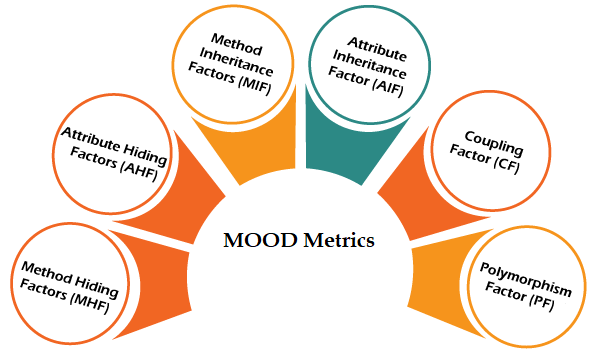
**MOOD Metrics**

João Miguel Lopes Romão Esteves - 47994

* Attribute Hiding Factor (AHF)
* Attribute Inheritance Factor (AIF)
* Coupling Factor (CF)
* Method Hiding Factor (MHF)
* Method Inheritance Factor (MIF)
* Polymorphism Factor (PF)

**Attribute Hiding Factor (AHF)**

O fator de ocultação dos Atributos mede como as variáveis são encapsulados em uma classe. A visibilidade é contada em relação a outras classes. AHF representa a média de ocultação entre todas as classes do sistema. Um Atributo privado está totalmente oculto.

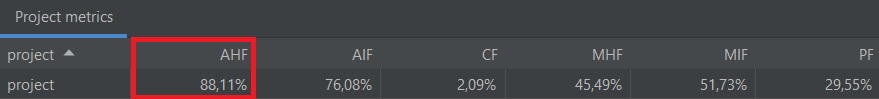
Este fator pode ser calculado pela seguinte fórmula:

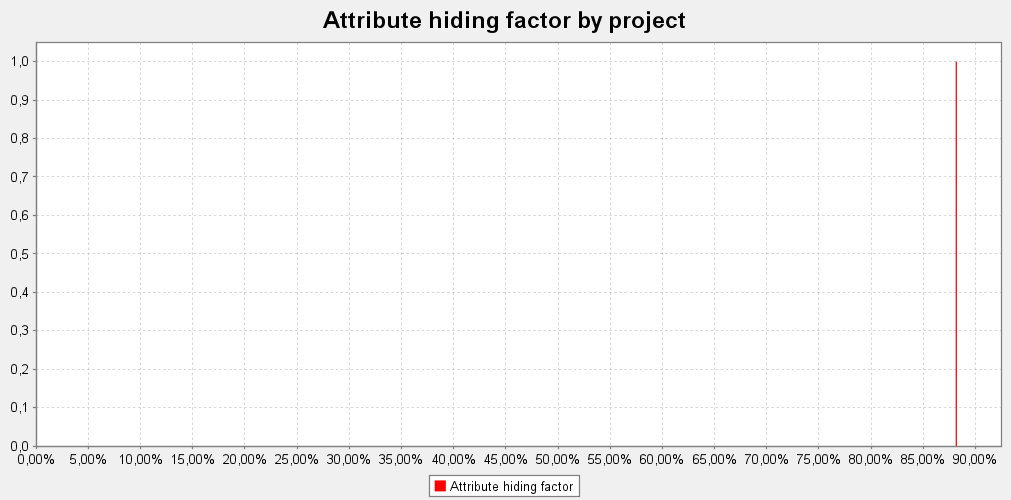
Uma imagem com texto

Descrição gerada automaticamente**Ah(Ci) =** Atributos ocultos na classe Ci

**Ad(Ci)** = Av(Ci) + Ah(Ci): Atributos definidos em Ci

**Av(Ci):** Atributos visíveis na classe Ci

**TC:** Número total de Classes.



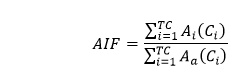
Geralmente, um valor alto de AHF é aconselhado, na medida em que os atributos de uma classe devem ser ocultos de outras classes, de modo que 100% é o valor ideal do AHF.

