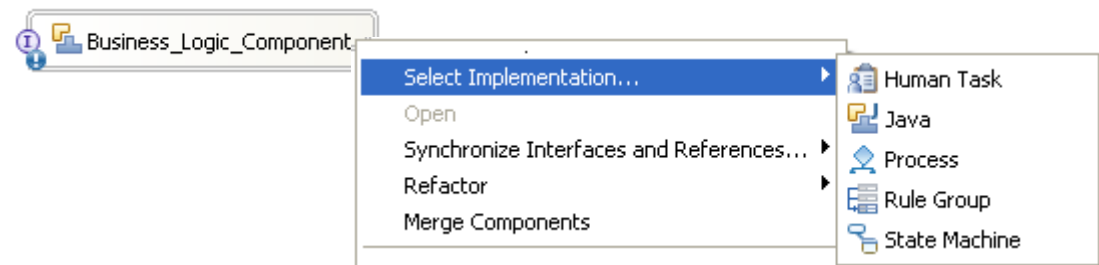


- All IBM Software Development platform products install in a consistent and extendable way within the 'platform'
 - First Product installs the 'platform' as well as its own product-specific 'installable units'
- Role-based tool approach, experienced as single integrated "desktop" IDE
- Reuse of Rational Application Developer and Eclipse components ('installable units') is baked into the design

Top-Down Development

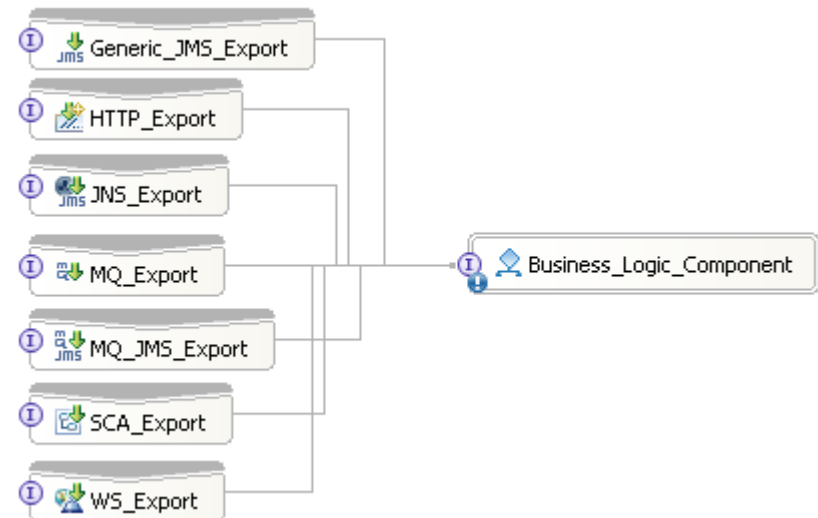
- **A Choice of several implementation types:**

- **Process Component**
- **State Machine**
- **Business Rule**
- **Hum Task**
- **Java**



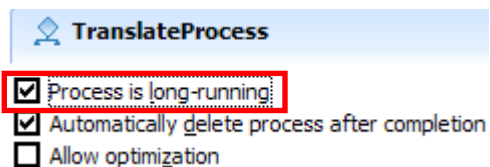
- **A choice of several invocation bindings**

- **Web Services**
- **Messaging**
 - **MQ, JMS, MQ-JMS, Generic-JMS**
- **HTTP**
- **SCA**
 - **Used for component to component interactions in WebSphere Integration Developer**

