POLITECNICO DI MILANO
Computer Science and Engineering
Software Engineering 2 Project

MyTaxiService

Project Reporting Document

Authors: Greta Ghiotti

Raffaele Malvermi

Mirco Mutti

Version 1.0

Date 2/02/2016

Reference Professor: Mirandola Raffaela

Index

1. Introduction

1.1 Revision History

Version 1.0
Version released on 2/02/2016

1.2 Purpose and Scope

This Project Reporting Document aims to define the Project plan of MyTaxiService system. Our project plan follows three steps:

- cost estimation using algorithmic methods (Function Points + COCOMO II);
- task scheduling and resource allocation (using Gantt diagrams);
- risk identification, evaluation and management;

The idea is to simulate the planning phase at the beginning of the process, in order to let the committee know about costs, deadlines and risks affecting the entire project.

The second chapter of the document is also devoted to inform the rest of the team members (we suppose that the document should be written with point of view of the project manager) about what concerns the project schedule and the roles of each team member.

1.3 List of Reference Documents

We report the list of documents we have considered as landmarks for our Project Reporting Document and also the documents we have referred-to:

- MyTaxiService Requirements Analysis and Specification Document (RASD);
- MyTaxiService Design Document (DD);
- "Project Management" and "Cost Estimation" sets of slides form Software Engineering 2 course;
- "Project Plan Assignment" document;
- "COCOMO II – Model Definition Manual" (http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf)

2. Cost Estimation

1.1 Function Points

In the following section we report the result of the function point identification: we have included a specific table for each function point type and then the total FP count.

In order to determine the complexity and the weight of each FP we have considered the classification provided by "COCOMO II – Model Definition Manual" (more details in the section 1.4) that we report in the tables below.

Table 2. FP Counting Weights

For Internal Logical Files and External Interface Files

	Data Elements		
Record Elements	**1 - 19**	**20 - 50**	**51+**
1	Low	Low	Avg.
2 - 5	Low	Avg.	High
6+	Avg.	High	High

For External Output and External Inquiry

	Data Elements		
File Types	**1 - 5**	**6 - 19**	**20+**
0 or 1	Low	Low	Avg.
2 - 3	Low	Avg.	High
4+	Avg.	High	High

For External Input

	Data Elements		
File Types	**1 - 4**	**5 - 15**	**16+**
0 or 1	Low	Low	Avg.
2 - 3	Low	Avg.	High
3+	Avg.	High	High

Table 3. UFP Complexity Weights

	Complexity-Weight		
Function Type	**Low**	**Average**	**High**
Internal Logical Files	7	10	15
External Interfaces Files	5	7	10
External Inputs	3	4	6
External Outputs	4	5	7
External Inquiries	3	4	6

The Data Elements count is related to the number of fields of the data structures (defined in case of FIL and EIF, modified in case of EO, EI and EIQ).

The Record Elements count is related to the number of homogenous subsets, recognizable by the User, of each File Type.

The File Type count is related to the number of FIL and EIF involved in the EI, EO and EIQ.

External Interface Files

In the External Interface Files set we have considered the entities of the External Database.

FP(ELFs) = 1x5 = 5

ELF	Complexity	FP
CityTaxiDriver	Low	5

Internal Logical Files

We have considered as Internal Logical Files the data structures corresponding to the Internal Database entities (User, MyTaxiDriver, Taxi Company and Zone) and the data structures managed in the Request Manager such as the different types of Request (Request, Reservation, Deletion, Availability) and the structure related to a sharing taxi. The combination of number of Data Elements and Record Elements of each ILF doesn't justify a higher than Low complexity.

FP(ILFs) = 7x9 = 63

ILFs	Complexity	FP
User	Low	7
MyTaxiDriver	Low	7
Request	Low	7
Reservation	Low	7
Deletion	Low	7
Availability	Low	7
TaxiCompany	Low	7
SharingTaxi	Low	7
Zone	Low	7

External Inputs

We have included in the External Input set every possible message from the users (both general User and MyTaxiDriver) to the Server.
We have considered a High complexity for the Reservation External Input because it involves a high number of ILFs in combination with a relevant number of Data Elements.
We have classified as Average complexity the SignUp TaxiDrivers, Request and Departure declaration which have a smaller number of File Type involved.

FP(EIs) = 5x3 + 3x4 + 1x6 = 33

EIs	Complexity	FP
Login User	Low	3
Login TaxiDriver	Low	3
SignUp User	Low	3
SignUp MyTaxiDriver	Avg	4
DeleteReservation	Low	3
Request	Avg	4
Reservation	High	6
Available	Low	3
Departure	Avg	4

External Outputs

We have included in the External Output set every message from the Server to the two type of users. These interactions involves a small number of Internal Logical Files, for this reason are classified as Low complexity.

FP(EOs) = 4x3 = 12

Eos	Complexity	FP
ReceivedRequest	Low	4
RouteMessage	Low	4
SharingFee	Low	4

External Inquiries

We have considered as External Inquiries every bidirectional interaction between Server and Users (ShowTaxi) and between Server and Google services. We have classified as Low the interaction with the Google Services because are only composed by a simple function calls with a return value.

FP (EIQs) = 3x7 = 21

EIQs	Complexity	FP
ShowTaxi	Low	3
ComputeRoute	Low	3
FIndCoordinates	Low	3
ComputeWaitingTime	Low	3
CreateShareRoute	Low	3
CheckSameDirections	Low	3
FindZones	Low	3

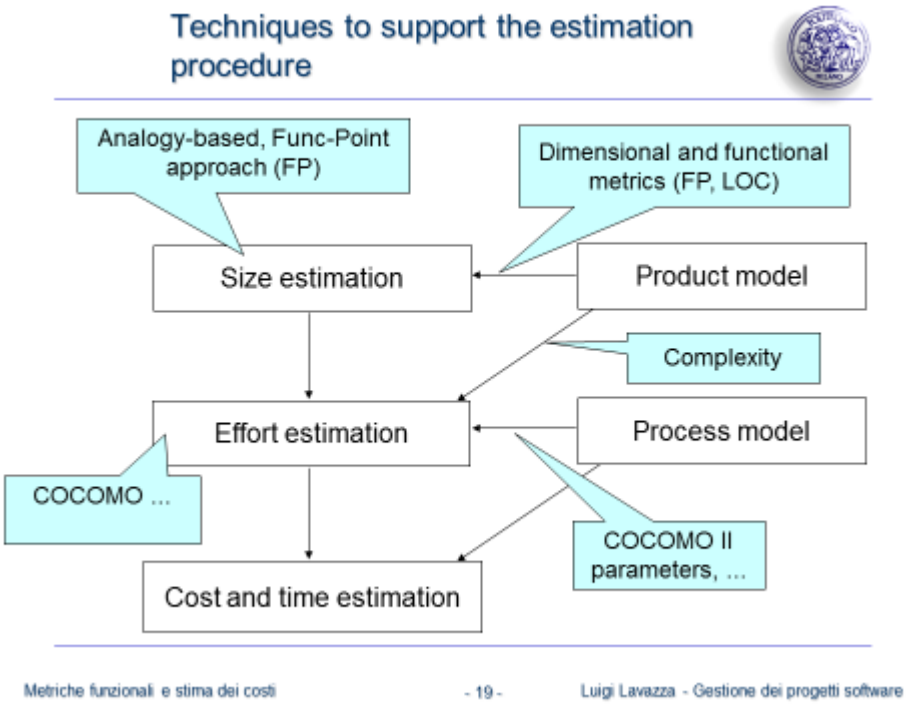
Unadjusted Function Points

Then, considering the weighted sum of each FP identified, we have obtained the Unadjusted Function Point count.

