Bringing MDA to Eclipse, using a pragmatic approach

Bastiaan Schönhage
One day I was thinking ...

- Is coding really the best to develop?
- Why is development sometimes so boring?
- Why are other coders so slow?
- Why do other guys always write such crappy code?
- A football team works best with 11 guys
  - so why is pair programming so popular?
- Is “new technology” always better?
  - Of course, but why?
Ingredients for this presentation

- **Eclipse**
  - Code centric software development
  - From a Java IDE to a platform
- **Model Driven Development**
  - Software development paradigms
  - Model transformations and code generation
- **Pragmatism**
  - Beautiful solutions are only beautiful when they *work beautifully*
What can you expect?

- Brief history of Eclipse
- Software development paradigms
- Model-Driven Architecture / Development
- MDA & Eclipse: a happy marriage?
- Examples / Demos
- Conclusions & Future
Very brief history of Eclipse

- First there was Visual Age Micro Edition
- Then Eclipse 1.0
- Then Eclipse 2.0
- Then Eclipse 3.0
- And now we have Callisto
Or without version numbers:

- In essence you can also look at it in this way:
  - J2ME
  - J2SE
  - J2SE with refactoring support
  - More focus on platform and external projects
  - 10 projects are bundled with Eclipse into Callisto: focus on J2EE, web tooling, EMF etc ...

- The next step: MDA?
Software Development Paradigms

- It looks like the world is divided in two:
  1. The agile, lightweight, extreme, test-driven, hardly any process, no-bullshit, hip, brain-aided software engineering (*what most people want*)
  2. The heavy-weight, fully-documented, requirements driven, non-creative, boring old-fashioned software engineering (*what most people actually do*)

- People tend to
  - Put code-centric development using Eclipse in (1)
  - Put Model-Driven Development or MDA in (2)

- WHY?
Fairy tale MDA promise:

- **MDA!!**
  - Model application & services completely in UML
  - Push some buttons in our MDA tool
  - Presto-magic, your business app is done!
We no longer believe in miracles ...

- Critics:
  - OMG’s MDA: “you can model everything in UML and generate a complete application.”
  - Will UML will ever be able to deliver on this complete promise.
  - Does MDA require too much skill?
  - Can we really share models?
  - Can MDA handle integration with legacy apps?
  - Is it worth the bother to do all this modeling?
  - Isn’t MDA just another go at the CASE Monster?

- Indeed, UML/MDA is neither perfect nor a silver bullet
- However, pragmatic MDA is a very real solution!
So, what is pragmatic MDA?

- **MDA : Model Driven Architecture**
  - Models and Meta-models
  - Model-to-Model transformations
  - Code Generation
  - Better: text Generation
    - Ant build scripts
    - Deployment descriptors
    - Documentation
    - Etc.

- **Pragmatic**
  - Use an agile process
  - Model ➔ Generate ➔ Adapt ➔ Test (very small loops)
MDA: Models & Meta-models

MOF Model
Classifier attributes

UML Model
UML Class
fields
operations

Car Model
Car Class
description : String
color : Color
power (bhp) : Integer

Data
Car
BMW 650i Black
367 (bhp)
How to use MDA?

**Data**
- BMW 650i Black 367 (bhp)

**UML Class**
- description : String
- color : Color
- power (bhp) : Integer

**Classifier**
- **attributes**

**Code Generation**

```java
public class Car {
    private String description;
    private Color color;
    private int power;

    public Color getColor() {
        return color;
    }

    public void setColor(Color color) {
        this.color = color;
    }

    Car car = new Car();
    car.setDescription("BMW 650i");
    car.setColor(Color.BLACK);
    car.setPower(367);
}
```
Back to the main topic:

- **Bringing MDA to Eclipse**
  - Offering MDA support on the Eclipse Platform

- **Required:**
  - Modeling environment
  - Integration with existing views (trees, properties)
  - Good integration with coding capabilities
  - Easy Model to Code navigation
MDA and Eclipse

- Offerings:
  - Generative Modeling Tools (GMT) & openArchitectureWare
  - NEW: Eclipse Modeling Project
    - Eclipse top-level project
    - Contributions from Borland, IBM and Compuware
    - Only just started but contains EMF, GMF and GMT
  - Compuware’s OptimalJ
  - Borland Together Architect

www.ddj.com/dept/architect/184415500
The commercial side of Eclipse

- Eclipse and Callisto are open-source but not so are:
  - BEA Workshop
  - XMLBuddy Pro
  - Borland Together
  - Compuware’s OptimalJ
  - Canoo
  - MyEclipse
  - Rational Application Developer / Software Architect
  - Omondo EclipseUML2 studio

- The Eclipse Foundation maintains good balance between opens-source and commercial software
Bringing MDA to Eclipse - examples

