A.Y. 2015/2016

Software Engineering 2 : “My taxi service”

Project Plan Document

Version 1.0

Ivan Antozzi(790962) , Riccardo Giambona(788904)

2 Febbraio 2016

Sommario

1. Function Points Estimation

1.1Complexity Weight

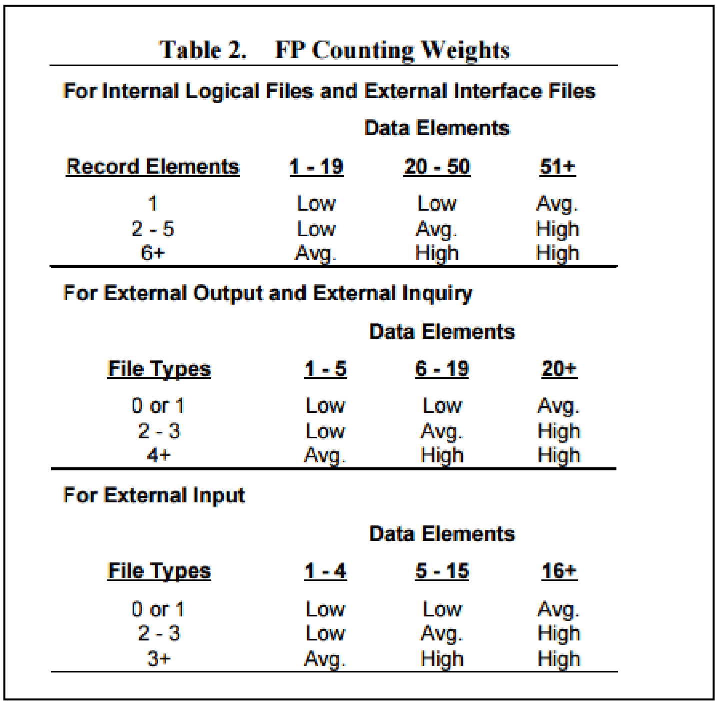


Table 2. FP Counting Weights

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

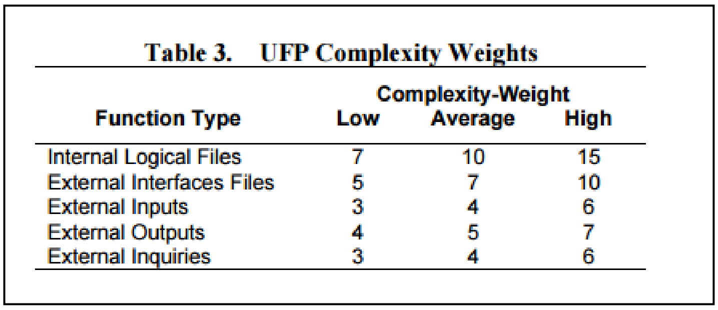
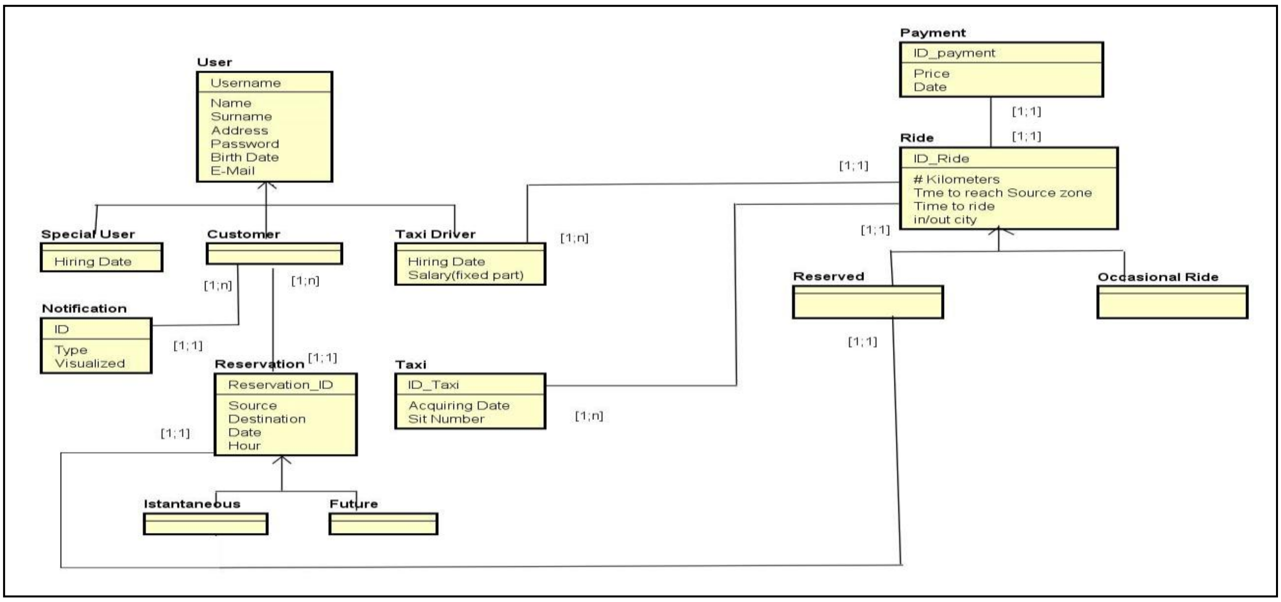


