Algorithm Design

In the project the majority of the operations are simple functions, the only complexity is given by communication between them. However, there are a few algorithms that are worth mentioning besides the classic sort and search processes:

* Queue Management

The queue system is based on a FIFO rule: every taxi is inserted in the queue in availability order; the first to give availability is the first in the queue. For every request the system asks the first taxi in the queue: if the taxi driver accepts, he serves the customer and is removed from the queue; otherwise if he refuses or does not answer, he is put in the last position of the queue after a fixed time.

Pseudo-code

\* received request \*

while (!accept){

taxi\_code=getFirstTaxi();

accept=sendRequest(requestId, taxi\_code);

while(!accept && timeout!=0){

wait;

}

if (accept){

return 0;

}

else taxiNotReady(taxi\_code);

}

return -1;

* Expected Time Computation

This procedure is thought to show a user an estimated time to wait before the arrival of the requested taxi.

It asks the GPS service to calculate the distance between taxi and user, and using this data make a valuation of the remaining time.

Pseudo-code

distance=calculateDistance(taxi\_pos, user\_pos);

expected\_time=distance/average\_speed + costants;

return expected\_time;

* Expected Fare

In view of a possible future implementation, it is possible to think of an algorithm that gives the user a possible fare for the current ride. It obtains, using GPS, the actual distance between start and end points, calculates an estimated travel time and by retrieving the average cost of fuel and considering some additional factors like traffic or service it provides a plausible fare (subject to changes) to give an indication to the passenger.

Pseudo-code

distance=calculateDistance(taxi\_pos, user\_pos);

expected\_fare=(distance/avg\_taxi\_consume) x avg\_fuel\_cost + costants;

return expected\_fare;