Use Cases and respective Diagrams

Diagram Manage Project - Luís Abreu, nº60157

This is a diagram showcasing every possible option the user can select to do the project.

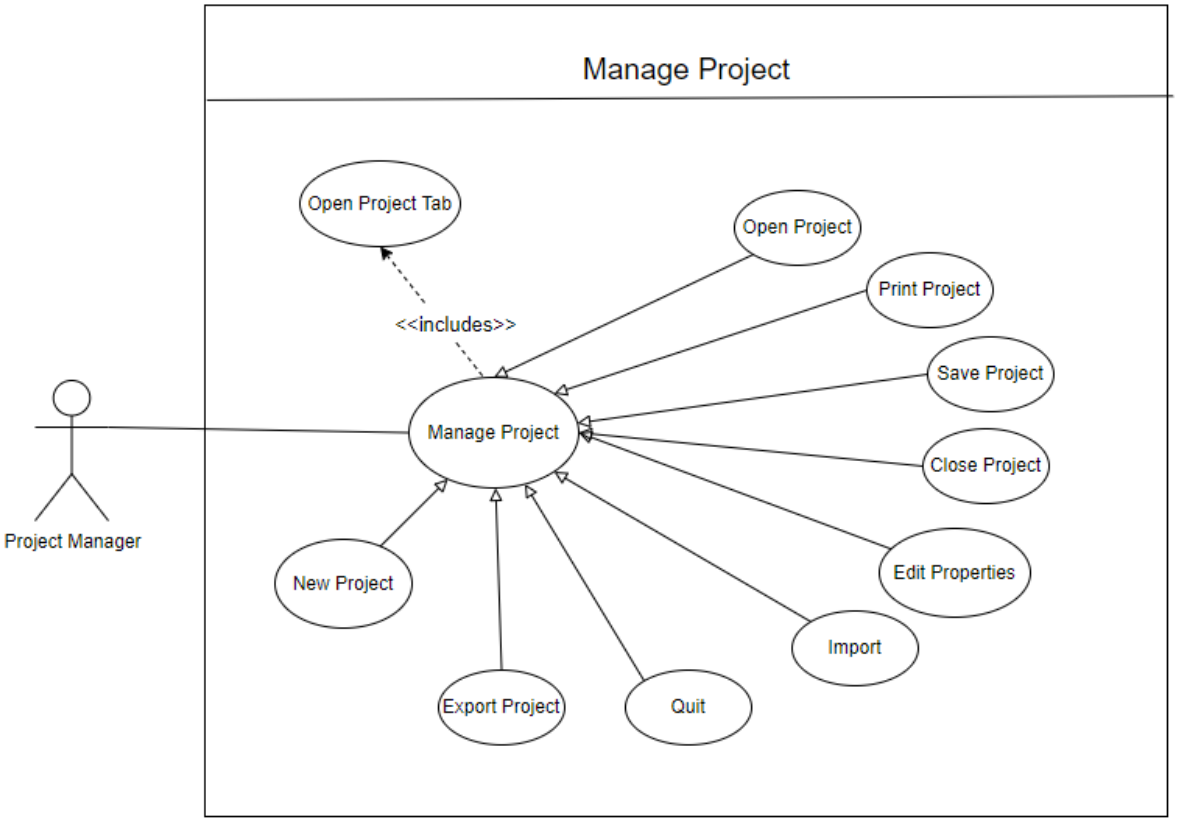


Diagram Edit Properties - Luís Abreu, nº60157

This is a diagram showcasing every property the user can change in the project.

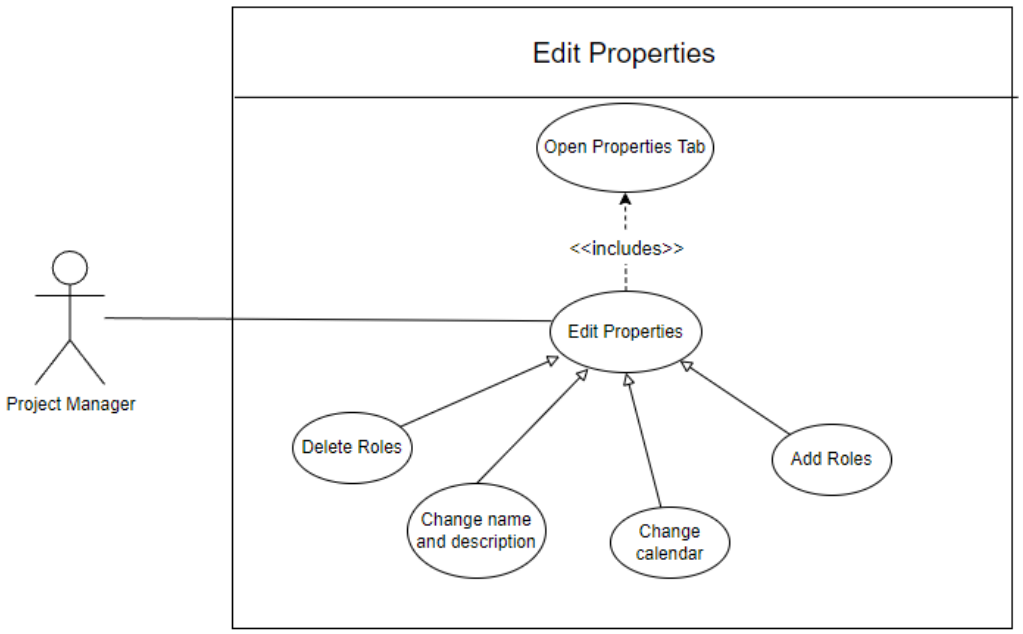
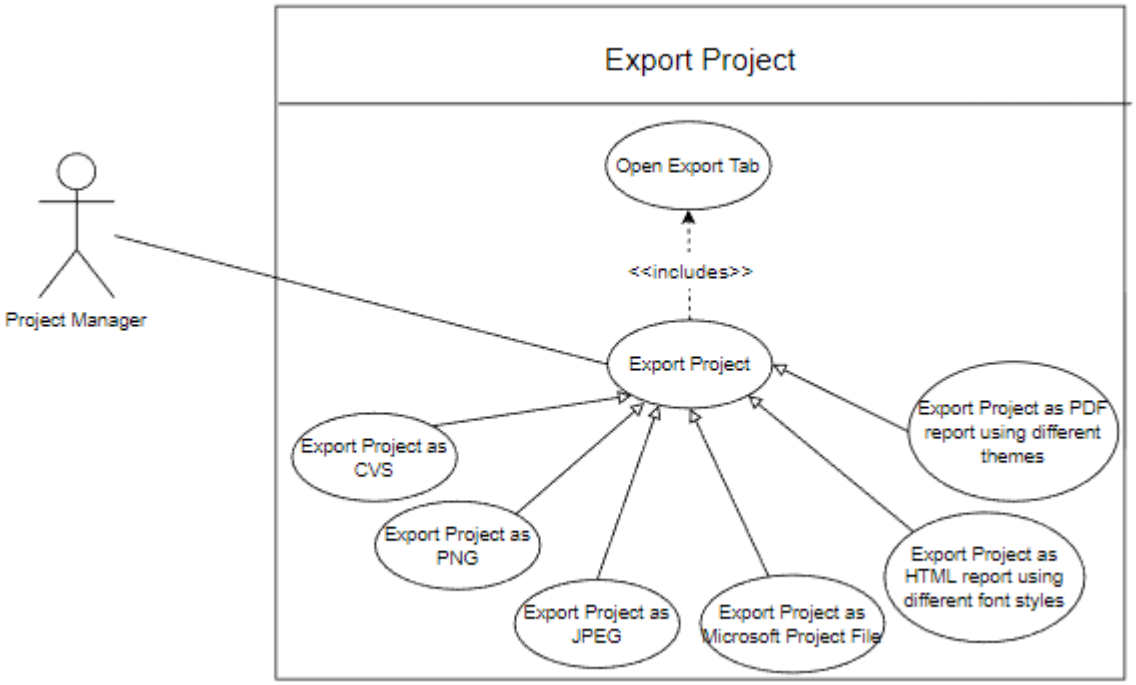


Diagram Export Project - Luís Abreu, nº60157

This is a diagram showcasing every possible “type” of exporting the project.



Use Case Edit Properties - Luís Abreu, nº60157

Name	Edit Properties
ID	1
Description	The project manager changes the project properties
Actors - Primary	Project manager
Actors - Secondary	None
Pre-conditions	A project must exist.
Main Flow	1. The use case starts when the project manager opens the properties tab. 2.The project manager chooses the tab which contains the properties he wants to change 3. System shows the properties on the chosen tab. 4. The project manager edits what he/she wants 5. The project manager Confirms the change by clicking “OK”.
Post - Conditions	The changes the project manager did are applied
Alternative Flows	Properties: Cancel

Use Case Edit Properties: Cancel - Luís Abreu, nº60157

Alternative Flow	Edit Properties:Cancel
ID	1.1
Description	The project manager cancels the changes made to the properties settings
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	The project manager had canceled the changes made to the properties
Main Flow	1. The alternative flow can begin at any point between steps 2-4 during the main flow if the project manager clicks “cancel” 2. The project manager cancels the changes he did previously
Post - Conditions	None

Use Case Change Name and Description- Luís Abreu, nº60157

Name	Change Name and Description
ID	2
Description	The project manager changes the project name and description.
Actors - Primary	Project manager
Actors - Secondary	None
Pre-conditions	A project must exist.
Main Flow	1. The use case starts when the project manager opens the properties tab. 2. (o2.)The project manager chooses the “general” tab 3. System shows the properties on the chosen tab. 4. (o4.) The project manager edits the name of the project or its description 5. The project manager Confirms the change by clicking “OK”.
Post - Conditions	The changes the project manager did are applied
Alternative Flows	Change Name and Description: Cancel

Use Case Change Name and Description: Cancel - Luís Abreu, nº60157

Alternative Flow	Change Name and Description:Cancel
ID	2.1
Description	The project manager cancels the changes made to the properties settings
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	The project manager had canceled the changes made to the properties
Main Flow	1. The alternative flow can begin at any point between steps 2-4 during the main flow if the project manager clicks “cancel” 2. The project manager cancels the changes he did previously
Post - Conditions	None

Name	Export Project
ID	3
Description	The project manager exports the project
Actors - Primary	Project manager
Actors - Secondary	None
Pre-conditions	A project must exist.
Main Flow	1. The use case starts when the project manager opens the export tab. 2.The project manager toggles the type of file he wants the project to be exported 3. The project manager can choose more specific settings on the type of file que chose to export 3. The project manager clicks “next” 4. The system shows some more options that the project manager can chose to take advantage of. 5. The project manager chooses the directory that he wants to export the project to. 6. The project manager Confirms the export by clicking “ok”. 7. The system exports the project to the location the project manager specified
Post - Conditions	None.
Alternative Flows	Export: Cancel

Use Case Properties: Cancel - Luís Abreu, nº60157

Alternative Flow	Export:Cancel
ID	3.1
Description	The project manager cancels the export that he was going to do
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	The project manager had canceled the exportation of the projection
Main Flow	1. The alternative flow can begin at any point between steps 2-6 during the main flow if the project manager clicks “cancel” 2. The project manager cancels the export of the project
Post - Conditions	None

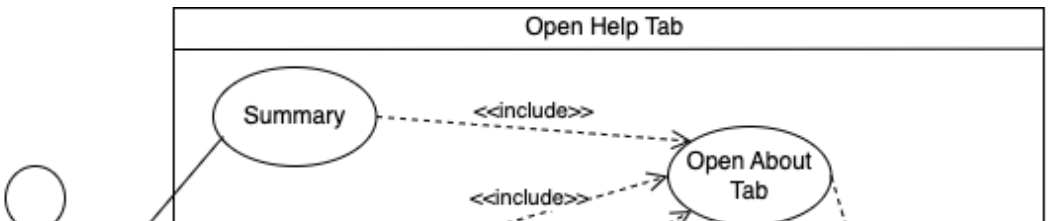
Use Case Open Export Tab- Luís Abreu, nº60157

Name	Open Export Tab

ID	4
Description	The project manager opens the export tab
Actors - Primary	Project manager
Actors - Secondary	None
Pre-conditions	A project must exist.
Main Flow	1. The use case starts when the user selects the project tab 2.The project manager selects the export tab 3. System shows the export options on the chosen tab.
Post - Conditions	None.
Alternative Flows	None.

