

4 de dezembro de 2022

Engenharia de Software

GanttProject

Relatório Fase Final

Vasco Malta 58592

Rafael Mira 59243

Bruno David 60011

Mafalda Batalha 60684

Gonçalo Cerveira 61696

<https://github.com/mafaldabatalha/ganttproject.git>

https://www.youtube.com/watch?v=\_NtY4wWpveE

Índice

[1º Fase 2](#_Toc121073022)

[Code Smells 2](#_Toc121073023)

[Bruno David 2](#_Toc121073024)

[Rafael Mira 3](#_Toc121073025)

[Vasco Malta 5](#_Toc121073026)

[Mafalda Batalha 6](#_Toc121073027)

[Gonçalo Cerveira 8](#_Toc121073028)

[Design Patterns 10](#_Toc121073029)

[Bruno David 10](#_Toc121073030)

[Rafael Mira 11](#_Toc121073031)

[Vasco Malta 12](#_Toc121073032)

[Mafalda Batalha 13](#_Toc121073033)

[Gonçalo Cerveira 16](#_Toc121073034)

[2º Fase 18](#_Toc121073035)

[Métricas 18](#_Toc121073036)

[Bruno David - Mood Metrics 18](#_Toc121073037)

[Gonçalo Cerveira - Martin Packaging Metrics 20](#_Toc121073038)

[Mafalda Batalha - Complexity Metrics 24](#_Toc121073039)

[Rafael Mira - Chidamber-Kemerer Metrics 27](#_Toc121073040)

[Vasco Malta - Lines of Code Metrics 35](#_Toc121073041)

[Use Cases 38](#_Toc121073042)

[User Stories 59](#_Toc121073043)

[Pop-up’s na adição de recursos a uma tarefa 59](#_Toc121073044)

[Visualização de recursos no diagrama PERT 59](#_Toc121073045)

[Notas/Problemas 60](#_Toc121073046)

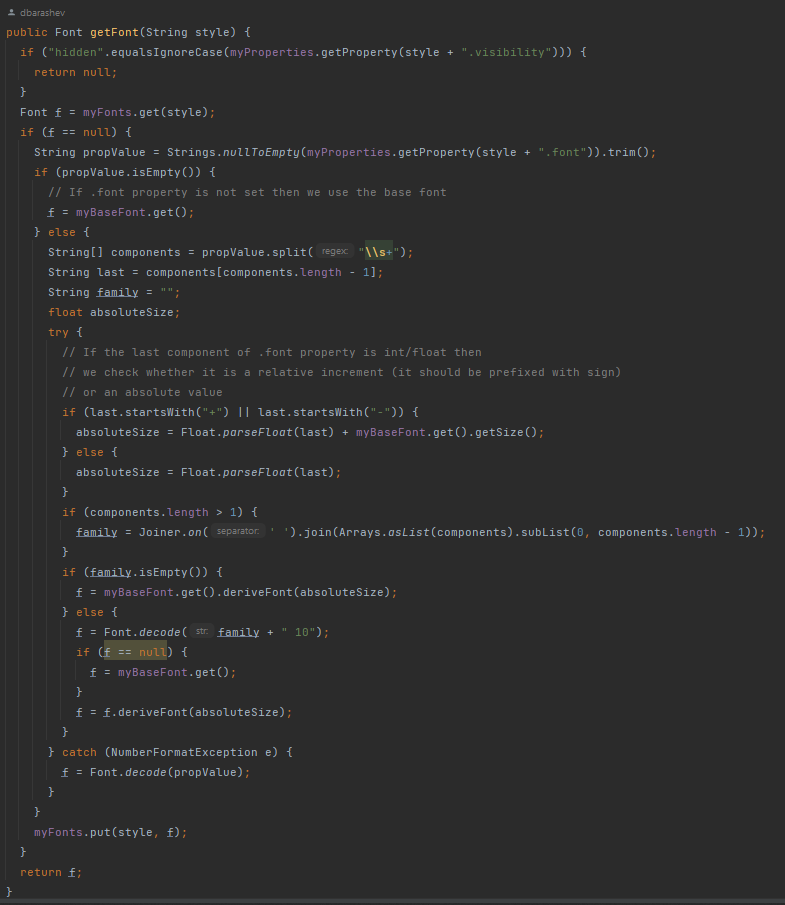
# 1º Fase

## Code Smells

### Bruno David

* Long Method :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/chart/canvas/FontChooser.java

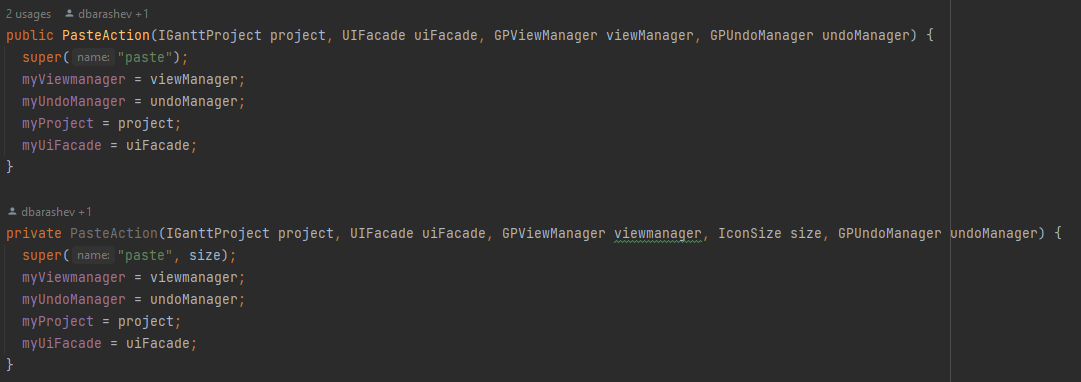
O método getFont é muito longo, sendo mais legível, fracioná-lo em métodos mais pequenos, como, por exemplo, o código a executar em cada condição.

Este code smell foi revisto por Gonçalo Cerveira e Vasco Malta.

* Speculative Generality :

Localização**:** net.sourceforge.ganttproject.action.edit (ganttproject.src.main.java), classe PasteAction

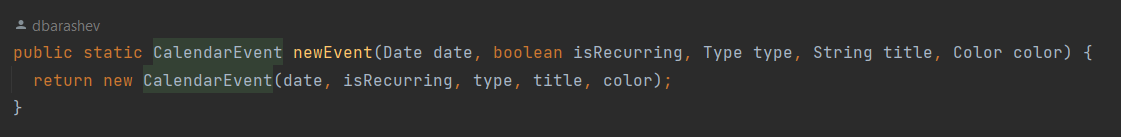
Esta classe tem dois construtores, porém um deles é privado e nunca chega a ser utilizado, provavelmente porque foi criado anteriormente com a expectativa de ser necessário.



Este code smell foi revisto por Vasco Malta e Mafalda Batalha.

* Dead Code :

Localização**:** ganttproject.core/src/main/java/biz/ganttproject/core/calendar/CalendarEvent.java

 O método, a seguir ilustrado, não é usado e, por isso, devia ser eliminado.

Este code smell foi revisto por Gonçalo Cerveira e Vasco Malta.

### Rafael Mira

* Speculative Generality :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/calendar, classe AlwaysWorkingTime-CalendarImpl.java

Nesta classe, existem vários métodos que têm o corpo vazio e, portanto, não fazem nada. Apenas estão na classe, porque esta implementa uma interface (GPCalendar.java) que os declara, mesmo quando não são necessários para a classe em questão.







Pode-se resolver esta situação utilizando o princípio da *Interface Segregation*. A interface GPCalendar continha apenas os métodos comuns a todas as classes que a implementam e, para cada classe, criava-se uma interface específica com os métodos que lhe são exclusivos, evitando, assim, ter métodos que não fazem nada.

Este code smell foi revisto por Bruno David e Vasco Malta.

* Message Chains :

Localização: ganttproject/src/main/java/net/sourceforge/ganttproject/task/dependency/constraint, classe Start-FinishConstraintImpl.java

Na classe, no método GetCollision, pode-se observar a mesma *message chain* por duas vezes.

Em concreto, .

Visto que em ambas as chamadas se está a usar o método , utilizar-se-ia uma variável local manager que recebia o resultado do método referido. O mesmo seria feito para o calendário do manager.

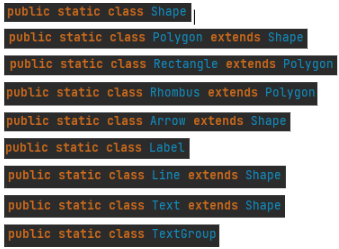
Evita-se, assim, estar a fazer chamadas a métodos dos mesmos objetos por várias vezes, o que diminui a complexidade do código.

Este code smell foi revisto por Vasco Malta.

* Large Class :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/chart/canvas/Canvas.java

Esta classe é um aglomerado de várias classes, contendo 10 no total. O facto de haver 9 classes todas agrupadas dentro de uma, torna a classe principal muito extensa e muito mais complexa do que devia ser.

Para melhorar esta parte do código, podia-se criar as classes fora da Canvas, cada uma num ficheiro separado, para tornar a classe mais pequena, legível e objetiva.

Este code smell foi revisto por Gonçalo Cerveira.

### Vasco Malta

* Comments :

Localização: ganttproject.core.option

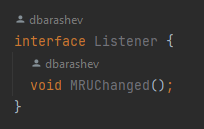
As classes da package acima referida, não apresentam quaisquer comentários, o que é considerado code smell, pois um programador que não conheça o projeto, ou mesmo o programador original, pode ter dificuldade em entender as funcionalidades que foram implementadas.

Este code smell foi revisto por Rafael Mira.

* Speculative Generality :

Localização: net.sourceforge.ganttproject.document(ganttproject.src.main.java), DocumentMRU.java

Nesta classe foi criada uma interface Listener, possivelmente com a necessidade de tal interface posteriormente. No entanto, acabou por nunca ser utilizada.

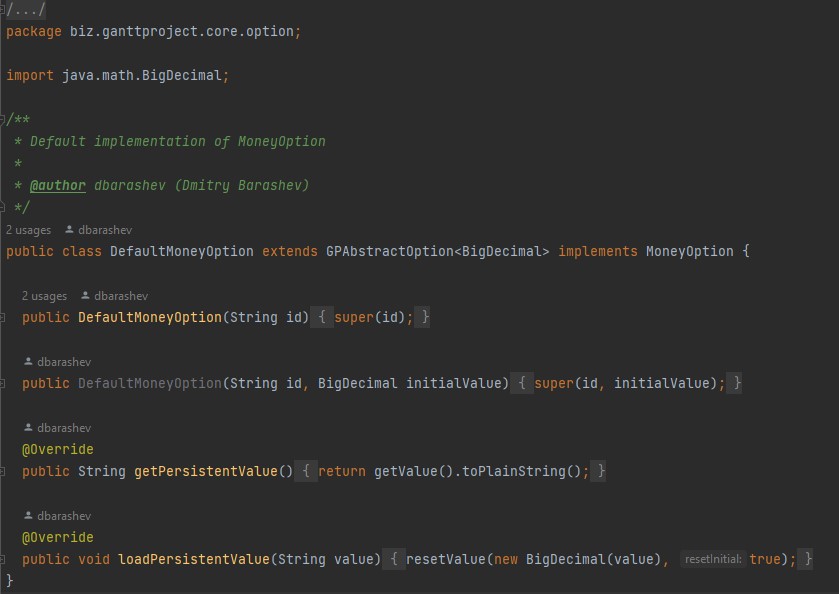


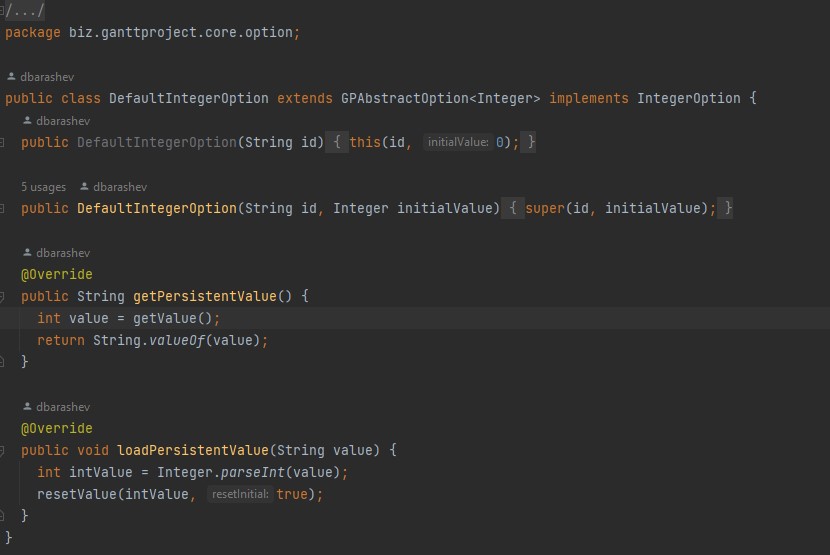
Este code smell foi revisto por Mafalda Batalha.