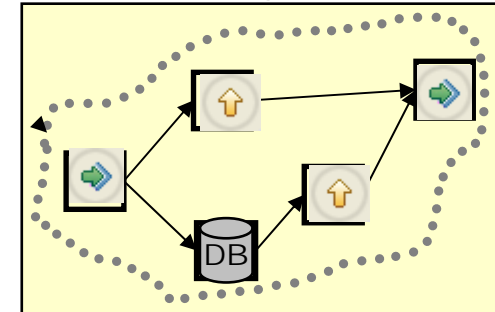


Process Component

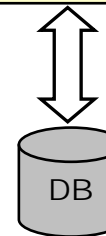
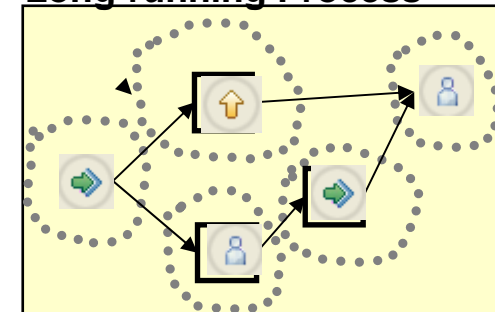
- **Process is a directed graph of BPEL Activity Nodes that represents a single business activity**
- **There are two types of Processes**
 - **Short-running**
 - Single transaction per Process
 - Basic Process Choreography
 - **Long-running**
 - Transaction scope at Activity level
- **Persistent**
 - Process is long-running
 - Automatically delete process after completion
 - Allow optimization
- **Asynchronous Activities allowed**
- **Compensation support**



Short-running Process



Long-running Process



Maintains execution
state in database

Component Assembly

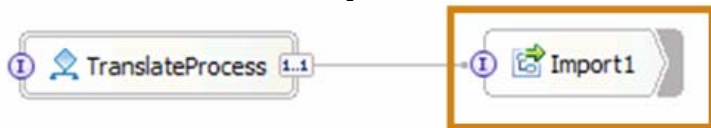
- **Add Exports**

- ▶ For inbound J2C or Messaging
- ▶ To expose Components outside of a Module



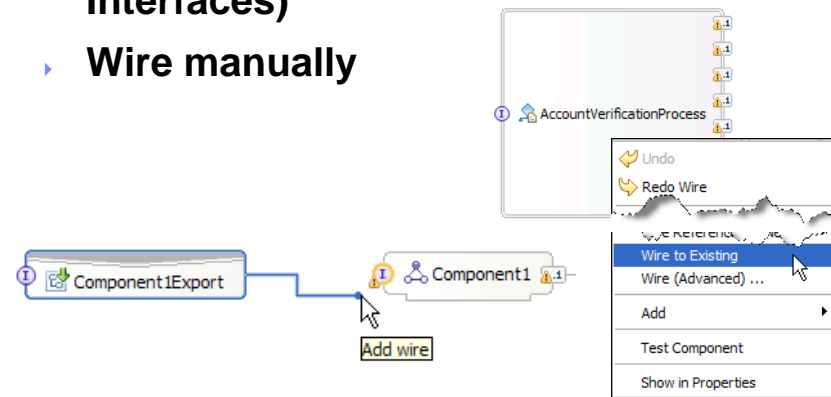
- **Add Imports**

- ▶ Web Services
- ▶ J2C Adapters
- ▶ To access Components in external

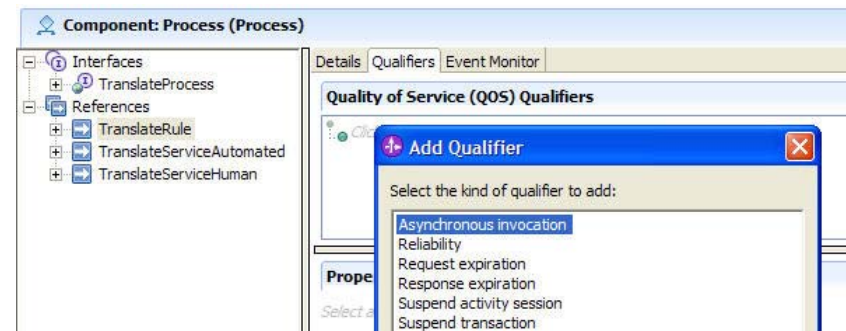


- **Wire Components**

- ▶ Use Wire to Existing (automatically connects matching References to Interfaces)
- ▶ Wire manually



