Software Engineering

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

Surname, name, matricola _____

Parcel handling and tracking

A logistic company (ex DHL, UPS) handles shipping of parcels all over the world. A key function is tracking the position and status of each parcel, from reception to delivery.
Each parcel is associated to a waybill, that contains name and address of sender, name and address of addressee, (the person to whom the parcel is sent) and a unique ID (also called tracking number). The waybill (and notably the ID written as bar code) is printed and attached to the parcel.

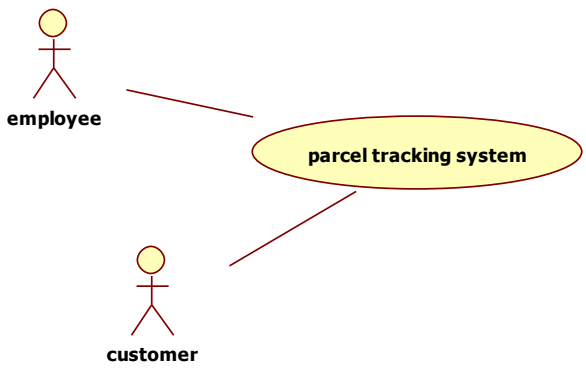
The parcel, from origin to destination, is routed through a number of intermediate points. In term of graphs, the parcel follows a path (sequence of nodes) on a graph. Routing the parcel means deciding the path. Tracking the parcel means collecting the actual sequence of nodes followed by the parcel, with the time of reception in each node. This information is collected through a specific device, that, upon reception of the parcel at a node, reads the bar code, and sends to a server time, bar code, location (i.e. node). The device is custom made, but in fact it can be seen as a modified smartphone (GPS for the location, bar code reader, internet connection to a server). The device also has a touch screen used to collect the signature of the addressee who receives the parcel at delivery (this signature will be the proof of delivery).

The tracking system has two types of users: employees of the company in charge of handling parcels, and end users who access the system via a web site. On the web site a user, entering a tracking number, can follow the shipping of a parcel, until delivery. Tracking information for each parcel is kept for 90 days.

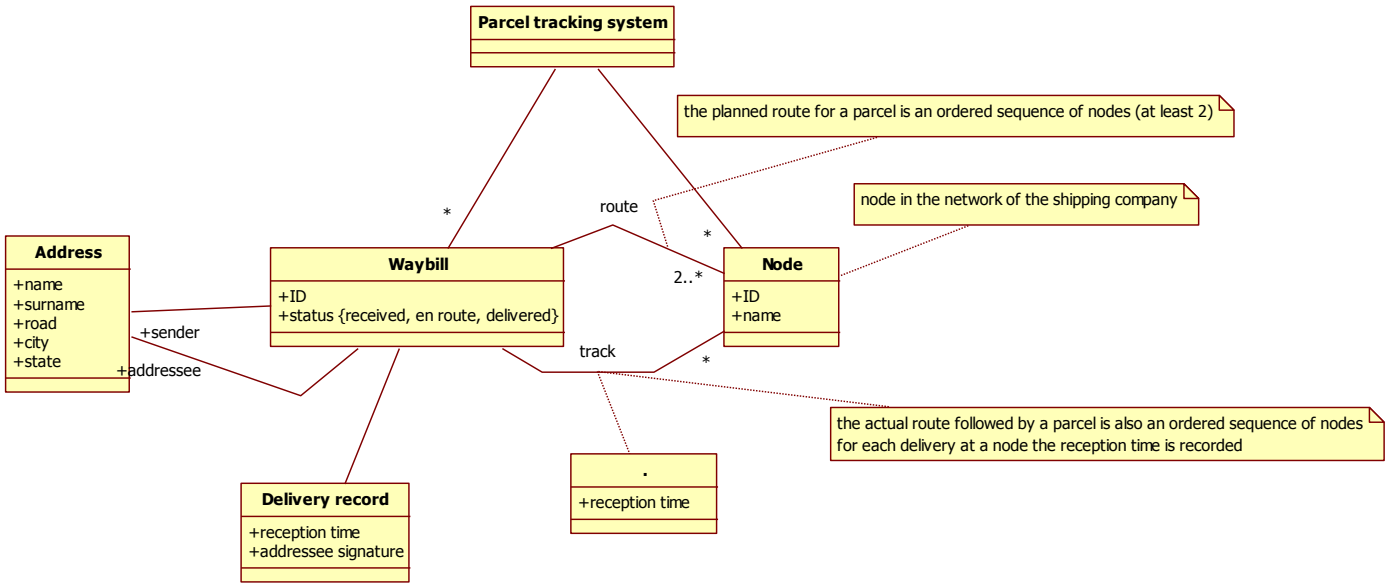
In the following consider the **system** for parcel tracking (including devices, servers, printers).