Table 3. UFP Complexity Weights

	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

1.2 FP Extimation

1.2.1 Internal Logic Files



We inserted the ER schema in the document because we wanted to see what the application stored and consequently to identify all the internal logic files used. First we start to identify the RETs. This identification should be done on the logical schema of the database, but since we didn't make the development we will suppose that each entity of the ER schema is a table of the logic schema. Of course, this is a pessimistic approximation, because in the logical schema, there will probably be less tables than entities. After that we proceed to group all these RETs in ILFs. In this section we don't consider ILFs for clients, since they are thin and they don't store anything locally.

ILF Name	RETs names	#RETs Names	Data Elements	#Data Elements	Complexity	FP
User	User, Special User, Customer, Taxi Driver	4	Name, Surname, Address, Password, E Mail, Username, Birth Date, Hiring Date(x2), Salary	10	Low	7
Notification	Notification	1	ID, Type, Visualized	3	Low	7
Reservation	Reservation, Instantaneous Res, Future Res	3	ID, Source, Destination, Date, Hour	4	Low	7
Taxi	Taxi	1	ID, Sit Number, Acquiring Date	3	Low	7
Ride	Ride, Reserved Ride, Occasional Ride, Payment	4	ID Payment, ID Ride, Price, Date, #Kilometers, Time to reach, time to ride, in/out city	8	Low	7
Total						35

1.2.2 External Logic Files

1.2.2.1 Backend Application

The Backend application hasn't external logic files, since it doesn't interact with external services.

1.2.2.2 All Clients

This section applies to all client that interfaces with the backend app via API interfaces. Below we will summarize the RETs of the external interface file used by each client. The RETs are the entities that are included in the responses of each API interface call and this is the reason why we consider only one ELF that includes all the RETs.

1.2.2.3 Mobile Client App

RETs names	#RETs Names	Data Elements	#Data Elements	Complexity	FP
Response	1	Response Mex,Type of response	2	Low	5
Reservation	1	ID, Source, Destination, Date, Hour,Type	6	Low	5
UserInfo	1	Name, Surname, Address, Password, E Mail, Username, Birth Date	7	Low	5
Notification	1	ID, Type, Visualized	3	Low	5
Total					20

1.2.2.4 Taxi Driver Mobile App

RETs names	#RETs Names	Data Elements	#Data Elements	Complexity	FP
Response	1	Response Mex,Type of response	2	Low	5
Ride	1	ID Ride, Price, Date,#Kilometers, Time to reach, time to ride, in/out city	7	Low	5
UserInfo	1	Name, Surname, Address, Password, E Mail, Username, Birth Date	7	Low	5
Zone	1	ID	1	Low	5
Total					20

