

Business Process
Choreography and Business
Process Execution Language
for Web Services

Smart
SOA



Kevin Lo

Software Engineer, IBM

IMS On-Demand

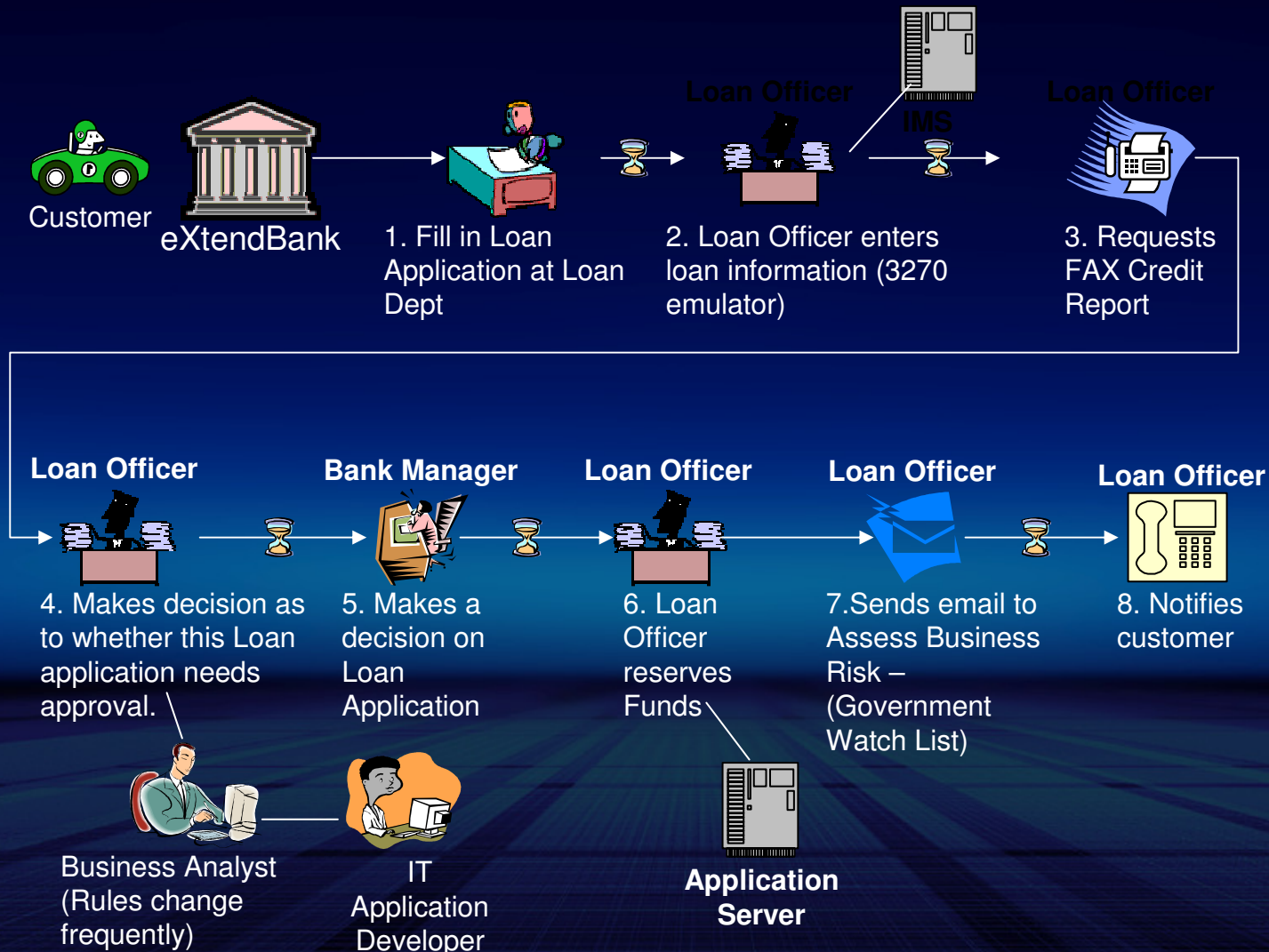
loke@us.ibm.com

What is a business process??

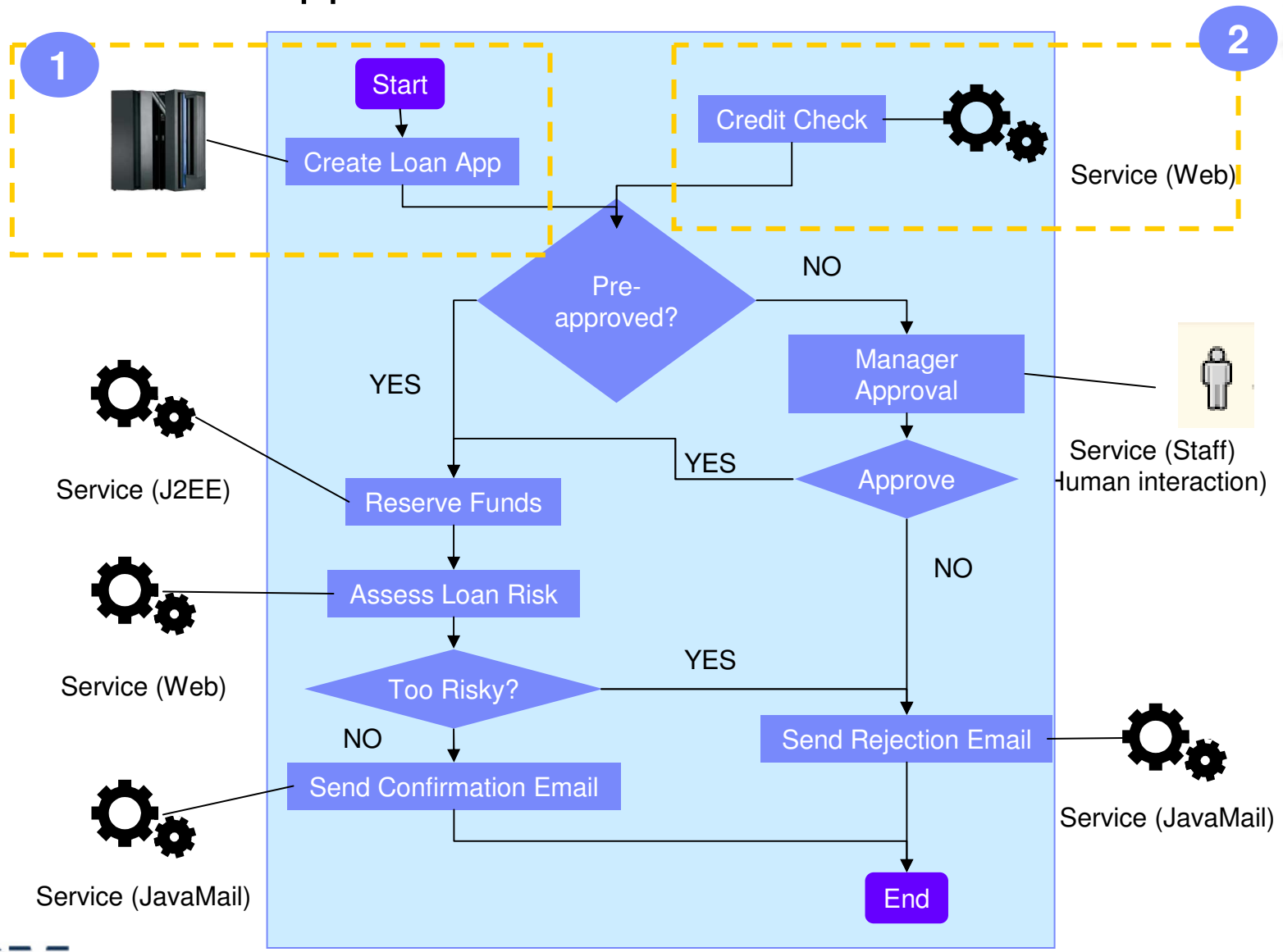
Smart
SOA

A series of individual tasks with each task executed in a specific order

A Typical Business Process - Loan Application



Modern Loan Application Business Process



Business Process Management

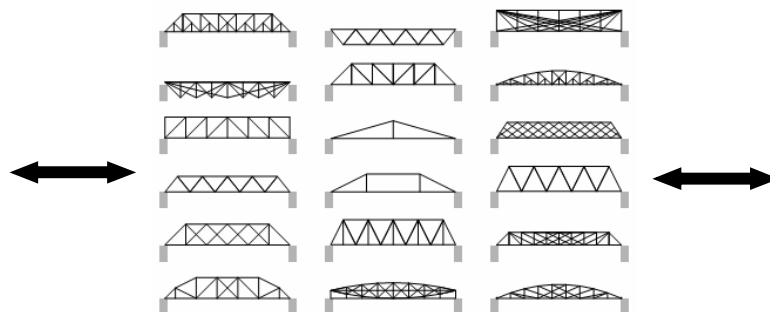
Smart
SOA

Bridging The Gap Between Executive and IT

Bridging the Gap



Executive focus



Business Processes!!!



IT Personnel Perspective

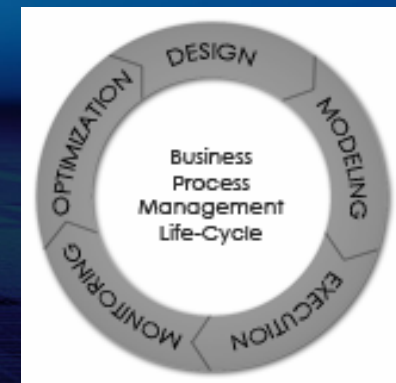
- Process Modeling Tool 1
- Process Modeling Tool 2
- Process Modeling Tool 3
- Process Modeling Tool 4



- Process Engine 1
- Process Engine 2
- Process Engine 3
- Process Engine 4

Business Process Management (BPM)

- The discipline of designing, deploying, and monitoring business processes using various standard technologies.
 - Discovering and designing processes
 - Iterative process
 - A tool needed to codify business processes so that they can be viewed, updated, and executed.
 - Modeling processes
 - Modeling takes the theoretical design and introduces combinations of variables
 - What-if analysis
 - Executing processes
 - Business processes written in script are fed into business process management application and executed.
 - Monitoring/Optimization
 - Much easier due to the wealth of information about each of the processes that the application executed.



BPM & BPEL

- Business process management suites (BPMS) license, services, and maintenance revenue from software vendors will grow from \$1.2 billion in 2005 to more than \$2.7 billion by 2009 (Forrester research)
- BPM is a systematic and iterative approach to improving an organization's business processes.
- BPM activities seek to make business processes more effective, more efficient, and more capable of adapting to an ever-changing environment.

BPEL is a new BPM **standard/specification**, providing organizations with ways to avoid vendor lock-in, therefore lower the total cost of ownership and provide future flexibility as technologies and products mature.

Business Process Management's Role in SOA

- “BPM is growing in popularity and is complementary to SOA due to its ability to help make business processes more efficient and effective while enabling an organization to more easily adapt to changing business requirements.”
- “The powerful combination of BPM to streamline business processes within an SOA strategy will help position companies to become industry leaders while ensuring they are poised for continued success.”

