Producing Web Services using UML and MDA

Enterprise Collaboration Architecture

A tutorial on applying Model Driven Architecture to web services using the OMG Enterprise Collaboration Architecture with UML
Introductions

Cory Casanave
cory-c@enterprise-component.com

Primary author of “CCA” in EDOC
What is the 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**.
- You can also think of this as a “modeling framework” for enterprise computing
- 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 RFP.
Collaboration is Key

- Collaboration is a key differentiation and key cost center (Healthcare Example)
  - Customer Collaboration
  - Claim processing
  - Disputes
  - Physician Collaboration
  - Payer Collaboration
  - Hospital Collaboration
  - Broker Collaboration
  - Government Collaboration
  - Employee Collaboration
  - Others…

The system integrates multiple collaborations
The Agile Enterprise

Has a competitive advantage in its capability to embrace collaboration and change
Integrating Enterprises, People & Systems - Worldwide

- Business Requirements
  - Virtual Enterprises
  - Enterprise Integration (EAI)
  - Supply-chain automation (B2B)
  - Customer Integration (B2B)
  - Web deployment (B2C)
  - Internet Marketplace (B2C)

- Collaboration and Integration
The dynamic reality

- The information system must facilitate:
  - Rapid realization of business goals
  - Integration of independent processes and systems
  - Multiple and Changing
    - business requirements
    - business processes
    - technologies
    - standards
    - enterprise boundaries
    - partners
Technology Stew

- Web services
- .NET
- C'
- XML
- EAI
- Active Web pages
- EJB
- Java Beans
- Java
- Corba
- MQ-Series
- C++
- SQL
- Cobol
- IMS
- CICS
- ...

Technology is transient, but we must embrace and adapt to it to provide meet current requirements
MDA Solution for Web Services

Enterprise Collaboration Architecture

Web Services for Enterprise Collaboration

Web Services Stack

Platform (J2EE, .NET...)

Platform Independent Model

Platform Dependent Model

Mapping

Mapping

Not yet standard

Standards

Future Standards
XML Components

ECA and Web services together provide
An XML component architecture
Independent of protocol and platform
Problem Space

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

*Typical solutions require buy-in (Lock-in) to expensive, pervasive and proprietary infrastructure*
Solution Triad

Development Process
Tooling & Infrastructure
Standards

Web Services
Corba

Service Based Architecture

Components

Model Driven Development

OMG ECA

J2EE .NET
The new center

- The strategic core of your systems must be the business itself.
- Only technology independent business focused models will survive the transience of technology and lock-in.
- These models can become part of your source code, driving enterprise applications.
- Enabler: Model Driven Architecture (MDA) with EDOC-ECA.

Extreme Modeling
Automated Model Driven Architecture

Meta-Model
UML Profile
(E.G. ECA)

Domain Model (PIM)

Domain Architecture

Infrastructure Mapping
(E.G. J2EE-WS)

Tools Produce & Integrate

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

Tools
Produce & Integrate

Minimize and structure manual implementation

Technical Architecture

Mapping is tuned to the infrastructure
Automated Model Driven Architecture

- Meta-Model UML Profile (E.G. ECA)
- Domain Model (PIM)
- Infrastructure Mapping (E.G. J2EE-WS, .NET-WS)
- Tools Produce & Integrate

Mapping is tuned to the infrastructure

Multiple and Changing Technology Support

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

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

Copyright © 2000-2003, Data Access Technologies, Inc.
Enterprise Architecture

How to slice and dice
Loose Coupling

- Loose coupling is the ability for independent parts of systems to be built and evolve independently.

- Tightly coupled systems
  - Prevent change (the next legacy system)
  - Cause lock-in
  - Become unmanageable
  - Prevent reuse

- Quality architecture is essential for loose coupling.
Architecture Goals

- Create a system from loosely coupled “enterprise components” that can evolve independently
- Provide well defined interfaces and interaction points between these enterprise components
- Make each enterprise component a reusable asset that can serve many business processes
- Build the information system as a community of interacting enterprise components
- Utilize open standards such as XML, EJB and Corba to integrate the enterprise components
Enterprise Components

- Enterprise Components must be independent
- While being able to interoperate with each other
- Making the information system a lattice of cooperating components
“Wrapping” Legacy Applications and Data