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

Actor	Physical interface	Logical interface
Employee	Custom device (aka smartphone)	GUI
Customer (sender or addressee)	PC or smartphone	GUI



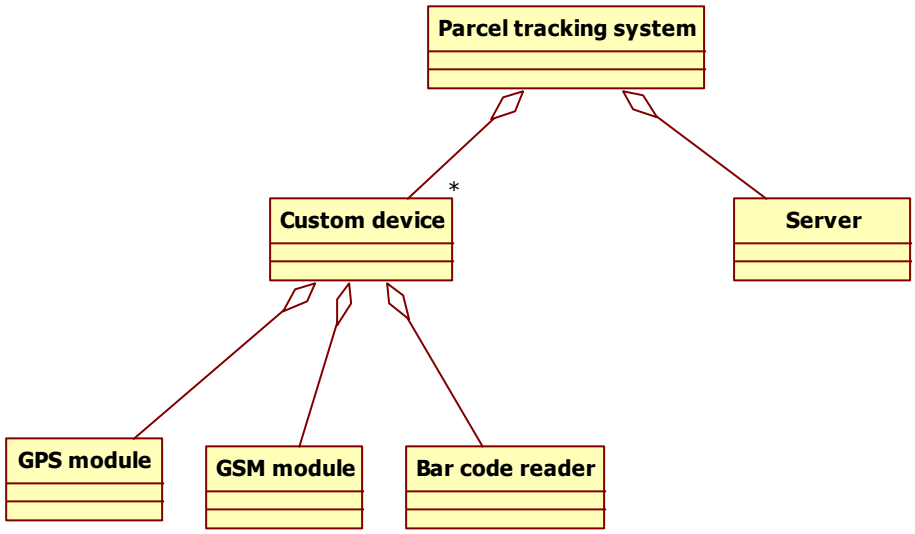
1-b Define the **glossary** (key concepts and their relationships) (UML class diagram) for the system



1-c List the **requirements** in tabular form (do not forget to list important NF requirements)

ID	Type (Functional Non Functional)	Description
1	F	CRUD waybill (creation == receive parcel and start managing it)
2	F	CRUD address
3	F	CRUD node
4	F	CRUD delivery record
5	F	Create planned route, given sender and addressee
6	F	Record reception of parcel at a node
7	F	Record reception of parcel at addressee
8	F	Search waybill given ID, show attached track and status
9	F	Delete tracking info after 90 days from waybill creation
10	F	Collect signature and attach to delivery record
	NF	Efficiency: response time for all functions <0,5 sec
	NF	Privacy: customer must only see data for his parcel

1-d Define the **system design** model (UML class diagram)



1-e Define one **scenario** describing the delivery of parcel P to the addressee A from sender S. (Don't forget that the addressee has to sign the proof of reception, that closes the shipping process).

Precondition: P is delivered to A. W is waybill for P. W is in state 'en route'

Postcondition: Delivery record D created for W. D contains date and signature. W is in state 'delivered'.

Step	Description	Req ID
1	Search waybill W given tracking number on parcel P	8
2	Create delivery record D Attach D to W	4
3	Collect signature and attach to D	10
4	Set status of W to delivered	1
5		

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

boolean acceptLuggage (int nLuggage, int lenght1, int width1, int depth1, int weight1,
int lenght2, int width2, int depth2, int weight2)

This function decides if luggage for a flight can be accepted. The rules are:

- max 2 pieces of luggage
- for each piece of luggage, the sum of the three dimensions must be <= 300cm
- the total weight of all luggage must be <= 30kg

the parameters are as follows

nLuggage: number of pieces of luggage of the passenger
lenght1, width1, depth1: dimensions of first luggage, in cm
lenght2, width2, depth2: dimensions of second luggage, in cm
weight1, weight2: weight of luggage, in kg