UFPs = FP(ILFs)+FP(ELFs)+FP(EIs)+FP(EOs)+FP(EIQs) = 63+5+33+21+12 = 134

1.2 COCOMO

The estimation provided by the Unadjusted Function Points (UFPs) tells us about the size of the system to-be-developed. Indeed, this method is based only on the functionalities that the system offers.
If we want to estimate the effort of the project, that is more relevant from the point of view of the committee and the project manager, we must adjust the UFPs taking into account other factors, related to product complexity, personnel experience and capabilities and technology constraints.
COCOMO II satisfy our needs, because it computes the effort and the duration of the project by using two equations, having the properties above as parameters.



From "Cost Estimation" lesson of Software Engineering 2 course

In this document, we use a Web-based tool to make the computation of the Effort Equation and the Duration Equation simpler and quicker (provided by the "Center for Systems and Software Engineering", University of Southern California).
In this paragraph we aim to show the rationale about the choices made on parameters and the outcome of the computation.

Software size

The first input required by the effort equation is the size of the software (in Source Lines Of Code). We can compute it by multiplying the amount of UFPs by a language-dependent constant (AVC).
Since during the project we have never specified a language, we assume the system will be developed in J2EE (AVC = 46); so, at the end we obtain 6164 SLOC.
One thing to be notice is the lack of reused code: we have made the estimation in the worst case, to obtain the maximum duration for the project (in the DD we explain some possibilities to reuse code, using patterns and architecture styles).

Software Scale Drivers

To evaluate the value for each scale driver, we have used the table below, taken from the "COCOMO II – Model Definition Manual" (see paragraph 3.1)

Table 10. Scale Factor Values, SF_i, for COCOMO II Models

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprecedented	largely unprecedented	somewhat unprecedented	generally familiar	largely familiar	thoroughly familiar
SF_i:	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
SF_i:	5.07	4.05	3.04	2.03	1.01	0.00
RESL	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
SF_i:	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
SF_i:	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	The estimated Equivalent Process Maturity Level (EPML) or					
	SW-CMM Level 1 Lower	SW-CMM Level 1 Upper	SW-CMM Level 2	SW-CMM Level 3	SW-CMM Level 4	SW-CMM Level 5
SF_i:	7.80	6.24	4.68	3.12	1.56	0.00

Product Cost Drivers

Our system has to be reliable and able to recover losses easily; on the other hand, high reliability as for financial systems is not required. We conclude that the software reliability parameter has to be set to Nominal.

Table 17. RELY Cost Driver

RELY Descriptors:	slight inconven-ience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.82	0.92	1.00	1.10	1.26	n/a

The database size parameter is set to Nominal; all the explanations can be read in the paragraph "Deployment View" of the DD.

Table 18. DATA Cost Driver

DATA* Descriptors		Testing DB bytes/Pgm SLOC < 10	10 ≤ D/P < 100	100 ≤ D/P < 1000	D/P ≥ 1000	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.90	1.00	1.14	1.28	n/a

The product complexity is set to Nominal, because we think that code should reflect the description made in the "Nominal" row of the table n.19, "Component Complexity Ratings Levels".

Nominal	Mostly simple nesting. Some intermodule control. Decision tables. Simple callbacks or message passing, including middleware-supported distributed processing	Use of standard math and statistical routines. Basic matrix/vector operations.	I/O processing includes device selection, status checking and error processing.	Multi-file input and single file output. Simple structural changes, simple edits. Complex COTS-DB queries, updates.	Simple use of widget set.

If we look at the DD, in the paragraphs concerning the architecture and the possible patterns used in the system (in order 2.2, 2.7, 2.8), we can set the parameter "Developed for Reusability" to High. The design phase has been one of the most important during the process of our system.

Table 21. RUSE Cost Driver

RUSE Descriptors:		none	across project	across program	across product line	across multiple product lines
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.95	1.00	1.07	1.15	1.24

Since the project was based on making documentation, we conclude we can set the "Documentation Match to Lifecycle Needs" parameter to Very High.

Table 22. DOCU Cost Driver

DOCU Descriptors:	Many life-cycle needs uncovered	Some life-cycle needs uncovered.	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.81	0.91	1.00	1.11	1.23	n/a

Personnel Cost Drivers

Since this project made us more expert about what concerns the first phases of the process (in particular, requirements elicitation and design), we can conclude that the analyst capability can be set to "Very High"

Table 26. ACAP Cost Driver

ACAP Descriptors:	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.42	1.19	1.00	0.85	0.71	n/a

Our experience as programmers can be compared with the Nominal value of the table, since we have worked with programming languages for almost 1 year.

Table 27. PCAP Cost Driver

PCAP Descriptors	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.34	1.15	1.00	0.88	0.76	n/a

We assume that our team will remain the same during the entire process. So, we set the parameter to the High value.

Table 28. PCON Cost Driver

PCON Descriptors:	48% / year	24% / year	12% / year	6% / year	3% / year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.29	1.12	1.00	0.90	0.81	

We set the APEX parameter to Nominal because our experience as programmers is based on making applications. This experience started improving almost one year ago.

Table 29. APEX Cost Driver

APEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 years	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.22	1.10	1.00	0.88	0.81	n/a

Concerning the knowledge of the platform, we mean the experience in using Databases and Object-Oriented programming languages. We can set the parameter to Nominal, because we had few occasions to improve our technical knowledge.

Table 30. PLEX Cost Driver

PLEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.19	1.09	1.00	0.91	0.85	n/a

Since we make this estimation assuming J2EE as language used to implement the software, we have to set the parameter to Very Low because we consider a 2-month experience.

