Business Process Integration Using UML and BPEL4WS

SI-SE 2004
University of Zurich
March 18-19, 2004

Sridhar Iyengar
Distinguished Engineer, IBM Rational Software
siyengar@us.ibm.com
Software Development & Integration
A ‘team sport’ that IBM Rational enables

Model, simulate & monitor business processes
Model Applications Documents & data
Visually construct, program, and generate code
Test code

Business Integration

Adopt best practices, track project status, Manage requirements, assets, and quality

Application Monitoring and Deployment

Collaboration & Information Management

TEAM UNIFYING PLATFORM

Analyst
Architect
Developer
Tester
“Components – The Future of …”

‘Components’ – Too many definitions, therefore different types of components
- Java Beans, EJBs, Eclipse plug-ins, CORBA components, Web Services (W3C, OSGi…), Classes (C++, Eiffel, Java, UML1, UML2), Structured Classes (UML2), Component (UML2)

Important to discuss the Architectural framework (J2EE, .Net, Web Services AF, MDA) in which the ‘components’ are defined, deployed, used
- Can’t wire a EJB component and a COM component WITHOUT defining the interoperability protocol (hopefully via Web Services)

Component Assembly (intentional wiring of components to compose coarser grained components) and composition is essential

Industry needs consistent definitions and implementations if the dream of reuse thru component assembly is realized

Work In Progress (Eclipse, J2EE, Web Services, .Net, MDA components)
What this session is & isn’t about

- We will cover
  - Motivation for the UML2BPEL work (i.e why this is important)
  - An overview of recent work on ‘bridging’ the OMG MDA (Model Driven Architecture) work to the SOA and Web Services work at W3C and OASIS
  - Relationship between Components and Web Services
  - Some UML patterns for modeling Web Services & Services Oriented Architecture
  - Progress in OMG on mapping UML 2.0 to BPEL4WS & BPMN
  - The profile is based on BPEL4WS 1.0 (this is being updated to 1.1)

- We will NOT cover
  - Details of UML or UML metamodel (we will show by example)
  - Details of MDA
  - Details of BPEL4WS and rest of the Web Services Stack
  - Or propose YACM (Yet Another Component Model)

- Please hold questions to the end – we have quite a bit to cover!
Business Models, Processes and Rules
Context for ‘Business Process Integration’

Managed Processes
Business Processes
Managed Rules
Business Rules
This Session
Business Transformation
Business Models, Artifacts
Managed Entities
Perf Measures
Business Performance
Solving the Integration Problem

What is needed?

- Short Answer ‘was’ XML and HTTP ala SOAP!
- Slightly longer answer is SOAP, WSDL, UDDI
- Even longer answer is SOAP, WSDL, UDDI, BPEL4WS, XQuery, WS* standards… with J2EE, .Net etc.
- The real answer is: many of the buzzwords above are fine, but
  - We need to think and build software & standards like engineers
  - Where is the process, methodology and architecture, discussion of the full application lifecycle? – A focus for OMG MDA and how it bridges modeling, methods, tools and middleware technologies
  - Somehow we need to simplify this complex mess!

- The OMG Architecture Board, W3C Web Services Architecture WG, OASIS TCs have started to look at this issue from different but complementary perspectives
- Need to form ‘bridges’ between the architectural communities & standards groups that are addressing piece parts of solution
Focus on Architecture is Essential

- Use component-oriented and service-oriented architectures
- Model visually

Business applications

Adapt quickly to changing business needs
Create strategic advantage
Are reliable, scalable and manageable

Design for change
Reduce complexity; work at the right level of abstraction
Ensure architectural integrity and ultimate quality

Focus on architecture to...
Characteristics of Service Oriented Architecture

- Reduced interdependency between software assets.
- Loosely Coupled
- Federated Control
- Standards Based
- Federated and policy based security, management and deployment.
- Leverages open standards to represent software assets as services (XML, SOAP, WSDL, UDDI, UML, RAS...).
- Allows individual software assets (components) to become building blocks that can be reused in developing other applications (application assembly & composition).

