Towards an MDA-Oriented UML Profile for Distribution

Raul Silaghi
Frédéric Fondement

Software Engineering Lab
Swiss Federal Institute of Technology Lausanne

3rd Key Workshop
June 7-9, 2004
Towards an MDA-Oriented UML Profile for Distribution

- Presentation out of an accepted paper for the next EDOC conference

Contents

- On MDA and Platforms
- Layered Platforms
- Platform Integration
- Conclusion
MDA Overview: The Y Pattern

- Can be iterative
  - More than one platform
  - Refinement steps
    - CIM->PIM

- Many other possibilities

© Object Management Group, Inc.: MDA Guide, V 1.0.1. omg/03-06-01
Platform

Platform description model (PDM):
- Domain-specific *metamodel*
- Domain-specific *profile*
  - Stereotypes, Tag Definitions and Constraints

Platform and technology
- Technology-Independent PDM
  vs.
- Technology-Dependent PDM
- Platforms *layers*

Platform integration
- Guiding the transformation: “Wizard” approach
Platform: Middleware

- UML as a system modelling notation
- Profiles as a platform modelling notation

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Concurrency</th>
<th>Transactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is remotely available in the system, and how to communicate?</td>
<td>What are the behaviours that cannot execute together?</td>
<td>What are atomic behaviours?</td>
</tr>
<tr>
<td>How to reach distributed elements with CORBA?</td>
<td>What are the locks to place in CORBA?</td>
<td>What CORBA transactional element?</td>
</tr>
</tbody>
</table>
## Platform: Middleware

- UML as a system modelling notation
- Profiles as a platform modelling notation

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Concurrency</th>
<th>Transactions</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Distribution Diagram" /></td>
<td><img src="image2.png" alt="Concurrency Diagram" /></td>
<td><img src="image3.png" alt="Transactions Diagram" /></td>
</tr>
</tbody>
</table>
An example

System

«Interface» BankI
+createAccount(name : String, bal : Double) : AccountI
+getAccount(name : String) : AccountI
+transfer(ac1 : String, ac2 : String, amount : Double)

Account
-name : String
-balance : Double
+getBalance() : Double
+withdraw(amount : Double)
+deposit(amount : Double)
-setBalance(amount : Double)

AccountI
+getBalance() : Double
+withdraw(amount : Double)
+deposit(amount : Double)

Bank
+createAccount(name : String, bal : Double) : AccountI
+getAccount(name : String) : AccountI
+transfer(ac1 : String, ac2 : String, amount : Double)
#getAccountList() : Account [*]

parameter

bank 1

* accounts
Distribution Profile

```mermaid
profile DistributionProfile

metaclass Interface

stereotype Distributed

metaclass InstanceSpecification

stereotype Servant

distributed 1..*

servants 0..*

{inv: self.classifier->forAll(c | c.conformsTo(self.distributed))}
```
Platform integration

- “The BankI is distributed”
CORBA Distribution Realization

 DistributionProfile

 AbstractDistributionRealizationProfile

 CORBANamingService

 host : String
 port : String

 CORBANamingRegistry

 CORBANameExposition

 CORBADistributionRealizationProfile

 «profile»

 «merge»

 «merge»

 «profile»
Platform integration

“The b servant is bound to a CORBA naming service at 127.0.0.1, on port 3028, with name BCV”
Platform integration

- «metamodel» UML
- «profile» Distribution
- «profile» CORBA Distribution
- Plugin for Distribution with OpenORB CORBA in Java
- Java Code using OpenORB CORBA

System

MTL1-D

Distributed System

MTL2-D

CORBA Distributed System

Parallax

Configuration

Configuration
Conclusion

- Platform separated into Concerns
- Concerns refined into Technologies
- Final support: extensible code generator

- “Wizard” model transformation for integrating platforms

- TBD:
  - More case studies
  - More technologies
  - More concerns
  - More platforms
  - More platform modelling languages
  - More system modelling languages