1.2.2.5 Web Client

RETs names	#RETs Names	Data Elements	#Data Elements	Complexity	FP
Response	1	Response Mex,Type of response	2	Low	5
Reservation	1	ID, Source, Destination, Date, Hour,Type	6	Low	5
EmployeeData	1	Name, Surname,	9	Low	5

		Address, Password, E Mail, Username, Birth Date, Hiring Date, Salary			
WorkerStats	1	KM/Month,H/Month	2	Low	5
ServiceStats	1	#Rides,#TotalRides	2	Low	5
Total					25

1.2.3 External Inputs

For all the clients since they are thin clients, number of file types is zero, because no ILFs are modified locally in the client to elaborate the input. The data attributes are the input fields or control used by the client to acquire the input from the user.

1.2.3.1 Mobile Client App

Name	**#Data Attributes(Input Fields)**	**Complexity**	**FP**
Login	2	Low	3
Logout	1	Low	3
Istantaneous Taxi Ride	5	Low	3
Future Taxi Ride	5	Low	3
Cancel Taxi Ride	2	Low	3
Update Taxi Ride	5	Low	3
Registration	11	Low	3
Change Password	4	Low	3
Change Profile Info	6	Low	3
Total			27

1.2.3.2 Taxi Driver Mobile App

Name	**#Data Attributes(Input Fields)**	**Complexity**	**FP**
Login	2	Low	3
Logout	1	Low	3
Accept request	1	Low	3
Deny request	1	Low	3

Set State	2	Low	3
Accept Occ Ride	2	Low	3
Choose Taxi	2	Low	3
PickUp Client	1	Low	3
Arrived to Destination	1	Low	3
Arrived to client	1	Low	3
See Ride Details	1	Low	3
Total			33

1.2.3.3 Web Client

Name	**#Data Attributes(Input Fields)**	**Complexity**	**FP**
Login	2	Low	3
Logout	1	Low	3
Istantaneous Taxi Ride	5	Low	3
Future Taxi Ride	5	Low	3
Cancel Taxi Ride	2	Low	3
Update Taxi Ride	5	Low	3
Registration	11	Low	3
Change Password	4	Low	3
Change Profile Info	6	Low	3
Total			27

1.2.3.4 Backend

For the backend app the inputs are all the API interfaces exposed to clients. The complexity is based on observing which ILFs are modified/read elaborating the input and how many DataElements are modified/read of those ILFs.Please note that for example LogIn and Logout the ILFs involved is NONE because the username and password are read to elaborate the output and not the input.

Name	**ILFs Involved**	**# ILFs Involved**	**Number of Data Elements**	**Complexity**	**FP**
Login	NONE	1	2	Low	3
Logout	NONE	0	2	Low	3
Istantaneous	User,	2	6	Average	4

Taxi Ride	Reservation				
Future Taxi Ride	User, Reservation	2	6	Average	4
Cancel Reserved Taxi Ride	User, Reservation	2	5	Average	4
Update Reserved Taxi Ride	User, Reservation	2	5	Average	4
Registration	User	1	7	Low	3
Change Password	User	1	1	Low	3
Change Profile Info	User	1	5	Low	3
Accept request	NONE	0	2	Low	3
Deny request	NONE	0	2	Low	3
Set State	NONE	0	2	Low	3
Accept Occ Ride	Ride	1	3	Low	3
Choose Taxi	NONE	0	1	Low	3
PickUp Client	Ride	1	2	Low	3
Arrived to Destination	Ride	1	3	Low	3
Arrived to client	Ride	1	3	Low	3
Update GPS Location	NONE	0	2	Low	3
Get Info	NONE	0	3	Low	3
Total					88

1.2.4 External Inquiries

For all the Clients, the number of specified data attributes are attributes inside the ELF, which is the API interface.