Table 31. LTEX Cost Driver

LTEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.20	1.09	1.00	0.91	0.84	

Platform Cost Drivers

Our system has to satisfy strict requirements about its availability (see the RASD), so we can set the "Time Constraints" parameter to Extra High

Table 23. TIME Cost Driver

TIME Descriptors:			≤ 50% use of available execution time	70% use of available execution time	85% use of available execution time	95% use of available execution time
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	n/a	1.00	1.11	1.29	1.63

Following the rationale explained in the DD (see "Deployment View"), we think there are no strict constraints about the system storage and so we can set the "Storage constraint" parameter to the Nominal value

Table 24. STOR Cost Driver

STOR Descriptors:			≤ 50% use of available storage	70% use of available storage	85% use of available storage	95% use of available storage
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	n/a	1.00	1.05	1.17	1.46

Following our design phase, we expect possible changes concerning the storage and computational power of our system. However, this changes should be less frequent than what happens in a normal software. So we set the parameter to the Low value.

Table 25. PVOL Cost Driver

PVOL Descriptors:		Major change every 12 mo.; Minor change every 1 mo.	Major: 6 mo.; Minor: 2 wk.	Major: 2 mo.;Minor: 1 wk.	Major: 2 wk.;Minor: 2 days	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.87	1.00	1.15	1.30	n/a

Project Cost Drivers

All the tools used during the process are listed in the previous documents (RASD, DD, ITPD), but we want to underline the fact that those are used to support only the lifecycle of the software. No advanced tools have been used during the process, so the parameter can be set to Nominal.

Table 32. TOOL Cost Driver

TOOL Descriptors	edit, code, debug	simple, frontend, backend CASE, little integration	basic life-cycle tools, moderately integrated	strong, mature life-cycle tools, moderately integrated	strong, mature, proactive life-cycle tools, well integrated with processes, methods, reuse	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.17	1.09	1.00	0.90	0.78	n/a

Our team is made up of members living in the same area, working always in the same sites during the process. We can conclude the team is fully collocated and uses interactive multimedia as communication channels.

Table 33. SITE Cost Driver

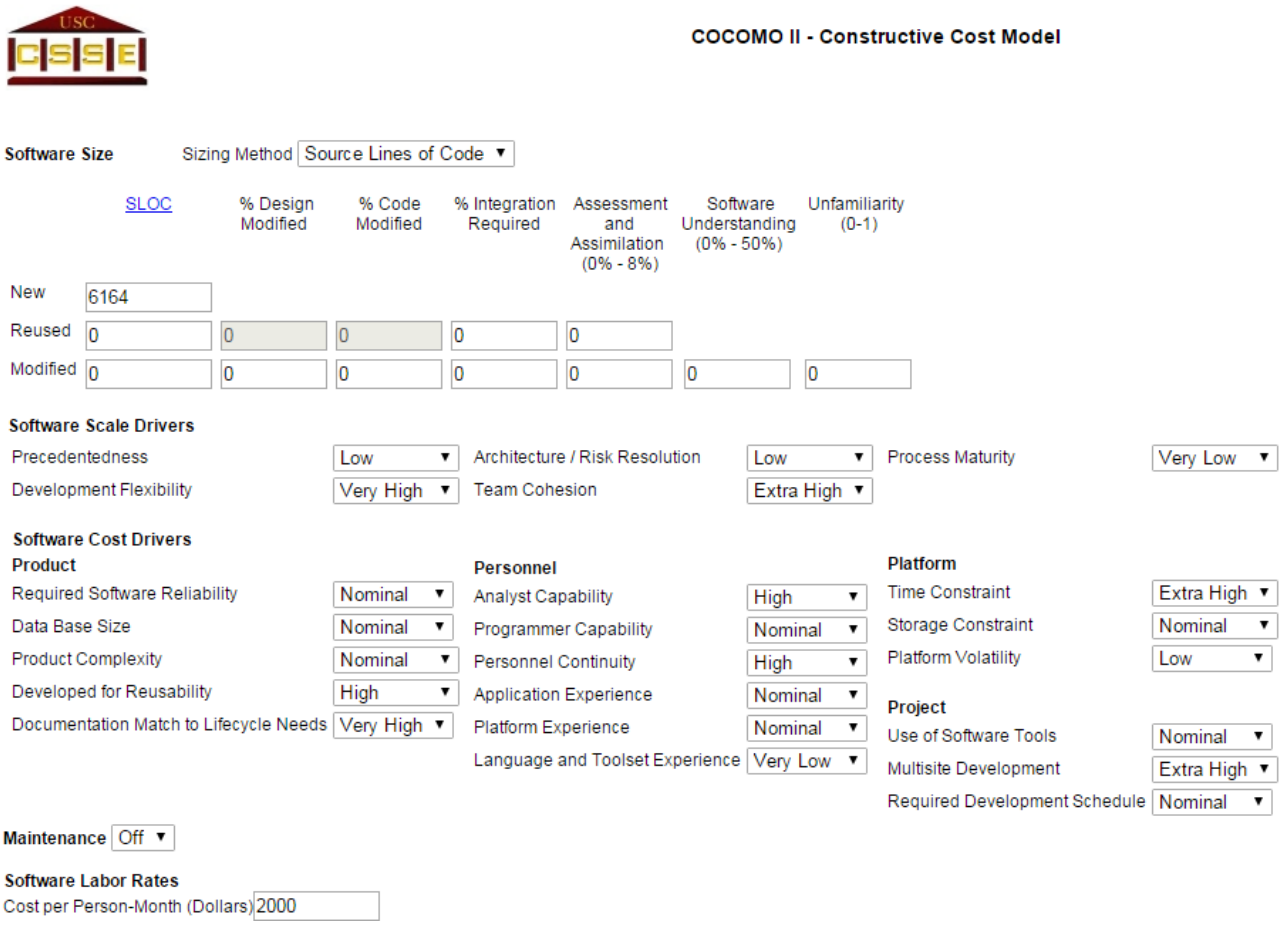
SITE: Collocation Descriptors:	Inter-national	Multi-city and Multi-company	Multi-city or Multi-company	Same city or metro. area	Same building or complex	Fully collocated
SITE: Communications Descriptors:	Some phone, mail	Individual phone, FAX	Narrow band email	Wideband electronic communication.	Wideband elect. comm., occasional video conf.	Interactive multimedia
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.22	1.09	1.00	0.93	0.86	0.80

The schedule has never been compressed or stretched to satisfy the deadlines of the project. We set to Nominal the SCED parameter.

Table 34. SCED Cost Driver

SCED Descriptors	75% of nominal	85% of nominal	100% of nominal	130% of nominal	160% of nominal	
Rating Level	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multiplier	1.43	1.14	1.00	1.00	1.00	n/a

In the image below we show how the Web-based tool looks like and how all the inputs must be inserted to make the effort computation.
We must focus now on the "Maintenance" parameter: in this estimation we assume that the process mandated to us will finish with the release and the execution of the system, while the maintenance will be at the expense of the city administration (that is the committee of the project). Adding also this factor, the effort will increase.