- Enterprise Components are defined in terms of their external contract; implementation may use existing applications
  - Can “call” existing application
  - Can read and write legacy DBMS
  - Can use “screen scraper” (Last resort)

- Legacy applications can appear as enterprise components but may not be implemented as components
Legacy “Wrapping”

Wrapping allows existing programs and data to work with and work as enterprise components.
Technology Independence

Adapters
- ebXml
- BizTalk
- Rosetanet
- EJB

Business Logic Component

Adapters
- EJB
- MQ
- Corba
- CICS
Typical Requirement

Buyer Web Page

HTML

Seller

B2B Buyer

Web Service

Seller

Redundant Work!
Multi-tier implementation

Could have multiple implementations using different technologies

Could have multiple implementations using different technologies
Multi-tier implementation

Buyer Web Page

HTML

Buyer Proxy

Web Service

Seller

Event

Event

Event Cloud

Implementing seller using events

Legacy Seller Applications
Collaborations and Roles

Conceptual Foundation

Portions copied with permission of Trygve Reenskaug - Taskon

OORAM

(http://www.ifi.uio.no/~trygver)
History

OORAM
Object Oriented Role Analysis

UML
Collaborations

Enterprise Collaboration Architecture

Influence
The Connected Enterprise
Content and Communication

Digital Map

Census Data

Police Records

House Drawings

Aerial Photos

Police Dispatcher Role
Multiple roles in a collaboration
Travel Expense Example

1: travelPermissionRequest
2: travelPermission
3: expenseReport
4: authorizedExpenseReport
5: paymentRequest
Diagram
Travel Expense Model

Objects --> ClassifierRoles

Adam (Chief Accountant)

/ Paymaster

/ BookKeeper

John (Cashier)

/ Authorizer

/ Traveler

/ Paymaster

Elsie (Programmer)

Kim (Methodologist)

/ Authorizer

Bill (Dispatcher)

Joyce (Sales clerk)

Ann (Customer consultant)

Douglas (Marketing manager)

Ruth (President)

Peter (Technical author)

Bill (Dispatcher)

Joyce (Sales clerk)

Douglas (Marketing manager)

Adam (Chief Accountant)

Ruth (President)

Copyright © 2000-2003, Data Access Technologies, Inc.
Collaboration Diagram

- Traveler
- Authorizer
- Book Keeper
- Paymaster
Synthesis - Components play several roles
Roles to Systems

Component in Role

Role

Collaboration

Interaction Path

Interaction (With Information)

Implementation

Framework, Middleware & Container

Operating System

Net

Hardware
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
Discussion Point

- What are other ways of representing business semantics?
- What role do these play?
  - Business Processes
  - Information Models (DBMS)
  - Events
  - Workflow
  - Internet
  - Ontologies
- What is used internally?
Standards for Global Internet Computing

XML
WSDL
SOAP
XML-Schema

.NET
BPML
XLANG

Creating A Single Global Electronic Market
XML Standards

- XML Schema & DTD
  - Description and packaging of data

- WSDL
  - Specification a services, operations and flows available via that service

- Soap
  - Basic messaging and packaging
  - Extensions for Soap-RPC with WSDL
  - May be extended to support collaborative messaging
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)
Summary of points thus far

- We must enable the emerging Internet Computing Model
  - Loosely coupled roles exchanging documents based on a contract of collaboration
- Web need interoperability at two levels
  - Messaging for the data
  - Metadata for the contract of collaboration, stored in repositories
- This model of collaborating roles is recursive, extending into the enterprise, into managed domains and into applications
  - Inside the enterprise we want to include resources, entities, business events and business processes
- Supporting the open domain has some required parts and can be augmented with a “treasure chest” of tools and infrastructure
- Between EDOC & ebXML we are covering B2B and intra enterprise
EDOC Component Collaboration Architecture

CCA

The model of collaborative work
The Marketplace Example

Mechanics Are Us  Buyer

Process Complete

GetItThere Freight  Shipper

Acme Industries  Seller

Status

Order

Conformation

ShipReq

Shipped

Delivered

Physical  Delivery

Copyright © 2000-2003, Data Access Technologies, Inc.
The Seller’s Detail

Order Processing
Shipping
Receivables

Event

Order
Conformation
Shipped
Ship Req
Shipped
Delivered
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
Identify a “community process”, the roles and interactions in a collaboration.
Protocols
Composition
ECA Entity Profile

