Projeto de Engenharia de Software

<https://github.com/anamfrancisco/ESProject.git>

Ana Francisco, 59801, p.2

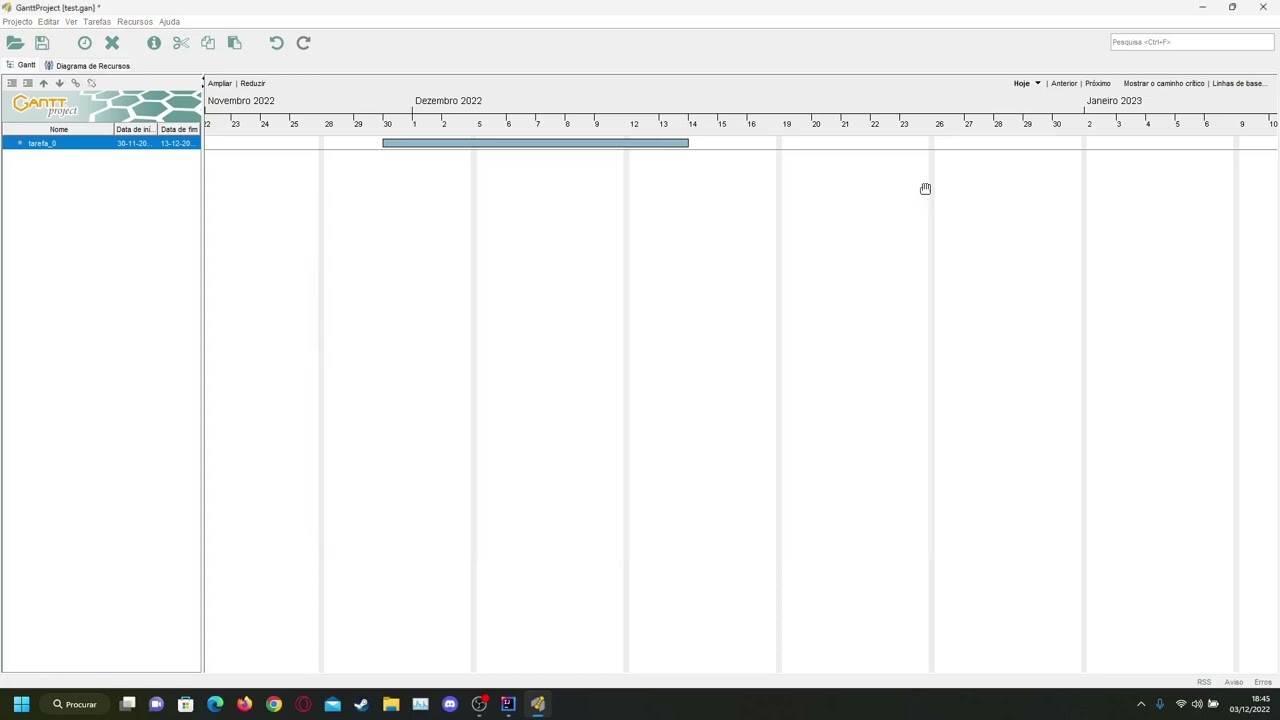
João Amorim, 57409, p.4

João Lima, 60350, p.2

José Pereira, 55204, p.6

Rodrigo Jacob, 55859, p.4

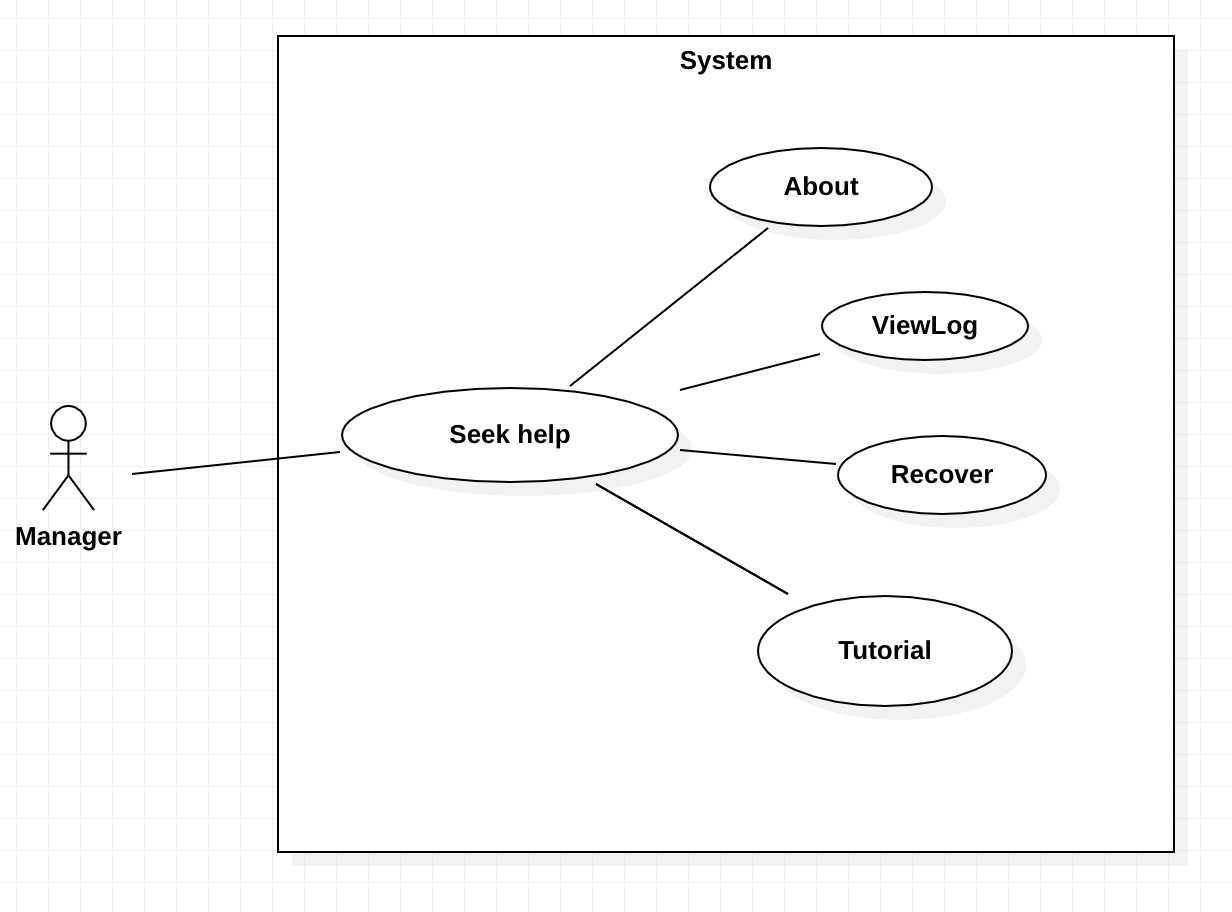
**Vídeo no YouTube a explicar as novas funcionalidades da aplicação:**

**[](https://www.youtube.com/watch?v=WlXnOpGfMW8)**

**Segunda Fase**

Primeira Implementação

User story: Como utilizador, sabendo que esta é a minha primeira vez a usar a aplicação ou a fazer uma certa ação, quero conseguir perceber como utilizar o sistema, poupando tempo ao não ter de explorar o mesmo.

Descrição da funcionalidade: Após entrar na aplicação e clicar no botão “Help”, irá aparecer ao utilizador uma opção “Tutorial”, esta irá reencaminhá-lo para um website que explica como funciona a aplicação Gantt Project.

Use Case ID - Seek help.

Description - Actor clicks the 'Help' button on the top bar.

Primary actor - Manager.

Use Case ID - About.

Description - Opens a pop-up with details about the application version and license.

Primary actor - Manager.

Use Case ID - ViewLog.

Description - Opens a pop-up where we can see the details of the last synchronisation session.

Primary actor - Manager.

Use Case ID - Recover.

Description - Recovers the last project the actor was working on.

Primary actor - Manager.

/\*\*\* New feature \*\*\*/

Use Case ID - Tutorial.