Once all the required inputs are inserted, the tool computes the effort and the duration (Schedule) of the project. It also shows the distribution of these two costs among the phases. One important result is the distribution of the staff members, useful to make a possible schedule according to this estimation.

Looking at the outcome, our software needs 11.2 months to be completed with a staff of 3 members: this is not true in every phases, there are some requiring less people on staff, but we consider the Average Staff size, computed by dividing the Effort by the Schedule value.

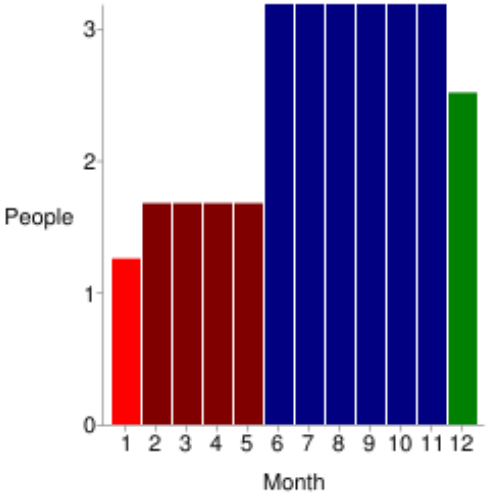
Results

Software Development (Elaboration and Construction)

Effort = 29.3 Person-months
Schedule = 11.2 Months
Cost = $58524

Total Equivalent Size = 6164 SLOC

Staffing Profile



Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	1.8	1.4	1.3	$3511
Elaboration	7.0	4.2	1.7	$14046
Construction	22.2	7.0	3.2	$44478
Transition	3.5	1.4	2.5	$7023

Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.2	0.8	2.2	0.5
Environment/CM	0.2	0.6	1.1	0.2
Requirements	0.7	1.3	1.8	0.1
Design	0.3	2.5	3.6	0.1
Implementation	0.1	0.9	7.6	0.7
Assessment	0.1	0.7	5.3	0.8
Deployment	0.1	0.2	0.7	1.1

3. Project Schedule

3.1 Task Identification and Schedule

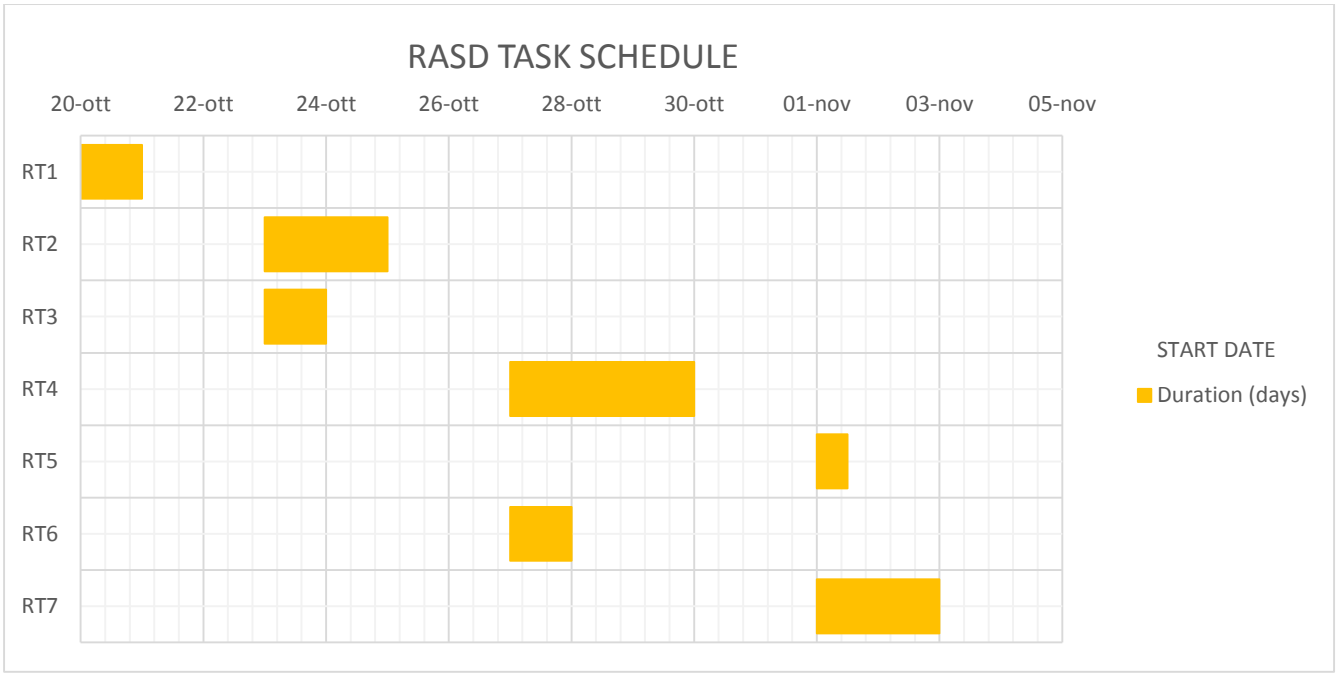
In the following chapter we report a table for each project macro-phase including the corresponding set of task, the effort and duration required and then the dependency between the different tasks (a task cannot be initiated before the completion of the tasks reported in the dependency field).
For each macro-phase we also report a gantt chart describing how we have organized the schedule of each task in the phase.
First we consider only the tasks that we have actually managed in order to release the project documentation; then we report also a virtual case (in the section "real" project schedule) in which we have taken into account also the implementation, testing and release phases (which are not part of the project).

In order to estimate the effort and duration of the implementation and testing phases we have considered the results obtained with the COCOMO II model (more details of the output at page 14).
We have chosen to use COCOMO II consistently to the cost estimation presented in the previous chapter, but we know that other estimation models exist, probably better than this one (without the strong assumptions of COCOMO II).
In particular the Testing effort estimation seems to be significantly different with respect to the application of the Brooks "golden rule" ("*The Mythical Man-Month",* Frederick P. Brooks Jr.) which suggests a 60:40 ratio between Implementation and Testing efforts.

Requirements Analysis and Specification Document

First we analyze the RASD task partition and the detailed schedule we have followed in order to release the Document into the deadline.

