

PLD AGILE
Xanome
4IF INSA Lyon 2020

Sommaire

Sommaire	2
Architectural Choices & Used Design Pattern	3
GUI :	3
MVC :	3
Observer :	3
Code	4
Javadoc	4
Package Diagram	5
Class Diagrams	6
Test Coverage	13
Plannings	14
Iteration 1	14
Iteration 2	15
Technical and Human Reviews	16
Kermani Benjamin	16
Versmée Erwan	17
Sierra Camilo	18
Ouvrard Mathieu	19
Galarza Javier	20
Bel Corentin	21
Girard Andrieu	22

Architectural Choices & Used Design Pattern

GUI

The graphical user interface was completely built on Swing. The process of selection was really hard, we considered many possibilities especially for drawing the map. First of all, we made a Swing interface. Then, when the time came to draw the map we had options : Draw it with Graphics or using an external library. First we tried using Google Maps, but this was only possible through javascript which wasn't compatible with our MVC design. Then, we had to decide between just drawing it or using mapjfx which is an external library designed to handle maps on JavaFX.

At that moment we took the decision as a team to just draw it using Swing, why? Because no one knew how to use JavaFX and because the skeleton of the interface was already built using Swing, these two reasons coupled with the fact that we were really short on time were determinant.

MVC

Our whole application is based upon the Model-View-Controller design pattern that is widely known. We made this choice for several reasons. First its modularity allowed us to better separate the work in terms of development. One team focused mostly on the model and algorithms while the other concentrated its efforts on the graphical interface, which is great for the AGILE framework. In addition to that, MVC is great for reusability: if we ever decided to make a different GUI we could use most of what we already have coded in the model and controller. But not only could we change the UI, we could make several and have them coexist.

Observer

In addition to MVC we also had the opportunity to use the observer design pattern. This was necessary to implement one of the key features of our second sprint: undo and redo. This design pattern is what allowed us to track the changes on our tour but most importantly trigger them to redo or undo actions.

Code

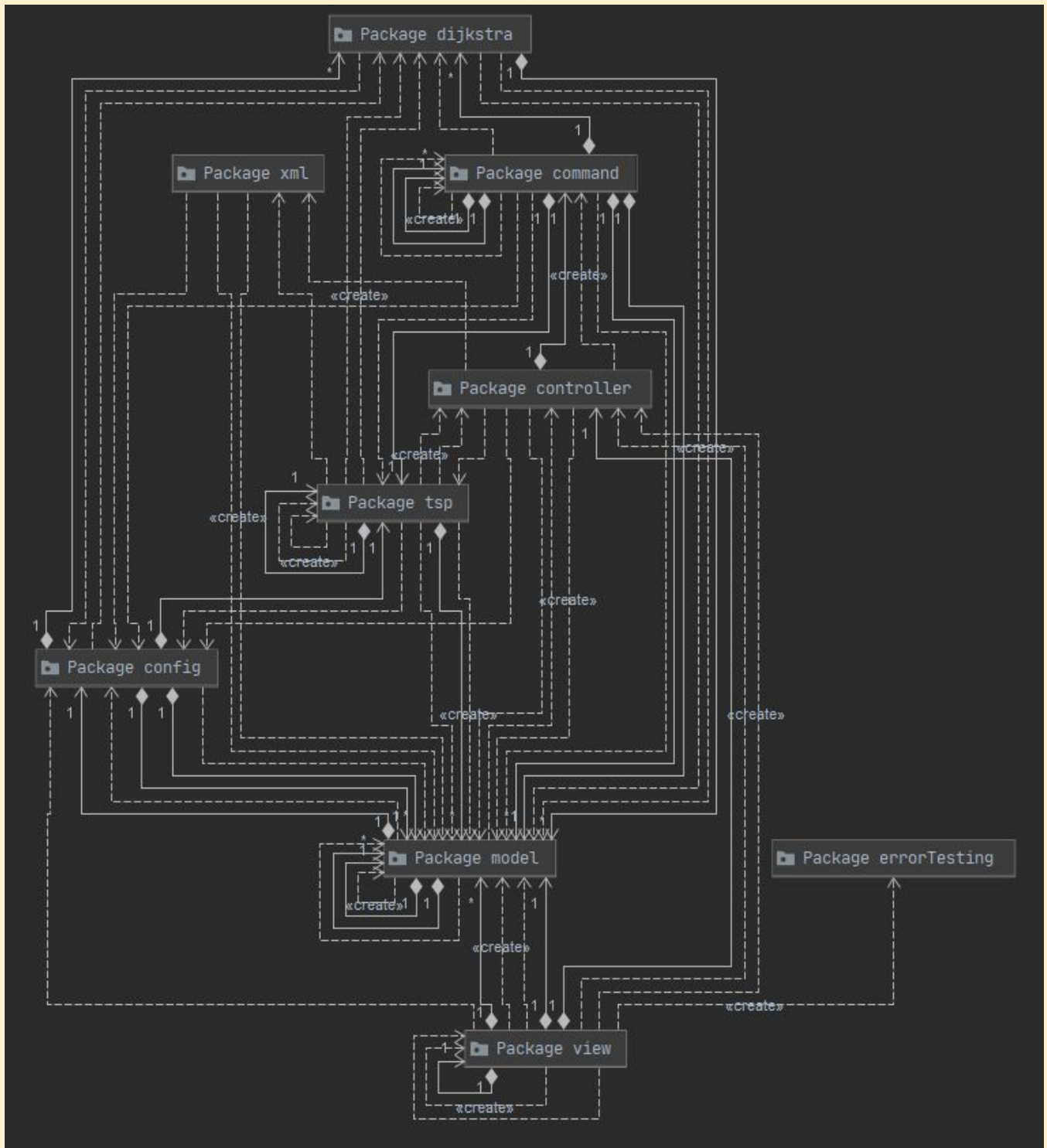
You can find all of our code on the PLD's github :

