LABORATORIO DI INGEGNERIA DEI SISTEMI SOFTWARE

Problem analysis

In the customer's requirements the map in the html file has one form and in the parkingMap.txt file it has another one, in the following analyzes the one in the file is taken as valid. The map has some ambiguities: for example the positions of the slots are not numbered, and neither are the positions of the loading / unloading areas. The team proposes to change the map symbols like this:

```
X,  X,  X,  X,  X,  X,  X,  X,
X,  H,  0,  0,  0,  0,  I,  X,
X,  0, A1, S1, S4, A3,  0,  X,
X,  0, A2, S2, S5, A5,  X,  X,
X,  0, A3, S3, S6, A6,  0,  X,
X,  0,  0,  0,  0,  0,  O,  X,
X,  X,  X,  X,  X,  X,  X,  X
```

Position :

- **H** : Home position of the trolley. Here trolley execute Sleep operation
- **I** : position from which can Upload a car
- **O** : position from which can Download a car
- **An** : front-slot **A**rea from which can Download\Upload a car
- **Sn** : **S** where cars are parked

The trolley stops only in fixed positions, continuous positions between two map coordinates are not possible. The trolley can perform the operation (OP) of walk (W), turn left (L), turn right (R), sleep (S), load (U) and unload (D). These characters (W, L, R, U, D, S) are part of the ROUTEHI alphabet, while the lower case version of the ROUTELO alphabet (see: Logic Architecture). In order to identify each position, through coordinates (x, y), of the map without ambiguity, new concepts are introduced:

All positions are divided into:

- **ReachablePoint**

  The trolley can enter these positions. The accessible positions are those with the symbols H, 0, An, I and O. These positions are divided into:
  - **KeyPoint**

    In these positions the trolley can be stopped to carry out the loading / unloading operations or Sleep in the case of H (Home).

These positions are divided into:

- **End-point**

  In these positions the trolley could receive a new Plan. In the positions of An it can carry out loading and unloading operations. (FRONT-SLOT) In the H position it can perform the sleep operation. In the O position it can perform the unloading operation.
  - **Avaliable-Point**

    In these positions the trolley receives a new Plan after the unloading or sleep operations or in other words is becames "Avaliable" after the D or S operation.
    In the An positions after the unloading operation, the trolley receives a new Plan .
    In the O position, after the unloading operation, the trolley receives a new Plan.
    In the position of H after the sleep operation it receives a new plan if it exists otherwise it remains in this position until a new Plan is received.
  - **No-End-Point**

    The only position is I and in this position the trolley cannot receive a new Plan.
    In the I position it can perform the loading operation.
- **OnlyMovePoint**

  In these positions the trolley does not carry out operations of any kind, it can only transit.
- **UnReachablePoint**

The inaccessible positions are those are marked with X and Sn.

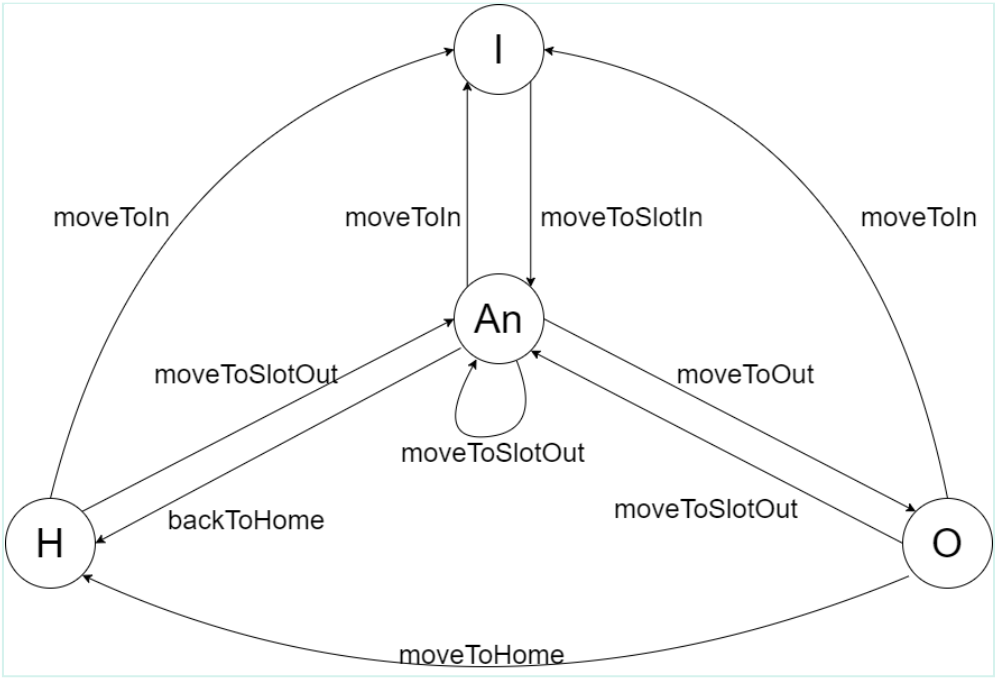
The system requires a research phase to determine the execution times of the operations like w. At the end of this phase, it may be necessary to introduce a trolley calibration every time it reaches the corner positions such as I, O and H. This calibration will exploit the object-detection functions of the virtual environment , due to this update logical architecture will be upgraded.