Structure, Relationships, Composition, Semantics, Usage Guidelines
Service Oriented Architecture & Component Based Development

- Secure, Reliable, Transacted Web Services
- Using Service Oriented Architecture & Component Based Development to build Web Services Applications
- Modeling XML Applications with UML
- From UML to BPEL
- OMG Business Process Definition Metamodel RFP
- Web Services best practices
An Overview of OMG Model Driven Architecture

An Architectural Style & a set of standards that enables the use Industry Standard Models, Metadata, Mappings (Patterns & Transformations) for building & integrating software.

MDA allows architects & developers to productively design, build, integrate and manage applications throughout the application lifecycle.
MDA: Model Driven Architecture

www.omg.org/mda

- An integration of best practices in Modeling, Middleware, Metadata and Software Architecture
- Model Driven (UML, MOF, CWM…)
  - Platform Independent Business Models (PIM)
  - Platform Specific Models (PSM)
  - Mappings: PIM $\leftrightarrow$ PSM, PSM $\leftrightarrow$ PSM (Relative term!)
- Metadata Driven (MOF, XSD, XMI)
- Key Benefits
  - Improved Productivity for Architects, Designers, Developers and Administrators
  - Lower cost of Application Development and Management
  - Enhanced Portability and Interoperability
  - Business Models and Technologies evolve at own pace on platform(s) of choice

Key Concept: Models & Model Transformations Based on standard Metadata Models
MDA as an ‘Architectural Style’ for Model Driven Integration

- Understand the problem domain (technology or business)
- Model the problem domain
  - Use UML for the visual modeling, analysis & design of meta model
  - Use a UML compliant tool/repository that supports XMI import/export
  - Define Domain Specific Modeling Languages (aka MetaModels) as needed
  - ‘In Progress’: Make UML & MDA more data/information/web services modeler friendly, provide Business/IT visualization – not just ‘OO/CBD’
- Formally represent the metamodel semantics using MOF
  - Simple class modeling is all you need to know
  - OCL (Object Constraint Language) can capture additional semantics
  - Reverse engineer existing DTD, XSD, XMI, Java to MOF (jump start)
- Use Standard transformation (mappings) patterns for
  - Metadata Interchange (XMI – MOF to XML, DTD, XSD)
  - Metadata Interfaces (JMI – MOF to Java, MOF to WSDL*, MOF to IDL etc.)
- Use open source meta modeling frameworks for metadata management
  - Eclipse EMF: www.eclipse.org/emf
  - Netbeans MDR: www.netbeans.org

Summary: Understand, Model, Map and Manage metadata to integrate
Model Transformations using OMG Model Driven Architecture (MDA)

- Mappings PIM - PSM
- XMI, XSD, DTD
- MOF, JMI
- JOLAP, JDM, UML4EJB...

Business

Computation Independent Business Model

Auto
Color : String
Door : Integer
Engine : Integer

Engine
Cyl : integer
Fuel : String
Comp : Float

Computation Independent Model View

Mappings to UDDI, WSDL & BPEL4WS being defined

IT

Platform Specific Design and Implementation Model

Java, C# (PSM)
Class Auto
{public String color;
public int Door;
public int Engine}

XML Doc. (PSM)
<Auto>
<Color> Red </Color>
<Door> 4 </Door>
<Engine> 1 </Engine>
</Auto>

XML DTD, Schema (PSM)
<!Element Auto
(Color*,
Door*,
Engine*)>
Choice of Modeling Paradigms

Adopt the right modeling paradigm for your needs

UML & MDA Standards
Supports multiple:
- Development languages
- Operating environments
- Skill levels
- Modeling paradigms

Key MDA Standards
- UML
- XMI
- MOF

www.omg.org
www.eclipse.org/emf
www.eclipse.org/uml2
www.eclipse.org/xsd

Model-Driven Development

Abstraction/productivity

Scope

Executable Models

Model Transformations

Pattern-Based Development

Roundtrip Engineering

Code Visualization

Code only

Code
Web Services Overview

Web services enables application/business process/people interaction

Business Process
Management
User Interaction

Simple Object Access Protocol (SOAP) - Connect the service

Reliable messaging

Quality of Service

Customers, business partners, suppliers, employees

Web Services Description Language (WSDL) - Describe what the service is and how to use it

Universal Description, Discovery, and Integration (UDDI) - Find the service

Copyright 2004, IBM Corporation
What are the hot standardization areas today in web services?