<https://github.com/camilosierrar/Agile.git>

Javadoc

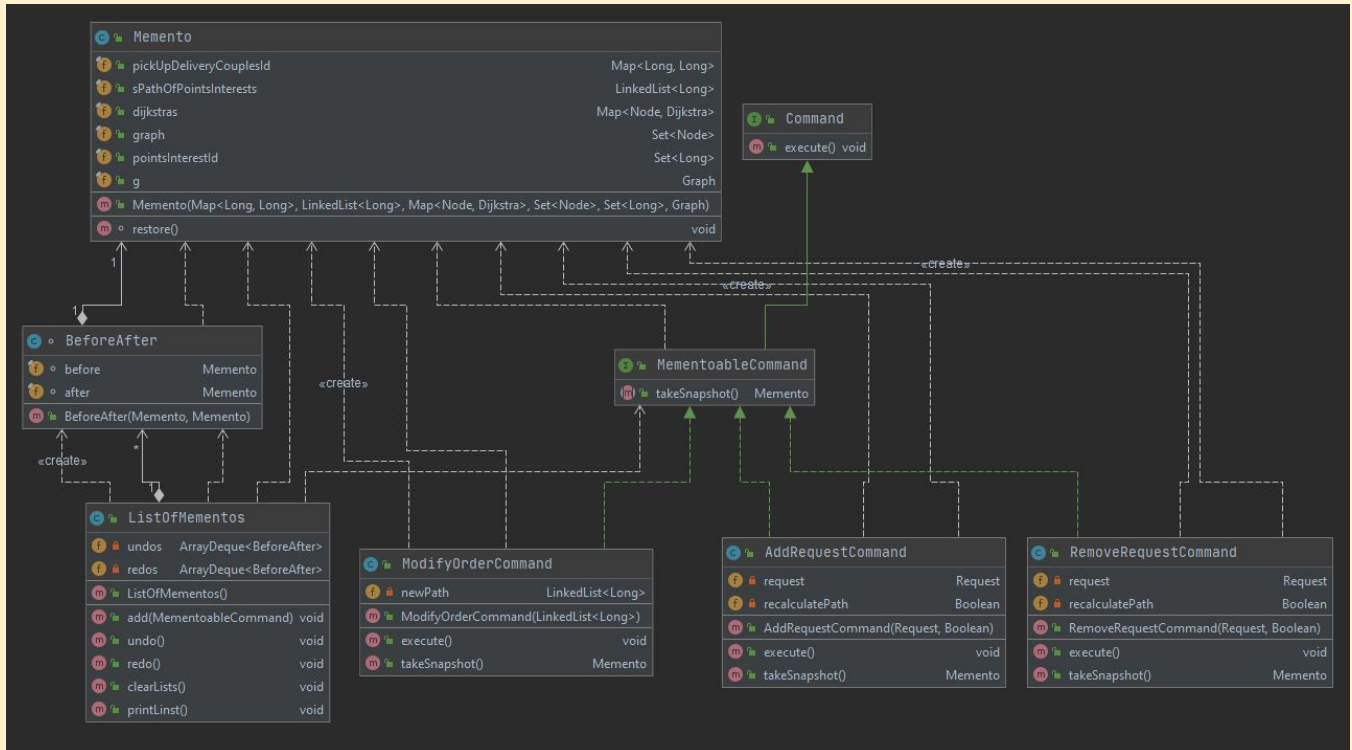
You can find the javadoc by opening the index.html file in javadoc directory

