Realizing the Benefits of MDA

OptimalJ by Compuware
May 2003 OMG Workshop

Chris Sirosky
Product Manager - OptimalJ
Business Challenges
In Java/J2EE application development

• Rapidly respond to business change – Time to market
• Increase developer productivity – “Do more with less”
• Enforce the use of best practices, standards and guidelines
• Stay current with new versions of technology
• Leverage existing investments
• Maximize application quality and reliability – Minimize risk
Object Management Group
Model-Driven Architecture

- A new way to specify and build systems
  - Based on modeling with UML
  - Modeling instead of programming

“MDA is about moving to the NEXT level – instead of offering another set of API’s, it offers an opportunity to model those API’s generically – and then generate code for accessing the API’s you choose (or get stuck with). More importantly from the view of the app developer, you model “your application” generically, ignoring the plethora of APIs”
Richard Soley

Finance
E-Commerce
Health Care
Transportation
Space
Manufacturing

More...
Closing the Gap
The need for Models and Patterns

Complex business process

Model
Develop
Test
Deploy

Models

Reducing business complexity

Patterns

Reducing technology complexity

Complex technology platform (J2EE)
A New Development Paradigm
Model-driven pattern-based

- Companies that want to maintain or increase their future competitive edge will need to begin evaluating, planning for and migrating development staff to at least one of the two alternative and more efficient forms of development, model-driven pattern-based (MDPB) or component assembly and orchestration (CAO).

- Organizations using a model-driven or pattern-based application development framework containing a large inventory of business components have the potential to be five to ten times more productive and responsive than those that do not
OptimalJ
Development & Integration with Patterns & Models

Integration to:
• Reduce integration complexity
• Accelerate integration
• Promote reuse

Models to:
• Reduce business complexity
• Rapidly respond to change
• Ensure reuse

Patterns to:
• Reduce technology complexity
• Accelerate development
• Enforce standardization

Development to:
• Improve customization
• Allow personalization
Model-driven Development
How OptimalJ maps to MDA

- Platform Independent Model
- Platform Specific Model
- Code Model

Transformation Patterns
- Technology Patterns
- Implementation Patterns

Domain Model
- Domain Patterns

Application Model
- Application Patterns

Code Model
- Code Patterns

Generate

Compuware Corporation
Visual Development Paradigm
Models and diagrams

- Domain Model diagram
  - Created by the designer
  - Imported from a modeling tool (UML/XMI)
  - Uploaded from an existing database

- Application models
  - Generated automatically
  - Improve application navigation
  - Improve application overview and understanding

- Code Model
  - One-to-one mapping to Java code
  - Visualization of the code
  - Reengineering supported
OptimalJ
Pattern-driven Application Generation

- Rapidly generate a J2EE application from a visual model
- Produces consistent, high-quality, reliable Java code
- Automates tedious, repetitive development tasks
- Patterns shield the designers from the complexity of the J2EE platform
OptimalJ Patterns
Transformation & Functional Patterns

- Transformation Patterns – Automation
  - Reduce complexity
  - Automatically transform high level model into lower level model.
    - Technology Patterns
    - Implementation Patterns

- Functional Patterns – Reuse
  - Accelerate development
  - Reuse best practices or implement standards
    - Domain Patterns
    - Application Patterns
    - Code Patterns (GoF)
Implementation Patterns
Pattern Editor

- Experienced J2EE architects can add, update and delete Implementation Patterns
- Allows architects to implement their internal coding standards
- Ensures designers/developers implement the standards
- Reduces the need for code reviews and training
- Automatically transfers expert knowledge to designers/developers
Pluggable Pattern Architecture
For example upgrading from EJB 1.1 to EJB 2.0

- OptimalJ’s pluggable pattern architecture
  - Install OptimalJ’s EJB 2.0 pattern
  - Regenerate the application

- Automatically implemented EJB 2.0 features
  - EJB 2.0 attribute declarations
  - Container Manager Relationships (CMR)
  - EJB Query Language (EQL) on Finder Methods
  - Deployment descriptors
  - Custom code preserved
Application Integration
Java, .NET and legacy Cobol

- Integrate with existing Java, .NET and legacy applications
- Import WSDL, IDL, Cobol Copybooks or Commarea and Java signatures
- Automatically generate Web Services
- Automatically generate JCA compliant stubs and skeletons
  - Integrate with existing JCA compliant resource adapters such as IBM’s CICS
OptimalJ from Compuware
Meeting today's Business Challenges

OptimalJ accelerates application delivery by simplifying Java development through the implementation of OMG’s Model-Driven Architecture (MDA), enabling a team of architects, designers and developers to rapidly produce reliable J2EE business applications.

OptimalJ is a Java development environment that generates working applications directly from visual models, leveraging patterns to implement best practices for coding to J2EE the specs.
public AccountRemoteViewListImpl(Collection base) {
    super(base);
}