Vice President, SOA & WebSphere Strategy, Sandy Carter:

"BPEL lays the foundation and a clear path for increasing portability of processes, protecting customer investments, reducing risk, and providing stability and a clear direction for the future of process execution semantics"

Pacorini Experience

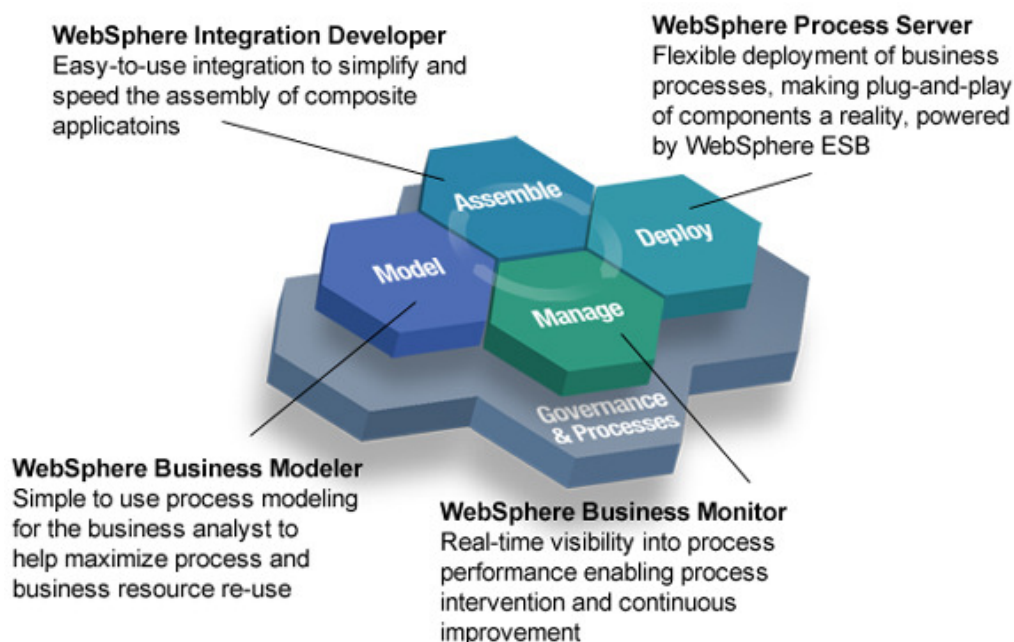
Pacorini used Business Process Management (BPM) to identify high-priority business processes and automated them by integrating its legacy system via a service oriented architecture (SOA)

- *"Service oriented architecture and business process management go hand in hand,"*—Cristian Paravano, CIO, Pacorini Group
- *"In each country, in each location and with each customer, we use many different information service components to build consistent business processes. This provides us with the flexibility we need to respond to customer and market demands as well as lower operating costs."*—Cristian Paravano, CIO, Pacorini Group



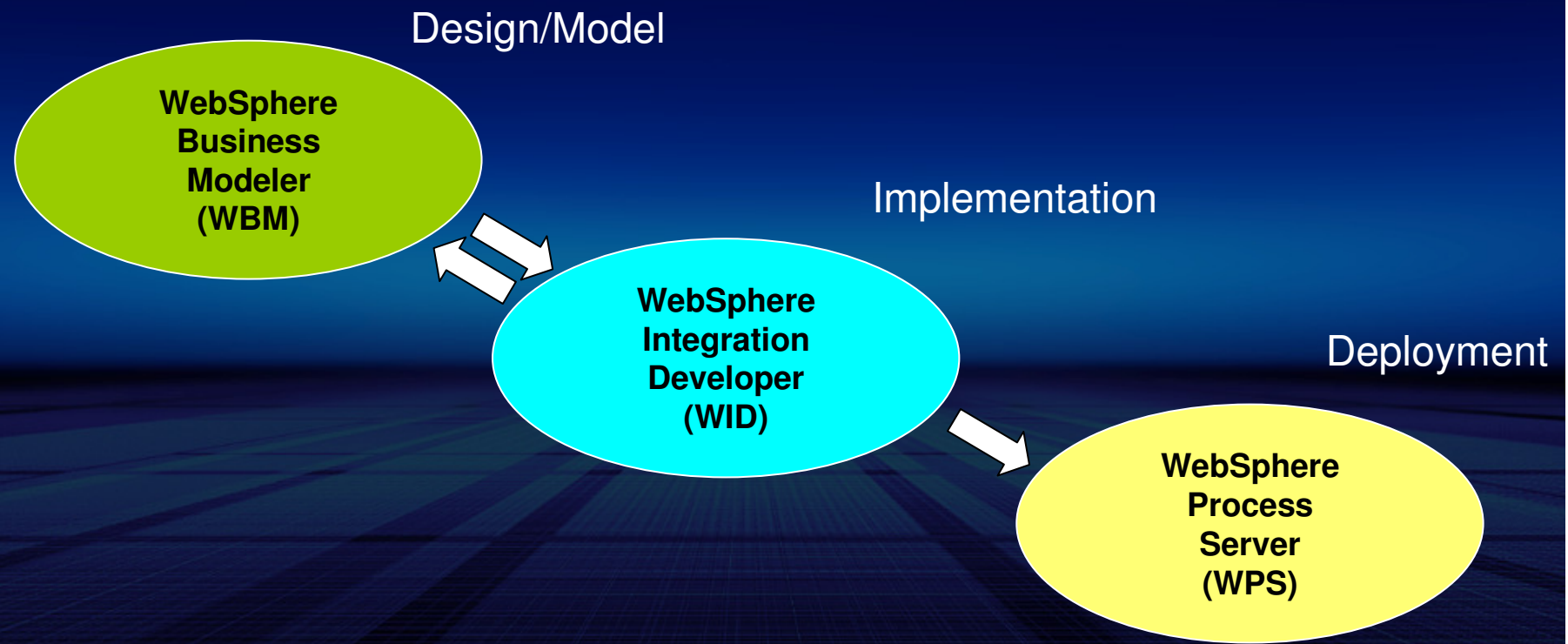
SOA Development and Deployment Cycle by IBM

- Business analysts model the business process in [WebSphere Business Modeler](#)
- Integration developers implement the business process in [WebSphere Integration Developer](#)
- Developers or administrators deploy the business process to [WebSphere Process Server](#).
- Business analysts receive feedback on key performance indicators and business events defined in the model, provided by the [WebSphere Business Monitor](#).



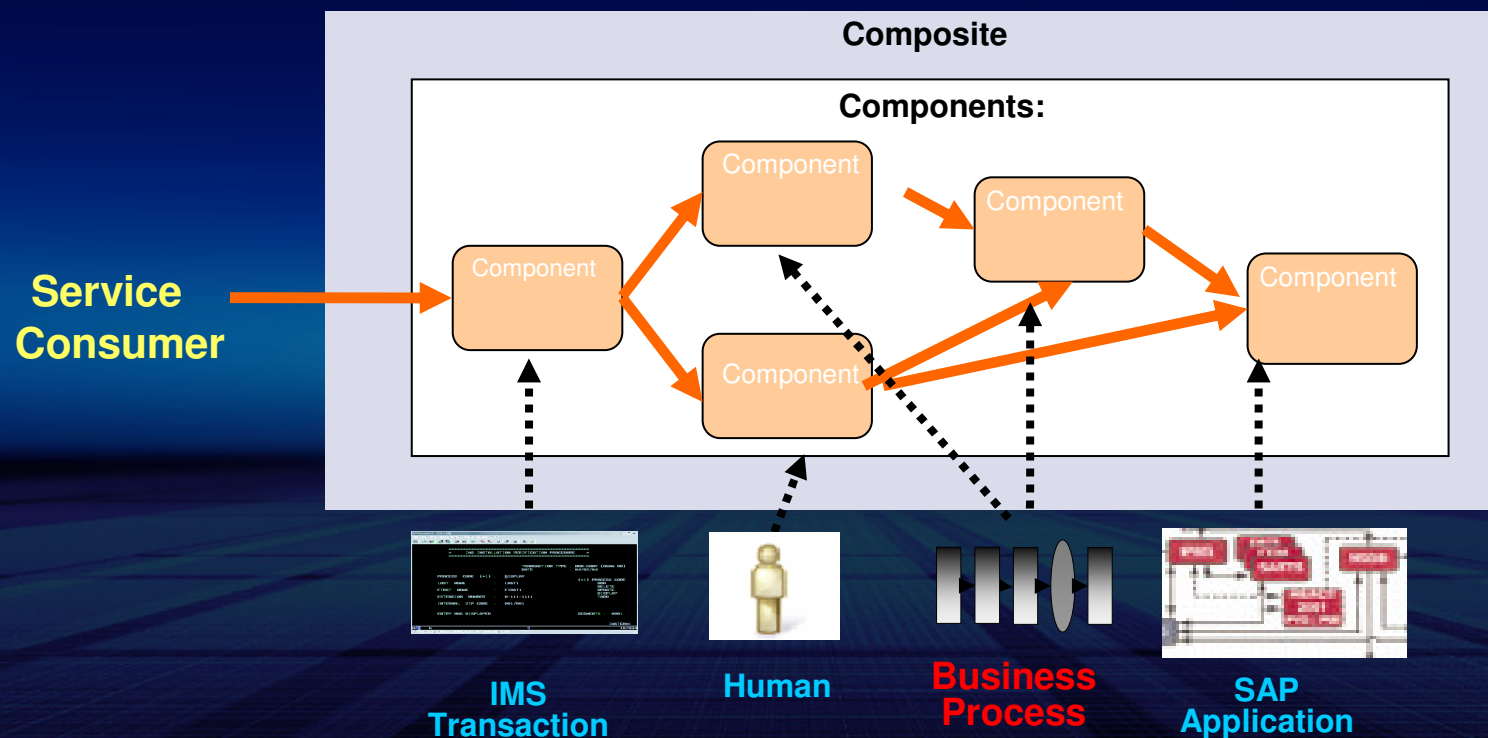
Business meets Technology with IBM

- Deploy a business process defined by business analysts to an IT infrastructure
- Model driven



Business Process as Service Component

- Composite solutions and reusable building blocks at a **business level**
- Dynamic assembly and delivery of services based on **business context**
- Present business application in a service-oriented way:
 - Web services, Enterprise Information System (EIS) service assets, business processes, etc.



A 3D rendering of the text 'Smart SOA' on a blue grid floor. The word 'Smart' is written in a white, cursive font and is positioned above the letters 'SOA'. The letters 'SOA' are large, blue, and three-dimensional, with a slight shadow cast on the floor. The background is a dark blue gradient.

Smart SOA

Take a closer look at BPEL.....

What is BPEL?

- Business Process Execution Language -
 - XML based programming language to describe high level business processes
 - A 'business process' is a term used to describe the interaction between two businesses or two elements in some business
 - Specifically geared to using Web Services
 - BPEL allows you to write programs that **use** web services, and to write programs that **are** web services.
 - BPEL—the first step towards abstracting the collaboration and sequencing logic out of platform-dependent code and into a formal process definition based on XML, WSDL, and XML Schema.