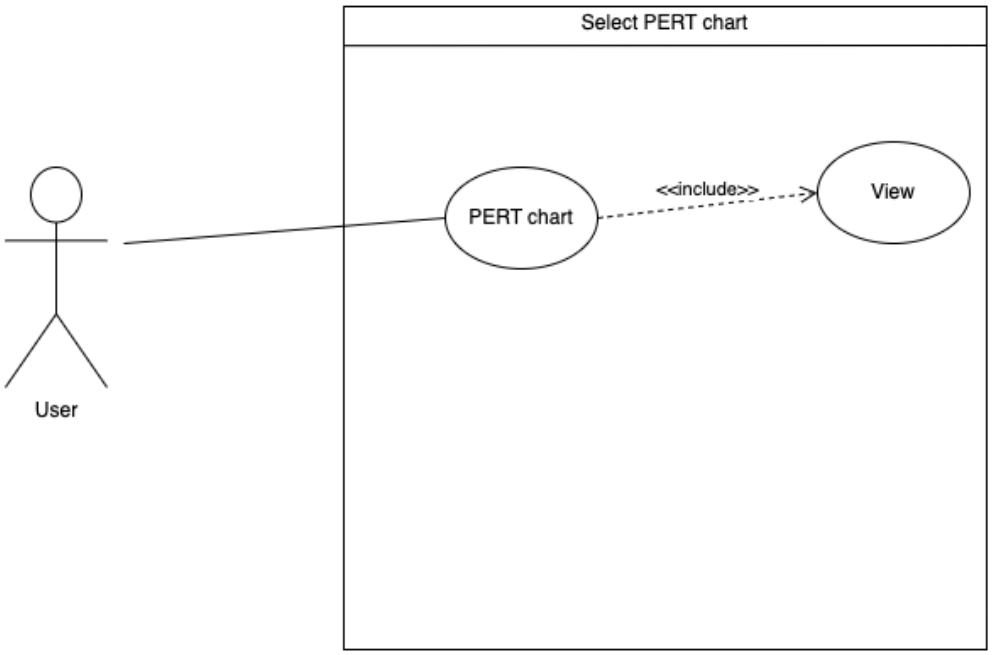
Use case Diagram Help - Tiago Francisco, nº60749

This diagram describes the functionalities available in the help tab.



Use case Diagram Select PERT chart - **Tiago Francisco, nº60749**

This diagram describes how a user can make the PERT chart appear.



Use Case Help option- Tiago Francisco nº60749

Name	Help
ID	5
Description	The user chooses help tab
Actors - Primary	User
Actors - Secundary	None
Pre - conditions	None
Main Flow	1. The user selects the help option. 2. The system shows the functionalities available in help.
Post - Conditions	None
Alternative flows	None.

Use Case About - Tiago Francisco nº60749

Name	About
ID	6
Description	The user chooses about tab
Actors - Primary	User
Actors - Secundary	None
Pre - conditions	None
Main Flow	1. Include(Help) 2. The user selects “about” option 3. The system shows the functionalities available in “about” 4. The use case finishes when “OK” button is pressed
Post - Conditions	None
Alternative flows	At any point the user may choose to cancel and exit the menu

Use Case Summary- Tiago Francisco nº60749

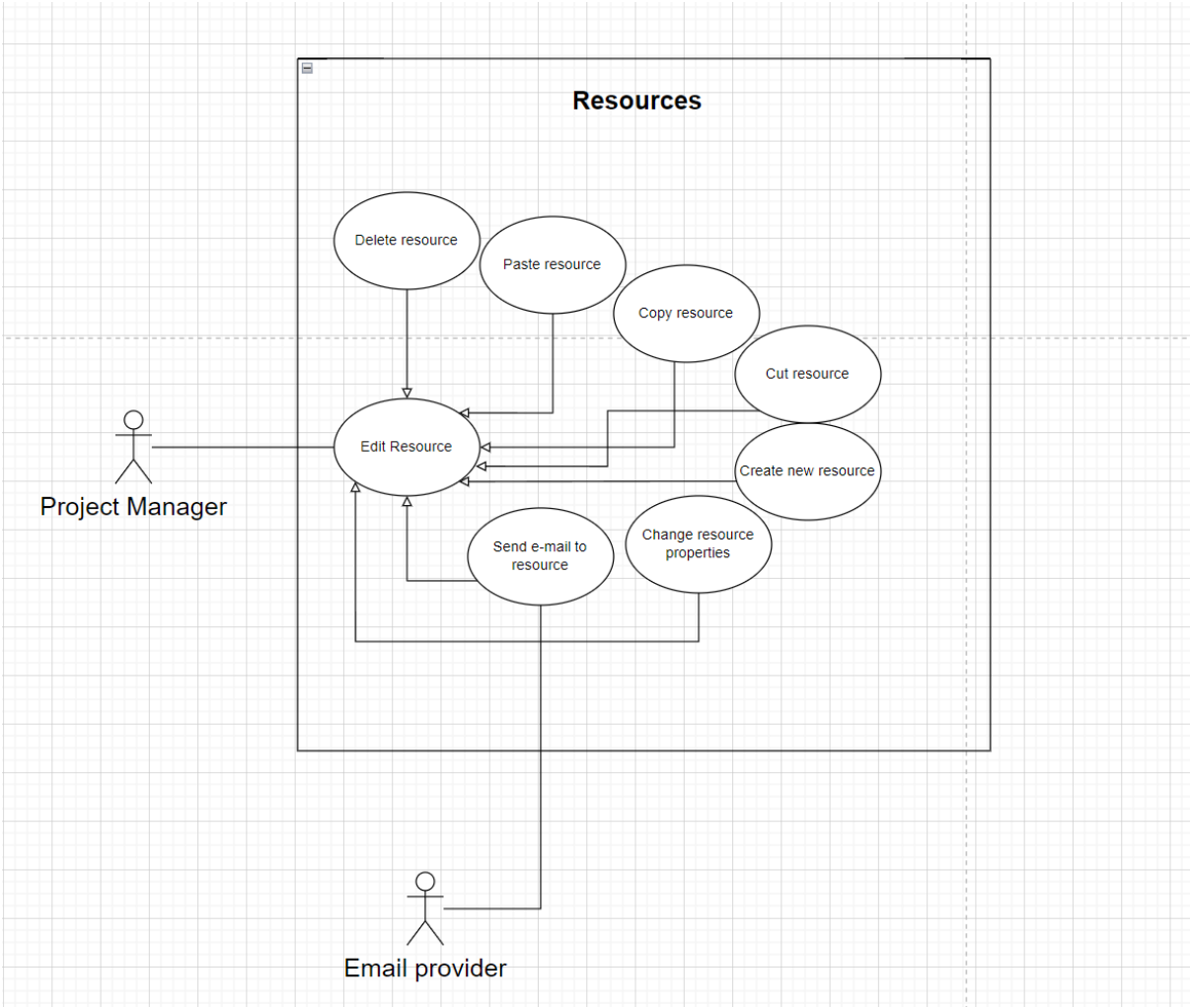
Name	Summary
ID	7
Description	The user chooses License tab
Actors - Primary	User
Actors - Secundary	None
Pre - conditions	None
Main Flow	1. Include(About) 2. The user selects the summary option 3. The system shows the information presented
Post - Conditions	None
Alternative flows	None.

Use Case PERT chart- Tiago Francisco nº60749

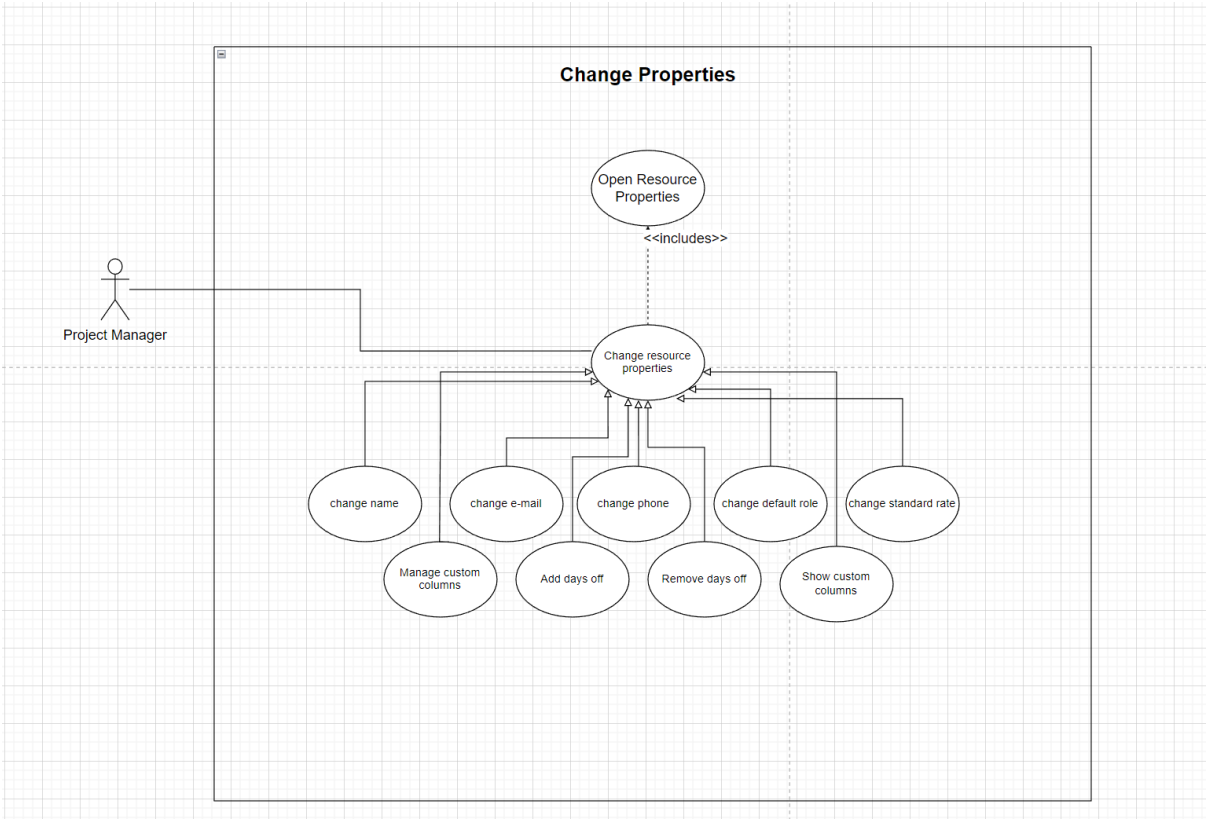
Name	PERT chart
ID	8
Description	The user chooses License tab
Actors - Primary	User
Actors - Secundary	None
Pre - conditions	None
Main Flow	1. Include(view) 2. The User selects PERT chart 3. The system shows the PERT chart
Post - Conditions	None
Alternative flows	None

Use Case Diagram Resources - Pedro Gouveia, nº 60479

This is a diagram showcasing every possible interaction with the system the user can have using a resource.



Use Case Diagram Change Resource Properties - Pedro Gouveia, nº 60479



This is a Sub diagram dedicated to the editing of a resource’s properties, since there are a lot of possible changes a user can make to a resource.

Use Case Delete Resource- Pedro Gouveia, nº 60479

Name	Delete Resource
ID	9
Description	The user deletes a resource
Actors - Primary	User
Actors - Alternative	None
Pre - conditions	Existing resource.
Main Flow	1. The use case starts when the user selects the resource chart menu. 2. The user selects the desired resource. 3. The user selects the options on the resource. 4. The user selects the ‘delete resource‘ option.
Secondary Flows	None.
Post - Conditions	The resource is removed.