- **Security**
  Program well underway in OASIS, uncontroversial. WS-Federation is well received.

- **Reliability**

- **Business Process Choreography and Transactions**
  Specs published August, 2002, convergence of work from IBM and Microsoft, now in OASIS.

- **Management**
  Work is early stage, but good coordination among OASIS, W3C, GGF, and DMTF.

- **Interoperability**
  WS-I.org is the place to be and the work to watch.

- **User Experience**
  WSRP is an OASIS TC specification with many vendors announcing product.
Web Services: The ‘Standards Stack’ Evolves in W3C, OASIS and WS-I

- Specific Composable Services and Processes (ebXML, OAG, RosettaNet PIP...)
- Service Composition & Flow (BPEL4WS..)
- Security
- Reliable Messaging
- Transactions
- XSD, WSDL, UDDI, WS-Policy
- XML, SOAP, WS-Addressing
- Network (HTTP, HTTPS, SMTP…)

Not intended to be viewed as a ‘software stack’
UML can be used to model Web Services
- Web Service Operations (WSDL Operations) as UML Operations
- Groups of Web Service structures Requests (WSDL PortTypes) as UML Interfaces or Classes
- XML Schemas using UML Class Diagrams
- Web Services data structures using UML Class Diagrams
- Web Services flow (E.g.:BPEL4WS) using UML Activity Diagrams
- UML2 Structured Classes & Components add additional support for Architectural Modeling

MOF and XMI can be used to define standard transformations from models to XML schemas, DTDs, Documents
- www.eclipse.org/emf, xsd, uml

But standard metamodels, transformations, UML profiles etc, would be useful
- http://www.omg.org (MOF, XMI, UML and CWM specs)
- XMI - Mappings from MOF/UML to XML DTD, XML Schema

Using Service Oriented Architecture & Component Based Development to build Web Services Applications
- Modeling XML applications with UML by David Carlson
  • http://www.xmlmodeling.com

UML to BPEL4WS mappings

From UML to BPEL
A peek at integrating Business Processes using UML and BPEL4WS

See the BPEL4WS spec for the examples, BPEL4WS syntax
See OMG UML spec for details on UML
Read many popular books, use tools

Possible standard UML profile for BPEL

BPEL4WS tools still emerging

Update: BPD Profile submitted to OMG
Mapping to BPEL4WS 1.1 in progress
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.

A short summary of research underway at IBM is described to illustrate how MDA and models can be used for business process integration.

BPEL4WS products are beginning to show up in the market – more in 2004 (tools and engines)

Submitted to OASIS by IBM, Microsoft, BEA, SAP & Siebel
Web Services, Components & Services Orchestration

- Components are implemented using J2EE, .Net, CICS transactions...

- Components support defined extension points (think Eclipse Plug-ins, UML2 Components and Ports)
  - Provided Interfaces (Services offered to other components)
  - Required Interfaces (Services required by this component)

- Components can be ‘wired’ together explicitly to assembly ‘coarser grained’ components
  - ‘Connectors’ in UML2 (influenced by ACME, UML4RT…)
  - ‘Service references’ in BPEL that allows Integration partners to refer to each other (‘soft references’)

- Components can be orchestrated at runtime (role of BPEL)
**Unified Modeling Language (UML)**

- Visual Modeling Language for analysis, design and construction of software
  - Widely used in the software architecture and ‘lead developer’ community for component based development
  - UML ‘Profiles’ are used to customize UML to a specific domain
- UML 1.0 became a standard in 1997 and unified several leading OO modeling notations (Booch, Rumbaugh, Jacobson, Mellor….) – UML 2.0 was just adopted
- The UML metamodel (model of UML) is defined using a meta language called Meta Object Facility & using UML notation
- UML models are serialized as XML documents based on UML 1.4 DTD or UML 2.0 XSD
  - XML Metadata Interchange (XMI) provides design patterns that automate model and metadata serialization
- UML, MOF, XMI are OMG MDA standards
Why map UML to BPEL4WS?