Real-world applications require the ability to express robust, **stateful**, **long-running interactions** between business applications across IT infrastructures and platforms!

BPEL definition by Prof. Frank Leymann

BPEL is a **recursive aggregation** model for Web Services

- Aggregation: A set of Web Services can be tied into one or more new Web Services by means of a business process model
- Recursive: These new Web Services can again be tied into other new Web Services

Simple BPEL lifeline

- Web Service client (user) requests something from the server (perhaps via a browser or another server)
- Client request is received by the BPEL server.
- Client request is identified as a new request and a new BPEL process is used to serve that client
- Client continues to interact with the BPEL process, making requests etc, until the interaction is complete
- BPEL process disappears and client goes about it's business

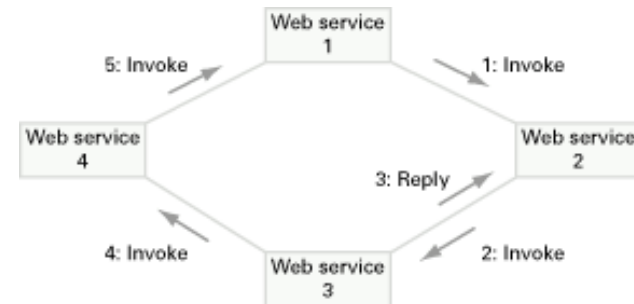
What can BPEL standard address?

- Open standard (Java/J2EE, JMS, XML, SOAP, WSDL)
- State and Context Management
- Loosely-Coupled Services
- Parallel Processing
- Exception Management
- Events/Notifications
- Open Nested Transactions
- Scalability and Reliability
- Management, Administration, and Business Visibility
- Version Control
- Audit Trailing
- Support for Existing Infrastructure

BPEL is an Orchestration/Choreography Language

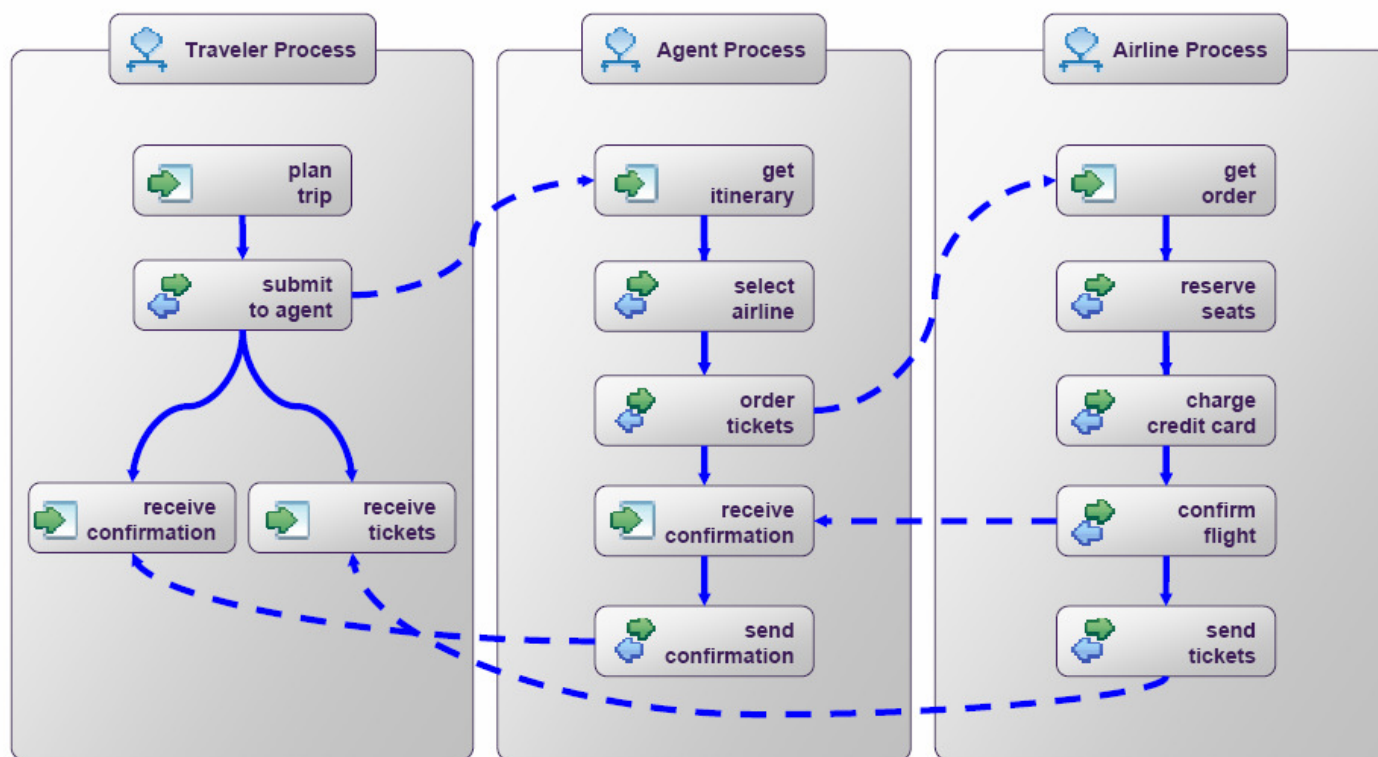
- Can BPEL be used to implement a business process??
- As a coordinator by passing messages from system to system
 - How to call
 - How to consume
 - Managing the calling of other web services.
- Coordination and management of messages as they are routed from service to service
- Thinking of applications as “the collaboration of message exchanges!”
- Forming new Web services through recombination and enables the assembly of complex business processes.

BPEL is a specification, NOT a product!!!



Orchestration (executable) vs. Choreography (abstract)

- *Executable Process*: In orchestration, there's someone — the conductor — who tells everybody in the orchestra what to do and makes sure they all play in sync. (procedure)
- *Abstract Process*: In choreography, every dancer follows a pre-defined plan — everyone independently of the others. (protocol)



BPEL & Web Services

A typical Web Service

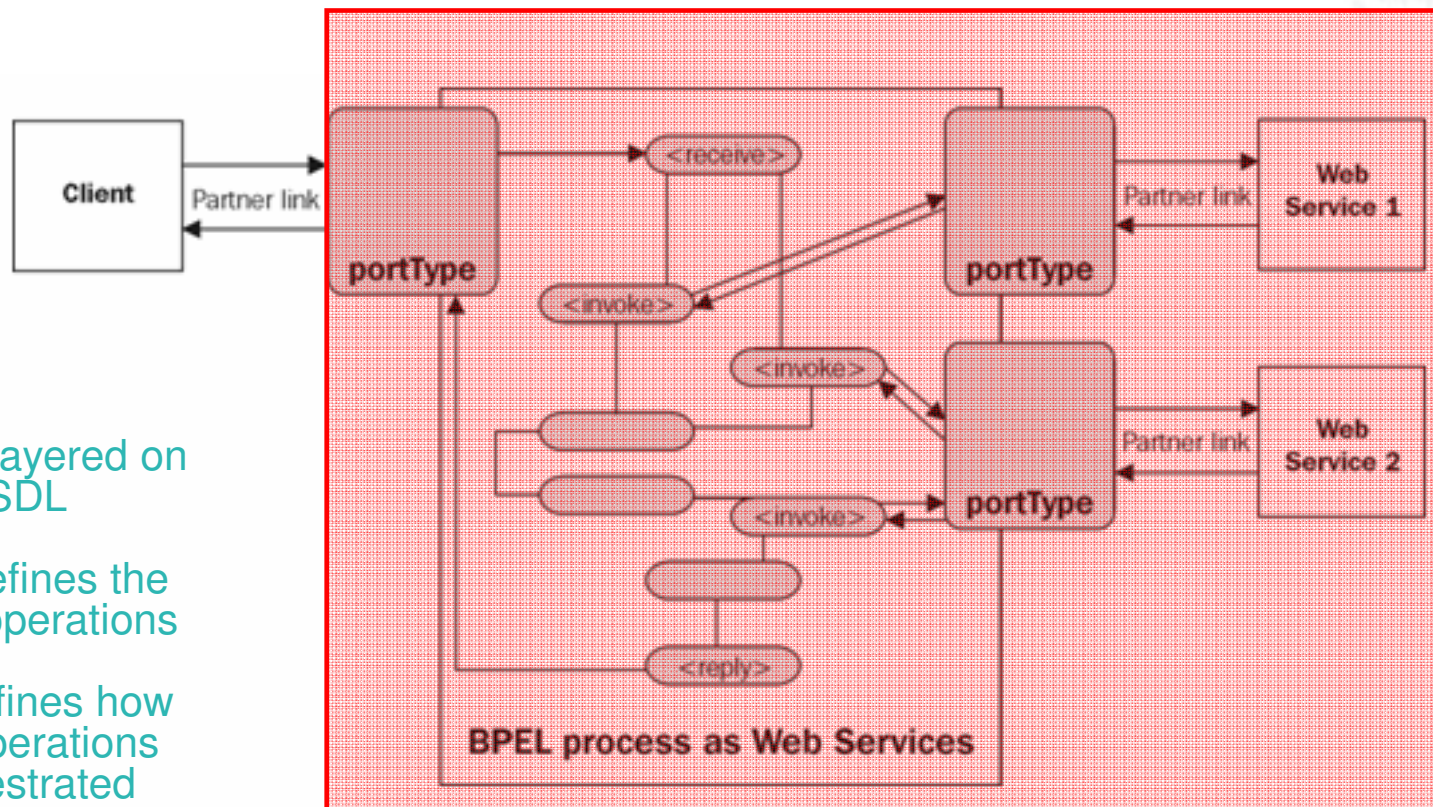
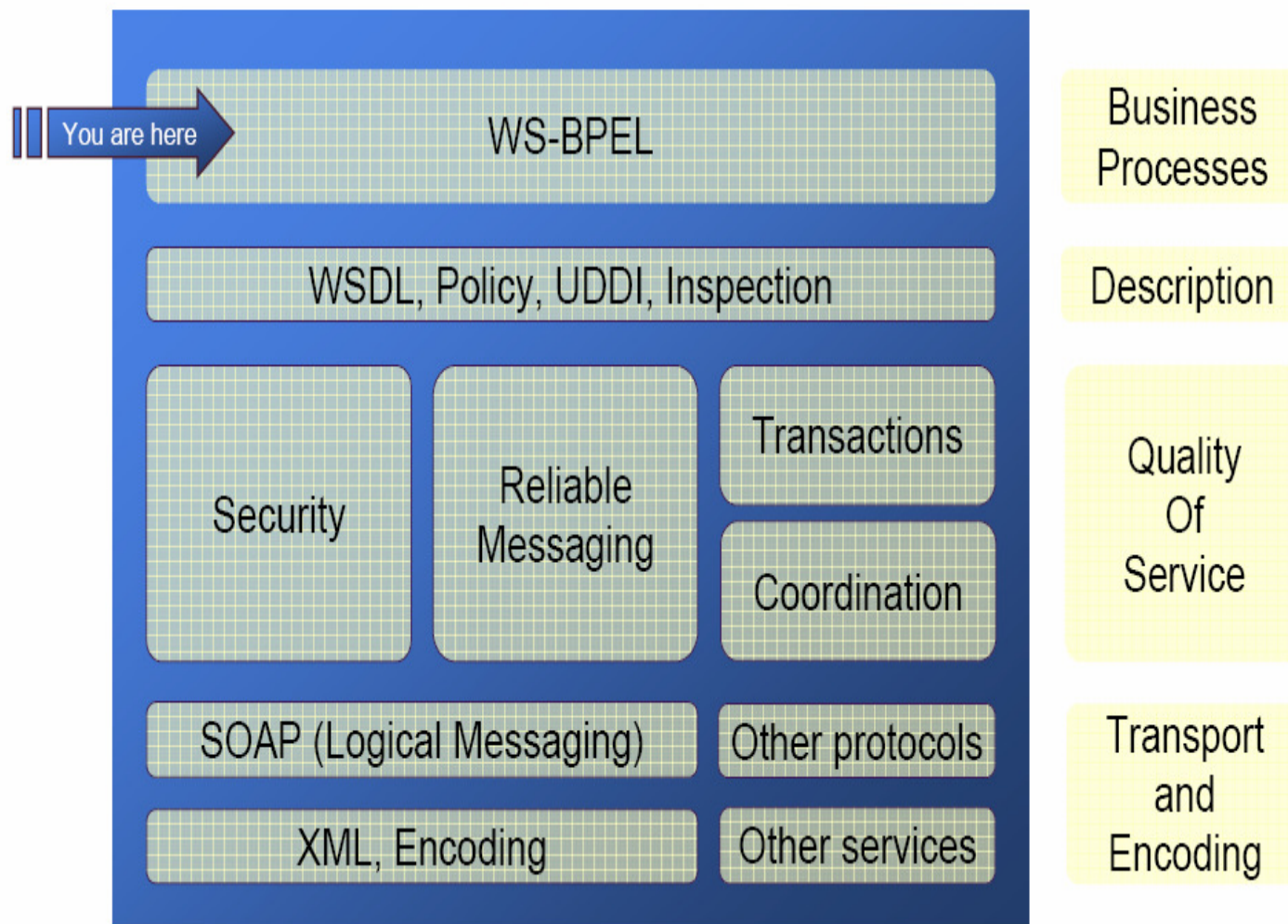