Package Diagram



General package diagram

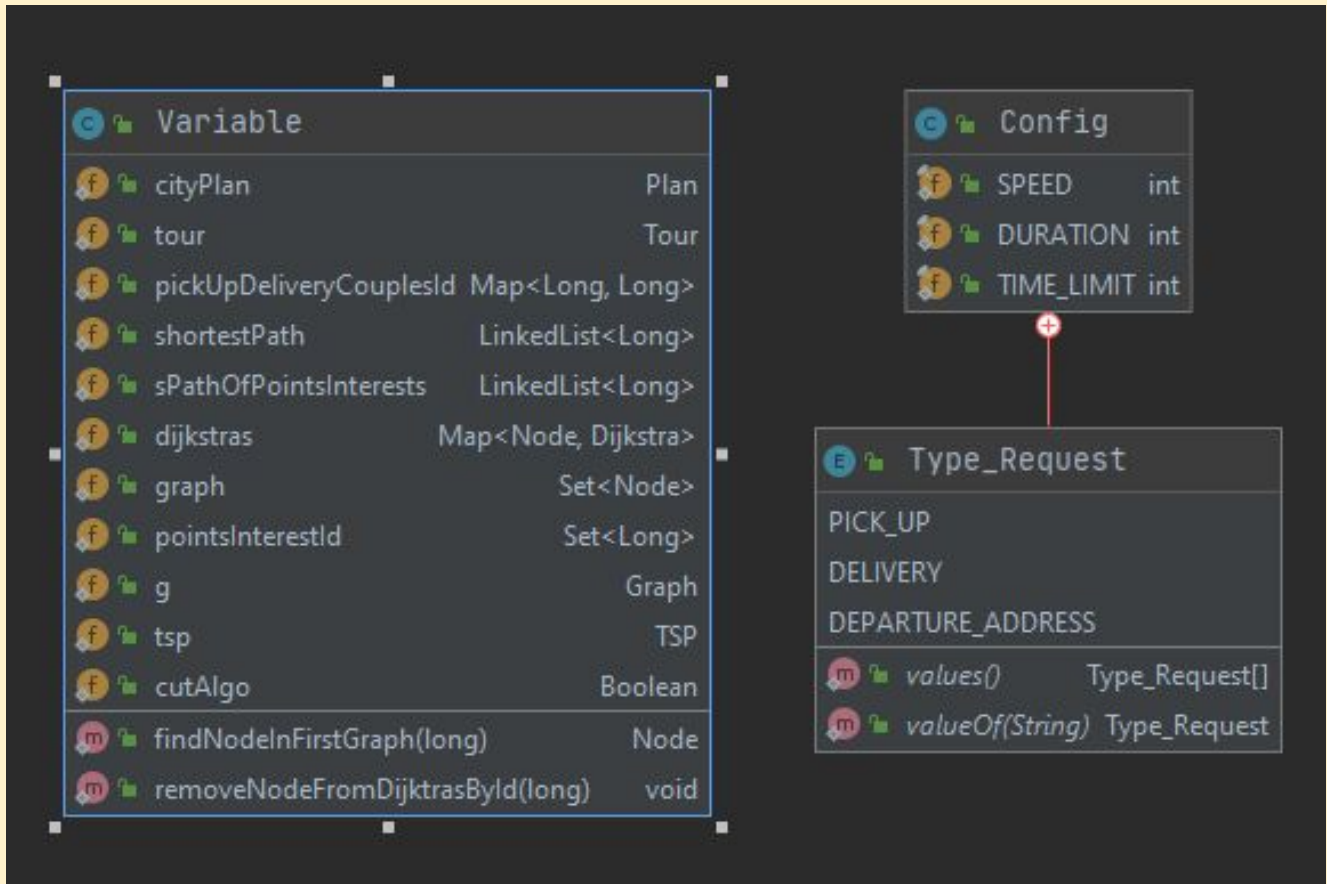
Class Diagrams



Package command

Package command

This package is new on the second sprint and is the one that allowed us to implement the undo/redo.