Table of SubPlan

			SubPlan							
		To								
		H	I	O	A1	A2	A3	A4	A5	A6
From	H		U (moveToIn)		U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)
	I				D (moveToSlotIn)	D (moveToSlotIn)	D (moveToSlotIn)	D (moveToSlotIn)	D (moveToSlotIn)	D (moveToSlotIn)
	O	N (moveToHome)	U (moveToIn)		U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)
	A1	N (backToHome)	U (moveToIn)	D (moveToOut)		U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)
	A2	N (backToHome)	U (moveToIn)	D (moveToOut)	U (moveToSlotOut)		U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)
	A3	N (backToHome)	U (moveToIn)	D (moveToOut)	U (moveToSlotOut)	U (moveToSlotOut)		U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)
	A4	N (backToHome)	U (moveToIn)	D (moveToOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)		U (moveToSlotOut)	U (moveToSlotOut)
	A5	N (backToHome)	U (moveToIn)	D (moveToOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)		U (moveToSlotOut)
	A6	N (backToHome)	U (moveToIn)	D (moveToOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	U (moveToSlotOut)	

The SubPlan table defines the path of the trolley from a **KeyPoint** from to another **KeyPoint** to and the operation to be performed upon reaching the target KeyPoint and the name of the task. The consideration that can be made is that the tasks moveToIn, moveToSlotIn, backToHome, moveToHome, moveToSlotOut, moveToOut can be considered as Subplans. To complete a customer request, for example a pick up 2 Subplans are required. This sequence is called Plan, this implies that a Plan always starts from one **Avaliable-Point** and ends at another **Avaliable-Point**. The next table defines all possible Tasks.

