UML Notation for an Automated Deployment Process

Julia Reznik, Marc Born

GMD Fokus
{reznik,born}@fokus.fhg.de
UML Notation for an automated deployment process

Agenda

• Motivation
• Packaging and Deployment
• Model Driven Solution in order to support the automation of software deployment process
  – UML Notation for software components
  – CCM: Packaging and Deployment
  – automated Generation of CCM Deployment Description
Motivation

• Large network environments have moved to the center stage in the field of software deployment
• New tools are needed to automate the component-based software deployment life cycle in a distributed environment
• Deployment life cycle: wide range of deployment tasks:
  – Packaging, installation, update, reconfigure, adapt, remove
UML Notation for an automated deployment process

Packaging and Deployment

Software components

Packaging

Application

Deployment requirements

Deployment Process

System properties

Component distribution, loading, configuration and execution

Middleware

Target environment

Deployment Tool

manual

automated
MDA Solution: Deployment Software Description

• Bridges the gap between design and deployment phase of distributed applications
• Provides conventions for applying and specializing standard UML to the graphical deployment notation
• Bases on the definitions of the XML DTDs used by the CORBA Components
• Can be realized with existing UML tools (e.g. Rational Rose)
• Provides automatic generation of XML descriptors with a UML tool
Test Service Example for distributed systems

• Computational Object Types (COs):
  – TestManager
  – Main Test Component (MTC)
  – Parallel Test Component (PTC)
  – System Under Test (SUT)
UML Extension Mechanisms

• Stereotypes:
  – Sub-classification of an existing UML element
  – The new element has its own special properties (expressed as tagged values), semantics and notation

• Tagged Values:
  – New information about model elements and presentation elements (new properties)

• Constraints:
  – Conditions and restrictions, that apply to model elements
Diagrams and Stereotypes

• Class diagram: Computational Object Types (CO Types) and their interfaces
• Collaboration diagram: Initial configuration of COs and their factories
• Component diagram: implementation components
• Stereotypes:

<table>
<thead>
<tr>
<th>UML-Metamodel element</th>
<th>Stereotype</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>&lt;&lt;CO&gt;&gt;</td>
</tr>
<tr>
<td>Class</td>
<td>&lt;&lt;Factory&gt;&gt;</td>
</tr>
<tr>
<td>Class</td>
<td>&lt;&lt;Interface&gt;&gt;</td>
</tr>
<tr>
<td>Component</td>
<td>&lt;&lt;implementation&gt;&gt;</td>
</tr>
<tr>
<td>Package</td>
<td>&lt;&lt;assembly&gt;&gt;</td>
</tr>
<tr>
<td>Association</td>
<td>&lt;&lt;uses&gt;&gt;</td>
</tr>
<tr>
<td>Association</td>
<td>&lt;&lt;provides&gt;&gt;</td>
</tr>
<tr>
<td>Dependency</td>
<td>&lt;&lt;instantiate&gt;&gt;</td>
</tr>
</tbody>
</table>
UML Notation for an automated deployment process

Class diagram: CO Types and interfaces

<<Interface>>
i_MTCControl
- setTestParams()
- setTestBehavior()
- startTest()
- stopTesting()
<<uses>>
mtc
<<provides>>
mtc_control

<<CO>>
TestManager
<<provides>>
mtc_notify

<<CO>>
MTC
<<uses>>
mtc
<<provides>>
ptc_notify

<<CO>>
PTC
<<uses>>
mtc_control
<<provides>>
control

<<CO>>
SUT

<<Interface>>
i_TCCControl
- setTestCaseParams()
- setTestBehavior()
- startTest()

<<Interface>>
i_TestManagerNotify
- submittTestResult()

<<Interface>>
i_MTCNotify
- submittTestResult()
UML Notation for an automated deployment process

Initial Configuration

Class diagram “COs”

<<Assembly>>
TestService

<<Factory>>
TestManagerFactory
(from Logical View)

<<Factory>>
PTCFactory
(from Logical View)

<<Factory>>
MTCFactory
(from Logical View)

<<Factory>>
SUTFactory
(from Logical View)

<<Instantiate>>

<<CO>>
TestManager
(from Logical View)

<<CO>>
PTC
(from Logical View)

<<CO>>
MTC
(from Logical View)

<<CO>>
SUT
(from Logical View)