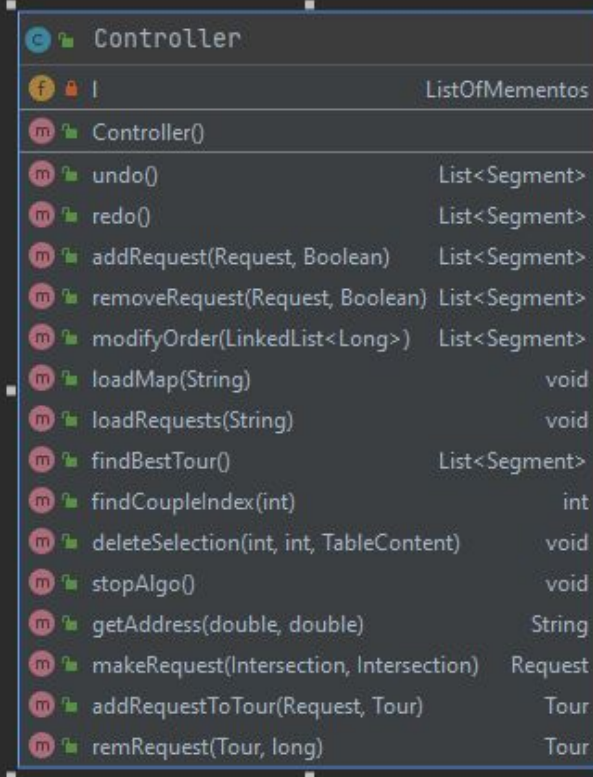


Package config

Package config

This package contains classes with some global variables to be used wherever and whenever we need.

Those classes have been added in the second sprint in order to facilitate the use of static informations.



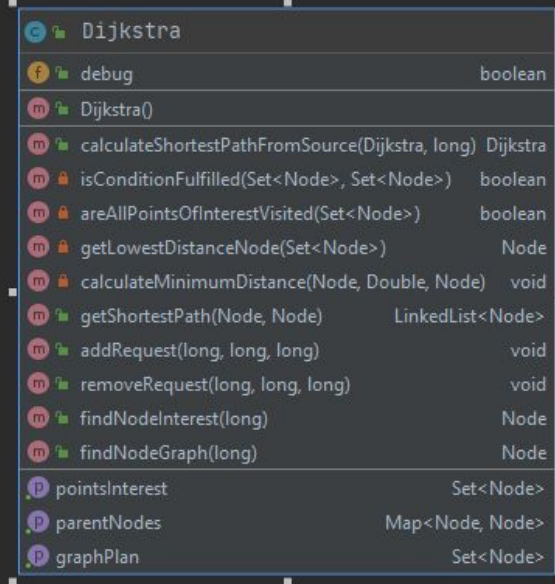
Controller	
↳	ListOfMementos
Controller()	
undo()	List<Segment>
redo()	List<Segment>
addRequest(Request, Boolean)	List<Segment>
removeRequest(Request, Boolean)	List<Segment>
modifyOrder(LinkedList<Long>)	List<Segment>
loadMap(String)	void
loadRequests(String)	void
findBestTour()	List<Segment>
findCoupleIndex(int)	int
deleteSelection(int, int, TableContent)	void
stopAlgo()	void
getAddress(double, double)	String
makeRequest(Intersection, Intersection)	Request
addRequestToTour(Request, Tour)	Tour
remRequest(Tour, long)	Tour

Package controller

Package controller

This package contains

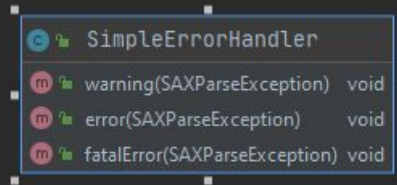
This class has been completed in order to add some functionalities during the second sprint.



Package dijkstra

Package dijkstra

This package contains the dijkstra algorithm from a given point.
This class hadn't been changed between first and second sprint.



Package errorTesting

Package errorTesting

This package contains a handling error class.

This class has been added between during the second sprint.



Package model

Package model

This package corresponds to the model of the MVC design pattern. Here we have the classes that store data.