* Data Class :

Localização: ganttproject.core.option**,** Classes DefaultStringOption, DefaultMoneyOption, DefaultIntegerOption

As Classes acima referidas, são exemplos de code smell, pois são usadas apenas para armazenar dados.





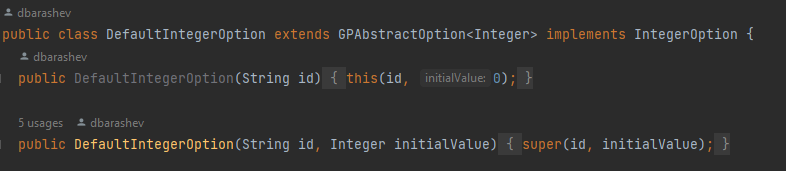


Este code smell foi revisto por Gonçalo Cerveira.

### Mafalda Batalha

* Dead Code :

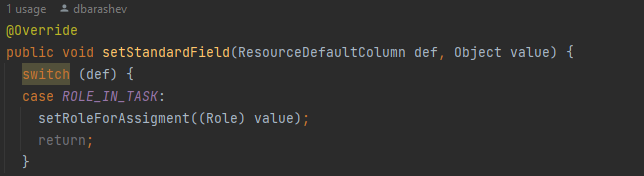
Localização: ganttproject.core.option (biz.ganttproject.core.src.main.java)

A classe DefaultIntegerOption tem dois construtores, porém um deles não é utilizado e, portanto, é um exemplo de Dead Code e deveria ser removido.

Este code smell foi revisto por Bruno David e Vasco Malta.

* Speculative Generality :

Localização: net.sourceforge.ganttproject.resource (biz.ganttproject.core.src.main.java)

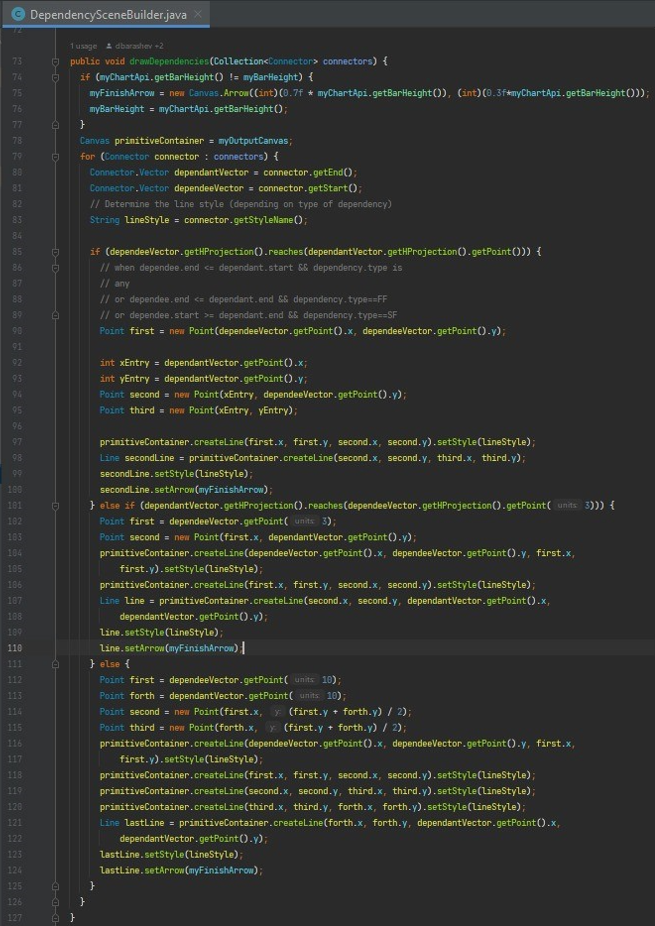
Na classe AssignmentNode, existe um método (setStandardField) que tem um switch statement que contém apenas um case. No caso, seria mais correto usar um if statement, dado que no momento só se está a considerar apenas um cenário.

Este code smell foi revisto por Rafael Mira.

* Long Method :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/chart/scene

A classe DependencySceneBuilder, tem alguns métodos como DrawDependencies e ConnectEndArrow que apresentam um tamanho considerável. Isto causa dificuldade na leitura e compreensão do código e, por isso, seria recomendável repartir em partes, por exemplo, em métodos auxiliares.



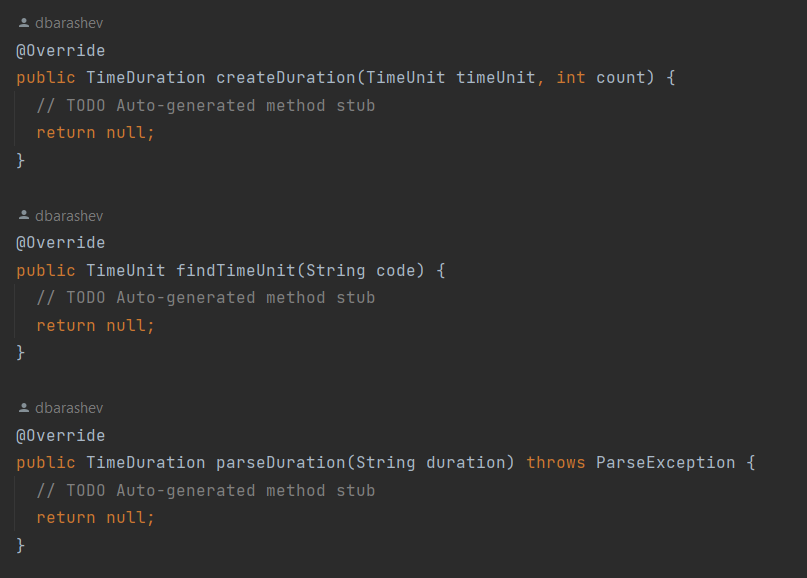
Este code smell foi revisto por Rafael Mira.

### Gonçalo Cerveira

* Speculative Generality :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/time/impl, Classe GregorianTimeUnit-Stack

Nesta classe existem métodos não implementados apenas porque existem na interface, mesmo estes não sendo usados em lado nenhum.

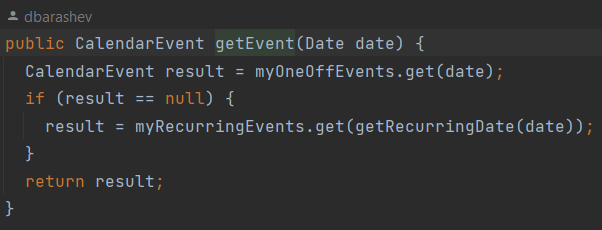
****

Este code smell foi revisto por Bruno David.

* Dead Code :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/calendar/WeekendCalendarImpl.java

Este método não é usado, pelo que deveria ser removido do código.

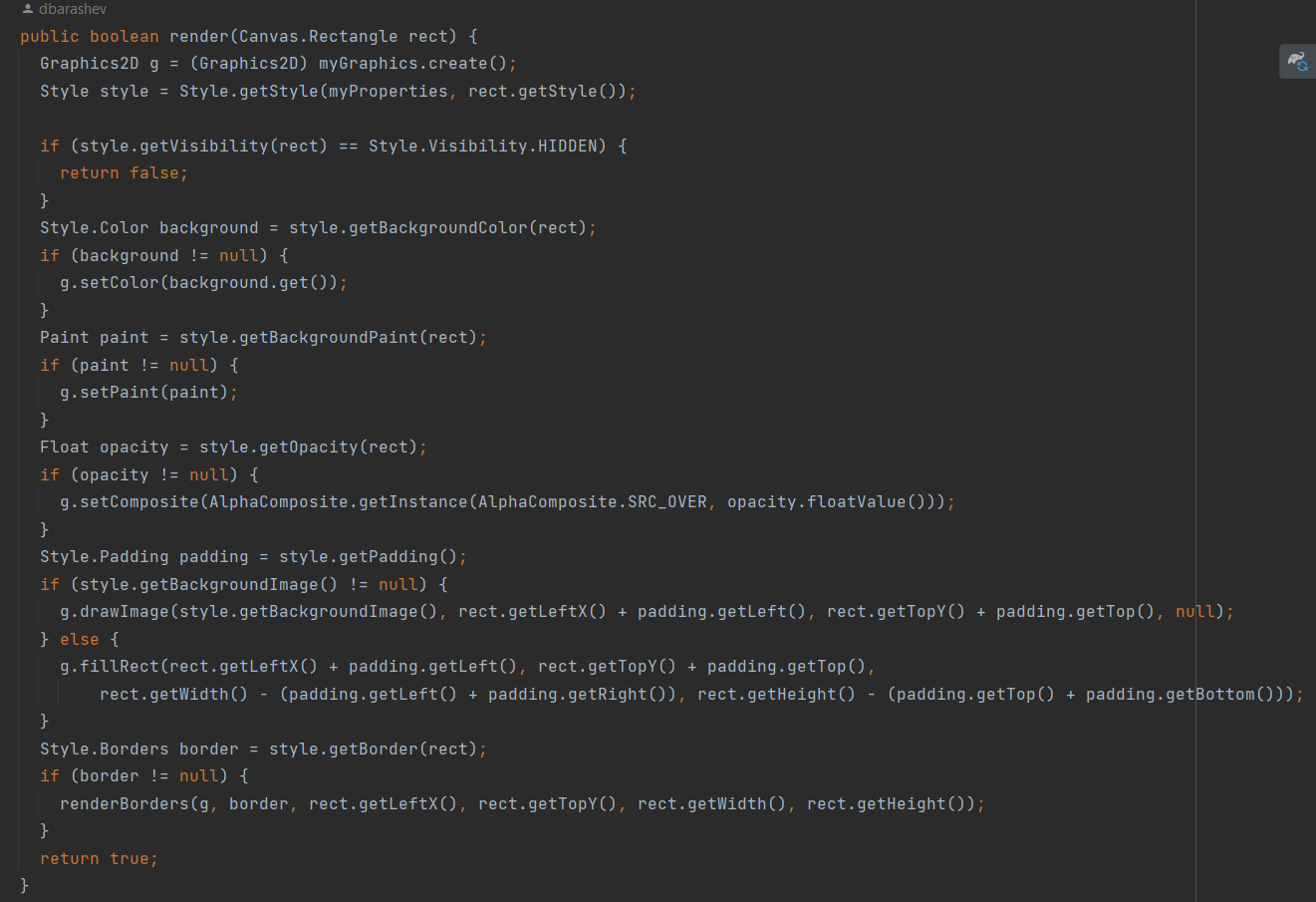


Este code smell foi revisto por Rafael Mira.

* Long Method :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/chart/render/RectangleRenderer.java

Este método é longo, tornando-se confuso para a pessoa que o tenta perceber, uma solução seria fragmentar o código em pequenas partes, conseguindo assim fazer as verificações necessárias para esta método perceptíveis e, caso necessário acrescentar informação mais detalhada, as verificações e os métodos continuavam a ser legíveis.

****

Este code smell foi revisto por Rafael Mira.

## Design Patterns

### Bruno David

* Decorator :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/calendar/walker, Classe WorkingUnit-Counter

A classe WorkingUnitCounter implementa os métodos inerentes aos seus objetos, no entanto esses mesmos objetos podem comportar-se de modo a que é inerente a outros (ForwardTimeWalker)

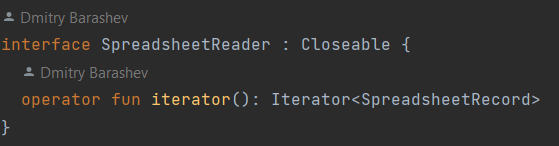


Este pattern foi revisto por Gonçalo Cerveira

* Iterator :

Localização: ganttproject.impex.csv, classe WorkingUnitCounter.java

Nesta interface é utilizado um iterador, aparentemente para iterar o registo de uma linha na spreadsheet.