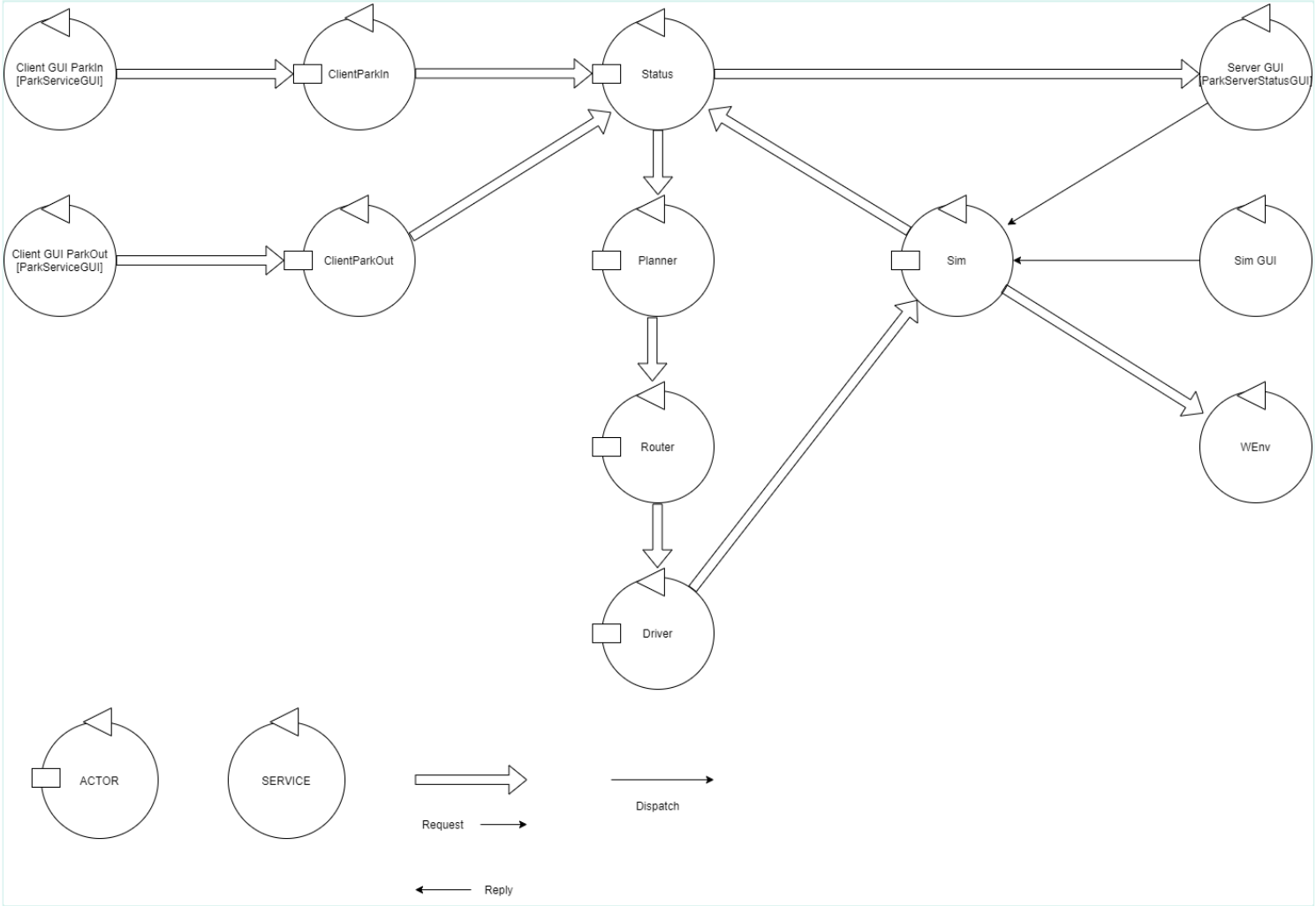
Table of SubPlan



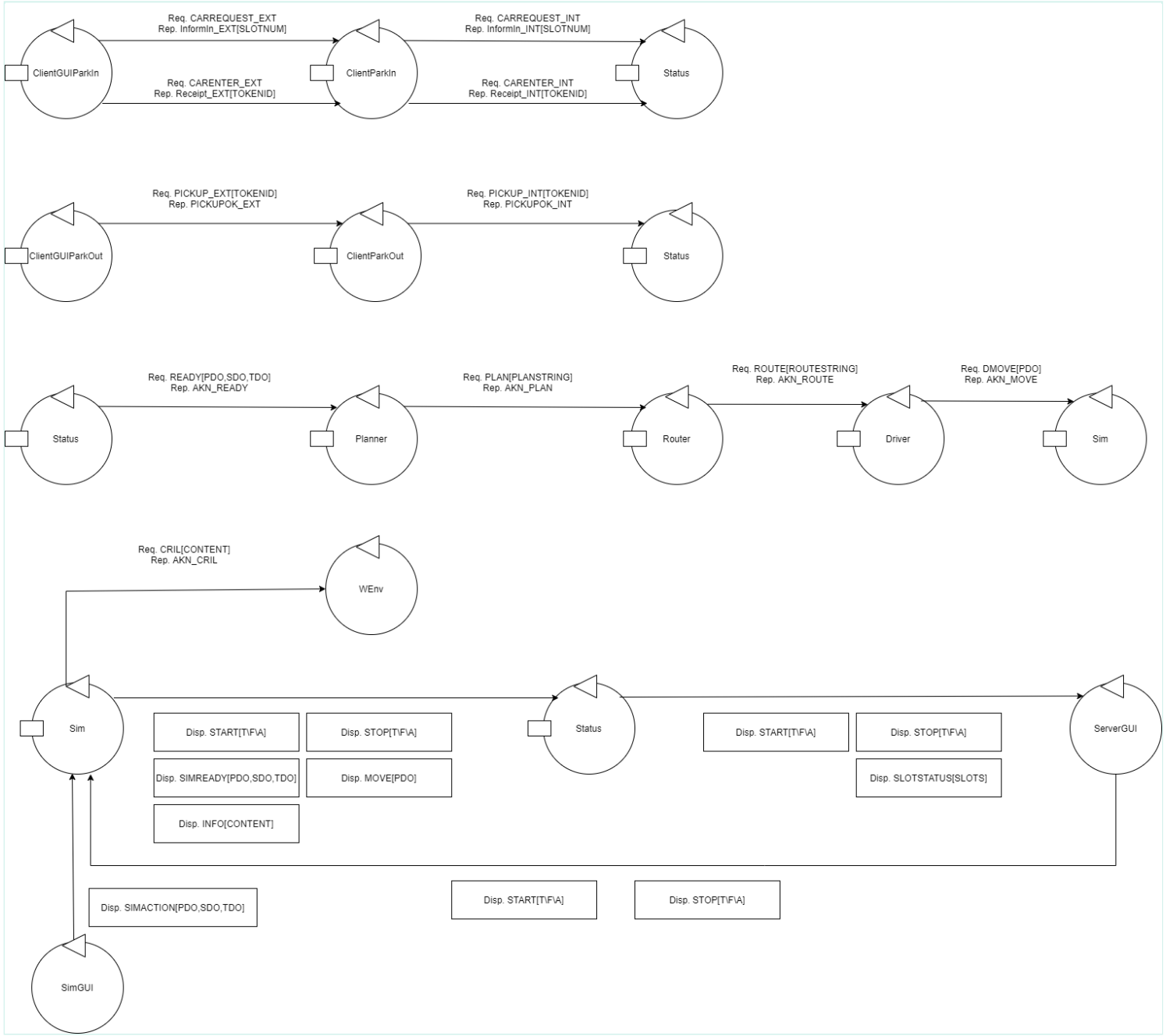
As already indicated in the requirements analysis (if SLOTNUM == 0 -> moveToHome), the existence of a requirement in contradiction with the condition that it is not possible to invoke a task that brings the trolley back to Home from any position, and whether or not it has the load, it leads at this point in the problem analysis to ignore this requirement.