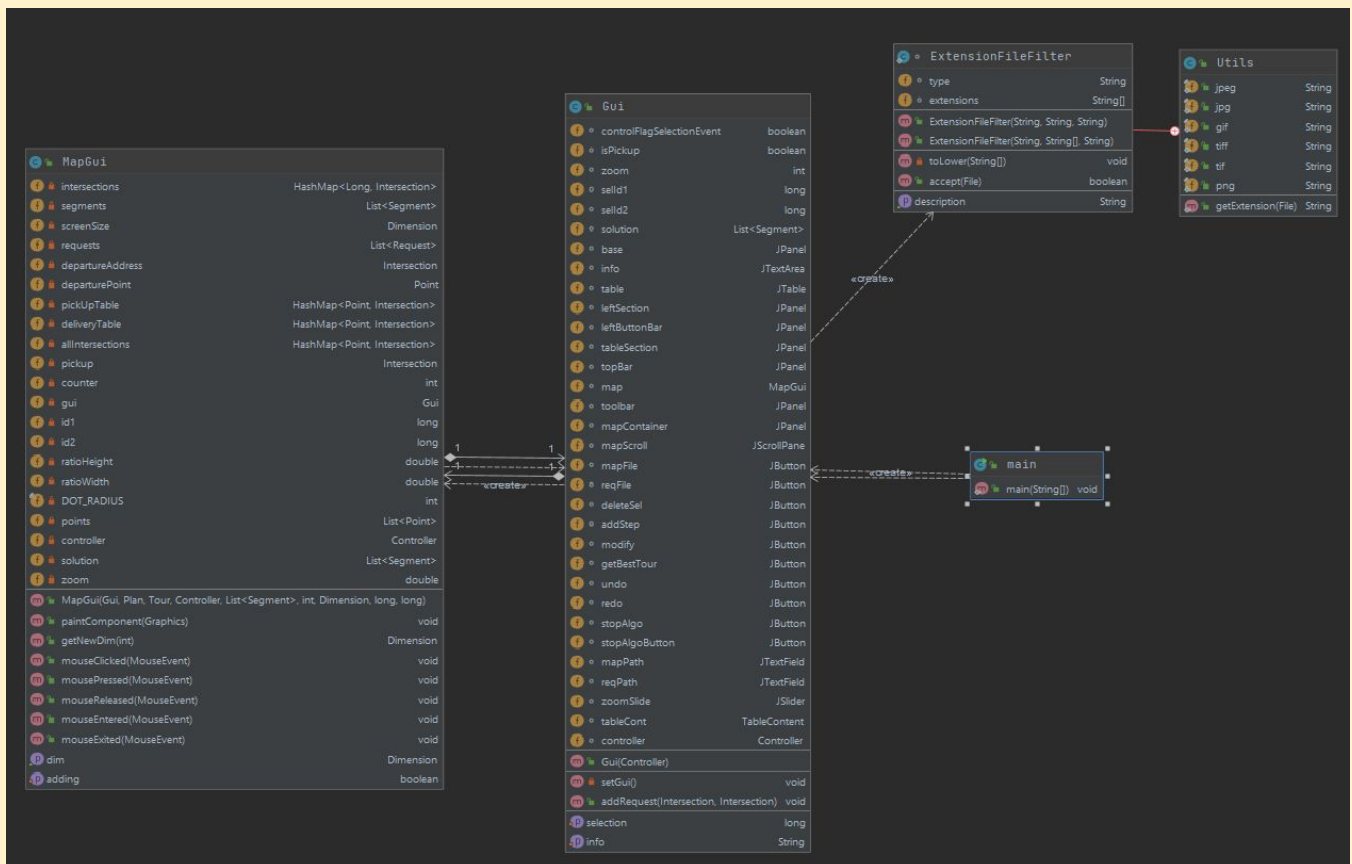


Package tsp

Package tsp

This package contains branch and bounds and iterator for travelling salesman problem solving in a short time.

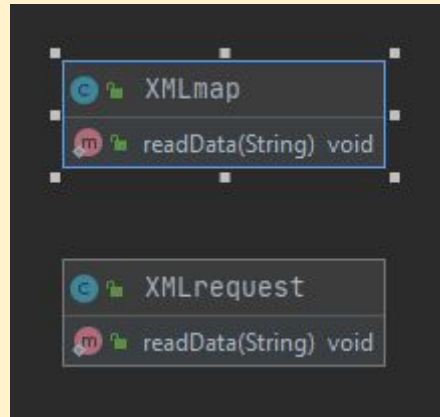
The TSPEnhanced has been added in the second sprint to implement branch and bound that shortens solving time.



Package view

Package View

This package corresponds to the View of the MVC design pattern. Here we have all the classes that strictly concern the User Interface.



Package xml

Package xml

This package contains classes that read xml files and store them.
Those classes have not been modified between first and second sprint

Test Coverage

Overall Coverage Summary

Package	Class, %	Method, %	Line, %
all classes	72.2% (26/ 36)	46.8% (101/ 216)	42.9% (639/ 1491)

Coverage Breakdown

<u>Package</u>	<u>Class, %</u>	<u>Method, %</u>	<u>Line, %</u>
command	85.7% (6/ 7)	89.5% (17/ 19)	92.3% (120/ 130)
config	66.7% (2/ 3)	71.4% (5/ 7)	83.3% (20/ 24)
controller	100% (1/ 1)	37.5% (6/ 16)	26% (13/ 50)
dijkstra	100% (1/ 1)	85.7% (12/ 14)	96% (95/ 99)
errorTesting	0% (0/ 1)	0% (0/ 4)	0% (0/ 7)
model	75% (6/ 8)	38.9% (28/ 72)	30.7% (61/ 199)
tsp	100% (8/ 8)	68.9% (31/ 45)	67.8% (253/ 373)
view	0% (0/ 5)	0% (0/ 35)	0% (0/ 520)
xml	100% (2/ 2)	50% (2/ 4)	86.5% (77/ 89)

The image on top shows the general overview of our tests coverage. More information on the [index.html](#) file in the Test Coverage directory.