- Enough said, let’s have a look at some real things
  - Example 1: MDA with MDA
  - Example 2: Generating Swing wizards
  - Example 3: Developing Business Applications using pre-defined patterns.
MDA with MDA:

- Developing parts of OptimalJ with MDA

- Eclipse integration
- NetBeans integration
- Shared GUI
- Core OptimalJ functionality
- Model repository
- Model transformation engine
- Code generators
The MDA way

Integration Model

Code generation

Eclipse integration

NetBeans integration
Preferences, the MDA way

MOF Classifier
attributes

MOF Model

MOF Classifier
attributes

UML Class
fields
operations

Settings Meta-Model

Settings Class
properties

Car Class
description : String
color : Color
power (bhp) : Integer

OptimalJ
Settings Model

Generic Diagram Settings
Text Color : Color
Line Color : Color
Shadow : boolean

Data

Current Settings
Black (text color)
Black (line color)
true (shadows)

Car
BMW 650i
Black
367 (bhp)
Preferences: the meta-model

- **Element**
  - name: String
  - annotation: String
  - getPersistencyKind(): Integer
  - getDocumentation(): String
  - setDocumentation(): void

- **Container**
- **Type**
  - typeName: String
  - primitive: Boolean

- **FolderItem**
- **Setting**
  - serialVersionUID: String
  - ideSpecific: String

- **SingleSetting**
  - global: Boolean
  - genericPath: String
  - idePath: String

- **MultiSetting**
  - genericPath: String
  - idePath: String

- **MultiSettingSpecialization**

- **Enumeration**
  - genericPath: String
  - idePath: String

- **TypedElement**

- **Property**
  - initialValue: String
  - propertyEditorClass: String
  - readOnly: Boolean
  - expert: Boolean
  - hidden: Boolean
  - ideSpecific: String
  - affectsIP: Boolean
  - affectsTP: Boolean
Preferences: the model

IDE Integration Model

Properties of textColor

- Properties
  - annotation: null
  - container: XeonDiagram
  - expert: false
  - hidden: false
  - initialValue: new java.awt.Color(0,0,0)
  - metaName: Property
  - name: textColor
  - propertyEditorClass: null
  - readOnly: false
  - type: Xeon.color

- Expert
  - affectsIP: false
  - affectsTP: false
  - ideSpecific: false
  - XMIPFileName: XeonPlainModule

textContent

Property
/**
 * Eclipse preference page for `<module.name>`
 */

public class `<UCF(setting.name)>`PreferencePage extends PreferencePage
  implements IWorkbenchPreferencePage {

  private Composite mainPanel;

  protected Control createContents(Composite parent) {
    mainPanel = new Composite(result, SWT.NONE);
    ...
    Composite panel = new Composite(mainPanel, SWT.NONE);
    ...
    `<BLOCK("createContents")></BLOCK>"
    ...
  }

  `<CREATEWIDGETS(imports, module, setting)>`
}
<TEMPLATE PUBLIC CREATEWIDGETS>(Set imports, Module module, Setting setting)>
  <FOR(Property property IN setting.part)>
    <IF (Util.hasBooleanType(property))>
      <BOOLEANWIDGET>(imports, setting, property)>
    </IF>
    <ELSEIF (property.type.name.equals("color"))>
      <COLORWIDGET>(imports, setting, property)>
    </ELSEIF>
    ...
  </FOR>
</FOR>
</TEMPLATE>
Example II: generating Swing wizards

- Investigation into generation of wizards
- Wizard model
- Code generation templates
- Run-time library
Example III: Business Applications

- Business applications have a lot common features
  - Database
  - Application server
  - Front-end

- But technology is changing very rapidly ... 
  - EJB 1.x / 2.x and now 3.0
  - Hibernate, TopLink, Spring, ...
  - Struts, JSF, Tiles, Ajax, ...

- Can MDA help here as well?
Developing Business Apps

- What if somebody would have already developed a code-generator for:
  - EJB
  - Struts
  - Hibernate
- Developing Business Applications with pre-defined patterns.
- This can increase productivity a lot.
- Commercial tools such as OptimalJ offer this
Conclusions

- **Eclipse**: from Java IDE to technology platform
- **MDA**: from academic exercise to pragmatic development approach
- **Eclipse & MDA:**
  - Commercial tools
  - Open source initiative: Eclipse Modeling Project
Future & more information

- Dr Dobb’s article on MDA tools:
  - www.ddj.com/dept/architect/184415500

- Keep an eye on the Eclipse Modeling Project
  - www.eclipse.org/modeling/

- OptimalJ Architecture Edition (the flagship)
  - Available on Eclipse soon
  - Tell your friends that you have already seen it ;-)