UML to BPEL4WS

- UML is a widely used, standard modelling language for software design with a visual notation.

- 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

  A profile is a customization of UML for modelling in a particular context.
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.
**Purchase Order (PO) Process**

- Customer sends purchase order to purchase service provided by PO process
- PO process asks for initial price, shipping information, and production schedule
- Shipping requestor determines where order items will be obtained and creates source and destination shipping information
- When price and shipping information are available, invoice provider calculates final price and sends invoice to PO process
- The PO process asks a scheduling provider to determine when each order item will be produced and instructs the scheduling provider to send a schedule to the customer
- Finally, the PO process replies to the customer with an invoice
The PO Process in BPEL4WS

A BPEL ‘program’

state

interaction points

behaviour

```
<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>
```
This is the UML 2 version, we approximate this in UML 1.4 (more detail later)
BPEL Partners as Ports – UML 1.4

PurchaseOrderProcess

«role» Purchase::PurchaseService
- customer «port»

«process» PurchaseOrder
- PO : PO
- shippingInfo : ShippingInfo
- shippingRequest : ShippingRequest
- shippingSchedule : ShippingSchedule
- invoice : Invoice

«port» - invoiceProvider
«port» - shippingProvider
«port» - schedulingProvider

«role» Invoice::InvoiceRequester
«role» Shipping::ShippingRequester
«role» Scheduling::SchedulingRequester
UML2 Activities: New Semantic Foundation

- Petri Net-like foundation (vs. statecharts) enables
  - Un-structured graphs (graphs with “go-to’s”)
  - Richer models of concurrency

**Pre- and post-conditions**

- ProcessOrder
  - RequestedOrder:Order
  - <<precondition>> Order complete
  - <<postcondition>> Order entered

Input pin

Parameter

Pre- and post-conditions

Partition

London (Order Processing)

Toronto (Payment Processing)
**Purchase Order Process**

- **customer**
  - «receive» receivePO
  - «reply» returnInvoice

- **invoiceProvider**
  - «invoke» initiatePriceCalculation
  - «invoke» sendShippingPrice
  - «receive» receiveInvoice

- **shippingProvider**
  - «invoke» requestShipping
  - «receive» receiveSchedule

- **schedulingProvider**
  - «invoke» requestScheduling
  - «invoke» sendShippingSchedule

- **partition**
  - «assign» initialiseShippingRequest

**Partitions correspond to ports (partners)**
Data and Message Types

- «external» SNS
  - «data» Invoice
  - «data» Schedule
  - «data» ShippingInfo
  - «data» OrderFaultType
  - «data» CustomerInfo
  - «data» PurchaseOrder
  - «data» ShippingRequest

+ customerInfo
+ purchaseOrder

Purchase

«messageContent» PO
**Interfaces**

- **interface** `PurchaseOrder`
  - `sendPurchaseOrder ([in] po : PO) : Invoice`

- **interface** `InvoiceCallback`
  - `sendInvoice ([in] invoice : Invoice)`

- **interface** `ShippingCallback`
  - `sendSchedule ([in] schedule : Schedule)`

- **interface** `ComputePrice`
  - `initiatePriceCalculation ([in] po : PO)`
  - `sendShippingPrice ([in] shippingInfo : ShippingInfo)`

- **interface** `Shipping`
  - `requestShipping ([in] shippingRequest : ShippingRequest) : ShippingInfo`

- **interface** `Scheduling`
  - `requestProductionScheduling ([in] shippingInfo : ShippingInfo)`
  - `sendShippingSchedule ([in] schedule : Schedule)`
Protocols

Protocols are defined independently of the processes that use them.

