Complexity Metrics – GanttProject

As Complexity Metrics, como o próprio nome indica, permite calcular a complexidade de código de um projeto. Quanto mais alta é a complexidade, mais difícil se torna não só a compreensão do código em si, mas, também, a sua manutenção e, portanto, deve-se apontar para valores mais baixos.

Neste relatório decidimos de analisar as métricas de complexidade aplicadas ao nível da classe. Estas são analisadas de acordo com três fatores sobre os quais iremos descrever posteriormente, assim como os seus valores no nosso projeto.

* WMC (Weighted Method Complexity)

### Este parâmetro refere-se à soma ponderada de métodos, cujos parâmetros de ponderação podem ser, nomeadamente, a complexidade CC (McCabe Cyclomatic Complexity), que é calculada a partir do número de decisões feitas (if/else, for, while, etc.), as linhas de código ou igual a um (no caso de WMC não ponderada). Portanto, quanto maior for este valor, maior é a complexidade da classe.

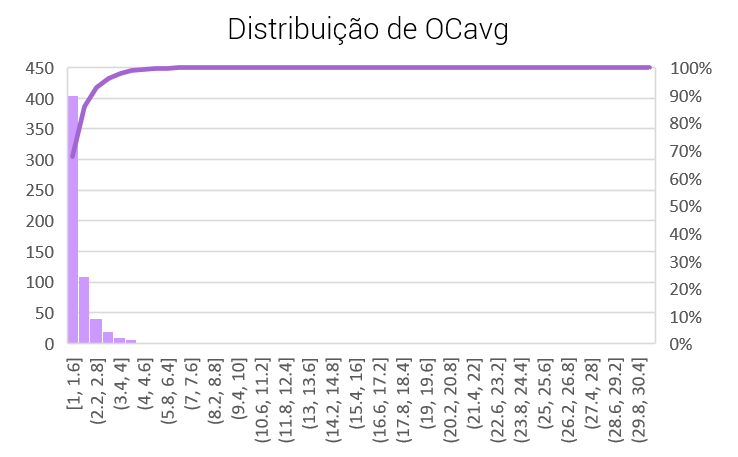
### No nosso projeto, os valores de WMC variam entre 0 e 173, sendo a média de cerca 13.8. Assim, podemos verificar que o valor máximo é mais de dez vezes superior à média, sendo que a maioria dos valores se concentram no intervalo de valores [0, 7.3], como se pode observar no seguinte gráfico.

### De forma a compreendermos a razão por detrás da dispersão de valores acima da média, podemos analisar, por exemplo, a classe net.sourceforge.ganttproject.task.TaskManagerImpl que corresponde ao valor máximo de WMC.

### Esta classe é responsável pela gestão das tarefas do GanttProject e, portanto, não só contém imensos métodos, mas, também, muitos destes métodos necessitam de fazer várias decisões e têm um número significativo de linhas. Por exemplo, no método createLength, que ocupa cerca de 94 linhas, e que toma várias decisões, nomeadamente do tipo for, if/elseIf/else e switch/case.

* COavg (Average operation complexity)

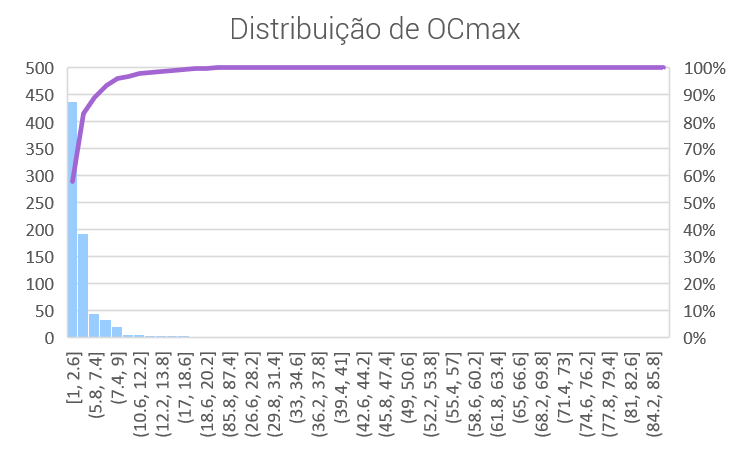
Este parâmetro refere-se à média da complexidade das operações.

No nosso projeto, os valores de COavg variam entre 1 e 31, sendo a média de cerca 1.6. Assim, podemos verificar que o valor máximo é quase trinta vezes superior à média, sendo que a maioria dos valores se concentram no intervalo de valores [1, 1.6], como se pode observar no seguinte gráfico.

Se observarmos, por exemplo, a classe relativa ao valor máximo de COavg que, neste caso, é net.sourceforge.ganttproject.GanttOptions.GanttXMLOptionsParser, podemos verificar a causa por detrás deste valor. Esta classe, tem um método chamado startElement, que tem uma complexidade elevada, tendo em conta que ocupa cerca de 210 linhas onde são feitas várias decisões (if/else, switch/case).

* COavg (Average operation complexity)

Este parâmetro refere-se à média da complexidade das operações.

No nosso projeto, os valores de OCmax variam entre 1 e 86, sendo a média de cerca 3.2. Assim, podemos verificar que o valor máximo é quase trinta vezes superior à média, sendo que a maioria dos valores se concentram no intervalo de valores [1, 2.6], como se pode observar no seguinte gráfico.

O valor máximo de OCmax é relativo à mesma classe associada ao valor máximo de COavg e razão pela qual este é significativamente superior à média é a mesma que foi dada anteriormente.

Se analisarmos outra classe, por exemplo, biz.ganttproject.impex.csv.GanttCSVExport que tem um valor de 22 (segundo maior valor para OCmax), podemos verificar algo semelhante, métodos longos que contêm várias decisões.

Em suma, a complexidade nas classes está normalmente relacionada com o número de métodos que esta contém, assim como, o tamanho e quantidade de decisões de cada um.

Podemos verificar que, em geral, o GanttProject tem complexidades baixas, exceto em certas situações onde há dispersões, estas podem estar associadas a code smells como Long Method e Large Class.

Referências:

* <https://www.aivosto.com/project/help/pm-complexity.html>
* <http://www.arisa.se/compendium/node97.html>
* https://refactoring.guru/refactoring/smells