1.2.4.1 Mobile Client App

Name	**#Data Attributes**	**Complexity**	**FP**
View Previous Reservations	4	Low	3
View Personal Info	4	Low	3
View Notifications	2	Low	3
Total			9

1.2.4.2 Taxi Driver Mobile App

Name	#Data Attributes	Complexity	FP
View Past Served Rides	4	Low	3
Total			3

1.2.4.3 Web Client

Name	#Data Attributes	Complexity	FP
View Previous Reservations	4	Low	3
View Personal Info	4	Low	3
View Notifications	2	Low	3
View Employees Data	3	Low	
View Service Stats	2	Low	3
View Worker Stats	4	Low	3
Total			15

1.2.4.4 Backend

The Backend Application doesn't have any external inquiries, since it doesn't make inquiries to itself.

1.2.5 External Outputs

For all the clients we don't consider each sngle output because, since the clients are thin, they don't elaborate much data and the main output types would be message boxes of confirmations or error information, so, since these outputs are extremely simple, we approximate the sum of clients' FP points to zero.

1.2.5.1 Backend

Notification Output

This is the output of the backend app when it sends notification to clients

Name	ILFs Involved	# ILFs Involved	Number of Data Elements	Complexity	FP
Notify Taxi Driver New Reservation	User, Reservation	2	5	Average	5
Notify Changed Zone	User	1	1	Low	4
Notify The Client Reservation Accepted	User, Reservation, Ride	3	2	Low	4
Notify Client	User,	3	2	Low	4

Taxi Arrived	Reservation, Ride				
Total					17

Api Manager Responses

In this section we identified all the API Manager interfaces that produce output (that is read by the clients in ELF)intended as the elaboration of one or more ILFs. Please note that many ILFs involved are set to NONE because in those cases ILFs are used to elaborate the input and not the output (see External Input section) and the Number of Data Elements are set to 2 except getInfo,because in all those api calls only the response mex and response type is returned.

Name	**ILFs Involved**	**# ILFs Involved**	**Number of Data Elements**	**Complexity**	**FP**
Login	User	1	2	Low	3
Logout	User	1	2	Low	3
Istantaneous Taxi Ride	NONE	0	2	Average	4
Future Taxi Ride	NONE	2	2	Average	4
Cancel Reserved Taxi Ride	NONE	2	2	Average	4
Update Reserved Taxi Ride	NONE	2	2	Average	4
Registration	NONE	1	2	Low	3
Change Password	NONE	1	2	Low	3
Change Profile Info	NONE	1	2	Low	3
Accept request	NONE	0	2	Low	3
Deny request	NONE	0	2	Low	3
Set State	NONE	0	2	Low	3
Accept Occ Ride	NONE	1	2	Low	3
Choose Taxi	NONE	0	2	Low	3
PickUp Client	NONE	1	2	Low	3
Arrived to Destination	NONE	1	2	Low	3

Arrived to client	NONE	1	2	Low	3
Update GPS Location	NONE	0	2	Low	3
Get Info	All	5	28	High	7
Total					65

Total External Output Backend = Total Notification + Total Api = 65+17 =82

1.3 Total System FP

In this section we sum all the Fps calculated before, taking count that in the design document we decided to support three different OS for the mobile apps and therefore all the things related to the mobile apps are multiplied by three.

Name	**Total**
Backend	205
Mobile Client App	56
Mobile Taxi Driver App	59
Web Client	67
Total	**617**

Total = Backend+3(mobile client app) + 3(mobile taxi driver app) + web client.

1.4 Lines of Code

Name	**Programming Language**	**Multiplying Factor**	**Lines of Code**
Backend	C#	64	205x64 = 13120
Mobile Client App(WPhone)	C#	64	56x64 = 3584
Mobile Client App(Android)	Java	53	56x53 = 2968
Mobile Client App (iOS)	Objective C	55	56x55 = 3080
Taxi Driver Mobile App(WPhone)	C#	64	59x64=3776
Taxi Driver Mobile App(Android)	Java	53	59x53 = 3127

Taxi Driver Mobile App(iOS)	Objective C	55	59x55= 3245
Web App	ASP.net	20	67x20 = 1340
Total			**34240**