Collaboration diagram “InitialConfiguration”

tm1 : TestManager

mtc1 : MTC

link1

link2

tm1 : TestManagerFactory

ptc : PTCFactory

sutfactory : SUTFactory

mtcFactory1 : MTCFactory
Tagged Values (1)

- DCL specific keyword pairs: property name and value
- For stereotypes <<CO>> and <<Implementation>> tagged values are defined in the profile
- Source: CORBA Components Descriptors
  - Software Package
  - CORBA Component
  - Component Assembly
Tagged Values (2)

UML Notation for an automated deployment process

Second Workshop on UML™ for Enterprise Applications: Model Driven Solutions for the Enterprise
Constraints

- <<assembly>> must have two diagrams: “COs” and “InitialConfiguration” diagram
- A class diagram of <<assembly>> package contains only classes with <<CO>> and <<Factory>>
- Generalisation: all elements of the same stereotype (e.g. <<CO>>)
- Valid association stereotype combinations:

<table>
<thead>
<tr>
<th>From:</th>
<th>To:</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&lt;CO&gt;&gt;</td>
<td>&lt;&lt;CO&gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt;Factory&gt;&gt;</td>
<td>&lt;&lt;instantiate&gt;&gt;</td>
</tr>
<tr>
<td>&lt;&lt;Interface&gt;&gt;</td>
<td>&lt;&lt;uses&gt;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;&lt;provides&gt;&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;&lt;provides&gt;&gt;</td>
</tr>
</tbody>
</table>
UML Notation for an automated deployment process

Realization with UML tool Rational Rose

- Rational Rose provides
  - graphical support for UML
  - automatic generation of XML-Descriptors using Rose Extensibility Interface (REI)

Class Specification for TestManager

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Package Name</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>TestManager_W</td>
<td>ApplicationDatabaseDOT</td>
<td></td>
</tr>
<tr>
<td>TestManager_S</td>
<td>ApplicationDatabaseDOT</td>
<td></td>
</tr>
<tr>
<td>MTC_W</td>
<td>ApplicationDatabaseDOT</td>
<td></td>
</tr>
<tr>
<td>PTC_W</td>
<td>ApplicationDatabaseDOT</td>
<td></td>
</tr>
<tr>
<td>MTC_S</td>
<td>ApplicationDatabaseDOT</td>
<td></td>
</tr>
</tbody>
</table>

CO Type

<CO>
TestManager

<implementation>
TestManager_W

<implementation>
TestManager_S

1

*
UML Notation for an automated deployment process

Generation of softpkg descriptor (1)

<<CO>>
TestManager

Open Specification...
Open Standard Specification...
Sub Diagrams

New Attribute
New Operation

Select In Browser
Reolate

Options
Format

Generate Softpkg Descriptor
Generate Component Descriptor
UML Notation for an automated deployment process

Generation of Softpkg descriptor (2)
UML Notation for an automated deployment process

Generation of Component descriptor

<<CO>>
TestManager

- Open Specification...
- Open Standard Specification...
- Sub Diagrams
- New Attribute
- New Operation
- Select In Browser
- Help Site
- Options
- Format
- Generate Softpkg Descriptor

Generate Component Descriptor

```
<idl version="1.0">
  <!DOCTYPE idlcomponent SYSTEM "orbacomponent.dtd">
  <corbacomponent>
    <component id="IDL_TestManager1.0">
      <operator name="submitTestResult">
        <transaction use="required">
          <required rights="submit"/>
        </transaction>
      </operator>
      <interface name="L_TestManagerNotify">
        <operation name="submitTestResult">
          <transaction use="required">
            <required rights="submit"/>
          </transaction>
        </operation>
      </interface>
      <export operations="true"/>
    </component>
  </corbacomponent>
</idl>
```
UML Notation for an automated deployment process

Generation of Assembly descriptor

```xml
<co:componentassembly SYSTEM="componentassembly.dtd">
<co:descriptionExample TestManager:></co:descriptionExample>
</co:componentassembly>
```

![Screenshot of a user interface for generating assembly descriptors](image)

SECOND WORKSHOP ON UML™ for Enterprise Applications: Model Driven Solutions for the Enterprise 19
Conclusion

• Graphical support for automated deployment process
  – Specification of component-based software systems requirements
  – Specification of Initial Configuration of component instances
  – Automatic generation of XML-Descriptors for the Deployment process

• Under development:
  – Automatic generation of UML Deployment diagram from the XML Description of target environment
Thank you!