The model of things
Data Inside a “shared domain”
Adding Entities

- Entities are added to manage entity data
- Entity Roles are managers that provides 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
ECA Business Events

The model of when...
Loosely coupled integration within the enterprise and with “aligned” business partners
Event Based Business Processes

Event Notification

Business Rules

Business Process

Business Actions

Business Entity

Business Events

Business Rules

Business Process

Business Events

Business Actions

Business Entity

Copyright © 2000-2003, Data Access Technologies, Inc.
Event Example
Events or Services

- Services - conversation between known parties
  - Good supply chain model

- Events - Actions based on notification of events
  - Parties do not know about each other!
  - More loosely coupled
  - Works best inside a managed domain (enterprise)
EDOC CCA Example

Buy-Sell Process
Community Process (CCA)
Community Process

UML Collaboration Diagram
Protocol (CCA)
Protocol

Class diagram for buy/sell protocol

<<CompositeData>>
Order

<<FlowPort>>
SendOrder (from BuySellProtocol)

<<Protocol>>
BuySellProtocol

<<FlowPort>>
GetConfirmation (from BuySellProtocol)

<<CompositeData>>
OrderConfirmation

<<FlowPort>>
GetDenied (from BuySellProtocol)

<<CompositeData>>
OrderDenied

UML Class Diagram

Copyright © 2000-2003, Data Access Technologies, Inc.
Protocol Choreography

When

SendOrder

GetConfirmation

GetDenied

Internal Logic

<<Success>>

<<BusinessFail>>
EDOC Exercise - Community

Part one
- Pick a collaboration from the first exercise
- Model it with EDOC notation as a community process

Identify an interaction
- Specify the documents of the interaction
  - What are some of the major elements?
- Specify any choreography
Validation Component (CCA)
Validation Component

UML Class Diagram
Choreography

Order Validation

checkOrder

CheckCustomer

acceptOrder
  success
  failure

reject

Internal Logic

When
Composition (CCA)
Composition (UML Collaboration)

Seller Composition

Validate : OrderValidation
  : checkOrder
  : reject
  : CheckCustomer
  : acceptOrder

Process : OrderProcessing
  : doOrder
  : ProcessedOrder

CustBean : CustomerComponent
  : EnqStatus

Seller : Sells
  : SendOrder
  : GetDenied
  : GetConfirmation

Seller Composition

1: checkCustomer(order : Order)

Copyright © 2000-2003, Data Access Technologies, Inc.
ECA Methodology

A simple methodology for creating collaborative business processes
Basic Steps

- Identify roles and organize roles into collaborations
- Define collaboration documents
- Create basic business transactions
- Organize into protocols and events
- Use protocols to define ports on roles
- Drill-down into role detail
- Implement roles
- Configure implementations for deployment with technology specifics
- Deploy
Identifying roles and collaborations
Identify Documents

OrderConfirmation
  name 1.1
  Order 1.1
  Total 1.1
  IdString 1.1
  AddressInternational 1.1
    AddressLine1 0.1
    AddressLine2 0.1
    City 0.1
    State 0.1
    PostalCode 0.1
    Country 0.1
Distinguish protocols and events
Create Business Transactions
Organize into protocols
Add ports to complete community process
Drill-down
Add implementation

- As component compositions
- In a programming language
- By using an external service
Add technology specifics for deployment
WSEC

Web Services for Enterprise Collaboration

Initial Proposal
Main Parts

- Distributed Component Profile
  - How to define & structure distributed components

- Aspects
  - How to augment a model for a technology

- Aspects defined for WSDL/Schema
  - The augmentations required for Web Services

- Mapping
  - Transformations between ECA and WSDL/Schema
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
Aspects
Mapping of a WSDL Engine

- <definitions xmlns="http://schemas.xmlsoap.org/wsd
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime
xmlns:http="http://schemas.xmlsoap.org/wsdl/http
ENC="http://schemas.xmlsoap.org/soap/encoding/
xmns:xs2000="http://www.w3.org/1999/XMLSchema
 xmlns:xs2001="http://www.w3.org/2001/XMLSchema
targetNamespace="urn:SellerServer" xmlns:tns="urn:S
xmlns:CoreTypes="urn:CoreTypes" xmlns:Ordering="uri

definitions obtained from component /BuySell/Deployment/Seller...
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; )  -->
  `</port>`
  `</service>`
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" 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

