UML Tool Actions - Project Actions 1.0 Requirements Specification

1. Scope

1.1 Overview

The Project Actions component provides the Actions related to the project. The actions are strategy implementations of the action interfaces in the Action Manager component. The provided actions are for loading a TCUML / XMI file, saving to a TCUML / XMI file, for creating a new project, generating source code, printing diagrams and exporting diagrams to image files.

1.2 Logic Requirements

1.2.1 Create a new project action

This action will create a new project. It will empty the Model, the list of ActivityGraphs and the list of Diagrams from the UMLModelManager. And it will set the project language.

The action will be configured with:

- the project language

This action is an Action, so only the execute method should be provided.

1.2.2 Load a TCUML file action

This action will create a new project and will load the TCUML file, by unzipping it and loading the XMI file inside. See below more information.

The action will be configured with:

- the TCUML file

This action is an Action, so only the execute method should be provided.

1.2.3 Load an XMI file action

This action will create a new project and will load the XMI file. It will use XMI Reader component to parse the contents into the UML Model.

After that, it will get the Model from the UMLModelManager and check if it has a "ProjectLanguage" TagDefinition. It will get the tag value and set it in the UMLModelManager. The default project language from ProjectConfigurationManager is used, if this value is missing.

A "ProjectLanguage" TagDefinition is created with the default language, in case it is missing, and add it to the Model.

The action will be configured with:

- the XMI file
- whether to save the diagram data or not

This action is an Action, so only the execute method should be provided.

1.2.4 Save to a TCUML file action

This action will save the project to a TCUML file, which is a zip file containing the actual XMI file inside. See below more information.

The action will be configured with:

- the TCUML file

This action is a TransientAction, so only the execute method should be provided.

1.2.5 Load to an XMI file action

This action will save the project to an XMI file. It will use XMI Writer component to transform the UML Model to XMI.

The action will be configured with:

- the XMI file

This action is a TransientAction, so only the execute method should be provided.

1.2.6 Generate code action

This action will generate code for the given packages or classifiers. It will use the Stub Class generator component to perform the actual logic.

The action will be configured with:

- the Classifier, the list of Classifiers, the Package or the list of Packages
- the language; if missing the project's language is used
- the location of the source files generated

This action is a TransientAction, so only the execute method should be provided.

1.2.7 Print diagram action

This action will print a diagram. The diagram should be received as a java.awt.Component, not as a Diagram. The diagram Component should be the graphical representation of the diagram. It will be passed to the Print Manager component.

The action will be configured with:

- the diagram Component or the list of diagram Components
- the viewport or the list of viewports (one for each diagram) - optional

This action is a TransientAction, so only the execute method should be provided.

1.2.8 Export diagram to image file action

This action will export a diagram to an image file. The diagram should be received as a java.awt.Component, not as a Diagram. The diagram Component should be the graphical representation of the diagram. It will be passed to the Image Exporter component.

The action will be configured with:

- the diagram Component or the list of diagram Components
- the viewport or the list of viewports (one for each diagram) - optional
- the location of the image files
- the image files
- the image format

This action is a TransientAction, so only the execute method should be provided.

1.3 Required Algorithms

None.

1.4 Example of the Software Usage

The component will be used in the TopCoder UML Tool to perform project related actions.

1.5 Future Component Direction

None.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None.

2.1.2 External Interfaces

The design must follow the interface found in the class diagram with the component interfaces. The designer is encouraged to add to the existing interface, but not to remove anything.

2.1.3 Environment Requirements

- Development language: Java 1.5
- Compile target: Java 1.5

2.1.4 Package Structure

com.topcoder.uml.actions.project

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

None.

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

None

3.2.2 TopCoder Software Component Dependencies:

- Action Manager 1.0
- UML Model Manager 1.0
- UML Project Configuration 1.0
- UML Model components
- XMI Reader 1.0
- XMI Writer 1.0
- Stub Code Generator 1.0
- Print Manager 1.0
- Image Exporter 1.0
- Configuration Manager 2.1.5 - recommended

**Please review the TopCoder Software component catalog for existing components that can be used in the design.

3.2.3 Third Party Component, Library, or Product Dependencies:

None

3.2.4 QA Environment:

- Solaris 7
- RedHat Linux 7.1
- Windows 2000
- Windows 2003

3.3 Design Constraints

The component design and development solutions must adhere to the guidelines as outlined in the TopCoder Software Component Guidelines. Modifications to these guidelines for this component should be detailed below.

3.4 Required Documentation

3.4.1 Design Documentation

- Use-Case Diagram
- Class Diagram
- Sequence Diagram
- Component Specification

3.4.2 Help / User Documentation

- Design documents must clearly define intended component usage in the ‘Documentation’ tab of Poseidon.