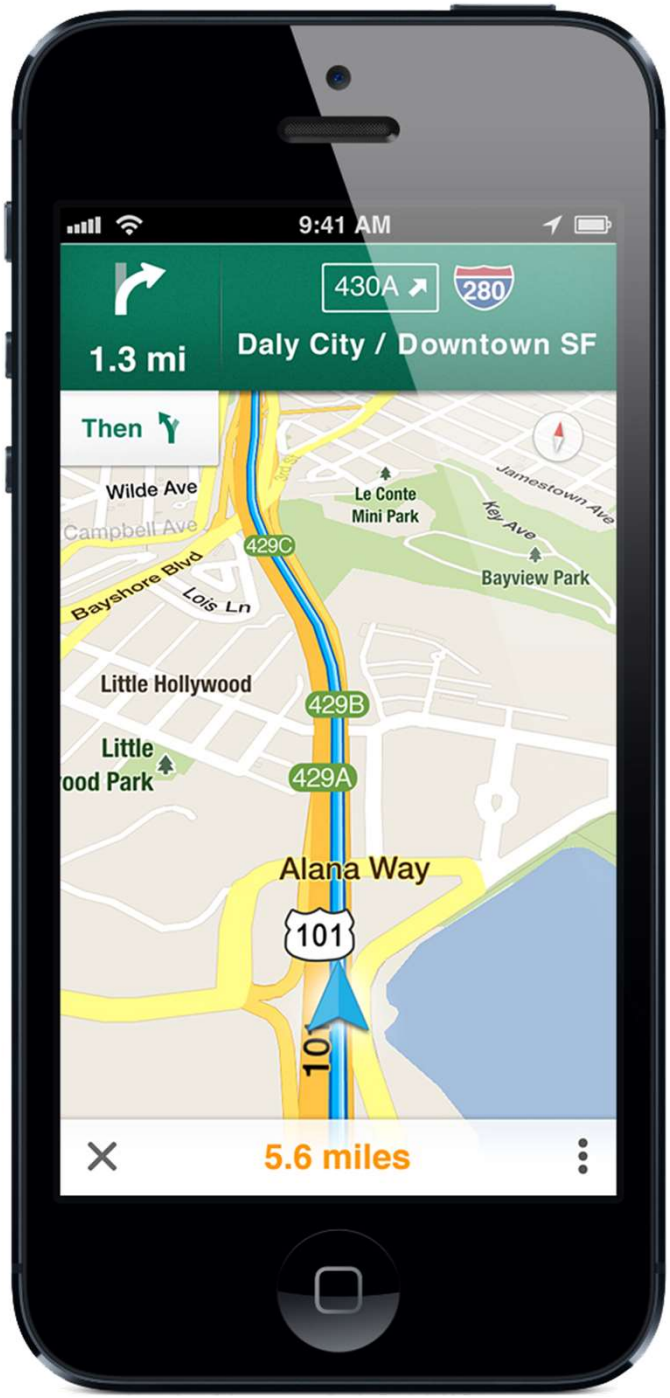


PROJECT 1:

Navigator

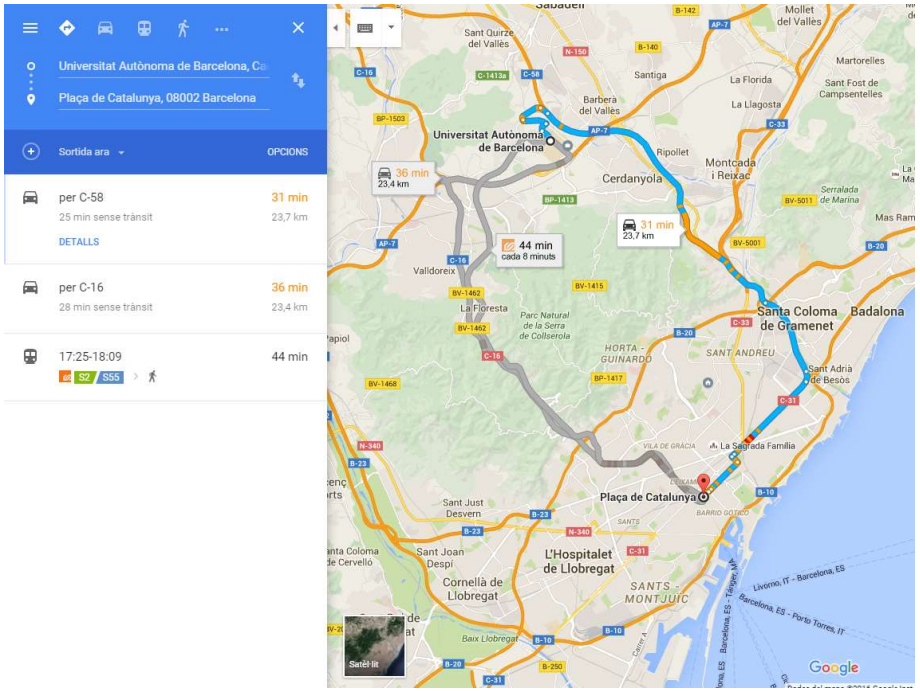
Artificial Intelligence

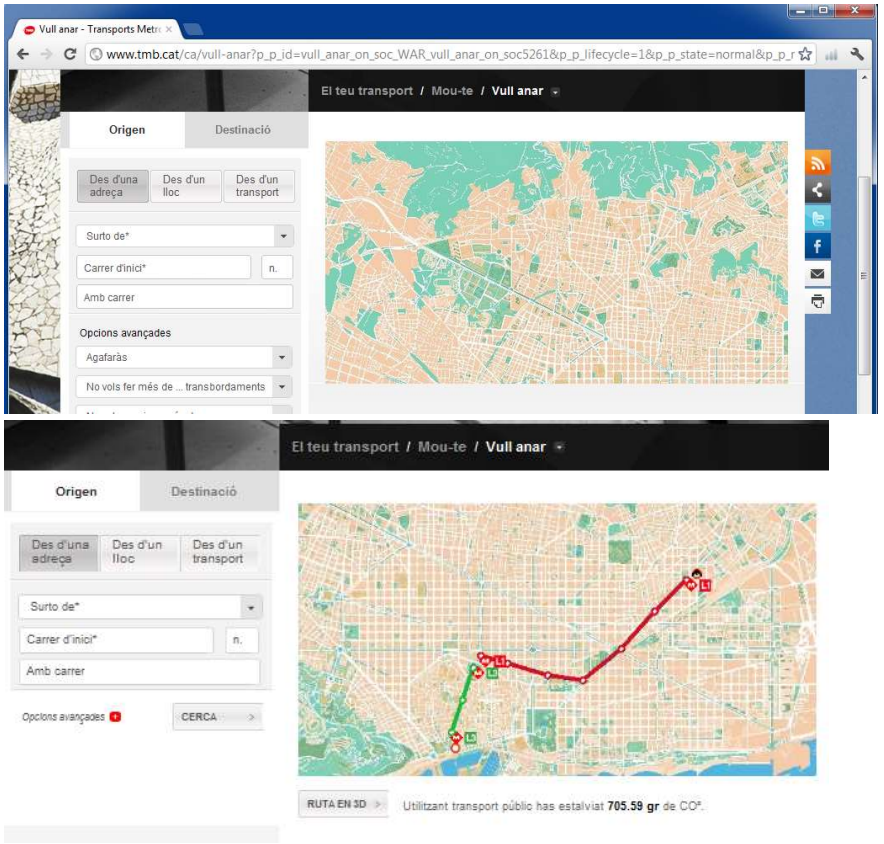
2023-2024

Universitat Autònoma de Barcelona

Project 1

Goal: To make a Navigation application, where the user give origin and destination and selects the preference criterion for the path search strategy.





Project 1

It can be very complex!!! ⟶ **We will simplify it**

Simplifications:

- We only consider **Metro Maps**
- Origin and destination will be given by the names of the stations or in **cartesian coordinates**, no use of street names and house numbers.
- The path between origins and destinations are provided in cartesian coordinates, and the distance between user and metro stations will be assumed as a **straight line**.
- The **preference criteria** can be the following, considered separately:
  - Time, this is to arrive as soon as possible (Minimum time)
  - Distance, ensure not to be doing unnecessary ways (Minimum distance)
  - Line-Changes, we do not want to move around a lot. (Minimum number of line changes)
  - Other criteria ...

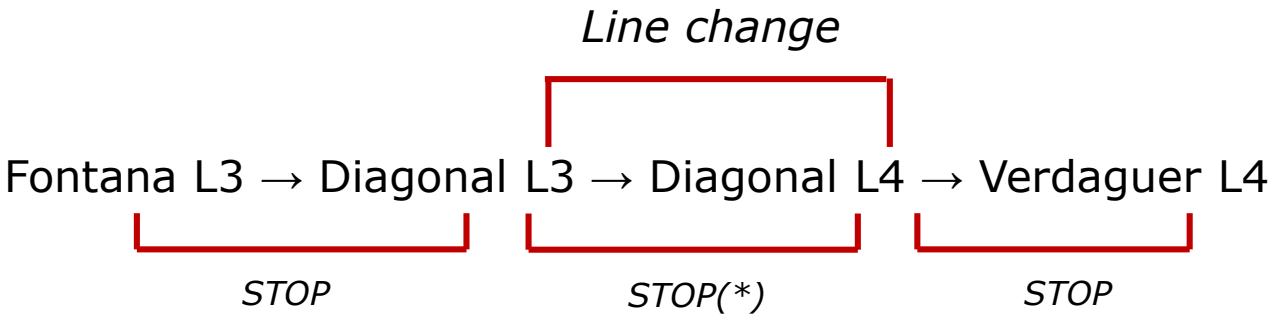
Project 1

Note: The concept of **STOP** can be ambiguous!!!, we define it like this:

Definition: A stop is **a trip between two stations or a change of line**.