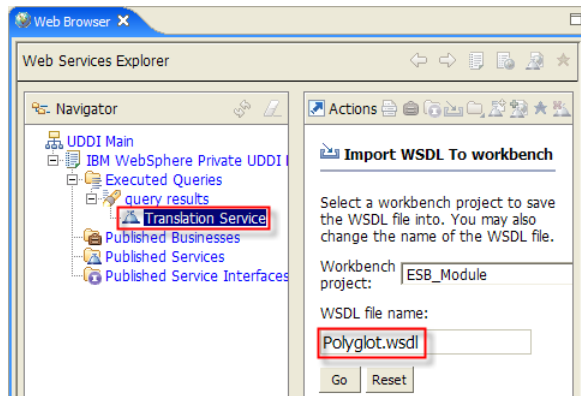
- **Configure QoS Qualifiers**



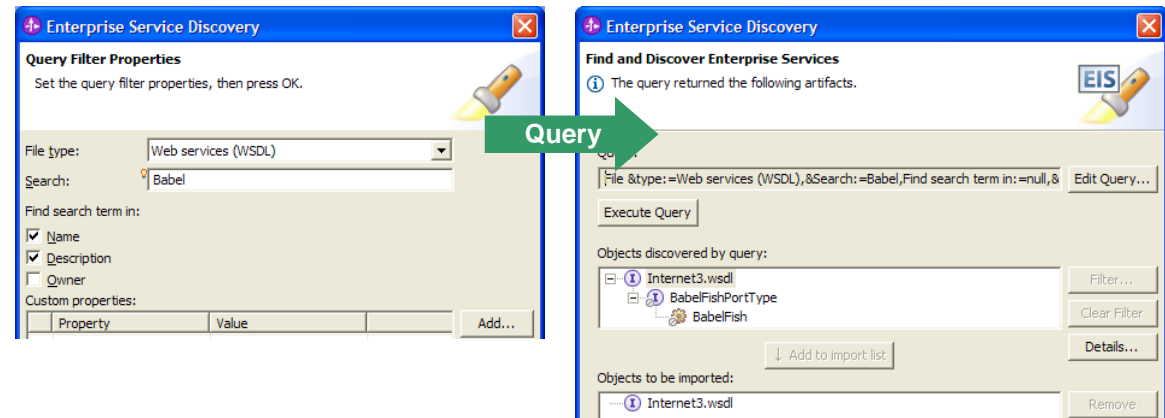


- Import a Web Service to a Module from:

