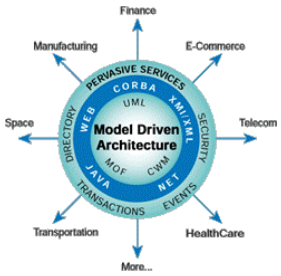
Mapping from UML to the Business Process Execution Language for Web Services (BPEL4WS)

Tracy Gardner <tgardner@uk.ibm.com>
IBM UK Laboratories, Hursley Park



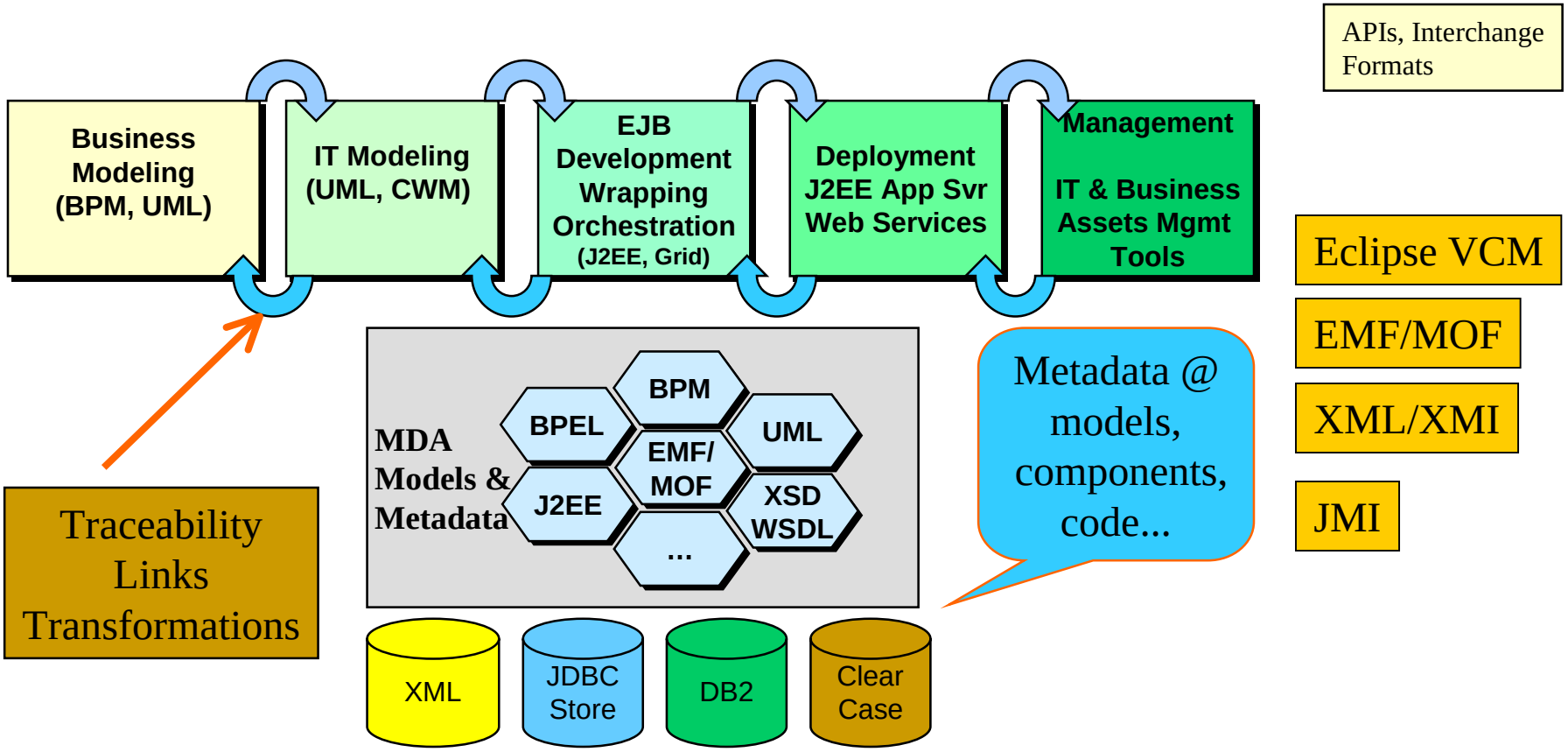
OMG MDA Implementers' Workshop
Orlando, May 2003

UML to BPEL4WS

- BPEL4WS is a language for specifying business processes which can be executed on a BPEL4WS runtime
- **Goal: Support automated mapping from (a profile of) UML to BPEL4WS**
- Why UML?
  - UML is a widely adopted standard: many of our customers and ISVs use UML

BPEL4WS, WSDL, XSD

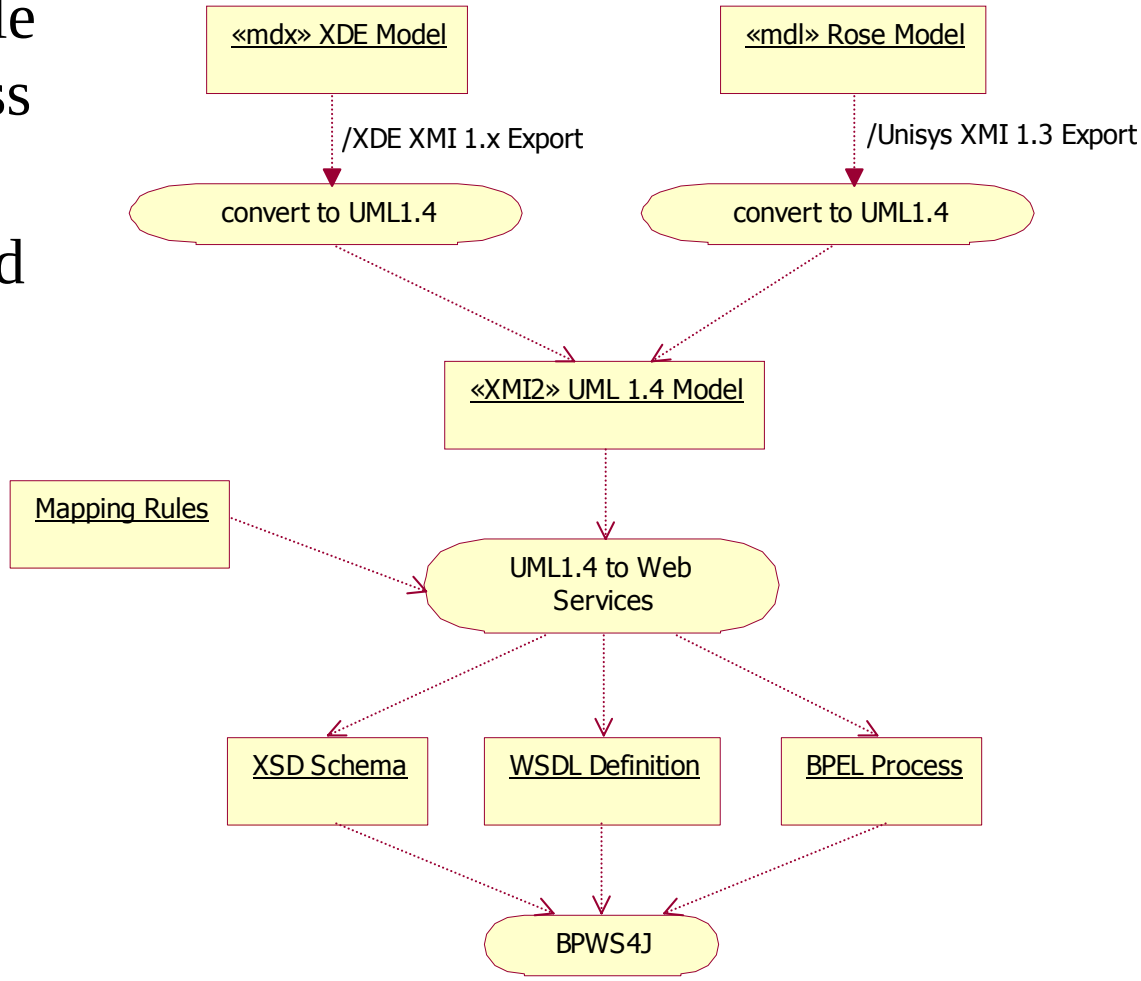
Model Driven Tools Integration
An end to end view - A peek ahead?