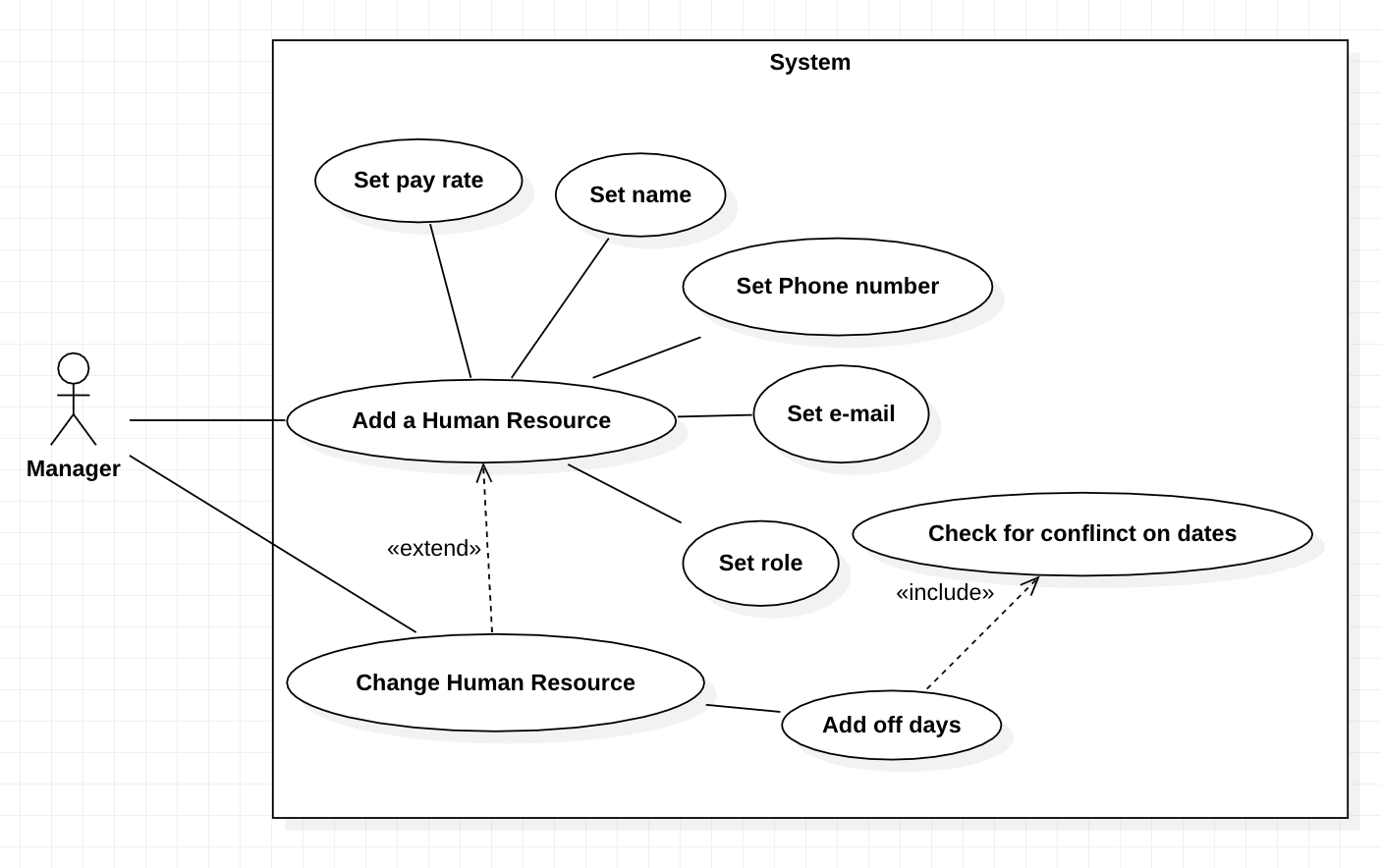
Description - Opens the user's default browser and goes to a url with a description of how the application works.

Primary actor - Manager.

Reviewer: João Lima

Segunda Implementação

User story: Como utilizador, quero que o sistema, ao inserir dias de férias e também ao inserir um trabalhador numa tarefa, faça verificações de conflito de datas, de modo que não haja conflitos entre os dias de férias de um trabalhador e os seus dias de trabalho.

Descrição da funcionalidade: Ao inserir dias de férias, o sistema faz verificações para garantir que os dias de férias pretendidos não estão em conflito com dias de trabalho, também quando o trabalhador é inserido numa tarefa são feita verificações para identificar se tem dias de férias em conflito com os dias de duração de dada tarefa.

Use Case ID - Add a Human Resource.

Description - Creates a Human Resource in the system, this object may have a name, phone number, e-mail, role and a list of the days it pretends to be off, this operation can be canceled.

Primary actor - Manager.

Use Case ID - Set name.

Description - Sets a name for the Human Resource created, if the name given is null for example the name that is set as default in the system.

Primary actor - Manager.

Use Case ID - Set phone number.

Description - Sets a phone number for the Human Resource created.

Primary actor - Manager.

Use Case ID - Set e-mail.

Description - Sets a e-mail for the Human Resource created.

Primary actor - Manager.

Use Case ID - Set role.

Description - Sets a role for the Human Resource created, if the role given is null for example the role given will be undefined.

Primary actor - Manager.

Use Case ID - Set phone number.

Description - Sets a phone number for the Human Resource created.

Primary actor - Manager.

Use Case ID - Set pay rate.

Description - Sets the rate that the Human Resource will be paid.

Primary actor - Manager.

Use Case ID - Change Human Resource.

Description - Changes a Human Resource, extends Add Human Resource being able to set name, phone number, e-mail and role.

Primary actor - Manager.

Use Case ID - Add off days.

Description - Adds to the system the days a Human Resource may want to go on vacation.

Primary actor - Manager.

/\*\*\* New feature \*\*\*/

Use Case ID - Check for conflict on dates.

Description - Checks if the pretended days off are in conflict with any working days, or in other words days that Human Resource has tasks to work on.

Primary actor - Manager.

