Software Engineering

Books or notes are **not** allowed.
Write only on these sheets. **Concise** and **readable** answers please.

Surname, name, matricola _____

Maps and traffic service
Most persons have smartphones with GPS, while detailed digital maps are available. This allows to offer services for location (showing where the user is on a map), and navigation.
Further, the traffic situation on most roads can be computed by continuously collecting and aggregating the position of many smartphones.

In the following you should analyze and model the client server application that supports location, navigation and traffic information.

The key role to be considered is the end user (smartphone owner)

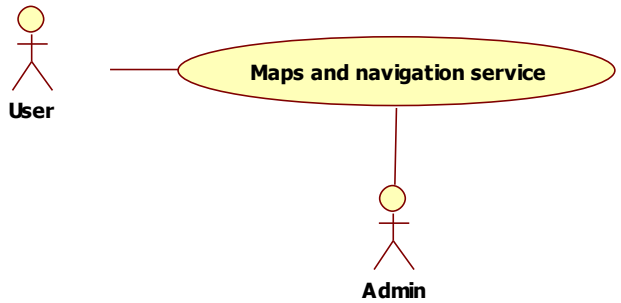
Key high level functions to be considered are:

- Location (show the position of the smartphone on a map)
- Navigation (compute one or more routes from place A to place B)
- Location and navigation (show position and continuously produce directions to follow a given route)
- Traffic information (collect and aggregate the position of smartphones on car roads, estimate the speed of cars on road segments)

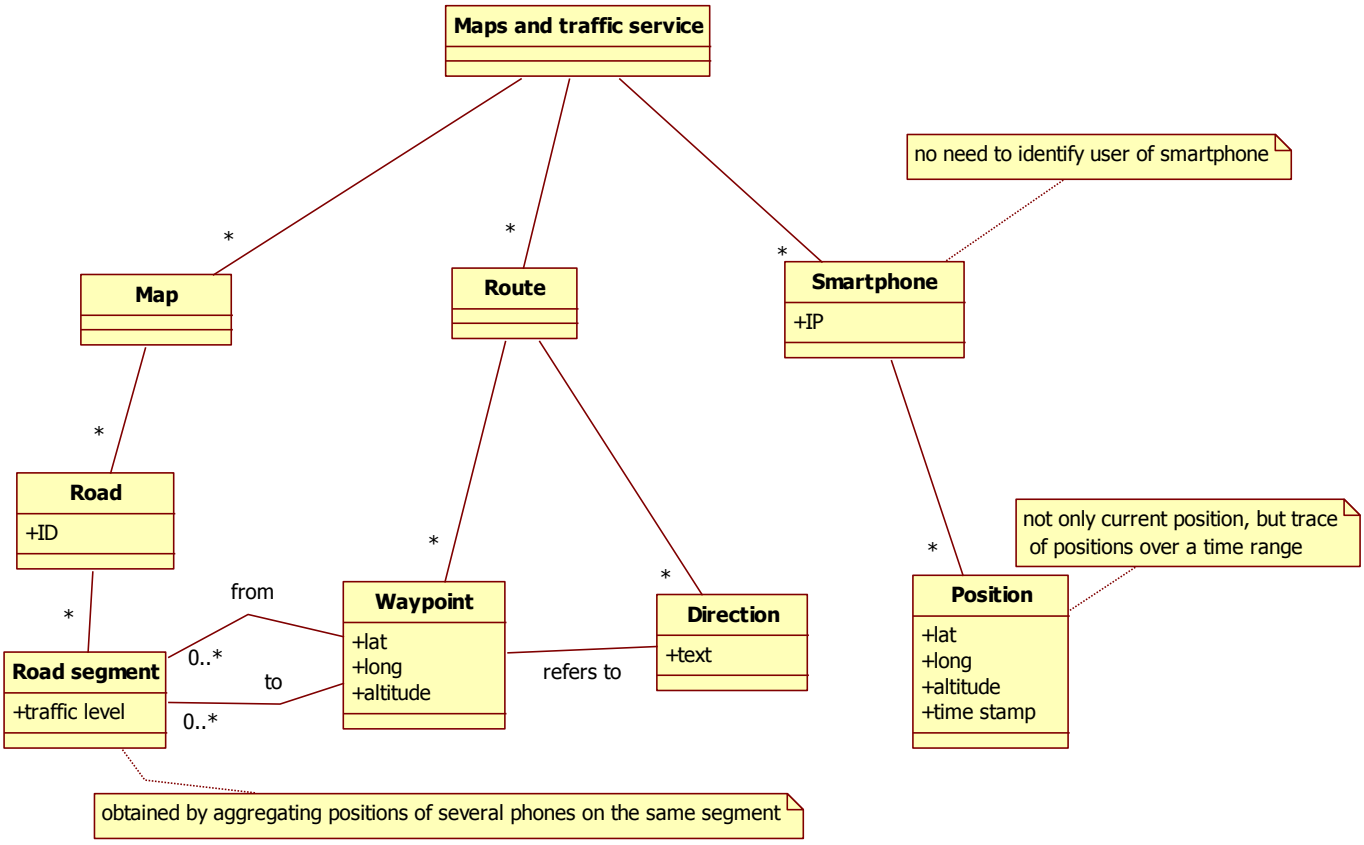
1 – a. Define the **context diagram** (including relevant interfaces)