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

- `<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>
Ways to map

- Generate code and other artifacts
  - Usually required to adapt to legacy
  - Emulates the manual process
- Assemble and configure existing generic and specific components
  - Both efficient and dynamic
- Interpret or otherwise “animate” the high-level model
  - High level & dynamic but may have performance issues
- Simulate
  - Evaluate technical and business impact through simulation
Example Generated Artifacts

Artifact generation involves multiple tools
- EJB Container provider; Deployment tools; Packagers;
- Java development tools (IDE); Persistence provider; ...

Typical 10-20 per PIM Classifier
0-20% manual override
Vision

Building and adapting systems for collaboration, reuse and change
Loosely coupled enterprise architecture

- Independent enterprise components
- Representing business concepts
- Link via open standards
Business Component Marketplace

- The business component marketplace is projected to be a $10b market in 5 years
- Consider the value of XML components that wrap popular legacy
- New application functionality built from components
- Components for integration and transformation
- XML and web services makes an excellent basis for such components
- Technology components, such as for repositories and DBMS
- Marketplace my be inside the enterprise or commercial
OMG Model Driven Architecture (MDA)

- High level – platform independent models
- Technology Models
- Mapping
  - Custom
  - Standard
- Standard Models produce technology specific standards artifacts
Automated MDA

Profile (E.G. EDOC)

Infrastructure Mapping (E.G. XML)

UML Design

Tools Produce & Integrate

Enterprise Components

Framework & Infrastructure (E.G. XML)

Mapping is tuned to the infrastructure

Copyright © 2000-2003, Data Access Technologies, Inc.
Technology Independence
High level tooling & infrastructure

- **MUST BE SIMPLE!**
  - We must be able to create better applications faster
  - We must separate the technology and business concerns, enable the user

- **Tooling + Infrastructure**
  - Executable models are source code
  - Tooling must be technology aware
  - Infrastructure must support tooling, not manual techniques

- Model based component architectures
High level tooling & infrastructure

- **MUST BE SIMPLE!**
  - We must be able to create better applications faster
  - We must separate the technology and business concerns

Executable Models
Iterative Development

Business Model Design

Automation

Infrastructure Development

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 asset that will outlive the technology of the day
Advanced EDOC

Platform Independent Models

Integration - viewpoints

CCA

Entities

Events

Process

Relationships

Patterns

Platform Specific Models

Technology mappings

Specific Technologies

FCM

MOM

ebXML

CORBA

COM

EJB

Copyright © 2000-2003, Data Access Technologies, Inc.
Business Process Profile

how things are coordinated
Business Processes

- Specialize CCA
- Activity-centric view of a Process
- Express
  - Complex temporal and data dependencies between business activities
  - Iteration of activities
  - Alternative required Inputs and Outputs of activities
  - Roles related to performers, artifacts and responsible parties for activities
Relationships Profile

Useful associations and dependencies
Relationships Profile

- Enables non-binary aggregations
- Defines useful association and dependency stereotypes

<<Aggregation>>
<<Packet>>
<<Assembly>>
<<Subordination>>
or
<<List>>

partA
aggregate
partB
Patterns Profile

reusing parameterised designs
Patterns Profile

- Profiles UML Parameterized Collaborations
- Based on Business Function Object Patterns (BFOP)
  - Multi-Layer
  - Based on Catalysis Approach
- Adds stereotypes for
  - Named Patterns
  - Inheritance
  - Composition
  - Pattern Binding with renaming
Platform Specific Modelling

- EJB, FCM, MOF
- Technology mappings from EDOC to Distributed Component and Message Flow Platform Specific Models
  - EDOC to J2EE/EJB mapping
  - EDOC to CORBA/CCM mapping
  - EDOC Business Process to FCM mapping
  - EDOC Business Process to CORBA mapping
Discussion

- How much are we using these kinds of concepts today?
- What would be the advantages to going in this direction?
- What are the barriers;
  - Technical?
  - Political?
- What would be a good first step?
Contact

Cory Casanave
Data Access Technologies
www.enterprise-component.com
cory-c@enterprise-component.com
(305) 234-7077