Business Modeling : MDA Computation Independent Model (BPM)
IT Modeling : MDA Platform Independent Model (UML, CWM)
MDA Platform Specific Model (J2EE…
Model Transformations across layers

UML to BPEL

- Start with a UML model conforming to a profile for automated business processes
- Translate to BPEL and related web services artifacts



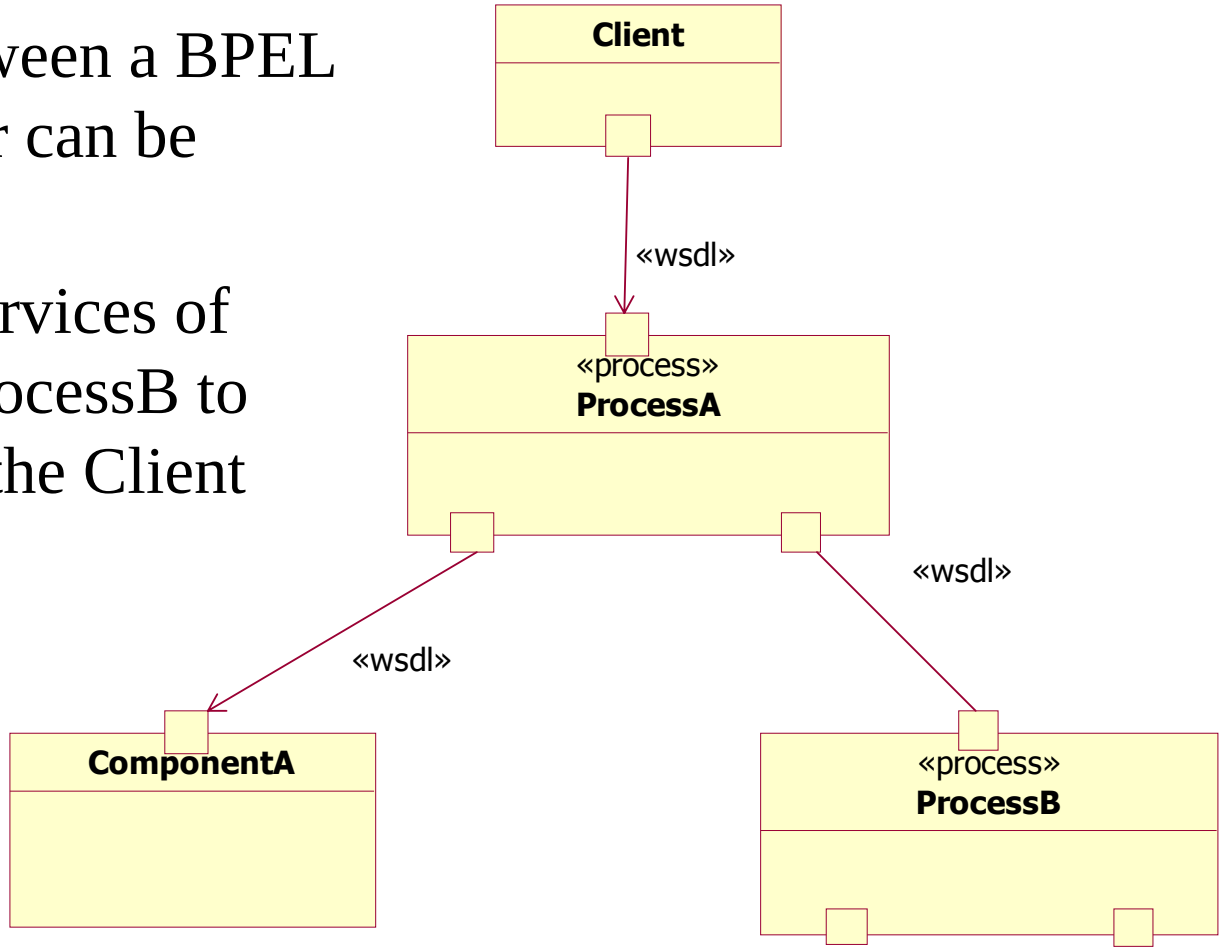
BPEL4WS

- The Business Process Execution Language for Web Services (BPEL4WS) provides an XML notation and semantics for specifying business process behavior based on Web Services.

- A BPEL4WS process is defined in terms of its interactions with partners. A partner may provide services to the process, require services from the process, or participate in a two-way interaction with the process.

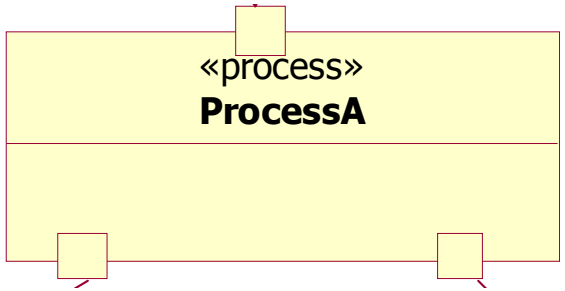
BPEL Overview

- All interfaces are expressed in WSDL
- Communication between a BPEL process and a partner can be one-way or two-way
- ProcessA uses the services of ComponentA and ProcessB to provide a service to the Client component



Automated Business Process Profile Scope

- This profile is concerned with modeling individual process components which will be deployed as BPEL processes



- The modeling of solutions containing many components which are ‘wired together’ will be covered in a component profile (work in progress).