UDDI



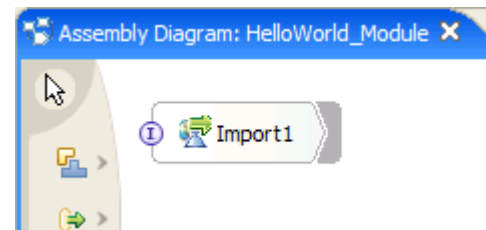
WSRR



- Creates Web Services port the in Navigator

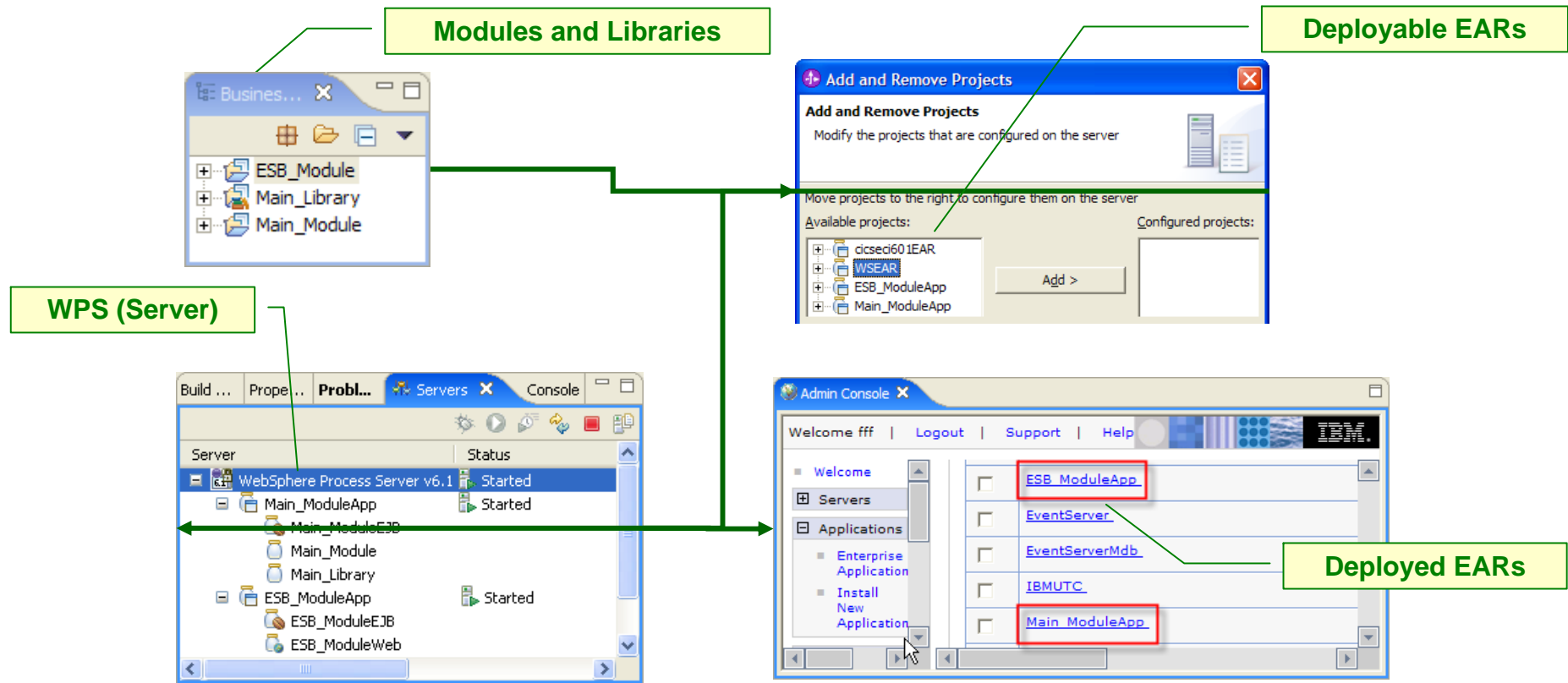


- Add Web Services port to Assembly Editor



Deployment

- **Modules and the associated files are called “Projects”**
 - In fact “Projects” are implemented as EARs
- **Add all “Projects” associated with the Integration Solution to an instance of the WebSphere Process Server v6.0 server**
 - This action will also start the server and publish all EARs