Actor	Physical interface	Logical interface
User	Smartphone touch screen	GUI
admin	pc	GUI

The GPS on the smartphone could also be considered as an actor, with an API as logical interface
The context diagram considers the whole service (i.e. both client and server, not the client only)

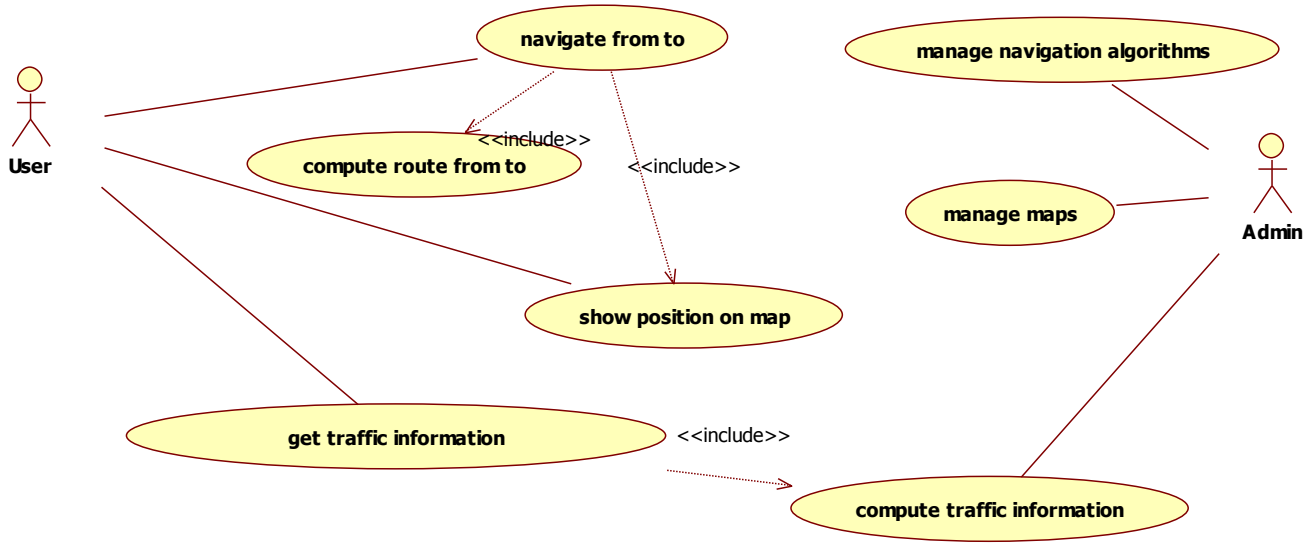


1-b Define the **glossary** (key concepts and their relationships) (UML class diagram) for the application



