A Model Based Enterprise Architecture for Web Services and XML

Enterprise Collaboration Architecture

Applying Model Driven Architecture using the OMG Enterprise Collaboration Architecture and XML Web Services
Introductions

Cory Casanave
cory-c@enterprise-component.com

Primary author of “CCA” in EDOC
The OMG-Enterprise Collaboration Architecture

- ECA is a “profile of UML”, a way to use UML for a specific purpose - it is an OMG standard
  - That purpose is modeling enterprise systems.
- ECA is part of the “Model Driven Architecture” (MDA) initiative of the OMG
  - Using precise modeling techniques as part of the development lifecycle to speed development and provide technology independence
- ECA has been adopted by the OMG as part of the EDOC Profile for UML specification.
- RFP in process for Web Services Mapping
Typical Problems

- Integration Nightmare
- Infrastructure, Version & Vendor lock-in
- Complex, divergent and manual development and deployment processes

Solutions typically require buy-in (Lock-in) to proprietary tools & infrastructure - your solution must be open
Automated Model Driven Architecture

Domain Model (PIM)

Tools Produce & Integrate

Enterprise Components

Framework & Infrastructure
(E.G. -J2EE-WS)

PSM

Mapping
(E.G. J2EE-WS)

Technical Architecture

Minimize and structure manual implementation

Mapping is tuned to the infrastructure

Domain Architecture

Automated Model Driven Architecture

Tools Produce & Integrate

Minimize and structure manual implementation

Mapping is tuned to the infrastructure

Technical Architecture
Automated Model Driven Architecture

- Meta-Model: UML Profile (E.G. ECA)
- Domain Model (PIM)
- Domain Architecture
- Automated Model Driven Architecture Framework & Infrastructure (E.G. J2EE--WS)
- Mapping (E.G. J2EE--WS)
- Tools Produce & Integrate

Mapping is tuned to the infrastructure

Multiple and Changing Technology Support

- J2EE-WS Enterprise Components
- .NET-WS Enterprise Components
- Framework & Infrastructure (E.G. -J2EE-WS) PSM
- Framework & Infrastructure (E.G. -.NET-WS) PSM

MDA Solution for Web Services

Enterprise Collaboration Architecture

Web Services
For Enterprise Collaboration

Web Services Stack

Platform (J2EE, .NET...)

Platform Independent Model
Mapping
Platform Dependent Model
Mapping
Not yet standard

XML Components

ECA and Web services together provide
An XML component architecture
Independent of protocol and platform
ECA as the normal form

The standard way to model and tool for multiple technologies

MDA Mappings

- EDOC-ECA
- Web Services (WSDL)
- ebXML (BPSS)
- J2EE (Java RMI)
- .NET
- MOM (MQ-Series)
EDOC Component Collaboration Architecture

CCA

The model of collaborative work
The Marketplace Example

Mechanics Are Us
Buyer

Process
Complete

Physical
Delivery

GetItThere Freight
Shipper

Order

Conformation

Status

Ship Req

Shipped

Delivered

Acme Industries
Seller

The Seller's Detail

Order Processing

Shipping

Event

Receivables
Multiple roles in a collaboration
Drilling down – inside a role

- The open domain should make no assumptions about the “inside” of a role.
- Inside one role you frequently find more collaborating “parts” of the enterprise - the same model may be used.
- Until you get to system inside a managed domain:
  - Shared resources (DBMS)
  - Common Management
  - Frequently a legacy system
Roles to Systems

Role

Collaboration

Framework, Middleware & Container

Interaction

Interaction Path

(With Information)

Implementation

Operating System

Hardware

Net

Component in Role
Parts of a CCA Specification

- Structure of process components and protocols
  - Process components, ports, protocols and documents
    - Class Diagram or CCA Notation

- Composition of process components
  - How components are used to specify components
    - Collaboration diagram or CCA Notation

- Choreography
  - Ordering of flows and protocols in and between process components
    - Activity Diagram
The Community Process

Identify a “community process”, the roles and interactions in a collaboration
Community Process (CCA)

CCA Notation
Community Process

Generic UML

UML Collaboration Diagram
Protocol (CCA)
Protocol Choreography

When

SendOrder

GetConfirmation

GetDenied

Internal Logic

<<Success>>

<<BusinessFail>>

Validation Component (CCA)
Validation Component

Generic UML

UML Class Diagram
Choreography

When

Order Validation Choreography

checkOrder

CheckCustomer

acceptOrder

reject

success

failure

Internal Logic
Composition (CCA)
Composition (UML Collaboration)

