Project Plan Document

Politecnico di Milano **A.A. 2015-2016**

Software Engineering 2 Raffaela Mirandola

Salvatore Perillo 853349 Claudio Sesto 841623

Summary

1.Function Points

The Functional Point approach is a technique that allows to evaluate the effort needed for the design and implementation of a project.
The funcionalities list has been obtained from the RASD document and for each one of them the realization complessity has been evaluated.
The functionalities has been groped in:

- **External Input**: elementary operation that elaborate input of data in the system.
- **External Output**: elementary operation that creates data used by the external enviroment
- **External Inquiry**: elementary operation that involves input and output operations.
- **Internal Logic Files**: it represents a set of homogeneous data handled by the system. In MyTaxiService application this corrisponds to the database table.
- **External Interface Files**: it represents a set of homogeneous data used by the application but created by external application. In MyTaxiService we have few devices that behave like that.

The following table shows the number of Functional Point based on funtionality and relative complexity:

Function Type	Complexity		
	Simple	**Medium**	**Complex**
External Input	3	4	6
External Output	4	5	7
External Inquiry	3	4	6
Internal Logic File	7	10	15
External Interface File	5	7	10

External Input 45
The Application can handle the following inputs:

1. *Login/Logout*: this are very simple operations, so 2*3=6 FPs.
2. *Become a registered user/Taxi Driver* :also this is very simple operation,so 2*3= 6 FPs.
3. *Create new Request/Reservation*:due to the fact that a bit of elaboration to parse the address, we consider this medium. We can use 2*4=8 FPs.
4. *Delete a Reservation*:different entities are involved here (all data of the reservation plus theuser) so can be considered medium. 1*4=4 FPs.
5. *See User History*: this is a simple DB query that involves one entity,1*3=3 FPs.
6. *Change Profile Informations*: one entity involved so simple function, 1*3=3 FPs

7. *Change Taxi driver state*: also here, only the taxi is involved so simple function 1*3=3 FPs.
8. *Taxi Driver Response*: in this case taxi, user and Q/R manager are involved in case of positive response, so this is a complex function, 1*6=6 FPs
9. *New Technician Feature*: is a very complex function because the whole system is actually modified 1*6 FPs.

External Output 12

1. *Taxi Request Allocation*: this task is complex because involve different entities plus the algorith described in the DD. 1*7=7 FPs.
2. *Taxi Queue Allocation*: this task is not so complex, few entities are involved and the process here is also described in the DD. So we can consider the medium weight 1*5=5 Fps.

External Inquiry 8

1. *See new Feature Information*: this is an easy task; all informations are provided by the technicians and can be easily seen from the web site; 1*3=3 FPs
2. *Check Reservation Info*: this task can be consider medium because different entities are involved 1*5=5 FPs

Internal logic files 91

1. *List of the entities of the databases*: as we have already described in the ER diagram present in the RASD, we have 13 simplex entities (such as Taxi ,Taxi Driver,Users,Ride ,Request…). So we have 13*7=91 FPs .

External Interface files 25

1. *Maps Data of the API integrated in the Q/R Manager*: This is a complex case due to the fact that there is the need to interface our Q/R manager with the API maps. 1*10=10FPs.
2. *GPS Coordinate*: This is the “interface“ that communicate between the GPS and the device of the taxi driver. In this case there is only a simple transfer of data. 1*5=5 FPs.
3. *WorkStation Communication:* In this case we have more than a simple transfer of data but a really integration of a new piece of software. 1*10=10 FPs.

The Final Result of our Function Points 181 FPs.
UFP=181 FPs.

