UML Tool Actions - Auxiliary Elements Actions 1.0 Requirements Specification

1. Scope

1.1 Overview

The Auxiliary Elements Actions component provides the Actions related to the auxiliary elements specific to all diagrams. The actions are strategy implementations of the action interfaces in the Action Manager component. The provided actions are for adding / removing / copying / cutting / pasting the elements and relationships. The elements are comment, free text and polyline.

1.2 Logic Requirements

1.2.1 Comment actions related to the model

1.2.1.1 Add / Remove / Copy / Cut / Paste Comment action

This component will provide a concrete action for each operation.

The Add / Remove / Cut / Paste actions are UndoableActions.

The Copy action is a TransientAction.

Note that the Cut action can be implemented as a CompoundUndoableAction, made of a transient Copy action (wrapped in TransientUndoableAction) and an undoable Remove action.

1.2.1.1.1 The Add action will be configured with:

- the Comment
- the parent Namespace (optional)

The action will simply add the Package to the parent namespace's owned elements, or directly to the Model.

The action will also pass the element to the ProjectConfigurationManager, to apply any initial formatting.

1.2.1.1.2 The Remove action will be configured with:

- the Comment

The action will remove the element from the model. However, it is not responsible for removing the relations connected to it.

1.2.1.1.3 The Copy action will be configured with:

- the Comment
- the Clipboard (defaults to the system clipboard)

The action will remove the element from the model. However, it is not responsible for removing the relations connected to it.

The copy information will be placed in the clipboard. Note that the Copy and the Paste action must function together.

The DataFlavor of the Transferable object used should be documented.

1.2.1.1.4 The Cut action will be configured with:

- the Comment

This action will Copy and Remove the element, as specified above.

1.2.1.1.5 The Paste action will be configured with:

- the Transferable content representing the Comment
- the parent Namespace (optional)

The action will paste the element into the model in the same namespace it was in, or in the provided namespace. It will get the information from the received Transferable object.

1.2.1.2 Add / Remove / Copy / Cut / Paste Comment Relationship action

These actions are similar to the ones above. They use (Comment, ModelElement) pairs, instead of Comment. (The relationship is represented simply by adding the Comment to the comments list of the ModelElement).

1.2.1.3 Change Comment text action

This action will change the text of a comment - Comment.body property.

The action is an UndoableAction.

The action will be configured with:

- the Comment
- the text

1.2.2 Comment actions related to the diagrams

1.2.2.1 Add / Remove / Copy / Cut / Paste Comment GraphNode action

These actions are similar to the ones above. The actions are applied to a diagram, not the model and they use a proper GraphNode with the proper compartments, instead of Comment.

1.2.2.2 Add / Remove / Copy / Cut / Paste Comment GraphEdge action

These actions are similar to the ones above. They use a GraphEdge and the GraphNodes connected, instead of the GraphNode.

1.2.3 Actions related to auxiliary diagram elements

1.2.3.1 Add / Remove / Copy / Cut / Paste FreeText action

These actions are similar to the ones above. They use a GraphNode with a FreeText inside, instead of Comment's GraphNode.

1.2.3.2 Add / Remove / Copy / Cut / Paste Polyline action

These actions are similar to the ones above. They use a Polyline, instead of Comment's GraphNode.

1.3 Required Algorithms

None.

1.4 Example of the Software Usage

The component will be used in the TopCoder UML Tool to perform auxiliary model and diagram 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.auxiliary.comment.model

com.topcoder.uml.actions.auxiliary.comment.diagram

com.topcoder.uml.actions.auxiliary.diagram

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
- Diagram Interchange 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.