MDA - The Real Value

Oliver Sims
Sims Associates
oliver.sims@simsassociates.co.uk
Background

- Consultant
  - Enterprise systems, component-based development, distributed system software architectures,
  - Transitioning to affective CBD

- Chief Architect
  - Component (app server) middleware, 1990s
  - OMG Architecture Board member

- Author
  - *Business Component Factory* (2000, with Peter Herzum)
  - *Building Business Objects* (1998, with Peter Eeles)
  - *Business Objects* (1994)

- Systems Engineer (IBM 1969-1993)
  - UK large complex systems
  - Specialist, distributed systems, programming technologies, communications, system design, manufacturing industry, etc.
Agenda

- MDA positioning
- MDA essentials
- MDA – the envelope
  - Product Line
  - Architecture
- MDA – the real value
- Next Steps
MDA Positioning (1)

- Application development is lengthy and complex
  - Software technology considerations
  - Business requirements become lost
  - Structural design re-invented
  - Architectural concerns often not separated
  - Integration demands often hugely complicated

- Many people have “done MDA” in the past

- MDA is the focus for resolution of the development crisis
  - A major strategy … from a major standards organization
  - Addresses a key architectural separation of concerns
  - Provides single conceptual framework and vocabulary
  - The focus for synergy between key resolution enablers…

* Systems Engineering Institute
http://www.sei.cmu.edu/plp/plp_init.html
MDA Positioning (2)

- Technology churn
- Integration
- Skills
- Legacy Systems
- App Development
- New application areas

IT Complexity

Architectures, Architectural Styles
Product Line* approach
CBD
Traceability Business->Code
EAI

* Systems Engineering Institute
http://www.sei.cmu.edu/plp/plp_init.html
Traditional Development

Traditional Development Projects

Tools  
SCM  
Patterns  
Glue Code  
Architecture

Business Function

Dev. Process

Traditional IT Infrastructure

User Support  
ORB  
DBMS  
Issue Management

Traditional Office Infrastructure

© Copyright Sims Associates 2002 - Slide 6
“Product Line” Development

Business Function Projects

Product Line Infrastructure

Traditional Office Infrastructure

© Copyright Sims Associates 2002 - Slide 7
• MDA positioning

• MDA essentials

• MDA – the envelope
  – Product Line
  – Architecture

• MDA – the real value

• Next Steps
MDA Essentials

Service Provisioning

- Business Model
- Analysis Model
- Design Model

Billing

- Business Model
- Analysis Model
- Design Model

Shared Mapping

Platform-Independent Model (CIM/PIM)
Platform-Independent Model (PIM) (Computation-Dependent)
Platform-Specific Model (PSM)

Code

Platform

© Copyright Sims Associates 2002 - Slide 9
The Importance of Context

- Business Model
- Analysis Model
- EJB Design Model
- App Server Model
- Operating System Model
- Microcode Model
- Hardware Model
- Chip Model
- Silicon Block Model
- Shipping Model
- Excavator Model
- Mining Model

Business System Developer

Middleware Developer

PIM

PSM
A “Mapping” is a set of (automated) rules & techniques

Models must be “marked up” in some way to help direct mapping

Add code – needs round-trip support

© Copyright Sims Associates 2002 - Slide 11
MDA Example (1)
MDA Example (2)

- Target run-time environment:

  - Rich Client
  - Browser
  - Web Server
  - CORBA
  - HTTP
  - App Server
  - MQ
  - DBMS

Below the line are the things provided for the developer.
MDA Example (3)

Billing
Business CIM

Billing
Analysis PIM

Billing
GUI PSM
Billing
HTTP PSM
Billing
COM PSM
Billing
Servlet PSM
Billing
CORBA PSM
Billing
J2EE PSM
Billing
MQ PSM
Billing
DB PSM

Glue-1
Glue-2
Glue-3
Glue-4

GUI
HTTP
COM
Servlet
CORBA
EJB
MQ
SQL

Rich
Client
Browser
HTTP
Web Server
CORBA
App Server
Messaging
DBMS

Mark PIM to help mapping to platform technology

Add code – (needs round-trip support)

Add code – (needs round-trip support)

Add code – (needs round-trip support)

Mark PIM to help mapping to platform technology

Add code – (needs round-trip support)

Add code – (needs round-trip support)

Add code – (needs round-trip support)

Add code – (needs round-trip support)
Agenda

- MDA positioning
- MDA essentials
- MDA – the envelope
  - Product Line
  - Architecture
- MDA – the real value
- Next Steps
MDA + Product Line