Example: Let us assume the route

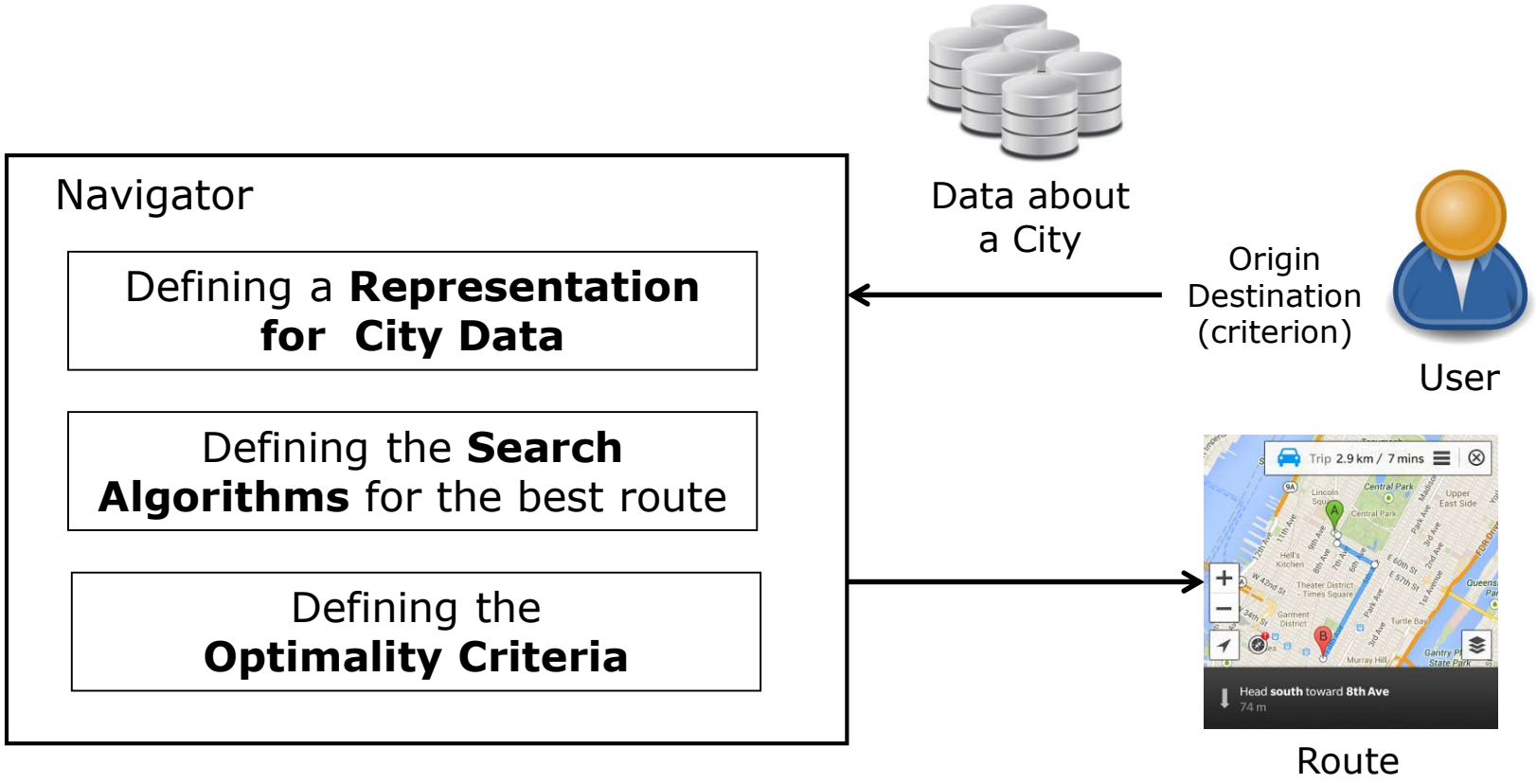
Fontana L3 → Diagonal L3 → Diagonal L4 → Verdaguer L4



(*) To unify a change of line is considered as a STOP

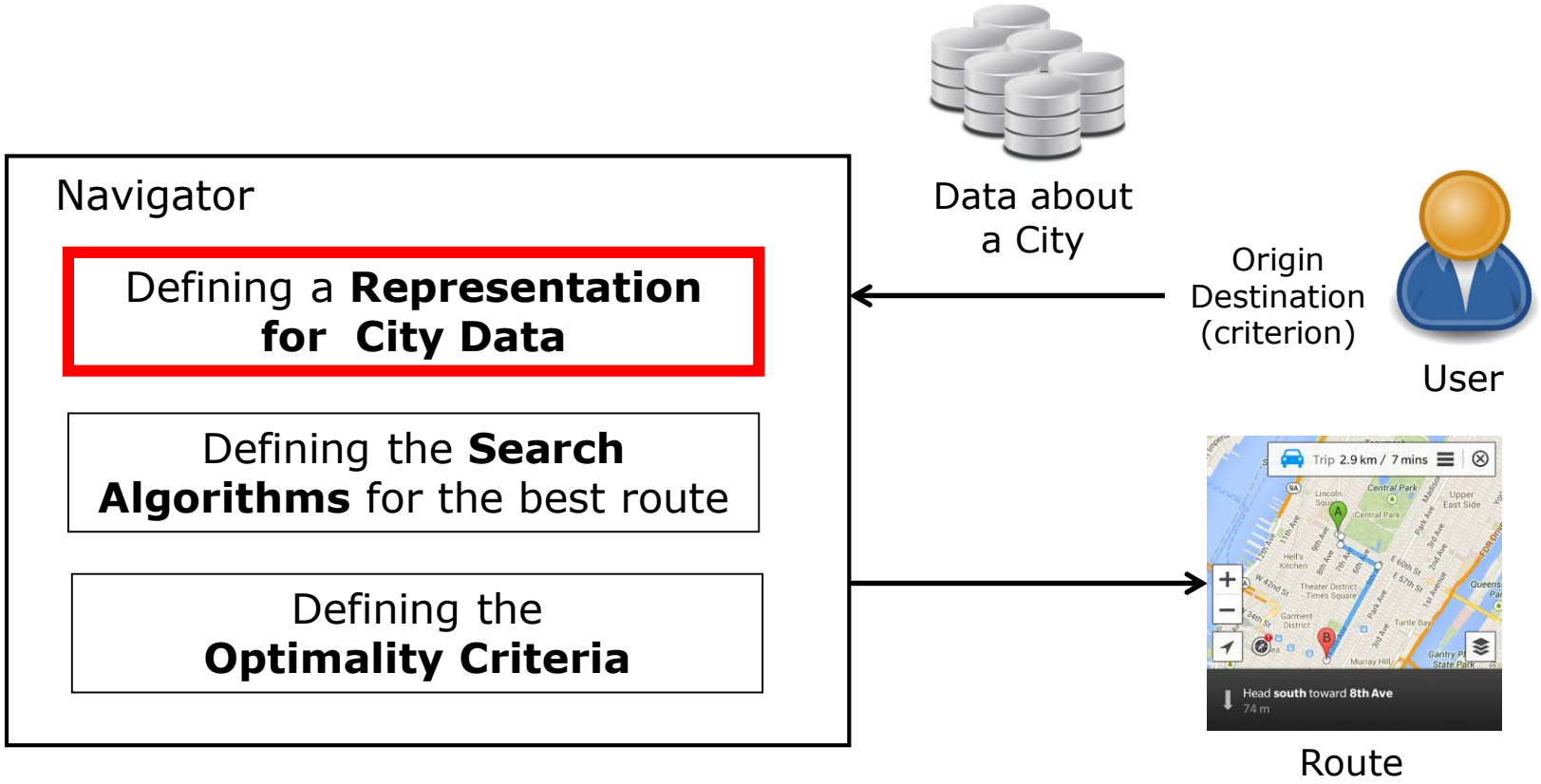
Project 1

Problems to be solved to implement a Navigator:

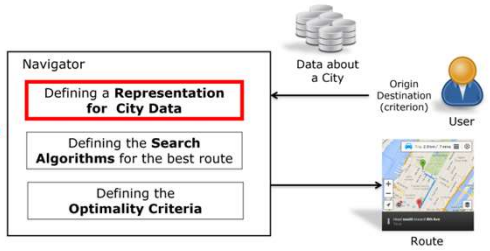


Project 1

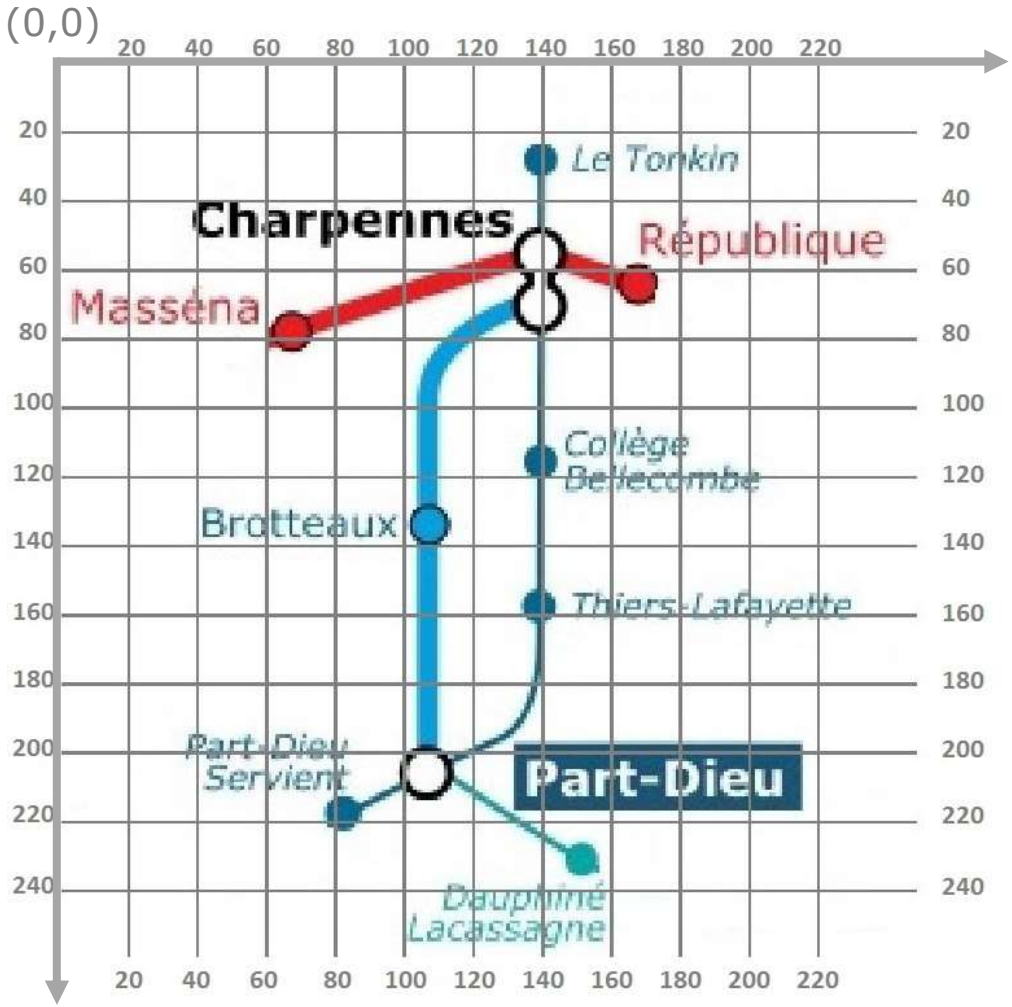
Problems to be solved to implement a Navigator:



Project 1



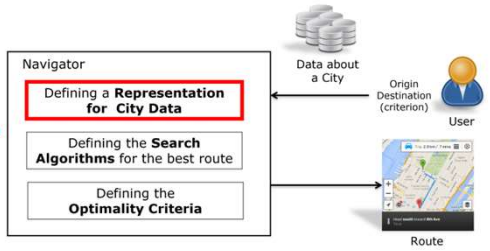
How we do represent the metro map?



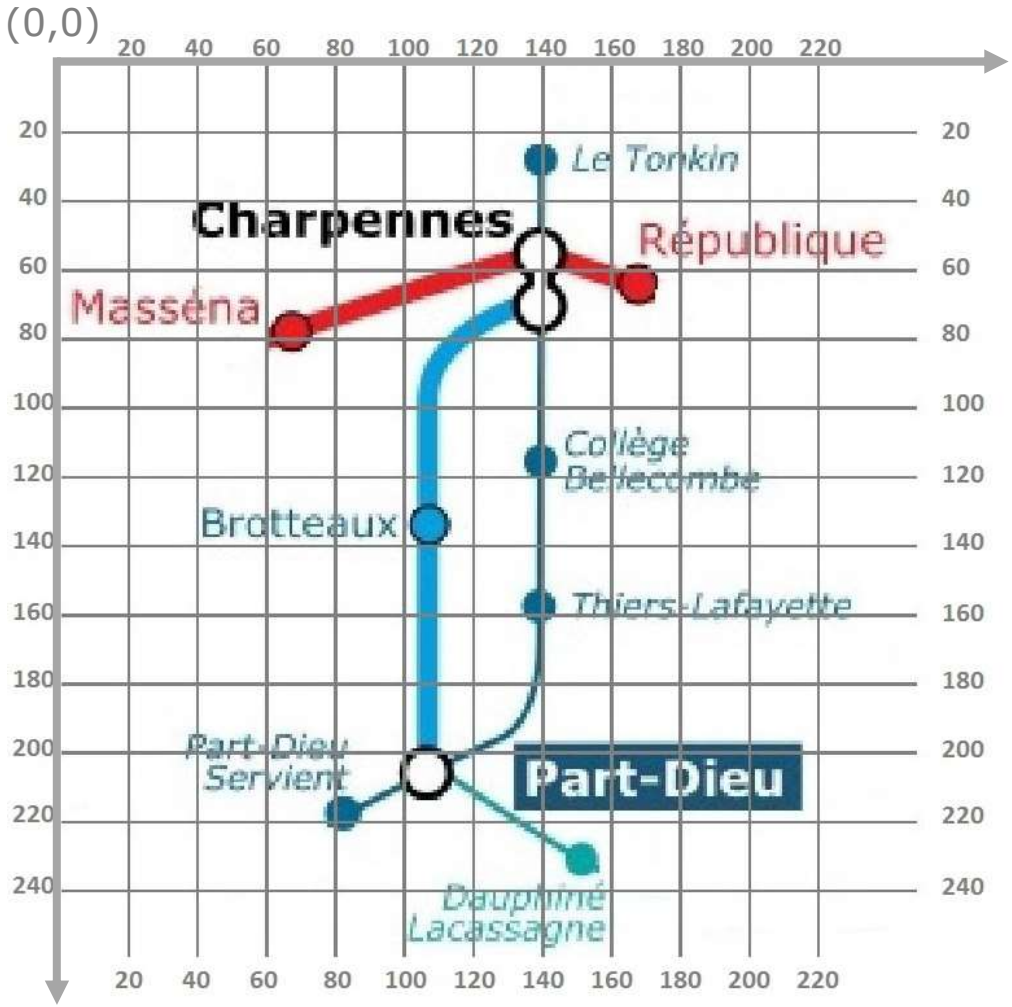
We need to represent
3 elements:

- **Stations**
- **Connections**
- **Line-Changes**

Project 1



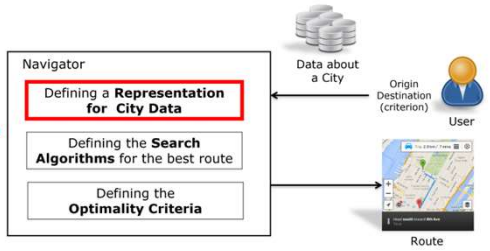
How we do represent the metro map?