Figure: Example BPEL process

- BPEL is layered on top of WSDL
- WSDL defines the specific operations
- BPEL defines how WSDL operations are orchestrated

BPEL In The Web Service Stack



BPEL Naming

- BPEL 1.1 (BPEL4WS) vs. BPEL 2.0 (WS-BPEL)
- Changes in WS-BPEL
 - Syntactic Changes
 - Extensibility of Expression/Query Languages
 - Links
 - Messaging
 - Compensation Handling
 - *Data Manipulation*

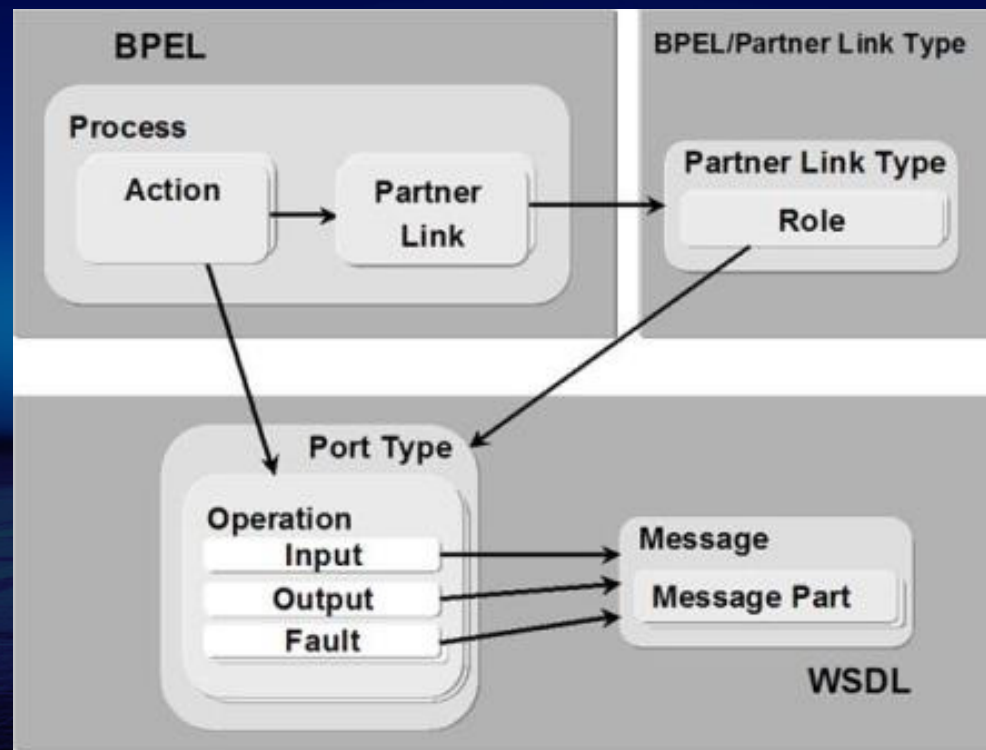
WS-BPEL 2.0 approved as an OASIS Standard for Web Services orchestration!

BPEL future extension – BPEL4People

- BPEL4People extends the capabilities of WS-BPEL to support a broad range of human interaction patterns, allowing for expanded modeling of business processes within the WS-BPEL language.
- Companies sponsoring include Active Endpoints, Adobe, BEA, IBM, Oracle, SAP, Software AG, and Sun Microsystems.

BPEL Components

- Partners: The actors in a business transaction
- Containers: The messages that need to be transmitted
- Operations: The type of Web services that are required
- Port types: The kinds of Web services connections that are required for operations



Can be found in number 8 in reference section

Elements of BPEL Process Definition

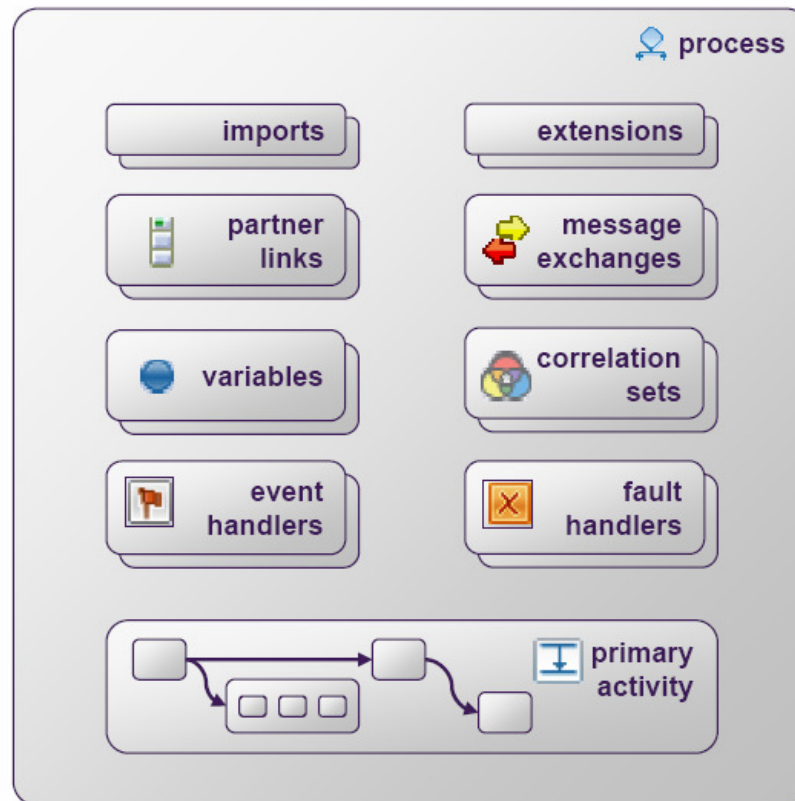
Declare dependencies on external XML Schema or WSDL definitions

Relationships that a WS-BPEL process will employ in its behavior

Data holding state of a business process or exchanged with partners

Concurrently process inbound messages or timer alarms

Perform the process logic – any number of activities may be recursively nested

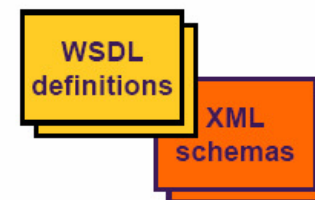


Declare namespaces of WS-BPEL extension attributes and elements

Relationship between inbound and reply message activities

Application data fields that together identify a conversation

Deal with exceptional situations in a process



BPEL Basic Activities

- Invoking an operation on some Web service (**<invoke>**)
- Waiting for a message to operation of the service's interface to be invoked by someone externally (**<receive>**)
- Generating the response of an input/output operation (**<reply>**)
- Waiting for some time (**<wait>**)
- Copying data from one place to another (**<assign>**)
- Indicating that something went wrong (**<throw>**)
- Terminating the entire service instance (**<terminate>**)
- Doing nothing (**<empty>**)

Business Process Basic Activities

Do a blocking wait for a matching message to arrive / send a message in reply

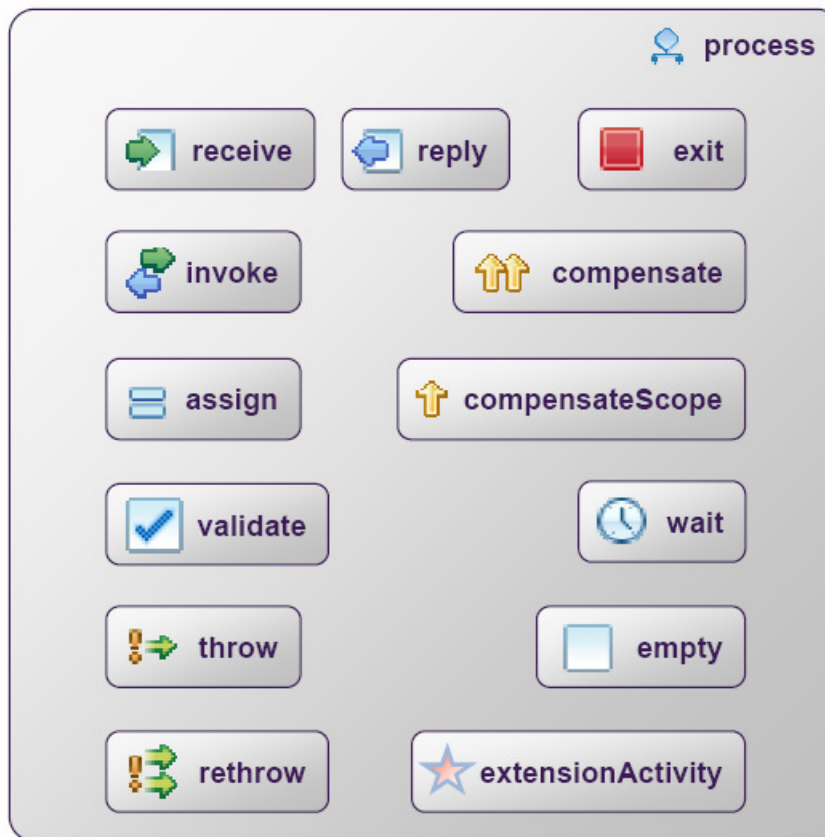
Invoke a one-way or request-response operation

Update the values of variables or partner links with new data

Validate XML data stored in variables

Generate a fault from inside the business process

Forward a fault from inside a fault handler



Immediately terminate execution of a business process instance

Invoke compensation on all completed child scopes in default order

Invoke compensation on one completed child scope

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

No-op instruction for a business process

Wrapper for language extensions

BPEL Partners

- Making invocations to other services and/or receiving invocations from *clients*
- Partners can be
 - services that the process invokes only.
 - services that invoke the process only.
 - or services that the process invokes and invoke the process (where either may occur first).

