



SOFTWARE EVOLUTION SERIES 2

Report

December 14, 2017

Students:

Cornelius Ries
11884827

Piotr Kosytorz
11876964

Tutor:

Riemer van Rozen

Course:

Software Evolution

1 Series 2

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

1.1 About

TODO: Write the "about" section

1.2 Design Decisions

TODO

1.3 Results

TODO

1.4 Tool usage

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

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 eclipse
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

1.5 Duplication Detection

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.

A more detailed explanation can be found in the next chapter.

1.5.1 How it works (Pseudocode)

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
-
- Else
- Add to new Set

For all filtered clones

- Collect them in output format

1.6 Visualization

TODO

1.7 Tests

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!

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 Baniias. “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.