1.5 Cocomo II

1.5.1 Scale Drivers

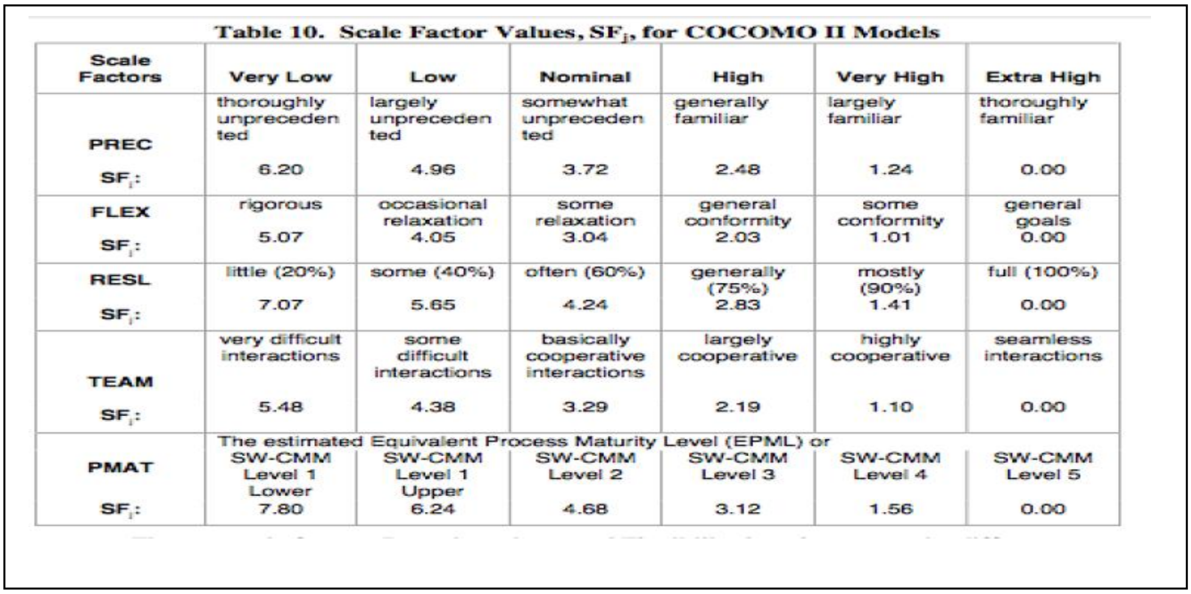


Table 10. Scale Factor Values, SF_j, for COCOMO II Models

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC SF_j:	thoroughly unprecedented 6.20	largely unprecedented 4.96	somewhat unprecedented 3.72	generally familiar 2.48	largely familiar 1.24	thoroughly familiar 0.00
FLEX SF_j:	rigorous 5.07	occasional relaxation 4.05	some relaxation 3.04	general conformity 2.03	some conformity 1.01	general goals 0.00
RESL SF_j:	little (20%) 7.07	some (40%) 5.65	often (60%) 4.24	generally (75%) 2.83	mostly (90%) 1.41	full (100%) 0.00
TEAM SF_j:	very difficult interactions 5.48	some difficult interactions 4.38	basically cooperative interactions 3.29	largely cooperative 2.19	highly cooperative 1.10	seamless interactions 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_j:	7.80	6.24	4.68	3.12	1.56	0.00

- Precedentness: Since we never developed such big projects we will set this value to very low.
- Development Flexibility: Since the assignment explained the problem of myTaxiService gave us general goals to be satisfied, but also specific information about queues management and reservation management we will set this value to high.
- Risk resolution: Since we didn't do a previous risk analysis in the design document, we will set this value to low.
- Team cohesion: Since it was our first project together, but considering also, that we worked well together, we will set this value to high.
- Process Maturity: Since we did this analysis after the RASD and the design document we will set this value to high.

Scale Driver	**Factor**	**Value**
Precedentness	Very low	6.20
Development Flexibility	high	2.03
Risk Resolution	Low	5.65
Team Cohesion	high	2.19
Process Maturity	high	3.12
Total		19.19

1.5.2 Cost Drivers

1.5.2.1 Reliability

Software failures don’t have critical consequences for the system. We set this to low.