BPEL Partner Links

- Partner link types represent the **interaction** between a BPEL process and the involved parties, including
 - The Web services the BPEL process invokes
 - The client that invokes the BPEL process.

- Synchronous interaction

```
<plnk:partnerLinkType name="employeeLT">
  <plnk:role name="employeeTravelStatusService">
    <plnk:portType name="tns:EmployeeTravelStatusPT" />
  </plnk:role>
</plnk:partnerLinkType>
```

- Asynchronous interaction

```
<plnk:partnerLinkType name="flightLT">
  <plnk:role name="airlineService">
    <plnk:portType name="tns:FlightAvailabilityPT" />
  </plnk:role>

  <plnk:role name="airlineCustomer">
    <plnk:portType name="tns:FlightCallbackPT" />
  </plnk:role>
</plnk:partnerLinkType>
```

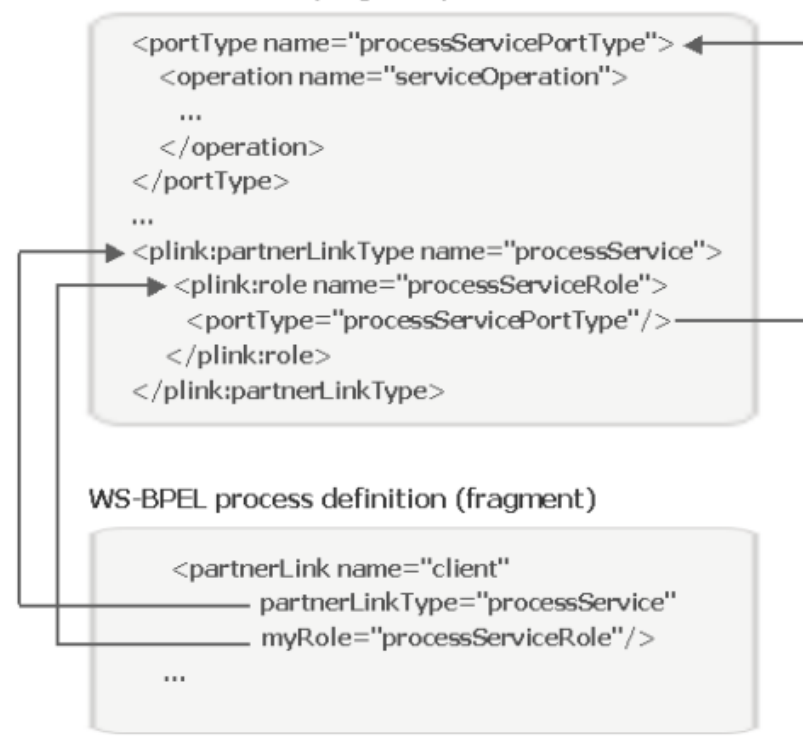
WSDL definition (fragment)

```
<portType name="processServicePortType">
  <operation name="serviceOperation">
    ...
  </operation>
</portType>
...
<plink:partnerLinkType name="processService">
  <plink:role name="processServiceRole">
    <portType="processServicePortType"/>
  </plink:role>
</plink:partnerLinkType>
```

WS-BPEL process definition (fragment)

```
<partnerLink name="client"
  partnerLinkType="processService"
  myRole="processServiceRole"/>
...

```



BPEL Variables

- Data are passed between BPEL activities using *variables*.
- Variables store the messages that are exchanged between the partners in a process and the data that is used in its business logic.
- Variables can be
 - Complete WSDL messages that allow for the storage of an entire input or output message of a Web service operation. (interface variable)
 - XML schema types that allow for the storage of individual entities that are relevant for the business process. (data type variable)

BPEL Variables Example

Variable Definition

- `<variables>`
- `<variable name="hello_world"`
- `messageType="print:PrintMessage" />`
- `</variables>`

Variable Assignment

- `<assign>`
- `<copy>`
- `<from><literal>Hello World</literal></from>`
- `<to>$hello_world.value</to>`
- `</copy>`
- `</assign>`

Web Service Invocation

- `<invoke partnerLink="printService" operation="print"`
- `inputVariable="hello_world" />`

BPEL Event Handlers

- **On Event element**

- Use this element to create a control path within an event handler, and define the source of the input that will execute it.

- **Timeout element**

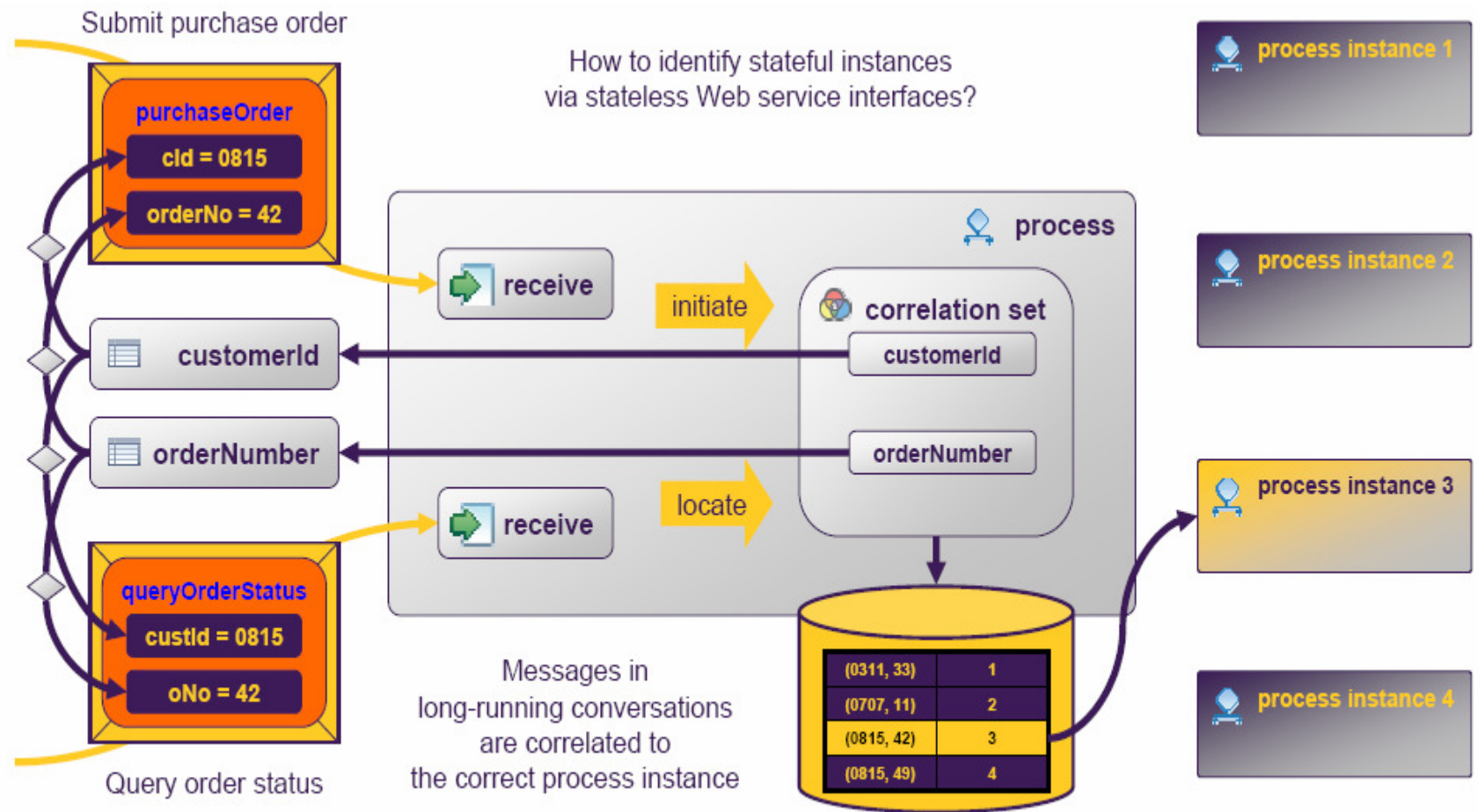
- Use this element to create a control path within an event handler that is executed when a specified time has either been reached or has elapsed.

BPEL CorrelationSet

- Correlation is the process of “**matching**” an inbound message to the BPEL engine with a specific process.
 - Matching can be done on any reasonable XML type, not just strings.
- A correlation set is a collection of properties used by the BPEL engine to identify the correct process to receive a message.
- Not needed when in synchronous call or using WS-addressing headers to pass around tokens
- Each correlation set define a way of identifying application level conversation within a business protocol
- A message can carry **multiple** correlation sets.

Message correlation is a mechanism which allows processes to participate in **stateful conversations!**

BPEL CorrelationSet



BPEL CorrelationSet - Matching

How does BPEL match "cid" to "customerID"???

Defining correlation properties such as customer ID..

```
<definitions name="properties"  
  <bpws:property name="customerID" type="xsd:string"/>  
</definitions>
```

CorrelationSet
definition

Defining purchase order and use aliasing to match..

```
<xsd:complexType name="PurchaseOrder">  
  <xsd:element name="CID" type="xsd:string"/>  
  <xsd:element name="order" type="xsd:int"/>  
  ...  
</xsd:complexType>
```

Incoming message
definition (purchase
order)

```
<message name="POMessage">  
  <part name="PO" type="tns:PurchaseOrder"/>  
</message>
```

```
<bpws:propertyAlias propertyName="cor:customerID"  
  messageType="tns:POMessage" part="PO"  
  query="/PO/CID"/>
```


BPEL CorrelationSet Example

- In WSDL:
 - `<bpws:propertyAlias propertyName="Ins:applicantFirstName" messageType="loandef:creditInformationMessage" part="firstName" query="/firstName"/>`
 - `<bpws:propertyAlias propertyName="Ins:applicantLastName" messageType="loandef:creditInformationMessage" part="name" query="/name"/>`
- In BPEL:
 - `<correlationSets>`
 - `<correlationSet name="loanIdentifier" properties="Ins:applicantFirstName Ins:applicantLastName"/>`
 - `</correlationSets>`
 - `<receive name="acceptance-receive" partner="customer" portType="Ins:loanApprovalPT" operation="obtain" container="acceptanceRequest">`
 - `<target linkName="reply-to-receive"/>`
 - `<source linkName="receive-to-grant"/>`
 - `<correlations>`
 - `<correlation set="loanIdentifier"/>`
 - `</correlations>`
 - `</receive>`

BPEL CorrelationSet in action.. (Async)

Seller received Async Purchase Order...