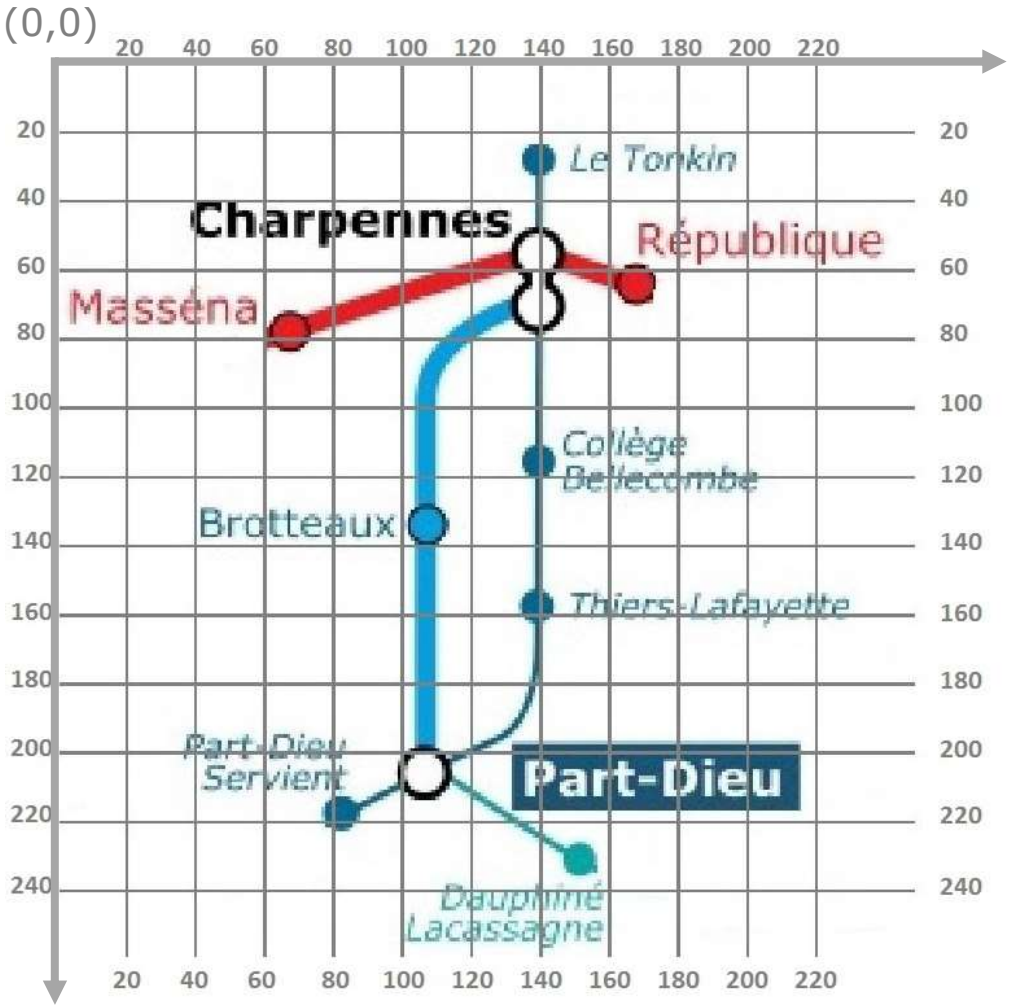
Stations

- **Name**
- **Line** where it belongs to
- **Coordinates** *(position in the map)*

Project 1



How we do represent the metro map?

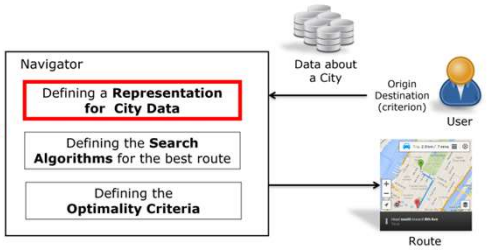


Station's Table

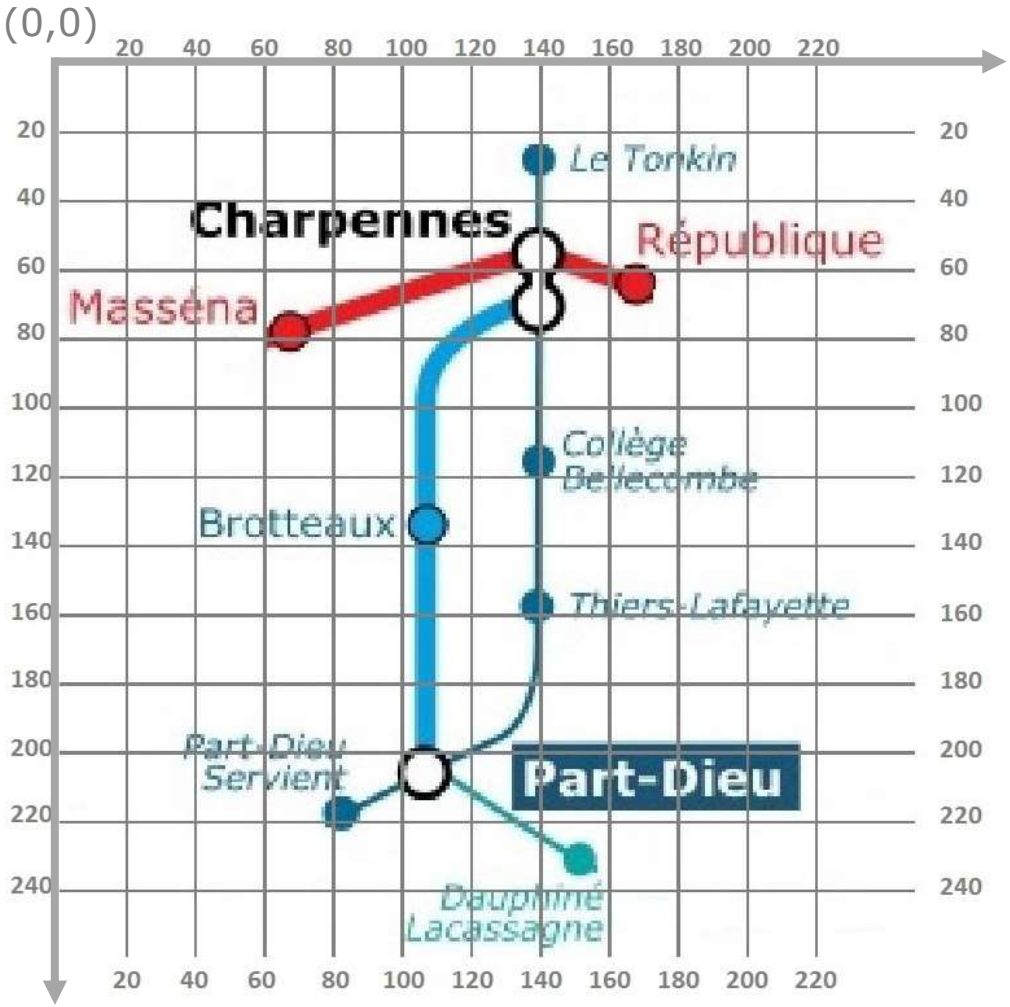
Name – Line - Coordinates

Station	Line(s)	X	Y
Masséna	1	67	79
Charpennes	1,2,3	140	56
République	1	167	64
Le Tonkin	2	140	27
Collège Bellecombe	2	140	115
Thiers-Lafayette	2	140	157
Part-Dieu	2,3,4	108	206
Part-Dieu Servient	2	82	217
Brotteaux	3	108	134
Dauphiné Lacassagne	4	152	230

Project 1



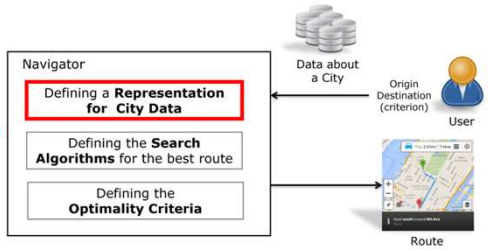
How we do represent the metro map?



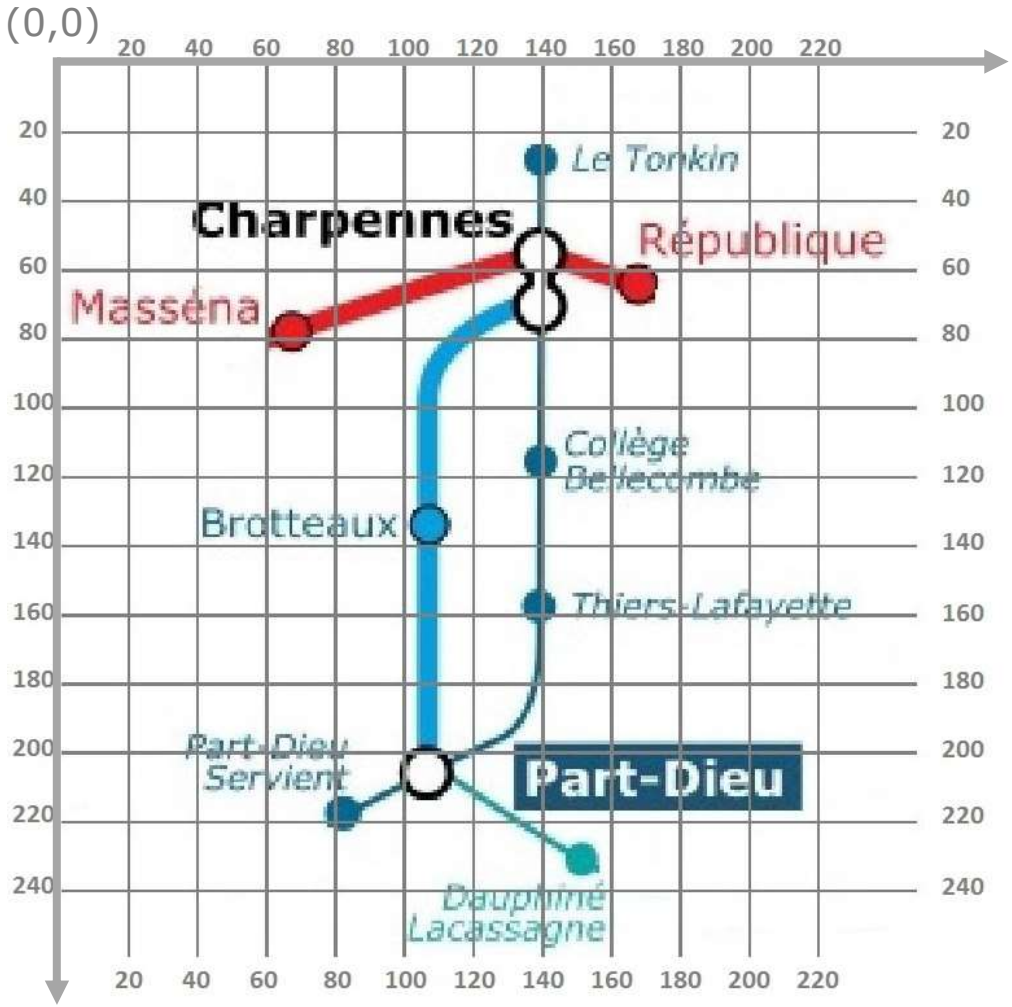
We need to represent
3 elements:

- **Stations** √
- **Connections**
- **Line-Changes**

Project 1



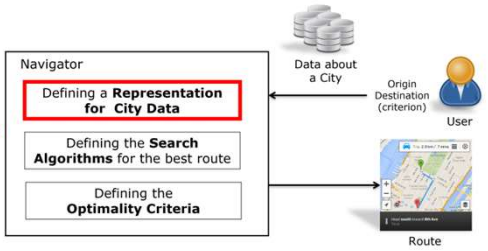
How we do represent the metro map?



