Graphical Model Editing Framework
MSU Requirements Engineering Proposal for Fall 2006

Networks and Infrastructure Research Lab

Name | Organization
---|---
Kabe VanderBaan | Networks and Infrastructure Research Lab, Illinois
Dave Raymer | Networks and Infrastructure Research Lab, Texas

September 13, 2006
Version 1.0
### Revision History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Change Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.1</td>
<td>8/22/2006</td>
<td>Initial Release</td>
</tr>
<tr>
<td>.2</td>
<td>9/9/2006</td>
<td>Overhaul to make it more appropriate for the Requirements course.</td>
</tr>
<tr>
<td>1.0</td>
<td>9/13/2006</td>
<td>Release For Requirements Course</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>PURPOSE</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>OVERVIEW</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>SHAPES</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>USER REQUIREMENTS</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>4.1 Eclipse</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>4.2 Editor</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>4.3 Connections</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>REFERENCES</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>
1 Purpose
Develop a framework to be used by programmers for the development of graphical model editors for Eclipse.

2 Overview
A model is a semantically complete abstraction of a system where completeness is defined by the requirements placed upon the model by a set of stakeholders. For example, a common language that is used to model software is UML. Among other things, UML can be used to create a class diagram which shows the static structure of classes within a model (i.e., a UML class diagram). Many programmers are familiar with graphical UML editing tools such as Rational Rose or Microsoft Visio. Both of the aforementioned tools maintain a model, and possibly a set of diagrams associated with said model. A diagram is the visual representation of a model or some portion thereof.

Therefore, a Graphical Modeling Editor Framework (GMEF) would be useful to allow for the programmatic creation of a set of graphical editors that can modify an underlying model. This framework should not be dependent upon any given modeling language or persistent representation thereof.

3 Shapes
Shapes represent the core elements that can be modeled in a modeling language. For example, in a UML Class Diagram, a class is represented as a rectangle consisting of the classes name and 2 sections representing the class’s attributes and operations. These shapes map directly to a model element.

4 User Requirements
The user requirements describe the core functionality that the framework must implement. This section describes, at a high level, the core functionality that is expected by the end user (a programmer).

4.1 Eclipse
- The framework should be created and deployed as an Eclipse plug-in and should conform to the best practices described in the book, Eclipse: Building Commercial Quality Plug-ins.
- The framework may (and probably should) be built as an extension to the Eclipse Graphical Editing Framework (GEF).
4.2 Editor

- The user of the framework should be able to programmatically create an editor and a palette of shapes. The palette should consist of the shapes that are valid within a given modeling language.
- Changes in the graphical editor (i.e., the “view”) should be reflected in the underlying model.
- The framework will be capable of saving a “description” of a diagram into a file separate from the model such that the diagram can be “reloaded” and manipulated at a later point in time.
- “Shapes” will be capable of displaying “0 .. n” compartments. By example, the graphical depiction of a class in a UML diagram can have 1 to 3 compartments: the identification compartment, the attribute compartment, and the operations compartment.
- Elements should be selectable, and when selected, can be moved and/or deleted. If deleted all connectors that terminate at the deleted element must also be deleted.

4.3 Connections

- The generic framework should assume the structure of the models being rendered conforms to that of a graph, and be capable of efficiently navigating undirected, cyclic graphs as well as directed acyclic graphs.
- At the minimum connectors (lines that connect one graphical entity to another) should be radial. That is the line should be anchored to the center of each “shape”.
  - It shall be possible to subsequently place vertices into the connector to manually route the connector in the diagram
  - orthogonal (non-radial, right angle) “best-guess” routing of connectors
- Connectors needs to have a distinct concept of ends (i.e., arrows can be dealt with)

5 References

1) Eclipse Framework, [www.eclipse.org](http://www.eclipse.org)
2) Graphical Editing Framework, [www.eclipse.org/gef](http://www.eclipse.org/gef)