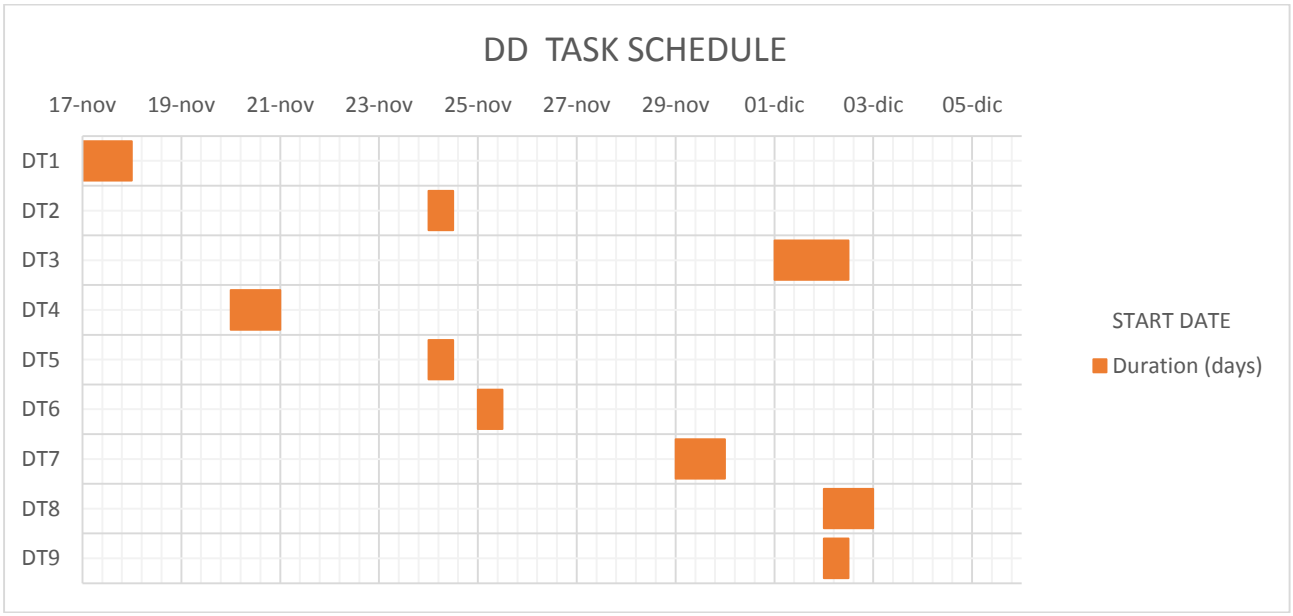
Task ID	Description	Effort(person-day)	Duration(days)	Dependencies
RT1	Context analysis	3	1	
RT2	Requirements elicitation	4	2	RT1
RT3	Scenarios	2	1	RT1
RT4	Use cases and sequence diagrams	5	3	RT2,RT3
RT5	State charts and class diagrams	0,5	0,5	RT4
RT6	Alloy modelling	2	1	RT2
RT7	User interface mockups	2	2	RT4



Design Document

Now we analyze the DD task partition and schedule. We have omitted the obvious dependency with the RASD macro-phase.

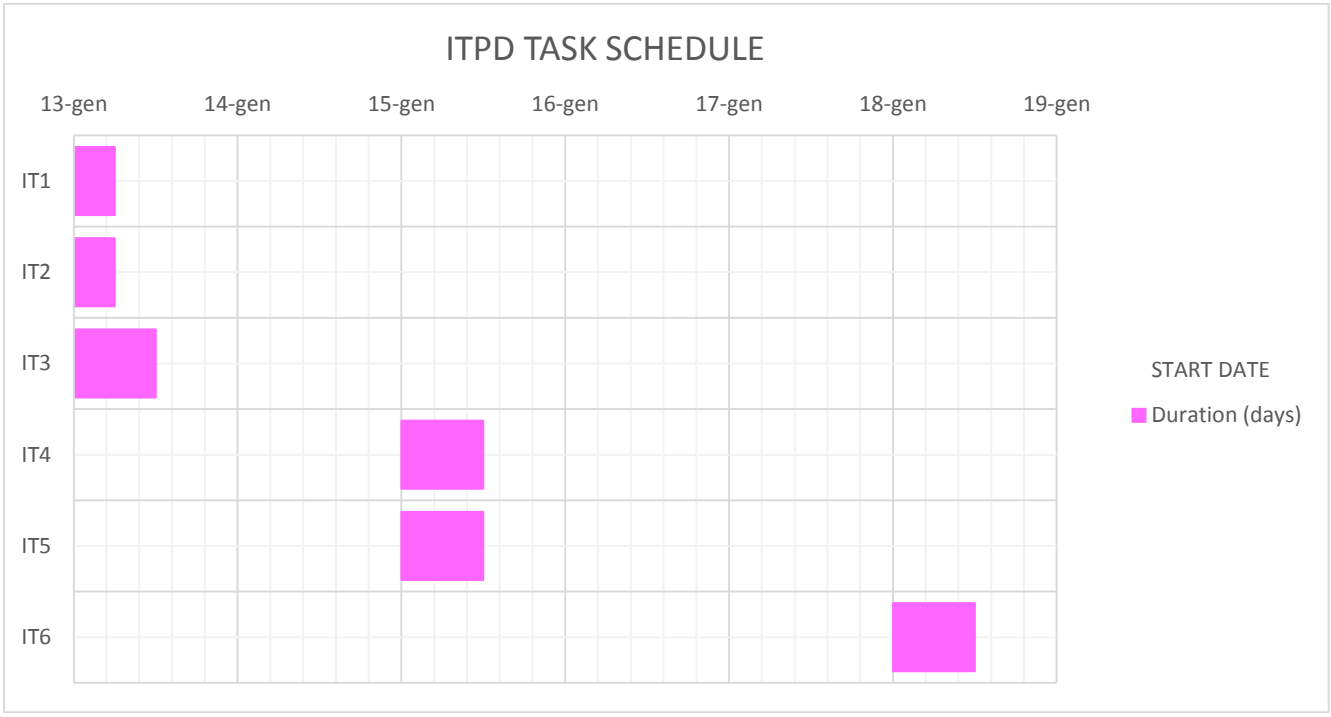
Task ID	Description	Effort(person-day)	Duration(days)	Dependencies
DT1	High level architecture	3	1	
DT2	Deployment view	1	0,5	DT1
DT3	Runtime view	3	1,5	DT4
DT4	Component interfaces	3	1	DT1
DT5	Styles and patterns	1,5	0,5	DT1
DT6	Other decision	0,5	0,5	DT5
DT7	Algorithm	3	1	DT4
DT8	Dynamic user interface mockups	1	1	
DT9	Requirements traceability	0,5	0,5	DT1



Integration Test Plan Document

Now we analyze the ITPD task partition and schedule. We have omitted the obvious dependency with the precedent macro-phases.