1-c Draw the Use Case Diagram for the application. For each Use Case give a short description here

Use Case ID	Description
1	Show map and position of smartphone (user) on it
2	Compute traffic information
3	Show traffic information on map
4	Compute route from position to position
5	Navigate from to: given a position and a route to another position provide step by step directions



1-D List the **NON functional requirements** that you deem important for the application

ID	Description
1	Privacy: the position of user / smattphone should be known only by herself (to compute traffic info position should be anonymized)
2	Efficiency: all functions should respond in <0,5sec
3	Usability: an average smartphone user (uses smartphone from more than 12 months) must be able to use the system in 10 minutes without any training

Remark that NF MUST BE measurable. NF such as ‘system should be easy to use, should be accurate, should be reliable’ are not acceptable.

1-e Choose a Use Case from 1-c and describe a corresponding scenario below

Use Case ID:
Precondition:

Postcondition:

Step	Description
1	
2	
3	
4	
5	

2 (7 points) -Define black box tests for the following function, using equivalence classes and boundary conditions.

double computeCarSharingFee (int driving_time, int stop_time, int reservation_time);

The function computes the fee due by the user for one session of car sharing. The fee depends on three parameters: driving time, stop time, reservation time. The fee is 0.25€ per minute of driving, 0.10€ per minute of stop, and 0.10€ per minute of reservation exceeding the first 15 minutes that are free.

When the user needs to stop temporarily (without ending the rental) the system applies the stop fee until the user drives again.
When the user reserves the car, the reservation is free for the first 15 minutes. If the user needs additional time to get by the reserved car, the system applies the reservation fee for every minute exceeding the first 15.

Examples:

computeCarSharingFee(10, 0, 10) → 10*0.25 + 0*0.10 + 0 = 2.5€
computeCarSharingFee(10, 10, 25) → 10*0.25 + 10*0.10 + 10*0.10 = 4.5€

Driving_time	Stop_time	Reservation_time	Valid / Invalid	TC
[minint, 0]	[minint, -1]	[minint, -1]	I	(-5, -5, -5) error
"	"	[0, 15]	I	(-5, -5, 5) error
"	"	[16, maxint]	I	(-5, -5, 20) error
"	[0, maxint]	[minint, -1]	I	(-5, 5, -5) error
"	"	[0, 15]	I	(-5, 5, 5) error
"	"	[16, maxint]	I	(-5, 5, 20) error
[1, maxint]	[minint, -1]	[minint, -1]	I	(5, -5, -5) error
"	"	[0, 15]	I	(5, -5, 5) error
"	"	[16, maxint]	I	(5, -5, 20) error
"	[0, maxint]	[minint, -1]	I	(5, 5, -5) error
				B (5, 5, -1) error
"	"	[0, 15]	V	(5, 5, 5) - 1.75
				B (5, 5, 14) - 1.75
				B (5, 5, 15) - 1.75
				B (5, 5, 0) - 1.75
"	"	[16, maxint]	V	(5, 5, 20) - 2.25
				B (5, 5, 16) - 1.85
				B (5, 0, 16) - 1.35
				B (5, 1, 16) - 1.45