Distribution Tiers Architecture
- User Interface (UI)
- User Services (US)
- Business Service (BS)
- Business Resource (BR)

Model structured in tiers. Mark PIM to define structure (modularisation)

Add code – Needs round-trip support

Rich Client
Browser
HTTP
Web Server
CORBA
App Server
Messaging
DBMS
“When an organization with a single project/product focus moves from CMM level 3 to level 4, the productivity gain is minimal because most of the improvements that can affect a single product or project will already have happened at level 3. …

“However, when the process improvement from level 3 to level 4 includes a shift to product line focus, the productivity increase is very significant. Vu’s data indicate as much as a 70% productivity improvement, as well as highly satisfied employees.”

Computational Completeness

• A model that can be executed - via code generation or interpretation - is said to be “computationally complete”

• Requires:
  – Action Language for algorithmic logic
  – Computational structure

• The aim is to build computationally complete PIMs
  – All development at the model level
  – Execute the model to test
  – Generate code where necessary

• Component architecture provides an excellent structure
What is a “Component”? 

• A managed executable – runs in a middleware “container”
• A pluggable artifact
  – Through the development lifecycle
    • A managed module/package
    – Built with non-component artifacts
• Has programmatic “interface”
  – Local/Remote transparency
• Designed to represent a single “business” concept
  – Throughout the development life cycle
• Composable with other components
  – “Autonomous” not isolated – composed by reference
  – Defined granularities
  – Defined ownership
• Conforms with OO concepts
  – Encapsulation, instantiation, state + behavior
  – Inheritance
    • (where technically possible)
• Can address distributed systems end-to-end
Product Line + CBD Architecture

- Architecture (for a given architectural style):
  - Conceptual models that define:
    - Component-oriented application structure concepts
      - Including component granularities and re-usable artifact types
      - Superior modularisation strategy
    - Scalability patterns
    - Separation of distributed system (and other) concerns
    - The product line for this architectural style, including design of technology/business separation
  - Development process
  - Skeleton structure for models
  - Mappings (and traceability) between models
    - Plus UML profile for models
  - Design for glue code
  - etc.
  *(there are a number of published examples)*

- Result:
  - Design for component-oriented application structure
  - Design for a product line

- Evolve the new product line
  - A new level of productivity

- **Re-use the product line** for all applications of the same architectural style

- Repeat for each architectural style
  - Re-using common architectural elements

The “program” for the MDA tool
MDA + Product Line + Architecture

Application Development

Billing
Business CIM

Billing
Component Analysis PIM
(computationally complete)

MDA Tool

Billing
Component Code

Product Line

CIM-PIM Generator

PIM Interpreter

Code Generator

CIM⇒PIM Mapping

PIM⇒Code Mapping

Component Technology PSM

Architecture
Conceptual Models

© Copyright Sims Associates 2002 - Slide 21
The Perfect MDA Tool (2002)

- Current “standard” capabilities
  - UML modeling
  - Models (or parts of models) can be web-published
  - Code generators for major (and some other) platforms
  - XMI model interchange
- Action Language and/or OCL
- Integrated IDE
- Integration of modules at the PIM level
  - Re-use previously-built PIM modules – e.g. in CBD
- Integration with EIA tools
- CIM/PIM/PSM differentiation available
- Reverse engineering
  - Round-trip engineering
- Executable models
  - in run-time as well as in development
- Architecture support:
  - Metamodel approach, so new UML and other metamodels can be developed and “plugged in”
    - Using MDA of course!
  - Pluggable PIM architectures
    - E.g. CBD architecture
  - Support for GUI and Data specification consistent with architecture
  - Pluggable generators and mappings if necessary
  - Pluggable “glue” code consistent with architectures supported
- Tool designed using MDA for fast evolution
- Scalable to large development teams
  - Repository support, versioning, sharing, revision-marking, etc.
- Excellent user interaction design
  - E.g. following Alan Cooper’s guidelines (see “The Inmates are Running the Asylum”)
- ???
CBD Architecture support
Configuration and Code Generation

ArcStyler Configuration

- C-GEN - Projections

Chosen Technology Projections:
- S:\carat\carat\gen\cartridges\carat.tpr
- S:\carat\java2\gen\cartridges\Java2.tpr

- Use precompiled templates: [ ] Yes [ ] No
- Cartridge source code directory: [ ] java2_gen
- Cartridge components working directory: [ ] java2
- Generate depending physical components: [ ] Yes [ ] No