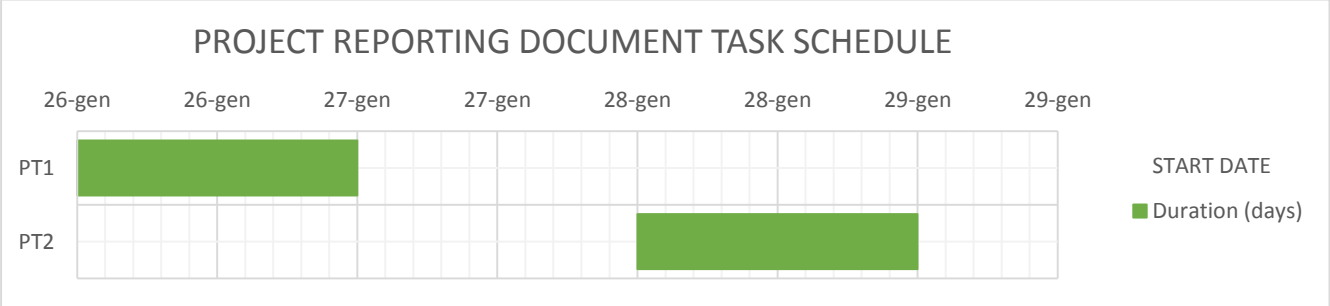
Task ID	Description	Effort(person-day)	Duration (days)	Dependencies
IT1	Entry criteria	0,75	0,25	
IT2	Elements to be integrated	0,5	0,25	
IT3	Integration testing strategy	1	0,5	
IT4	Sequence of integration	1,5	0,5	IT1,IT2,IT3
IT5	Test description	1	0,5	IT4
IT6	Tools, stubs and drivers	0,5	0,5	IT4



Project Reporting Document

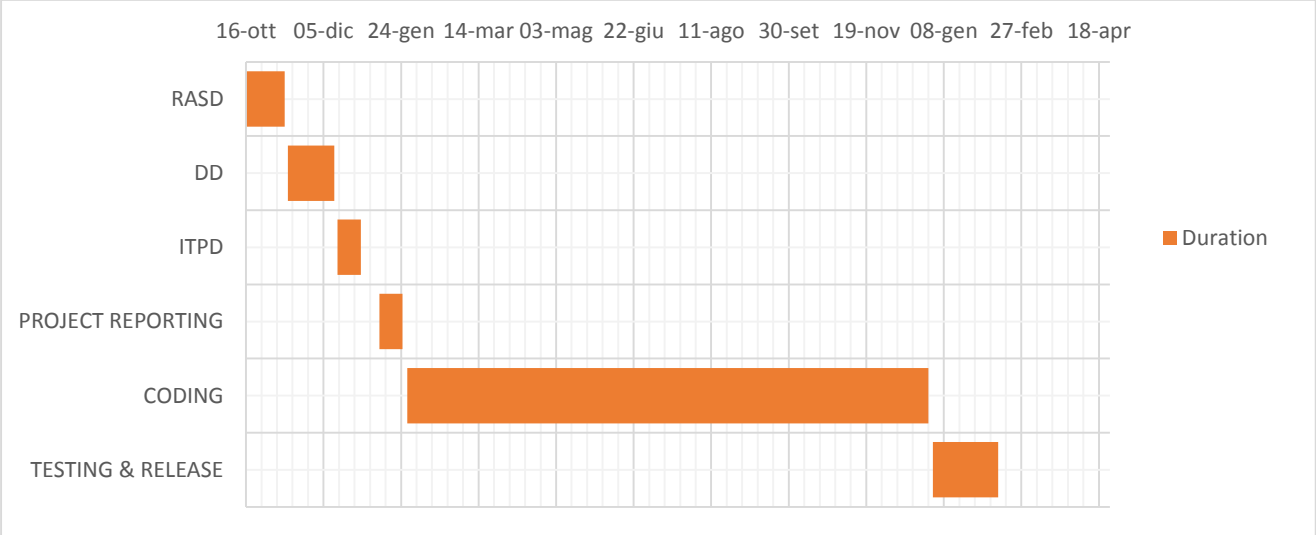
Now we analyze the Project Reporting task partition and schedule. We have omitted the obvious dependency with the precedent macro-phases.

Task ID	Description	Effort(person-day)	Duration (days)	Dependencies
PT1	Cost estimation	3	1	
PT2	Project plan redaction	3	1	PT1



"Real" project schedule

In the following chart, we report the schedule that we have obtained taking into account also the estimated effort of the Coding and Testing&Release phases (respectively "Elaboration" plus "Construction" and "Transition" in the output table at page 14).
We have considered the real duration for the other phases, corresponding to thirty days for the RASD and DD phases, fifteen days for ITPD and Project Reporting.
Since we have used COCOMO II model to estimate the effort, we suppose to follow a waterfall process to develop the system (that is an assumption of the model).



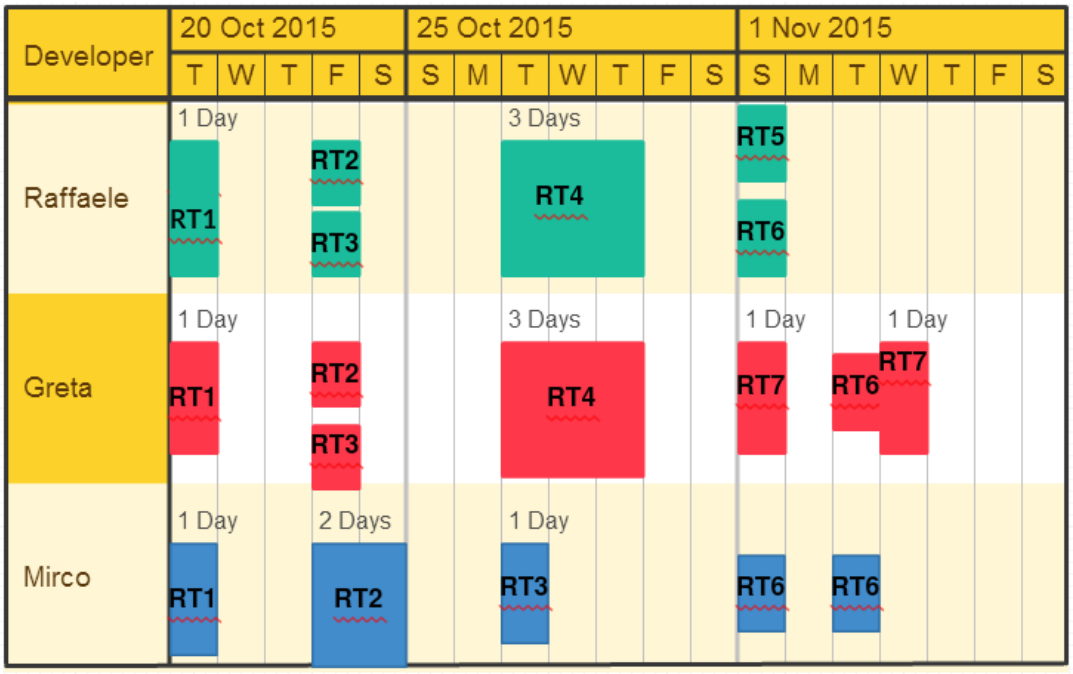
3.2 Resources Allocation

In the following paragraph, we report a Gantt chart for each project macro-phase, showing the number of hours (expressed in day and part of day) that each member of the group employed in the different phases of the MyTaxiService project.

