Storm Cloud Development

Project CM Metrics

Version 1.0

Revision History

Date	Version	Description	Author
15/May/2014	1.0	Initial creation of the document	Storm Cloud Development

Table of Contents

Metrics

1. Metrics

A metric is a measurement scale for coding. There are several different methods that can be used to classify the code. This document describes two methods that are used for *ProjectCM*.

1.1 Brief Description

The metrics for *ProjectCM* are calculated with every travis-ci build. So the developer has the possibility to check the metrics after every build on https://travis-ci.org/tobias-engelmann/ProjectCM in the console output of the worker.
Example build: https://travis-ci.org/tobias-engelmann/ProjectCM/jobs/25241022

Also it is possible to run the metric checks manually via bash. You must switch to the “metrics” folder. Then you run “bash metricsAll.sh” in your console. The output is stored to the folder “result” in the “metrics” folder. If you want to have the output directly in your console, run “bash metricsAll.sh console”. Also it is possible to check a single metric. Just replace “metricsAll with the name of the file you want to execute”. The bash files can be found here: https://github.com/tobias-engelmann/ProjectCM/tree/master/metrics

The two metrics we have a look on are the Cyclomatic Complexity and the Maintainability Index.

1.2 Cyclomatic Complexity

The Cyclomatic Complexity is an indicator for the number of decisions to make when going through a specific code part. Every decision which is made like “if”, “elif”, “assert” or even Boolean operators raises the counter by one. This means that a higher score indicates a worse Cyclomatic Complexity as there are more paths to go and the code part is more complex.
The output of the result of the tool we use is structured as followed:

```
type lineNumber:columnOffset fullName - ranking(score)
```

As our system has no bad rating, it is not possible to show an example of bad code. So we would explain the metric based on a good piece of code:
https://github.com/tobias-engelmann/ProjectCM/blob/master/src/adressbook/views.py

If you check the methods, each method is with a low complexity, as there is usually only one decision to be made. So the tests for the functions are easy and it is not necessary to write many tests to check all possible paths.

1.3 Maintainability Index

The Maintainability Index shows how maintainable the source code is. Maintainable means how easy it is to change and support the source code. The Maintainability Index is a calculation based on the source lines of code, the Cyclomatic Complexity and the Halstead Volume. The Halstead Volume indicates how big the actual implementation of a specific program code part is.
The output of the result of the tool we use is structured as followed:

```
fileName – ranking
```

As our system has no bad rating, it is not possible to show an example of bad code. So we would explain the metric based on a good piece of code:
https://github.com/tobias-engelmann/ProjectCM/blob/master/src/adressbook/views.py

As you can see, each method is short and has a specific functionality. The file size is ok, as it is easy to keep the overview over all methods. This helps to find bugs easy without too long searching and it is also possible to add new features easily.