Implementing Model Driven Web Services Architecture using UML, XML, WSDL & BPEL4WS

Web Services for the Integrated Enterprise
Munich, Germany
February 10-13, 2003

Sridhar Iyengar
Distinguished Engineer, IBM
Member, OMG Architecture Board
siyengar@us.ibm.com
Presentation Overview

- Introduction - The integration problem
- OMG MDA - Model Driven Architecture Overview
  - What is MDA and why is it important
  - Emerging MDA standards for Model Driven Business Integration
- Web Services Architecture Overview
  - What are Web Services and why are they important
  - Emerging Web Services Standards and Architecture
- MDA and Web Services – How do you use them together
  - Modeling Web Services
  - Advanced Modeling (Web Services Orchestration)
  - Model Driven Business Integration – A peek ahead
- Wrap-up
Acknowledgements

- OMG - For use and reference to many OMG specs and trademarks including MDA, UML, XMI, MOF and CWM
- W3C - For use and reference to many W3C specs including XML
- OASIS – For use and reference to UDDI which is now being progressed thru OASIS
- IBM - For its continued pioneering of open standards in middleware, data management, web services and metadata
- SUN - For use and reference to many Java standards including J2EE, Java and JMI
‘Muddleware’ Architect’s Dilemma

Web Services!

Distributed, Heterogeneous, Client/Server!

Multiple Clients, Servers, Tools, Apps, O/S, File systems, Databases, Repositories, Data Models, Object Models...
## The Middleware Salad Bar

<table>
<thead>
<tr>
<th>Capability</th>
<th>J2EE</th>
<th>COM+</th>
<th>CORBA/OMA</th>
<th>Web Services</th>
<th>OMG MDA</th>
<th>.Net</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Layer</td>
<td>TCP/IP</td>
<td>TCP/IP</td>
<td>TCP/IP</td>
<td>TCP/IP</td>
<td>TCP/IP</td>
<td>TCP/IP</td>
</tr>
<tr>
<td>Web Protocol</td>
<td>HTTP</td>
<td>HTTP</td>
<td>HTTP</td>
<td>HTTP</td>
<td>HTTP</td>
<td>HTTP</td>
</tr>
<tr>
<td>Interface Definition</td>
<td>Java</td>
<td>Microsoft IDL</td>
<td>CORBA IDL</td>
<td>WSDL</td>
<td>IDL/WS...</td>
<td>WSDL/C#</td>
</tr>
<tr>
<td>Meta Language</td>
<td>XML</td>
<td>XML</td>
<td>MOF/XML</td>
<td>XML</td>
<td>MOF/XML</td>
<td>XML</td>
</tr>
<tr>
<td>RPC Mechanism</td>
<td>RMI/IIOP</td>
<td>DCOM</td>
<td>IIOP</td>
<td>SOAP; XML</td>
<td>SOAP; IIOP</td>
<td>SOAP</td>
</tr>
<tr>
<td>Registry/Repository</td>
<td>JNDI; LDAP</td>
<td>LDAP; ADSI</td>
<td>CORBA IR</td>
<td>UDDI</td>
<td>MOF; UDDI</td>
<td>UDDI</td>
</tr>
<tr>
<td>Process Flow</td>
<td>JPC</td>
<td>Proprietary</td>
<td>Proprietary</td>
<td>BPEL4WS..</td>
<td>UML</td>
<td>BPEL4WS</td>
</tr>
<tr>
<td>Modeling Language</td>
<td>UML</td>
<td>UML</td>
<td>UML</td>
<td>UML, XSD?</td>
<td>UML, MOF</td>
<td>UML</td>
</tr>
</tbody>
</table>

Tiny fragment of the landscape
Solving the Integration Problem
What is needed?

- Short Answer is XML and HTTP ala SOAP!
- Slightly longer answer is SOAP, WSDL, UDDI
- Even longer answer is SOAP, WSDL, UDDI, BPEL4WS, ebXML…
- The real answer is: All the buzzwords above are fine, but
  - We need to think and build software like engineers
  - Where is the process, methodology and architecture, discussion of the full lifecycle? – A focus for OMG MDA and how it bridges modeling, methods and middleware

- The OMG Architecture Board, OMG Web Services SIG, W3C Web Services Architecture WG have started to look at this issue
- First let us recap ‘Web Services’!
  - And peek at some recent advances (ex: BPEL4WS)
The Enterprise Application Life Cycle

Architecture Centric, Business driven, Complex Life Cycle, Many Tools

Discovery and Transformation

Modeling Architecture, Object, Biz...