- `<receive partnerLink="Buyer" portType="SP:PurchasingPT"`
- `operation="AsyncPurchase"`
- `variable="PO">`
- `<correlations>`
- `<correlation set="PurchaseOrder" initiate="yes">`
- `</correlations>`
- `</receive>`

Condition 1: Seller replies with invoice...

- `<invoke partnerLink="Buyer" portType="SP:BuyerPT"`
- `operation="AsyncPurchaseResponse" inputVariable="POResponse">`
- `<correlations>`
- `<correlation set="PurchaseOrder" initiate="no">`
- `<correlation set="Invoice" initiate="yes">`
- `</correlations>`
- `</invoke>`

Condition 2: Seller rejects the order...

- `<invoke partnerLink="Buyer" portType="SP:BuyerPT"`
- `operation="AsyncPurchaseReject" inputVariable="POReject">`
- `<correlations>`
- `<correlation set="PurchaseOrder">`
- `</correlations>`
- `</invoke>`

BPEL CorrelationSet in action continued.. (Sync)

Buyer initiated Sync Purchase Order...

- `<invoke partnerLink="Seller" portType="SP:PurchasingPT"`
- `operation="SyncPurchase"`
- `inputVariable="sendPO"`
- `outputVariable="getResponse">`
- `<correlations>`
- `<correlation set="PurchaseOrder" initiate="yes" pattern="out">`
- `<correlation set="Invoice" initiate="yes" pattern="in">`
- `</correlations>`
- `<catch faultName="SP:RejectPO" faultVariable="POReject">`
- `<!-- handle the fault -->`
- `</catch>`
- `</invoke>`

BPEL Fault Handler

- Use the fault handler to associate specific fault activities on either an invoke or a scope activity that will execute when a fault is thrown by the invoke activity or an activity inside the scope activity.
- Different kinds of fault
 - Faults explicitly thrown in BPEL process
 - Faults thrown due to the consumption of partner service
- Fault handling activities
 - Compensation activity
 - Use this activity within a scope's fault or compensation handler to invoke a specific compensation handler within the scope.
 - Re-throw activity
 - Use this activity to take an internal fault that has already been thrown, and throw it again.
 - Terminate activity
 - Use this activity to halt the execution of a process.
 - Throw activity
 - Use this activity to signal an internal fault.

BPEL Fault Handler Example

- <faultHandlers>
- <catch faultName="Ins:loanProcessFault" faultContainer="error">
- <sequence name="fault-sequence">
- <assign>
- <copy>
- <from expression="invalid request: amount too high"/>
- <to container="approvalInfo" part="accept"/>
- </copy>
- </assign>
- <reply partner="customer" portType="Ins:loanApprovalPT"
- operation="obtain" container="approvalInfo" faultName="
- "Ins:loanProcessFault"/>
- </sequence>
- </catch>
- </faultHandlers>

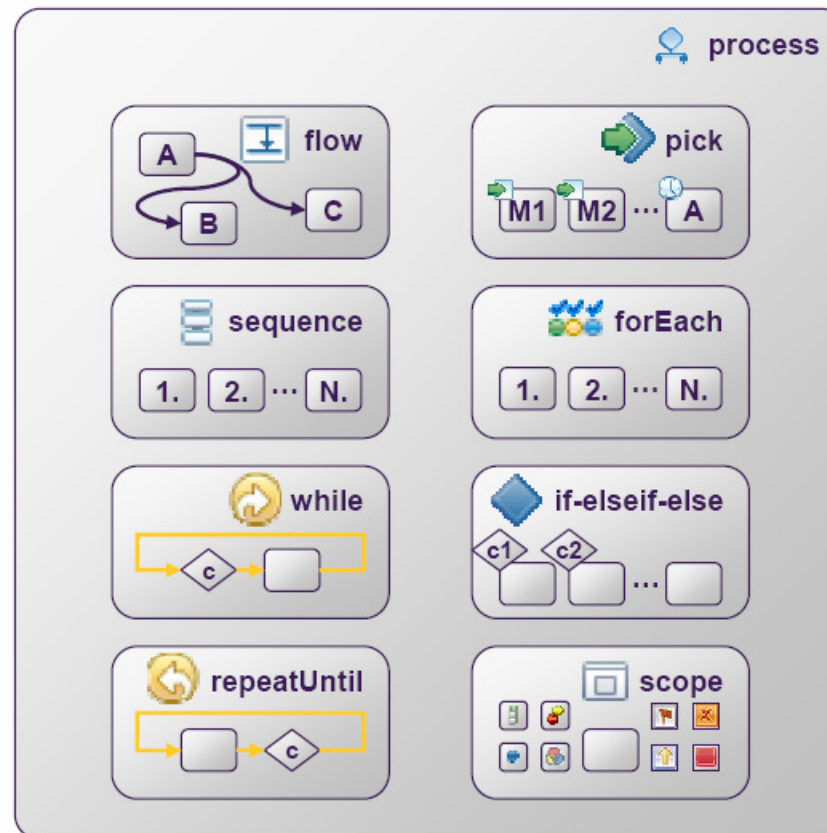
BPEL Structured Activities

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

Contained activities are performed sequentially in lexical order

Contained activity is repeated while a predicate holds

Contained activity is repeated until a predicate holds



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

Contained activity is performed sequentially or in parallel, controlled by a specified counter variable

Select exactly one branch of activity from a set of choices

Associate contained activity with its own local variables, partner links, etc., and handlers

BPEL Structured Activities – Pick

- Exactly one of the branches will be selected based on the occurrence of the event associated with it before any others. Note that after the pick activity has accepted an event for handling, the other event are no longer accepted by that pick.
- `<pick>`
- `<onMessage partnerLink="buyer"`
- `portType="orderEntry"`
- `operation="inputLineItem"`
- `variable="lineItem">`
- `<!-- activity to add line item to order -->`
- `</onMessage>`
- `<onMessage partnerLink="buyer"`
- `portType="orderEntry"`
- `operation="orderComplete"`
- `variable="completionDetail">`
- `<!-- activity to perform order completion -->`
- `</onMessage>`
- `<!-- set an alarm to go after 3 days and 10 hours -->`
- `<onAlarm for=""P3DT10H"">`
- `<!-- handle timeout for order completion -->`
- `</onAlarm>`
- `</pick>`

BPEL Structured Activities – Flow

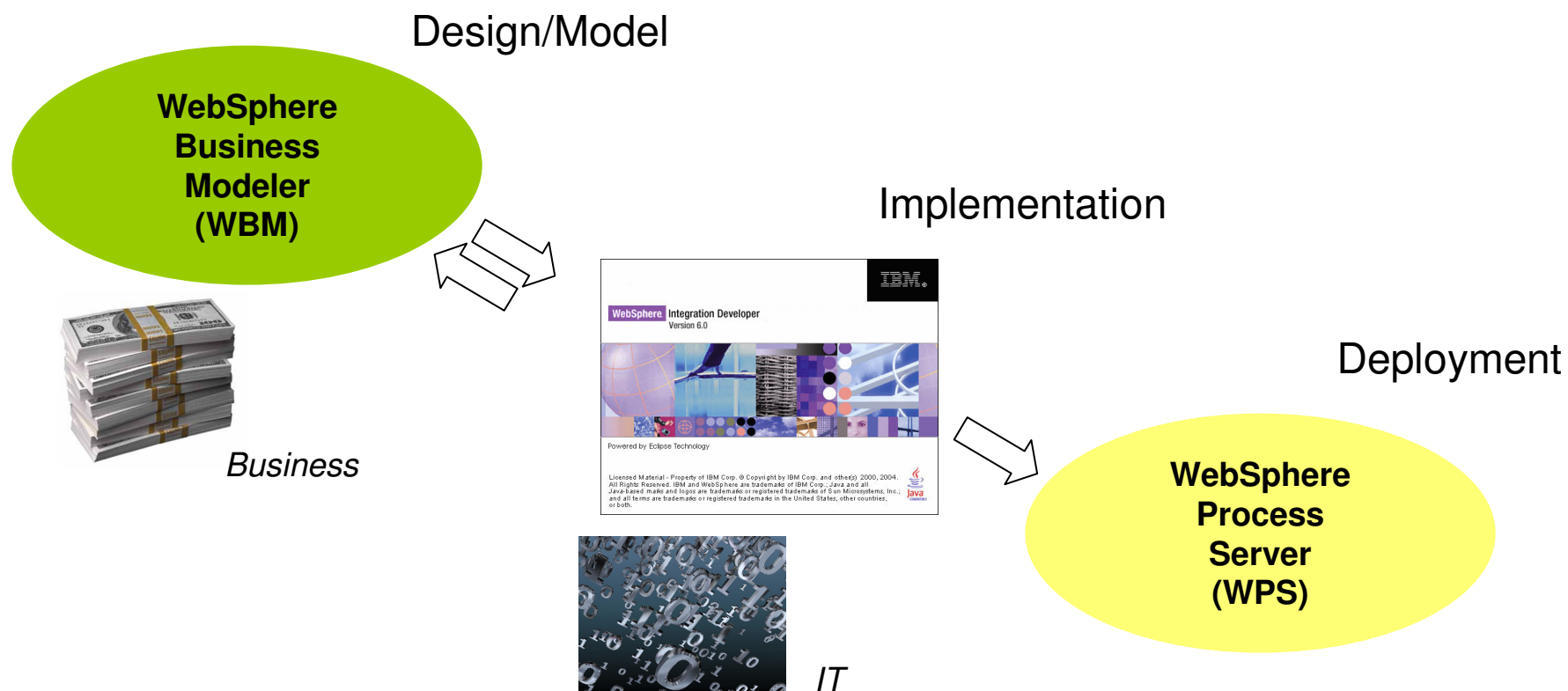
- The flow construct provides concurrency and synchronization (link). A flow completes when all of the activities in the flow have completed.
- `<sequence>`
 - `<flow>`
 - `<invoke partnerLink="Seller" .../>`
 - `<invoke partnerLink="Shipper" .../>`
 - `</flow>`
 - `<invoke partnerLink="Bank" .../>`
- `</sequence>`

IBM BPEL Model Implementation Tool

Smart
SOA

WebSphere Integration Developer

IBM BPEL Suites



WebSphere Integration Developer (WID)