A BPEL 'program'

state

interaction points

behavior

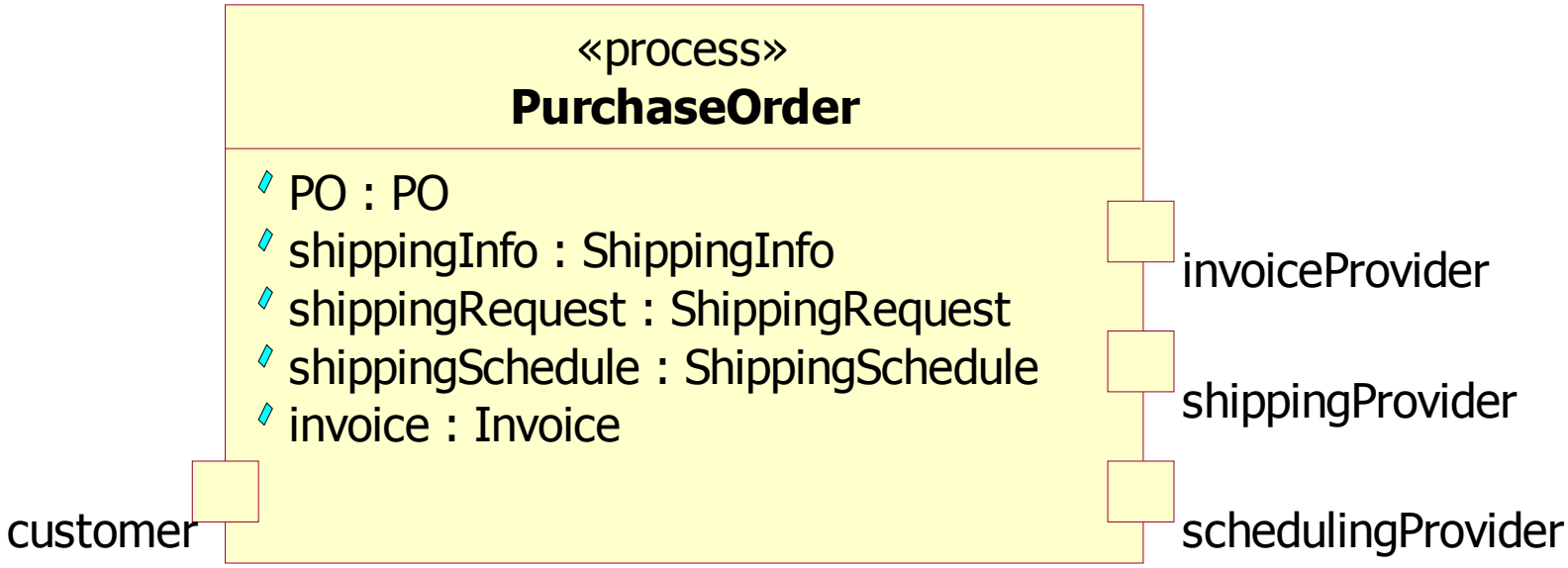
```
<process name="purchaseOrderProcess" ...>

   <containers>
      <container name="PO" messageType="lns:POMessage"/>
      <container name="Invoice" messageType="lns:InvMessage"/>
      ...
   </containers>

   <partners>
      <partner name="customer" serviceLinkType="lns:purchaseLT"
               myRole="purchaseService"/>
      ...
   </partners>

   <sequence>
      <receive partner="customer"
               portType="lns:purchaseOrderPT"
               operation="sendPurchaseOrder"
               container="PO">
      </receive>
      ...
      <reply partner="customer" portType="lns:purchasePT"
             operation="sendPurchaseOrder"
             container="Invoice"/>
   </sequence>

</process>
```


Profile Principles

- The profile should cover broadly the same set of concepts as BPEL
- Support the concepts of XSD and WSDL that are required to support BPEL, but don't cover the whole of service oriented architecture in this profile
- Standard UML terminology for concepts is used where available, e.g. Interface rather than PortType
- Where UML 2 will have more direct support for concepts then the profile adopts a UML 2 style (e.g. introducing a notion of ports)
- In areas of UML that are better defined in UML 2 then the UML 2 semantics is assumed
- It should be possible to create models conforming to the profile using multiple UML editors, specifically Rose and XDE

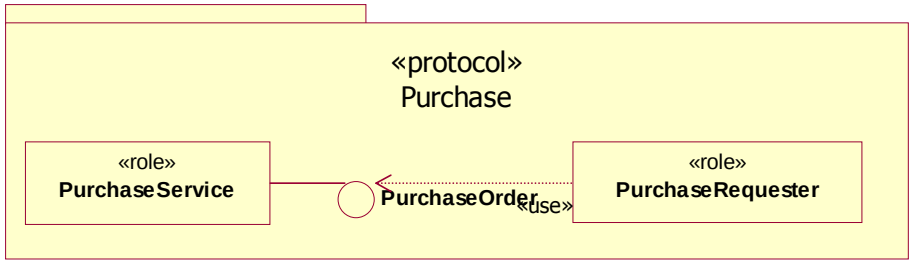
BPEL partners as ports

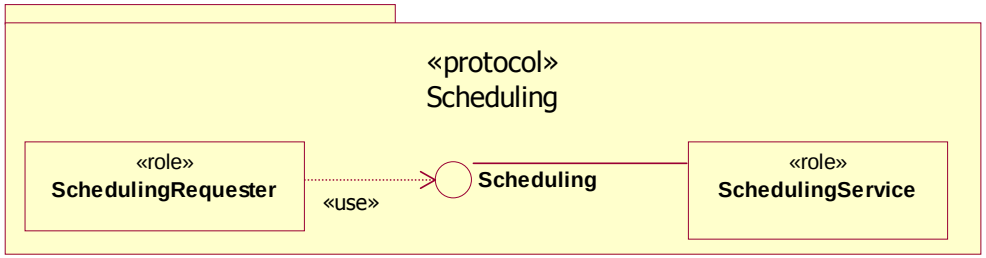
- This is the UML 2 version, we approximate this in UML 1.4.