Add Business Logic

Build /Wrap Components

Assemble & Test Components

Configure & Deploy Components

Manage Component

Component Runtimes (J2EE, COM+)

Acquired Components

Models, Metadata, Mappings, Middleware

Rigorous

Rapid

And do this with quality in a distributed environment
Presentation Overview

- Introduction - The integration problem
- OMG MDA - Model Driven Architecture Overview
  - What is MDA and why is it important
  - Emerging MDA standards for Model Driven Business Integration
- Web Services Architecture Overview
  - What are Web Services and why are they important
  - Emerging Web Services Standards and Architecture
- MDA and Web Services – How do you use them together
  - Modeling Web Services
  - Advanced Modeling (Web Services Orchestration)
  - Model Driven Business Integration – A peek ahead
- Wrap-up
An Overview of OMG Model Driven Architecture

An Architectural Style that recommends the use of Industry Standard Models, Metadata, Mappings (Patterns & Transformations) for integrating software.

MDA allows developers and users to productively design, build, integrate and manage applications throughout the lifecycle while separating technology & business concerns.

Sridhar’s Usage
OMG History

- 700+ Vendors and End User members (www.omg.org)
- 1989 to 1997: OMA and CORBA gain prominence
  - 1989 OMA Vision & Architecture
  - 1991 CORBA 1.0
  - 1995 CORBA 2.0 IIOP - CORBA Interoperability
- 1995 – 2001: The foundation for MDA is established
  - 1995 UML and MOF work begins, Java arrives on the scene
  - 1997 UML and MOF adopted, Domain specs begin to be adopted
  - 1998 XML arrives on the scene, Java and XML gain momentum
  - 1999 XMI (integration of MOF, UML and XML) adopted
  - 2000 CWM, XMI for XML Schema work begins
  - 2001 UML for EDOC, EAI, UML 2.0 work begins
- 2001 OMG unveils Model Driven Architecture - MDA
  - 2002 XMI 2 adopted, UML2 and MOF 2 proposals arrive
  - 2003 UML2 and MOF2 standards to be adopted
  - 2004 Business Rules and Business Modeling standards expected
The use of **Modeling** and **Metadata** enabled architectures allows us to manage the complexity of software development, application integration and data warehouse management.

**Modeling and Metadata standards** are necessary for interchange of software artifacts and interoperability between tools, applications, middleware and data stores across platforms.

Largely influential in OMG evolution from OMA to MDA.

Major efforts underway:
- UML2
- MOF2
- Action Semantics
Solution for Managing Complexity: MDA: Model Driven Architecture

- An integration of best practices in Modeling, Middleware, Metadata, Internet and Software Architecture

- Model Driven (UML, MOF, CWM…)
  - Platform Independent Models (PIM)
  - Platform Specific Models (PSM)
  - Mappings: PIM <=> PSM,
  - Applies across the software life cycle

- 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

* PIM usually applies to a class of platforms/protocols (ex: Messaging)
The Enterprise Application Life Cycle

Architecture Centric, Business driven,
Complex Life Cycle, Many Tools

Discovery and Transformation

Modeling Architecture, Object, Business...

Add Business Logic

Build /Wrap Components

Assemble & Test Components

Configure & Deploy Components

Component Runtimes (EJB, COM+)

Manage Components

Models, Metadata, Mappings, Middleware

Rigorous

Rapid

Source Meta Model (PIM, PSM)

Model Transformation

Target Meta Model (PIM, PSM)

Acquired Components

Copyright 2003, IBM Corporation
**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
  - Opportunity: Make UML more data/information modeler friendly, provide E/R visualization
- 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 IDL etc.)
- Use open source metamodeling frameworks for metadata management
  - Eclipse EMF: [www.eclipse.org](http://www.eclipse.org)
  - Netbeans MDR: [www.netbeans.org](http://www.netbeans.org)
- **Summary**: Understand, Model, Map and Manage metadata to integrate
Platform Independence using OMG Model Driven Architecture (MDA)

Platform Independent Models (PIM)
- Computation Independent Business Model
- Auto
  - Color : String
  - Door : Integer
  - Engine : Integer
- Engine
  - Cyl : integer
  - Fuel : String
  - Comp : Float
- Computation Independent Model View

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

Platform Specific Models (PSM)
- 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*)>

Mappings to UDDI and WSDL being defined

Copyright 2003, IBM Corporation
**OMG MDA - Specific Models/Work in Progress**

- Community & Enterprise Knowledge Portals (KM*)
  - HealthCare*
  - Financial*
  - Life Sciences*
  - Insurance*...

- E-Business Application Development (UML, SPEM SPM*..)
- E-Business Intelligence, E.I. Portals (CWM, CWM IP, WS*)
- E-Business Application Integration (UML4EAI, UML4BPEL*...)

- Mappings to Platforms (UML4IDL, UML4EJB, UML4WSA*)

- Model & Metadata Infrastructure (MOF, XML, XMI, JMI,...)