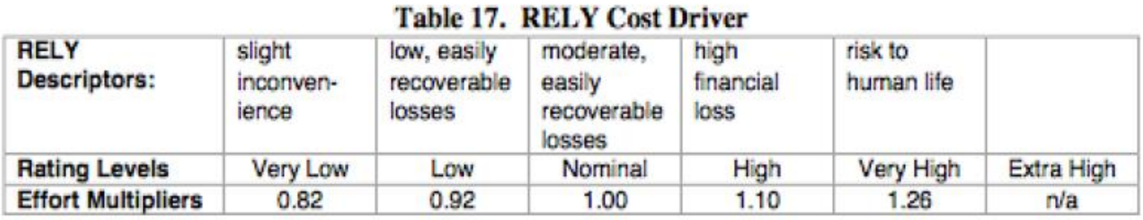


Table 17. RELY Cost Driver

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

1.5.2.2 Data Base Size

We didn’t estimate how big is our database, but to store Users,Taxi and Notification info if it is a big city we suppose that the database will have a big size,so we set this value to high.

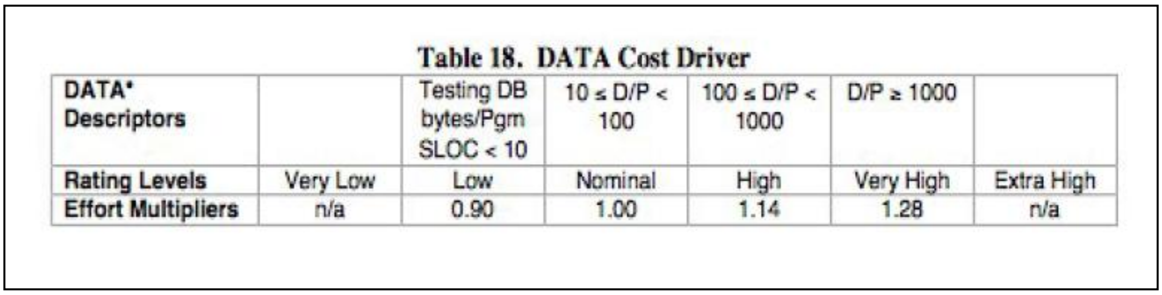


Table 18. DATA Cost Driver

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

1.5.2.3 Product Complexity

We set this value to high because we think that managing 3 clients and a central system it’s not an easy task.

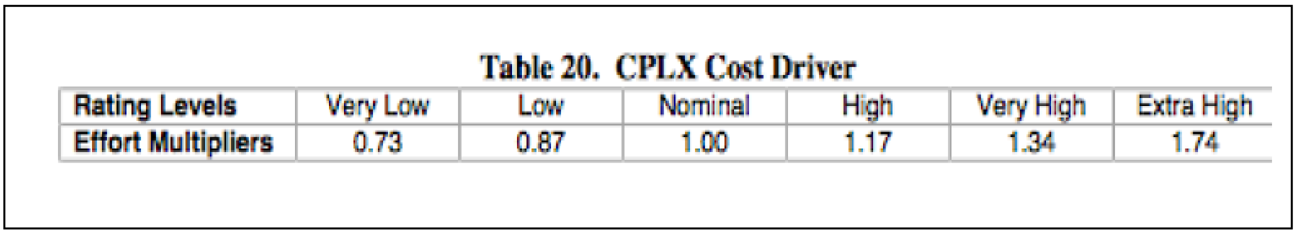


Table 20. CPLX Cost Driver

Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.73	0.87	1.00	1.17	1.34	1.74

1.5.2.4 Required Reusability

We set this to high especially to optimize the development of the mobile apps for different platforms,because if we structure the code in a reusable way (with much more code as platform-independent) the development of the same app for 3 different OS will be easier.