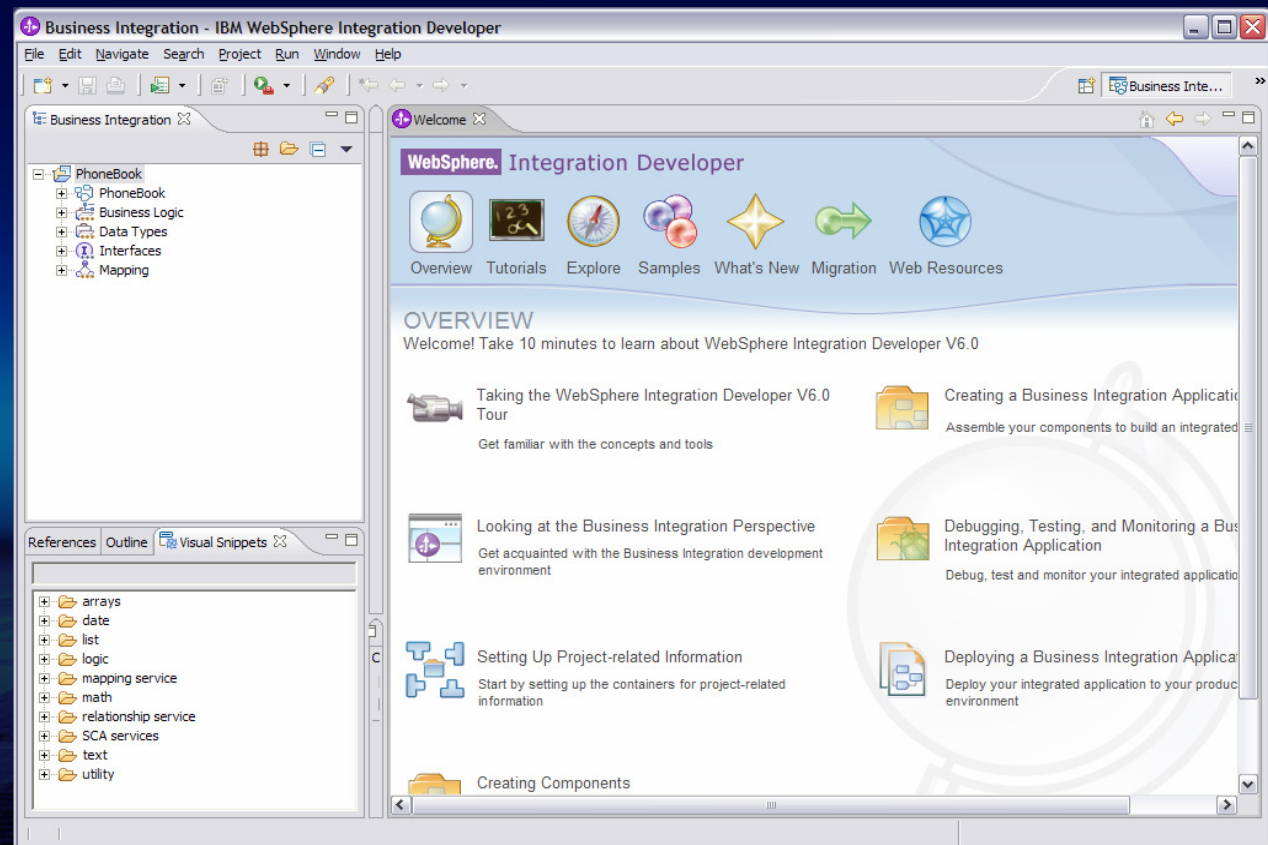
- Integration Development Environment (IDE)
- Focus: Integration between business units and enterprises



WebSphere Integration
Developer

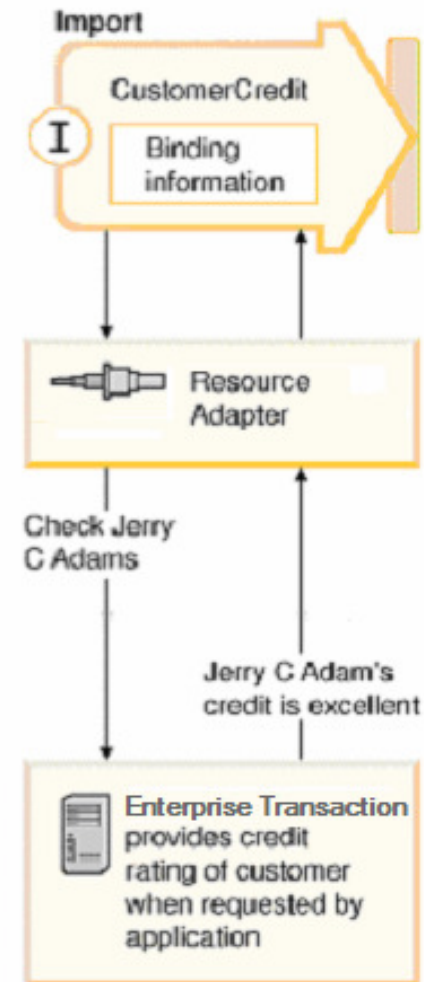
Rational Application
Developer

Eclipse



Accessing Enterprise Application

- IMS or CICS application represented as an **EIS Reference Partner**
- IMS TM Resource Adapter (a.k.a. IMS Connector for Java) submits transactions to IMS Connect
- CICS ECI Resource Adapter submits transactions to CICS Transaction Gateway

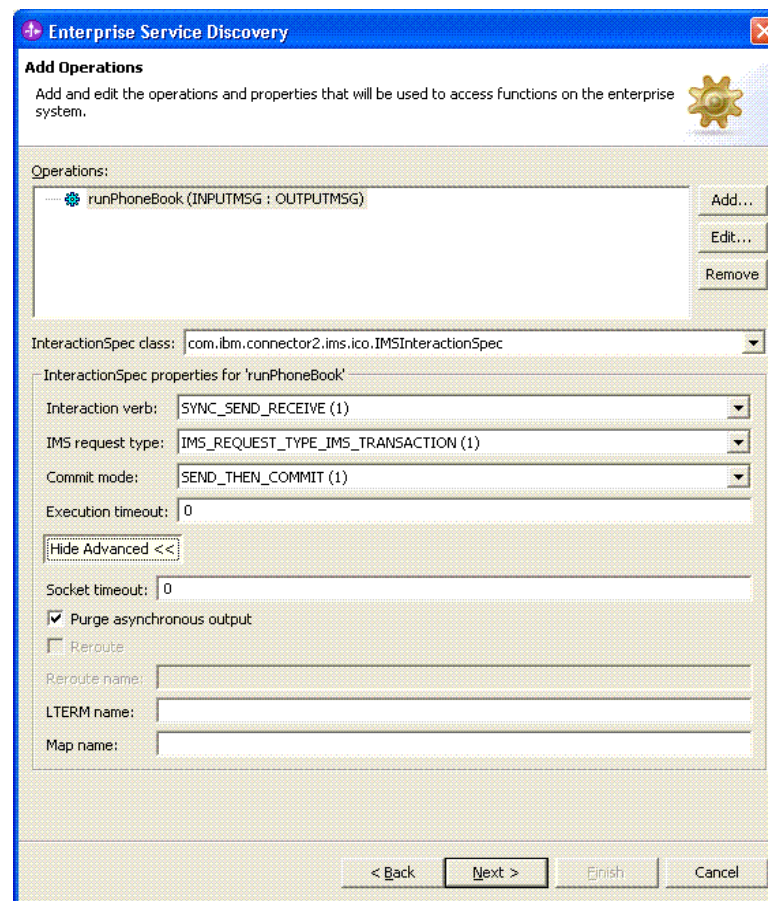


Create an EIS Reference Partner

1. Run Enterprise Service Discovery wizard:

- Import IMS/CICS Resource Adapter
- Specify Connection Properties
- Add Operation
 - Import Cobol copybook / C source
 - Specify Importer properties
- Specify Interaction Spec Properties

2. Add Reference Partner to Business Process Component



Enterprise Service Discovery

Add Operations
Add and edit the operations and properties that will be used to access functions on the enterprise system.

Operations:

- runPhoneBook (INPUTMSG : OUTPUTMSG)

Buttons: Add..., Edit..., Remove

InteractionSpec class: com.ibm.connector2.ims.ico.IMSInteractionSpec

InteractionSpec properties for 'runPhoneBook'

Interaction verb: SYNC_SEND_RECEIVE (1)

IMS request type: IMS_REQUEST_TYPE_IMS_TRANSACTION (1)

Commit mode: SEND_THEN_COMMIT (1)

Execution timeout: 0

Hide Advanced <<

Socket timeout: 0

Purge asynchronous output

Reroute

Reroute name: _____

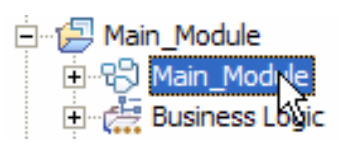
LTERM name: _____

Map name: _____

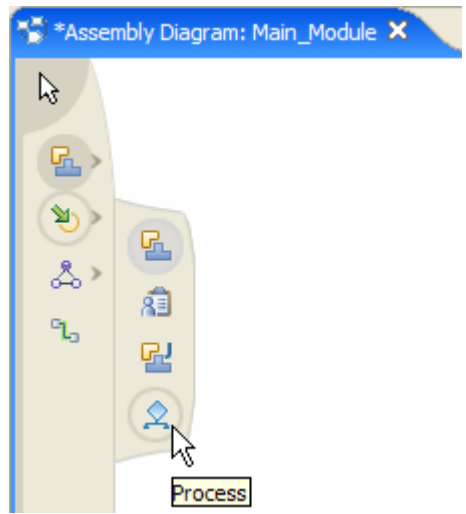
Buttons: < Back, Next >, Finish, Cancel

Define and Implement Business Process

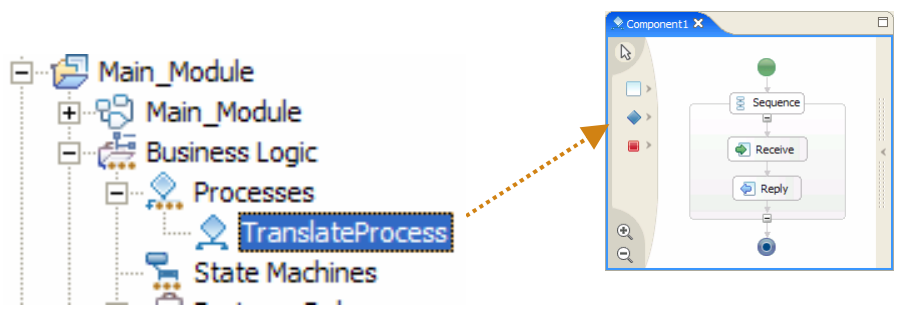
1 Open Module in Assembly Diagram Editor
- Double click on the Module



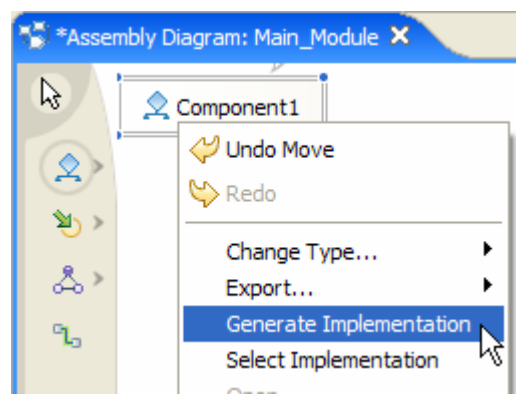
2 Drag and drop Business Process Component from the palette