Use Case Add New Resource - Pedro Gouveia, nº 60479

Name	Add new resource
ID	10
Description	The user creates a new resource in the system
Actors - Primary	User
Actors - Secondary	None
Pre - conditions	None
Main Flow	1.The use case starts when the user presses the button New Resource. 1.2 The user chooses the name of the resource 1.3 The user chooses an email 1.4 The user chooses a phone number 2. The system creates a new resource and places it in the diagram.
Post - Conditions	The resource is created in the system.
Alternative Flows	None

Use case Open resource properties- Pedro Gouveia, nº 60479

Name	Open resource properties
ID	11
Description	The project manager asks the system a certain resource's information
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	The resource must exist
Main Flow	1. The use case starts when the project manager choses a resource. 2. The project manager asks the system to show the resource's properties 3. The system returns the resource's properties.
Alternative Flows	None
Post - Conditions	None

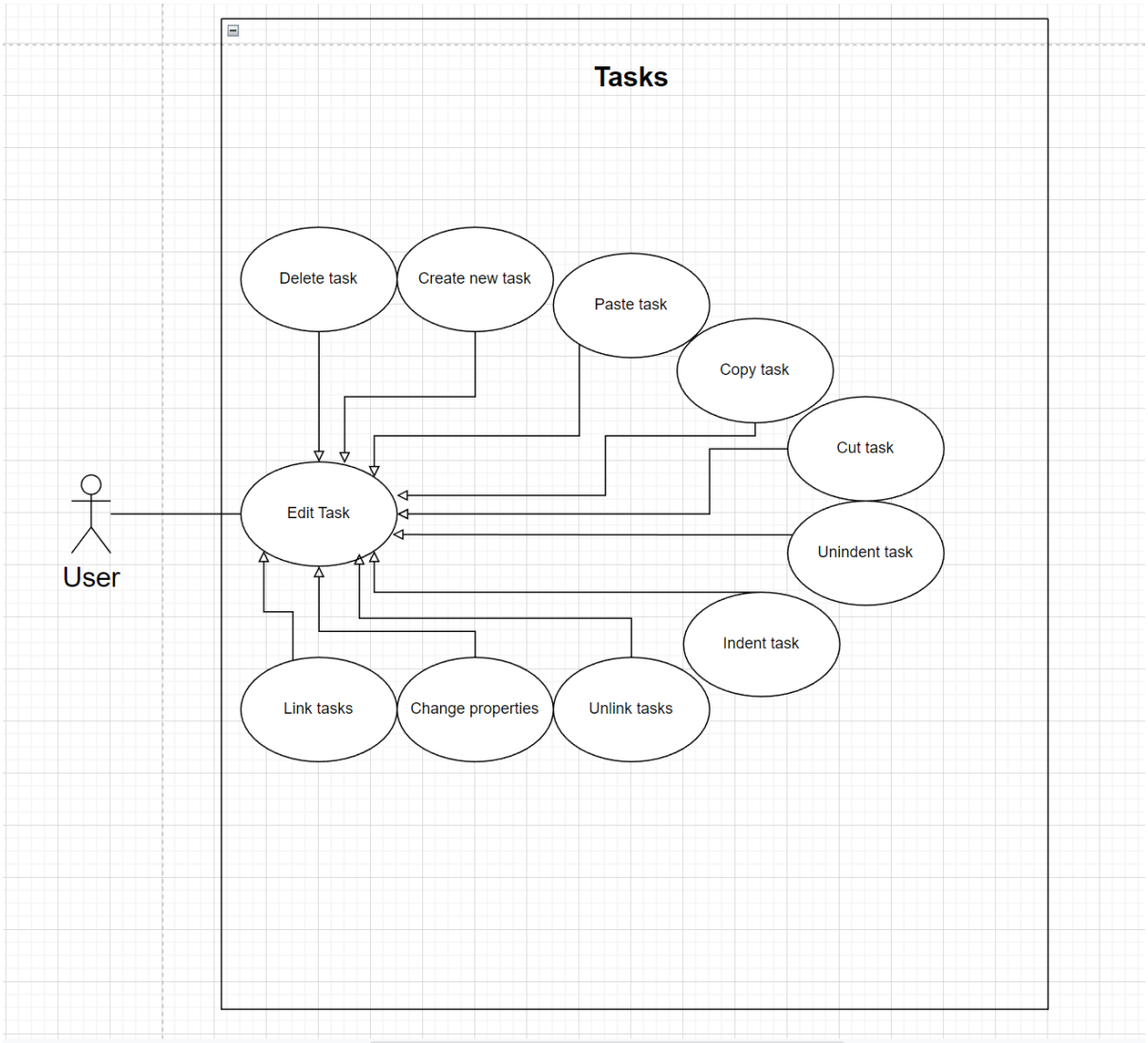
Name	Change properties
ID	12
Description	The project manager changes a resources' property
Actors - Primary	Project manager
Actors - Secondary	None
Pre-conditions	Resource exists
Main Flow	1. The use case starts when the project manager choses a resource. 2. Include (Open resource Properties) . 3. The project manager selects the tab that contains the property he wishes to change 4. The system shows the properties under the selected tab 5. The project manager chooses the property he wishes to change 6. The project manager inputs the new value. 7. The project manager Confirms the change by clicking "ok".
Post - Conditions	The resource has a new property.
Alternative Flows	Change Resource Properties:Cancel

Alternative Flow	Change Resource Properties:Cancel

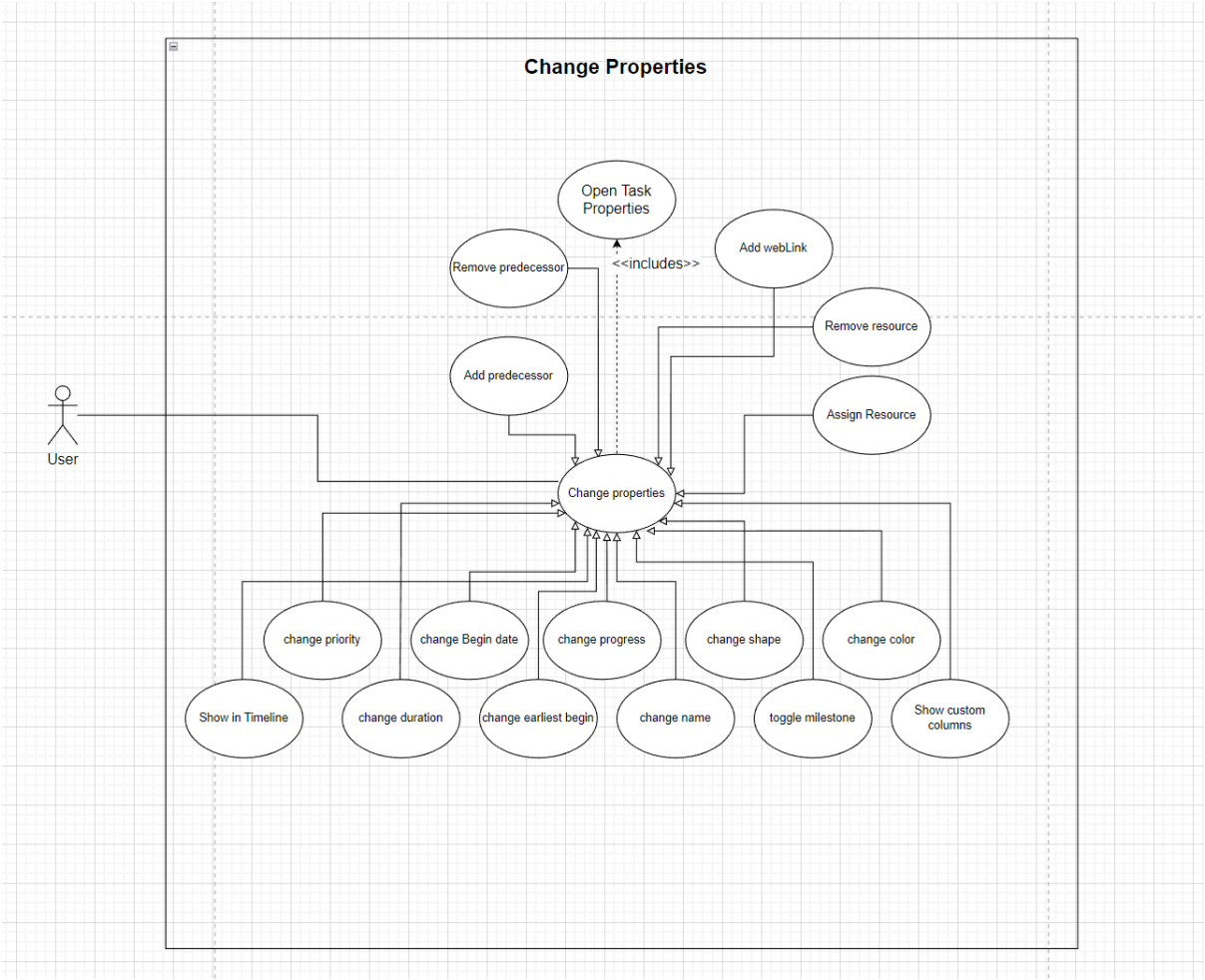
ID	12.1
Description	The project manager cancels the changes made to the settings
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	The project manager had canceled the changes made to the properties
Main Flow	1. The alternative flow can begin at any point during the main flow if the project manager clicks “cancel” 2. The project manager cancels the changes he did previously
Post - Conditions	None

Name	Change e-mail
ID	13
Description	The project manager changes a resource's e-mail
Actors - Primary	Project manager
Actors - Secondary	None
Pre-conditions	Resource exists
Main Flow	1. The use case starts when the project manager choses a resource. 2. Include (Open resource Properties) . 3. (o3) The project manager selects General tab 4. (o4)The system shows the properties under the General tab 5. (o5) The project manager clicks on the e-mail field. 6. (o6) The project manager inputs the new e-mail he wants to assign to the resource. 7. (o7) The project manager Confirms the change by clicking "ok".
Post - Conditions	The resource has a new e-mail.
Alternative Flows	None

This is a diagram showcasing every possible interaction with the system the user can have using a task.



Use Case Diagram Change Properties - Guilherme Franco nº 60226



This is a Sub diagram dedicated to the editing of a tasks properties, since there is a lot of possible changes a user can make to a task

Use Case Change properties - Guilherme Franco, nº60226

Name	Change properties
ID	14
Description	The project manager changes a tasks' property
Actors - Primary	Project manager
Actors - Secondary	None
Pre-conditions	Task exists
Main Flow	1. The use case starts when the project manager choses a task. 2. Include (Open Task Properties) . 3. The project manager selects the tab that contains the property he wishes to change 4. The system shows the properties under the selected tab 5. The project manager chooses the property he wishes to change 6. The project manager inputs the new value or toggles the existing value. 7. The project manager Confirms the change by clicking "ok".
Post - Conditions	The task has a new property.
Alternative Flows	Change properties: Cancel

Alternative Flow	Change Properties:Cancel
ID	14.1
Description	The project manager cancels the changes made to the settings
Actors - Primary	Project manager
Actors - Secondary	None.
Pre - conditions	The project manager had canceled the changes made to the properties
Main Flow	1. The alternative flow can begin at any point during the main flow if the project manager clicks “cancel” 2. The project manager cancels the changes he did previously
Post - Conditions	None.