Ex. acceptLuggage (2, 100, 50, 10, 15 , 110, 60, 15, 10) → = accept (total weight 25, dimensions 160 and 185)
acceptLuggage (2, 100, 50, 10, 28 , 110, 60, 15, 10) → = NO accept (total weight 38, dimensions ok)

nLuggage	Lenght1	Width 1	Depth 1	Weigh t1	Lenght 2	Width 2	Depth 2	Weig ht2	Total dim1	Total dim2	Total weig ht	V / NV	Test case
[minint, 0[	-	-	-	-	-	-	-	-	-	-	-	NV	T(-1, 1,1,1,1,1,1,1,1)→ error
[0]	-	-	-	-	-	-	-	-	-	-	-	V	T(0, 0,0,0,0,0,0,0,0)→ accept
[1]	[minint, 0]	-	-	-	-	-	-	-	-	-	-	NV	T(1, -1, 1,1,1,1,1,1,1)→ error
	[1, maxint]	[minin t, 0]	-	-	-	-	-	-	-	-	-	NV	T(1, 1,-1, 1,1,1,1,1,1)→ error
		[1, maxint]	[minin t, 0]	-	-	-	-	-	-	-	-	NV	T(1, 10,1,-1, 1,1,1,1,1)→ error
			[1, maxint]	[minin t, 0]	-	-	-	-	-	-	-	NV	T(1, 10,1,1,-1,1,1,1,1,)→ error
				[1, maxint]	-	-	-	-	<=300	-	<=30	V	T(1, 100,20,5, 22,0,0,0,0) → accept T_B(1, 100,20,5, 30,0,0,0,0)

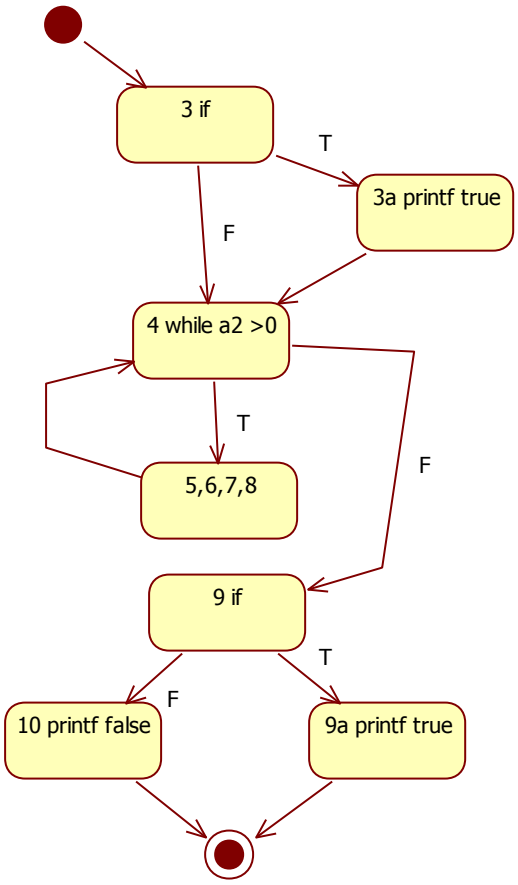
													→ accept T_B(1, 100,100,100, 22,0,0,0,0) → accept
					-	-	-	-		-	>30	V	T(1, 100,20,5, 35,0,0,0,0) → no accept T_B(1, 100,20,5, 31,0,0,0,0) → no accept
					-	-	-	-	>300	-	<=30	V	T(1, 100,150,70, 25,0,0,0,0) → no accept T_B(1, 100,100,101, 25,0,0,0,0) → no accept
					-	-	-	-		-	>30	V	T(1, 100,150,150, 35,0,0,0,0) → no accept T_B(1, 100,100,101, 31,0,0,0,0) → no accept
[2]					[minint, 0]	-	-	-	-	-	-	NV	T(2,1,1,1,1,-1,1,1,1) → error
					[1, maxint]	[minint, 0]	-	-	-	-	-	NV	T(2,1,1,1,1,1,-1,1,1) → error
						[1, maxint]	[minint, 0]	-	-	-	-	NV	T(2,1,1,1,1,1,1,-1,1) → error
							[1, maxint]	[minint, 0]	-	-	-	NV	T(2,1,1,1,1,1,1,1,-1) → error
								[1, maxint]	<= 300	<=300	<=30	V	T(2,80,50, 10, 15, 70, 70, 10, 10)→ accept
											>30	V	T(2,80,50, 10, 15, 70, 70, 10, 20)→ no accept
										>300	<=30	V	T(2,80,150, 10, 15, 170,170, 10, 10)→ no accept
											>30	V	T(2,80,150, 10, 15, 170,170, 10, 20)→ no accept
									>300	<=300	<=30	V	T(2,180,150, 10, 15, 70,70, 10, 10)→ no

