BPM Implementation with SOA and MDA

Marc J. Balcer
VP, Product Development
AZORA™ Technologies
Sunnyvale, California
Promise

Business Process Model

Existing Systems and Data

How do you get from here to there?
It doesn’t scale
The ultimate result?

Gartner: “Enterprise Application Spaghetti”
Solution Approach

- Use “right kinds” of services for building BPMs
- Organize the complete set of services to be usable in our context
- Use MDA to realize the design
Partitioning Principles

• Business process models should be in the language of the business
• The steps in the BPMs should map to services that provide equivalent business functionality
• Business services should be built atop layers of services that reach down to implementation technologies
• A domain is a separate real, hypothetical, or abstract world inhabited by a distinct set of objects, all subject to the same rules and policies.
Domains and Objects

- Each domain has its own independent object model
  - classes
  - attributes
  - associations
  - specializations
Domains and Services

- Domains expose their capabilities as services
  - operations
  - types
  - documents

Document definition
Bridges

- A bridge is a contract between domains that states how one domain makes use of services provided by another.

The bookstore uses Package Shipping to send completed orders to customers.
Domains as Layers

- **Business**
  - the actual business

- **Utility**
  - generic capabilities

- **Integration**
  - combine operational resources

- **Operational**
  - existing systems

- **Foundation**
  - provided by the core software architecture
Layered SOA Architecture

Interfaces defined by enterprise model

Interfaces defined by enterprise semantics and requirements

Business Processes

task definition

task implementation

Business Services

Integration Services

Operational Resources

IS

IS

IS

IS

IS

IS

IS

Mainframes

Servers

Data

Mainframes

Servers

Data
Use Cases and Scenarios

- Use cases define the context in which the domain will be used
- Individual scenario steps map to business services
- Messages carried between the steps map to business documents
Model Interconnections

Diagram:
- Use Case
- Scenario
- Step
- Actor
- Operation
- Service
- Class
- Type
- Attribute
- State Model
- Document

AZORA
Integrating Existing Systems

Existing systems can be enabled as services

Different existing systems provide capabilities to realize a domain

The integration service combines the existing systems’ capabilities

One existing system provides all the capabilities needed to realize a domain
Combining Multiple Systems

• Suppose we need to create a single customer view
  – billing
  – provisioning
  – trouble ticketing

• Hint:
  it’s not just a UI problem
Unified Enterprise Model

- Create one model of the combined domain
- Then map elements of the combined domain down to the existing capabilities
Enterprise Organizations

- Organize a business domain by the layers in the business
  - whole enterprise
  - lines in the business
  - industry standards

- Provides scope, visibility, and governance
What’s the Problem?

Analysts create application models in a UML modeling tool…

…print them out…

… then the developers look at the models and write the production code.
What’s the Problem!

But once something is in a machine representation…

…we shouldn’t have to manually reinterpret it ("re-code it") into another.

The analyst’s diagrams don’t have to be complete, correct, or consistent models.

Keeping the diagrams in sync with the code is a daunting task.

The re-coding introduces development errors and obscures analyst errors.
Software Architecture

• Basic design policy
• Conceptual Architecture
  – Platform-independent representation
  – “Character” of the architecture
• Platform Architecture
  – Platform-specific representation
  – Select technologies
  – Define actual artifacts
• Once conceptual architecture can have many platform architectures
“Platform” is relative

Platform Independent Model

Platform Specific Model 1
- .Net® Web Services
- J2EE Web Services

Platform Specific Model 2
- J2EE with JMS
- .Net with message queues

CIM
PIM
PSM

AZORA
One Technology Platform
Example

Information Model
- classes
- attributes
- associations

Relational Schema
- tables
- columns
- primary keys
- foreign keys

SQL Create Table Script

Model element
(from the SORA)

Conceptual entity

Platform artifact

AZORA
Additional Information

Information Model
- classes
- attributes
- associations

Relational Schema
- tables
- columns
- primary keys
- foreign keys

SQL Create Table Script

Type Model
- domain-specific type

Type Mark
- Map domain-specific to platform type
SOA elements defined in metamodel

Marks
(additional data needed to translate)

Platform Elements

Message Class

XML Schema
Automation Overview

Graphical Modeling Environment

Provides Reference Architecture

Create Analysis and Design Models

Reference Architecture

Model Verification Ensures Compliance

Design Model Specifies Business Services, not Platform Constructs

Generate Artifacts

Business Logic

‘Glue’ code

Integration Platform

Application Software Layers

Business Architecture

Application Architecture

Platform Architecture

Provides Reference Architecture

Create Analysis and Design Models

Generate Artifacts

Structure Generated

100% Generated

COTS

AZORA.
Summary

• Three dimensions of layering
  – subject matter
  – organization
  – modeled / realized

• SOA Reference Architecture

• Translation
  – ensures platform independence
  – improves development performance
Marc J. Balcer
VP, Product Development
AZORA™ Technologies
Sunnyvale, California
Marc.Balcer@azoratech.com
http://www.azoratech.com