Roles provide ‘port types’ – groupings of provided and required interfaces that must be supported.
Basic Interactions

- Action syntax based on XPATH which BPEL uses as its default expression language

  - «receive»
    - receivePO
    - Entry/sendPurchaseOrder(PO)
  
  - «reply»
    - returnInvoice
    - Entry/sendPurchaseOrder() := invoice

  - «invoke»
    - requestShipping
      - Entry/shippingInfo := requestShipping(shippingRequest)

  - «invoke»
    - requestScheduling
      - Entry/requestProductionScheduling(PO)

  A receive with a corresponding reply handles a synchronous request

  Synchronous requests have a response
<process
   abstractProcess="no"
   containerAccessSerializable="no"
   enableInstanceCompensation="no"
   name="LoanApproval"
   suppressJoinFailure="yes"
   targetNamespace="http://www.bpel-examples.ibm.com/LoanApproval/LoanApprovalProcess/LoanApproval.bpel">
   ...
   <containers>
      <container messageType="LoanDefinitions:CreditInformation"
               name="request"/>
      <container messageType="LoanAssessor:RiskAssessmentDetails"
               name="riskAssessment"/>
      <container messageType="LoanApprover:Approval"
               name="approvalInfo"/>
      <container messageType="LoanDefinitions:LoanRequestError"
               name="error"/>
   </containers>
   ...
</process>
Mapping: Partners

```xml
<partners>
  <partner myRole="Approver" name="customer"
    serviceLinkType="LoanApprovalProtocols:LoanApproval"/>
  <partner name="approver" partnerRole="Approver"
    serviceLinkType="LoanApprovalProtocols:LoanApproval"/>
  <partner name="assessor" partnerRole="Assessor"
    serviceLinkType="LoanApprovalProtocols:RiskAssessment"/>
</partners>
```
Mapping: Assign Activities

```xml
<assign name="assign">
  <copy>
    <from expression="'yes'"/>
    <to container="approvalInfo" part="accept"/>
  </copy>
</assign>
```
## BPEL Mapping Overview

<table>
<thead>
<tr>
<th>&lt;&lt;process&gt;&gt; class</th>
<th>BPEL process definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity graph on a &lt;&lt;process&gt;&gt; class</td>
<td>BPEL activity hierarchy</td>
</tr>
<tr>
<td>&lt;&lt;port&gt;&gt; associations</td>
<td>BPEL partner declarations</td>
</tr>
<tr>
<td>&lt;&lt;process&gt;&gt; class attributes</td>
<td>BPEL containers</td>
</tr>
<tr>
<td>Hierarchical structure and control flow</td>
<td>BPEL sequence and flow activities</td>
</tr>
<tr>
<td>Decision nodes</td>
<td>BPEL switch activities and transition conditions</td>
</tr>
<tr>
<td>&lt;&lt;receive&gt;&gt;, &lt;&lt;reply&gt;&gt;, &lt;&lt;invoke&gt;&gt; activities</td>
<td>BPEL receive, reply, invoke activities</td>
</tr>
<tr>
<td>&lt;&lt;protocol&gt;&gt; package with &lt;&lt;role&gt;&gt; classes</td>
<td>BPEL service links types and roles</td>
</tr>
</tbody>
</table>
UML to BPEL alphaWorks Demo

- Part of the Emerging Technologies Toolkit
- Available now
- Supports an end-to-end scenario: Loan Approval
- Version supporting full forward generation from user-defined models coming in July 2003
UML2BPEL Components – All these are EMF based Eclipse plug-ins!

- Eclipse infrastructure: [www.eclipse.org](http://www.eclipse.org)
- EMF component: [www.eclipse.org/emf](http://www.eclipse.org/emf)
- XSD component: [www.eclipse.org/xsd](http://www.eclipse.org/xsd)
- BPEL component: Internal IBM metamodel (for tools) and BPEL engine (BPWS4J)
- Will use UML2 component: [www.eclipse.org/uml2](http://www.eclipse.org/uml2)
  - The proof of concept of EMF UML 1.4 implementation was a precursor to the current UML2 project
- WSDL component: Internal IBM metamodel
- UML2BPEL mapping component: Internal IBM metamodel component (mapping component)
UML2BPEL has been Submitted to OMG (IBM, Unisys, BEA...)