Name	Assign resource
ID	15
Description	The project manager assigns a resource
Actors - Primary	Project manager
Actors - Secondary	None
Pre-conditions	Task exists and Resource exists
Main Flow	1. The use case starts when the project manager choses a task. 2. Include (Open Task Properties) . 3. (o3) The project manager selects resources tab 4. (o4)The system shows the properties under the resources tab 5. (o5) The project manager chooses the add option 6. (o6) The project manager inputs the resource name he wants to assign to the task. 7. (o7) The project manager Confirms the change by clicking “ok”.
Post - Conditions	The task has a new resource.
Alternative Flows	None

Use case Open task properties- Guilherme Franco, nº60226

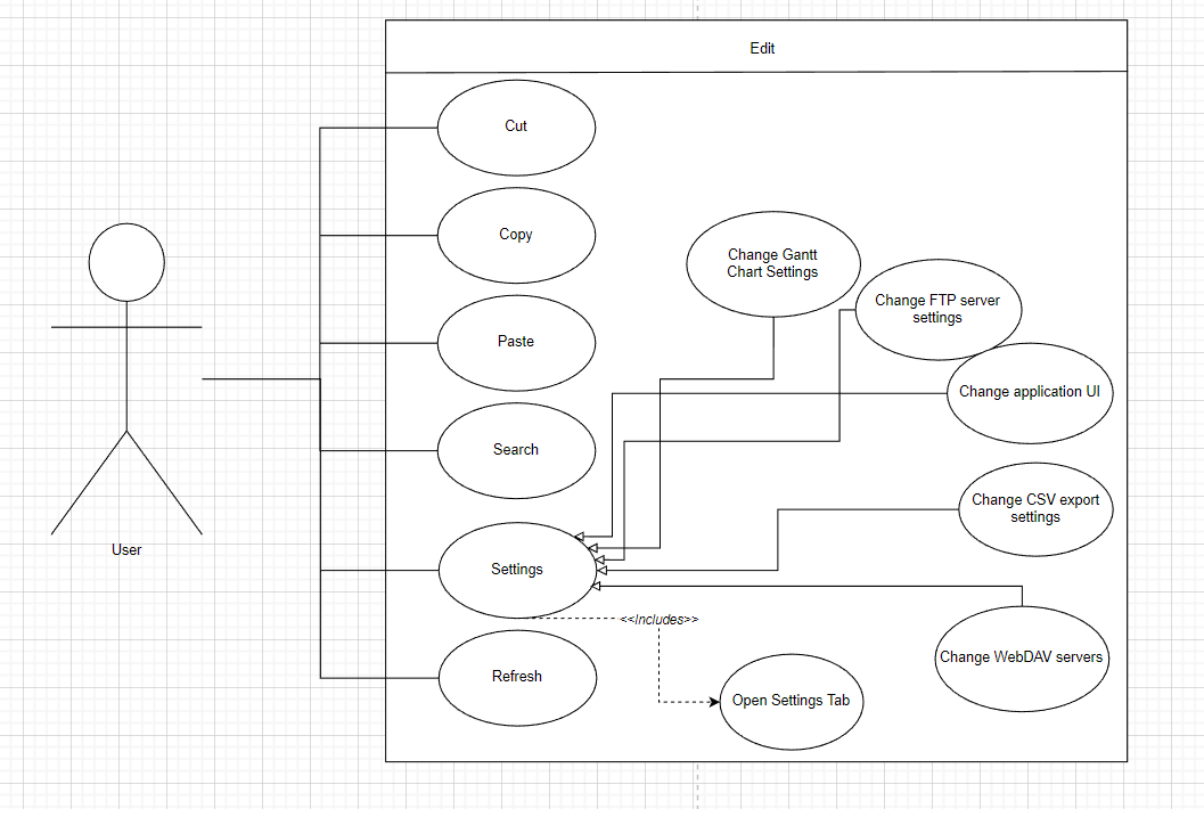
Name	Open task properties
ID	16
Description	The project manager asks the system a certain task's information
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	The task must exist
Main Flow	1. The use case starts when the project manager choses a task. 2. The project manager asks the system to show the task's properties.. 3. The system returns the task's properties.
Alternative Flows	None
Post - Conditions	None

Name	Link tasks
ID	17
Description	The project manager links one or more tasks
Actors - Primary	Project manager
Actors - Secondary	None
Pre-conditions	Task exists and Resource exists
Main Flow	1. The use case starts when the project manager choses one or more tasks. 2. The user asks the system to link them together 3. The system according to the order(top-down) will make a chain of predecessores
Post - Conditions	The gantt chart has new dependencies
Alternative Flows	None

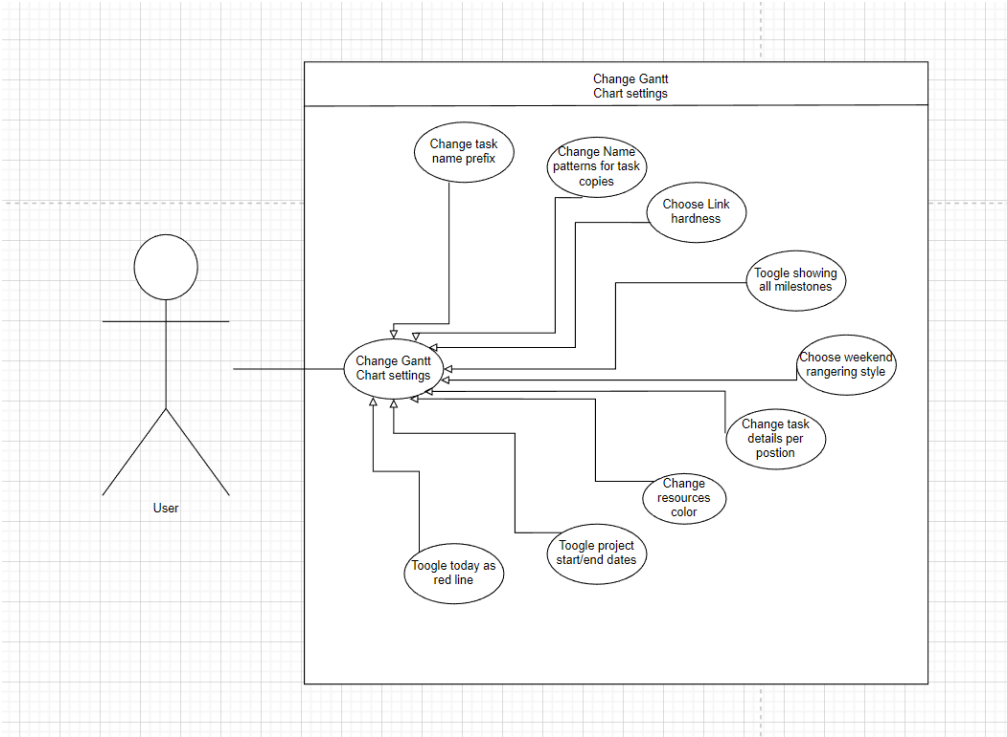
Name	Add new task
ID	18
Description	The user creates a new task in the system
Actors - Primary	User
Actors - Secundary	None
Pre - conditions	None
Main Flow	1.The use case starts when the user presses the button New task. 2. The user assigns a name to the Task that he wants to create 3.The system creates a new task and places it in the diagram.
Post - Conditions	The task is created in the system.
Secondary Flows	None

Use Case Diagram : Edit - Francisco Vale nº 60201

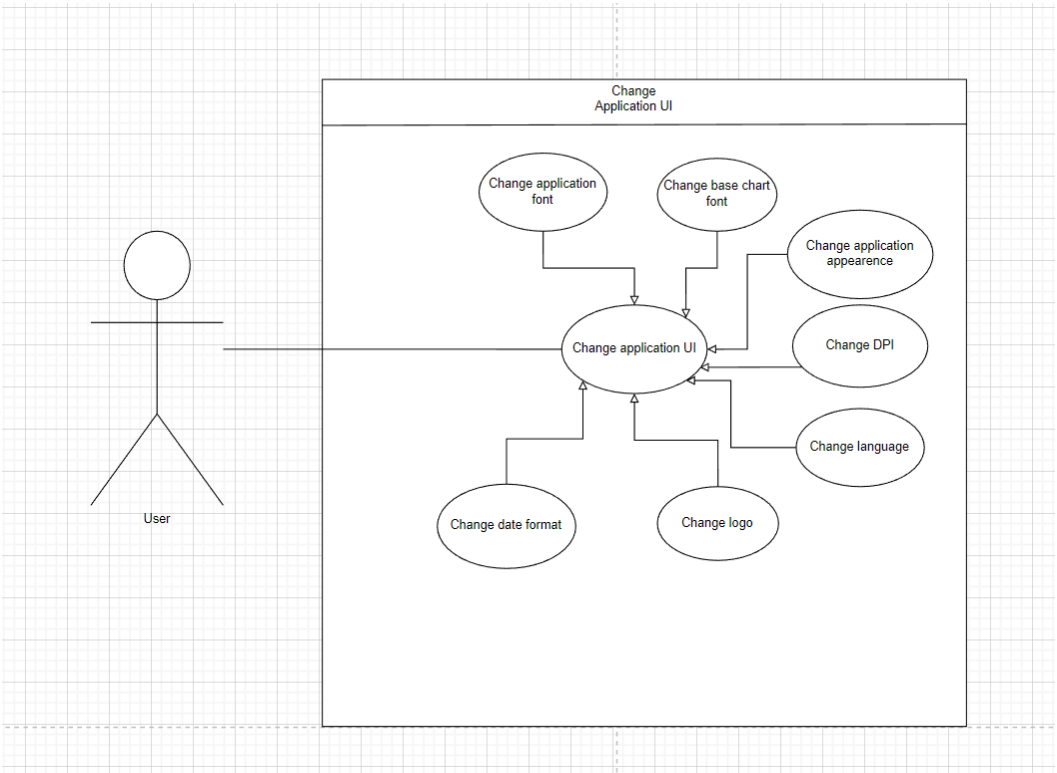
This Use Case represents the possibility offered by the system for the project manager to change the settings of the project, such as Idiom, Font and Aspect, cut and paste either a task or a resource or just search for a specific object. Since the setting changes are all pretty similar they all inherit from a more general Use Case.



Use Case Diagram : Change Gantt Chart Settings- Francisco Vale nº 60201



Use Case Diagram : Change Application UI: Settings- Francisco Vale nº 60201



Use Case : Change Setting- Francisco Vale nº 60201

Name	Change Setting
ID	19
Description	The project manager modifies a setting
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	Setting exists
Main Flow	1. Include(Open Settings Tab) 2. The project manager selects the tab that contains the setting he wants to modify 3. The system shows every setting under the selected tab. 4. The project manager selects the setting that he desires to modify, by clicking on the button on the right. 5. The application shows a plethora of options to choose from. 6. The project manager selects the option which best suits his intentions. 7. The project manager clicks in OK.
Post - Conditions	The project has now some settings modified.

Alternative Flow	Change Setting:Cancel

ID	19.1
Description	The project manager cancels the changes made to the settings
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	The project manager had canceled the the changes made to the settings
Main Flow	1. The alternative flow begins after step 5 of the main flow. 2. The project manager cancels the changes he did previously
Post - Conditions	None

Use Case : Change Application UI- Francisco Vale nº 60201

Name	Change Application UI
ID	20
Description	The project manager modifies a Application UI
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	Setting exists
Main Flow	1. Include (Open Settings Tab) 2. (o2) The project manager selects the application tab 3. The system shows every setting under the application tab 4. The project manager selects the setting that he desires to modify, by clicking on the button on the right. 5. The application shows a plethora of options to choose from. 6. The project manager selects the option which best suits his intentions. 7. The project manager clicks in OK.
Post - Conditions	The project has now some settings modified.

Use Case : Change Font- Francisco Vale nº 60201

Name	Change Font
ID	21
Description	The project manager modifies the project font.
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	Font exists
Main Flow	Include(Open Settings Tab) 2. The project manager selects the application Tab 3. The system shows every setting under the application Tab. 4.(o4.) The project manager selects the font that he desires to modify, by clicking on the button on the right. 5.(o5.) The application shows a plethora of fonts to choose from. 6.(o6.) The project manager selects the font which best suits his intentions. 7. The project manager clicks in OK.
Post - Conditions	The project has now its font modified.
Alternative Flows	Change Setting:Cancel

Use Case : Open settings- Francisco Vale nº 60201

Name	Open Settings Tab
ID	22
Description	The project manager opens the settings tab.
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	None
Main Flow	1. The use case starts when the Project manager selects the edit tab in the upper part of the application. 2. The system will show a variety of options to click on. 3. The project owner selects the last option, Settings. 4. The system opens a new window with many settings that can be modified.
Post - Conditions	A new window with changeable settings is open.
Alternative Flows	None

Use Cases of the New Implementations

Use Case : Export to Calendar - Pedro Gouveia, nº 60479

Name	Export to Calendar
ID	23
Description	The project manager exports the current tasks to his google calendar.
Actors - Primary	Project manager
Actors - Secondary	Google
Pre - conditions	There is a project currently opened
Main Flow	1. The use case starts when the Project manager selects the project tab in the upper part of the application. 2. The system will show a variety of options to click on. 3. The Project manager selects the export option. 4. TheProject manager selects the Google Calendar Option 4.1 If there is no account logged in: 4.1.1 The user must log in to his google account. 4.1.2 Google will ask to validate the account. 4.1.3 The Project manager validates the account and returns to the app. 4.2 The Project manager clicks “ok”

	4.3 The system connects the tasks to his google Calendar. 4.4 The resources that are allocated to the task in the case that they got an email associated receive an invitation to the task.
Post - Conditions	None.
Alternative Flows	The project manager can cancel the action between the steps 2 - 4.2 of the main flow. Export to Google Calendar: No account logged in

Alternative Flow	Export to Google Calendar: No account logged in
ID	19.1
Description	The project manager cancels the changes made to the settings
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	The project manager had canceled the the changes made to the settings
Main Flow	1. The alternative flow begins after step 5 of the main flow. 2. The project manager cancels the changes he did previously
Post - Conditions	None

Use Case : Export Burndown Chart to XLSX - Francisco Vale nº 60201

Name	Export Burndown Chart to XLSX
ID	24
Description	The project manager exports the current project to a XLSX file, containing the Burndown chart related to it.
Actors - Primary	Project manager
Actors - Secondary	None
Pre - conditions	There is a project currently opened
Main Flow	1. The use case starts when the Project manager selects the project tab in the upper part of the application. 2. The system will show a variety of options to click on. 3. The Project manager selects the export option. 4. TheProject manager selects the BurndownChart in the XLSX file option. 5. The user clicks on next. 5.1. The user selects the place where he wants to save the file. 5.2. The user chooses if he wants to public the files in FTP 6. The user clicks in ok.