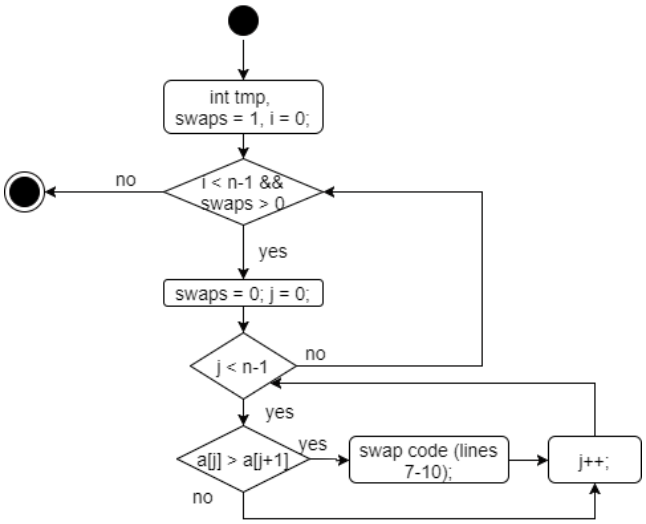
Equivalence classes:
driving_time: [minint, 0], [1, maxint] supposing that driving_time = 0 makes no sense for a car rental
stop_time: [minint, -1], [0, maxint]
reservation_time: [minint, -1], [0, 15], [16, maxint]

Boundary values to be tested (some combinations provided among test cases in table):
driving_time: -1, 0, 1
stop_time: -1, 0, 1
reservation_time: -1, 0, 1, 14, 15, 16

3 (7 points) – For the following function define the control flow graph, and define test cases to obtain the highest possible node coverage, edge coverage, multiple condition coverage, loop coverage, path coverage. For the test cases, **write only the input value**.

WRITE control flow graph here

```
1 void bubblesort (int *a, int n) {
2     int tmp, swaps = 1;
3     for (int i=0; i<n-1 && swaps > 0; i++) {
4           swaps = 0;
5           for (int j=0; j<n-1 ; i++) {
6                 if (a[j] > a[j+1]) {
7                       tmp = a[j];
8                       a[j] = a[j+1];
9                       a[j+1] = tmp;
10                      swaps++;
11                }
12          }
13    }
14 }
```

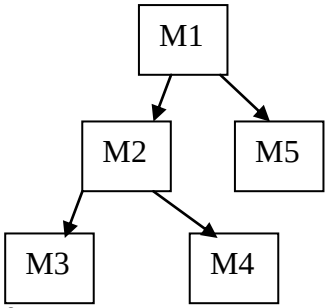


Coverage type	Number of test cases needed to obtain 100% coverage	Coverage obtained with test cases defined (%)	Test cases defined
Node	1	100%	TC1
Edge	1	100%	TC1
Multiple condition line 3	4 (3 in practice)	100%	TC1, TC2, TC3
Loop line 3	3	100%	TC1, TC2, TC3
Path	o(2^(n-1)(n-1))		

TC1: ({1, 3, 2}, 3) covers multiple condition TT, FF, multiple iterations of loop;
TC2: ({1}, 1) covers multiple condition FT, no interations of loop;
TC3: ({1, 2, 3}, 3) covers multiple condition TF, single iteration of loop.

The amount of iterations to achieve full path coverage depends on the size of n, so it is not easy to achieve full coverage at the variation of n. The total amount of paths is upper bounded by 2^(n-1)(n-1): due to the possible lack of swap in initial iterations that makes the control flow exit from the outer for loop, some of the possible theoretical paths are not possible.

4 (1 points) – Given this dependency graph propose and justify an integration strategy.



If bottom up: test M3, test M4, test M2+M3+M4, test M5, test all
If top down: test M1 +stubM2 + stubM5, test M1+M2+M5+stubM3 + stubM4, test all
Bottom up requires less stubs / drivers

5 (1 point) – Describe the SCRUM process

See slides

6 (1 point) — In the context of configuration management, explain what is versioning and why it is useful.

Versioning = keep copy of each instance of a configuration item (CI). This allows to keep the history of all modifications to a CI, and allows to roll back to any past instance of a CI

7 (1 point) – Describe a maintenance software process

Receive change requests (CR)
Filter CR
Rank CR
Assign CR to developer – design, code, unit test, integration test
Merge CR with next release of application
release

8 (1 point) – Define the repository architectural pattern. List its pros and cons.

See slides
(question is about architectural pattern, not about configuration management / repository as git, svn etc)