- Middleware Platforms (J2EE, CORBA, .Net, Web Services…)

* Opportunity
Early examples of MDA

- OMG has been using MDA principles for about 5 years in several specifications starting with MOF & UML in 1997

- Metadata and Object Interchange/Integration
  - UML and MOF for modeling, MOF and XMI for metadata integration
  - MOF is a subset of UML (Class diagrams) used to model metadata
    - MOF 2.0 and UML 2.0 reuse a common UML2 Infrastructure

- Application Development
  - UML for modeling, XMI and IDL for tool integration and interchange
  - UML profile EJB, CORBA: UML for modeling, J2EE for Integration

- Data Warehousing
  - UML and CWM for Modeling metadata, XMI, JMI, JOLAP, JDM for Warehouse integration

- Application Integration
  - UML profile for Enterprise Application Integration (UML4EAI), ‘Business Objects’ (UML4EDOC)
OMG Model Driven Architecture
MDA for Metadata Integration

UML
Model & Design (PIM)

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

Metadata for
Metadata Mgmt

IDL
MOF2IDL (PSM)

1997
1999

JMI
MOF2Java (PSM)

1997
2001

PIM: Platform Independent Model
PSM: Platform Specific Model

Vertical Industry Specifications
Data Warehousing
B2B Application Integration
Model Driven App Development
MetaData Management
And So On

IBM
Meta Object Facility Overview
Current Revision : MOF 1.4

Discover & Manipulate Metadata

<<Reflection>>
MOF Reflective

<<OMG MetaModel>>
MOF Model

Model Metadata using UML Class Diagrams

Manipulate Distributed Metadata

<<MOF Mappings >>
MOF to XML (XMI)
XML to MOF (XMI)
MOF to Java (JMI)
MOF to CORBA IDL
Applying the MDA Architectural Style for Metadata Management

TARGET
Syntax and Encoding (Target Language..)
XML, Java, IDL...

MOF
Metadata Definitions & Management

SOURCE
Analysis & Design of Meta Model
UML, CWM, UDDI...

PSM TRANSFORMS

TRANSFORMS

Schemas, Models
(1 per source used for validation)

Validate

Documents, Instances

PSM

UML Models
SQL Schema
Travel Objects

UML, CWM, UDDI...

UML DTD
CWM XSD
Travel I/F

XMI, JMI...

Copyright 2003, IBM Corporation
Transform Models to DTD

XMI 1.1 or XMI 1.2 (1999, 2001)
Transform UML Models to Schemas

XMI 2.0  OMG Document : ad/01-06-12,13 (2002)

XML Syntax and Encoding

MOF Metadata Definitions & Management

UML Model Analysis & Design

XML Streams (Models)
(Many - based on each metamodel Schema)

Validate

XML Schema (MetaModels)
(1 per metamodel used for validation)

PSM
Transform Models to Java

JSR-40 : Java Metadata Interface (JMI) 2002

Java Interfaces
(Many - based on each metamodel Schema)

CWM Java I/F
Travel Java I/F
UML Java I/F

Java Metamodel Servers
(1 per metamodel used for validation)