Este pattern foi revisto por Gonçalo Cerveira

* Abstract Factory :

Localização: ganttproject.task.dependency, classe WorkingUnitCounter

Ambas as classes estendem uma classe abstrata, o que mostra claramente a existência de dois objetos que têm semelhanças e, por isso, partilham métodos, todavia não representam a mesma instância.





Este pattern foi revisto por Mafalda Batalha.

### Rafael Mira

* Chain of Responsibility :

Localização:ganttproject.core/src/main/java/biz/ganttproject/core/chart/scene/gantt/DependencyScene-Builder.java

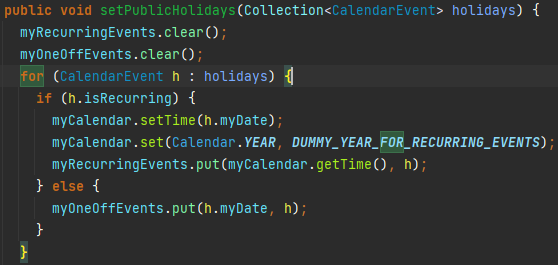
No método createConnector() da classe, podemos ver uma cadeia de if que são responsáveis por devolver um valor para a variável xDantEntry.

Este pattern foi revisto por Bruno David.

* Iterator :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/calendar/WeekendCalendarImpl.java

No método setPublicHolidays() da classe, está-se a iterar sobre os elementos da coleção holidays utilizando um for each.

******

Este pattern foi revisto por Mafalda Batalha.

* Template Method :

Localização: ganttproject.core/src/main/java/biz/ganttproject/core/calendar, classes GPCalendarBase, AlwaysWorkingTimeCalendarImpl, WeekendCalendarImpl

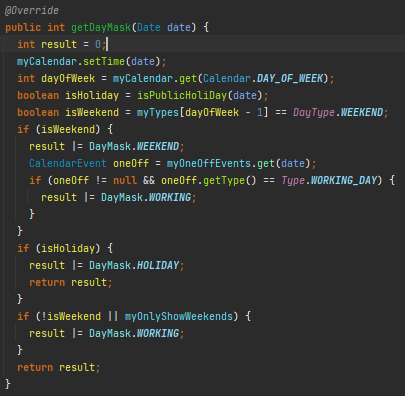
Na classe GPCalendarBase, o método getDayMask é implementado de formas diferentes pelas suas sub-classes.







Este pattern foi revisto por Gonçalo Cerveira.



### Vasco Malta

* Decorator :

Localização: biz.ganttproject.core.option

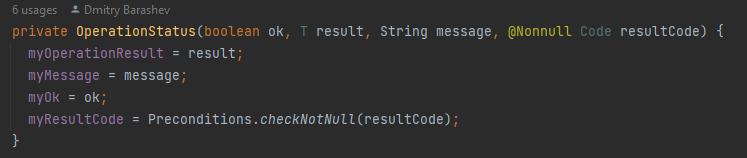
Escolhi a Decorator Design Pattern pois todas as classes implementam a mesma abstract class que esconde a complexidade do programa.



Este pattern foi revisto por Gonçalo Cerveira.

* Singleton :

Localização: biz.ganttproject.core

A classe OperationStatus implementa o Singleton Pattern uma vez que pode apenas existir uma instanciação da mesma, tendo em conta que o seu construtor é privado.

Este pattern foi revisto por Mafalda Batalha.

* Facade :

Localização: biz.ganttproject.core.chart.text

Escolhi a facade pattern pois as 5 seguintes classes implementam diretamente na mesma interface tendo métodos com assinatura iguais mas com implementações diferentes.



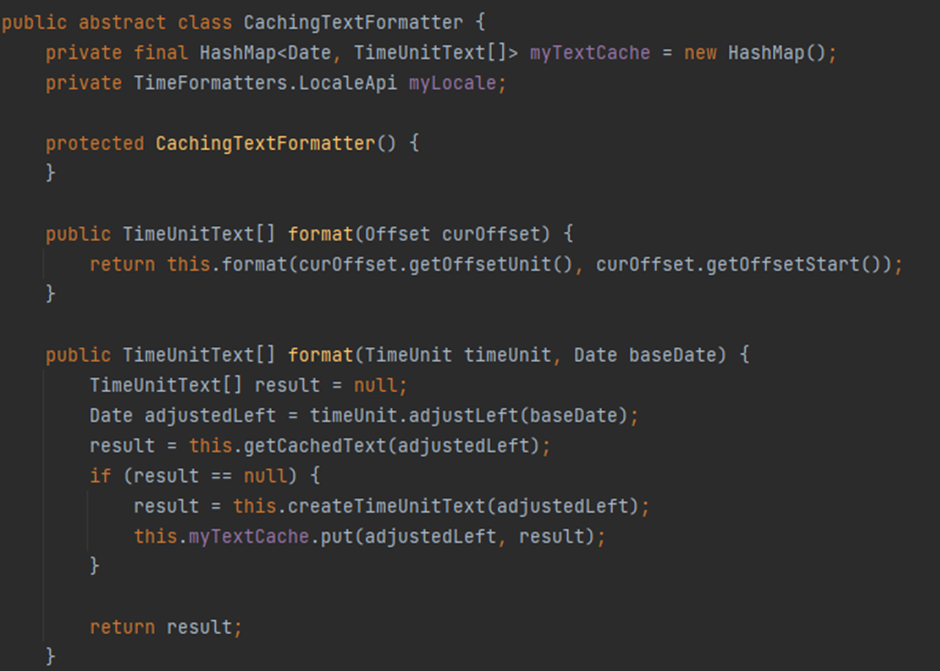
Este pattern foi revisto por Bruno David.

### Mafalda Batalha

* Template Method :

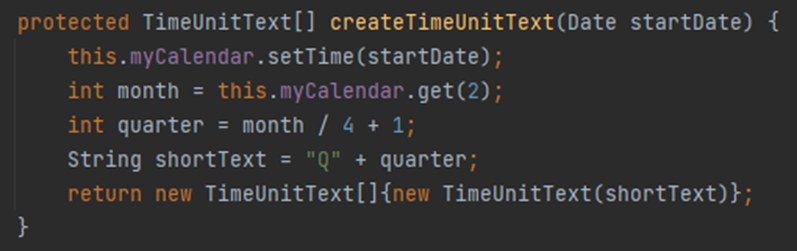
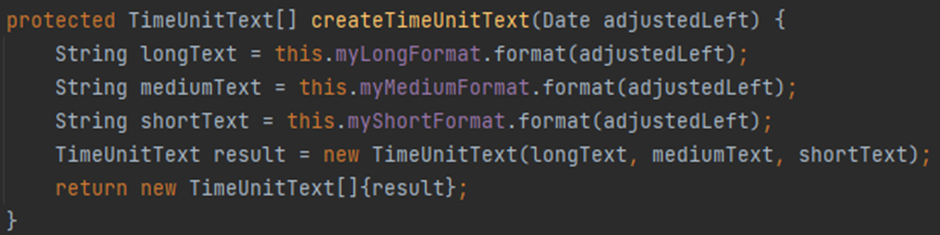
Localização: biz.ganttproject.core.chart.text (biz.ganttproject.core.src.main.java)

A classe abstrata CatchingTextFormatter contém um método inicializado CreateTimeUnixText(), utilizado no método implementado format(), que será implementado de forma diferente por cada classe que a estende.



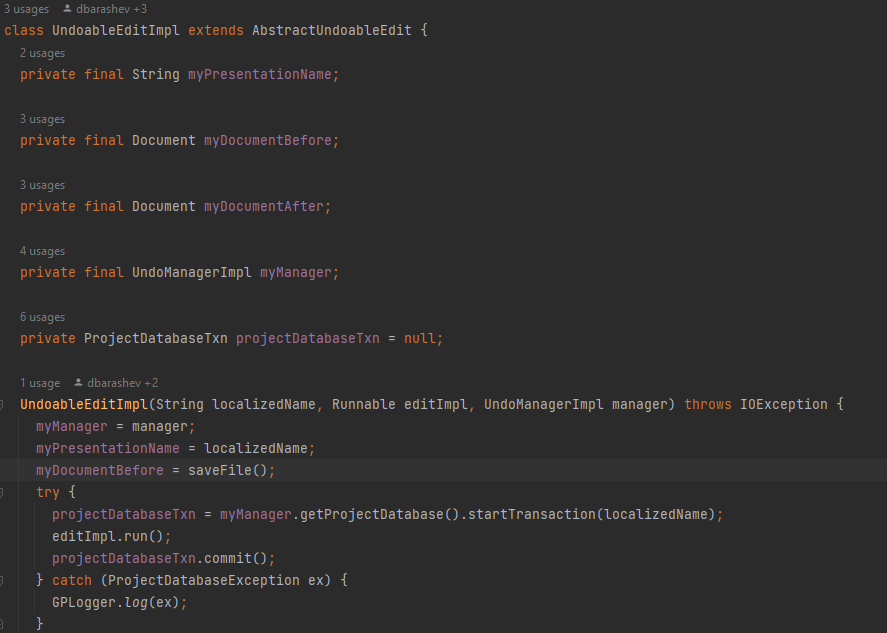
As classes WeekTextFormatter, QuarterTextFormatter, MonthTextFormatter e YearTextFormatter são as classes mencionadas anteriormente que estendem a classe abstrata.

**

** Este pattern foi revisto por Bruno David e Vasco Malta.

* Memento :

Localização: net.sourceforge.ganttproject.undo (ganttproject.src.main.java)

A classe UndoableEditImpl implementa o Memento Pattern uma vez que guarda duas versões de um objeto Document (myDocumentBefore e myDocumentAfter), permitindo, assim, executar operação de “redo” e “undo”.

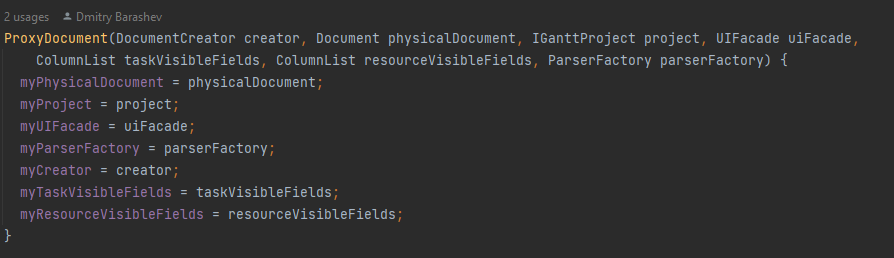
**

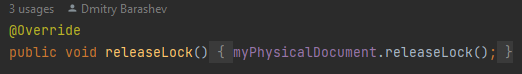


Este pattern foi revisto por Rafael Mira.

* Proxy :

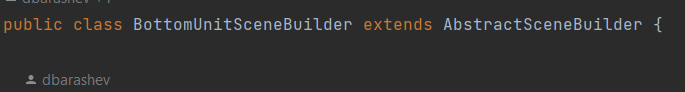
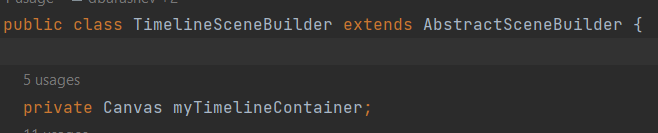
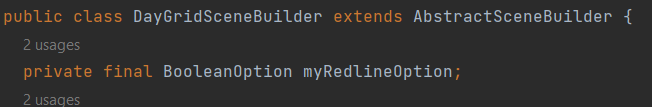
Localização: net.sourceforge.ganttproject.document (ganttproject.src.main.java)

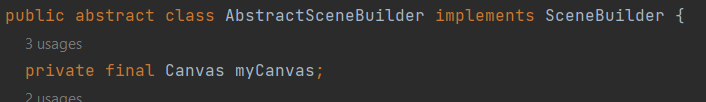
Tal como o próprio nome indica, a classe ProxyDocument demonstra o design pattern Proxy. Esta implementa a mesma interface (Document) que a classe do objeto real (myPhysicalDocument) que contém e sobre a qual delega algum dos pedidos.