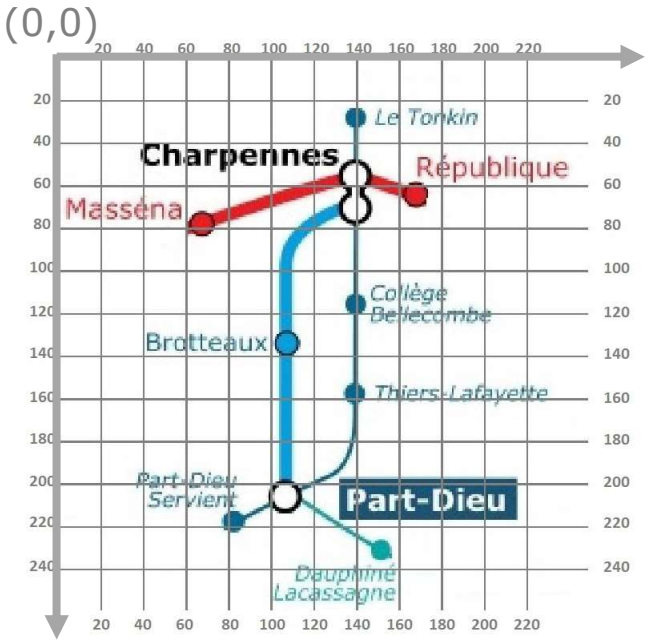
Connections:

- **Adjacency Matrix**

Project 1



Example of adjacency matrix

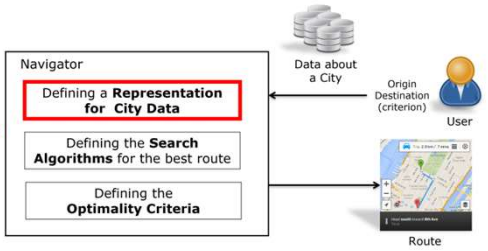


Adjacency Matrix

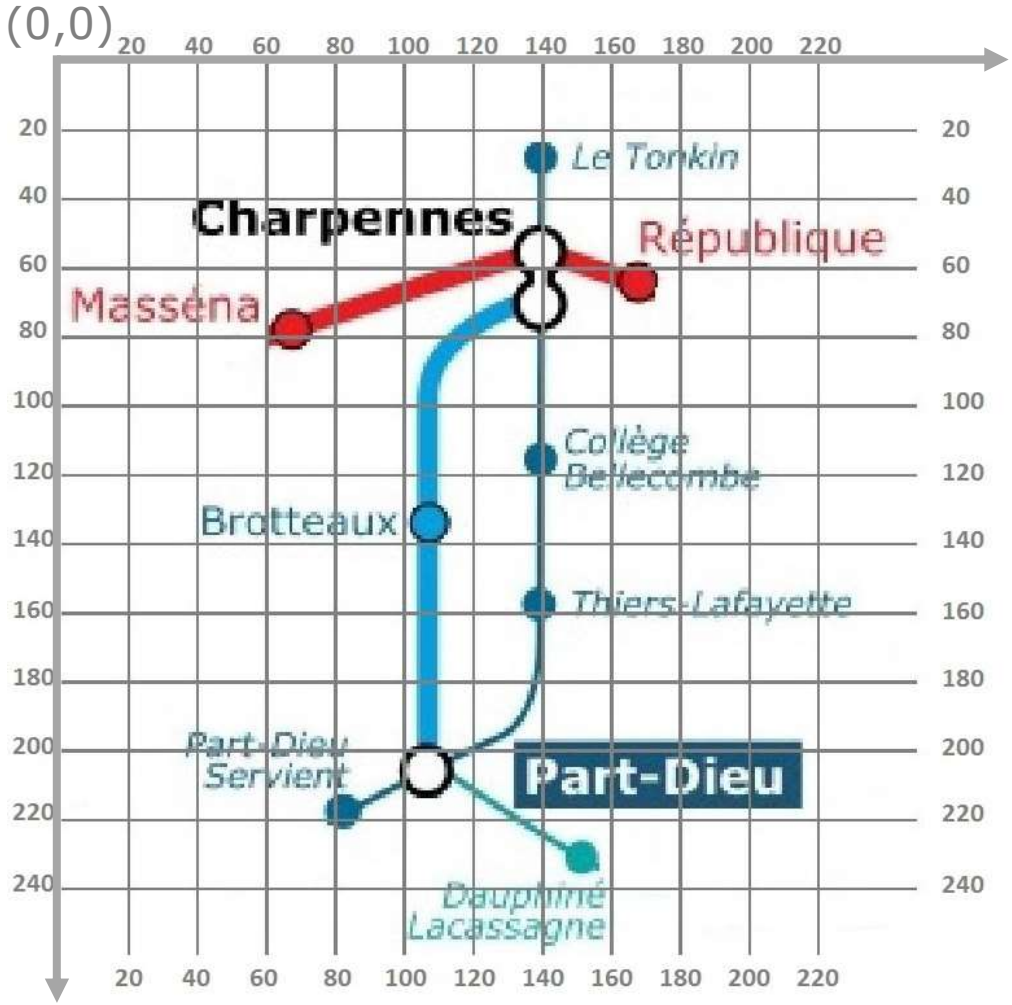
	Masséna	Charpennes	République	Le Tonkin	Collège Bellecombe	Thiers – La fayette	Part-Dieu	Part-Dieu Servient	Brotteaux	Dauphiné
Masséna	0									
Charpennes	1	0								
République		1	0							
Le Tonkin		1		0						
Collège Bellecombe		1			0					
Thiers – La fayette					1	0				
Part-Dieu						1	0			
Part-Dieu Servient							1	0		
Brotteaux		1						1	0	
Dauphiné							1			0

symmetric

Project 1



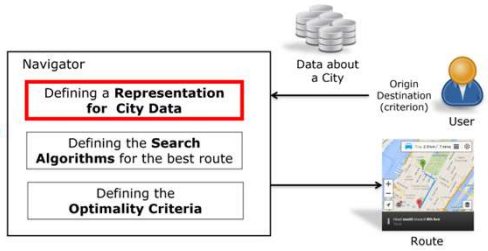
How we do represent the metro map?



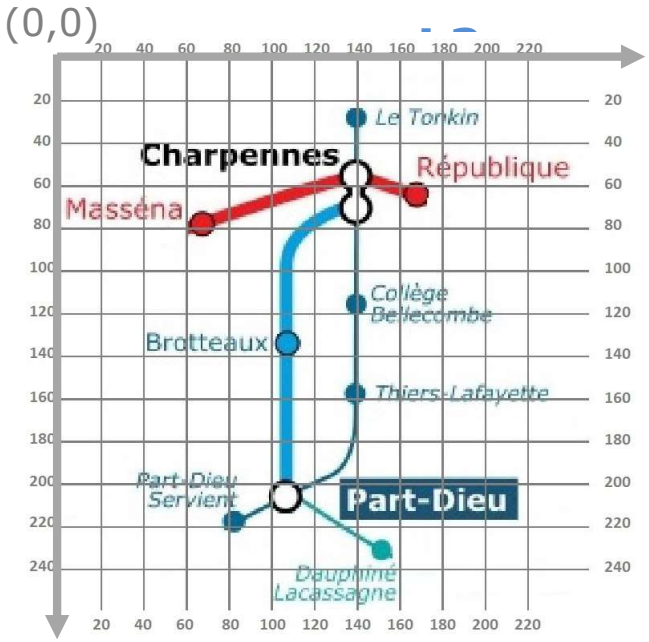
Connections

- **Adjacency Matrix** √
- **Costs**
  1. Time
  2. Distance
  3. #Line-Changes
  4. #Stops

Project 1



Example of a time cost matrix
(it will always be given)

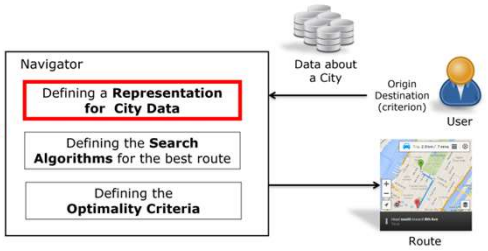


Cost Matrix (Time)