- **Notation(s)**: UML 2.0, BPMN, Other...
- **Metamodel**: Business Process Definition Metamodell
  - Information
  - Process
  - Organization
- **Runtime**: J2EE, BPEL4WS, Other...

Inclusion of domain-specific languages.
Modeling for Web Services - Recap

- UML, MOF and XMI can be used as-is to model & manage Web Services
  - Web Service Operations (WSDL Operations) are UML Operations
  - Groups of Web Service structures Requests (WSDL Ports) as UML Interfaces or Classes
  - XML Schemas using UML Class Diagrams
  - Web Services data structures using UML Class Diagrams
  - Web Services flow (E.g.:BPEL4WS) using UML Activity Diagrams

- But some standard transformations, UML profiles etc, would be useful to enable tools interoperability
  - [http://www.omg.org](http://www.omg.org) (MOF, XMI, UML and CWM specs)
  - XMI - Mappings from MOF/UML to XML DTD, XML Schema
  - Modeling XML applications with UML by David Carlson
    - [http://www.xmlmodeling.com](http://www.xmlmodeling.com)
  - UML to BPEL mappings (UML profile for BPEL) - Proposed standard
  - OMG Business Process Definition Metamodel RFP
  - OMG Business Rules RFP Drafts : Rule Expression Language, Rule Management…
Conclusions

- The experiences of modelling with UML can be applied to the development of systems that will be deployed using emerging web services standards.

- It is possible to specify a profile of UML with sufficient detail that it can be translated automatically to a language such as BPEL4WS.

- The approach provides an integration mechanism for multiple standards and specifications which need to be used to build a complex solution. This is especially relevant in business integration scenarios.

IBM will integrate its tools portfolio using a set of shared Metamodels and Profiles based on the Eclipse platform (as plug-ins). The UML 2.0, XSD, WSDL, BPEL4WS and related transformations are an integral part of our software development platform.
Wrap Up

- Most customers have and will continue to have components and information/data from multiple sources and formats that need integration.

- Web Services promise to be a key enabler for application integration and business process integration:
  - Embraced by all industry players
  - Pragmatic loosely coupled integration
  - Promise of ubiquitous availability
  - Modeling and Web Services on a convergence path
    - OMG MDA, OASIS ebXML UMM, BPMN

- Model Driven Architecture enables architects, designers and developers to use models and metadata with web services and existing technologies with a focus on full life cycle architecture.

- Web Services standards (like other standards!) will come and go, but some of the business models and processes will be more lasting (manage and protect your models and metadata).

- Use models to communicate, understand, analyze and design, use metadata to integrate flexibly within an architectural context.
For More Information

- OMG : [www.omg.org](http://www.omg.org)
- MDA : [www.omg.org/arma](http://www.omg.org/arma)
- IBM : [www.ibm.com](http://www.ibm.com)
- UML : [www.omg.org/uml](http://www.omg.org/uml), [www.rational.com](http://www.rational.com)
- CWM : [www.cwmforum.org](http://www.cwmforum.org), [www.omg.org/uml](http://www.omg.org/uml)
- W3C : [www.w3c.org](http://www.w3c.org)
- DSTC : [www.dstc.edu.au](http://www.dstc.edu.au)
- Sridhar : siyengar@us.ibm.com
- UML RTF : ulm-rtf@omg.org
- MOF RTF : mof-rtf@omg.org
- XMI RTF : xmi-rtf@omg.org
- CWM RTF : cwm-rtf@omg.org
- Eclipse : [http://www.eclipse.org](http://www.eclipse.org) (/emf, /xsd, /uml2)
Backup
Tool Integration across Modeling & Web Service Tools – How are we doing it?