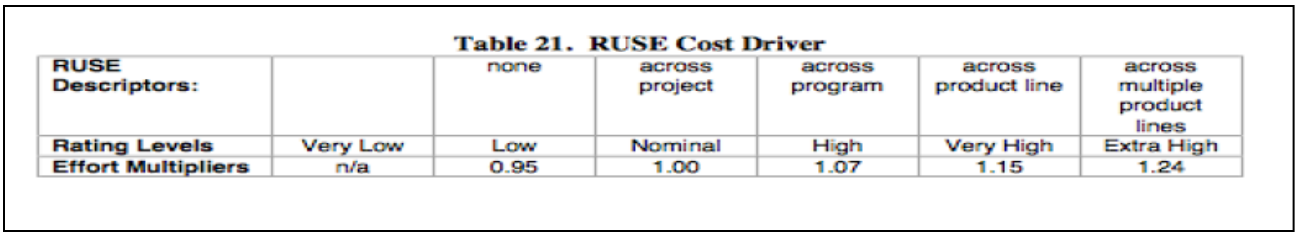


Table 21. RUSE Cost Driver

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

1.5.2.5 Documentation match to life cycle needs

Our project has been clearly explained in each part in the RASD and in the design document. So we put this value to nominal.

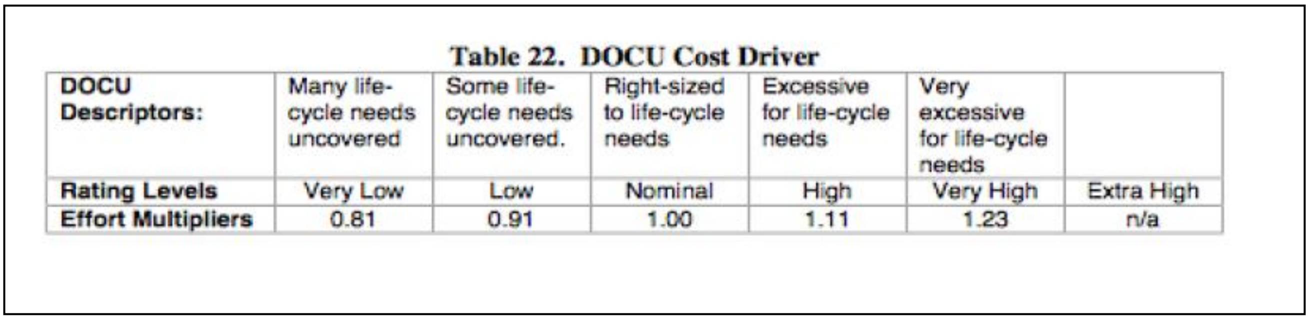


Table 22. DOCU Cost Driver

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

1.5.2.6 Execution Time Constraint

The execution time is not too relevant for our application, so we put this to low.

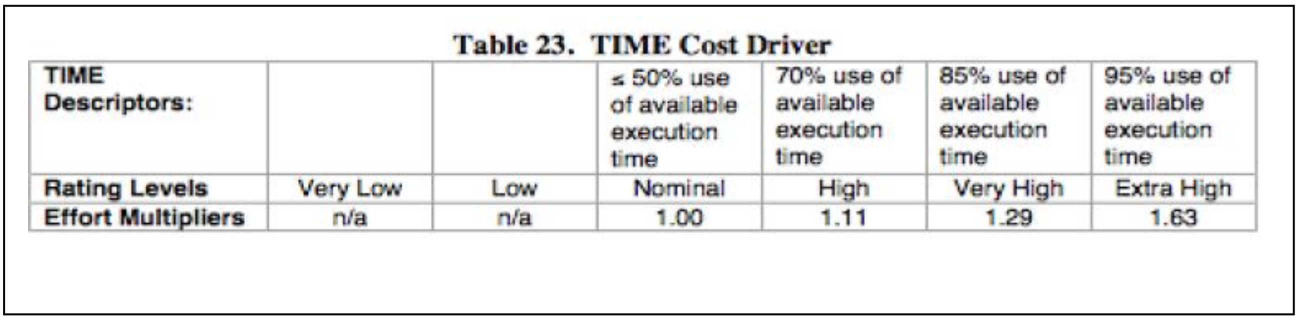


Table 23. TIME Cost Driver

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

1.5.2.7 Main Storage Constraint

In our project this is not too relevant, so we put this to low.