2.COCOMO

If we want to use the COCOMO approach, we need to pass from FP to SLOC: in order to do that we have to multiply the FPs by a factor depending on the programming language chosen. For this implementation we choose J2EE because it provides developing tools that furnish high modularity and as we have described in the RASD (to allow technicians to develop new features without changing lot of the already implemented system) and so, ad described in the table at http://www.qsm.com/resources/function-point-languages-table, we multiply by 46

181 FPs * 46 =8326 SLOC

For the Effort Equation we consider a project with the Cost and Scale Drivers that produce an EAF of 1.00 and exponent of 1.0997.
EAF:Effort Adjustment Factor derived from Cost Drivers.
E: Exponent derived from Scale Drivers.

$$effort = 2.94 * EAF * (KSLOC)^{E}$$

so we obtain

effort=2,94*(1,0)*(8,326)^1,0997= 30,24 Person/Months

Now we estimate the duration of the whole project in month by applying the formula
This time we consider a different exponent E=0.3179

$$Duration = 3{,}67 * (effort)^{E} = 10{,}85$$

The last point of COCOMO is to calculate the number of people for his project

#people=effort/Duration
#people=30.24/10.85=3 people

3.Schedule of Tasks

To create an efficient schedule about our tasks we have listed the main parts of our assignment for each document:
RASD:

- Introduction [T0]
- Interfaces [T1]
- Goals [T2]
- Assumption formulations [T3]
- Requirements [T4]
- Constraints [T5]
- Alloy [T6]

DD:

- Introduction [T7]
- Architectual design [T8]
- Algorithms design [T9]
- Requirements Traceability [T10]

Code inspection:

- getRequestedEJBModuleOrLibrary analysis [T11]
- skipJar analysis [T12]
- addModule [T13]

Test plan:

- introduction [T14]
- Integration Stategy [T15]
- Individual Step and Test description [T16]
- Tools and Test Equipment Required [T17]
- Program Stubs and Test Data Required [T18]

Project plan:

- Function points [T19]
- COCOMO [T20]
- Schedule of task [T21]
- Resource Allocation [T22]
- project risks [T23]

Another important point is to define the scheduling for each deadlines:

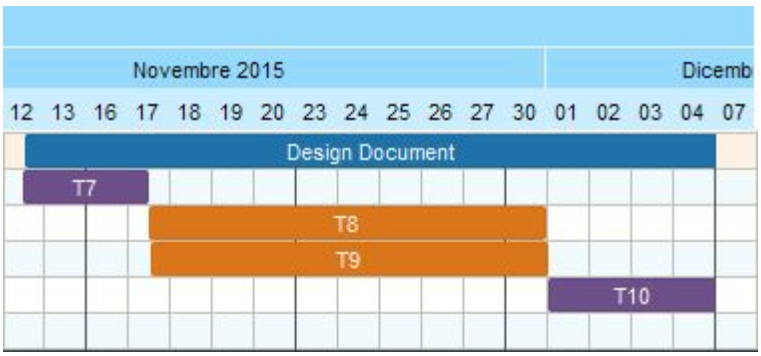
- RASD 15/10/2015 - 6/11/2015
- DD 12/11/2015 - 4/12/2015
- Code Inspection 16/12/2015 - 05/01/2016
- Testing document 08/01/2016 - 21/01/2016
- Project Plan 21/01/2016 - 02/02/2016

Gantt's diagram of schedules

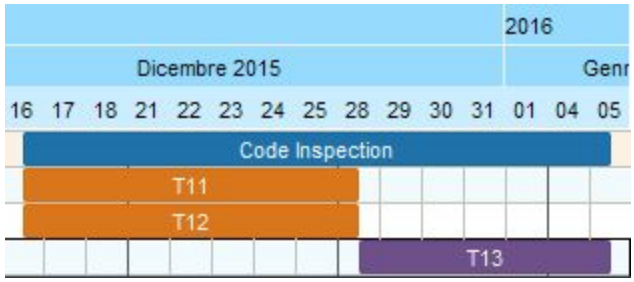
This project consist in more than one assignment, and so in more deadlines. Below is listed the scheduling for each assignment, all described with a Gantt's diagram: these diagrams are just a guideline about the deadlines of each assignment's task. Of course they are estimated approximately(without weekends) so if some task are completed before the estimation, the new one can start: this will avoid bad time management and goldplating risks.

