MDA Tool Support for SOI

Mike Rosen
CTO, AZORA Technologies, Inc.
Mike.Rosen@Azoratech.com
Mike Rosen

- Consultant
  - Chief Enterprise Architect for service and component based systems
    - Finance, Insurance, Telecom
  - SOA, EA, MDA implementation and training
  - 20 years experience in distributed applications

- Author
  - Cutter Consortium
    - “Service Oriented Integration – Aligning SOA with Enterprise Integration”
    - “Implementing SOA on Common Technologies”
  - Implementing SOA Applications, due 2005
Agenda

- Architecture Driven Development
- MBD and Enterprise Architecture
- DSL for SOI
- Demo
- Conclusion
SOI Implementation Example

Channels

Customer Management  Marketing  Pricing

ESB Business Service Bus

Account Service  Loan Service  Credit Service  Other Services

ESB Integration Service Bus

Bank 1 Account System  Bank 1 Loan System  Bank 1 Credit System  Bank 2 Account System  Bank 2 Loan System  Bank 2 Credit System

Data
SOI Application Architecture

channels

application

enterprise

resource

Service infrastructure

Management / Monitoring Utility

BAM

SLAs

Service Clients

... Logging Manage Config

Legacy System

Packaged Application

SOR
SOI Technical Architecture

channels

application

enterprise

resource

B2B

BPM / Enterprise Bus. Process

Business Service

Integration Service

Application Adapter

Portal

Process Session / Data

ODS DataMart DW

Data Integration Framework

Resource Adapter

BPM / Enterprise Bus. Process

Process Session / Data

ODS DataMart DW

Resource Adapter

Enterprise Service Bus

utility

infrastructure

Security Service

Profile Service

BPM Service

Audit Service

SLA Mgmt

Registry

xform Service

Transaction Service

legacy System

packaged application

SOR
Architecture-Driven Design

Enterprise Architecture

- Business Requirements
- Information Requirements
- Enterprise Requirements
- Program Requirements
- Technical Requirements
- Implementation Requirements
- Operational Requirements

Business Architecture

Information Architecture

Application Architecture

Technical Architecture

Implementation Architecture

Operational Architecture

Application Analysis and Design

Application Requirements

Deployment of Service

© Michael Rosen 2005
Architectural Style

- Architectural Style
  - A set of principles, elements, patterns, and constraints designed to meet a specific set of requirements within a specific scope
  - A style can be thought of as a set of constraints on an architecture – constraints on the component types and their patterns of interaction – and these constraints define a set or family of architectures that satisfy them.”

- Architectural style provides the “rules of engagement” when building an architecture. It is a set of patterns that provide guidance on the proper use of the different types of components that exist within your architecture.
  - The best way to convey an architectural style is via a formal model of it, referred to as a metamodel.

Example Architectural Styles

Architectural Views
- Business Architecture
- Application Architecture
- Technical Architecture

provides scope

Architectural Styles
- SOA / SOI
- Product, Service
- Enterprise Application, Product Line
- Stand-alone, Client-Server, 3-Tier, N-Tier
- Message Bus

provides patterns

‘ACME, Inc’ Architecture
- ACME SOA
- ACME Product Line Architecture
- ACME N-Tier Architecture

Provides scope which defines separation of concerns, responsibilities, stakeholders
Provides a common solution to a common set of requirements within a specific scope
Provides a specific solution to a specific set of requirements for a specific enterprise
Metamodels

- Provide rules for how to build a correct model for a particular purpose, e.g. “business integration metamodel”

- UML Profile
  - Provides a targeted subset of UML
  - Standard mechanism for extending UML

- Refinement and Constraint
  - Metamodels refine the definition of modeling elements by placing constraints on their behavior through the use of stereotypes

- Stereotypes
  - Standard UML Stereotypes
    - <<boundary>>, <<control>>, <<entity>>
  - Extending the UML Stereotypes
    - Inheritance used to extend and refine the meaning of stereotypes
    - Tagged Values use to apply specific properties
SOI Architectural Style Metamodel

“Rules of Engagement”

Metaclass Interactions
MDA Overview

- Business Analyst
- Architect / Designer
- Developer / Tester
- Business Model
- Application Model
- Implementation Model
- Code
MDA Under the Hood

- Computation Independent Model
- Platform Independent Model
- Platform Specific Model