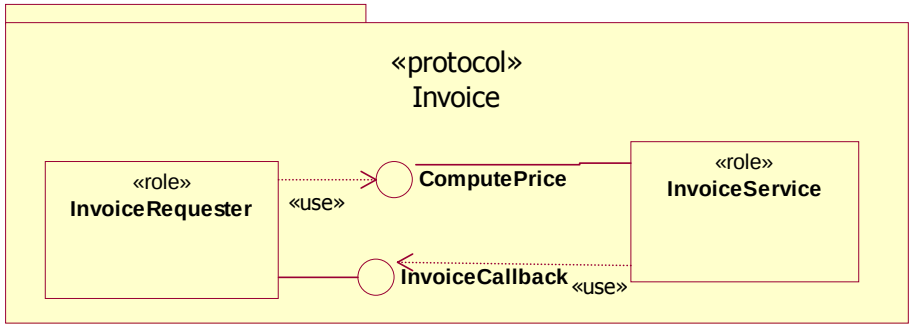


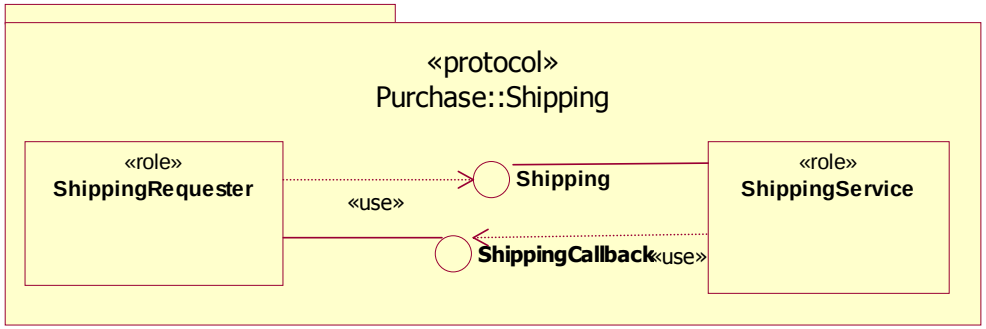
Protocols

- Protocols are defined independently of the processes that use them
- BPEL protocols are always binary
- Roles provide ‘port types’ – groupings of provided and required interfaces that must be supported through the same port