**Este pattern foi revisto por Gonçalo Cerveira.

### Gonçalo Cerveira

* Abstract Factory :

****Localização: ganttproject.core/src/main/java/biz/ganttproject/core/chart/scene/AbstractSceneBuilder.java

As classes *DayGrindSceneBuilder*, *TimeLineSceneBuilder* e *BottomUnitSceneBuilder* funcionam através de uma superclasse, esta sendo a *AbstractSceneBuilder,* que assim arranja uma boa forma de criar objetos sem ter especificar as suas classes.

Este pattern foi revisto por Mafalda Batalha.

* State Pattern :

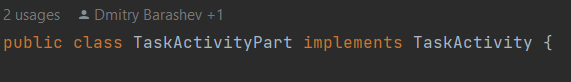
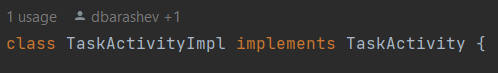
Localização: biz.ganttproject.core/src/main/java/biz/ganttproject/core/OperationStatus.java

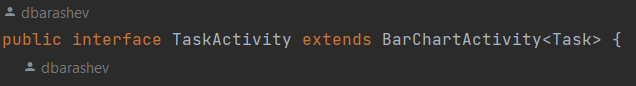
A classe ***OperationStatus*** trata-se de um padrão que muda o comportamento de acordo com o estado.

Este pattern foi revisto por Rafael Mira.

****

* Facade :

**Localização: ganttproject/src/main/java/net/sourceforge/ganttproject/task/TaskActivity.java

As classes *TaskActivityPart* e *TaskActivityImpl* implementam diretamente na interface *TaskActivty*, ajudando o código estruturalmente, fazendo com que desse para esconder a complexidade da interface.

Este pattern foi revisto por Rafael Mira.

# 2º Fase

## Métricas

### Bruno David - Mood Metrics

(As MOOD Metrics foram revistas por Rafael Mira)

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)

### Gonçalo Cerveira - Martin Packaging Metrics

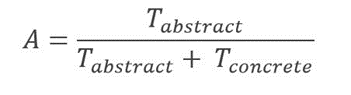
(As Martin Packaging Metrics foram revistas por Bruno David)

A métrica Martin Packaging tem como objetivo identificar packages mal organizadas/planificadas e assim melhorar o desenvolvimento de software.

A métrica Martin Packaging tem 5 parâmetros de avaliação:

* A - Abstractness :

Esta métrica é utilizada para medir o grau de abstração dos packages. Indo pela definição, a abstração é o número de classes abstratas no package para o número de todas as classes.

Tabstract – Nº de classes abstratas num package;

Tconcrete – Nº de classes concretas num package.

Os valores considerados normais para esta métrica deveriam ser valores entre 0 e 1. Packages que são considerados estáveis (outra métrica que irá ser abordada mais tarde no relatório), ou seja, que têm uma dependência pequena noutros packages, também deveriam ser abstratas (ter um valor de A próximo de 1). Noutro caso, os packages mais instáveis deveriam consistir em classes concretas (ter um valor de A próximo de 0).

* Ca – Afferent Coupling :

Esta métrica é utilizada para medir tipos de dependências entre os packages. Permite-nos medir a sensibilidade dos outros packages, às mudanças do package observado.

Os valores considerados normais desta métrica são valores entre o 0 e o 500. Valores altos nesta métrica sugerem uma alta estabilidade do componente estudado. Isto porque a classe depende de muitas outras classes e, por isso, não pode ser modificada significativamente, pois a probabilidade de espalhar as mudanças é muito grande.

* Ce - Efferent Coupling :

Esta métrica é utilizada para medir relações entre classes. É o número de classes num package dado, que dependem de outras classes num outro package. Permite-nos medir a vulnerabilidade dos packages, em relação a mudanças, em packages que esse depende.

Os valores considerados normais nesta métrica são valores entre 0 e 20, valores mais altos que isto criam problemas de desenvolvimento de código. Um valor alto neste tipo de métrica (Ce > 20) indica instabilidade do package, qualquer tipo de mudança de classes externas pode causar a necessidade de fazer alterações neste package.

* D – Normalized Distance from Main Sequence :

Esta métrica é utilizada para medir o balanço entre a estabilidade (métrica que irá ser abordada mais tarde no relatório) e abstração. É calculada da seguinte maneira:

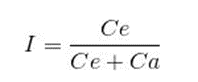


A – Abstração I – Instabilidade

O valor desta métrica devia ser o valor mais baixo possível, para que os componentes estivessem perto da sequência principal.

* I – Instability:

Esta métrica é utilizada para medir a suscetibilidade das classes mudarem. De acordo com a definição, instabilidade é a proporção das **Ce** para todas as dependências de packages.

A instabilidade é calculada através de:

Ce - Efferent Coupling

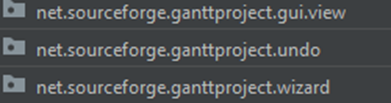
Ca - Afferent Coupling

Os valores considerados normais para esta métrica deveriam ser valores entre 0 e 1. E dentro desses valores deveríamos ter packages instáveis ou estáveis, ou seja, devíamos evitar packages com uma estabilidade intermédia.

* Verificação dos valores obtidos ao calcular as métricas no trabalho:

Começando por verificar a *Abstractness (A)*:

- Através dos dados recolhidos, conseguimos verificar que a maior parte dos packages têm um valor baixo de abstração, significando que a maior parte deles têm um nível baixo de dependência de outros packages. Alguns dos packages preocupantes são aqueles que se encontram com valor aproximado do meio (aprox. 0.5), pois dessa forma estes packages terão mudanças que poderão ser difíceis.

- Alguns dos packages problemáticos serão packages como:

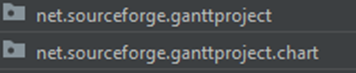
Passando agora para *Afferent Coupling (Ca):*

- Ao verificar os valores, verifica-se que a maioria dos packages têm um valor baixo de Ca, o que é positivo, pois significa que a maior parte dos packages não tem dependência de outros, pelo que ao mudá-los trará uma alteração significativa dentro do código do package.

-Uns dos packages mais problemáticos serão o net.sourceforce.ganttproject.task e net.sourceforge.ganttproject, o que ao analisar os packages faz sentido, sendo estes os packages que se vai desenvolver mais software e que se vai tratar da estrutura de dados da aplicação, ao desenvolver código nestes packages, que são dependentes de todos os outros, simplesmente irão se notar as diferenças nesses mesmos packages, uma vez que os outros já foram contruídos/implementados com intenção de não alterar muito ao longo do desenvolvimento de novo software na aplicação.

*Efferent Coupling (Ce):*

- Verificando os valores recolhidos observamos que a média de valores encontra-se muito elevada comparado com o que é proposto (**Ce** <= 20), pelo que isto significa que a aplicação poderá ter muitos problemas de desenvolvimento de software no futuro. Este tipo de valores indica que o projeto encontra-se com valor elevado de instabilidade de packages e qualquer tipo de mudança de classes, externas aos packages referentes, poderá implicar um elevado número de alterações a esses mesmos packages externos.

- Alguns dos packages problemáticos serão packages como:

Estes packages encontram-se com os valores de **Ce** mais elevados, pelo que com qualquer tipo de alteração externa, teria uma chance muito elevada de ser preciso alterar código nestes packages, mas ao analisar os mesmo verificamos que estes packages fazem parte do pequeno número que vai ser preciso alterar código, pelo que, neste projeto, os problemas de acordo com esta métrica não vão ser muito notados.

*Normalized Distance from Main Sequence (D):*

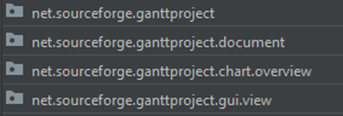
-Com os valores obtidos observamos que metade dos packages do projeto têm um valor extremamente baixo, ou seja, têm um bom balanço entre instabilidade e abstração.

- Alguns dos packages problemáticos serão packages como:

Ao analisar estes packages mais profundamente e com auxílio de I(Instability) e A(Abstractness) verificamos que temos uma situação que não é muito agradável, ambos os packages têm o valor de I e de A de 0, pelo que significa que os packages são extremamente estáveis e concretos, ou seja, nada flexíveis e não podem ser estendidos.

*Instability (I):*

- Através dos dados recolhidos, conseguimos verificar que a maior parte dos packages têm um valor elevado de Instabilidade, o que faz sentido, comparado com os resultados obtidos em A(Abstractness), significando assim que os packages são considerados instáveis, pois serão facilmente feitas mudanças no código que implicam alterar esses packages. Também verificamos que das restantes, a maior parte tem um valor baixo de Instabilidade, pelo que significa que esses packages serão mais difíceis de serem alterados devido a sua importância (estabilidade).

-Alguns dos packages problemáticos serão packages como:

Estes packages encontram-se com valores próximos de 0.5. Estes packages são considerados packages de estabilidade intermédia, pelo não fornecem uma boa solução para desenvolvimento de software.

### Mafalda Batalha - Complexity Metrics

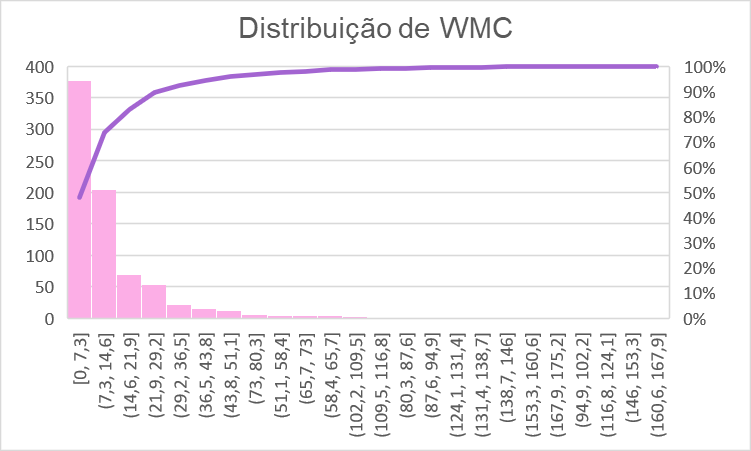
(As Complexity Metrics foram revistas por Vasco Malta)

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 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):

Esta métrica refere-se à soma total de Cyclomatic Complexity (CC) dos métodos de uma classe e, portanto, quanto maior for este valor, maior é a complexidade da classe. CC refere-se à quantidade de caminhos linearmente independentes, o que equivale ao número de decisões (if/else, switch/case, for, ect.) feitas num método +1.

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, toma várias decisões, nomeadamente do tipo for, if/elseIf/else e switch/case.

* OCavg (Average Operation Complexity):

Esta métrica refere-se à média da CC dos métodos não abstratos ou herdados de uma classe.

No nosso projeto, os valores de OCavg 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. (página seguinte)



Se observarmos, por exemplo, a classe relativa ao valor máximo de OCavg que, neste caso, é net.sourceforge.ganttproject.GanttOptions.GanttXML-OptionsParser, 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).

* OCmax (Average Operation Complexity):

Esta métrica refere-se ao valor máximo da CC dos métodos não abstratos ou herdados de uma classe.

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://en.wikipedia.org/wiki/Cyclomatic_complexity>

<https://refactoring.guru/refactoring/smells>

### Rafael Mira - Chidamber-Kemerer Metrics

(As Chimdamber-Kemerer Metrics foram revistas por Gonçalo Cerveira)

A métrica Chidamber-Kemerer tem como objetivo medir a complexidade do design e melhorar o desenvolvimento de software. Isto faz-se através da identificação de valores fora do aceitável para os parâmetros de avaliação da métrica.

A métrica Chidamber-Kemerer consiste em seis parâmetros de avaliação:

* DIT – Depth of Inheritance Tree:

Representa o comprimento máximo desde a classe até à classe raiz.

Quanto maiores são os valores de DIT, significa que existe um número maior de métodos que é possível herdar dos seus antecessores tornando mais complexo prever o seu comportamento. Por outro lado, existe um maior potencial para reutilizar os métodos herdados.

Com valores mais pequenos de DIT, significa que ao desenvolver o software não demos tanta importância à reutilização em prol da facilidade de compreensão.

Portanto o objetivo deste parâmetro é um trade-off no sentido em que, ao escolher dar maior importância à reutilização, torna-se o código mais complexo. Se se prefere facilitar a compreensão, limita-se a reutilização.