RASD

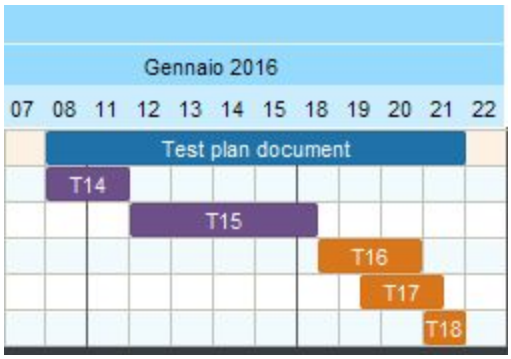
DD



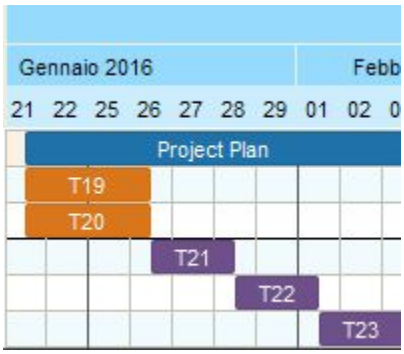
Code Inspection



Test plan document



Project Plan document



NB: The task in purple involve all members of the group so necessarly must be accomplished sequentially; the orange ones are the individual task so we can exploit parallelism to optimize the workload and respect the deadlines.

4.Resource Allocation

To assign a person to a specific task, we have listed below all tasks with the group member that will be work to that specific task. Some tasks requires the efforts of all group members because there is the need to find a common solution before continue with the development of the subsequent tasks:

T00: Claudio/Salvatore
T01: Claudio/Salvatore
T02: Claudio/Salvatore
T03: Claudio/Salvatore
T04: Claudio/Salvatore
T05: Claudio/Salvatore
T06: Salvatore
T07: Claudio/Salvatore
T08: Salvatore
T09: Salvatore
T10: Claudio/Salvatore
T11: Salvatore
T12: Claudio
T13: Claudio/Salvatore
T14: Claudio/Salvatore
T15: Claudio/Salvatore
T16: Claudio
T17: Salvatore
T18: Salvatore
T19: Salvatore
T20: Claudio
T21: Claudio/Salvatore
T22: Claudio/Salvatore
T23: Claudio/Salvatore

Actually we mainly worked together by also supervisioning each other on how the work was proceeding.

5.Project's Risks

For this project we have identified some of the main risks that can be found in our path:

Bad Time Managment:

This is a very common risk that can happen for many reasons:

- People focus their efforts on some points for too much time.
- People start to work on assignment too late.
- No optimal tasks division

This risk can become very dangerous for the project, because in most of the cases in which it occurs, people notice this risk too late and the only correction action is to increase the effort for the assignment.

People illness:

People are people so they can get sick.

This can be a common risk and the only recovery action is (if the person is not able to work) to increase the weight of other group members.

Capability Shortfalls:

This is an uncommon risk due to the fact that this project is assigned to us under certain circumstances that ensure us to avoid it. But if for some reasons this happens, the recovery action is to change the assigned task to an other group member.

Requirements Misunderstanding:

This risk can happen for different reasons:

- Unclear Assignments
- Students Misunderstanding

It can be very dangerous becasuse if a group complete the assignment, at the moment of the validation they will understand to have worked and produced something that is not the project requested. This risk doesn't have a really recovery actions, the only way is prevent this by asking previously if something is unclear.

Goldplating:

This risk in uncommon in general but can happen; it consists in working on already met requirements by adding stuff (tipically useless and not required) without focusing on the new requirements that must be met.

Here there is no specific action to solve the risk but can be avoided if the programmers are supervised.

6:Hours of Work

Claudio Sesto 7h

Salvatore Perillo 7h