Project No. 6: Using OPM as a DSM

Group Members:
Amir Hasson  Lital Peretz
Neta Kedem  Ortal Betesh
Oshrit Saad
Main Subjects:

• DSM Definition (reminder).
• Relevance of the articles.
• OPM Meta-Model Additions.
• DSM Creating Wizard.
• Opcat Extensions.
• Example: Modeling DSM in OPCAT.
DSM Definition (reminder)

- **DSM**: Domain-specific modeling - a software engineering methodology for designing and developing systems.

- **The goal** of the project: proposing an extension to the OPM meta-model for OPM to be used as a DSM.
Relevance of the Articles

• For the project we had to read and summarize three articles:
  
  1. Collected Experiences: Defining Domain Specific Modeling Languages
  2. Integration Of DSML & UML Through UML Profile Extension Mechanism
Relevance of Article #1

Collected Experiences: Defining DSM Languages

• In the article, four approaches were presented:
  1. Domain expert’s or developer’s concepts.
  2. Generation output.
  3. Look & Feel of the system build.
  4. Variability space.

• In our project we integrated all of them.
1. Domain expert’s or developer’s concepts:
   - We would like to create a high level of abstraction, which allows the domain’s experts to **define** the DSM with no previous experience in modeling.
   - That’s why we thought about the **Wizard**.
   - In this way, the domain’s experts will be able to define the DSM using the wizard, and other entities will perform the actual modeling.
2. **Generation output:**

   - In this approach, we saw the difficulty of defining behavioral and logical parts, basing on the domain’s **rules**.
   - In order to handle with this difficulty, we’ve added special part to the wizard that will support this issue and relate objects to processes.
3. **Look & Feel of the system build:**
   We allowed using **colors** and **images** in order to support products that belong to this category.

4. **Variability space:**
   The solution for the domain’s dynamic problem that was raised in this approach is expressed in the ability to **edit** an existing DSM.
Relevance of Article #2
Integration Of DSML & UML Through UML Profile Extension Mechanism

- This article presents the idea of customization of UML by means of extensions defined in the UML metamodel.
- The metamodel represents the abstract syntax related to the semantics required for the MDD.
- The use of UML in that proposal is parallel to the use of OPM in our proposal.
Relevance of Article #2

• The extension mechanisms that can be used in order to introduce the required semantic precision into UML is **UML Profile extension mechanism**, which is part of the UML standard.

• In our Project we use the existing tools in **OPM** and propose extensions to **OPM-metamodel** in order to create the DSM.
Relevance of Article #2

• The main issue of the article is to present the **process** of making the **UML profile** adapted to DSM.

• Accordingly, in our project we present the **wizard** that was developed in order to create the **OPM workspace** adapted to DSM.
The article proposes using DSM languages to model business processes.

The article suggests using UML profiles and UML activity diagrams as the semantic base for these DSM languages.
Relevance of Article #3

- The article presents tools that help to create a DSM language and tool support for a given domain:

1. **AD-Modeler** implements a UML activity diagram editor within a specific domain of UML profile.

2. **AD-Specializer** can define and generate UML profiles for the AD-Modeler.
• In our project, we suggest a special tool which integrates into OPCAT and assists defining the DSM workspace.
OPM Meta-Model Additions

- The following changes were made:
  1. Insert an object named “Domain” as the “Language” object characteristic.
2. The “Domain” object consists of: “Domain Process” & “Domain Object” and characterized by: “Name”, “Color” & “Description”.

- Domain
  - Name
  - Color
  - Description

- Domain Object
- Domain Process
3. “Domain Process” is a Process and “Domain Object” is an Object and both are characterized by an “Image”.
4. Another important characteristic of a "Domain Object" & "Domain Process" is the fact that they are the **building blocks** of the domain. This fact is expressed by the following limitation:

It is impossible to perform **unfolding** or **in-zooming** on a process or an object of that type.
DSM Wizard

• The description of the workflow definition of DSM using the wizard will be illustrated with examples from the family domain.

• The family domain was chosen because it is well known and easy to illustrate.

