# Background

Several customers have requested enhancements for the pop-up display of model-element descriptions. A good deal of overlap exists among these requests. Separately, a number of customers including have also requested the ability to annotate models with arbitrary bodies of text (and in one case, graphics) loosely attached to a model element. Unlike model-element descriptions these “sticky notes” can be related many to many with model elements or not attached to a model element at all.

This document specifies the requirements for these two different features because the capabilities for editing and displaying them are closely related. While the capabilities are quite similar, the constructs themselves (descriptions and notes) are, in fact, quite different. Each model-element description is a formal part of the specification that is an xtUML model while each note is an informal annotation, the lifetime of which may be much shorter than that of the associated model fragment.

# Model-element Descriptions

The present tool-tip implementation that displays a model-element description when the cursor hovers over that element is immensely helpful…and frustrating at the same time. For short descriptions the five-second delay is sufficient, but for longer ones it is maddening. The most common request from our customers is therefore to leave the description visible as long as the cursor is either hovering over the model element or the description pop-up display. Other requests include the ability to size, locate and specify whether the description remains visible on the canvas regardless of the cursor location.

# Annotations (UML Notes)

For years our customers have requested an ability to add UML notes (think sticky-notes) to their models. These may be used for a wide variety of purposes including explanations of work remaining, tracking questions and quandaries over various abstractions-in-progress, and providing review comments.

# Common Editing and Display Capabilities

Many of the capabilities envisioned for descriptions and notes are common, and those common requirements are specified below using the following terminology:

* Content attribute – The contents of a note or a model-element description.
* Content display – The on-canvas display of a note or model-element description.
* Content editor – The editor for a note or model-element description.

Since BridgePoint supports layers which can be hidden or displayed as the user sees fit, the content display on the canvas leverages the layer facility to provide the user with a convenient way of hiding or displaying all notes or descriptions of a particular type.

# Use Cases

This use cases specifications below are provided to illustrate the intended use of some of the less obvious capabilities. This is not an exhaustive list of the use cases envisioned for the specified capabilities.

### UC01 – Address review comments

#### Pre-conditions

1. A number of review comments, each one entered as a UML Note and associated with a layer named “ReviewComments”, exist.

#### Post-conditions

1. Review comments have been addressed by modifying the models as necessary.
2. All review comments that have been addressed have been deleted, leaving only those that must be addressed remaining.

#### Scenario

1. User displays the content attribute list.
2. User filters the content attribute list to include only those attached to the ReivewComments layer.
3. For each review comment in the list,
   1. User employs automatic navigation from content attribute list to the content attribute
   2. Editor displays the content attribute along with the model element(s) associated with the content attribute.
   3. User modifies the model elements in question to address the review comment.
   4. User deletes the review comment.

### UC02 – Clarify a review comment

#### Pre-conditions

1. A number of review comments, each one entered as a UML Note and associated with a layer named “ReviewComments”, exist.

#### Post-conditions

1. Review comment has been clarified, allowing user to modify models to address it.

#### Scenario

Duplicate steps 1 and 2 from UC01.

1. User employs automatic navigation from content attribute list to the content attribute.
2. Editor displays the content attribute along with the model element(s) associated with the content attribute.
3. User determines that this review comment requires clarification before it can be addressed.
4. User notices the identity of the author as displayed on the note.
5. User contacts author of the note who clarifies the review comment.
6. User modifies the model elements in question to address the review comment.
7. User deletes the review comment.

# Common Requirements

|  |  |
| --- | --- |
| **ID** | **Description** |
| C-1 | A content attribute may contain text. |
| C-2 | A content attribute may contain arbitrary graphical data, including static images, to be displayed in both the editor and in the content display on the canvas. |
| C-3 | A content attribute may contain hyperlinks to arbitrary locations including directories, documents, spreadsheets, presentations, videos, and web pages. |
| C-4 | A content display may be resized. |
| C-5 | A content display may be relocated on the canvas. |
| C-6 | The (existing) content editor may be opened conveniently and intuitively from the canvas. |
| C-7 | Content is wrapped across lines in the content editor when necessary rather than providing a horizontal scroll bar. |
| C-8 | The font, including type, size, color, etc. for all content editors may be specified as a preference. |
| C-9 | The font, including type, size, color, etc. for all content displays may be specified as a preference. |
| C-10 | Vertical scrolling is available for a content display whenever the contents cannot be displayed within the current height of the display. |
| C-11 | Content is wrapped across lines in the content display when necessary rather than providing a horizontal scroll bar. |
| C-12 | Each instance of content display may be associated with any number of layers. |
| C-13 | A (new) on-canvas content editor may be opened conveniently and intuitively. |
| C-14 | A list of all content attributes can be displayed. |
| C-15 | The content attribute list may be filtered by layer. |
| C-16 | Simple navigation from an entry in the content attribute list to the editor(s) for the associated model element(s) is provided. |
| C-17 | Size and location information for each content display is stored and committed to the repository similarly to existing graphical data. |
| C-18 | Once open, the on-canvas content editor must be deliberately dismissed. |
| C-19 | Changes made within the on-canvas content editor may be saved. |
| C-20 | Changes made within the on-canvas content editor may be discarded. |

# Model-element Description Requirements

|  |  |
| --- | --- |
| **ID** | **Description** |
| D-1 | Whenever the cursor is hovering over a model element with a non-empty description, the description is displayed within a content display which remains visible until the cursor is no longer hovering over either the model element or the content display in question. |
| D-2 | A pre-defined layer exists for each type of description that may be displayed in a content display (class, association, component, interface, etc.), and each content display associated with such a description is automatically associated with the appropriate pre-defined layer. |

# Note Requirements

|  |  |
| --- | --- |
| **ID** | **Description** |
| N-1 | Deprecated. |
| N-2 | A note may be displayed in a content display. |
| N-3 | A note may be associated with any number of model elements. |
| N-4 | A model element may be associated with any number of notes. |
| N-5 | Each association between a note and a model element is shown graphically on the diagram. |
| N-6 | Each note may be associated with any number of layers. |
| N-7 | A single pre-defined layer exists for all notes and each note is automatically associated with this layer when the note is created. |
| N-8 | The identity of the user who creates a note is captured and stored as an attribute in the meta-model. |
| N-9 | The identity of the user who creates a note is visible as part of the note display. |

# Implementation Thoughts

## Content Attribute List (C-14, 15, 16)

It’s important for reviewers to have an easy way of determining when they have handled all open review comments, having either accepted or rejected each one. It’s also important for a developer to have a simple way of determining when he has addressed all review comments attached to model elements he owns.

Two approaches have been considered thus far, one based on the Problems view and another based on search results.

The first is attractive because it can be sorted by path, allowing users to see all the content attributes associated with a particular model element, grouped together in one area of the list.

Using the xtUML search facility for this purpose would likely require an extension of the facility to allow searches for instances of meta-model classes. This approach would also likely require the ability to search using complex criteria such as “find all instances of Note assigned to <someLayer> and attached to <meta-model instance>.” For example, this would allow reviewers to find all open review comments attached to a particular use case.