WebSphere Process Server 6.1

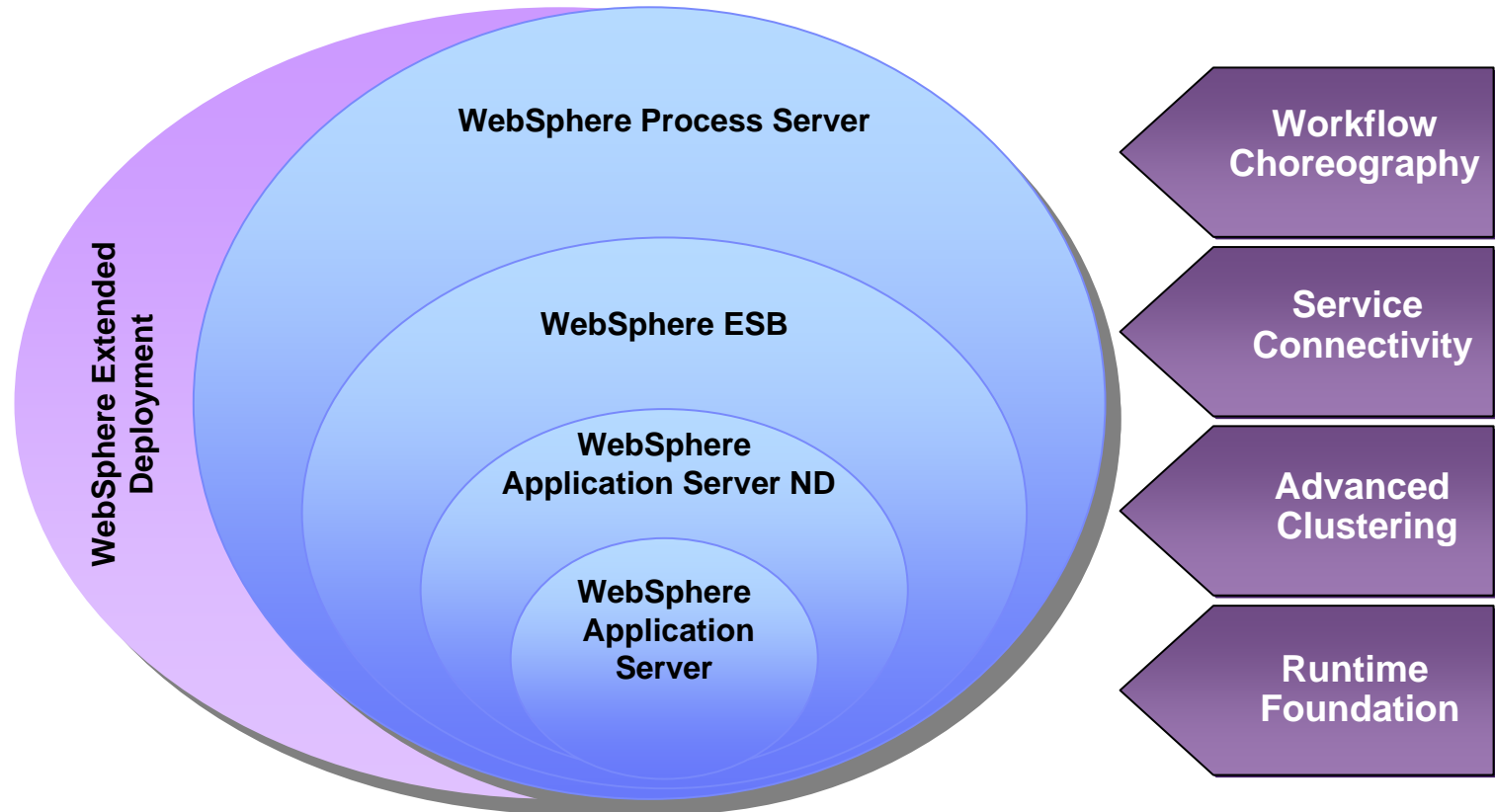
Based on WebSphere Application Server 6.1 and ESB

OS

- Windows
- AIX
- HP-UX
- Solaris
- Linux
- I5/OS
- z/OS
- 32/64 bit exploitation