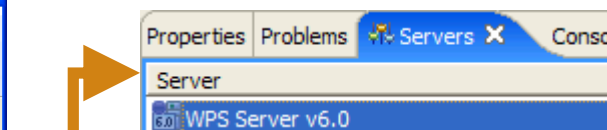
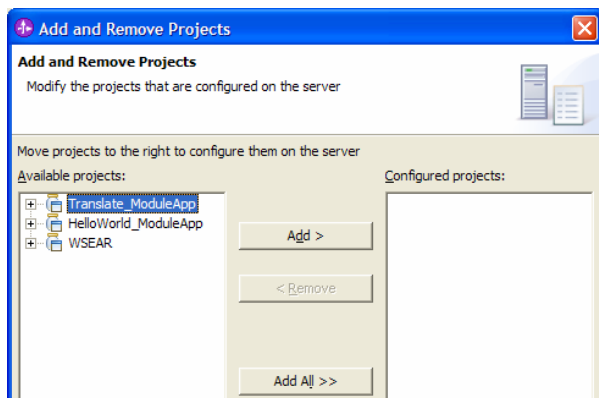
3 Opens Business Process Editor and add activities, partner, variables, or correlation set. Component appears in the Component tree



4 Add Interface/References and generate implementation



Test and Debug



Deploy

Launch Integrated Test Client

Configuration: Default Module Test

Module: Main_Module

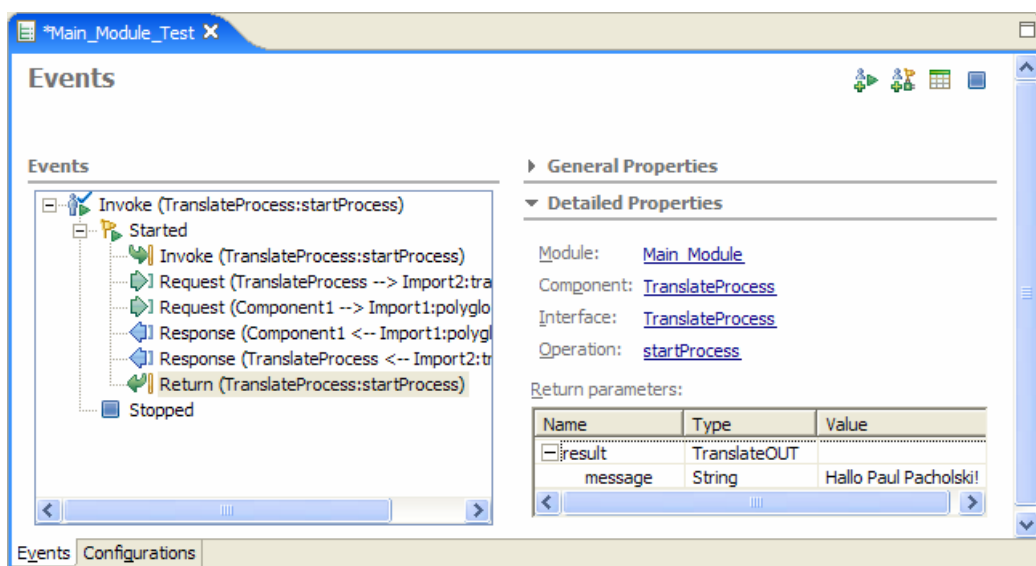
Component: TranslateProcess

Interface: TranslateProcess

Operation: startProcess

Initial request parameters

Name	Type	Value
input1	Translate_IN	
name	string	Paul Pacholski
message	string	hello
language	string	german



Continue

Enter Input Data & Launch Operation

Sample Business Process for Insurance Claim



Available Resource Adapters:
 IMS, CICS, SAP, PeopleSoft,
 Siebel, FTP, Email, FlatFile,
 JDBC, JDE

Reference Partner to EIS IMS Application

Reference Partner to EIS IMS Application

Reference Partner to SAP Application

```

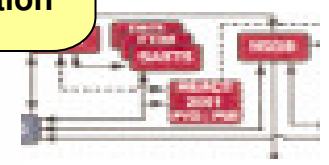
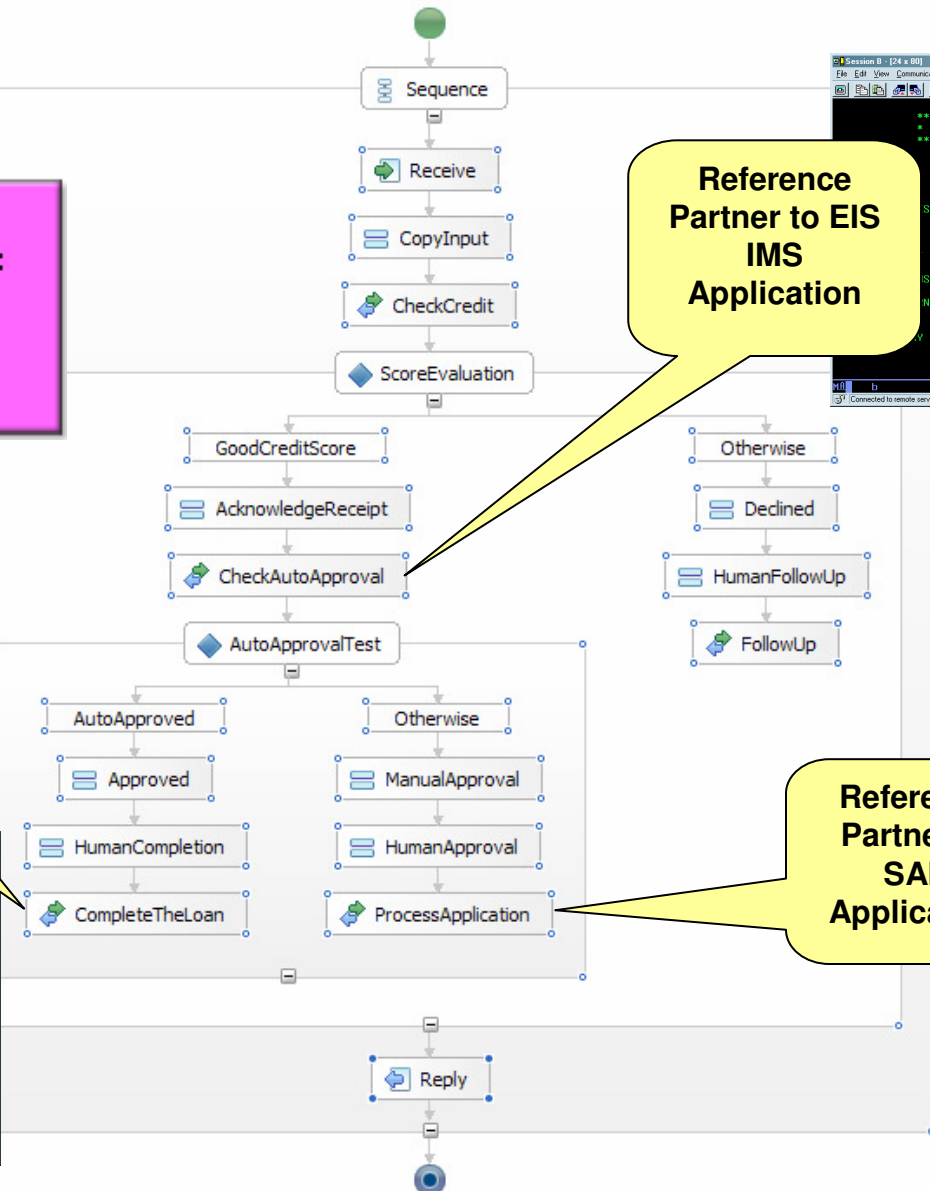
Session 8 : [24 x 80]
File Edit View Communication Actions Window Help
-----
* IMS INSTALLATION VERIFICATION PROCEDURE *
-----
TRANSACTION TYPE : NON-CONV (DSAM DB)
DATE              : 04/02/04

SS CODE (*1) : DISPLAY
NAME         : LAST1
FIRST NAME   : FIRST1
EXTENSION NUMBER : 8-111-1111
INTERNAL ZIP CODE : 001/R01
ENTRY WAS DISPLAYED
SEGMENT# : 0001
    
```

```

Session 8 : [24 x 80]
File Edit View Communication Actions Window Help
-----
* IMS INSTALLATION VERIFICATION PROCEDURE *
-----
TRANSACTION TYPE : NON-CONV (DSAM DB)
DATE              : 04/02/04

PROCESS CODE (*1) : DISPLAY
LAST NAME         : LAST1
FIRST NAME       : FIRST1
EXTENSION NUMBER : 8-111-1111
INTERNAL ZIP CODE : 001/R01
ENTRY WAS DISPLAYED
SEGMENT# : 0001
    
```



2008 IMPACT

Questions & Answers

References

2008 IMPACT

1. <http://www.developer.com/services/article.php/3609381>
2. http://en.wikipedia.org/wiki/User:Jayvdb/Comparison_of_BPEL_engines
3. http://www.ibm.com/developerworks/webservices/library/ws-bpelcol7/?S_TACT=105AGX04&S_CMP=EDU
4. <http://www-128.ibm.com/developerworks/webservices/library/ws-bpelcol6/>
5. http://webservices.sys-con.com/read/155617_1.htm
6. http://www.ibm.com/developerworks/websphere/library/techarticles/0605_marshall/0605_marshall.html
7. ftp://ftp.software.ibm.com/common/ssi/rep_sp/9/G2102159/G2102159.PDF
8. <http://www.theserverside.com/tt/articles/article.tss?l=BPELJava>
9. <http://ode.apache.org/ws-bpel-20.html>
10. http://www.innoq.com/blog/st/2005/02/16/choreography_vs_orchestration.html
11. <http://www.dmreview.com/issues/20070501/1082553-1.html>
12. <http://www-935.ibm.com/services/au/cio/flexible/enflex-cs-pacorini.pdf>
13. <http://www.oracle.com/technologies/integration/pdf/bpel-case-whitepaper.pdf>
14. <http://www.ibm.com/developerworks/webservices/library/ws-bpelcol7/>

© IBM Corporation 2007. All Rights Reserved.

The workshops, sessions and materials have been prepared by IBM or the session speakers and reflect their own views. They are provided for informational purposes only, and are neither intended to, nor shall have the effect of being, legal or other guidance or advice to any participant. While efforts were made to verify the completeness and accuracy of the information contained in this presentation, it is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this presentation or any other materials. Nothing contained in this presentation is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries. For a complete list of IBM trademarks, see www.ibm.com/legal/copytrade.shtml

AIX, CICS, CICSplex, DB2, DB2 Universal Database, i5/OS, IBM, the IBM logo, IMS, iSeries, Lotus, OMEGAMON, OS/390, Parallel Sysplex, pureXML, Rational, RCAF, Redbooks, Sametime, Smart SOA, System i, System i5, System z, Tivoli, WebSphere, and z/OS.

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

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

Intel and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries 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.