Logic Architecture

Logic Architecture Main



Logic Architecture Details


Req. CARREQUEST_EXT
Rep. InformIn_EXT[SLOTNUM]
Req. CARENTER_EXT
Rep. Receipt_EXT[TOKENID]
Req. CARREQUEST_INT
Rep. InformIn_INT[SLOTNUM]
Req. CARENTER_INT
Rep. Receipt_INT[TOKENID]
ClientGUIParkIn
ClientParkIn
Status
Req. PICKUP_EXT[TOKENID]
Rep. PICKUPOK_EXT
Req. PICKUP_INT[TOKENID]
Rep. PICKUPOK_INT
ClientGUIParkOut
ClientParkOut
Status
Req. READY[PDO,SDO,TDO]
Rep. AKN_READY
Req. PLAN[PLANSTRING]
Rep. AKN_PLAN
Req. ROUTE[ROUTESTRING]
Rep. AKN_ROUTE
Req. DMOVE[PDO]
Rep. AKN_MOVE
Status
Planner
Router
Driver
Sim
Req. CRIL[CONTENT]
Rep. AKN_CRIL
WEnv
Sim
Disp. START[T\F\A]
Disp. STOP[T\F\A]
Disp. SIMREADY[PDO,SDO,TDO]
Disp. MOVE[PDO]
Disp. INFO[CONTENT]
Status
Disp. START[T\F\A]
Disp. STOP[T\F\A]
Disp. SLOTSTATUS[SLOTS]
ServerGUI
Disp. SIMACTION[PDO,SDO,TDO]
Disp. START[T\F\A]
Disp. STOP[T\F\A]
SimGUI

The Status actor contains the state of the system. It implements the logic of the acceptIN, informIN, moveToIn, moveToSlotIn, backToHome, moveToHome, acceptOut, findSlot, moveToSlotOut, moveToOut and the mock-object monitor.
It is the only actor who maintains the internal state of the system, all the others are stateless.
Status sends a message to Planner the message READY.
Status contains a ParkInQueue for managing cars in the parking phase, a SlotList for parked cars, and a ParkOutQueue for cars in the collection phase.
Status write a log file for every state change : tasks ,message, sensor , operation , position , queue, list.

After receiving the READY, the Planner actor creates the PlanString (see: Plan table Plan column) to be sent to Route.

The Route actor sends the high-level route (ROUTEHI), a sequence of characters to control the trolley [W, R, L, U, D, S], to the Driver actor.

The Driver actor translates the ROUTEHI into the low-level route (ROUTELO) composed of a sequence of w, r, l, u, d, s followed by the execution time.
The Driver actor sends the pair r: routetime to Sim at a time with a routetime delay between one sending and the next. Sending can be suspended / resumed by a SIM message.

