**MOOD Metrics – GanttProject**

As MOOD Metrics permitem calcular a qualidade, em geral, de um projeto orientado a objetos, sendo úteis por descreverem-no como um todo. Assim, só estão disponíveis ao nível do projeto.

* *Attribute Hiding Factor* (AHF)

Mede o encapsulamento dos atributos dentro de uma classe, ou seja, a média de vezes que uma variável se encontra “escondida” das diversas classes em que não foi declarada. Deste modo, quanto maior o AHF, mais corretamente se estão a usar as variáveis e a manter o encapsulamento exigido para estas. Um AHF baixo, deve gerar preocupação sobre a implementação, dado que significa pouco encapsulamento.

Para calcular essa média, são contadas as classes em que um determinado atributo se encontra declarado e é feita, posteriormente, a divisão pelo número de classes existentes no projeto.

No nosso projeto, o valor é de, aproximadamente, 90%, o que significa que os atributos estão encapsulados nas suas classes, existindo apenas alguns casos em que tal não se verifica. Exemplo disso, são algumas variáveis que se encontram declaradas como protected.

* *Attribute Inheritance Factor* (AIF)

Mede a quantidade de atributos herdados de superclasses, com base em todos os atributos declarados na classe que os herda. Idealmente, visto que os atributos devem ser declarados como privados, o AIF deve ser zero e, no pior dos casos, menor do que cerca de 50%. No entanto, os valores aceitáveis dependem do padrão de desenho utilizado para o projeto. Se a classe herda atributos e não os redefine, então o AIF será mais alto, verificando-se o contrário quando a classe redefine esses mesmos atributos.

No nosso projeto, o valor é de, aproximadamente, de 75%, o que mostra um mau uso de herança sobre atributos. Os atributos não devem ser acessíveis fora das classes em que foram declarados. No entanto, mais uma vez, o valor de AIF mostra que foi feito o contrário. Este resultado também explica a razão por detrás do valor de AHF, anteriormente obtido e discutido.

* *Coupling Factor* (CF)

Mede os acoplamentos (i.e. chamadas de métodos, acesso a variáveis) presentes entre as classes definidas no projeto. CF não inclui herança de classes, pelo que o seu valor apenas reflete relações de classes. Classes que estabelecem um acoplamento bidirecional, têm um efeito duas vezes maior, do que acoplamentos unidirecionais. É de notar que acoplamentos aumentam a complexidade, reduzem encapsulamento e reutilização de código e limitam, ainda, a manutenção e compreensão do mesmo. Assim, mesmo que esperada a interação entre classes, o valor de CF deve ser baixo.

No nosso projeto, o valor é de, aproximadamente, 2%, o que revela muito poucos acoplamentos entre classes. Isto é excelente, visto que demonstra a inexistência de dependências diretas entre classes que, por diversas razões, podem vir a ser apagadas, alteradas, etc.

* *Method Hiding Factor* (MHF)

Mede a funcionalidade dos métodos implementados numa classe. Quanto maior essa funcionalidade/utilidade, menor será o valor de MHF. No entanto, um valor muito baixo indica uma implementação pobre em abstrações, pois um número elevado de métodos “específicos” gera uma maior probabilidade de erros. Um valor muito alto para MHF, indica um maior número de métodos que não se encontram disponíveis para serem reutilizados e, portanto, existe pouca funcionalidade nestes.

No nosso projeto, o valor é de, aproximadamente, 45%, o que revela a existência de alguns métodos, existentes na implementação do projeto, aos quais falta funcionalidade/utilidade. Não é surpreendente, tendo em conta que existe Dead Code em algumas classes (biz.ganttproject.core.src.main.java).

* *Method Inheritance Factor* (MIF)

Mede a quantidade de métodos herdados de superclasses, com base em todos os métodos implementados na classe que os herda. Idealmente, o MIF deve encontrar-se entre 20% a 80%. No entanto, os valores aceitáveis dependem do padrão de desenho utilizado para o projeto. Se a classe herda métodos e não os reimplementa, então o MIF será mais alto do que aquela que redefine o que herda.

No nosso projeto, o valor é de, aproximadamente, 50%, o que revela uma boa herança entre classes (i.e. abstração encontra-se equilibrada).

* Polymorphism Factor (PF)

Mede o grau com que os métodos dão Override nas classes que herdam esses mesmos métodos. Se todos os métodos sofrem Override, então o valor de PF deve ser 100%. Um valor muito baixo é indicativo da falta de polimorfismo, na implementação do projeto, ou falta de herança entre classes. Um valor muito alto é indicativo de código complexo, dado o facto de que se “abusou” do polimorfismo.

No nosso projeto, o valor é de, aproximadamente, 30%, o que revela a existência de polimorfismo. Apesar de ser um valor relativamente bom, provavelmente ainda se podia exigir um pouco mais de polimorfismo.

Em suma, as MOOD metrics são congruentes entre si. Indicaram também, que o desenho do projeto não é o melhor, em alguns aspetos, existindo alguns erros de modelação e de implementação. No entanto, esses não são graves o suficiente para gerar uma implementação errada para este projeto poder ser mantido e compreendido.

NOTA: no ficheiro xlsx não estão desenhados gráficos, porque só existem uma linha para cada

metric, o que se traduz em função constante.

REFERÊNCIAS

<https://www.aivosto.com/project/help/pm-oo-mood.html> (1 dezembro 2022)