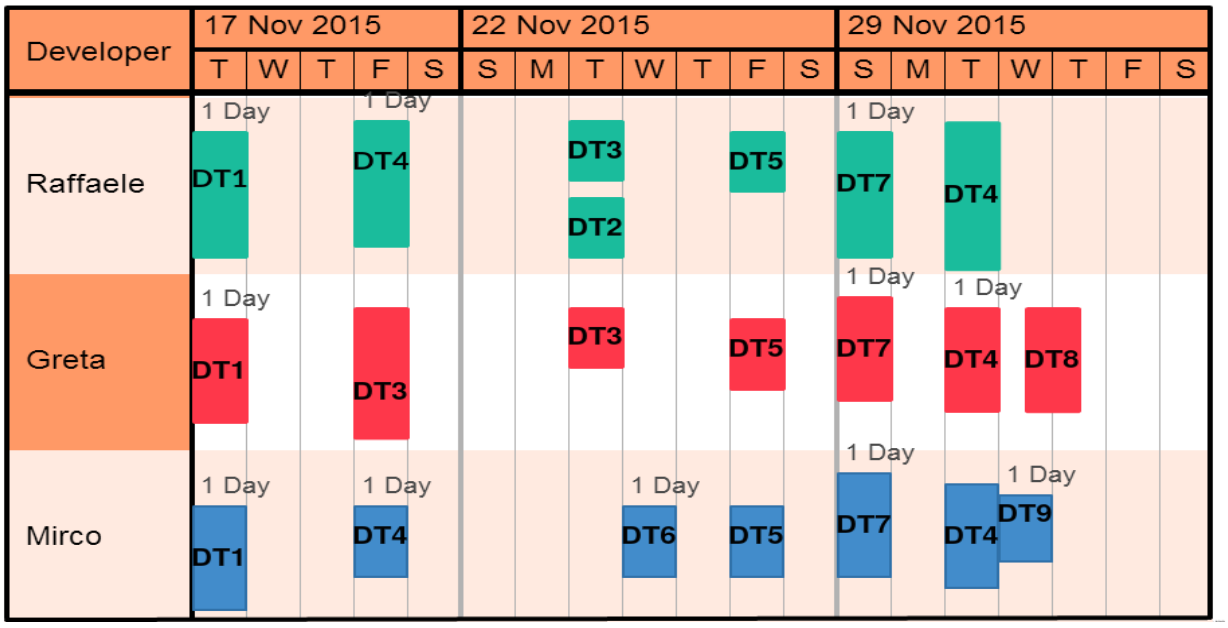
A brief explanation on how to read the charts:

- The first row reports the date corresponding to the first cell of the field below
- In the second row we have the days of the week (S=Sunday, M=Monday, T=Tuesday, W=Wednesday, T=Thursday, F=Friday, S=Saturday)
- On each task block we report the task identifier
- On the top of each task block we report the duration of the task (in case of less than one day we omit the information)

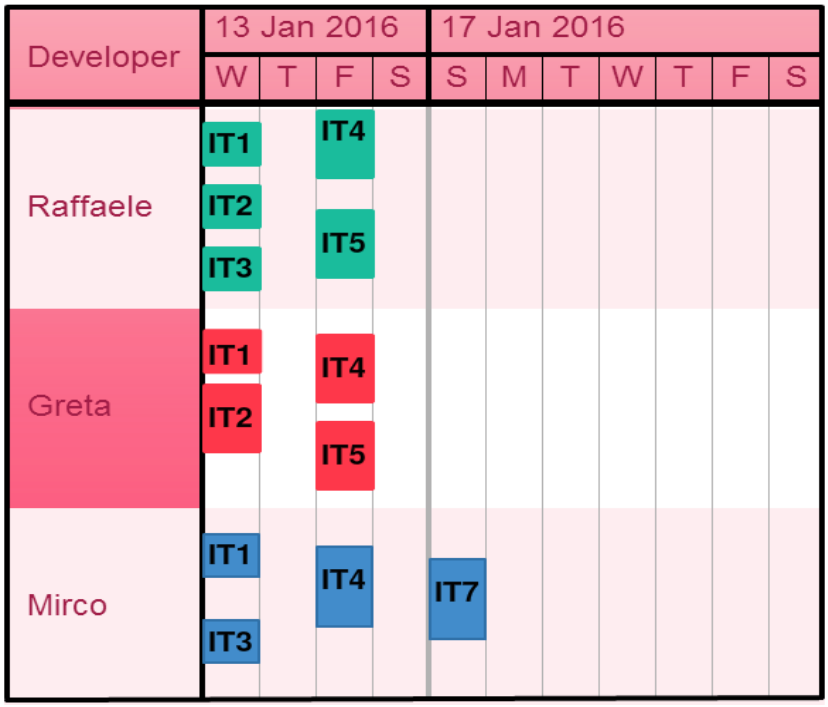
Requirements Analysis and Specification Document



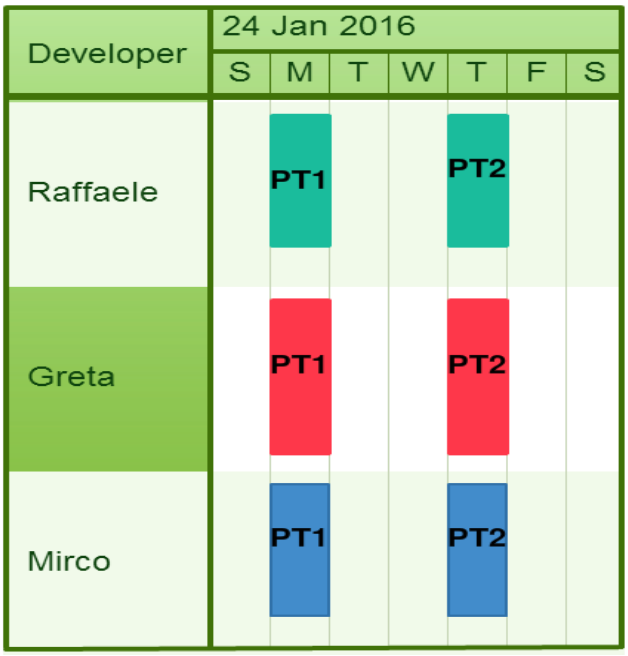
Design Document



Integration Test Plan Document



Project Reporting Document



4. Risks Analysis

The last paragraph of this document focuses on the identification of possible risks and their management.