Generic UML
Composition
Aspects

Generic UML

Tagged Values
WSEC

Web Services for Enterprise Collaboration

Initial Proposal
Distributed Components

- Define “role” as the abstract contract
- Define “Engine” as exposing a set of DCs
- Define “Endpoint” as consuming a set of DCs
- Define “Proxy” as the use of an external role
Engine exposing a DC
Defining an external component resource
Using a proxy
Mapping of a WSDL Engine

- `<definitions xmlns="http://schemas.xmlsoap.org/wsd`
Mapping of a DC

- `<service name="MySeller">`
- `<!--
implemented service role
/BuySell/Deployment/SellerServer/MySeller  -->
  <documentation>"<p> </p>"</documentation>
- `<! -->
- `<port name="BuySellProtocol"
binding="tns:BuySellProtocol">`
  - `<!--
original service port was
/BuySell/Deployment/SellerServer/MySeller/BuySellProtocol (extending Component
&\lt;/BuySell/SellerImplementation/MySeller/BuySellProtocol\gt; )  -->
  <soap:address
`
Mapping of a protocol binding

```xml
<binding name="BuySellProtocol" type="tns:BuySellProtocol">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="rpc" />
  <operation name="Order">
    <soap:operation soapAction="urn:/BuySell/Community/BuySellProtocol/Order" style="rpc" />
    <input name="Order">
      <soap:body use="encoded" namespace="urn:SellerServer/Order" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
    </input>
  </operation>
</binding>
```
Mapping of a protocol

- `<portType name="BuySellProtocol">`
  - `<!-- original cx operation = /BuySell/Community/BuySellProtocol/Order -->`
  - `<operation name="Order">`
    - `<!-- original cx flow port = /BuySell/Community/BuySellProtocol/Order/Order -->`
    - `<input name="Order" message="tns:Order" />`
    - `<output name="OrderConfirmation" message="tns:OrderConfirmation" />`
    - `<fault name="OrderDenied" message="tns:OrderDenied" />`
  - `</operation>`
- `</portType>`
Mapping of message types

Aspects
- WSDL
- WSDL-SOAP

```
<message name="Order">
  <part name="Order" type="Ordering:Order"/>
</message>

<message name="OrderConfirmation">
  <part name="OrderConfirmation" type="Ordering:OrderConfirmation" />
</message>

<message name="OrderDenied">
  <part name="OrderDenied" type="Ordering:OrderDenied" />
</message>
```
Mapping of data types

```xml
<xs2001:complexType name="Order">
  <xs2001:sequence>
    <xs2001:element minOccurs="1" maxOccurs="1" name="CompanyID" type="CoreTypes:CompanyID" />
    <xs2001:element minOccurs="1" maxOccurs="1" name="OrderID" type="Ordering:OrderID" />
    <xs2001:element minOccurs="0" maxOccurs="unbounded" name="Item" type="Ordering:Item" />
  </xs2001:sequence>
</xs2001:complexType>
```
Adding Entities

- Entities are added to manage entity data
- Entity Roles are managers that provide a view of the same identity in another context
- The Entities have ports for managing and accessing the entities
- Non-entities which are owned by (aggregate into) an entity are managed by the entity
Event Tied to Information
Iterative Development

Business Model Design

Automation

Infrastructure Development

Build Build Build Build Build

Release Build

Deploy
MDA Solution Factory

- Put together the
  - Best practices
  - Expertise
  - Enterprise Architecture
  - Infrastructure
  - Automated tooling

- To produce and integrate robust business collaborations quickly & reliably
Net effect

Using these open standards and automated techniques we can;

- Achieve the strategic advantage of an open and flexible enterprise
- Produce and/or integrate these systems FASTER and CHEAPER than could be done with legacy techniques
- Provide a lasting software asset that will outlive the technology of the day
Data Access Technologies

Products & Technologies

- Component-X (Product)
  - Implements the EDOC-CCA and ebXML “BPSS” using XML and Java
  - Provides drag and drop specification & assembly of enterprise components for collaborative web services

- Model Driven Architecture (Technology)
  - Based on OMG-MOF & UML
  - Provides forward and reverse engineering between models and technology artifacts
  - Use to automate and integrate multiple technologies
Information & Contact

OMG MDA & ECA

www.omg.org/mda

Cory Casanave
cory-c@enterprise-component.com
(305) 234-7077