Penultimate UML Builder
Functional Specifications Document

Bits, Please

1. Introduction

The Penultimate UML Builder (PUB) is a graphical UML Editor that allows the user to create, edit, and save UML diagrams in multiple formats for use in the software development lifecycle. The user is presented with a document view panel and one or more toolbars, which can be selected to create the UML diagram that best suits their needs. Using this toolbar, the user can create, resize, move, and clone class objects within the document view panel.

1.1 System Personnel

Joe Martello
Josh Wakefield
Kevin Fisher
Merv Fansler
Michael Sims
Vincent Viggiano

1.2 Operational Setting

This application is to be used in a Java Runtime Environment (JRE).

2. Functional Requirements

2.1 User Interface Overview

The application will provide a graphical user interface (GUI) window where all interactions will occur. The main components of the window will be:

1. A *document view pane* which displays the current UML document.
2. A *toolbox pane* with icons of objects that can be added to documents and operations that carried out on document objects.
3. A *properties pane* that displays and provides an interface for editing properties of selected document objects.
4. A *menu bar* which provides access to program-level commands, as well as some document operations.
5. A *context menu* which provides access to context-relevant commands.

In addition to a graphical user interface, some commands will be available to the user via keyboard shortcuts.

2.2 System-Specific Requirements

2.2.1 Object creation

The menu interface will provide options to generate basic UML objects. The following UML objects will be supported: Class, Association, and Dependency.

2.2.2 Object Editing

All objects in the field will have editable text fields in all appropriate places associated with the objects. Text fields can be edited either by double-clicking on the text in the document view panel or by selecting the object and editing the text fields that appear in the Properties panel. Selected objects can be deleted.

2.2.2.1 Object Resizing

Objects representing things in UML (e.g., Classes) will be resizable. The user will be able to drag the edge of such objects to expand or contract the object in either the horizontal or vertical direction. The user will also be able to drag the corner to expand/contract in both the horizontal and vertical directions simultaneously. Additionally, the dimensions of individually selected objects will appear as editable text fields in the Properties pane.

2.2.2.2 Object Translation

All objects in the document view panel will be selectable. Once an object is selected, the arrows will trigger the translation of the object within the document grid in the direction corresponding to the arrow. Objects can also be translated via dragging of selected objects. Translation of objects that have arrows associated with them will also transform the arrows to maintain relative association.

2.2.3 Copy-Paste Operations

Upon object selection, objects in the document view panel can be copied using either the command menu (**Edit > Copy**), the keyboard shortcut **Ctrl-C**, or the context menu. Copied objects can be pasted using either the command menu (**Edit > Paste**), the keyboard shortcut **Ctrl-V**, or the context menu. The center of the bounding box of the pasted items with be located at the of the document upon pasting. UML objects representing relationships (e.g., dependency) can only be copy-pasted if the associated objects are also included in the operation.

3. Non-Functional Requirements

3.1 System-Related Non-Functional Requirements

a. **Performance**
   The application will provide a responsive user interaction experience, allowing the user to create, edit, resize, and clone class objects in real-time. The application will perform normally on all systems satisfying the base system requirements specified for Java 8.0 [http://java.com/en/download/help/sysreq.xml].

b. Operational Environment

i. Hardware Platform

All hardware platforms supported by the Java 8.0 runtime will be capable of executing the application.

ii. Software Platform

The application will be cross-platform, capable of functioning in Windows, Linux, and Mac OS X environments.

c. General Characteristics

i. Reliability

The application will perform the basic functions of creating UML notations.

ii. Robustness

PUB is built in such a way that it will be able to handle any situation that is inputted by the user.

iii. Accuracy of Data

Data will be stored on the local drive to maintain the accuracy of all information.

iv. Portability

The application will be built to run on the latest Java 8.0 Runtime Environment and therefore it will be available for Mac OS, Linux, and Windows.

v. Modifiability and Extensibility

The initial iteration will not include any interfaces for modules or extensions.

3.2 Process-Related Non-Functional Requirements

a. Development Time

The implementation and testing deliverables for the current iteration of the application have a planned two week development period.

b. System Delivery

The implementation and testing deliverables will be delivered to the "*~hutchens420/Projects/BitsPlease/Iter1*" directory on the **cs.millersville.edu** server.

3.3 Personnel-Related Non-Functional Requirements

a). For Developers:

i. Credentials

Developers will have completed CSCI 362 (Data Structures) or have equivalent development experience.

b). For Users:

i. Skill Level

Users should have basic knowledge of UML notation and functionality.

4. Requirements Specification Rationale

The present limited set of features specified in this iteration is aimed at providing an initial foundation for working out the essential architecture of the application and developing familiarity with any framework libraries that may be used in the application. The feature set is intended to capture a small but diverse range of features that encourage the developers to identify how the range of functionality is to be managed. For instance, the subset of UML notation that is to be made available in the first iteration covers only what is needed to carry basic object-relation diagrams. As the operations on those generic objects is refined, future iterations will expand the symbols available to the user. Some features were postponed because of their dependency on the potential architecture of the UML document representation. For example, document saving is left out until the document representation is better established.