- Use Eclipse tools integration framework
  - www.eclipse.org (Java IDE, tools integration framework)

- Use Eclipse EMF for modeling & metadata management
  - Simplification of OMG MOF 1.4 with support for XMI 2.0
  - Use UML models, XMI, XSD or Annotated Java as metamodel input
  - Code generation for metadata interfaces and implementation
  - XML Serialization, XSD generation. Support for MOF2 soon

- Used in IBM WebSphere Studio & in WebSphere Application Server (ex: configuration metadata), Rational, Borland..

- Expect the usage to increase across software development lifecycle
  - UML modeling, testing, software quality improvement…
  - UML2 open source project launched: www.eclipse.org/uml2
  - IBM Research area: ‘Model Driven Business Integration’
OMG Model Driven Architecture
EMF in WebSphere Tools

UML
Model & Design (PIM)
UML, Java, XSD Input

XMI
MOF2XML (PSM)
DTD, Document XSD (XMI 2.0)

EMF
Framework (PIM)

EMF2Java
(PSM)

SQL*

Etc.
Application Server Configuration
Model Driven Tool Integration
Model Driven App Development
MetaData Management
XML Schema Management

PIM : Platform Independent Model
PSM : Platform Specific Model

* In progress
Sample EMF Metamodels
Used in WebSphere
Integration of key software domains

XMI  XML  HTML  JSP  EJB  Java  C/C++  COBOL  SQL  BPM*
MOF  XSD  BPEL*  WSDL  UDDI  FCM  Mapping  WCCM  UML*

End-to-End Systems Management

Responsive  Dynamic & Flexible  Scalable & Robust

* In R&D
EMF::Ecore Details

Influenced MOF2::EMOF
Using MDA to execute the Mapping

- Start with EMF models of UML, XML Schema, WSDL, and BPEL
- Use EMF to generate Java APIs for all the EMF models
  - Each model uses a serializer for its preferred external representation
- Create an EMF model of the mapping between UML and the Web Services components
- Use EMF to generate an API for the model-to-model mapping
- Implement the methods to map between source and target models
- Can be deployed as an Eclipse builder, no user input required to do the mappings
XML Schema model
From www.eclipse.org/xsd
WSDL Model – From WSAD

```
<interface>
  <definition>
    <addBinding/>
    <addImport/>
    <addMessage/>
    <addNamespace/>
    <addPortType/>
    <addService/>
    <createBindingFault/>
    <createBindingInput/>
    <createBindingOutput/>
    <createBindingOperation/>
    <createBinding/>
    <createFault/>
    <createImport/>
    <createInput/>
    <createMessage/>
    <createOperation/>
    <createOutput/>
    <createPart/>
    <createPort/>
    <createPortType/>
    <createService/>
    <getBinding/>
    <getBindings/>
    <getImports/>
    <getMessages/>
    <getNamespaces/>
    <getPortType/>
    <getPortTypes/>
    <getPrefix/>
    <getService/>
    <getServices/>
    <getExtensionRegistry/>
    <getExtensionRegistry/>
    <getDocumentBaseURI/>
    <getDocumentBaseURI/>
    <createTypes/>
    <removeService/>
    <removeBinding/>
    <removePortType/>
    <removeMessage/>
    <getTypes/>
    <getTypes/>

  </definition>
</interface>
```
**UML2BPEL Components – All these are EMF based Eclipse plug-ins!**

- Eclipse infrastructure: [www.eclipse.org](http://www.eclipse.org)
- EMF component: [www.eclipse.org/emf](http://www.eclipse.org/emf)
- XSD component: [www.eclipse.org/xsd](http://www.eclipse.org/xsd)
- BPEL component: Internal IBM metamodel (for tools) and BPEL engine (BPWS4J)
- Will use UML2 component: [www.eclipse.org/uml2](http://www.eclipse.org/uml2)
  - The proof of concept of EMF UML 1.4 implementation was a precursor to the current UML2 project
- WSDL component: Internal IBM metamodel
- UML2BPEL mapping component: Internal IBM metamodel component (mapping component)