

### SOFTWARE EVOLUTION SERIES 2

# Report

December 14, 2017

Students: Cornelius Ries 11884827 Piotr Kosytorz

Tutor: Riemer van Rozen Course: Software Evolution

### 1 Introduction

11876964

This documents contains our notes and answers to the questions about software metrics (practical lab Series 2).

Rewrite the introduction

### 2 Definitions

- Clone (reference to kamiya2002ccfinder)
- Clone class (reference to kamiya2002ccfinder)
- Clone type-1 (provide reference)
- Clone type-2 (provide reference)
- Clone type-3 (provide reference)
- The biggest clone
- The biggest clone class

# 3 Algorithm design

- General algorithm design (sketch: text description)
- Type-1 clean method (pseudocode)
- ullet Type-2 clean method (pseudocode)
- Type-3 clean method (pseudocode)
- Formalised algorithm (pseudocode with references to type-1, type-2 clean methods)
- Describe how the algorithm is covered in the literature (and provide references).

### 3.1 Code duplication detection algorithm

The idea and algorithm of our duplication detection is based on the information from [2] and [1]. The main idea behind this approach is to hash the nodes of an ast into different buckets and collect the duplications if a bucket has more than 1 element. For type 2 the papers suggest to clear unnecessary information from the nodes (variable names, type etc.).

For our implementation we decided to use a map as a utility to do the matching. We also had to clean the nodes initially because of a change in rascal that shifted the loc and other informations of a node from annotations on the node to information contained in the node. This messed up the matching because every location was unique.

#### 3.2 The main algorithm

```
Build the AST of the project.
For all nodes in AST if size > threshold
- Clean nodes for type 1 detection.
- Clean nodes for type 2 detection.
- Collect nodes in map with cleaned node as key,
  relation of original node and location as value
For all keys in Map build a set of duplications
- Collect all values
- If size of values > 1 add to set
Filter subclones
- For all duplications
- If another duplication exists for which all locations
   include the locations of the current one Then
  Else
    Add to new Set
For all filtered clones
```

## 4 Main program validity

- Collect them in output format

Write about automatic tests - describe what the tests do (be convincing)

All tests are in separate files that extend their original rascal module:

- DuplicationsAnalyzerTests
- RaterTests
- UtilsTests
- VolumeAnalyzerTests

To run the tests, import all the modules above and execute :test in the rascal console. The projectLocation in Configuration.rsc has to be set to the projects location in your eclipse!

### 5 The tool

- General description and purpose, used solutions (webserver, REST api, ReactJS app, d3 graphs, etc)
- The 3 main requirements (your tool satis es from the perspective of a maintainer) [ref to storey1999cognitive]:
  - 1. Get a comprehensible overview (of code quality and duplications)
    - Readable table
    - SIG maintanability index
  - 2. Get a deep insight into the clones
    - navigate easily through them
    - provide and extended search function (easy to use = we reduce eort)
  - 3. See how the clones spread over my project (=Improve comprehension )
    - provide insightful visualization = Provide eective presentation styles (graph)
    - show how the les containing clones relate to each other (=indicate options for further exploration)
- Implementation choices
- The visualisation (constellation) how does it help a maintainer or a developer?

### 6 Tool manual

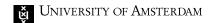
- How to run
- How to use
- Where is the Rascal project
- Where is the original visualisation project

To use the tool we provide the source code as a eclipse project

#### 6.1 How to run

In order to run the program, please follow the steps:

- 1. Please import the project into your eclipse with a working rascal installation.
- 2. Open Configuration.rsc and adjust the location of the projectLocation to match the path of the project to your eclise
- 3. Do the same for the smallSqlProject and hqSqlProject
- 4. Start a rascal console and import the Main module
- 5. run startServe();
- 6. open a browser and point it towards http://localhost:5433 or to the location of serveAddress in case you changed it



# 7 Summary

- Sum things up
- Write how the tool performs on hsqlsb,
- Write how it can be improved

### References

- [1] Ira D Baxter et al. "Clone detection using abstract syntax trees". In: Software Maintenance, 1998. Proceedings., International Conference on. IEEE. 1998, pp. 368–377.
- [2] Flavius-Mihai Lazar and Ovidiu Banias. "Clone detection algorithm based on the Abstract Syntax Tree approach". In: Applied Computational Intelligence and Informatics (SACI), 2014 IEEE 9th International Symposium on. IEEE. 2014, pp. 73–78.