Isaac Ahn (esa9405@cs.washington.edu)
1429219
Project Proposal
March 31, 2016

MR (Method Repair) Tool

Background

In many programming scenes, developers are frequently required to make changes to implementation of existing methods in order to fix bugs or allow new features to their programs. To do so, the developers would often choose to change inputs, by reparametrizing–add or remove inputs, or change input types of the method–or modify output types. After then, they would soon notice that they now have to search through every compiler error messages so that they could fix all super and sub implementations of the method. MR tool will be a smart tool that could auto-complete this task.

Overview

Our tool, MR Tool, is a tool for Eclipse which will allow developers to easily modify an existing methods. MR Tool mostly targets developers who needs to make changes to a method that has already been implemented a lot elsewhere. The biggest idea behind this tool is to apply changes to all super and sub implementations of a method whenever the method is modified, allowing easy reparametrization of a method. Here is an example:

Assume that there is an overridden method that takes in three int parameters: @Override public int sum(int a, int b, int c). A change has been made so that the method takes only two parameter, by removing the second parameter: @Override public int sum(int a, int c). Then, the tool will change its super class implementation so that it takes only two int parameters. Also, if the method was being used in main and called sum(1, 2, 3), it will automatically be modified to sum(1, 3) with the removal of the second parameter.

This tool will also implement some highly related functions, such as method locators, object casting, and constructor auto-generator. These features will be discussed in additional features section.

Additional Features

1. Method locator
   While Eclipse allows clients to locate a method's super implementation, implementation, or its declaration using a ctrl key (or command key on mac), it doesn't allow clients to know where the method is actually being used. MR Tool will enable clients to list out specifically where each function is used (or sub implemented) with the class names and line numbers to further simplify the method modifying task for developers.

2. Object casting
   Oftentimes, modifying function involves a creation of new object types. For example, if we are currently outputting an integer from a method, but we are modifying it so that it outputs both an integer and a boolean from the method, we would have to create a new object type – let's call it a Pair – that contains both an integer and boolean. MR Tool will be modifying super and sub implementations to take Pair. For places where the method was already in use, MR Tool will automatically determine which value (or field) from the new object type and use it so that it doesn't need any more fix.

3. Constructor auto-generator
   This feature is already implemented in Eclipse. However, MR Tool will allow more advanced version of this so that any time a new field is added to the class, then the class constructor's parameters and implementation gets updated, thus a developer can make changes faster within a class.

Potential Technology

We should be able to use some scripting language so that our tool constantly gets updated with the source code and the modifications. Then, we should be able to use some searching algorithms over the source code so that we know which methods are implemented and where they are being used. Also, we might be doing some string equality checks over the source code to see which changes were made and how the tool can update the code to make optimal changes.

Challenges and Risks

The most challenging aspect of this tool will be applying all these functionalities without causing another bug in the code. Since all of these functionalities will make changes to the source code, it is very important that they will not cause any compiler errors. Moreover, it will be quite significant to keep up with runtime errors that might be caused from the tool's modifications. It might be a good idea to add an "undo" functionality so that, in case the tool's modification resulted in an error, a client can just undo the modification.