* NOC – Number of Children:

Na hierarquia de classes, representa o número de subclasses que são diretamente filhas de uma classe.

Quanto maior o valor de NOC, maior é a reutilização e maior é a probabilidade de existir abstração inapropriada da classe pai.

Valores baixos de NOC podem ser um indicador de falta de comunicação entre os desenvolvedores do projeto.

Portanto, o objetivo deste parâmetro é também um trade-off no sentido em que, se com valores baixos, provavelmente não se trabalhou bem em equipa. Por outro lado, valores elevados podem significar abstração inapropriada..

* CBO – Coupling Between Objects:

Representa o número de outras classes às quais está acoplado.

Valores baixos de CBO promovem a encapsulação, indicam independência da classe tornando-a mais fácil de reutilizar. Torna também mais fácil fazer a sua manutenção e criação de testes.

Portanto o objetivo é manter este valor baixo.

* RFC – Response for Class:

Representa a soma entre o número de métodos de uma classe e o número de métodos chamados pelos métodos dessa classe.

Se o valor de RFC for elevado, então o número de métodos invocados também é elevado e, isto torna a manutenção e o teste da classe mais complexo.

Portanto o objetivo é manter este valor baixo.

* WMC – Weighted Method per Class:

Representa o número de métodos existentes numa classe.

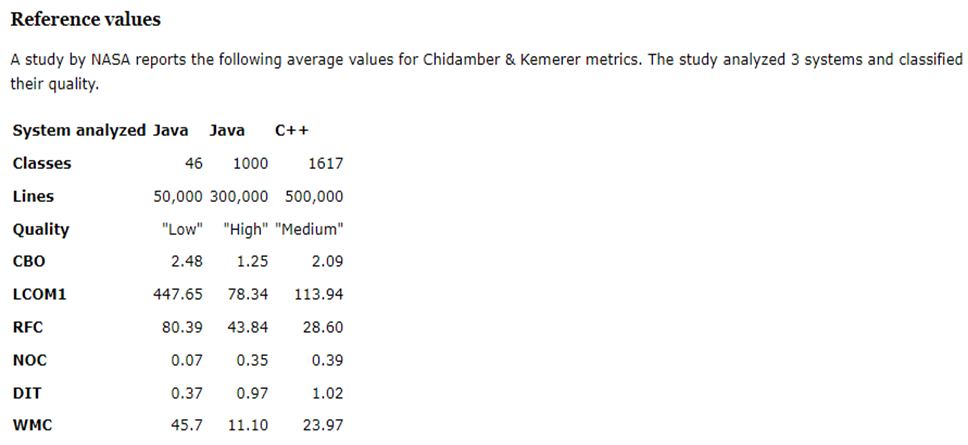
Estima o tempo e o esforço necessário para desenvolver e manter a classe. Obviamente, quanto maiores os valores de WMC, maior o esforço e o tempo necessário para desenvolver e manter a classe. Portanto o objetivo ao desenvolver software é manter este valor baixo.

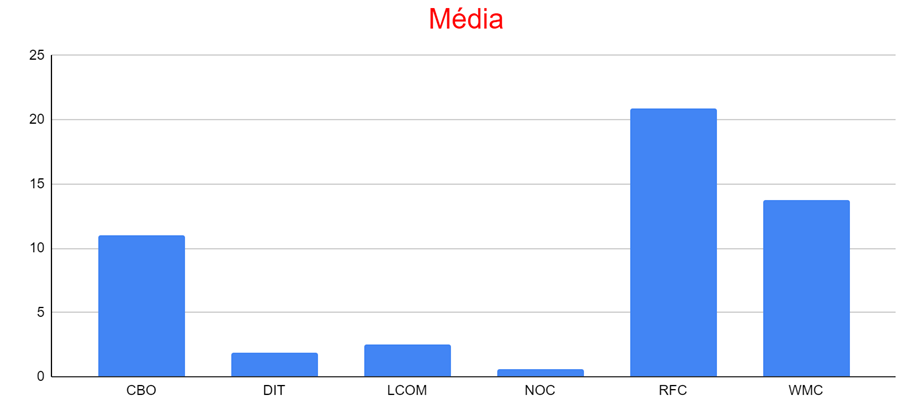
* LCOM – Lack of Cohesion of Method:

Representa a diferença entre o número de métodos que não têm variáveis de instância em comum e o número de métodos que têm variáveis de instância em comum.

Quanto maior for o valor de LCOM, maior é a complexidade da classe. Valores elevados indicam também que não se promove a encapsulação. O objetivo é manter um valor baixo.

A tabela abaixo contém valores de referência para podemos classificar a qualidade do software e portanto, iremos ter em conta estes valores para classificar os resultados da métrica no nosso projeto. Apenas iremos considerar os resultados para Java pois é a linguagem usada na maioria do nosso projeto.

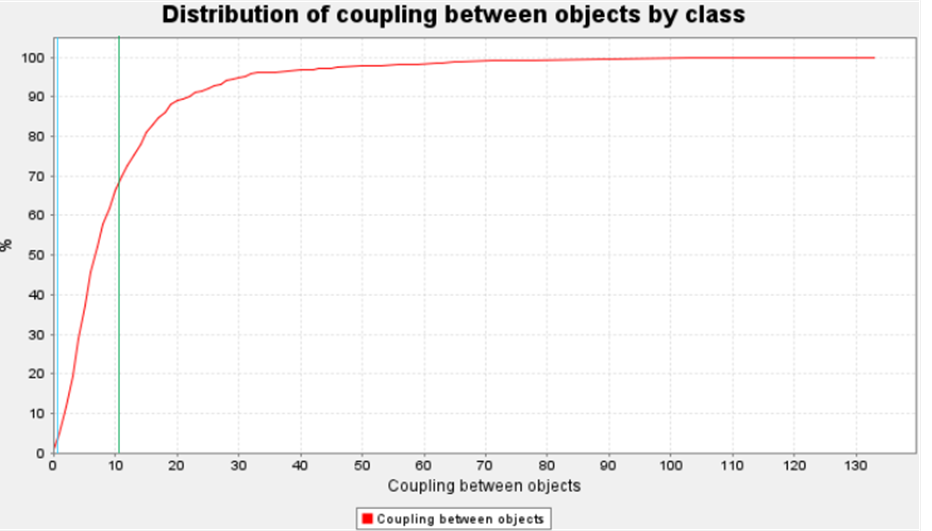
<https://www.aivosto.com/project/help/pm-oo-ck.html>



*Gráfico com os valores médios para a métrica Chidamber-Kemerer do ganttproject.*

*Nota*: Todos os gráficos que se seguem têm o valor médio representado por uma linha verde e o valor de referência de alto nível (valor da tabela) representado por uma linha azul.

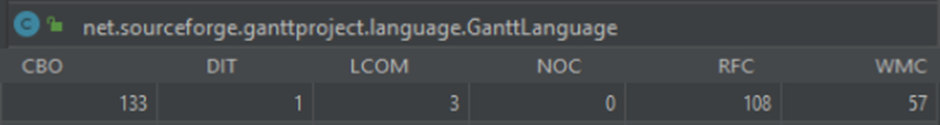
*Análise CBO*



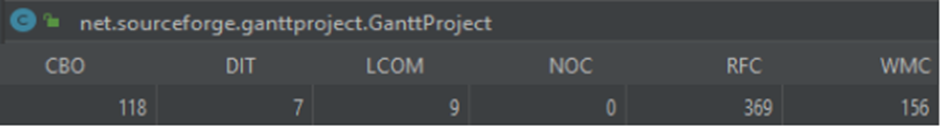
*Gráfico com a distribuição de CBO entre as classes.*

Começando pelo CBO, é o valor que se destaca de imediato dos outros valores. Isto é, comparando a proporção com os valores de referência e, considerando os casos em que foi usada Java como linguagem, o valor de CBO é bastante mais baixo do que o nosso valor (cerca de 5 vezes no caso de baixa qualidade e 10 vezes no caso de alta qualidade).

Isto é um mau sinal pois nós queremos tentar baixar o coupling, ou seja, baixar o valor de CBO. Analisando a classe com o valor de CBO mais elevado (133). (Classe A)



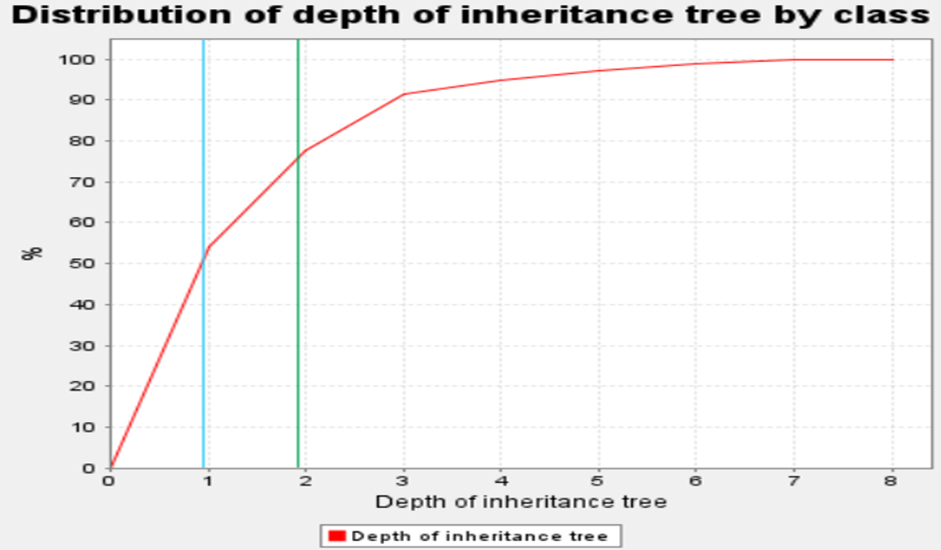
Esta classe realmente tem o valor mais elevado de CBO mas, se formos analisar a classe em si, verificamos que este valor é tão elevado pois utiliza bastantes classes da própria linguagem Java. Isto não é tão problemático como utilizar classes criadas pelo developer porque, a probabilidade de uma classe da linguagem Java ser alterada e isso fazer com que a classe que a utiliza deixe de funcionar corretamente é menor do que, utilizar várias classes criadas no âmbito de um projeto e, por necessidade, ter que alterar alguma delas e causar efeitos laterais negativos para uma classe que a utiliza.

Analisando a classe com segundo valor de CBO mais elevado (118). (Classe B)

Observamos que esta classe utiliza muitas classes criadas no âmbito deste projeto. Mesmo com um valor mais baixo de CBO em relação à outra classe analisada, o coupling nesta classe pode ser mais preocupante.

Olhando para os restantes parâmetros de comparação, observamos que a classe B tem os valores superiores em todos eles excepto no NOC. Isto indica que, embora a classe A tenha um valor de CBO superior à classe B, a classe B aparenta ser mais complexa que a classe A.