Architectural Standards and Guidelines Enforced in Model Profiles

Enterprise QOS and non-functional requirements implemented in transformations

Code
SOI Architecture Profiles

Each Profile formally specifies a particular architecture.
MDA and Architecture-Driven Design

Enterprise Architecture

- Business Architecture
- Information Architecture
- Integration Architecture
- Application Architecture
- Technical Architecture
- Implementation Architecture
- Operational Architecture

MDA

- Application Analysis and Design
- Application Implementation
- Deployed Service

Requirements:
- Business Requirements
- Information Requirements
- Enterprise Requirements
- Program Requirements
- Technical Requirements
- Implementation Requirements
- Operational Requirements
Parallel Paths of Application and Platform

Application Architect

Application Design to meet Business Functionality

Implementation

Deployment

Technology Architect

Infrastructure Design to meet QOS Requirements
MBD Approaches

- Elaboration
  - Start with CIM or PIM
  - (Incorporate customizable architecture)
  - Perform gradual refinement, with human input to PSM
  - Generate structure and glue code

- Translation
  - Define detailed PIM using xUML profile
  - Add algorithmic logic with ALS (computationally complete)
  - Generate fully functional (non-modifiable) code

- Both are reasonable, useful approaches at using MDA standards, but are aimed at solving different problems
Domain Specific Languages

- Define the construction abstractions for building solutions in a particular domain, using the concepts, terminology and notation of that domain.

- Can be focused on higher level ‘business’ abstractions

- Must support common tasks and patterns

- Must have a well defined set of rules, called a grammar, governing the way the concepts can be combined to form expressions

- The meaning of well formed expression must be well defined
  - Users can build models that other users understand
  - Tools can validate and generate implementations from models
  - Metadata captured by models supports tasks like configuration

- Described in a formal metamodel

- Might be built with a DSL
ADD, MBD, DSL

- “An architectural style is a coarse grained pattern for a family of systems. It defines the kinds of components and relationships that can be used in their assembly, constraints on the way they are assembled, and assumptions about the meaning of an assembly.

- Architecture Driven Design uses architectural styles as a framework for guiding design decisions

- Model Based Development uses models as first class source artifacts in the development process. The models are higher level input to ‘machines’ that generate other implementation artifacts.

- Domain Specific Languages provide ‘easy to use’ modeling abstractions and tools, targeted at a specific domain, which conform to an architectural style, for use during MBD.

- ADD, MBD, DSL…a winning combination!
Benefits of MBD

- Improves tracing between business and technology models
- Separates business and technical modeling roles
- Leverages the knowledge of experts to make all developers more productive
- Promotes abstraction leading to better, more comprehensible designs
- Improves portability to other platforms
- Eases modifications to either the business or technology
But, How do you get there?

Requirements
Platform
Tools
Methodology

Architect/Analyst Designer
Let’s see...

- **Generation of Artifacts**
  - Starting with the Application Design Model…
  - Validate the correctness of the model
  - Generate the necessary platform code to support the design
  - Understand how the reference architecture supports the design and code generation

- **Implementation Guidance**
  - What if you don’t know how to create a design model?
  - Step by step instructions for specifying design model
  - Overall support for development process and architecture
Application Design Model

Architecture Profiles Influence design

Channels
Customer Management Marketing Pricing

ESB Business Service Bus
Account Service Loan Service Credit Service Other Service

ESB Integration Service Bus
Himawari Himawari Himawari PBJ PBJ PBJ

© Michael Rosen 2005
Generating to an ESB Platform
Code Generation

AZORA Reference Architecture

Graphical Modeling Environment

Provides Reference architecture

Business Architecture

Application Architecture

Platform Architecture

AZORA Reference Architecture

Model verification ensures compliance

Design model specifies business services, not ESB constructs

Create SOI analysis and design models

Generate code

Structure generated

100% generated

COTS

Business Logic

‘Glue’ code

ESB Platform

Application Software Layers

Generate code

Add business logic in “protected sections”

© Michael Rosen 2005
AZORA Methodology

- Phases
- Activities
- Work Products
- Steps
- Roles
- Guidance

Step-by-step process instructions

AZORA Reference Architecture

- Business Architecture
- Application Architecture
- Platform Architecture

Architecture Details

Browser Based Knowledge Management
SOI Development Suite

1. Business Analysis Phase
2. Architecture Phase
3. Analysis and Design Phase
4. Implementation Phase
5. Quality Assurance Phase
6. Accept and Deploy Phase

AZORA Methodology

© Michael Rosen 2005
Questions