Stream : XMI

Instantiate

UML Objects
CWM Objects
Travel Objects

Java Interfaces

Java Syntax and Semantics

MOF
Metadata Definitions & Management

UML
Model Analysis & Design

MOF2

PIM

PSM

Copyright 2003, IBM Corporation
Slide 24
## The 4Ms of MDA

### Models, Metadata, Mappings, Middleware

<table>
<thead>
<tr>
<th>Problem Domain</th>
<th>Source Meta Model (PIM)</th>
<th>Mapping Specification</th>
<th>Target Middleware, Language (PSM)</th>
<th>Interchange Format/API</th>
<th>Comments</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Development</td>
<td>UML</td>
<td>UML Profile for CORBA</td>
<td>CORBA</td>
<td>XMI</td>
<td>OMG Spec</td>
<td>Adopted 1997</td>
</tr>
<tr>
<td>OO Analysis &amp; Design</td>
<td>UML</td>
<td>UML Profile for SPEM, SPEM Metamodel</td>
<td>Agnostic</td>
<td>XMI</td>
<td></td>
<td>Adopted 2001</td>
</tr>
<tr>
<td>Metadata Management</td>
<td>MOF</td>
<td>MOF to IDL</td>
<td>CORBA</td>
<td>IDL interfaces</td>
<td>PIM for metadata</td>
<td>Adopted 1997</td>
</tr>
<tr>
<td>Software Process</td>
<td>UML</td>
<td>MOF to XML DTD (XML)</td>
<td>XML DTD, XML Documents</td>
<td>XMI</td>
<td>Common Interchange Format</td>
<td>1999</td>
</tr>
<tr>
<td>Engineering</td>
<td>UML</td>
<td>MOF to XML Docs (XML)</td>
<td>XML Schema, XML Documents</td>
<td>XMI</td>
<td>Common Interchange Format</td>
<td>2002</td>
</tr>
<tr>
<td>Enterprise Components/</td>
<td>UML</td>
<td>MOF to Java (JMI)</td>
<td>Java</td>
<td>JMI, XMI</td>
<td>Common Programming Model for MOF</td>
<td>2002</td>
</tr>
<tr>
<td>Middleware</td>
<td>CORBA (CCM spec)</td>
<td>CORBA Component Model</td>
<td>CORBA 3.0</td>
<td>IDL, XMI</td>
<td>First metamodel of the CORBA platform itself</td>
<td>2001</td>
</tr>
<tr>
<td>Enterprise Components/</td>
<td>UML</td>
<td>UML Profile for EJB</td>
<td>J2EE</td>
<td>XMI, JMI</td>
<td>JSR-26 Java Community process</td>
<td>Public Final Draft 2002</td>
</tr>
<tr>
<td>Middleware</td>
<td>UML</td>
<td>UML Profile for EDOC, CCA, EJB and Java Metamodels</td>
<td>J2EE, Java and EJB</td>
<td>XMI, IDL (JMI)</td>
<td>Reuse UML Profile for EJB</td>
<td>2002</td>
</tr>
<tr>
<td>Enterprise Components/</td>
<td>CWM</td>
<td>MOF to IDL, XMI, JMI*</td>
<td>Various Database Middleware</td>
<td>XMI, JMI*</td>
<td>JMI via Java for OLAP and Java for Data Mining</td>
<td>2001</td>
</tr>
<tr>
<td>Middleware</td>
<td>CWM</td>
<td>Java for OLAP*</td>
<td>OLAP - Multi Dimensional Database</td>
<td>JOLAP, XMI</td>
<td>Uses JMI</td>
<td>Public Draft 2002</td>
</tr>
<tr>
<td></td>
<td>CWM</td>
<td>Java for Data Mining</td>
<td>Data Mining, Data base middleware</td>
<td>JDM, XMI</td>
<td>Uses JMI</td>
<td>Public Draft 2002</td>
</tr>
<tr>
<td></td>
<td>CWM</td>
<td>CWM for Web Services</td>
<td>Database Middleware</td>
<td>SOAP, UDDI and WSDL</td>
<td>Uses MOF to WSDL and UDDI mapping</td>
<td>Initial Submission</td>
</tr>
<tr>
<td>Data Warehousing</td>
<td>MOF</td>
<td>MOF to WSDL</td>
<td>WSDL/SOAP</td>
<td>WSDL</td>
<td>Used by CWM for Web Services</td>
<td>Initial Submission</td>
</tr>
<tr>
<td>Business Intelligence</td>
<td>MOF</td>
<td>MOF to UDDI</td>
<td>UDDI/SOAP</td>
<td>UDDI</td>
<td>Used by CWM for Web Services</td>
<td>Initial Submission</td>
</tr>
</tbody>
</table>
How are IBM and IBM partners using MDA for Tool Integration

- Use Eclipse tools integration framework
  - [www.eclipse.org](http://www.eclipse.org) (Java IDE, tools integration framework)

- Use Eclipse EMF for modeling & metadata management
  - [www.eclipse.org/emf](http://www.eclipse.org/emf), [www.eclipse.org/xsd](http://www.eclipse.org/xsd)
  - 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 interchange (XMI, XSD serialization)

- 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…

- Research area: ‘Model Driven Business Integration’
UML
Model & Design (PIM)
Java, XMI, XSD Input

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

EMF2Java
(PSM)

WSDL*

EMF Framework (PIM)

Metadata Interchange
Metadata Interfaces

PIM: Platform Independent Model
PSM: Platform Specific Model

* In progress

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

Copyright 2003, IBM Corporation
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
Model Driven Tools Integration
An end to end view - A peek ahead?

Business Modeling (BPM, UML)
IT Modeling (UML, CWM)
EJB Development Wrapping Orchestration (J2EE, Grid)
Deployment J2EE App Svr Web Services
Management IT & Business Assets Mgmt Tools

MDA Models & Metadata
Traceability Links Transformations

MDA
BPEL
BPM
UML
EMF/MOF
J2EE
XSD
WSDL
...

Metadata @ models, components, code...

Eclipse VCM
EMF/MOF
XML/XMI
JMI

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

Model Transformations across layers
Presentation Overview

- Introduction - The integration problem
- OMG MDA - Model Driven Architecture Overview
  - What is MDA and why is it important
  - Emerging MDA standards for Model Driven Business Integration
- Web Services Architecture Overview
  - What are Web Services and why are they important
  - Emerging Web Services Standards and Architecture
- MDA and Web Services – How do you use them together
  - Modeling Web Services
  - Advanced Modeling (Web Services Orchestration)
  - Model Driven Business Integration – A peek ahead
- Wrap-up
What are Web Services?