• For the illustration we will define the building blocks of the domain: the basic processes and basic objects and their characteristics.
DSM Wizard

• When we are defining the **building blocks**, we need to make sure we're not going to tell more details than their states and their characteristics, because we can not perform Unfolding or In-Zooming in the modeling.
# Building Blocks For The Family Domain

<table>
<thead>
<tr>
<th>Processes</th>
<th>Objects</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Building Blocks For The Family Domain

<table>
<thead>
<tr>
<th>Processes</th>
<th>Objects</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cleaning</td>
<td>Parent</td>
<td>Name, Gender, Age</td>
</tr>
<tr>
<td>Driving</td>
<td>Child</td>
<td>Name, Gender, Age</td>
</tr>
<tr>
<td>Cooking</td>
<td>Car</td>
<td>Model, Color</td>
</tr>
<tr>
<td>Baking</td>
<td>Oven</td>
<td>Firm, Condition</td>
</tr>
<tr>
<td>Purchasing</td>
<td>Refrigerator</td>
<td></td>
</tr>
<tr>
<td>Playing</td>
<td>Sink</td>
<td></td>
</tr>
<tr>
<td>Eating</td>
<td>Kitchenware</td>
<td>Quantity, Condition</td>
</tr>
<tr>
<td></td>
<td>Gardening</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tools</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Toilet Articles</td>
<td></td>
</tr>
</tbody>
</table>
Domain Definition:
Name: Family
Color: Yellow

Description:
This domain includes tasks and instruments which are required for family activities.
Characterization:

Name
Gender
Age

Object Image:

\( c:\\text{pictures}\\text{father.jpg} \)

Is Human
Add New OPM Object

Object Name: Car
Initial Value:

Compound Object
- Basic Types
- Advanced Types
- Custom Types

Essence:
- Physical
- Informatical

Origin:
- Environmental
- Systemic
Characterization:

Model
Color

Object Image:
c:\pictures\car.jpg

Is Human
Process Name

Eating

Essence
- Physical
- Informatical

Origin
- Environmental
- Systemic
<table>
<thead>
<tr>
<th>Process Image:</th>
<th>c:\pictures\eating.jpg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td></td>
</tr>
</tbody>
</table>
Error
You can't connect agent link to a car. Try to choose another link.

Edit car properties...
Arrange processes

Move processes by logical order to define time line:

- Cooking
- Eating
- Cleaning
- Driving
- Purchasing
- Playing
- Baking

- Driving
- Purchasing
- Cooking
- Eating

Save Group  Next
Final Report

• The purpose of the report is to receive an interim summary of the objects and processes defined in the system, with the connections between them.

• The report will include the follows:
  1. A list of defined **objects** - including the characteristics and conditions set for them.
  2. A list of defined **processes**.
  3. For each object / process, the connections will be presented as OPL sentences.
Final Report

• After reviewing the report, there will be two possibilities to choose from:

  Approve the report.  or

  Return to the wizard and make changes in the DSM definition.

• The report is displayed as html.
Family Domain:

**Parent**
- Parent is human.
- Parent is Physical.
- Parent exhibits Name, Gender, and Age.
- Parent handles Purchasing, Driving, Cooking, Cleaning and Eating.

**Car**
- Car is Physical.
- Car exhibits Model and Color.
- Driving requires Car.

**Eating**
- Eating is Physical.
- Parent handles Eating.
- Child handles Eating.
Steps for creating new DSM via OpCat:

1. Open OpCat as usual.
2. Open DSM creating wizard.
3. Fill the wizard pages with the relevant data for the DSM.
4. In the end of this process a new DSM will be created in the system.
Press: Generation ➔ DSM ➔ New DSM

You can also edit an existing DSM
OpCat Extensions

Steps for using an *existing* DSM via OpCat:

1. Open a new project in OPCAT
2. In the page: “New System Properties”, in Model Type choose: "DSM".
3. For "DSM Type" choose the DSM that you want to model with.
4. The relevant workspace to the chosen DSM will be created in OPCAT.
Steps for using an existing DSM via OpCat:

5. Now, in the bottom of the window a Toolbox with all the domain objects and domain processes of the DSM will be appeared.

6. Start modeling!
Press: System → New → "OK"
Choose the DSM you want to model with
Drag the objects/processes to the workspace.
Exemple: Modelling DSM in OPCAT

SD: Having Dinner
Exemple: Modelling DSM in OPCAT

SD1: Having Dinner
Exemple: Modelling DSM in OPCAT

View1: Kitchen
Thank You!
Questions?