Em análise ao nosso programa, como temos um valor de 88,1%, podemos concluir que a maior parte dos nossos atributos são ocultos para outras classes, sendo utilizados apenas nas classes onde foram criados, e com isto temos uma menor complexidade do nosso programa.

**Attribute Inheritance Factor (AIF)**

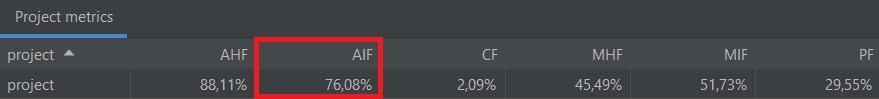
Em Herança, o filho ou a subclasse herda as propriedades (Atributos) da classe mãe ou da superclasse. A extensão em que esses atributos são herdados é definida pelo Fator de Herança do Atributo (AIF).

Este fator pode ser calculado pela seguinte fórmula:

**Ah(Ci) =** Atributos herdados

**Aa(Ci)** = Ad(Ci) + Ah(Ci): Atributos definidos em Ci

**Ad(Ci):** Atributos definidos na classe Ci

**TC:** Número total de Classes.

Uma imagem com mesa

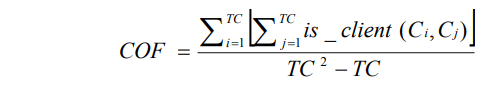
Descrição gerada automaticamente

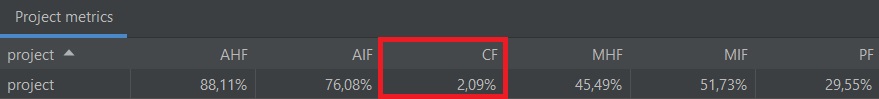
Geralmente, a faixa de AIF está entre 0% e 48%. De acordo com o nosso programa onde temos uma percentagem de 76,08%, podemos concluir que as classes filho herdam um grande número de atributos de sua classe mãe.

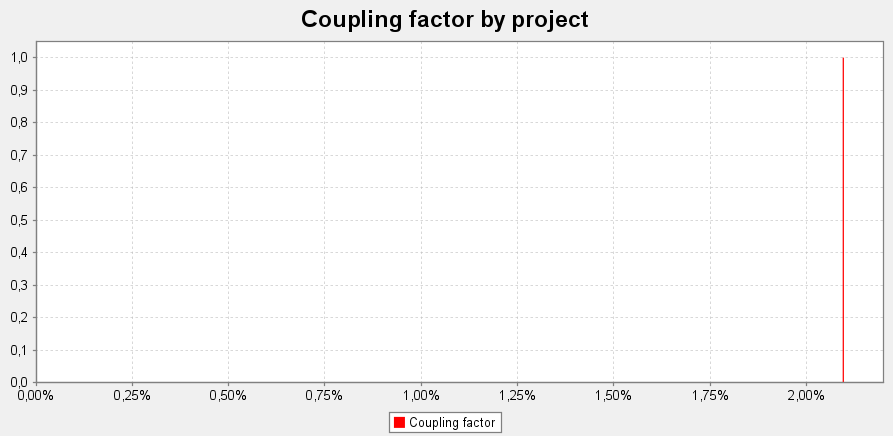
**Coupling Factor (CF)**

O Fator de Acoplamento (CF) é a razão entre o acoplamento real entre diferentes classes e o acoplamento máximo possível que pode acontecer no sistema. Se uma classe pode acessar aos atributos da segunda classe, então diz-se que a primeira classe é acoplada à segunda classe.

Este fator pode ser calculado pela seguinte fórmula:

**is\_client**(Cc,Cs) =| 1 **if** (Ci⇒Cj)^(Ci≠Cj) , **else** 0

**TC:** Número total de Classes.



O alto valor de CF mostra que as classes do sistema são mais interconectadas e interdependentes, no leva ao problema de que às vezes é muito difícil alterar ou reparar o sistema em caso de qualquer bug ou problema, porque a funcionalidade em que o bug está, poderia ser implementada por mais de duas classes e temos de fazer alterações em todas as classes relacionadas.