- ‘Modular and reusable’ applications that can be
  - Invoked over the web (using a protocol like SOAP) for exchanging XML documents
  - Described using a service description language (like WSDL)
  - Published into a registry (like UDDI) so the service can be searched, bound and invoked
  - Orchestrated using a business process execution language (like BPEL4WS)

- Web Services do not imply component or object based development – but obviously they can be used together.
  - In fact a 20 year old CICS transaction can be wrapped as a web service

- Web Services need to be composable
  - Enable better granularity of services and solutions being delivered
  - Enable service orchestration/flow

- Can be simple to very complex; free to expensive
What is a Web Service?

‘The challenge of defining it’

Source: W3C Technical Plenary - 2002
Web Services: The basic idea

<<Subsystem>>
Service Provider
Ex: UAL

<<Subsystem>>
Service Requestor
Ex: Travel Agent

<<Subsystem>>
Service Registry
Ex: AOL
UDDI Registry

Publish Service
Frequent Flyer Service (WSDL)

Bind to Service

Find Service
SOAP Messages
**Web Services – Some Key Events (When!)**

- The early pioneers
  - XML RPC at WebMethods
  - Dave Winer (CEO Userland Software) defines SOAP
  - Microsoft jumps on the SOAP bandwagon
  - IBM endorses SOAP and then the flood follows

- The UDDI (Universal Description, Discovery and Integration) efforts by Microsoft, IBM and Ariba gave another strong push

- Soon WSDL (Web Services Description Language) joined the party

- SOAP (XML/HTTP), UDDI and WSDL form the core formats and protocols of any Web Services Architecture today

- All major vendors, middleware platforms and architectures have embraced/endorsed these technologies
  - Platforms: WebSphere, WebLogic, .Net Servers

- Now we enter the second phase: Orchestration/Flows, Security, Transaction standardization
Web Services
The Hype

- Middleware ‘nirvana’ that ‘solves’ the integration problem
- Everyone is using it
- Works well
- Makes CORBA, J2EE, COM+ obsolete
- Brand new innovation
- Every developer better learn it ‘or else’…
- There is a huge market
- Standards are simple and usable and complete
**Web Services**

**The Reality**

- An important step in the right direction towards lower cost ubiquitous middleware
  - Key technologies – XML, HTTP, SOAP and WSDL are freely available on most platforms
  - Key enabler for ‘Service Oriented Architecture’
- Most vendors and some customers are incorporating web services into their architectural frameworks and plans
  - Few production quality implementations, but growing
- Standards and tools are beginning to mature
  - More work – especially Transactions, Security etc.
- Being integrated with J2EE, .Net and CORBA frameworks
- Products : WebSphere, WebLogic, Microsoft Biztalk server…
- Market potential : Promising but will need wide spread interoperable implementations (WS-I role)
Web Services
Why are they important?

- Promise of delivering ‘information services’ to anyone, any program, anywhere at anytime
  - Some services free, others not
- All the major vendors have (at least for now!) called a truce and embraced web services
- Better chance for improved interoperability across technology silos
- Development tools & Middleware from Microsoft, IBM, BEA… have rapidly embraced web services
- Open standards for ‘Service Oriented Architectures’
  - More pragmatic infrastructure for business model and process integration standards
- W3C, OASIS and WS-IO leading the definition of additional standards
Web Services: The Stack - 1

- Network
  - TCP/IP, HTTP, FTP, SMTP...

- XML Messaging
  - Messages conform to well defined XML Schemas
  - SOAP is the preferred messaging infrastructure
    - Eventually W3C XMLP will supplant SOAP

- Service Description
  - These are essentially interface definitions (we called these IDLs before!)
  - WSDL is the interface definition language for web services
  - Implementation (binding) information is also captured by WSDL
Service Publication and Discovery
- Publish the service definition and implementation metadata
- Capture additional business oriented service description information (D&B, Phone, URL…)
- This metadata is published into a UDDI Registry
  - Registry is searchable
- Service discovery happens dynamically at run time
  - Similar to CORBA Interface Repository and DII; Java Reflection + JNDI
  - Services are bound statically at design/development time

Service sequencing and flows build on XSD and WSDL
- BPEL4WS – Unification of IBM WSFL and Microsoft XLANG
- Endorsed by BEA, SAP and many more vendors
- BPML now builds on BPEL4WS
Service Flows

- Individual services can be steps in a flow (BPEL4WS)
- Flows can be graphically modeled
  - Various notations including UML Activity Diagrams have been used
  - No dominant industry standard for visual notations
  - OMG Business Process Definition Metamodel RFP to address this issue
- Service composition needed for multi party service integration

This area does not have as much consensus as lower level protocols

Standardization track still being resolved
Web Services: The Stack Evolves

Specific Composable Services and Processes (ebXML, RosettaNet PIP...)

Service Flow (BPEL4WS..)

Service Publication and Discovery (UDDI)

Service Description (WSDL)

XML Messaging (SOAP, XMLP)

Network (HTTP, FTP...)