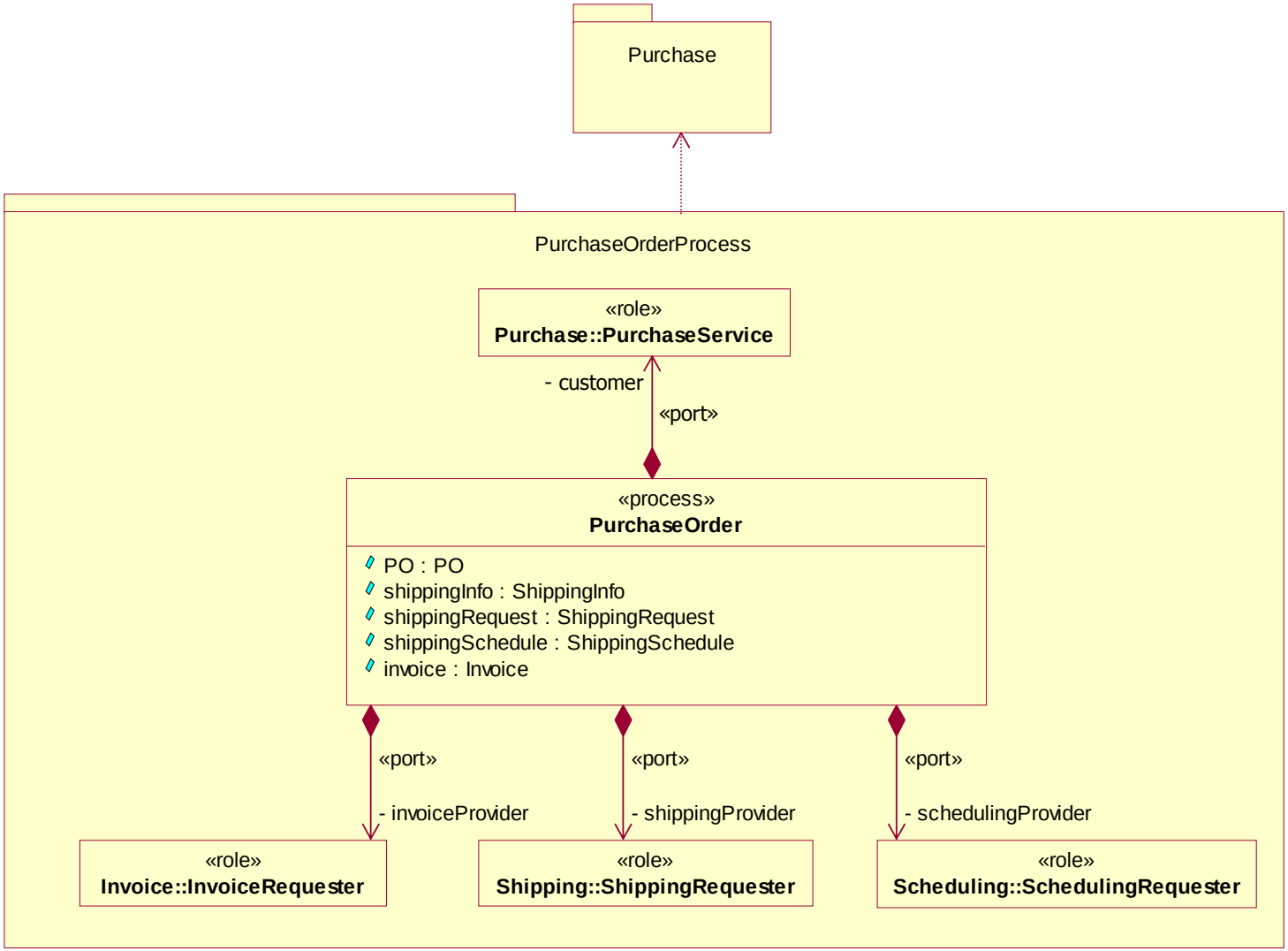






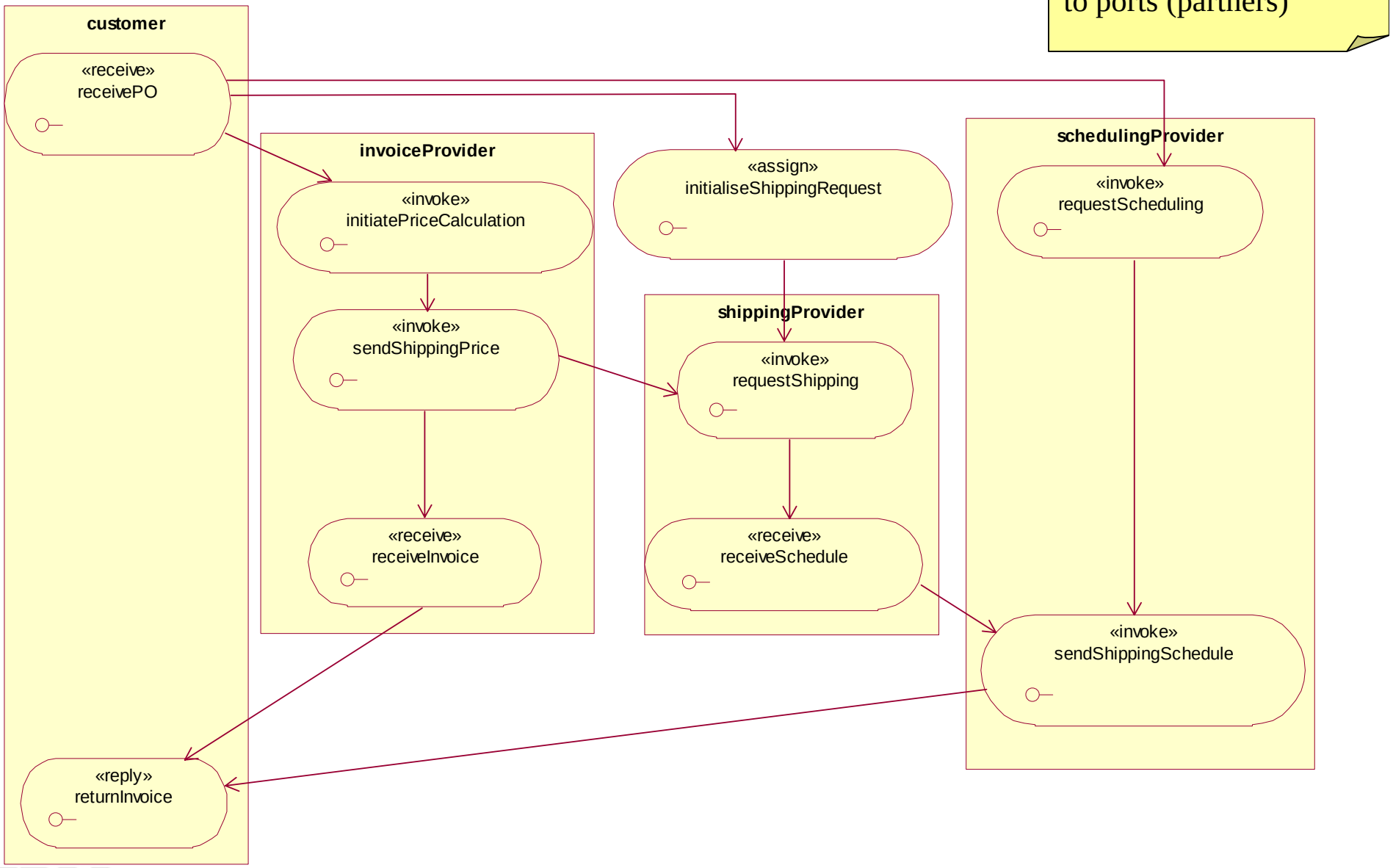


Purchase Order Process - Structure



Purchase Order Process

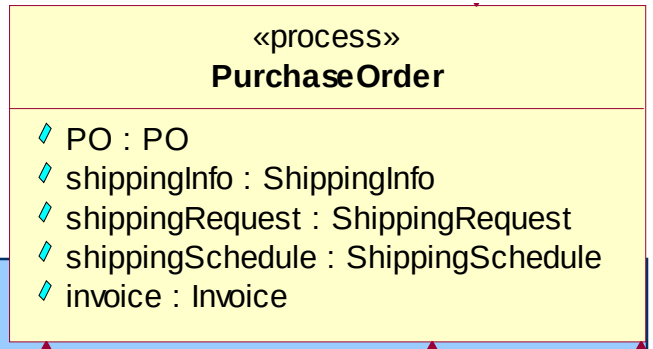
Partitions correspond to ports (partners)



BPEL Mapping Overview

<<process>> class	BPEL process definition
Activity graph on a <<process>> class	BPEL activity hierarchy
<<port>> associations	BPEL partner declarations
<<process>> class attributes	BPEL containers
Hierarchical structure and control flow	BPEL sequence and flow activities
Decision nodes	BPEL switch activities and transition conditions
<<receive>>, <<reply>>, <<invoke>> activities	BPEL receive, reply, invoke activities
<<protocol>> package with <<role>> classes	BPEL service links types and roles

