MODEL DRIVEN ARCHITECTURE (MDA)

Sacevski Igor
Veseli Jadranka
Introduction

- Object Management Group (OMG)
- CORBA (Common Object Request Broker Architecture)
- UML (Unified Modeling Language)
- MOF (Meta Object Facility)
- XMI (XML Metadata Interchange)
Why Models?

• Easier to understand
• Allows simpler representation of complex system
• Higher levels of abstraction
• Today’s standard is UML
Goals of MDA

• Speeding up the development process
• Increasing the quality of software
• Central error correction
• Increasing the re-use
• Reducing complexity through abstraction
Advantages of the MDA

• Reduces the complexity of a system
• Allows easy adaptation to new technologies
• Standardization allows integration between different tools
• Simple models allow easier communication with the customer
• Automatic code generation
Disadvantages of the MDA

• Manual changes in code
• Manual changes in lower level models
• How optimized will the generated code be
• Still unresolved standard issues
• Increased complexity of the modeling languages
Models in the MDA

- Computation Independent Model (CIM)
- Platform Independent Model (PIM)
- Platform Specific Model (PSM)
Basic principles

• First create a CIM of the system
• Then create a PIM from the CIM
• Define ways to transform the PIM into PSMs
• Use the PSM to generate a source code
Transformations in MDA

- Two basic ways to transform a PIM into a PSM
  - Model Type Mapping
  - Model Instance Mapping
- Combination of the two can also be used
- Additional information can be added
Model Type Mapping

- PIM
  - Platform Independent Metamodel
  - Language used

- Transformation
  - Source language

- Transformation Specification
  - Target language

- PSM
  - Platform Specific Metamodel
  - Language used
Model Instance Mapping

PIM

marked PIM

Transformation

Marks

Mapping

PSM

Platform
Mapping with additional information
Record of Transformation

• A result of the transformation process
• Very important role in MDA
• Allows round-tripping around models
• Provides traceability
Languages and standards

• UML 2
  – Complete functionality of UML 1.x
  – Increased semantic precision
  – Conceptual clarity
  – Represents a suitable foundation for MDA
  – Support for Extensions (Profiles)
UML 2 standard profiles

- UML Profile for Software Radio
- UML Profile for CORBA® and CORBA® Component Model (CCM)
- UML Profile for Enterprise Application Integration (EAI)
- UML Profile for Modelling QoS and Fault Tolerance Characteristics and Mechanisms
- UML Profile for Schedulability, Performance and Time
- UML Profile for System on a Chip (SoC)
- UML Profile for Systems Engineering (SysML)
- UML Testing Profile
- UML Profile for Enterprise Distributed Object Computing (EDOC)
Technologies used in MDA

• The Meta-Object Facility (MOF)
• XML Metadata Interchange (XMI)
• Common Warehouse Model (CWM)
Transformation example

UML Model (PIM)

<table>
<thead>
<tr>
<th>Konto</th>
</tr>
</thead>
<tbody>
<tr>
<td>KontoNr : String</td>
</tr>
<tr>
<td>Kontostand : Double</td>
</tr>
<tr>
<td>Name : String</td>
</tr>
</tbody>
</table>

XML Document (PSM)

```
<Konto>
  <KontoNr> 213213 </KontoNr>
  <Kontostand> 620.0 </Kontostand>
  <Name> John Doe </Name>
</Konto>
```

IDL, Java… (PSM)

Class Konto
```
{public String  KontoNr;
  public double Kontostand;
  public String Name;
}
```

XMI DTD, Schema (PSM)

```
<!Element Konto
  (KontoNr*,
   Kontostand*,
   Name*)>
```
MDA Tools

• A large number of tools exist
• Divided in two groups:
  – Open-source: AndroMDA, Kermeta, MOFScript, ModFact, OpenMDX, XDoclet, OOMEGA
  – Commercial: OptimalJ, ArcStyler, Xactium XMF, MosiacMetaEdit, MDWorkbench
Conclusion

• MDA is widely accepted
• Very useful in large distributed software development
• Issues with formal standard need to be resolved
• Too soon to tell if MDA is going replace the current development process
Thank you for your attention

Any questions?