Project Configuration: E:\catools\carat\java2\model\Java2.asprj
Domain Configuration: C:\ide\ArcStyler\cfg\asrose.cfg
Reverse Engineering

- Hierarchical Action Log
- Detailed Exploration
- Referenced Java Framework
- “Compact“ EJB Components
- Configuration
- “Compact“ EJB Components
- Referenced Java Framework
- Configuration
- Hierarchical Action Log
IDE Integration
Action Language

HRI Access Management v2

Reason For Access

Logged In User

Pat has been denied

Patient Data Viewed

Deleting User

Viewing Patient Data

Selecting Application

Database: NHSTA
Domain: HRI Access Management, HRI
Version: v2, HRI Version 2
Class: Role (R)
Operation: 2, LinkRoleToTypeOfAccess

Visibility: FALSE
Scope: Class
Referrer: R

Description

Contract Type: Closed Blocking
Contract Description

Input Parameters

Method
myRole = find-one Role where roleName = theRoleName
if myRole = UNDEFINED then
    roleFound = FALSE
else
    roleFound = TRUE
    myTypeOfAccess = find-one Type Of Accessor where accessorTypeName = theAccessorsTypeName
    if myTypeOfAccess = UNDEFINED then
        fail theAccessorsTypeName
    endif
    if myTypeOfAccess.roleType = myRole
       _transactionComplete(user)
    endif
    link myRole R3 myTypeOfAccess
endif

Exception Handling Code

Attached Tags

Codec Block

© Copyright Sims Associates 2002 - Slide 27
Model execution
Agenda

• MDA positioning
• MDA essentials
• MDA – the envelope
  – Product Line
  – Architecture
• MDA – the real value
• Next Steps
MDA – The Real Value

- MDA possibilities will force middleware upwards (in simplicity and capability)
  - IT organisations will demand higher-level middleware
    - Because IP can be moved up from code to PIMs
    - Computationally-complete PIMs
    - Easier integration at the PIM level
    - Better management of technical churn
    - Recover PSMs and PIMs from legacy
  - Suppliers will start delivering standard “glue” code
    - E.g. concurrency, transactions, client proxy mechanism, client-side frameworks, etc.
    - The “platform” will become more productive, more developer-friendly

- MDA possibilities will drive tools upwards
  - IT organisations will demand higher-function tools
    - Along lines of “perfect MDA tools”
  - Suppliers will provide standard architectures (probably component-based)
    - CBD architecture a priority
    - Architectures will grow in scope and applicability

- Result …
Managing Complexity

Consistent, Enterprise Class, Product Line Infrastructure(s)
“Naked Objects”

• “‘Naked objects’ are core business objects, such as Customer, Product, and Order, that show directly through to the user, rather than being hidden behind the menus, forms, process-scripts and dialogue boxes that make up most user interfaces.” (www.nakedobjects.org)

• Used to build working prototypes while business modelling…

• … and then used to implement the system!

• GUI is automatically-generated
Naked Objects Example (1)
Naked Objects Example (2)
Naked Objects Example (3)
Possible MDA Evolution

**Today**

- MDA And MDA Tools
- “Naked Objects” approach
- Platforms

**Future**

- Application PIM
- Future Platforms
- Architecture, CBD, Product Line, etc.
- Future MDA tool
- Application GUI
- UML Notation
- Other Views

© Copyright Sims Associates 2002 - Slide 36
Future MDA Development Process?

**Task**
- Model the business
- Develop the application
- Technical validation & handover

**Deliverable**
- Working Prototype PIM
- Working Application PIM
- Deployed Application (PIM?)

**View 1**
- Prototype GUI
- Application GUI
- Application GUI

**View 2**
- UML CIM
- UML PIM
- UML PIM

© Copyright Sims Associates 2002 - Slide 37
Agenda

• MDA positioning
• MDA essentials
• MDA – the envelope
  – Product Line
  – Architecture
• MDA – the real value
• Next Steps
Next Steps

- Tomorrow:
  - Schedule an MDA planning meeting

- Next week:
  - Review the current development situation
  - Define the goal
  - Plan the first step
  - Define process for evolution to the goal
    while continuing with development schedule

- Next month – Start!
  - Launch the first MDA project, using an MDA tool
    - Real application
    - Architecture capture
    - Product Line initiation
  - Manage the evolution to the goal

- 😊 … call us if you need help!
MDA

- The Real Value

Oliver Sims
Sims Associates
oliver.sims@simsassociates.co.uk