We focused our tests on the algorithm which is where we do most of the calculations. No automatic tests were written for the view because it doesn't do many calculations. Our tests for the View were mainly visual and manually made regularly.

Plannings

Iteration 1

Tasks	ALL	Andrieu	Benjamin	Camilo	Corentin	Erwan	Javi	Mathieu	Time Spent (h)
Documentation	X								3
Week 1 & 2									
Writing the Model									
Use-Case Diagram	X								1.25
UML Diagram	X								1.25
Request			X						0.5
Intersection						X			0.5
Map								X	0.5
Segment		X							0.5
XMLmap							X		0.5
XMLrequest							X		0.5
Config			X						0.5
Tour					X				0.5
Start of algorithms and GUI									
Gui				X	X			X	2
Algo			X						2
Week 3									
View - Map rendering - Intersections		X		X	X			X	4
Controller							X		2
Model - Algorithm			X		X	X			4
Pre-Week 4									
View - Map rendering - Segments					X			X	1
Algos			X			X			3
Google API loading							X		4
Week 4									
UI - Adapt map rendering to Google API				X	X		X	X	1
Complete Graph			X			X			4
Finishing Swing Solution				X					2.5
Js Solution Exploring		X							2
Jfx exploring							X		2
Post-Week 4									
GUI team meeting to decide best Solution		X		X			X	X	0.25
Add Mouse Listener to Map on GUI							X		1.5
Finishing Algorithms			X		X	X			2
Finishing spring 1									
Putting it all together	X								3

Iteration 2

Tasks	ALL	Andrieu	Benjamin	Camilo	Corentin	Erwan	Javi	Mathieu	Time Spent (h)
Week 1									
Zoom				X					6
Addresses API				X			X		4
Improving Algorithm Performance			X			X			8
Files System					X			X	6
List View of Tour		X						X	8
Week 2									
Table and Map Integration		X		X				X	8
Map Selection		X		X				X	4
Add Request			X	X		X	X		8
Remove Request		X	X	X		X	X		8
Undo		X	X	X		X	X		4
Redo		X	X	X		X	X		4
Error Management					X				4
Unit Tests			X						2
Improving readability of Code (comments)					X				4

Technical and Human Reviews

Kermani Benjamin

Technical point of view

I was assigned to develop the back-end of the application, including finding algorithms to compute the best tour and overcome time complexity issues.

It was interesting as we had to take business perspectives into account. From an algorithmic point of view, computing the best tour for 15 or plus requests on a detailed map (such as Lyon-Est) is time intensive. But from the business context we found answers as we know that such a scenario is unlikely to happen. Moreover waiting 10-15 sec to generate the best tour for a reasonable amount of requests is acceptable.

By deducting these requirements, we had to adapt our solution to best suit our needs. This project reassured me on using and completing an already existing structure, which I found as being one of the hardest tasks in programming. Moreover it confirmed that I was able to implement not so trivial algorithmic solutions (branch and bound).

This “PLD” also introduced us to implementing new design patterns (memento, state) which is a major aspect of programming if you want to develop quality code. Also I wrote the unit tests, which was not a new task to me..

Personal point of view

It was quite hard to work in a group of seven students as a minority of the group was unaware of git functionalities (branches, merge requests).

We were 2 assigned to the development of the back-end of the application and personally I struggled finding tasks to work on my own when developing the algorithms. Indeed, the structure changing constantly due to Dijkstra’s implementation and state pattern development, it was hard to work independently of my teammate.

However it was really pleasant to work on a real business problem and to concretely develop a working application that clients could use and pay for. Our application was not completely finished during the presentation (all use cases were implemented but some valuable business functionalities were not implemented) which was quite frustrating for the group and myself.

In conclusion, what I retain from this project for the future is to communicate more often and more efficiently. But also to consider business perspectives more seriously to develop, in the end, a better product.