Mapping: Process

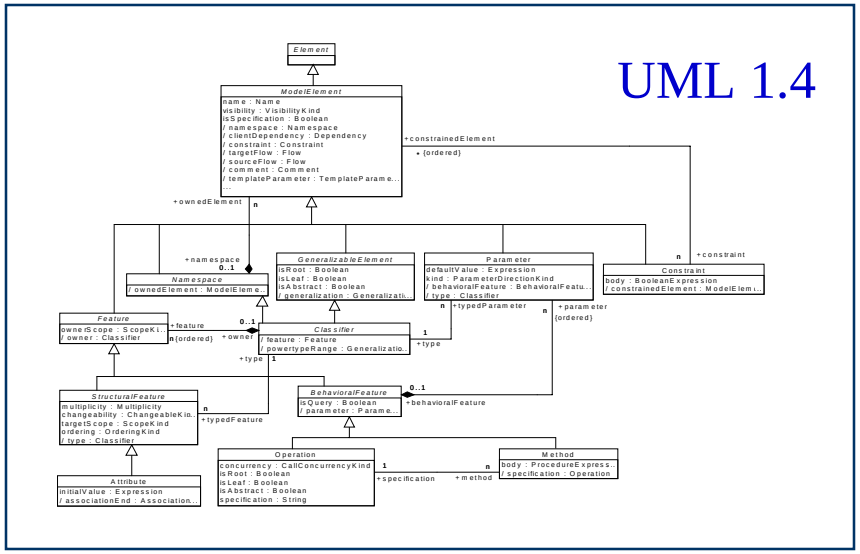


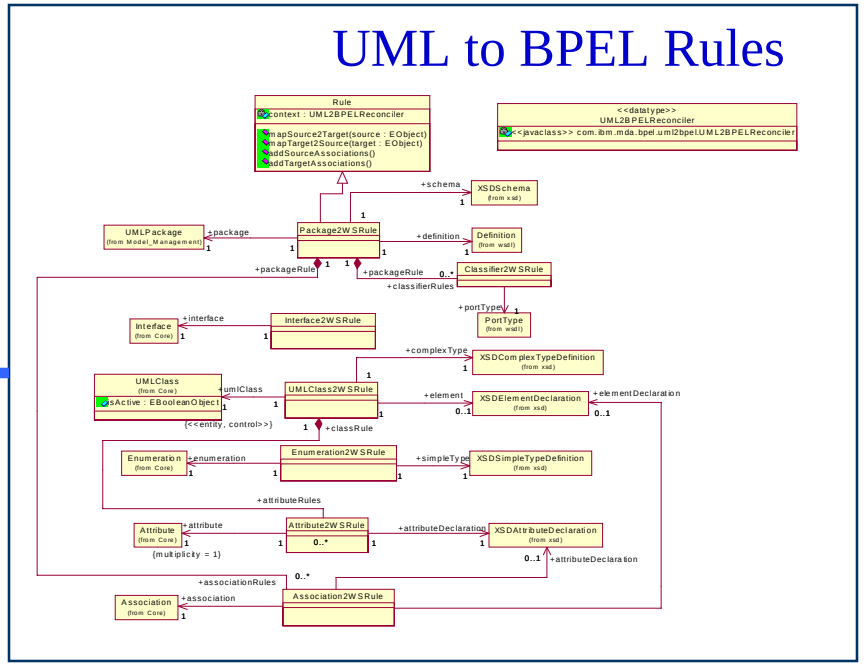

```
<process name="PurchaseOrder" ...>
   ...
   <containers>
      <container name="PO" messageType="PurchaseTypes:PO"/>
      <container name="invoice"
                 messageType="PurchaseTypes:Invoice"/>
      <container name="shippingRequest"
                 messageType="PurchaseTypes:ShippingRequest"/>
      <container name="shippingInfo"
                 messageType="PurchaseTypes:ShippingInfo"/>
      <container name="shippingSchedule"
                 messageType="PurchaseTypes:ShippingSchedule"/>
   </containers>
   ...
</process>
```

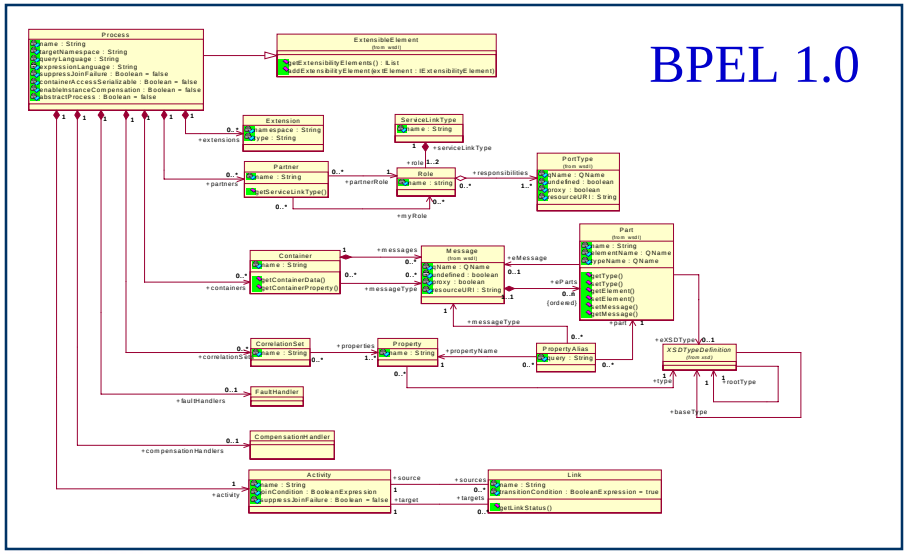

The Eclipse Modeling Framework (EMF)

- EMF provides a Java-based implementation of a cutdown version of MOF
- New metamodels can be defined in Rational Rose and imported into EMF
- EMF supports generation of Java code for creating, manipulating and persisting instances of a meta model
- XMI is the default persistence format, others can be provided (for example, BPEL4WS has its own XML schema)