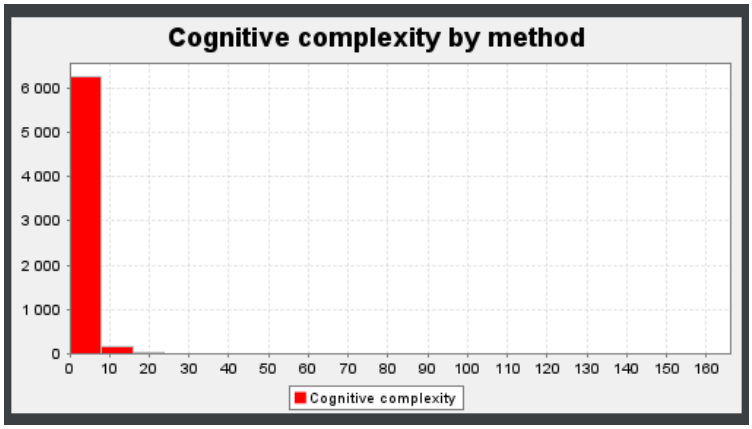
	7. The system will download the file to the user's machine and the use case ends
Post - Conditions	None.
Alternative Flows	The project manager can cancel the action between the steps 2 - 4.2 of the main flow. The Directory where the user wants to save the file might not be available, in step 5.1.

Metrics

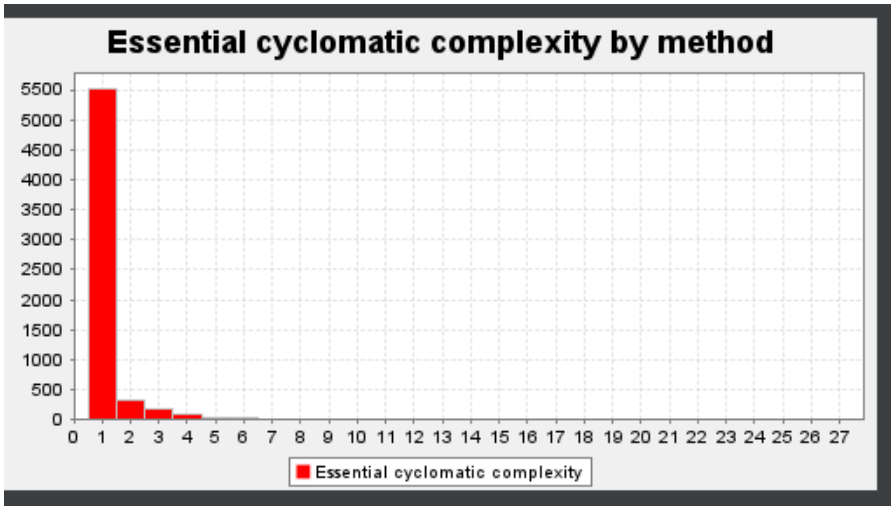
Complexity Metrics - Luís Abreu, nº60157

CogC: Calculates the Cognitive Complexity of each non-abstract method. The metric is similar to Cyclomatic Complexity, but is intended to explicitly measure understandability, which can be quite different from testability. Cognitive Complexity is increased with each control structure used and is higher the more nested control structures are.

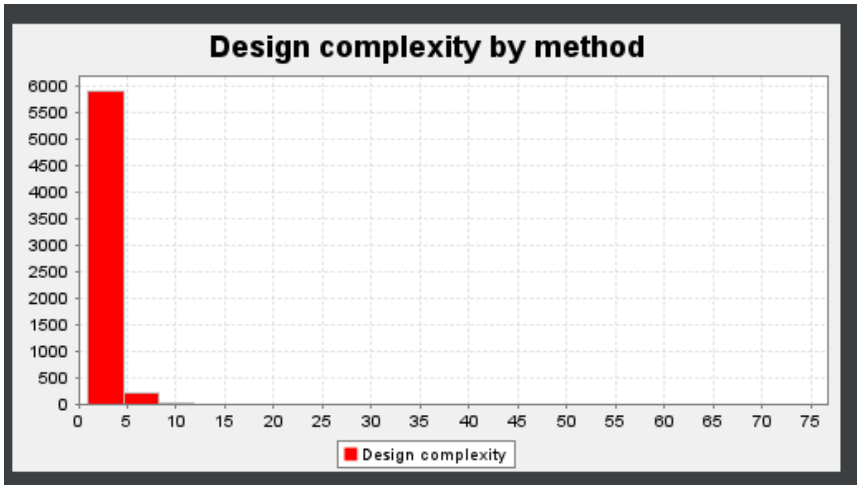
To put it in a simple way, Cognitive Complexity is a measure of how difficult the code will be to read and understand .



ev(G):Calculates the Essential Complexity of each non-abstract method. Essential Complexity is a graph-theoretic measure of just how ill-structured a method's control flow is. Essential Complexity ranges from 1 to v(G), the Cyclomatic Complexity of the method.

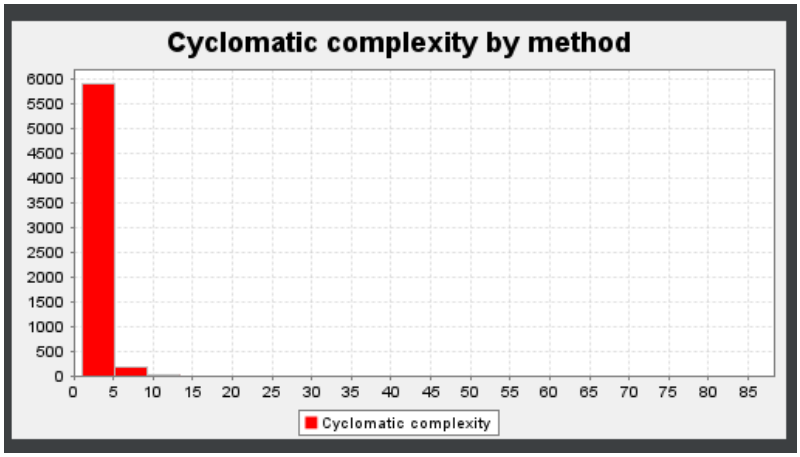


iv(G): Calculates the design complexity of a method. The design complexity is related to how interlinked a methods control flow is with calls to other methods. Design complexity ranges from 1 to v(G), the cyclomatic complexity of the method. Design complexity also represents the minimal number of tests necessary to exercise the integration of the method with the methods it calls.



v(G): Calculates the Cyclomatic Complexity of each non-abstract method. Cyclomatic complexity is a measure of the number of distinct execution paths through each method. This can also be considered as the minimal number of tests necessary to completely exercise a method's control flow. In practice, this is 1 + the number of if's, while's, for's, do's,switch cases, catches, conditional expressions, &&'s and ||'s in the method.

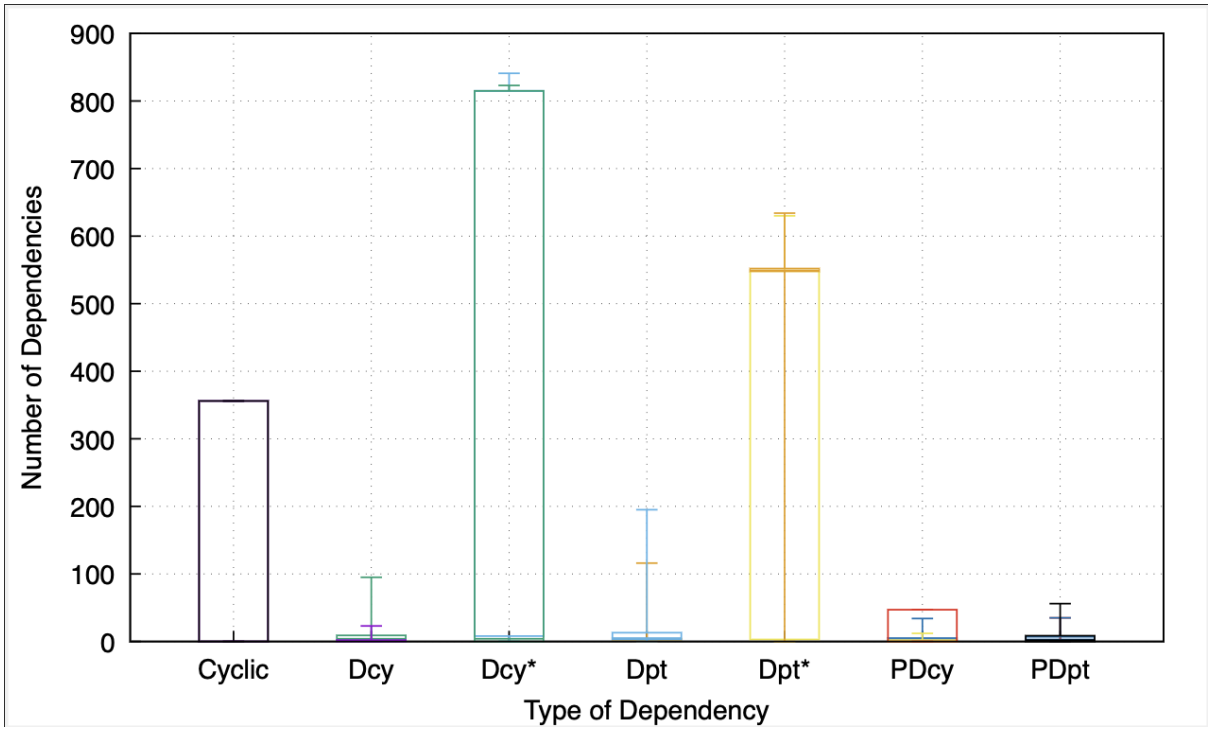
To put it in a simple way, Cyclomatic Complexity determines how difficult your code will be to test.



The graphic of cognitive complexity, as we can see above, shows that most methods have values of cognitive complexity between 0 and 10, which is good, once that cognitive complexity at a method level has 15 as a recommended maximum.

Reviews: Tiago Francisco: The explanation of all complexity metrics are very clear. In the conclusion I would just try to describe a little more beyond being inside the recommendation.

Dependency Metrics - Francisco Vale nº 60201



Cyclic: Calculates the number of classes or interfaces which each class directly or indirectly depends on, and which in turn directly or indirectly depend on it. Such cyclic dependencies may result in code which is difficult to understand and test.

Dcy: Calculates the number of classes or interfaces which each class directly depends on.

Dcy*: Calculates the number of classes or interfaces which each class directly or indirectly depends on.