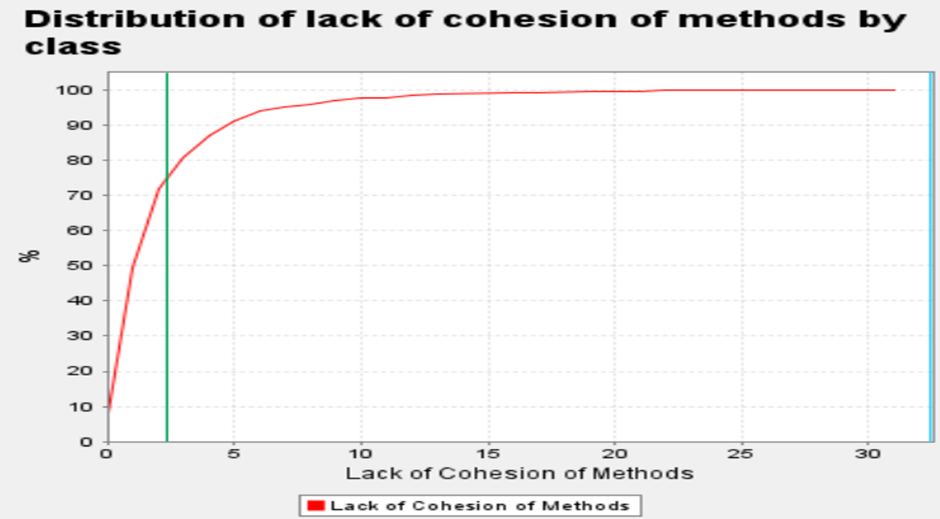
*Análise DIT*



*Gráfico com a distribuição de DIT entre as classes.*

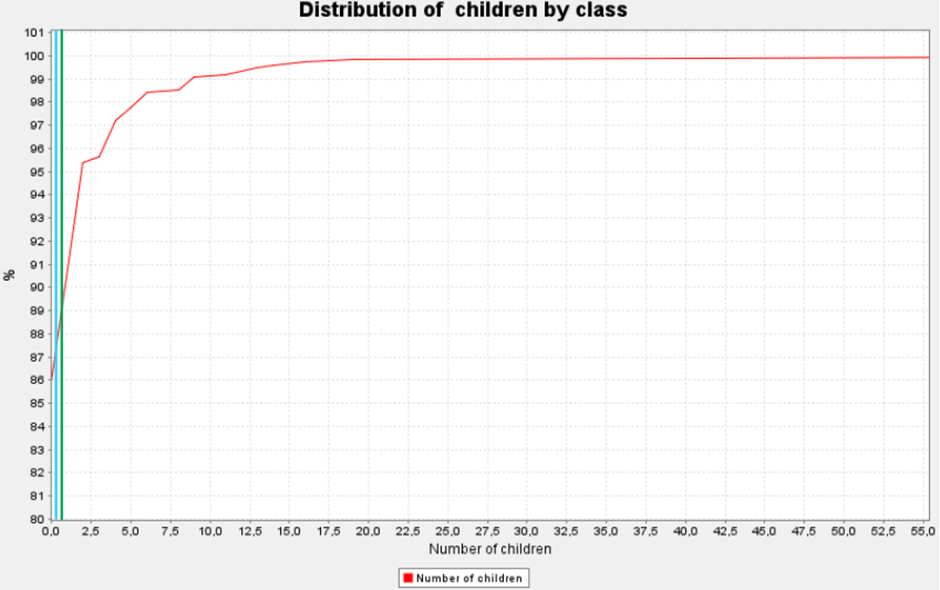
Podemos observar que em aproximadamente 75% das classes, o DIT está abaixo do valor médio. Em pouco mais de 50% o valor de DIT está abaixo de 1. Tendo em conta os valores de referência, podemos considerar que em 50% das classes o valor de DIT está de acordo com o pretendido. Mas como este indicador é um trade-off entre reutilização e facilidade de compreensão, podemos assumir que em 75% das classes (DIT < 2), foi escolhido dar prioridade à compreensão e nos restantes 25% foi preferida a reutilização.

*Análise LCOM*



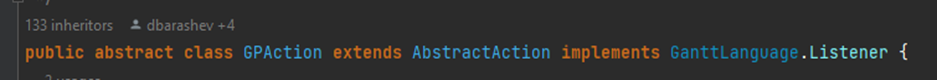
*Gráfico com a distribuição de LCOM entre as classes.*

O valor médio de LCOM é cerca de 2,5. Este valor é muito baixo quando comparado com o valor de referência que é 78,34. Outro bom indicador é que o valor máximo encontrado para LCOM é 31 que também está abaixo do valor de referência. Com estes valores podemos concluir que se promoveu a encapsulação e que se tentou aumentar a coesão das classes.

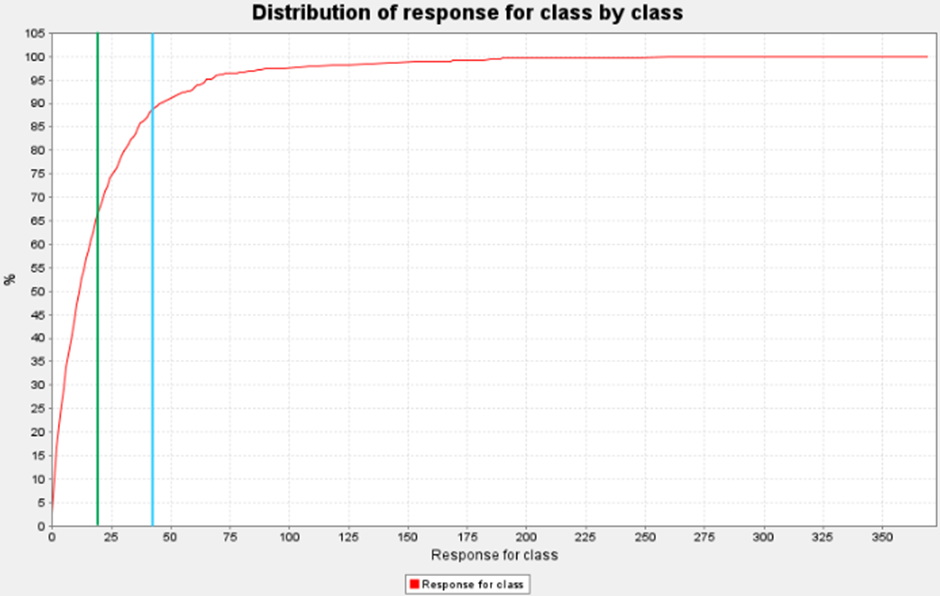
*Análise NOC*

*Gráfico com a distribuição de NOC entre as classes.*

Tendo em conta o valor de referência (0.35), o valor médio de NOC (0.56) é quase o dobro. Em 85% das classes o valor de NOC está abaixo do valor médio, enquanto que nos 15% restantes, o valor de NOC está muito acima do valor médio e portanto do valor de referência. Estes valores tão elevados podem ser resultado de abstração indevida.

Por exemplo, a classe com maior valor de NOC (86).

Fazendo uma análise sucinta, podemos observar que esta classe tem 133 herdeiros. Ou seja, para além dos 86 herdeiros diretos, tem ainda 47 herdeiros por consequência. Provavelmente existem classes que poderiam estar modeladas de forma diferente de modo a diminuir o valor de NOC.

*****Análise RFC*

*Gráfico com a distribuição de RFC entre as classes.*

Este parâmetro tem o valor médio igual a 20.85. Este valor é 2 vezes menor que o valor de referência do projeto de alto nível o que é bastante positivo uma vez que se quer tentar manter este valor baixo.

*Análise WMC*

Legenda *- Gráfico com a distribuição de WMC entre as classes.*

O valor médio de WMC é aproximadamente 14. Este valor está muito próximo do valor de referência do projeto de alto nível que é 11,1, o que é algo positivo.

No entanto, podemos observar que em 5% das classes, o número de métodos é superior a 50 e em alguns casos o valor de WMC é elevadíssimo. Isto é um indicador de que se poderiam reestruturar estas classes em várias classes mais pequenas.

### Vasco Malta - Lines of Code Metrics

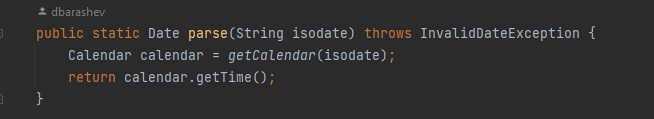
(As Lines of Code Metrics foram revistas por Mafalda Batalha)

Lines of Code Metrics calcula o número de linhas de código presentes num projeto, classes e ficheiros.Existem algumas sugestões relativamente á utilização desta métrica:

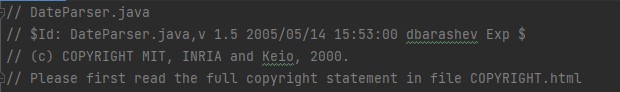
* O tamanho das funções deve estar entre as 4 e 40 linhas de programação. Uma função tem por definição 4 linhas (1 linha cabeçalho, 1 linha de código, e 2 linhas para o par de chavetas). Se uma função contém mais de 40 linhas provavelmente implementa mais do que 1 função.
* O tamanho de um ficheiro deve estar entre os limites de 4 e 400 linhas de programação. Um ficheiro deve ter no mínimo 1 função daí o valor mínimo de um ficheiro ser de 4 linhas pois este é também o valor mínimo de tamanho para uma função. Obviamente quanto maior forem os ficheiros, mais complicado será os compreender.
* CLOC (comment lines of code) devem ocupar entre 30% e 75% de um ficheiro caso contrário pode-se presumir que o ficheiro forneça pouca informação sobre o mesmo ou inversamente tenha um excesso de comentários explicando em demasia as funções implementadas.

Lines of Code Metrics baseia-se em 3 fatores.

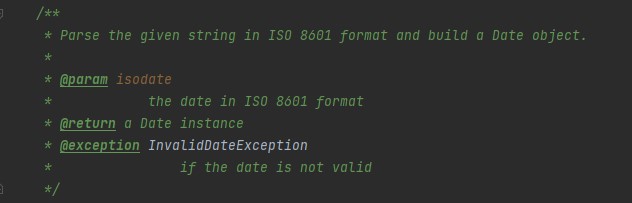
LOC (Lines of code) - linhas de código



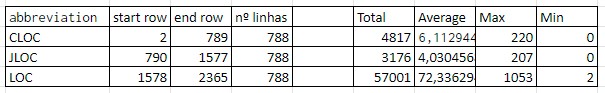
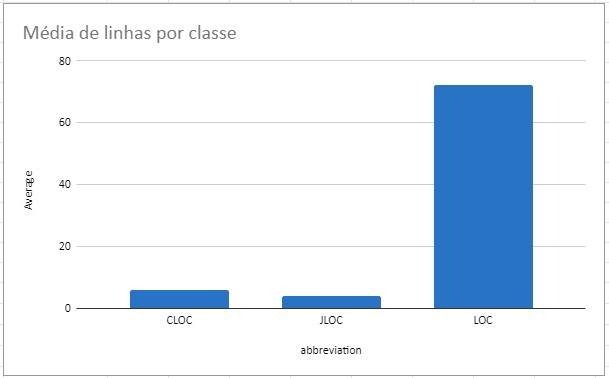
CLOC (Comment lines of code) - linhas de comentários presentes no projeto geralmente presentes entre //



JLOC (Javadoc lines of code) - linhas de cabeçalho que explica os parâmetros das funções, geralmente entre /\* \*/



Apresenta-se de seguida os valores registados na análise de todo o projeto.



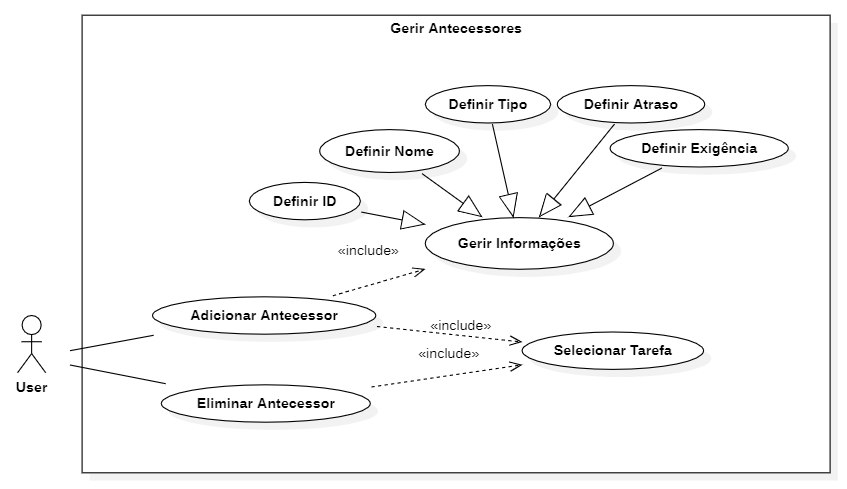
Através da análise tanto do gráfico como da tabela, é possível verificar a não verificação de pelo menos 2 das sugestões acima referidas. Este projeto apresenta poucos comentários comparativamente ao número de LOC que apresenta sendo este abaixo dos 30% e existem classes com tamanhos excessivos como é possível ver através da tabela onde o maior ficheiro do projeto apresenta 1053 linhas apenas de LOC quando o tamanho recomendado acima era de no máximo 400 linhas .