Versmée Erwan

Technical point of view

This PLD allowed me to explore many new points of programming. First discovery, working in a large group of seven people may be complicated, but using AGILE method, making sprints at the beginning of each iteration, separating the work into logical tasks and using Git to merge our work correctly allowed us to overcome this issue. I think working together on the PLD succeeded to improve competencies of all the members of the group to a better understanding and use of AGILE method and Git. Secondly, since one of my jobs was to take care of Dijkstra and TSP algorithms, I believe that I have deeply increased my skills in this domain. Algorithms in code might be counterintuitive or difficult to assimilate at the beginning, but making research and coding one myself enhanced my understanding. Using a design pattern to implement the undo/redo functionality was also a first experience ; it opened us for the future to a new side of programming.

The size and complexity of the project allowed us to realize the importance of having a well-structured architecture and code. As we divided work into small tasks and assigned each task to a pair of people, when putting results in common, it was essential to have a clear architecture. Thus, we were able to get a better understanding of the Model-View-Controller architecture and its advantages

Personal point of view

From a personal point of view, I found this project very interesting, I believe I would like to work in the same area of this project. It was pleasant putting our knowledge and skills to the service of a real demand in today's world (ie computing of shortest path) inside a simulated situation. However, the fact that we couldn't provide a completely finished project at the end of the project time was a real frustration. We learned lessons from this ending : even though every functionality was correctly implemented and passing test, the fact that we didn't synchronise often enough our work with the one of the front-end team caused some issues in code that caused delay.

On another hand, I really enjoyed working on my part. The skills I learned and developed during these sessions will allow me for the future to go deeper in the research and understanding of algorithms and design patterns.

Sierra Camilo

Technical point of view

During this project I was on the GUI team, which is great since I really like this part of coding, even if I'd rather just do the model. It was a great opportunity to grow in this area, but I feel like I missed it. At first we tried to use an external library for our map, but the way we did it wasn't compatible with the goal of the project so we went back to what we know best: Basic Swing. We made this choice because, with time running out, we had our back against the wall. It's a pity because I ended up using the knowledge that I already had without learning as much as I would've if we had explored a new framework. But also because Swing is quite outdated and I just don't find it practical. Now with more time on my hands I wish we would have chosen to use JavaFX for instance (I'm also open for suggestions). Especially since that would've allowed us to use an external library for maps. Don't get me wrong, I was the one suggesting Swing from the beginning, because at the moment I didn't feel like the GUI should be our main concern. On the other hand, I did get the opportunity to explore new things such as making HTTP requests from Java which turned out to be way more complex than I thought. Even if that was a mistake, because it made our application much slower, I'm happy that I did it because I learned a lot in the process.

Personal point of view

In a more personal point of view I really liked this project. First, just the fact that we did an application that seemed useful was great. Then, the fact that we tried using the Agile framework was also a plus, just because of the fact that it allowed us to have a working application in just two weeks. But also because it made us focus more on communication, we had meetings and subteams. With that being said, to be honest we lacked communication during the whole project (as a team but also with the client) but most importantly on the second sprint, which is my fault as Scrum Manager. This really took a toll on our end product that was unfinished. Finally, what I loved about the project was that when the second sprint was finished, everyone showed what they had done to the class. It was one of the most frustrating experiences I've had. Just seeing what a good work everyone did got me ashamed, I wanted to just leave the 'room'. I really needed that, I haven't been this frustrated in years and I really hope that this experience will drive me moving forward.

To conclude I would like to thank our teacher that was always really present and trying (and succeeding) to be helpful and comprehensive. But also my team, because even if our application didn't end up being great, it was usable and we still put in the work on an always positive, fun and relaxed environment which isn't always the case.

Ouvrard Mathieu

Technical point of view

From a technical view the AGILE project was an occasion to grow up my skills in programming and teamwork. As the conception of application in java is not something I was really experienced with, i learned a lot of things. Furthermore the fact that we had to do it in an AGILE way taught me a lot on how an application development can be managed. I also discovered a new IDE that I never used and discovered a lot of possibilities that I didn't expect. Also the UI programming was something new to my group so we had difficulties to know how to begin it and we lost time by choosing a bad solution and working with Swing. But we realized that it was too late and we had to continue with it. But we managed to and I'm very proud of my team that didn't give up and surmounted the difficulties to achieve what we had in mind. The last technical thing that was important was GitHub, i already used it but never in that way because it was integrated to the IDE and we were numerous to work on. So I've discovered a new way to manage a Git repository that has revealed all the potential of GitHub and how powerful it is.

Personal point of view

It was the most complex project I've been working on, complex because we were seven on the same project and the fact that we should manage the git while programming. And that's the main point of this project: work as a team on the same project can be hard. But we managed to communicate as much as we could, at the beginning of each meeting we were talking all together about the project. Sharing the advancement each group made and planning on the next things to do. The fact that it was in a distant mode was not an easy thing too. But on the other hand we learned a lot with this project, on several things: programming, team working and organisation. Even if the subject is not something that I'm passionate for, it was really interesting to work on it and pleasant in some ways.