Databases

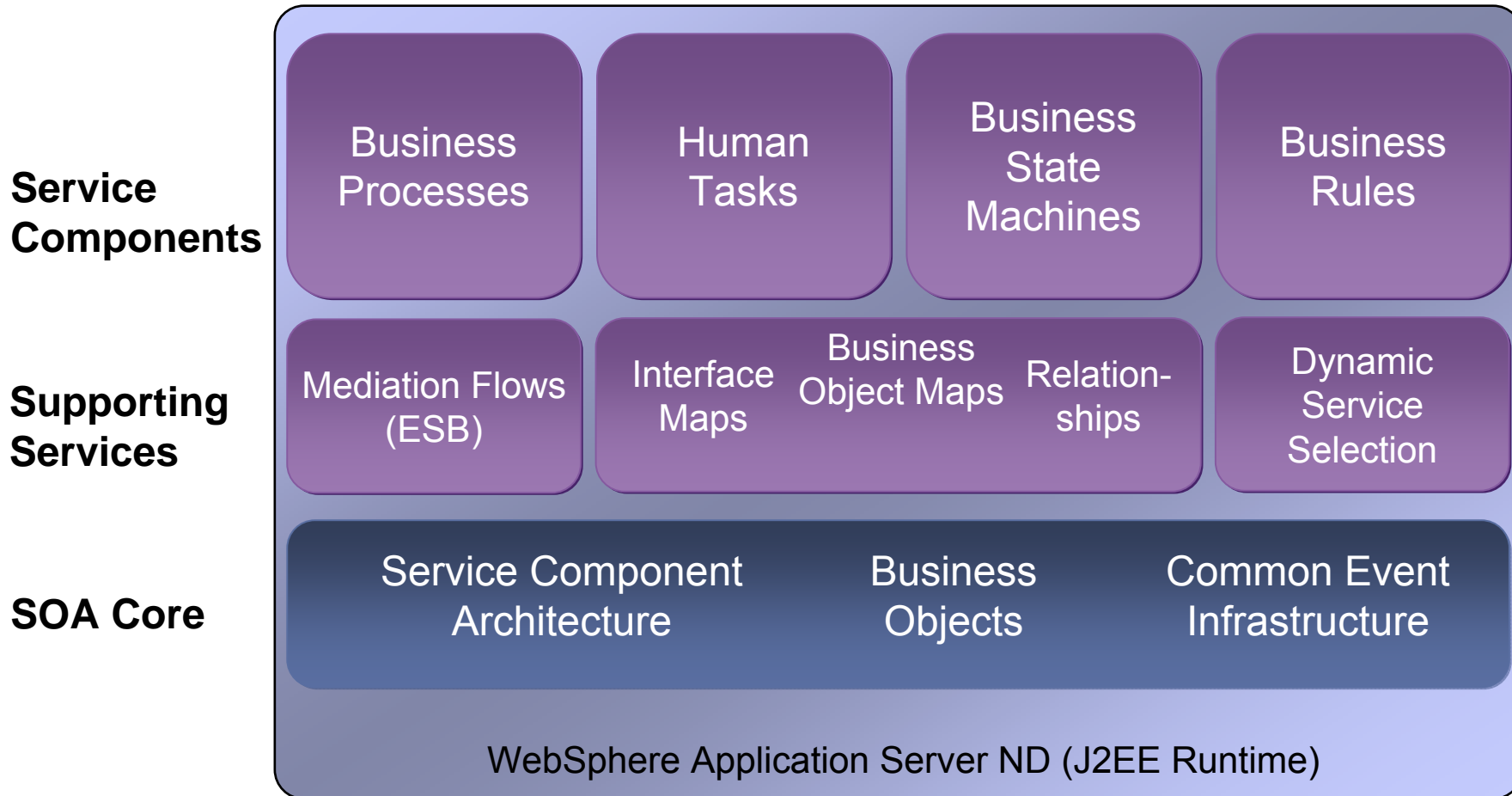
- DB2
- Cloudscape
- Oracle
- MS SQL Server
- Informix
- IBM Information Integrator