The first table lists all the risks by classifying them into 3 classes: Project, Technical and Business.
Project risks affect any aspects concerning the schedule and the resource allocation (staff problems and changes in project specifications, as instance);
Technical risks concern the low phases of the process (development, testing, release and maintenance of the product);
Business risks are related to marketing aspects and budgets.

Each risk is evaluated with respect to the likelihood of its occurrences and its effects on the project: in particular, the effects are estimated using the values below (starting from the most critical to the less one).

Catastrophic: the occurrence of the risk doesn't let the project finish successfully;
Serious: the occurrence of the risk causes a huge delay in the project schedule (increase of the duration);
Reasonable: the occurrence of the risk is easily to manage, even if it causes delay in the schedule;
Low: the occurrence of the risk is expected and managed without changes in the project schedule and resource allocation;

ID	RISK	TYPE	PROBABILITY	EFFECTS
R1	insufficient staff skills during development phase	Project Risk	High	Catastrophic
R2	Underestimated number of staff members	Project Risk	Low	Serious
R3	Impossibility to fulfill the development within the deadlines	Project Risk	Moderate	Reasonable
R4	Difficulty in organizing a proper schedule	Project Risk	Moderate	Reasonable
R5	Late addition of new functionalities required by the customer	Project Risk	Moderate	Serious
R6	Change in the management of customer organization	Project Risk	Low	Reasonable
R7	Difficulty in exploiting pre generated code	Technical Risk	Moderate	Reasonable
R8	Sub-estimation of required storage resources	Technical Risk	Low	Serious
R9	Sub-estimation of required computational and network resources	Technical Risk	High	Catastrophic

R10	Some aspects of the designed architecture resulting impossible to implement	Technical Risk	Moderate	Serious
R11	Failure occurrences during integration testing	Technical Risk	High	Low
R12	Final product resulting not innovative w.r.t. the state of art	Business Risk	High	Low
R13	Insufficient budget w.r.t. the estimated cost of the entire project	Business Risk	Moderate	Catastrophic

Few considerations about the table above:

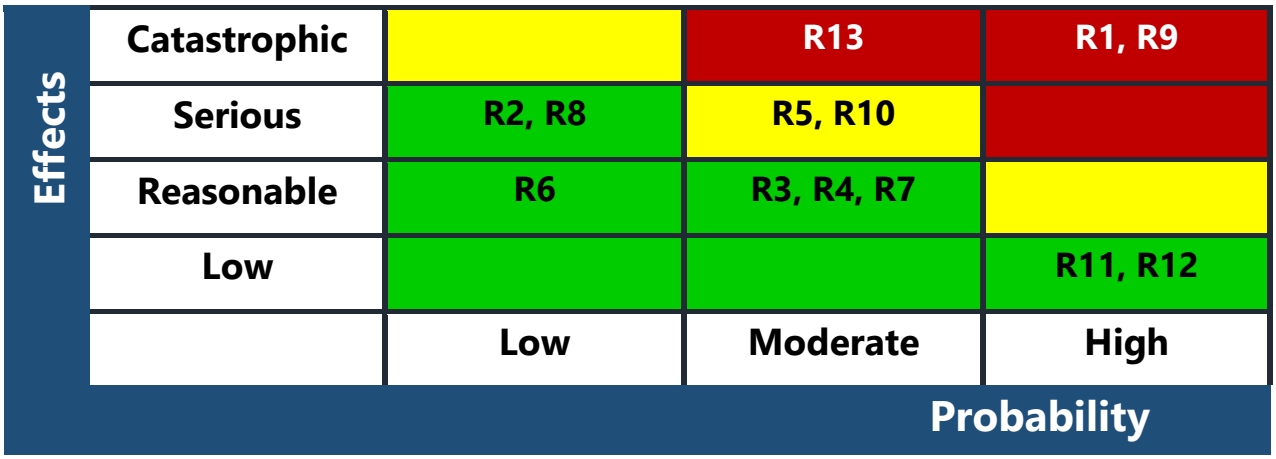
- We have evaluated risk R2 with low probability because we used COCOMO II to estimate the number of members; since this algorithmic method fails only in 20% of the cases, we can state that occurrences of this risk are very unlikely;
- We have evaluated risk R8 with low probability because we have explained in the DD (see paragraph "Deployment View") that the size of the stored data will reach an upper-bound value;
- We have assumed that all the risks which we can't control should be very likely (a sort of 'pessimistic' point of view).

The presence of risks needs the fulfillment of predefined strategies to make the process continue without problems. The table below defines all the strategies, showing the relative risks and the classes to which they belong.
Indeed, these can be 'proactive' or 'reactive', based on the execution approach: proactive strategies are taken into account before the occurrence of the risk, in order to prevent it; on the other hand, reactive strategies are carried out only when the occurrence of a risk is detected.

ID	RISK	STRATEGY	TYPE
R1	insufficient staff skills during development phase	Two alternatives: - Consider a formative period to raise the staff members skills; - Recruit additional personnel with required experience;	Reactive
R2	Underestimated number of staff members	Estimation of the required personnel resources using COCOMO II	Proactive
R3	Impossibility to fulfill the development within the deadlines	Estimation of the project duration using COCOMO II; in case of too optimistic estimation, re-negotiation of the deadlines with the customer	Proactive/ Reactive
R5	Late addition of new functionalities required by the customer	Re-negotiate budget and deadlines with the customer	Reactive

R6	Change in the management of customer organization	In case of new requests, re-negotiate budget and deadlines with the customer	Reactive
R7	Difficulty in exploiting pre generated code	Estimation of the project effort with COCOMO II, considering the worst case: entire project made of new LOC	Proactive
R8	Sub-estimation of required storage resources	Estimation of the required database size during the first phases of the project; during the system execution, control of the current situation	Proactive/ Reactive
R9	Sub-estimation of required computational and network resources	Estimation of the required computational power during the first phases of the project; during the system execution, control of the throughput and load	Proactive/ Reactive
R10	Some aspects of the designed architecture resulting impossible to implement	Repetition of the design phase in order to make feasible the implementation	Reactive
R13	Insufficient budget w.r.t. the estimated cost of the entire project	Ask to the customer the magnitude of the budget in order to check the feasibility of the proposed solution	Proactive

Once we have identified all the possible risks, we can consider their priority looking at the probability and the criticality. With the term priority we mean the urgency in managing a certain risk: if the risk is very likely and causes very critical issues, it will be evaluated with a high priority.
The diagram below shows the priority of all the risks listed in the first table.



Number of hours

We have spent 15 hours per person to redact the Project Reporting Document.

Greta Ghiotti: 15 hours

Raffaele Malvermi: 15 hours

Mirco Mutti: 15 hours