													accept
											>30	V	T(2,180,150, 110, 25, 70,70, 10, 10)→ no accept
										>300	<=30	V	T(2,180,150, 110, 15, 170,170, 10, 10)→ no accept
											>30	V	T(2,180,150, 110, 25, 170,170, 110, 10)→ no accept
[3, maxint]	-	-	-	-	-	-	-			-	-	NV	T(3, 0,0,0,0,0,0,0,0,)→ error

Note1. In case of nLuggage = 0 the function should return accept, but the value of the other parameters is not defined. (In principle any value could be acceptable). This is a defect in the requirements. I assume value for other parameters should be zero.
Note2. In case of nLuggage = 1 the values of parameters for the second luggage are not defined. This is another defect in the requirements. I assume value for other parameters should be zero.

The total number of classes is $5 * 2^8$ so it is reasonable to prune the decomposition tree.
I have defined, for length width depth, two ranges only (negative, positives). Another better option is [minint, 0], [1, 300],]300, maxint] – however this increases the classes. Similarly for weight.

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**.




```
1     int f (int a1, int a2, int a3)
2           {
3           if ((a1 <= a2) && (a1 > a3)) printf("true");
4           while (a2 < 0)
5                 {
6                 a1 = a1 + 1;
7                 a2 = a2 + 1;
8                 }
9           if (a1 != a3) printf("true");
10          else printf("false");
11          }
```

Coverage type	Number of test cases needed to obtain 100% coverage	Coverage obtained with test cases defined (%)	Test cases defined
Node	1 for if line3, 1 for while, 2 for if line9, if statements are considered independently Otherwise they can be reduced to 2	100	T1(-2,-1,-22) covers 3,3a, 4, 6,7, 9, 9a T2(-2,-1,-1) covers 3, 4, 6,7,9, 10

Edge	2+1+2 considering statements independently Otherwise 2	100	T1 covers all except 3-4 and 9-10 T2 covers 3-4 9-10
Multiple condition line 3	4		T1 TT T2 TF T3 FT (2,1,0) T4 FF (2,1,3)
Loop line 4	3		T1, T2 enter many times T3, T4 no enter T5 (1,-1, 1) enter once

Path	Depends on a2.
	- 4 paths if a2>=0
	- 4 * |a2| if a2<0
	Finally 4 + 4*|a2|
	
	a2 can be as large as minint, but an algorithm can be written to generate nearly all test cases (see on the ide) - so path coverage is feasible
			Foreach (a2 from -1 to minint) produce 4 test cases:
			{(a2-1, a2, a2-2) // 3-3a, 9a
			(a2-1 , a2, -1) // 3-3a, 10
			(a2+1 , a2, 0) // 3-4, 9a
			(a2+1 , a2, 1) // 3-4, 10
			}

Write test case ID (T1, T2 ..) in the rightmost column, and test cases here

4 (1 points) – Describe shortly the Test Driven Development technique

Write one test case that fails (using requirements), write corresponding code until test case passes, repeat until all requirements are satisfied and all test cases pass

5 (1 point) –Considering GIT, what are the three project sections that it defines, and how are they used?

Git directory, working directory, staging area

6 (1 point) — In project management, what are the units of measure for duration and effort? And what is the difference between these measures?

Duration: calendar time (hours, days, weeks) – can be relative (two days) or absolute (from 12 7 to 14 7)
Effort: person * hours

As by the definition, duration measures the time needed to complete a project, effort the amount of work needed. Given the effort of a project, calendar time depends on how many people work on it (staffing profile).

7 (1 point) –What measures can be used to evaluate the quality of software?

Number of defects found over a period of time (defect rate, MTBF)
Number of defects / size
User satisfaction (using questionnaires)

(remark the question asked ‘measures’ not ‘properties’)

8 (1 point) – Describe shortly the Facade Design pattern
See slides