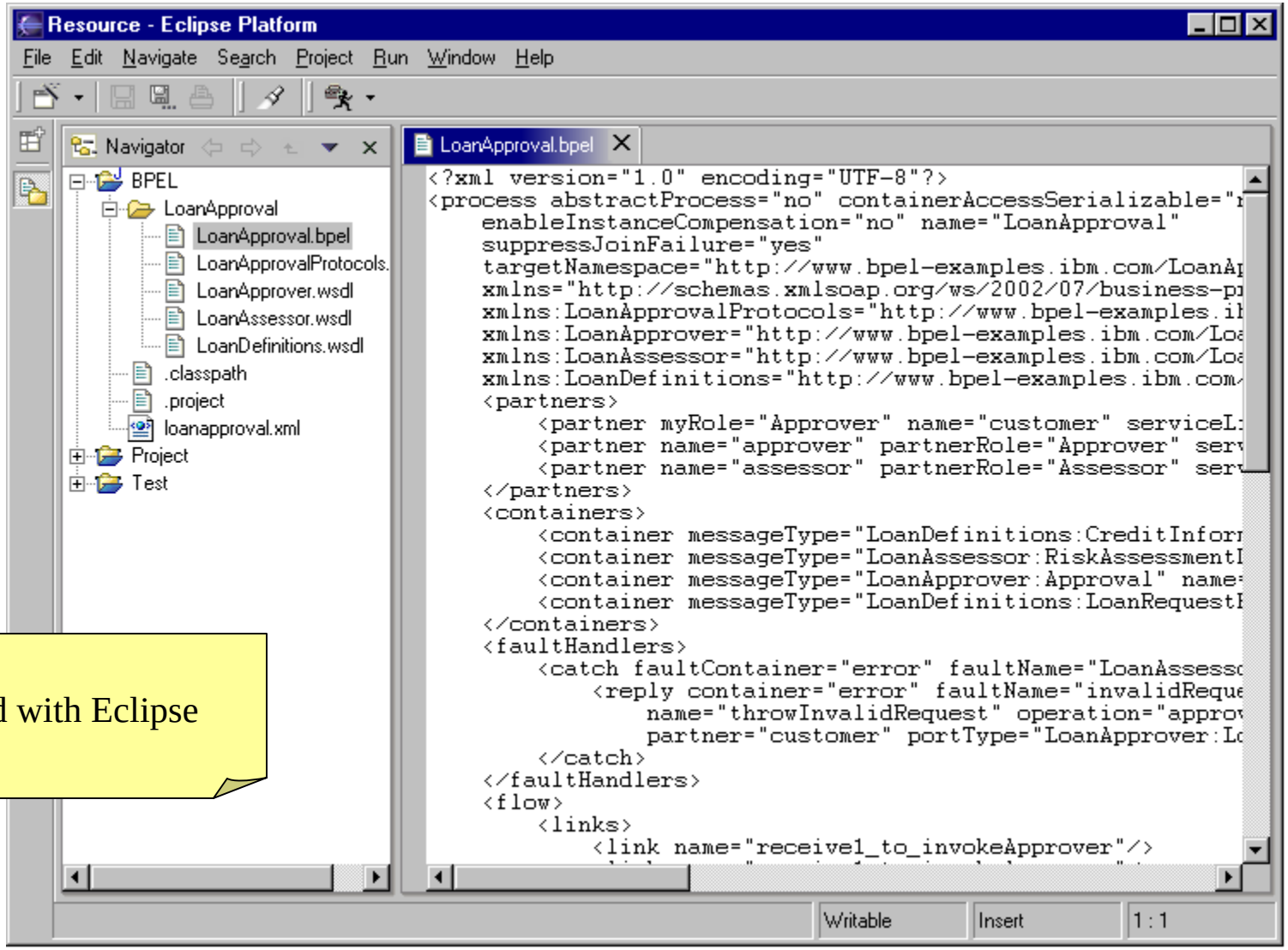
EMF to EMF Model Mapping





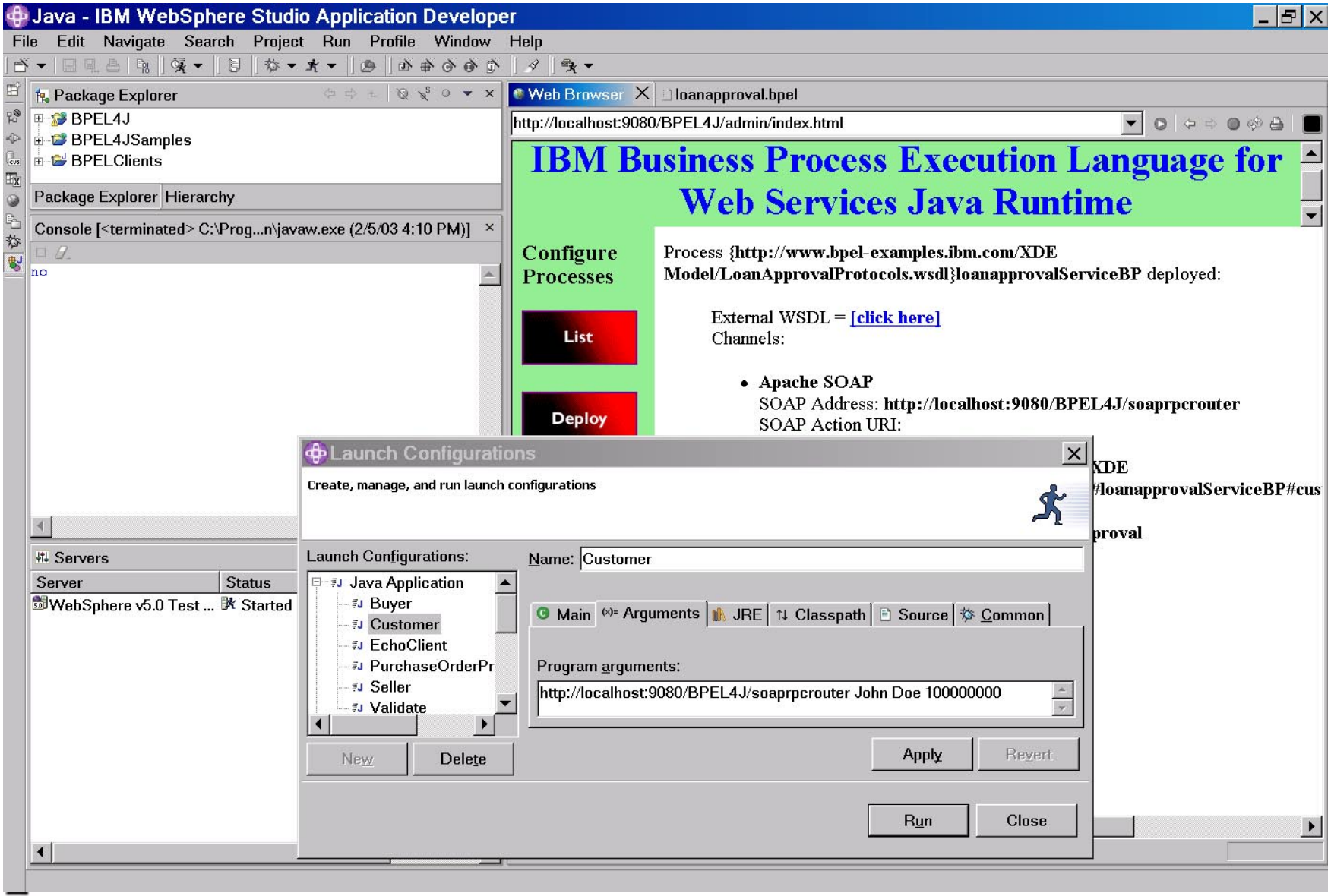


Automated Mapping



Integrated with Eclipse

Executing the process using BPWS4J



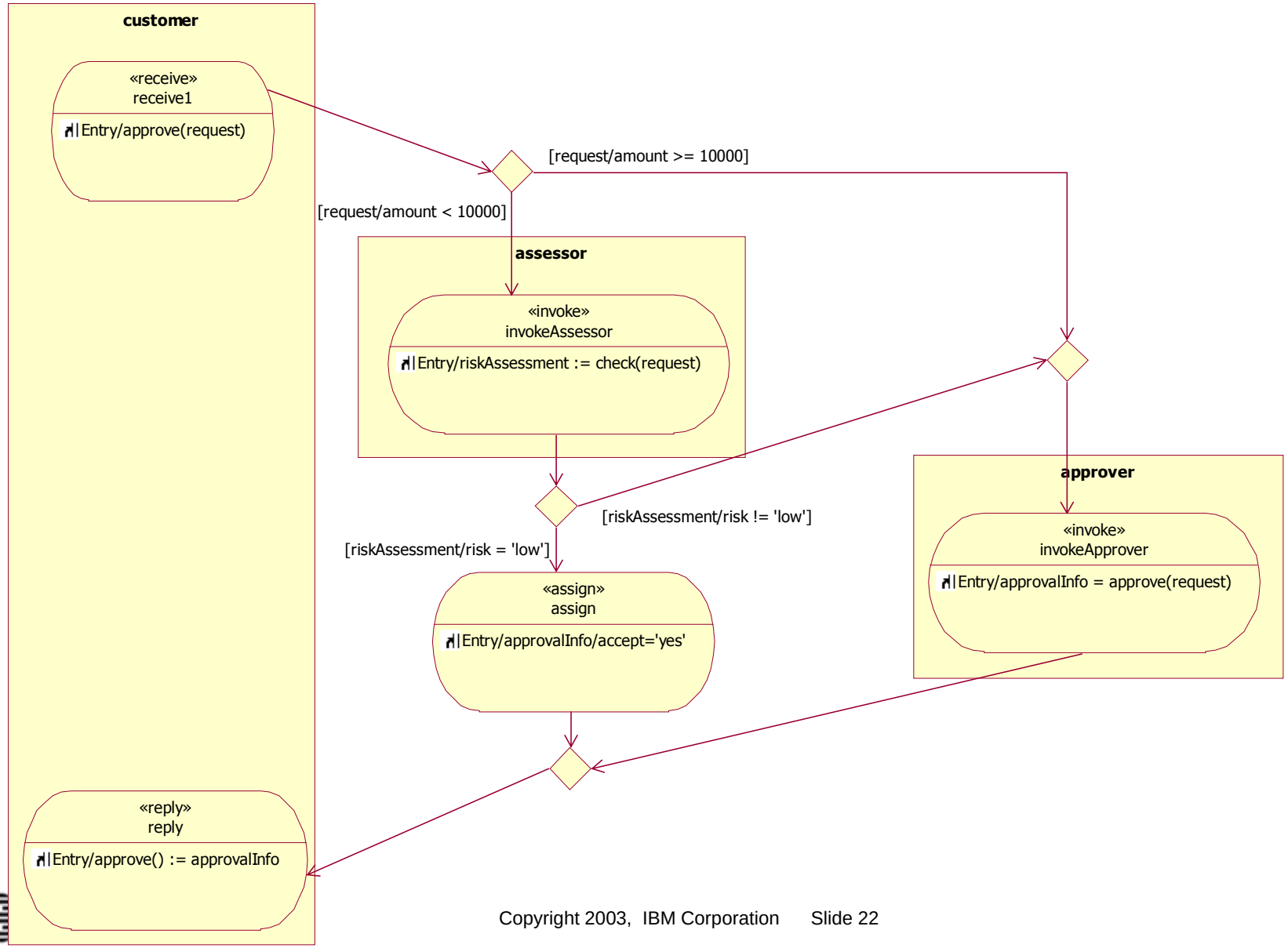
Resources

- EMF:
  http://www.eclipse.org/emf
- BPEL4WS Spec:
  http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/