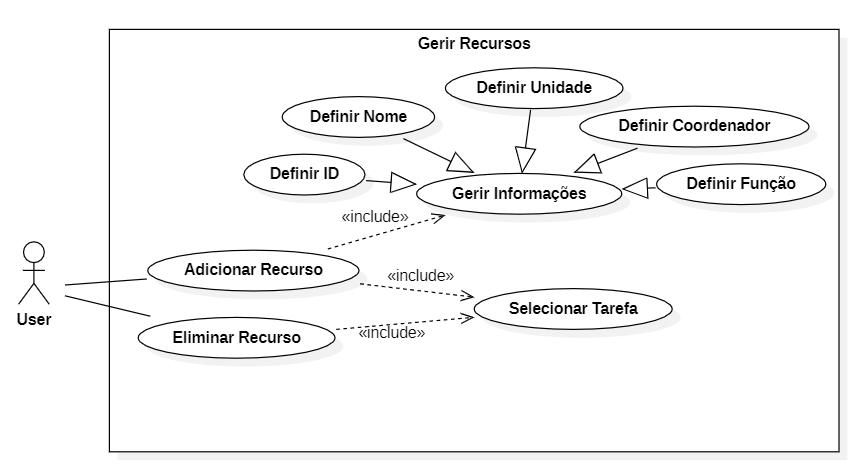
REFERÊNCIAS

<https://www.verifysoft.com/en_linesofcode_metrics.html>

## Use Cases

### Bruno David – Tarefas



****

Nome: Adicionar Tarefa

ID: 1

Descrição: o Task Manager (User) adiciona uma nova tarefa à tabela de tarefas que já se encontram registadas.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome: Eliminar Tarefa

ID: 2

Descrição: o Task Manager (User) seleciona uma tarefa e elimina-a, das que já se encontram registadas na tabela de tarefas.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Propriedades da Tarefa

ID: 3

Descrição: o Task Manager (User) seleciona uma nova tarefa, da tabela de tarefas que já se encontram registadas, e consegue aceder/modificar todas as informações relativas a esta.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Renomear a Tarefa

ID: 4

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e renomeia a tarefa.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Definir Data Início da Tarefa

ID: 5

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e seleciona uma nova data de início.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Definir Duração da Tarefa

ID: 6

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e seleciona uma nova duração.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Definir Prioridade da Tarefa

ID: 7

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e seleciona a prioridade.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Definir Progresso da Tarefa

ID: 8

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e seleciona uma nova taxa de progresso.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Definir Atalho Web da Tarefa

ID: 9

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e adiciona um atalho Web à tarefa.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome: Definir Antecessor da Tarefa

ID: 10

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e adiciona um antecessor à tabela de antecessores dessa tarefa..

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Gerir Antecessor da Tarefa

ID: 11

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e acede/modifica um antecessor da tabela de antecessores dessa tarefa..

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Definir Recurso da Tarefa

ID: 12

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e adiciona um recurso à tabela de recursos dessa tarefa.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

Nome:Gerir Recursos da Tarefa

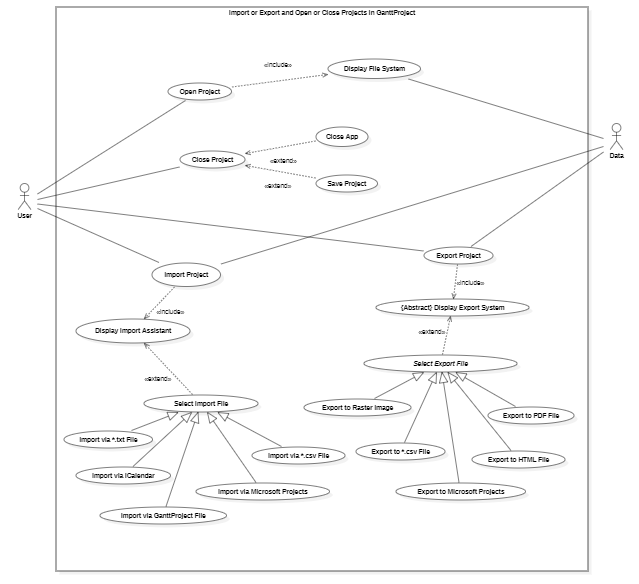
ID: 13

Descrição: o Task Manager (User) seleciona uma tarefa, da tabela de tarefas que já se encontram registadas, acede às suas propriedades e acede/modifica um recuso da tabela de recursos dessa tarefa.

Ator Principal: Task Manager (User)

Atores Secundários: Nenhum

### Gonçalo Cerveira – Projetos



Nome: Open Project

ID: 43

Descrição: Abre um projeto no GanttProject, guardado numa diretoria a escolha do utilizador.

Ator Principal: User

Atores Secundários: Data

Nome: Display File System

ID: 44

Descrição: Aparece um ecrã com as diretorias do sistema para poder escolher o projeto

Ator Principal: User

Atores Secundários: Data

Nome: Close Project

ID: 45

Descrição: Guarda um projeto, com possíveis alterações e consequentemente fecha a aplicação.

Ator Principal: User

Atores Secundários: Nenhum

Nome: Close App

ID: 46

Descrição: Fecha a aplicação

Ator Principal: User

Atores Secundários: Nenhum

Nome: Save Project

ID: 47

Descrição: Em caso de alterações no projeto, aparece a hipótese para salvar o projeto

Ator Principal: User

Atores Secundários: Nenhum

Nome: Import Project

ID: 48

Descrição: Importa o projeto, de várias maneiras, para a aplicação

Ator Principal: User

Atores Secundários: Data

Nome: Display Import Assistant

ID: 49

Descrição: Aparece um ecrã com as opções possíveis para importar o projeto

Ator Principal: User

Atores Secundários: Nenhum

Nome: *Select Import File*

ID: 50

Descrição: A escolha do tipo de ficheiro a importar

Ator Principal: User

Atores Secundários: Nenhum

Nome: Import via \*.txt File

ID: 51

Descrição: Ficheiro importado através .txt

Especializa: *Select Import File*

Ator Principal: User

Atores Secundários: Nenhum

Nome: Import via iClanedar

ID: 52

Descrição: Ficheiro importado através de iCalendar

Especializa: *Select Import File*

Ator Principal: User

Atores Secundários: Nenhum

Nome: Import via GanttProject File

ID: 53

Descrição: Ficheiro importado através de GanttProject

Especializa: *Select Import File*

Ator Principal: User

Atores Secundários: Nenhum

Nome: Import via \*.csv File

ID: 54

Descrição: Ficheiro importado através .csv

Especializa: *Select Import File*

Ator Principal: User

Atores Secundários: Nenhum

Nome: Import via Microsoft Projects

ID: 55

Descrição: Ficheiro importado através Microsotf Projects

Especializa: *Select Import File*

Ator Principal: User

Atores Secundários: Nenhum

Nome: Export Project

ID: 56

Descrição: Exporta o Projeto para fora da aplicação para um tipo de ficheiro que o user pretende.

Ator Principal: User

Atores Secundários: Data

Nome: Display Export Assistant

ID: 57

Descrição: Aparece um ecrã com as opções possíveis para exportar o projeto

Ator Principal: User

Atores Secundários: Nenhum

Nome: *Select Export File*

ID: 58

Descrição: A escolha do tipo de ficheiro a exportar

Ator Principal: User

Atores Secundários: Nenhum

Nome: Export to \*.csv File

ID: 59

Descrição: Ficheiro exportado para .csv

Especializa: *Select Export File*

Ator Principal: User

Atores Secundários: Nenhum

Nome: Export to Raster Image

ID: 60

Descrição: Ficheiro exportado para imagem raster

Especializa: *Select Export File*

Ator Principal: User

Atores Secundários: Nenhum

Nome: Export to Microsoft Projects

ID: 61

Descrição: Ficheiro exportado para Microsoft Projects

Especializa: *Select Export File*

Ator Principal: User

Atores Secundários: Nenhum

Nome: Export to HTML File

ID: 62

Descrição: Ficheiro exportado para HTML

Especializa: *Select Export File*

Ator Principal: User

Atores Secundários: Nenhum

Nome: Export to PDF File

ID: 63

Descrição: Ficheiro exportado para um ficheiro PDF

Especializa: *Select Export File*

Ator Principal: User

Atores Secundários: Nenhum

### Mafalda Batalha – Definições Gráficas



Os use cases do diagrama que fiz estão relacionados às definições gráficas do projeto Gantt, nomeadamente as funcionalidades que permitem redimensionar os separadores Gantt e Resources Chart ou mostrar/ocultar o critical path.

Os use cases estão associados a apenas um ator que representa o utilizador da aplicação GanttProject. Assim, têm todos apenas um ator primário/principal (utilizador) e não têm atores secundários.

Nome: *Abrir separador*

ID: 64

Descrição: o utilizador abre um separador da aplicação

Nome: Abrir Gantt

ID: 65

Descrição: o utilizador abre o separador Gantt

Especializa: *Abrir separador*

Nome: Abrir Resources Chart

ID: 66

Descrição: o utilizador abre o separador Resources Chart

Especializa: *Abrir separador*

Nome: Abrir PERT chart

ID: 67

Descrição: o utilizador abre o separador PERT chart

Especializa: *Abrir separador*

Nome: *Ajustar zoom do painel*

ID: 68

Descrição: o utilizador ajusta o zoom que afeta o tamanho dos conteúdos do painel quando tem os separadores Gantt ou Resources Chart abertos

Nome: Zoom in

ID: 69

Descrição: o utilizador ajusta o zoom, aumentando o tamanho dos conteúdos do painel

Especializa: *Ajustar zoom do painel*

Nome: Zoom out

ID: 70

Descrição: o utilizador ajusta o zoom, diminuindo o tamanho dos conteúdos do painel

Especializa: *Ajustar zoom do painel*

Nome: *Ajustar separador*

ID: 71

Descrição: o utilizador ajusta a janela do separador aberto (Gantt ou Resources Chart)

Nome: Reduzir separador

ID: 72

Descrição: o utilizador reduz o tamanho do separador

Especializa: *Ajustar separador*

Nome: Aumentar separador

ID: 73

Descrição: o utilizador aumenta o tamanho do separador

Especializa: *Ajustar separador*

Nome: Ocultar separador

ID: 74

Descrição: o utilizador oculta o separador

Especializa: *Ajustar separador*

Nome: Mostrar separador

ID: 75

Descrição: o utilizador escolhe mostrar o separador

Especializa: *Ajustar separador*

Nome: Ocultar baselines

ID: 76

Descrição: o utilizador oculta as baselines no separador GanttNome: Adicionar baseline

ID: 77

Descrição: o utilizador adiciona uma baseline no separador Gantt

Nome: Remover baseline

ID: 78

Descrição: o utilizador remover uma baseline no separador Gantt

Nome: *Mudar indentação de tarefa*

ID: 79

Descrição: o utilizador muda a indentação das tarefas no separador Gantt

Nome: Indentar tarefa

ID: 80

Descrição: o utilizador indenta uma tarefa no separador Gantt

Especializa: *Mudar indentação de tarefa*

Nome: Desindentar tarefa

ID: 81

Descrição: o utilizador desindenta uma tarefa no separador Gantt

Especializa: *Mudar indentação de tarefa*

Nome: *Ajustar critical path*

ID: 82

Descrição: o utilizador ajusta o critical path no separador Gantt

Nome: Mostrar critical path

ID: 83

Descrição: o utilizador escolhe mostrar o critical path no separador Gantt

Especializa: *Ajustar critical path*

Nome:Ocultar critical path

ID: 84

Descrição: o utilizador oculta o critical path no separador Gantt

Especializa: *Ajustar critical path*

Nome*: Mudar ordem de objetos*

ID: 85

Descrição: o utilizador altera a ordem pela qual os objetos aparecem nos separadores Gantt e Resources Chart

Nome: Mover objetos para baixo

ID: 86

Descrição: o utilizador move um ou mais objetos para baixo (tarefas ou recursos) nos separadores Gantt e Resources Chart