Copyright 2003, IBM Corporation
Summary Web Services Standards

- W3C
  - XMLP (SOAP+), WSDL, XAML...
  - W3C Architecture Working Group

- UDDI Consortium ➔ UDDI 3.0 being progressed in OASIS

- SUN Java Community Process
  - Integrate J2EE and Web Services Technologies
  - JAXP, JAX RPC, JAXB, JAXR, JAXM (J2EE 1.4 target – JSR 109)
    - http://jcp.org/jsr/detail/109.jsp

- OMG
  - Web Services for Business Intelligence
  - Modeling and metadata management for Web Services
  - UML2 Activity Graphs, UML2 Infrastructure, MOF2
  - Business Process Definition Metamodel RFP

- OASIS
  - ebXML, UDDI...

- Web Services Interoperability Organization (WS-IO)

- Web Services Security (WS-Security, WS-Trust)
  - IBM, Microsoft, Verisign joint proposal submitted to OASIS
  - www.ibm.com/developerworks/library/ws-secure
Presentation Overview

- Introduction - The integration problem
- OMG MDA - Model Driven Architecture Overview
  - What is MDA and why is it important
  - Emerging MDA standards for Model Driven Business Integration
- Web Services Architecture Overview
  - What are Web Services and why are they important
  - Emerging Web Services Standards and Architecture
- MDA and Web Services – How do you use them together
  - Modeling Web Services
  - Advanced Modeling (Web Services Orchestration)
  - Model Driven Business Integration – A peek ahead
- Wrap-up
Business Models, Processes and Rules
Model Driven Tools Integration
An end to end view - A peek ahead?

Business Modeling (BPM, UML)
Arch, Process, Rules...

IT Modeling (UML, CWM)
Arch, Objects, Data, Flows...

Developing Testing
Wrapping Orchestration
(J2EE, Grid)

Deployment J2EE App Svr Web Services

Management Monitoring
IT & Business Assets Mgmt Tools

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

Model Transformations across layers

Traceability Links Transformations

Metadata @
models, components, code...

MDA Models & Metadata

BPEL BPM UML
J2EE EMF/ MOF
XSD WSDL

APIs, Interchange Formats

Eclipse VCM
EMF/MOF
XML/XMI
JMI

XML JDBC Store DB2 Clear Case

Copyright 2003, IBM Corporation
Modeling for Web Services

- UML can be used to model Web Services
  - Web Service Operations (WSDL Operations) are 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

- MOF and XMI can be used to define standard transformations from models to XML schemas, DTDs, Documents

- But standard metamodels, transformations, UML profiles etc, would be useful
  - [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 BPEL4WS mappings
**Business Process Definition**

**Metamodel RFP**

- Issued January 2003
- Goal is to define a platform independent model that unifies the modeling concepts in emerging flow/choreography standards
- Proof of concept mappings to emerging choreography standards (BPEL4WS, XPDL…)
- Basis for providing alternate visualizations as well as textual/machine readable syntax
- Link to rest of application development/integration cycle
E-Business Integration using XML

Source: Software Development – February 2002
Modeling XML Applications with UML, David Carlson
Web Services for Book Store – An Example

<<Subsystem>>
Service Provider
Ex: Amazon.com

<<Subsystem>>
Service Requestor
Ex: Orange County Public Library

<<Subsystem>>
Service Registry
Ex: AOL Retail UDDI Registry

Publish Service
Bind to Service
Find Service
SOAP Messages

Book Info Request (WSDL)
Modeling Web Services and XML

Usage: Model the Application

Source: David Carlson www.xmlmodeling.com
An XML Document that conforms to the UML model

Usage: Generate XML Schemas and Documents
Web Services for Metadata Management – An Example

- **Subsystem>>** Service Provider
  - Ex: Adaptive
  - Publish Service
  - Export Service (WSDL)
  - Bind to Service

- **Subsystem>>** Service Requestor
  - Ex: SAS CWM Server
  - Find Service
  - SOAP Messages

- **Subsystem>>** Service Registry
  - Ex: IBM MDA UDDI Registry
  - Metadata Import

Web Services for Metadata Management

WSDL Header

<?xml version="1.0" encoding="UTF-8" ?>
<definitions name="AdaptiveWebService" xmlns="http://schemas.xmlsoap.org/wSDL/


xmlns:soap=\"http://schemas.xmlsoap.org/soap/\"

xmlns:wsdl=\"http://schemas.xmlsoap.org/wsdl/\"

xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\"

xmlns:serviceNS=\"urn:ws.adaptive.com\" xmlns:typeNS=\"urn:ws.adaptive.com.types\">