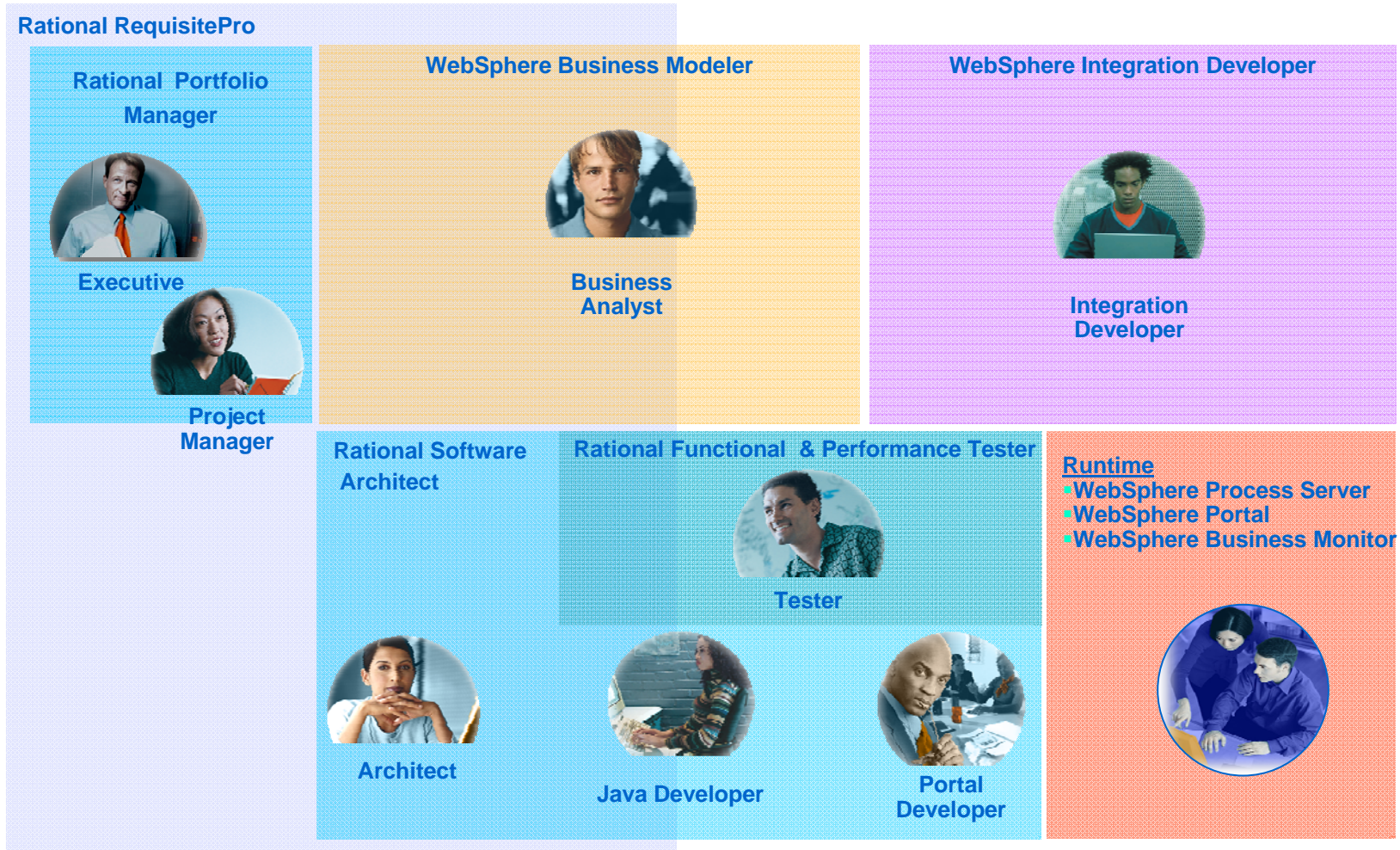
Common administration interface for all products