Especializa: *Mudar ordem de objetos*

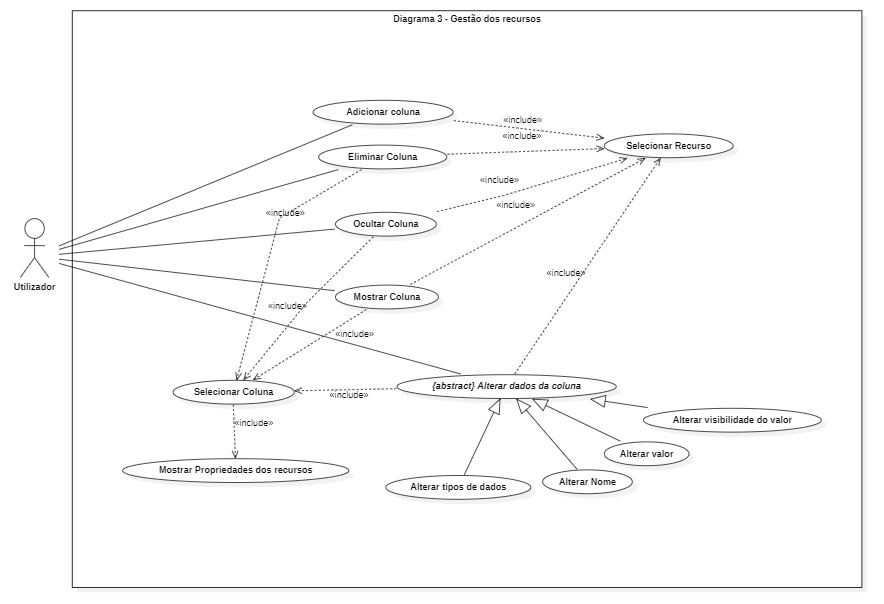
Nome: Mover objetos para cima

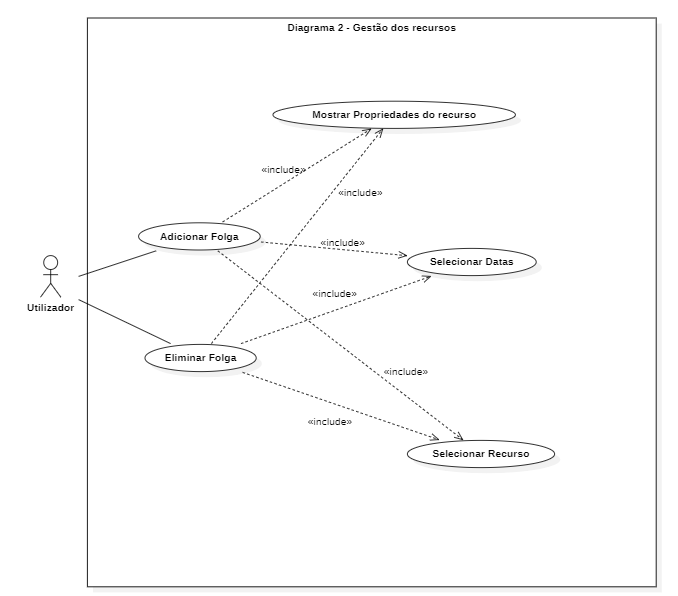
ID: 87

Descrição: o utilizador move um ou mais objetos para cima (tarefas ou recursos) nos separadores Gantt e Resources Chart

Especializa: *Mudar ordem de objetos*

### Rafael Mira – Recursos





Nome: Adicionar recurso

ID: 14

Descrição: O utilizador adiciona um novo recurso ao diagrama de recursos.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Alterar dados

ID: 15

Descrição: O utilizador altera dados do recurso.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Eliminar recurso

ID: 16

Descrição: O utilizador elimina um recurso do projeto.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Copiar recurso

ID: 17

Descrição: O utilizador copia um recurso do projeto.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Cortar recurso

ID: 18

Descrição: O utilizador corta um recurso do projeto.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Colar recurso

ID: 19

Descrição: O utilizador cola um recurso do projeto.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Enviar e-mail

ID: 20

Descrição: O utilizador envia e-mail para um recurso do projeto.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Selecionar Recurso

ID: 21

Descrição: O utilizador seleciona um recurso.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Consultar dados

ID: 22

Descrição: O utilizador consulta dados sobre o recurso.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Aceder às propriedades do Recurso

ID: 23

Descrição: O utilizador acede à janela de propriedades do recurso.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: *Definir dados*

ID: 24

Descrição: O utilizador define informação geral do recurso.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Definir Nome

ID: 25

Descrição: O utilizador define o nome do recurso.

Especializa: *Definir informação geral*

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Definir Telefone

ID: 26

Descrição: O utilizador define o telefone do recurso.

Especializa: *Definir informação geral*

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Definir E-mail

ID: 27

Descrição: O utilizador define o e-mail do recurso.

Especializa: *Definir informação geral*

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Definir Função

ID: 28

Descrição: O utilizador define a função do recurso.

Especializa: *Definir informação geral*

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Definir Taxa de pagamento

ID: 29

Descrição: O utilizador define a taxa de pagamento do recurso.

Especializa: *Definir informação geral*

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Adicionar folga

ID: 30

Descrição: O utilizador adiciona dias de folga ao recurso.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Eliminar folga

ID: 31

Descrição: O utilizador elimina dias de folga do recurso.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Selecionar Dias

ID: 32

Descrição: O utilizador seleciona os dias que quer adicionar ou eliminar do recurso.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Adicionar coluna

ID: 33

Descrição: O utilizador adiciona colunas ao diagrama de recursos.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Eliminar coluna

ID: 34

Descrição: O utilizador elimina colunas do diagrama de recursos.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Ocultar coluna

ID: 35

Descrição: O utilizador oculta colunas do diagrama de recursos.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Mostrar coluna

ID: 36

Descrição: O utilizador torna visíveis colunas do diagrama de recursos.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Selecionar coluna

ID: 37

Descrição: O utilizador selecionar uma coluna.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Alterar dados da coluna

ID: 38

Descrição: O utilizador altera dados da coluna.

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Alterar Nome

ID: 39

Descrição: O utilizador altera o nome da coluna.

Especializa: *Alterar dados da coluna*

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Alterar valor

ID: 40

Descrição: O utilizador altera o valor da coluna.

Especializa: *Alterar dados da coluna*

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Alterar tipo de dados

ID: 41

Descrição: O utilizador altera o tipo de dados da coluna.

Especializa: *Alterar dados da coluna*

Ator Principal: Utilizador

Atores Secundários: Nenhum

Nome: Alterar visibilidade do valor

ID: 42

Descrição: O utilizador altera a visibilidade do valor da coluna.

Especializa: *Alterar dados da coluna*

Ator Principal: Utilizador

Atores Secundários: Nenhum

### Vasco Malta – Funcionalidades Extra

### 

Todos os use cases seguintes têm como ator primário o utilizador, que representa uma pessoa que utiliza a aplicação GanttProject, e não têm atores secundários.

Nome**:** Tentar adicionar recurso a uma tarefa

ID**:** 88

Descrição**:** Utilizador tenta adicionar um recurso a uma dada tarefa.

Nome: *Receber Pop-up*

ID: 89

Descrição: Utilizador recebe um pop-up ao adicionar um recurso a uma dada tarefa.

Nome: Receber Pop-up Férias

ID: 90

Descrição: Utilizador recebe um pop-up ao adicionar um recurso que está de férias a uma dada tarefa.

Especializa: *Receber Pop-up*

Nome: Receber Pop-up Sobrecarga

ID: 91

Descrição: Utilizador recebe um pop-up ao adicionar um recurso a uma dada tarefa que causa sobrecarga.

Especializa: *Receber Pop-up*Nome: *Responder ao Pop-up*

ID: 92

Descrição: Utilizador responde ao pop-up recebido.

Nome: Cancelar

ID: 93

Descrição: Utilizador responde ao pop-up recebido cancelando-o.

Especializa: *Responder ao Pop-up*

Nome: Responder sim

ID: 94

Descrição: Utilizador responde ao pop-up recebido respondendo que sim.

Especializa: *Responder ao Pop-up*

Nome: Responder não

ID: 95

Descrição: Utilizador responde ao pop-up recebido respondendo que não.

Especializa: *Responder ao Pop-up*

Nome: Adicionar recurso à tarefa.

ID: 96

Descrição: O recurso é adicionado à tarefa caso o utilizador responda sim aos pop-up(s).

Nome: Visualizar Diagrama PERT

ID: 97

Descrição: O utilizador visualiza o diagrama PERT.

Nome: Ver Nó

ID: 98

Descrição: O utilizador visualiza um nó do diagrama PERT, que representa uma tarefa.

Nome: Ver Data de Início

ID: 99

Descrição: O utilizador visualiza a data de início da tarefa no diagrama PERT.

Nome: Ver Data de Fim

ID: 100

Descrição: O utilizador visualiza a data de fim da tarefa no diagrama PERT.

Nome: Ver Duração

ID: 101

Descrição: O utilizador visualiza a duração da tarefa no diagrama PERT.

Nome: Ver Recursos

ID: 102

Descrição: O utilizador visualiza o nome dos recursos associados à tarefa no diagrama PERT.

Nome: Ver Nome

ID: 103

Descrição: O utilizador visualiza o nome da tarefa no diagrama PERT.

## User Stories

Os use cases associados a estas user stories estão descritos nos ficheiros da pasta do Vasco Malta do Sprint 4 da segunda fase.

### Pop-up’s na adição de recursos a uma tarefa

User Story: Enquanto utilizador do GanttProject, quero receber avisos quando a associação de um recurso numa tarefa causa sobrecarga ou quando este se encontra de férias, de forma a evitar erros desnecessários sem ter que estar sempre a verificar a disponibilidade que cada um. (As a user of GanttProject, I want to receive warnings whenever I assign a resource to a task when it causes an overload or when they are on vacation so that I can avoid unnecessary mistakes without having to constantly check each person’s availability.)

No projeto original, quando adicionamos uma pessoa (resource) a uma tarefa que provoca sobrecarga é possível observar no resources chart, porém não existe qualquer tipo de aviso ou maneira de saber enquanto nos encontramos no separador Gantt. O mesmo acontece quando a pessoa se encontra de férias e, portanto, não pode ter uma tarefa delegada nesse período de repouso.

Assim, achámos que seria útil receber notificações no ecrã sempre que criamos sobrecarga, ou que tentamos interromper o período de férias, sendo necessário que o utilizador confirme para proceder com esta ação.

No caso em que tentamos a adicionar uma tarefa a uma pessoa que causa ambos os problemas, inicialmente recebemos o pop-up relativo à sobrecarga e depois o das férias. (Se o utilizador não confirmar o primeiro pop-up, então o outro não chega a aparecer, uma vez que se não está interessado a adicionar uma tarefa numa dada condição não faria sentido mostrar a outra condição.)

Os use cases relativos a esta funcionalidade são os que têm os id’s de 88 a 96, inclusive.

### Visualização de recursos no diagrama PERT

User Story: Enquanto utilizador do GanttProject, quero visualizar facilmente os recursos associados às tarefas no diagrama PERT, de forma a não ter de os procurar individualmente selecionando cada tarefa. (As a user of GanttProject, I want to be able to easily see the resources associated to the tasks on the Pert chart, so that I don’t have to search for them individually by selecting each task.

Originalmente, o diagrama PERT apenas mostrava informações relativas às tarefas (as suas datas de início e fim e a sua duração) e, por isso, se fosse necessário verificar quais eram os recursos associados a essas mesmas tarefas, era preciso mudar para o separador das tarefas e aceder às suas propriedades. Desta forma, conseguimos ver diretamente no diagrama PERT os recursos associados às tarefas, tornando-o mais útil.

Os use cases relativos a esta funcionalidade são os que têm os id’s de 97 a 103, inclusive.

## Notas/Problemas

Para além das duas funcionalidades extra pedidas, decidimos alterar a forma como algumas se encontravam originalmente implementadas.

Corrigiu-se um bug que o projeto tinha. Ao adicionar o mesmo recurso à mesma tarefa várias vezes, dentro da tabela de recursos da tarefa, a adição do recurso não era implementada, enquanto que na tabela dos recursos, mais especificamente na carga do recurso, era, pelo que significava o aparecimento de alguns problemas, como o excesso de carga de um recurso não justificado e impossível de remover, a menos que eliminássemos o próprio recurso, algo que não é muito desejável.

No momento da escolha das funcionalidades a implementar, não nos apercebemos que a funcionalidade de remoção de setas já existia, tendo em conta que a versão do ganttproject que estávamos a utilizar era instável. Para além disso, os botões que permitem fazer as ações de link e unlink de tarefas passaram-nos despercebidos. Assim, decidimos alterar a localização destes botões (agora encontram-se à direita dos botões que permitem adicionar e eliminar tarefas, na barra de ferramentas), de forma a tornar-se mais intuitivo.