Documentation Panel 1.0 Requirements Specification

1. Scope

1.1 Overview

The Documentation Panel component provides a SWING panel that allows the user to view and enter the documentation for model elements. It also provides a way to signal the listeners of changes.

1.2 Logic Requirements

1.2.1 Editor

The component will provide an editor for the documentation. The editor should be pluggable. It should be configured through the Configuration Manager.

The component will provide a simple text area, with scroll bars, which are shown only when needed. The text will be wrapped (this property should also be configurable).

1.2.2 Supported data structures

The component must work with the UML ModelElements. It should get the documentation from the “documentation” taggedValue of the element provided.

1.2.3 Documentation tags panel

The component will provide a list of arguments (only for operations) and documentation tags in the right. The editor and this section should be in a split panel (the position of the divider should be kept between consecutive usages during the same application run).

This section should be displayed for the concrete nodes, not for the relations. The documentation for the tags will be obtained from the “documentation#tagName” tagged values of the elements.

The Operation will have, at the top of this list, the list of arguments’ documentation tags. The documentation for the arguments will be obtained directly from the arguments, from the “documentation” tagValues.

For all the nodes there should be an “Insert tag” option to add new tags. This combo-box should be configured with the default tag names to be provided. To remove a tag, the documentation should be cleared by the user.

The documentation for the tags should be displayed on one line, even if it contains line breaks.

1.2.4 Groups

The panel will display a blank panel if a group is selected. There should be a method to tell the component to display the blank panel.

1.2.5 Events

The component will generate events when the values are changed and it should provide a way for the application to register for these events.

1.3 Required Algorithms

None.

1.4 Example of the Software Usage

The component will be used in the TopCoder UML Tool as the documentation panel of the model elements.

1.5 Future Component Direction

None at this moment.

2. Interface Requirements

2.1.1 Graphical User Interface Requirements

None.

2.1.2 External Interfaces

None.

2.1.3 Environment Requirements

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

2.1.4 Package Structure

com.topcoder.gui.panels.documentation

3. Software Requirements

3.1 Administration Requirements

3.1.1 What elements of the application need to be configurable?

- The documentation editor
- The names of the tags, according to the type of element.
- The line wrap property of the text area.

3.2 Technical Constraints

3.2.1 Are there particular frameworks or standards that are required?

None.

3.2.2 TopCoder Software Component Dependencies:

- UML Model components
- Configuration Manager 2.1.5

**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.