WebSphere Process Server 6.1

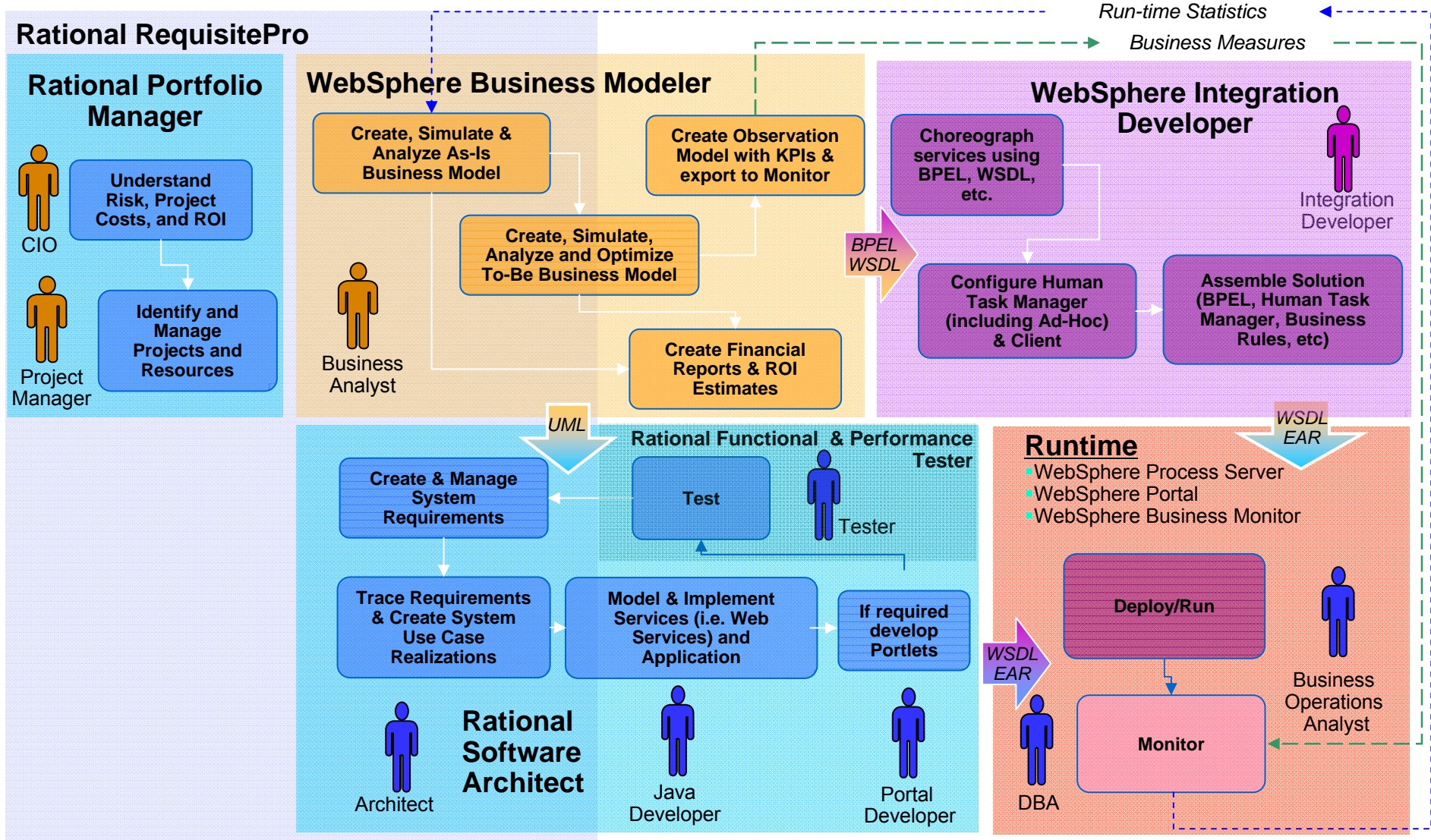
Components



Areas for Business Driven Development (BDD) – Roles




BDD for SOA (new role: Integration Developer)



NEXT: Labs

What are we going to do in these Labs?

4. Assemble the new process model

- Import the Model from Lab 2
- Implement the Tasks of the Business Process
- Do a little coding 
- Deploy and test the completed business process

5. Publish the Service Interface for re-use

- Publish the completed Credit Report Service to the Repository

Lab 4 - Objectives

- **Assemble service components to form business process**
- **Lower business costs and achieve IT flexibility**
 - **Apply the building-block approach**
 - **Integrate using modular service components**
 - **Loosely-couple service components for flexibility**

Role: Integration Developer

The tool we will use is WebSphere Integration Developer

- **Eclipse based tooling**
- **Similar look and feel to all other Eclipse based tooling – you have just used Modeller**
- **Different eclipse perspectives allow us to show/hide unneeded functionality – we will be in the “Business Integration Perspective”**

Lab 4

45 Minutes

Lab 5 - Objectives

- **Encourage reuse at the enterprise level**
- **Promote standards**
 - **Store the service in the repository for re-use**
- **Use WSRR again**

Role: Integration Developer

Lab 5

5 to 10 Minutes

Thank You