Galarza Javier

Technical point of view

In this project I worked almost all the time on the UI and the model. At the beginning we tried to implement the UI using the Google Maps API so we could print out a real map that looked really good.

But this API didn't fit our expectations for the project. This caused us a delay for the first sprint, as we had to reimplement the UI using Java Swing.

Although I already knew some of Swing basics I could learn a lot from this project, like how to draw a map from raw data and how to manage all kind of events that came from the UI.

Also using the MVC design pattern was familiar to me, but I think I got more sharp at it due to this project which was large.

I think that if we have made the right decisions from the beginning (like not using the maps api) we could have explored other more interesting options for the UI, which could have been new for us and from which we could have learned many things.

Personal point of view

It has been a challenging project in terms of teamwork, as having to work remotely makes communication a bit more difficult sometimes. However, this has been quite good considering the circumstances.

Also it was a bit more difficult for me to communicate, as an exchange student who doesn't speak really good french, but my teammates did a great job and they really helped and made me feel involved in the project despite the language lack.

As a person who likes to code I really enjoyed this project as it was something really practical and near to some real applications, and I think that the skills and abilities we acquired in this project will be applicable to other projects in the future.

Bel Corentin

Technical point of view

From a purely technical point of view, the PLD AGILE has been a really upbringing experience, allowing me to discover a different approach on programming and team managing. We also had the opportunity to discover new programming concepts that we never had the chance to implement on previous projects, such as the undo/redo or the controller. It was also using a new IDE that had a lot of useful functions and possibilities allowing us to spend less time on some parts.

On the other hand, it was for me one of the hardest projects I've ever done, there were a lot of very different functions we were asked to implement, from UI to complex algorithms. The fact that it was purely online didn't help since I usually try to ask a lot of questions in person to understand the reach and the point of the project, which was in my opinion a bit harder online. In fact, using a new IDE also has its difficulties, especially when it is a complex and high level one.

Personal point of view

On a more personal note, I enjoyed working for this project, even if I don't want to work as a developer nor in object oriented programming. We managed to ask each person of the group what they wanted to work on, and I had the chance to work mostly on UI, which I like more than the rest so I could enjoy my time working on this project. The fact that it was online wasn't that much of an issue in my opinion, even if there were some points that would have been easier offline, as I pointed out before in the technical point of view part. On the one hand, it was definitely disappointing not being able to deliver a final product that was totally finished, with a clean and intuitive interface, like the other groups did. Our decisions (thus mine as part of the group) were probably not the best, and that gives me a lot to reflect onto. On the other hand, we did learn a lot of new things, and even if the final product wasn't as perfect as initially planned, I think that we all could learn and focus on what was important to us, thus developing new skills and abilities. The way of working we learnt will definitely be very useful for all the projects we'll encounter next, even if they aren't really close to the programming we did.

Girard Andrieu

Technical point of view

This project has been a real challenge for me from a technical point of view. Indeed, I had to quickly learn many skills and technologies that I wasn't familiar with. The first challenge was to get familiar with the UI of a java project. I am usually a very efficient UI coder on Web Application projects and I hadn't yet had the opportunity to write a user interface on any other type of application. I was a bit lost at the beginning and so was the rest of the group. Indeed we started to work with Swing that may have been a little less efficient and old school than could have been JavaFX. We also tried to use Maps API which wasn't appropriate to the MVC pattern. Once we became aware of the not very optimal choice we made it was too late, we had to continue with Swing. This wasn't unsurmountable though and we can be proud of the endeavours we made in order to catch up.

This project also allowed me to put into practice my knowledge about the MVC pattern and helped me clarify its purpose.

I have also been able to improve my knowledge about event handling.

But what for me is the most important step forward is the use of Git. Indeed I had never been well confronted to git and all its particularities. It took me a lot of time to get familiar with github and git command but I am able now to manage different branches, to merge them, pull some files, push some others etc..

Personal point of view

It wasn't easy at first to work in a group. My lack of skill about git, made it really hard to make something valuable at the beginning. However we managed to organise the work in order to make it easier to produce something. The first AGILE step has been to define who would work on the algorithm and who would work on the interface. That being done we have been able to work by pair increasing our efficiency. We had then a really close communication with our pair and we tried to have some time each session to share with other pairs about the progress. Maybe we could have been more precise when reporting the progression to the group because sometimes there were little incomprehensions that slowed down the whole project.

That being said, I really enjoyed working in group. This is not only about giving time for the project but also about giving your time for all the group members. Moreover the fact that the subject was real and current demand (especially in this pandemic context) was a real motivation.