Object Spyglass Project Requirements

Peter Brandt Martin Geisler

Object Oriented Software Construction
Summer Semester 2005

1 Introduction

The purpose of this project is to build a tool to graphically explore object state called “Object Spyglass”, and a client system for this tool. The client system is a Testing Framework that uses contract violations to signal bugs, and passes objects to the Object Spyglass to be displayed.

1.1 Testing Framework

The Testing Framework must provide a facility for code under test to save the system state (defined as object state, routine name, and routine arguments) at the beginning of each routine call. The framework tests code using the concepts of Test Case and Test Driver. A Test Case specifies the test inputs and conditions and contains the calls to the tested routines. In general, a Test Case can also specify the expected result, but in our approach this is the purpose of contracts. The Test Driver runs the test cases on the system under test. This architecture allows the user to write their own code for testing, and their own test cases.

When a contract violation is detected, the Testing Framework shows the user the following information:

- Test Case name
- Type of contract violation
- Class and routine where the violation occurred
- Tag of the violated assertion

A GUI for the testing framework indicates testing progress, and allows the user to browse information about individual test cases. If the user wants to see last saved state for the currently displayed test case, a button launches a new window. In this window, the testing framework displays the information by passing the saved state information to the Object Spyglass. Multiple instances of this window are possible for side-by-side viewing of saved states.

1.2 Object Spyglass

The Object Spyglass is a very general tool for the display of object state information. It can accept any type of object, from objects of basic types such as integers and characters to objects of arbitrary complexity. The Object Spyglass then builds a collapsible tree which contains information about the object and its fields, including class names, variable names, and values. For complex data types and fields, the Object Spyglass recurses so that the user can browse information about those classes as well.

Because the Object Spyglass is so general, the testing framework can use it to display both the routine arguments and object state of the last saved state by creating multiple Object Spyglass objects. It has a very simple interface, and does not rely on any classes from the testing framework. Below are characteristics the Object Spyglass must exhibit:

- Display object state information for both simple and complex objects in a clear way
- Intuitive and user-friendly navigation of the object tree
- Handling of cyclic objects
- Reusable and extensible code
- A simple interface
- Independent of the client implementation

2 Final Words

This project is intended to demonstrate the concepts of Design by Contract, as well as good Object-Oriented design. Therefore, in addition to following all good coding conventions, the code must have a sensible and sufficiently abstracted structure, and must make use of contracts to verify correct behavior.