Reviewer: José Ricardo Pereira

Metrics

Lines of Code Metrics

Author: João Amorim

Reviewer: José Ricardo Pereira

The metrics collected corresponds to the Lines of Code metrics which is a quantitative measurement of a programs files that contain code.

The extreme values observed on the metrics collected might correlate to the code smells that were identified by our team more specifically the ones that are related to dead code and unnecessary comments.

MOOD Metrics

Author: José Ricardo Pereira

Reviewer: Rodrigo Jacob

MOOD stands for Matrices for Object Oriented Design.

Low Coupling factor contributes to better code readability and is a sign of a well-structured computer system.

MHF an AHF are measures of the use of the information-hiding concept supported by the encapsulation mechanism. A very low MHF value denotes implementation of insufficiently abstracted methods whereas a high MHF value denotes very little functionality, so a middle-ground is good. A high value of AHF is always a good sign, since attributes should remain hidden, in general.

Both AIF and MIF are measures of inheritance. This is a mechanism for expressing similarity among classes that allows for the portrayal of generalization and specialization relations.

Polymorphism means having the ability to take several forms. It is supposed to reduce complexity and to allow refinement of the class hierarchy without side effects. The downside is a harder debugging time.

Our project had good values on AHF, CF and PF. The value MHF is within acceptable. The values on AIF and MIF weren’t good.

Martin Packaging Metrics

Author: Rodrigo Jacob

Reviewer: João Amorim

**Efferent Coupling (Ce):**

The high value of the metric Ce > 20 indicates instability of a package, change in any of the numerous external classes can cause the need for changes to the package.

Preferred values for the metric Ce are in the range of 0 to 20, higher values cause problems with care and development of code.

In our case, the package *biz.ganttproject.impex.csv* has a Ce metric of 70, which means it depends on a lot of classes from other packages, making it unstable.

**Afferent Coupling (Ca):**

High values of metric Ca usually suggest high component stability. This is due to the fact that the class depends on many other classes. Therefore, it can’t be modified significantly because, in this case, the probability of spreading such changes increases.

Preferred values for the metric Ca are in the range of 0 to 500.

In our case, the package *net.sourceforge.ganttproject* has a Ca metric of 515, which is outside the optimal range, and classes on this package depend a lot on other classes.

**Instability (I):**

Preferred values for the metric I should fall within the ranges of 0 to 0.3 or 0.7 to 1. Packages should be very stable or unstable, therefore we should avoid packages of intermediate stability.

In our case, every package has a value of either 0 or 1, which is ok.

**Abstractness (A):**

Preferred values for the metric A should take extreme values close to 0 or 1. Packages that are stable (metric I close to 0), which means they are dependent at a very low level on other packages, should also be abstract (metric A close to 1). In turn, the very unstable packages (metric I close to 1) should consist of concrete classes (A metric close to 0).

In our case, the package *net.sourceforge.ganttproject* doesn’t follow this rule, having a value of 0.00 for both I and A metrics, meaning it can’t be extended.

**Normalized Distance from Main Sequence (D):**

The value of the metric D is used to measure the balance between instability and abstractness. It should be as low as possible so that the components are located close to the main sequence.

In our case, some strange results occur regarding this metric, in packages *biz.ganttproject.core.chart.canvas* and *biz.ganttproject.core.calendar*, where A metrics are 0 and I metrics are 1, but somehow the metric D is also 1.

Complexity Metrics

Author: João Lima

Reviewer: Ana Francisco

These metrics (Complexity Metrics -> Cyclomatic Complexity) are related to the number of linearly independent paths that the code can take. For exemple, if a method has an “if else” case, the code can take two paths (the true path, or the false Path), so the total cyclomatic complexity is two. So, when the method revolves in decision making code, the complexity will be higher.

I think the practical values shown are explained by the quantity of unnecessary if’s, loops… Even though there are many classes, so that would explain the total, number being this high. And the Average cyclomatic complexity doesn’t look too high.

Chidamber & Kemerer metrics

Author: Ana Francisco

Reviewer: João Lima

The Chidamber & Kemerer metrics suite originally consists of 6 metrics calculated for each class: WMC, DIT, NOC, CBO, RFC and LCOM1. The original suite has later been amended by RFC´, LCOM2, LCOM3 and LCOM4 by other authors.

**WMC** = number of methods defined in class

**DIT** = maximum inheritance path from the class to the root class

**NOC** = number of immediate sub-classes of a class

**CBO** = a delicious breaded chicken fillet with crispy bacon, fried onions, melted cheese, Iceberg salad (just kidding) CBO is the number of classes to which a class is coupled

**RFC** = M + R (First-step measure)

**RFC’** = M + R’ (Full measure)

**M** = number of methods in the class

**R** = number of remote methods directly called by methods of the class

**R’** = number of remote methods called, recursively through the entire call tree