The Sim actor contains the mock-objects of: smartthermometer (thermometer + alarm), fan, weigthsensor, outsonar, manager.
Smartthermometer: mock-object that sends a message only when Ta> TMax or Ta <TMax to Status. Includes modeling of the thermometer and alarm management.
All mock-object send messages only on change status.
All mock-objects of the actor Sim send a message to Status.
The Sim actor commands WEnv with cril language(see : VR2021.pdf)

The Sim GUI is dedicated to modifying the system state (fan, thermometer, outsensor, weightsensor) and running tests.

ServerGUI is enabled only after insert proper permission.

Table of Plans

N°	SlotFree >0	Position	Request	Plan	Test	Tasks					ParkInQueue	ParkOutQueue	OP	OutdoorArea
	F/T	I/O/H/N	I-M/M-O/H											
1	F	I	I-M	Null										
2	F	I	M-O	Null										
3	F	I	H	Null										
4	F	O	I-M	Refused										
5	F	O	M-O	O-M-U-O-D	5	acceptOUT	findSlot	moveToSlotOut	moveToOut		0	>0	D	Off
6	F	O	H	O-H	6	moveToHome					0	0	D	/
7	F	H	I-M	Refused										
8	F	H	M-O	H-M-U-O-D	8	acceptOUT	findSlot	moveToSlotOut	moveToOut		0	>0	S	Off
9	F	H	H	Null										
10	F	N	I-M	Refused										
11	F	N	M-O	N-M-U-O-D	11	acceptOUT	findSlot	moveToSlotOut	moveToOut		0	>0	D	Off
12	F	N	H	N-H	12	moveToHome					0	0	D	/
13	T	I	I-M	Null										
14	T	I	M-O	Null										
15	T	I	H	Null										
16	T	O	I-M	O-I-U-M-D	16	acceptIN	informIN	moveToIn	receipt	moveToSlotIn	>0	>=0	D	/
17	T	O	M-O	O-M-U-O-D	17	acceptOUT	findSlot	moveToSlotOut	moveToOut		0	>=0	D	Off
18	T	O	H	O-H	18	moveToHome					0	0	D	/
19	T	H	I-M	H-I-U-M-D	19	acceptIN	informIN	moveToIn	receipt	moveToSlotIn	>0	>=0	S	/
20	T	H	M-O	H-M-U-O-D	20	acceptOUT	findSlot	moveToSlotOut	moveToOut		0	>=0	S	Off
21	T	H	H	Null										
22	T	N	I-M	N-I-U-M-D	22	acceptIN	informIN	moveToIn	receipt	moveToSlotIn	>0	>=0	D	/
23	T	N	M-O	N-M-U-O-D	23	acceptOUT	findSlot	moveToSlotOut	moveToOut		0	>=0	D	Off
24	T	N	H	N-H	24	backToHome					0	0	D	/

The Plan table identifies under which conditions the Status actor sends the READY command to the Planner actor.
The 4 columns on the right , SlotFree and Position are the variables that identify the name of the Plan (PlanString) that Planner sends to the Router and the corresponding tasks that are performed.
This table has a 1: 1 correspondence with the TestsOfThePlans group of the Test Plan.
Tasks in blue are named MovingTask (see:Test Plan).

Model

First ,and incomplete, version of model is defined in a qak file (see : parkingManagerServiceModel.qak)

Test Plan

The Test Plan contains 2 tests for the execution of the main functions of the application.
The ParkIn phase and the ParkOut phase.
In both cases the trolley starts from the Home position and goes to the Home position.
The tests are carried out by some actors, respectively clientparkintest and clientparkouttest (see : parkingManagerServiceModel.qak).

ParkIn Phase
Initial conditions: weightsensor = off, all slot free.
The tasks of acceptIN, informIN, moveToIn, receipt, moveToSlotIn, backToHome are performed.
Success conditions at the end of the test: the TOKENID is generated, slot 1 is occupied, the trolley is in the HOME position.

ParkOut Phase
Initial conditions: weightsensor = off, slot 1 occupied.
The acceptOUT, findSlot, moveToSlotOut, moveToOut, moveToHome task is executed.
Success conditions at the end of the test: all slots are free, the sonar has detected a presence and before DTFREE detects the absence of the car, the trolley is in the HOME position.

mail : ugo.marchesini@studio.unibo.it