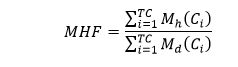
Em análise ao nosso programa o valor de CF é de apenas 2.09% , o que nos permite concluir que irá facilitar-nos em caso de bugs ou de novas implementações.

**Method Hiding Factor (MHF)**

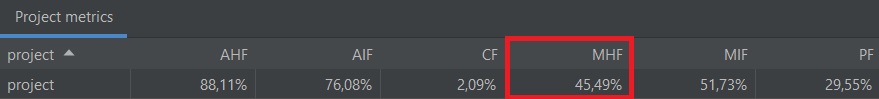
O fator de ocultação dos Métodos mede como estes são encapsulados em uma classe. A visibilidade é contada em relação a outras classes. MHF representa a média de ocultação entre todas as classes do sistema. Um Método privado está totalmente oculto.

Este fator pode ser calculado pela seguinte fórmula:

**Mh(Ci) =** Métodos ocultos na classe Ci

**Md(Ci)** = Mv(Ci) + Mh(Ci): Métodos definidos em Ci

**Mv(Ci):** Métodos visíveis na classe Ci

**TC:** Número total de Classes.

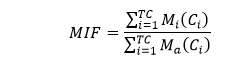
Um MHF baixo indica uma implementação insuficientemente abstrata. Uma grande proporção de métodos está desprotegida e a probabilidade de erros é alta. Um MHF alto indica muito pouca funcionalidade. Pode também indicar que o desenho ou modelo inclui uma alta proporção de métodos especializados que não estão disponíveis para reutilização. Um valor aceitável de MHF é de 8% a 25%.

Em concordância com o nosso programa temos um valor de 45,49% de MHF, em que nos permite utilizar e reutilizar uma boa quantidade de métodos, mas que a implementação também é suficientemente abstrata, no que resulta uma boa funcionalidade do programa.

**Method Inheritance Factor (MIF)**

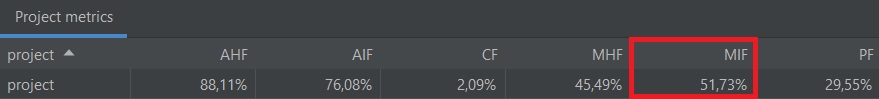
Em Herança, o filho ou a subclasse herda as propriedades (Métodos) da classe mãe ou da superclasse. A extensão em que esses métodos são herdados é definida pelo Fator de Herança do Método (AIF).

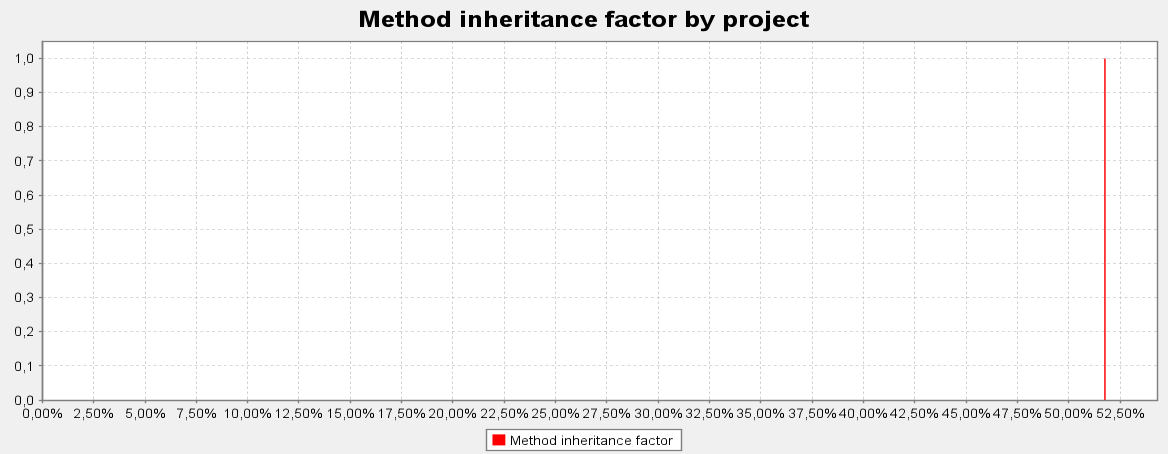
Este fator pode ser calculado pela seguinte fórmula:

**Mi:** Métodos herdados

**Ma(Ci)** = Md(Ci) + Mi(Ci): Métodos definidos em Ci

**Md(Ci):** Métodos definidos na classe Ci

**TC:** Número total de Classes.

À primeira vista, podemos ser tentados a pensar que a herança deve ser usada extensivamente. No entanto, a composição de várias relações de herança constrói um grafo acíclico direcionado (árvore de hierarquia de herança), cuja profundidade e largura fazem com que a compreensibilidade e a testabilidade desapareçam rapidamente. Geralmente, a faixa de MIF está entre 20% a 80%****.

De acordo com o nossos valores, temos um MIF de 51,73%, o que nos dá uma boa compreensibilidade e testabilidade do nosso programa.

**Polymorphism Factor (PF)**

No polimorfismo, a classe filho pode implementar o método de uma maneira diferente. O mesmo método pode ser implementado de maneiras diferentes na classe filho e na classe mãe. Ele é definido pela razão entre um número real de substituições do método e o número máximo de substituições totais do método.

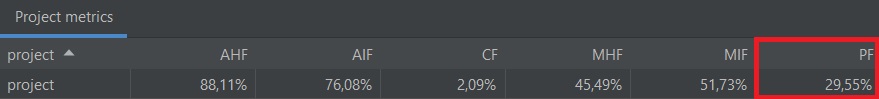
Este fator pode ser calculado pela seguinte fórmula:

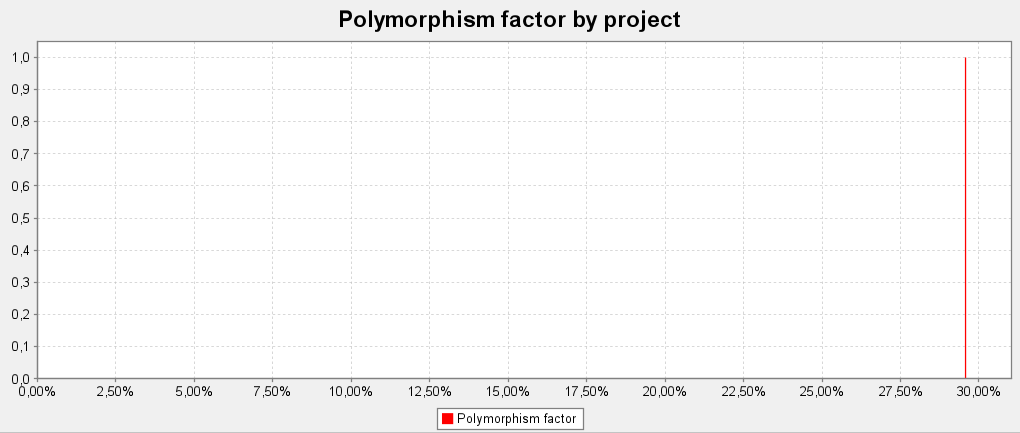
**Mo(Ci):** Métodos substituídos na classe Ci

**Uma imagem com texto

Descrição gerada automaticamenteMn(Ci):** Novos Métodos em Ci

**DC(Ci):** Número de descendências da classe Ci (classes derivadas)

**TC:** Número total de Classes.



O polimorfismo surge da herança e tem seus prós e contras. Podemos intuitivamente esperar que o polimorfismo (Overrides) possa ser usado em uma extensão razoável para manter o código claro, mas esse código excessivamente polimórfico pode ser muito complexo para entender (porque vários métodos alternativos podem ser executados para uma instrução de chamada). A PF deve estar em uma faixa razoável com um limite inferior e um limite superior.

Ao analisar o PF nosso programa obtivemos um valor de 29,55%, onde podemos concluir que temos um sistema suficientemente claro e limpo com uma complexidade razoável, que nos permite uma melhor compreensão do mesmo.