Object-Oriented Requirements Annotator
Prepared for OOSC
Summer Semester 2004

Dr. Bertrand Meyer
Joseph N. Ruskiewicz
Introduction

System Purpose

The purpose of the desired system, called “Object-Oriented Requirements Annotator,” or OORA, is to establish, manage and track the connections between:

• A requirements document for a system.
• A software implementation for that system.

System Rationale

One of the most difficult issues in software projects is to control the relationship between the requirements for a system, usually expressed as a natural-language “requirements document”, and the software that is supposed to implement these requirements, expressed in a programming language. This relationship determines:

• For a particular element of the requirements document (chapter, section, paragraph), which elements of the software (classes and features in an object-oriented approach) are directly influenced by the requirements element.
• In the reverse direction, for a given element of the software, what part of the requirements, if any, justifies the software element’s presence.

Tracking this two-way relationship makes it possible in particular to determine:

• Whether any part of the requirements has been missed in the software.
• Whether any part of the requirements is handled by two different parts of the software, possibly leading to inconsistencies.
• Most importantly, when requirements change, what parts of the software are affected and hence must be checked for possible updating; and the other way around, when the implementation changes, what part of the requirements must be checked to ensure that the implementation still reflects the requirements faithfully.

The OORA system will enable developers to define explicitly the connections between elements of the requirements and elements of the software, and will provide a set of tools for requirements tracking and change management.

Technology choices

The OORA system assumes the following decisions:

• The requirements are expressed as a Microsoft Word document.
• The software is expressed as an Eiffel system.

It is expected that the OORA system will consist of two major parts:

• The Requirements Connector (OORA-RC): a GUI (Graphical User Interface) tool making it possible visually to define connections between elements of a Microsoft Word window and elements of an EiffelStudio window.
• The Requirements Tracker (OORA-Tracker): a tool providing information about the connections and about the consequences of a change in either the requirements or the software.
System Functionalities

Defining the precise set of functionalities is part of the project. Here are some examples that give an idea of what should be supported.

OORA-RC
1. Select an element from the requirements document: glossary entry, sentence, paragraph, section, or chapter.
2. Pick a glossary term (requirements document), pick a class (software), connect the two visually, e.g. through drag and drop. This generates a comment in the software text, perhaps an added element in the requirements text. A window might come up prompting the user to enter information (comments) about the connection.
3. Select a sentence or entire paragraph in the requirements document, pick a feature in the software text and connect the two visually. Again, prompt the user to add comments about the connection.

OORA-Tracker
1. Given a class or feature in software text, find all the text elements to which it has been connected.
2. Given a text element (sentence, paragraph, section, glossary, entry, etc.) find all the software elements to which it has been connected.

Testing
The OORA project should have its own requirements document. It should be applied to connect the requirements with the implementation and provide a showcase of the functionalities.