COMPLEXITY METRICS

There are many different ways of measuring the complexity of a project. We can analyse the complexity through more specific ways like per method or class, or more general ways like per package, module or the project itself.

Method metrics

Cognitive complexity (CogC) – It can be described as a measure of how difficult a portion of code is to read and intuitively understand.

Cyclomatic Complexity (v(G)) – It's a count of the linearly independent paths through source code. In other words, it can be described as the number of decisions a given block of code needs to make. It can also serve as a way to measure how difficult a certain portion of code is to be tested.

Essential Cyclomatic Complexity (ev(G)) – It tells how much complexity is left once we have removed the well-structured complexity.

Class Metrics

Maximum Operation Complexity (OCmax) – Shows the maximum complexity present in a certain class.

Average Operation Complexity (OCavg) – Determines the average complexity of operations in a certain class.

Weighted Method Complexity (WMC) – Measures the complexity of a certain class. Its value can be determined by the cyclomatic complexities of its methods.

Package, Module and Project Metrics

As the name suggests, these metrics are related to packages, modules or the project itself respectively. All of these 3 types of metrics use the same type of measurements:

Average Cyclomatic Complexity (v(G)avg) – It describes the average complexity of a certain file by analysing the complexity of its functions. It can be determined by the sum of the cyclomatic complexities of all function definitions divided by the number of function definitions in the file.

Total Cyclomatic Complexity (v(G)tot) – It describes the total cyclomatic complexity of a certain package, module or project.

JabRef’s overall analysis

When evaluating the Cyclomatic Complexity of a certain project, it’s common to use the following range values:

v(G)avg below 4 – Good complexity

v(G)avg below 5 and 7 – Medium complexity

v(G)avg below 8 and 10 – High complexity

v(G)avg above 10 – Extreme complexity

By analysing the overall complexity of the JabRef project, we can see that the average Cyclomatic Complexity is about 1,79, which means that JabRef has a very good complexity overall.

RELATION BETWEEN COMPLEXITY AND CODE SMELLS

Although JabRef has a pretty good overall complexity throughout the whole project, there are some exceptions. For example, some methods and classes have extremely high complexity values. Here are some cases:

In method metrics:

formatName(Author,String,Warn) : present in jabref -> src -> main -> java -> org.jabref -> logic -> bst -> BibTexNameFormatter.java, which has a cognitive complexity (CogC) of 154.0, when the usual maximum value for cognitive complexity should be about 15.

In class metrics:

RTFChars.java : present in jabref -> src -> main -> java -> org.jabref -> logic -> bst, which has an average operation complexity (OCavg) of 25.0, maximum operation complexity (OCmax) of 46 and Weighted method complexity (WMC) of 75, which are all very high values of complexity.

Possible code smell relation:

After analysing some of these examples from both methods and classes, the main conclusion is that most of these are all related to a common code smell, in this case, the “Long method” code smell. This can be explained by the fact that usually, the longer a method is, the higher is the probability that this method is responsible for a great variety of functionalities, therefore increasing its level of complexity.