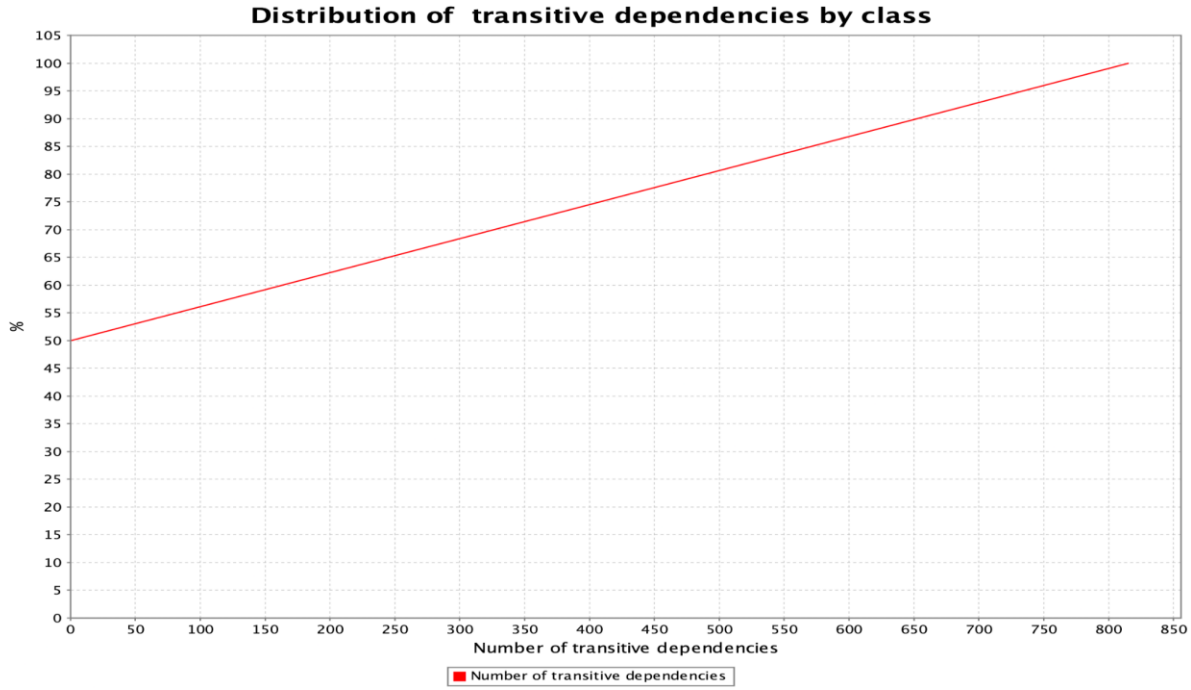
Dpt: Calculates the number of classes or interfaces which directly depend on each class.

Dpt*: Calculates the number of classes or interfaces which directly or indirectly depend on each class.

PDcy: Calculates the number of packages on which each class directly or indirectly depends.

PDpt: Calculates the number of packages which directly or indirectly depend on each class.

As we can see there are some exceptional high values in the number of interfaces and classes which each class directly or indirectly depends on. Searching for those classes I found the one which stood out, the Task Interface.



This Interface has around 800 classes dependent on it, directly or indirectly and approximately 50% of every class has some kind of dependency. This might bring out some code smells and so, if we intended to refactor our code and to test it this should be one of the Interfaces to check first.

Reviews

Luís Abreu: All the concepts present in this metric are well explained. Furthermore, the analysis that Francisco did to the class Task and the relation to a possible code smell is very well identified and explained.

Martin Packaging Metrics - Pedro Gouveia, nº 60479

Packages	Abstractness	Afferent couplings	Efferent couplings	Distance from the main	Instability
net.sourceforge.ganttproject.test.task	0.1111111111111	444.0	9.0	0.8690213392200148	0.01986754966887
biz.ganttproject.core.calendar	0.0	0.0	0.0	1.0	1.0
biz.ganttproject.core.chart.canvas	0.0	0.0	0.0	1.0	1.0
biz.ganttproject.impex.csv	0.0	0.0	70.0	0.0	1.0
biz.ganttproject.impex.msproject2	0.0	0.0	5.0	0.0	1.0
net.sourceforge.ganttproject	0.0	515.0	0.0	1.0	0.0
net.sourceforge.ganttproject.action.task	0.0	0.0	19.0	0.0	1.0
net.sourceforge.ganttproject.chart	0.0	0.0	82.0	0.0	1.0
net.sourceforge.ganttproject.chart.gantt	0.0	0.0	36.0	0.0	1.0
net.sourceforge.ganttproject.chart.mouse	0.0	0.0	12.0	0.0	1.0
net.sourceforge.ganttproject.customProperty	0.0	0.0	0.0	1.0	1.0
net.sourceforge.ganttproject.document.webdav	0.0	0.0	0.0	1.0	1.0
net.sourceforge.ganttproject.gui.taskproperties	0.0	0.0	7.0	0.0	1.0
net.sourceforge.ganttproject.resource	0.0	0.0	12.0	0.0	1.0
net.sourceforge.ganttproject.roles	0.0	0.0	0.0	1.0	1.0
net.sourceforge.ganttproject.task.algorithm	0.0	2.0	365.0	0.00544959128065392	0.99455040871934
net.sourceforge.ganttproject.test.task.calendar	0.0	0.0	53.0	0.0	1.0
net.sourceforge.ganttproject.test.task.dependency	0.0	0.0	193.0	0.0	1.0
net.sourceforge.ganttproject.test.task.event	0.0	0.0	9.0	0.0	1.0
net.sourceforge.ganttproject.test.task.hierarchy	0.0	0.0	89.0	0.0	1.0
net.sourceforge.ganttproject.test.time	0.0	0.0	0.0	1.0	1.0

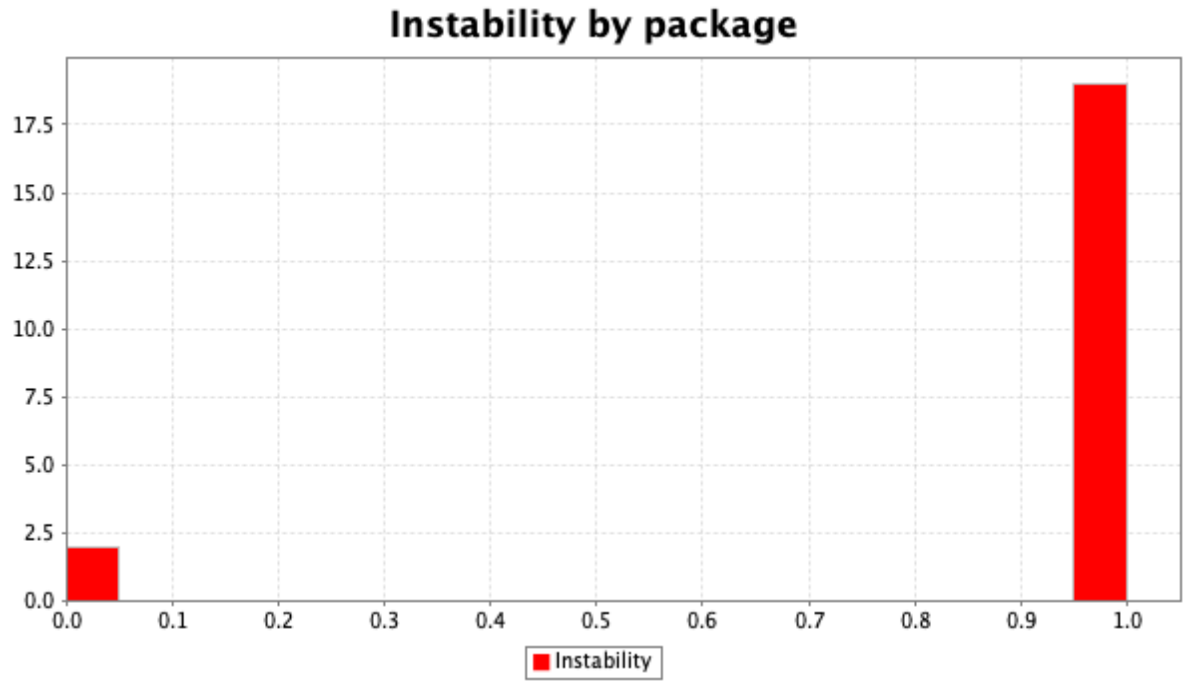
Abstractness: Calculates the Abstractness for each package. Abstractness is defined as the number of abstract classes and interfaces divided by the total number of classes of the package. This metric has a range of [0,1]. A=0 indicates a completely concrete package, A=1 indicates a completely abstract package.

Afferent couplings: Calculates the number of Afferent Couplings for each package. An Afferent Coupling is a reference from a class or interface external to the package to a class or interface internal to the package. That is Afferent Couplings are Arriving references, Efferent Coupling are Exiting or Escaping references. References from test classes and library classes are not included.

Efferent couplings: Calculates the number of Efferent Couplings for each package. An Efferent Coupling is a reference from a class or interface internal to the package to a class or interface external to the package. That is Afferent Couplings are Arriving references, Efferent Coupling are Exiting or Escaping references. References to test classes and library classes are not included.

Distance from the main: Calculates the Distance from the Main Sequence for each package. The Distance from the Main Sequence metric is defined as the absolute value of 1 - Abstractness - Instability (|1 - A - I|). This metric has a range of [0,1], where the closer D is to 0, the better.

Instability: Calculates the Instability of a package. The Instability of a package is defined as the package's Efferent Couplings divided by the sum of the package's Afferent and Efferent Couplings (Ce÷(Ca+Ce)). This metric has a range of [0,1]. I=0 indicates a maximally stable package, I=1 indicates a maximally unstable package.



Almost every package has an instability of almost 1, which indicates that almost every single package is a maximally unstable package. This means that every package has a lot of efferent couplings resulting in a huge danger when changing code. This is a big code smell that we have to have in consideration when developing our new features.

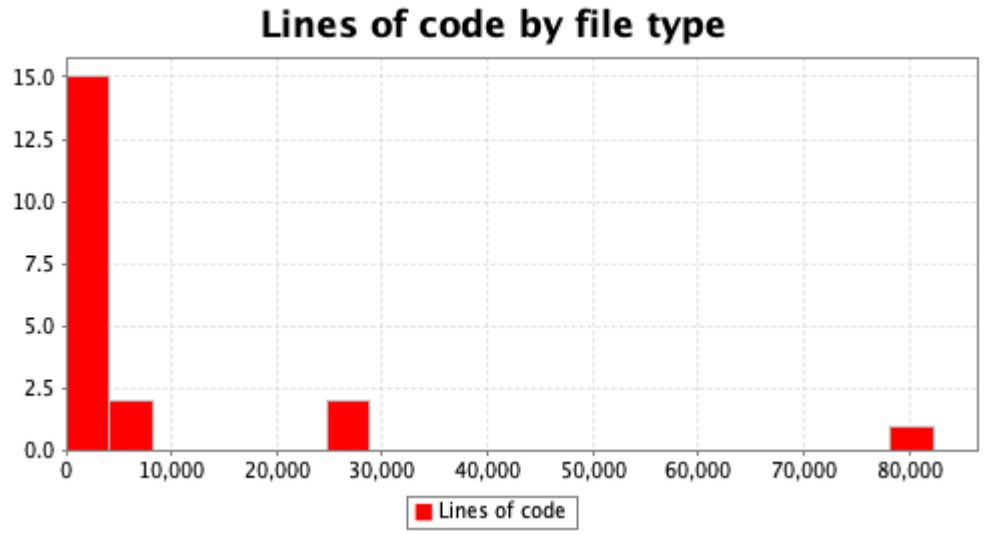
Reviews

Guilherme Franco: I like the fact that we can see the table with every package and their respective score by metric, it really helps to know exactly where the problems are. Other than that I agree with the way everything is explained but I would suggest more data visualization techniques so it's easier to understand each metric.