	Masséna	Charpennes	République	Le Tonkin	Collège Bellcombe	Thiers-Lafayette	Part-Dieu	Part-Dieu Serv.ient	Brotteaux	Dauphiné L
Masséna	0									
Charpennes	9	0								
République		4	0							
Le Tonkin		5		0						
Collège Bellecombe		7			0					
Thiers – La fayette					4	0				
Part-Dieu						6	0			
Part-Dieu Servient							2	0		
Brotteaux		2					2		0	
Dauphiné L.								21		0

Folder: `CityInformation`
File: `Time.txt`

Project 1



Assumptions to compute costs:

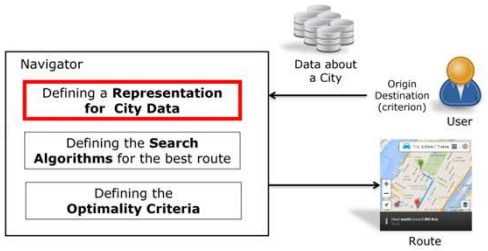
- Each line goes always at a constant speed

Folder: `CityInformation`
File: `Infovelocity.txt`

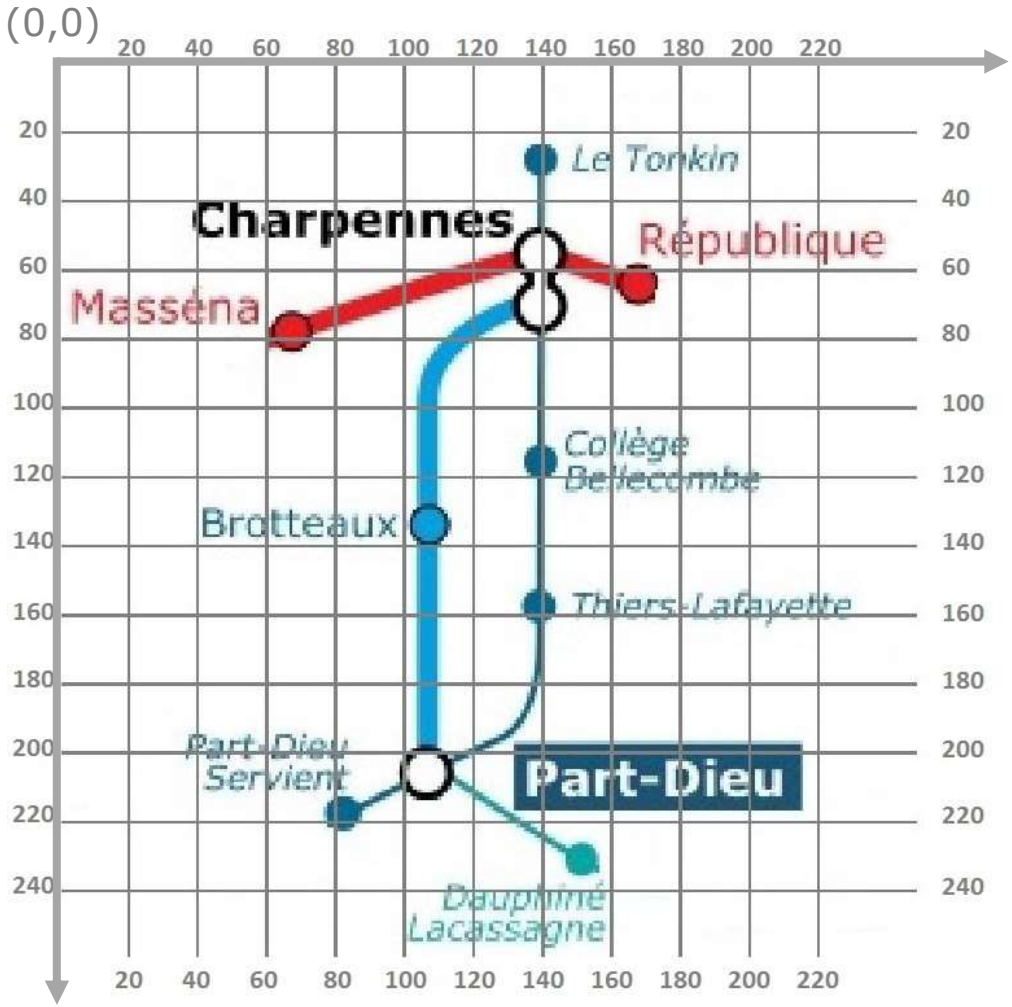
- The railways connecting between two stations are not always straight.

- We will always have the Cartesian coordinates of all the station positions.

Project 1



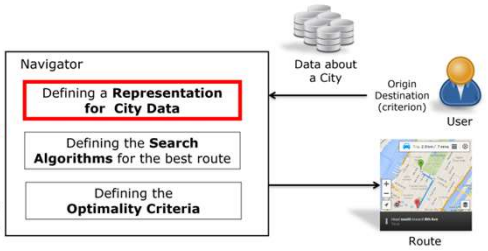
How we do represent the metro map?



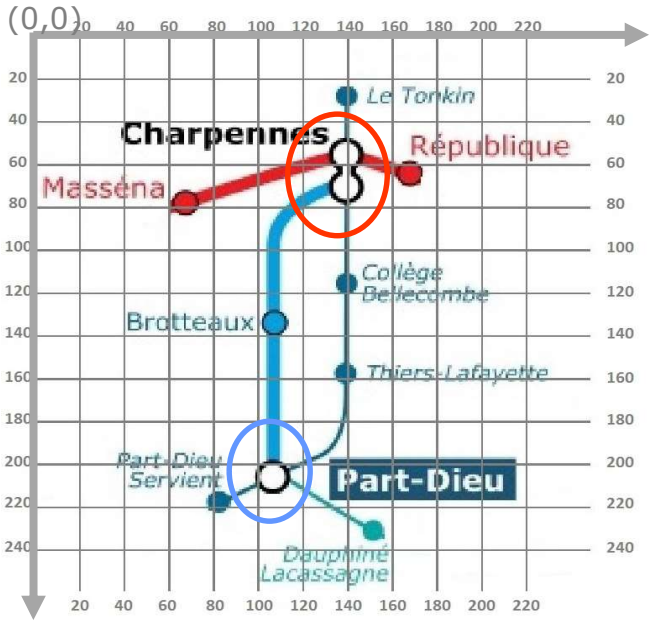
We need to represent
3 elements:

- **Stations** √
- **Connections** √
- **Line-Changes**

Project 1



Previous examples, adjacency matrix and cost matrix



Adjacency Matrix

	Masséna	Charpennes	République	Le Tonkin	Collège Bellcombe	Thiers-Lafayette	Part-Dieu	Part-Dieu Servient	Brotteaux	Dauphiné L.
Masséna	0									
Charpennes	1	0								
République		1	0							
Le Tonkin		1		0						
Collège Bellecombe		1			0					
Thiers – La fayette					1	0				
Part-Dieu						1	0			
Part-Dieu Servient							1	0		
Brotteaux		1						1	0	
Dauphiné							1			0

Cost Matrix (Time)

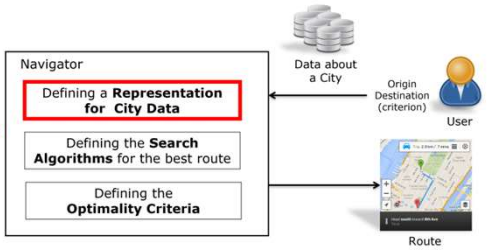
	Masséna	Charpennes	République	Le Tonkin	Collège Bellcombe	Thiers-Lafayette	Part-Dieu	Part-Dieu Servient	Brotteaux	Dauphiné L.
Masséna	0									
Charpennes	9	0								
République		4	0							
Le Tonkin		5		0						
Collège Bellcombe		7			0					
Thiers – La fayette					4	0				
Part-Dieu						6	0			
Part-Dieu Servient							2	0		
Brotteaux		2					2		0	
Dauphiné L.								21		0

How do we represent the Line-Change?