- <types>

- <schema targetNamespace="urn:ws.adaptive.com.types"

xmlns=\"http://www.w3.org/2001/XMLSchema\">

- <complexType name="ArrayOf_xsd_string">

- <complexType>

- <restriction base="soap-enc:Array">

- <attribute ref="soap-enc:arrayType" wsdl:arrayType="xsd:string

[]" />

</restriction>

</complexType>

</complexType>

</schema>

</types>

</definitions>
Web Services for Metadata Management

WSDL for Adaptive Web Services Port

```xml
<definitions>
  - <service name="AdaptiveWebService">
    <documentation>A set of Adaptive Web Services for interacting with the Adaptive Repository via Adaptive Portal Pro. Includes XMI import/export, store creation and a general searching mechanism</documentation>
    - <port binding="serviceNS:AdaptiveWebServiceSoapBinding" name="AdaptiveWebServicePort">
      <soap:address location="http://localhost:8080/adaptive/services/AdaptiveWebService"/>
    </port>
  </service>
</definitions>
```
Web Services for Metadata Management

WSDL Ports and Operations

```xml
<message>
  - <portType name="AdaptiveWebServicePortType">
    - <operation name="importXMI">
      <documentation>Imports an XMI datastream into the store represented by
      the defaultWorkArea of the supplied context. The username and
      password must be a valid login account for the facility specified by the
      rid parameter. A server must have already been set running for this rid
      using the ServerPoolManager. The context parameter may be either its
      name or its UUID. The import can be controlled through the
      importParams which should be a list of strings. None or more of the
      following strings can be included with the following effect: isWarning -
      indicates whether the XML parser should report warnings on import
      isValidatingParser - indicates whether a validating parser should be used
      for the import skipInvalidElements - indicates whether to skip elements
      not found in the model The import returns a list of uids corresponding
      to the top level imported objects.</documentation>
      <input message="serviceNS:importXMIRequest" />
      <output message="serviceNS:importXMIResponse" />
      <fault message="serviceNS:WSException" name="message" />
    </operation>
  - <operation name="exportStoreXMI">
    <documentation>Exports the contents of an entire store as an XMI
    datastream. The store exported corresponds to the defaultWorkArea of
    the supplied context. The context parameter may be either its name or
    its UUID.</documentation>
  </operationType>
</message>
```
Web Services for Metadata Management

WSDL messages for XMI Import/Export

```xml
<types>
  <message name="importXMIRequest">
    <part name="userName" type="xsd:string"/>
    <part name="password" type="xsd:string"/>
    <part name="rid" type="xsd:string"/>
    <part name="context" type="xsd:string"/>
    <part name="validationParams" type="typeNS:ArrayOf_xsd_string"/>
    <part name="dataStream" type="xsd:base64Binary"/>
  </message>
  <message name="importXMIResponse">
    <part name="importXMIResult" type="typeNS:ArrayOf_xsd_string"/>
  </message>
  <message name="exportStoreXMIRequest">
    <part name="userName" type="xsd:string"/>
    <part name="password" type="xsd:string"/>
    <part name="rid" type="xsd:string"/>
    <part name="context" type="xsd:string"/>
    <part name="exportUUIDs" type="xsd:boolean"/>
  </message>
  <message name="exportStoreXMIResponse">
    <part name="exportStoreXMIResult" type="xsd:base64Binary"/>
  </message>
  <message name="exportObjectsXMIRequest">
    <part name="userName" type="xsd:string"/>
  </message>
  <message name="exportObjectsXMIResponse">
    <part name="exportObjectsXMIResult" type="xsd:base64Binary"/>
  </message>
</types>
```
Web Services for Metadata Management

WSDL messages for XMI Import/Export

 Courtesy : Adaptive
How do Web Services fit into existing Software Architectures

- **W3C Architecture Working Group**
  - Architectural framework and liaison with other groups

- **OMG Model Driven Architecture**
  - OMG CWM for Web Services, MOF to WSDL, UDDI
  - IDL/WSDL and WSDL/IDL Mappings
  - OMG Web Services SIG

- **J2EE – Web Services Architectural extensions**
  - JAXP, JAXM, JAXB, JAXR, JAX-RPC

- **Microsoft .Net ‘connected’**
  - Weaving Web Services everywhere

- **OMG – Integrating Business Models, Processes & rules with J2EE, Web Services and MDA**
  - UML2, Business Process Definition Metamodel, Business Rules RFP (coming)
A peek at integrating Business Processes using UML and BPEL4WS

This is early work and is intended to spur Discussion!