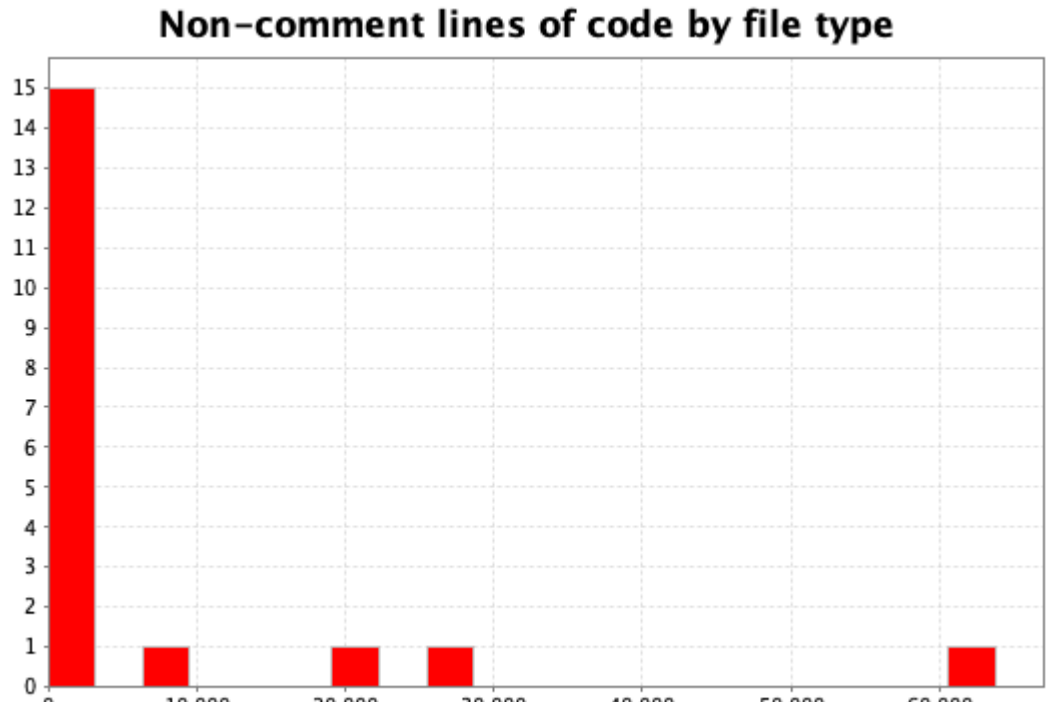
Lines of Code Metrics - Tiago Francisco, nº 60749

It is a quantitative measurement for files that contains code from a computer programming language, in text form. The number of lines indicates the size of a given file and gives some indication of the work involved. It is used to quantify the software complexity.

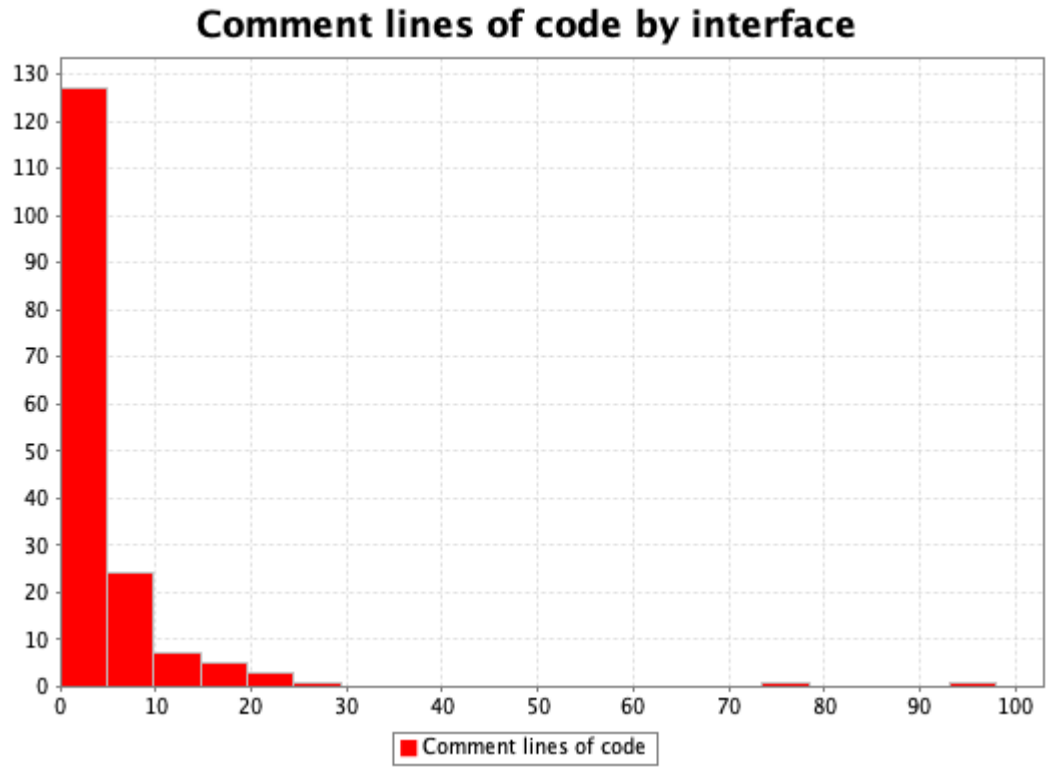
LOC - Calculates the number of lines of code for each file type. Comments are counted, but whitespace is not.



NLOC - Calculates the number of non-comment lines of code for each file type. Comment and empty lines are not counted.



CLOC - Calculates the number of lines of code in each interface which contain comments. Anonymous inner classes are included in their containing class, while named inner classes are evaluated separately. Whitespace lines are not counted.



Comparing the histograms of LOC and NLOC we can see that there are a few differences between them. This means that there is almost no comments in the whole project which makes it challenging to understand some of the implementations.We can confirm that by analyzing the histogram of CLOC where the majority of the interfaces have minus than five lines with comments. That being said, we are in the presence of a code smell.

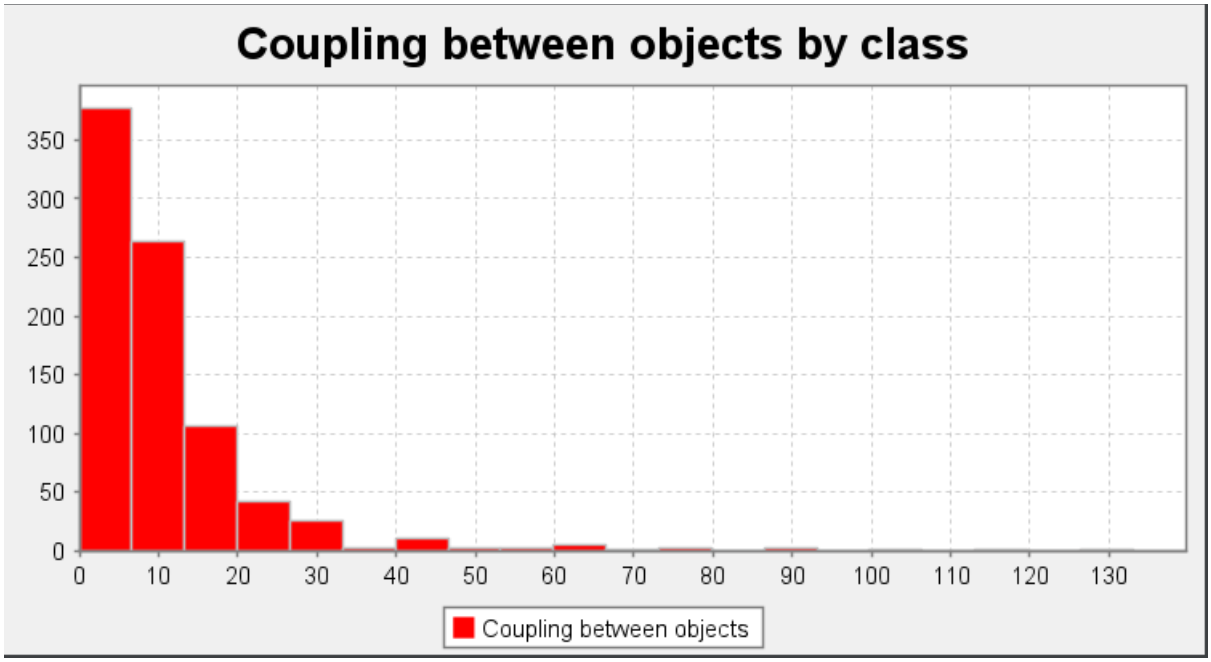
Reviewed by: Luís Abreu: Hi Tiago, you forgot to put the name of the metrics you chose and there is a description of something at the beginning of your metrics that is not identified too. I would also develop a little bit more the conclusion you made based on the values of the graphics of this metric. Other than that, the descriptions of the concepts present in this metric are well described, good work!

Reviewed by Francisco Vale: It is well explained and I liked the way that you analyzed it by concluding one of the many code smells present in this application.

Chidamber-Kemerer metrics - Guilherme Franco, nº 60226:

Coupling between classes: Calculates the number of classes or interfaces which each class is "coupled" with. A class is declared to be coupled with another if it depends on that class or is depended on by that class. Dependencies due to inheritance are not counted.

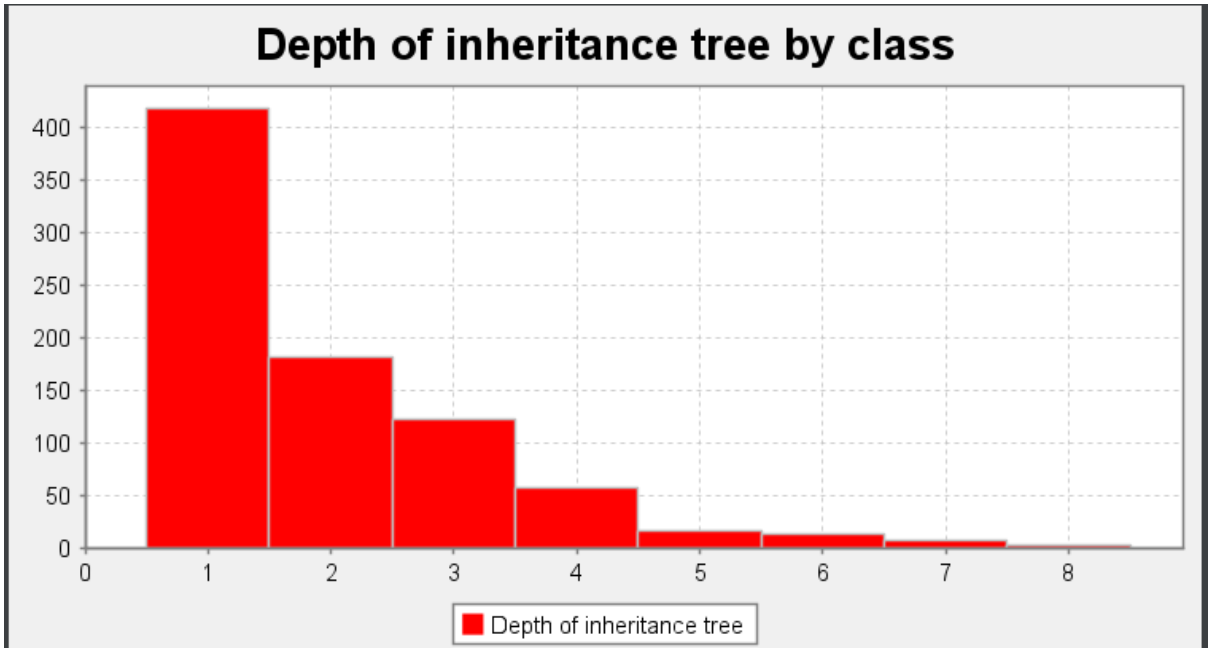
Abbreviation: CBO



Overview: As we can see above there are a couple of classes that have more than 40 dependencies. This shows that the classes depend too much on each other and it's very likely that this means there are Inappropriate Intimacy code smells in these classes!

Depth of inheritance tree: Calculates the depth of the inheritance tree for each class. The depth is calculated as the number of inheritance steps between the class and java.lang.Object.