- BPWS4J on AlphaWorks:
  http://www.alphaworks.ibm.com/tech/bpws4j
- UML to BPEL on AlphaWorks
  http://www.alphaworks.ibm.com/tech/ettk

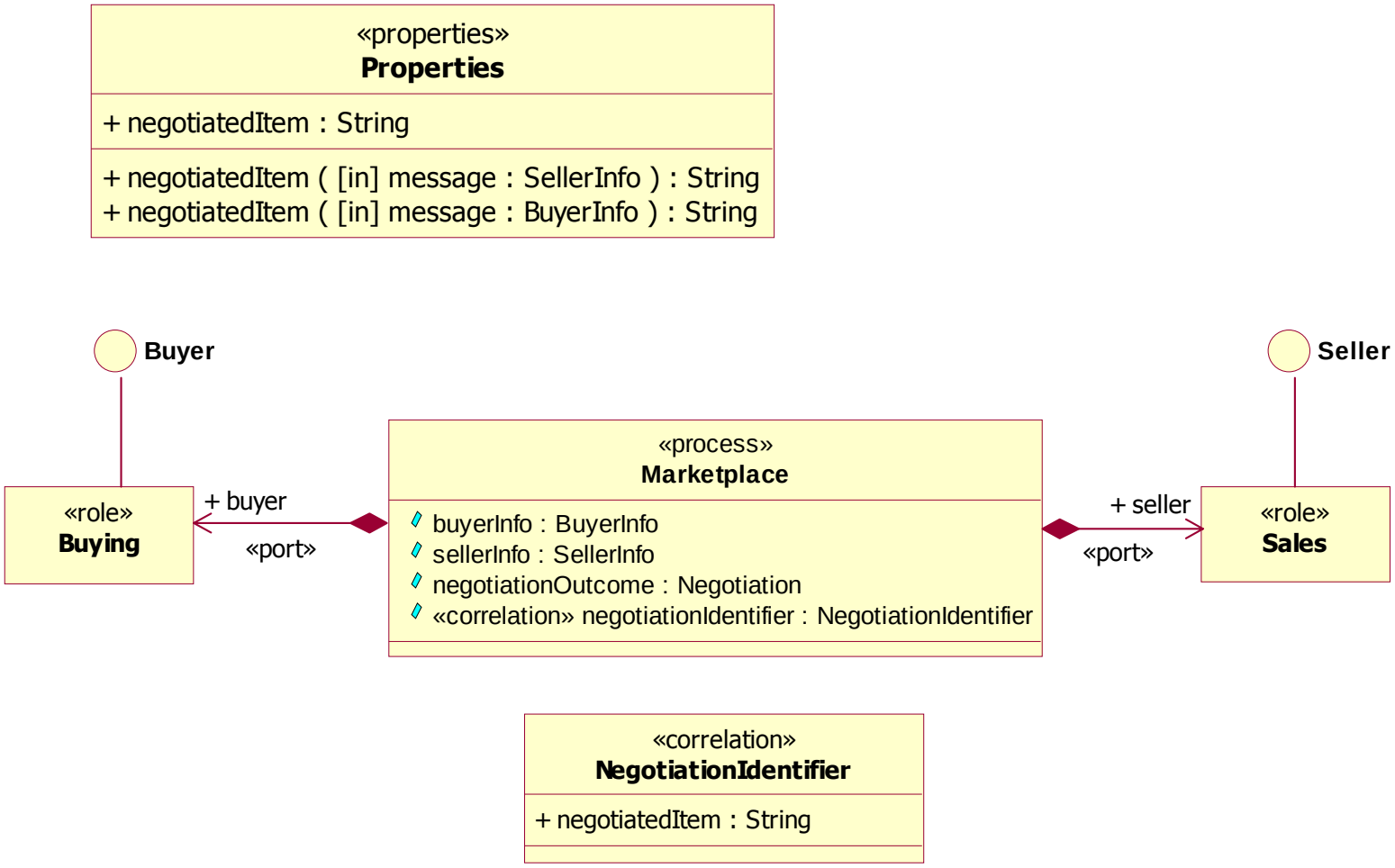
- Contact: Tracy Gardner <tgardner@uk.ibm.com>

Additional Examples

Loan Approval – Conditional Behaviour



Marketplace - Correlation



Correlated Activities