See the BPEL4WS spec for the examples, BPEL4WS syntax

May be this will result in a standard UML profile for BPEL
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.
‘Business Integration Profile’ using MDA and Web Services

- «mdx» XDE Model
  - /XDE XMI 1.x Export
  - convert to UML1.4

- «mdl» Rose Model
  - /Unisys XMI 1.3 Export
  - convert to UML1.4

- «XMI2» UML 1.4 Model
  - Mapping Rules
  - UML1.4 to Web Services
    - XSD Schema
    - WSDL Definition
    - BPEL Process

- BPWS4J
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
This is the UML 2 version, we approximate this in UML 1.4 (more detail later)
Behavior Basics

Receive a synchronous operation call:

```
receivePO
```

Reply to a previous synchronous request:

```
reply

returnInvoice
```

Invoke an operation synchronously or asynchronously on a partner:

```
invoke

initiatePriceCalculation
```
Purchase Order Process

Partitions correspond to ports (partners)
Data and Message Types

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

+ customerInfo
+ purchaseOrder
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
Purchase Order Process - Structure
### BPEL Mapping Overview

<table>
<thead>
<tr>
<th>BPEL Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&lt;process&gt;&gt; class</td>
<td>BPEL process definition</td>
</tr>
<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</td>
<td>BPEL containers</td>
</tr>
<tr>
<td>attributes</td>
<td>BPEL sequence and flow activities</td>
</tr>
<tr>
<td>Hierarchical structure and control flow</td>
<td>BPEL switch activities and transition conditions</td>
</tr>
<tr>
<td>Decision nodes</td>
<td>BPEL receive, reply, invoke activities</td>
</tr>
<tr>
<td>&lt;&lt;receive&gt;&gt;, &lt;&lt;reply&gt;&gt;, &lt;&lt;invoke&gt;&gt; activities</td>
<td>BPEL service links types and roles</td>
</tr>
<tr>
<td>&lt;&lt;protocol&gt;&gt; package with &lt;&lt;role&gt;&gt; classes</td>
<td></td>
</tr>
</tbody>
</table>
Mapping: Process

```xml
<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

```
<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: Receive/Reply Activities

```xml
<receive container="request" createInstance="yes"
  name="receive1" operation="approve"
  partner="customer"
  portType="LoanApprover:LoanApproval"
  suppressJoinFailure="no">
  <source linkName="receive1_to_invokeApprover"
    transitionCondition="bpws:getContainerData('request','amount') \geq 10000"/>
  <source linkName="receive1_to_invokeAssessor"
    transitionCondition="bpws:getContainerData('request','amount') \lt 10000"/>
</receive>

... 

<reply container="approvalInfo" name="reply"
  operation="approve"
  partner="customer"
  portType="LoanApprover:LoanApproval"
  suppressJoinFailure="no">
  <target linkName="assign_to_reply"/>
  <target linkName="invokeApprover_to_reply"/>
</reply>
```
Mapping: Assign Activities

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

Definition
- targetNamespace : String
- qName : QName
- encoding : String

SetDocument

GetDocument

Import

- schema

GetSchema

Types

- eTypes

0..1

Message

- eMessages

0..n

PortType

- ePortTypes

0..n

Binding

- eBindings

0..n

Service

- eServices

0..n
Draft BPEL Model

Process
- name : String
- targetNamespace : String
- queryLanguage : String
- expressionLanguage : String
- suppressJoinFailure : Boolean = false
- containerAccessSerializable : Boolean = false
- enableInstanceCompensation : Boolean = false
- abstractProcess : Boolean = false

ExtensibleElement
- getExtensibilityElements() : IList
- addExtensibilityElement(element : ExtensibilityElement)

ServiceLinkType
- name : String

PortType
- name : QName
- defined : boolean
- proxy : boolean
- resourceURI : String
  (from wsdl)

Role
- name : string
  1..2
  1
  + role
  1..2
  + responsibilities

Part
- name : String
  0..*

Link
- name : String
- transitionCondition : BooleanExpression = true

Property
- name : String
- XSDDTypeDefinition
  (from xsd)

ExtensibleElement
- getExtensibilityElements() : IList
- addExtensibilityElement(element : ExtensibilityElement)

Extension
- namespace : String
- type : String

CompensationHandler

Partner
- name : String

FaultHandler

Container
- name : String

CorrelationSet
- name : String

Activity
- source : String
- targets : List

Message
- qName : QName
- undefined : boolean
- proxy : boolean
- resourceURI : String
  (from wsdl)

CorrelationSet
- properties : List

FaultHandler

CompensationHandler

Property
- name : String
- propertyName

PropertyAlias
- query : String
- propertyName

Message
- qName : QName
- message

FaultHandler

Activity
- activity

Link
- target
- targetLinkStatus

Copyright 2003, IBM Corporation Slide 80
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?)
  - OMG Business Process Definition Metamodel RFP
  - OMG Business Rules RFP Drafts : Rule Expression Language, Rule Management…
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

- 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 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
- MDA : www.omg.org/mda
- IBM : www.ibm.com
- W3C : www.w3c.org
- DSTC : www.dstc.edu.au
- Sridhar : siyengar@us.ibm.com
- UML RTF : uml-rtf@omg.org
- MOF RTF : mof-rtf@omg.org
- XMI RTF : xmi-rtf@omg.org
- CWM RTF : cwm-rtf@omg.org