Abbreviation: DIT

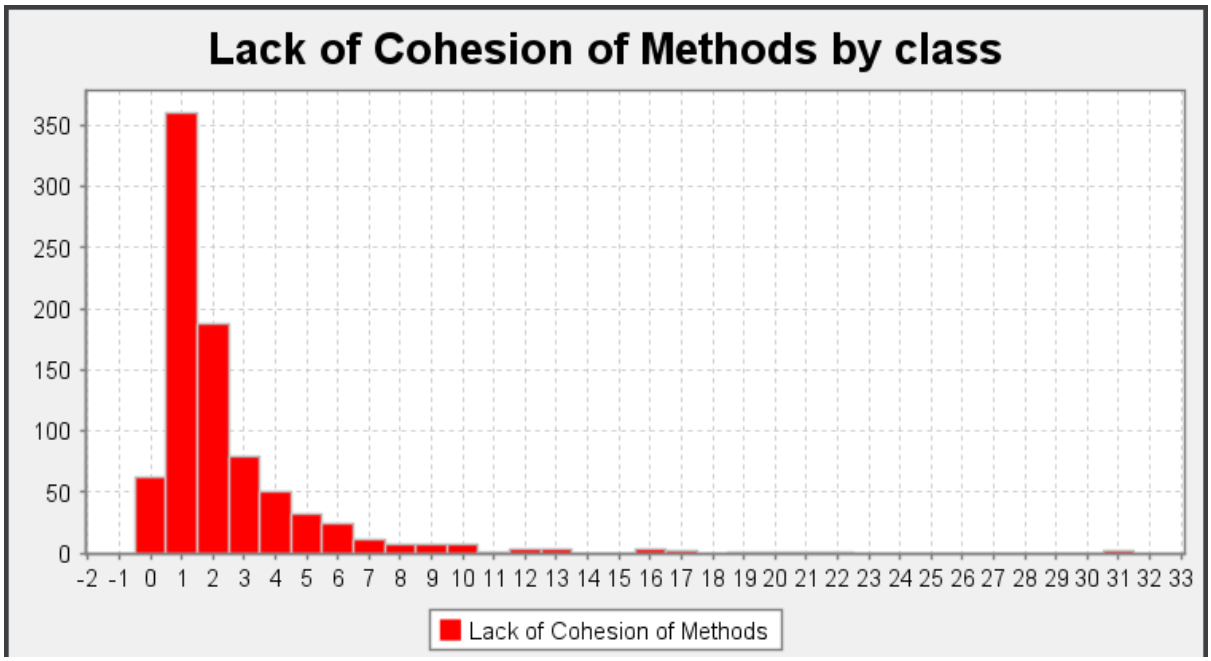


Overview: Other than a few classes that have a high amount of DIT score (very specialized classes), which is not very desirable since it creates confusing code, overall the DIT score of most classes shows that there aren't many troubled areas.

Lack of Cohesion of Methods: Calculates on the degree of cohesiveness of a class. We use a variant of the LCOM metric designed by Hitz and Montazeri, which is more appropriate for Java. The metric says that two methods of a class are related if they share a variable use, or one method calls another. The metric is then the count of the number of components of the method relation graph. A value of 1 indicates a highly cohesive class, which can not easily be

split into smaller classes. Higher values may indicate that the class may be "doing too much", and should be split. Note that constructors, equals(), hashCode(), toString(), clone(), finalize(), readObject(), and writeObject() methods are not considered, as these scaffolding methods often touch all variables in a class, and would thus result in metrics values indicating more cohesiveness than is actually apparent in the design.

Abbreviation: LCOM



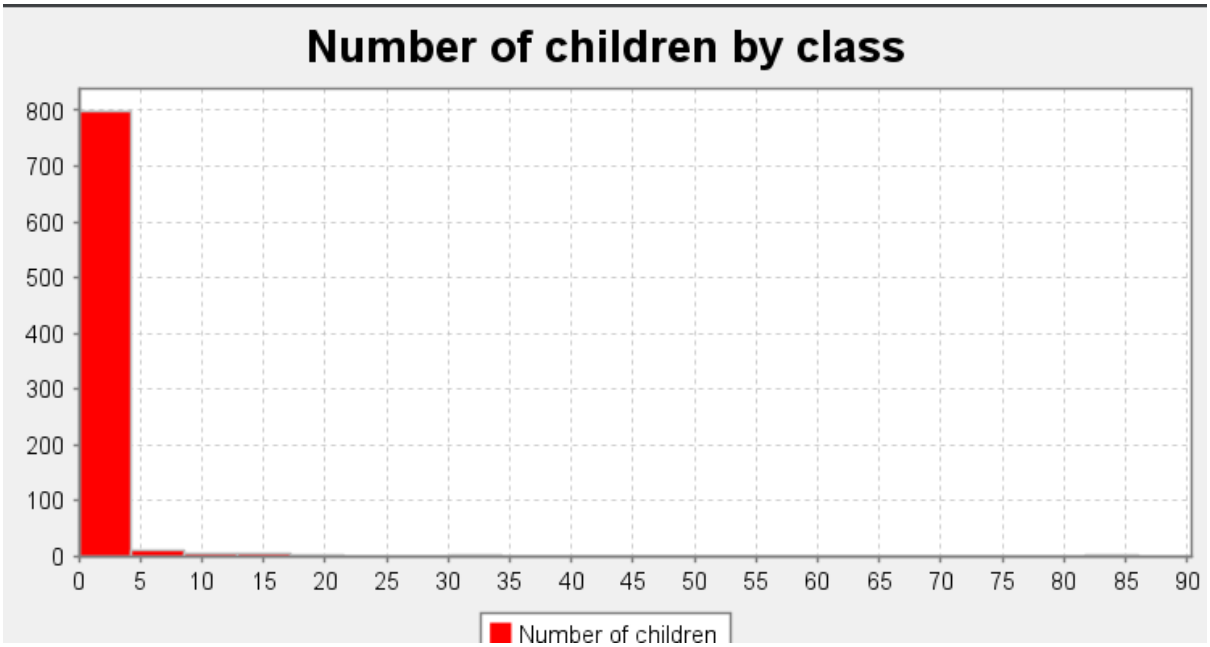
- LCOM=1 indicates a cohesive class, which is the "good" class.
- LCOM>=2 indicates a problem. The class should be split into so many smaller classes.
- LCOM=0 happens when there are no methods in a class. This is also a "bad" class.

As we can see in the histogram, the majority of classes have a LCOM score of 1 which indicates they are cohesive.

On the other hand there are more than 50 classes with a LCOM score of 0, meaning we have over 50 classes where the only method they have implemented is one of the following: constructors, equals(), hashCode(), toString(), clone(), finalize(), readObject(), or writeObject(); This is obviously not the desired behavior of a class.
Finally there are over 200 classes with a LCOM score of >=2. This means that most of those classes could and should be split into smaller classes so as to avoid Large Class code smells and God Object code smells.

Number of Children: Calculates the total number of direct subclasses of each class that occur in the project.

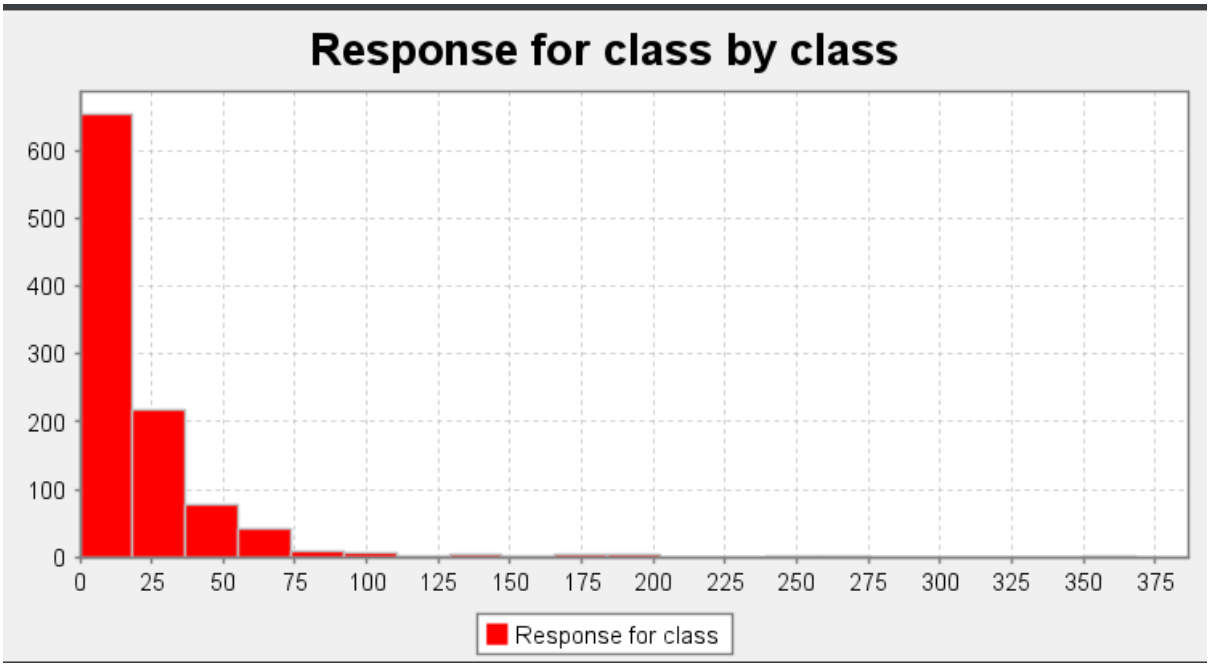
Abbreviation: NOC



Response for class : Calculates the Response For Class value of the class. This is defined as the total number of methods that can potentially be executed in response to a message received by an object of a class. In practice, this is the sum of the number of methods and constructors in the class, plus the number of methods and constructors that the class may directly call. Methods from or called by superclasses are not counted.

> Classes with high Response For Class have a higher complexity, may be less stable, and require higher amounts of integration testing.

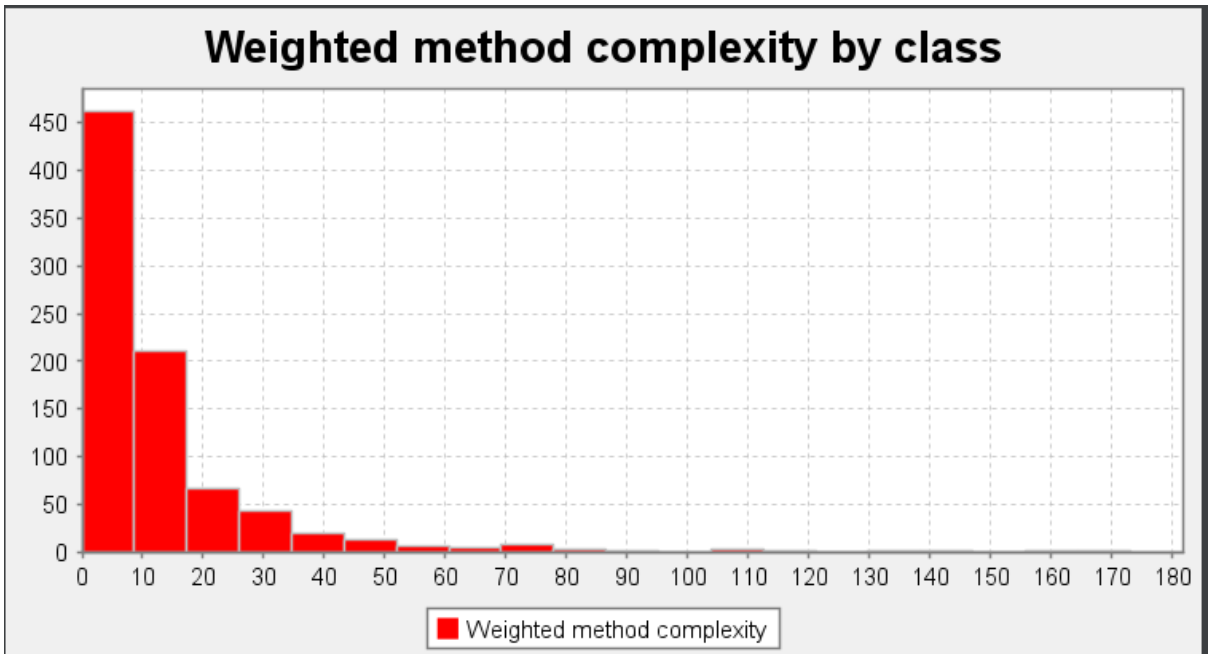
Abbreviation: RFC



Overview: On the histogram above we can see that there are over 100 classes with a RFC score above 50. This isn't desired, not only because, as mentioned above, classes with a high RFC score have a higher complexity but because there is a very high chance of these classes having "law of Demeter"/"message chains" code smells!

Weighted Method Complexity: Calculates the total cyclomatic complexity of the methods in each class.

Abbreviation: WMC



Reviews

Reviewed by Pedro Gouveia - I think that this review of the Chidamber-Kemerer metrics is very well done and extensive, when I started reading this piece I had no idea of what these metrics were or for what they were used and if they were important for our project but after reading this I was more aware of the code smells I might encounter and prepared for them.