There are two: *Charpennes* (3 lines) and *Part-Dieu* (3 lines)

We said Line-Changes would be regarded as **STOPS**

Project 1



Solution: Repeat stations belonging to more than one line

Example: Adjacency matrix

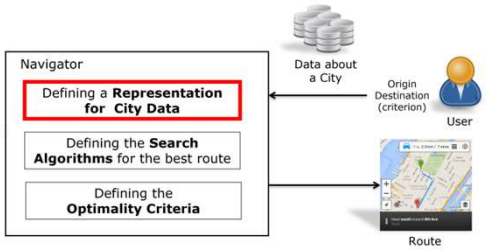
	Masséna	Charpennes	République	Le Tonkin	Collège Bellcombe	Thiers-Lafayette	Part-Dieu	Part-Dieu Serv. Servient	Brotteaux	Dauphiné Lacassagne
Masséna	0									
Charpennes	1	0								
République		1	0							
Le Tonkin		1		0						
Collège Bellecombe		1			0					
Thiers – La fayette					1	0				
Part-Dieu						1	0			
Part-Dieu Servient							1	0		
Brotteaux		1						1	0	
Dauphiné							1			0

	Masséna L1	Charpennes L1	Républicque L1	Le Tonkin L2	Charpennes L2	Collège Bellecombe L2	Thiers Lafayette L2	Part-Dieu L2	Part-Dieu Servient L2	Charpennes L3	Brotteaux L3	Part-Dieu L3	Part-Dieu L4	Dauphiné L. L4
Masséna L1	0													
Charpennes L1	1	0												
République L1		1	0											
Le Tonkin L2				0										
Charpennes L2		1		1	0									
Collège Bellecombe L2					1	0								
Thiers Lafayette L2						1	0							
Part-Dieu L2							1	0						
Part-Dieu Servient L2								1	0					
Charpennes L3		1			1					0				
Brotteaux L3										1	0			
Part-Dieu L3								1			1	0		
Part-Dieu L4								1				1	0	
Dauphiné Lacassagne L4													1	0

Result: Move from a 10x10 matrix to a 14x14 matrix

(Charpennes x 3) i (Part-Dieu x 3)

Project 1



Solution: Repeat stations belonging to more than one line

Example: Cost Matrix

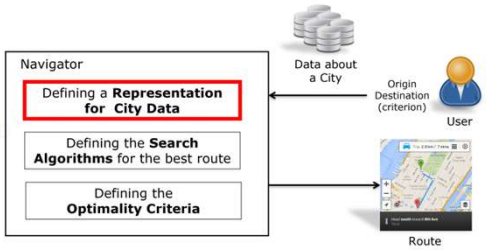
	Masséna	Charpennes	République	Le Tonkin	Collège Bellcombe	Thiers-Lafayette	Part-Dieu	Part-Dieu Serv.	Servient Brotteaux	Dauphiné Lacassagne
Masséna	0									
Charpennes	10	0								
République		10	0							
Le Tonkin		5		0						
Collège Bellecombe		5			0					
Thiers – La fayette					5	0				
Part-Dieu						5	0			
Part-Dieu Servient							5	0		
Brotteaux		20						20	0	
Dauphiné							15			0

	Masséna L1	Charpennes L1	Républicque L1	Le Tonkin L2	Charpennes L2	Collège Bellecombe L2	Thiers Lafayette L2	Part-Dieu L2	Part-Dieu Servient L2	Charpennes L3	Brotteaux L3	Part-Dieu L3	Part-Dieu L4	Dauphiné L. L4
Masséna L1	0													
Charpennes L1	9	0												
République L1		4	0											
Le Tonkin L2				0										
Charpennes L2		20		5	0									
Collège Bellecombe L2					7	0								
Thiers Lafayette L2						4	0							
Part-Dieu L2							6	0						
Part-Dieu Servient L2								2	0					
Charpennes L3		15			18					0				
Brotteaux L3										2	0			
Part-Dieu L3								12			2	0		
Part-Dieu L4								6				15	0	
Dauphiné L. L4													21	0

Result: Move from a 10x10 matrix to a 14x14 matrix

(Charpennes x 3) i (Part-Dieu x 3)

Project 1

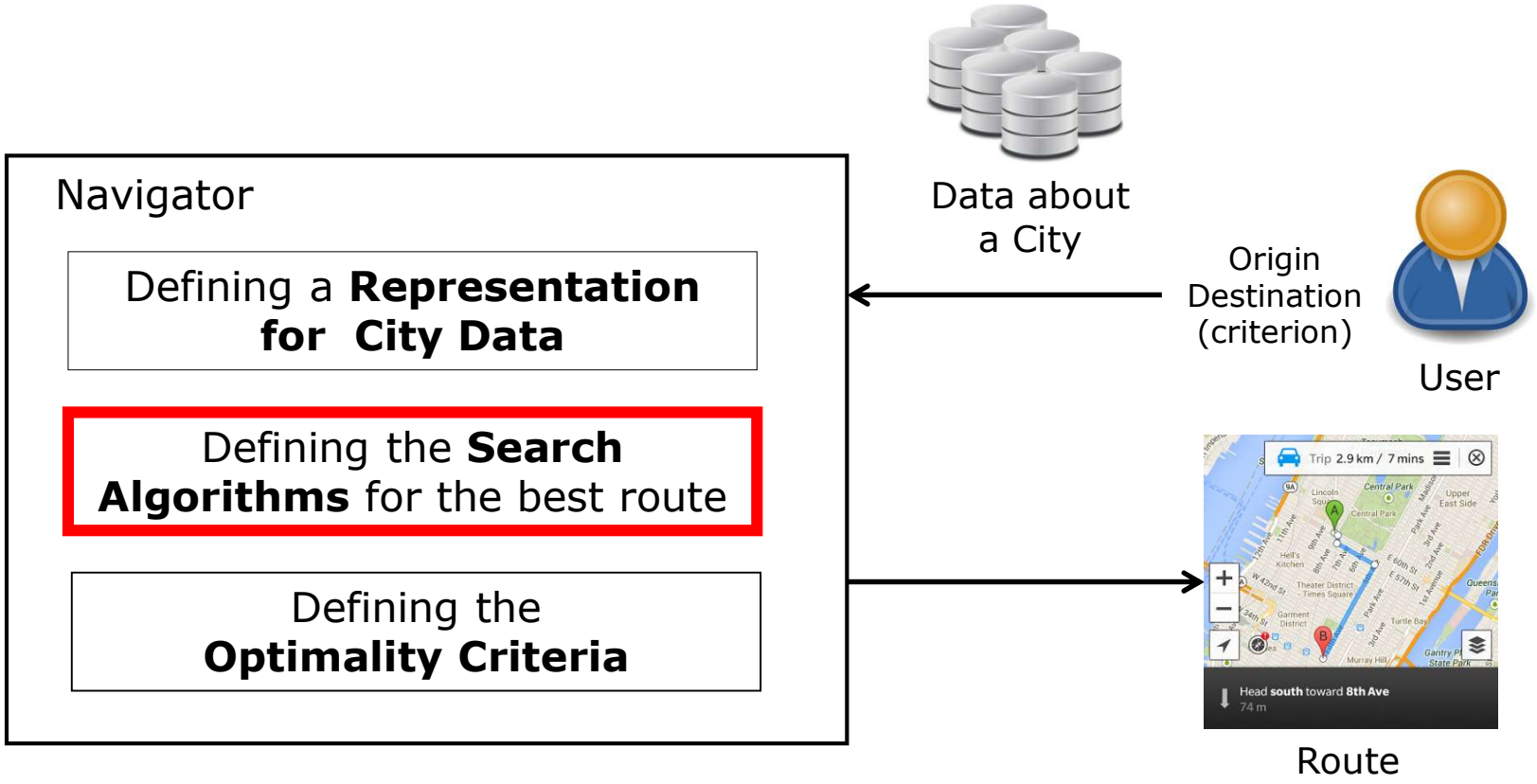


Important Note: the cost matrices we provide you have already duplicated the stations of different lines

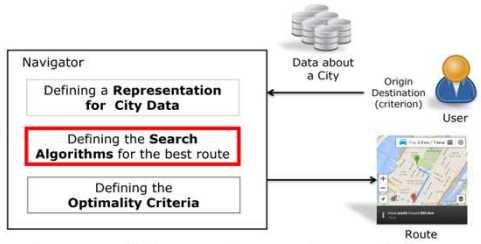
Folder: `CityInformation`
File: `Infovelocity.txt`

Project 1

Problems to be solved to implement a Navigator:



Project 1



Which algorithm do we have to apply, to implement a Navigator?

Search Algorithms, allow to find the path between an origin and a destination

Which are the differences between them?

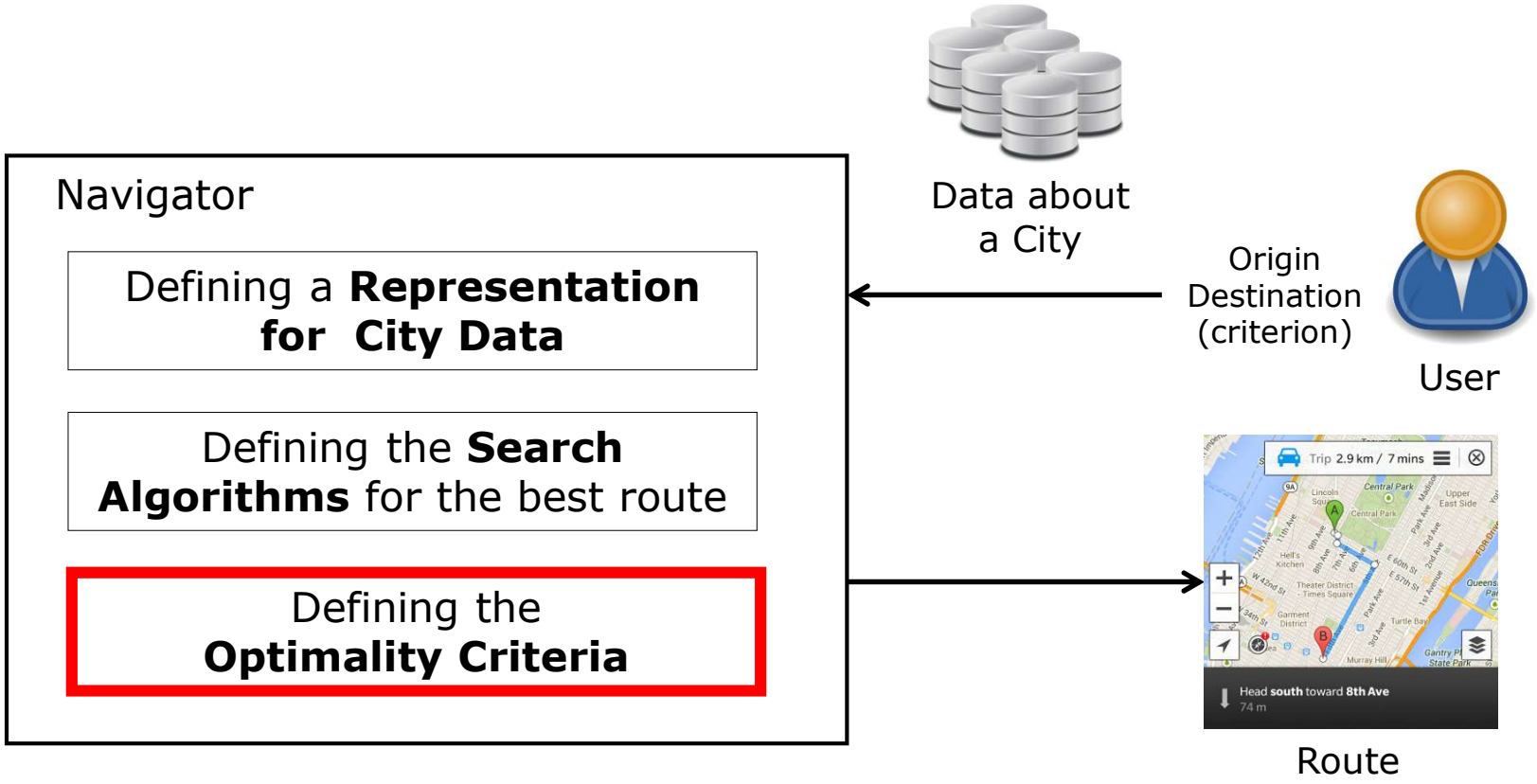
	Non-Optimal Search	Optimal Search
Uninformed Search	Depth first Search Breadth first Search	Uniform Cost Search
Informed Search	Greedy-BFS Search	A* Search

Two questions:

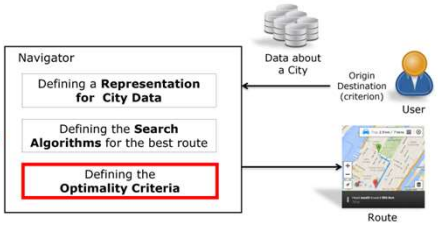
- Do we want optimal routes in a given sense?
- Can we define heuristics on these searches?

Project 1

Problems to be solved to implement a Navigator:



Project 1



How do we apply each criterion?

- Time criterion
  - What the cost is?
  - Which heuristic?
- Distance criterion
  - What is the cost?
  - Which heuristic?
- Number of Line-Change criterion
  - What is the cost?
  - Which heuristic?
- Number of Stops criterion
  - What is the cost?
  - Which heuristic?

Remember:

- We know the time between all stops in all metro lines
- Each line has constant speed
- We know the position coordinates of all stations
- Railways between stations are not straight lines

Project 1

Planning

Week 3: Orientation Session (1h on Monday)
Support Session (1h on Wednesday or Thursday)
Delivery:
What? Exercises indicated in Part 1 (`Practice1_1.pdf`)
When? Before Sunday **March, 3rd** at 23:55h.

Week 5: Orientation Session (1h on Monday)
Support Session (1h on Wednesday or Thursday)
Delivery:
Delivery:
What? Exercises indicated in Part 2 (`Practice1_2.pdf`)
When? Before Sunday **March, 17th** at 23:55h.

AUTOMATIC CODE CORRECTION. After these two deliveries we will publish the results of the automatic correction and you can improve it before the exam. To do the exam you need to have passed the Code Auto-Correction.

Project 1

Practical tips for Session 1:

- You will find the Guidelines at <`Practice1_1.pdf`> at cv.uab.cat. This document will guide you through everything you need to program.
- Save all functions in the file <`SearchAlgorithm.py`>
- Code the functions as it is specified in the Guidelines in regard of input parameters and return values.
- Delivery will be through the Campus Virtual. You will upload a file containing all the functions worked out in Part 1.
- For this Part 1 you will use the map of the Lyon metro, which is in the directory: <`Lyon_smallCity`>
- Recommendations before to go to the Practice Sessions:
  - **Orientation Session:** Have read the guide, <`Practica1_1.pdf`>, have downloaded the necessary files and understood its contents, have started to program the functions.
  - **Support Session:** You should have started to code all the functions you have to deliver; in this way you can solve all the doubts you might have during the session.

Project 1

Practical tips for Session 2:

- You will find the exercises in the file <`Practica1_2.pdf`> at cv.uab.cat, this document will guide you through everything you need to program.
- Save all functions in the file <`SearchAlgorithm.py`>
- Delivery will be through cv.uab.cat. You will upload a file containing all the functions worked out in Part 2.
- You will program optimal algorithms for all the criteria
- For this Part 2 you will use the small map of Lyon metro, which is in the folder: <`Lyon_smallCity`>
- As for the correction, we will can use any other map.

Project 1

Practical tips for the Exam:

- For the exam you will use your own laptop, in the case you cannot bring one, please let us know beforehand.

- Goal of the exam is:
  - ✓ Evaluate your skills on identifying in your own code the specific functions you need to run in order to answer the exam questions.
  - ✓ Evaluate yout ability running the needed functions using the proper parameters.
  - ✓ Evaluate your skills in managing all the data structures you have used in the code: classes, dictionaries, etc.

- The exam about Project 1 will be on the same day of the Theory Exam, you will receive the exam once you have finished and delivered the Theory Exam.

Project 1

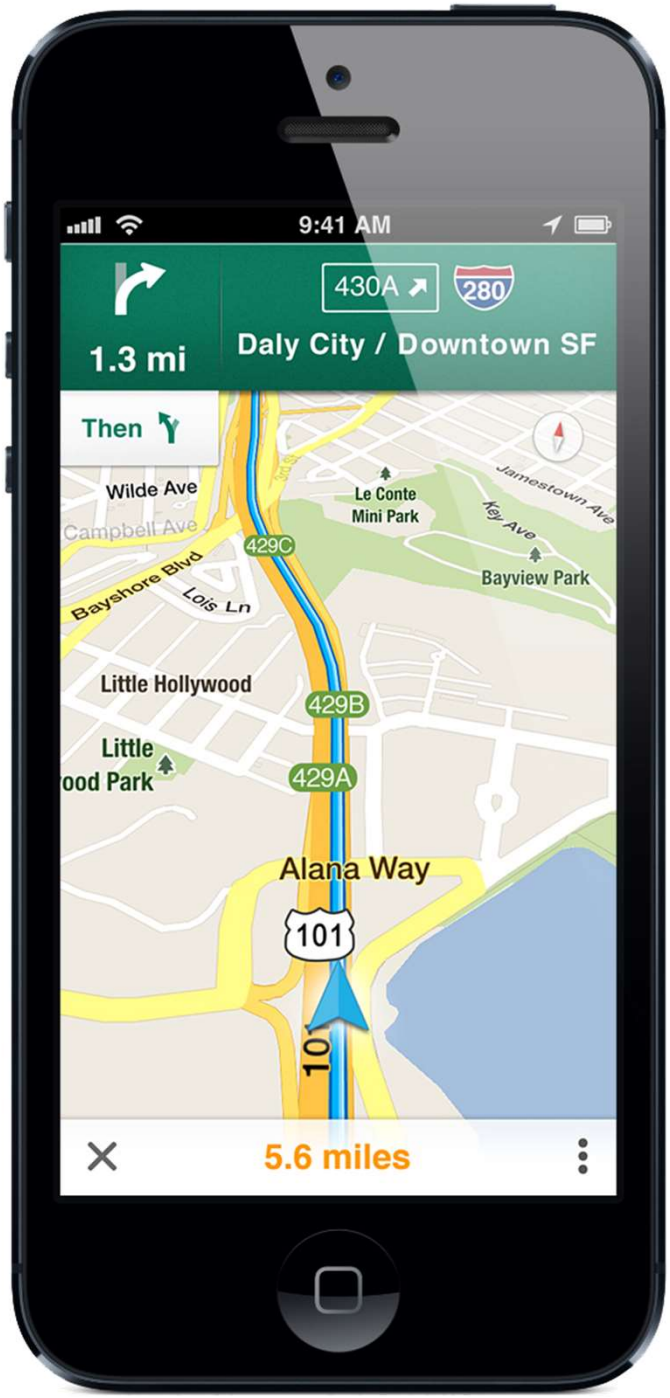
Evaluation:

Mark = 0,6 * Code + 0,4 * Exam

- **Code Score**: It is evaluated with a series of tests that will be run after every delivery. The test evaluates correctness and efficiency of the code.

- **Exam Score**: Is evaluated with an exam while you are allowed to run your own code.

The Code Score and the Exam Score must be greater than or equal to 5.



PROJECT 1:

Navigator

Artificial Intelligence

Universitat Autònoma de Barcelona