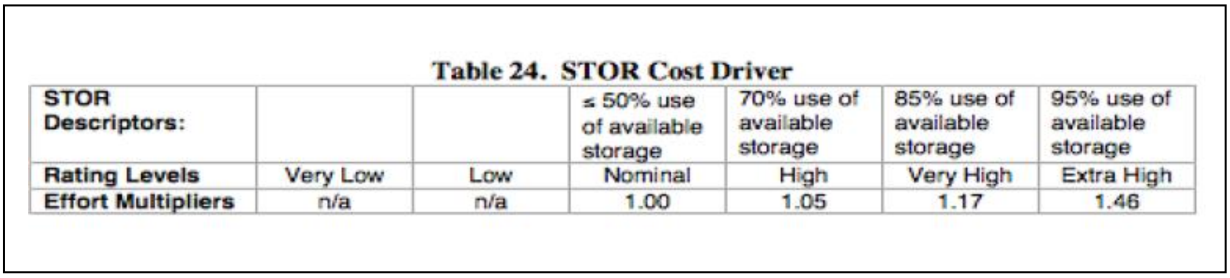


Table 24. STOR Cost Driver

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

1.5.2.8 Platform Volatility

The changes in the platform(DB and server) are not so frequent, and so we set this value to very low.

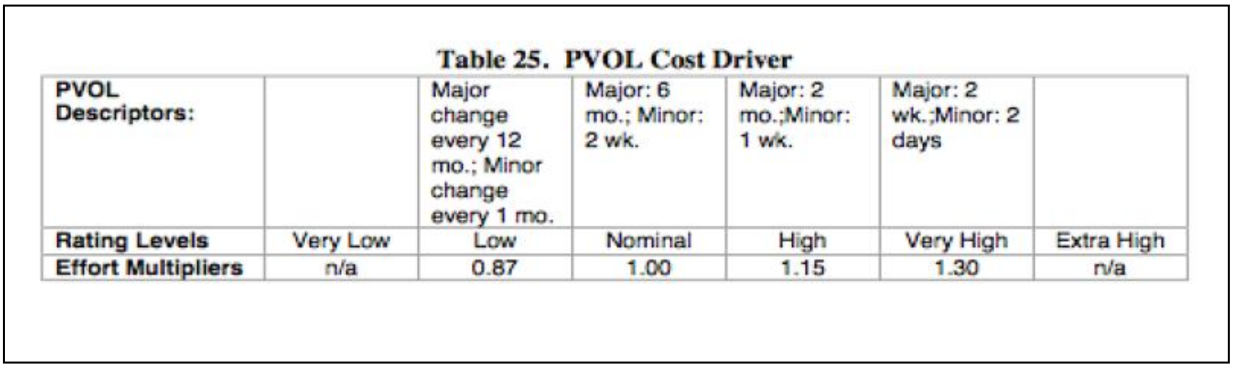


Table 25. PVOL Cost Driver

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

1.5.2.9 Analyst Capability

Since this is our first project that we focus on analyzing problem we set this value to Low.

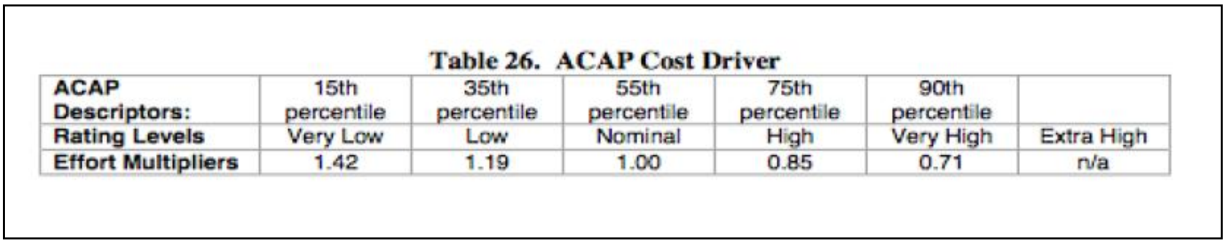


Table 26. ACAP Cost Driver

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

1.5.2.10 Programmer Capability

We have previous experience in programming from SW1 course,from other courses and from

Personal experience,so we will set this value to nominal.

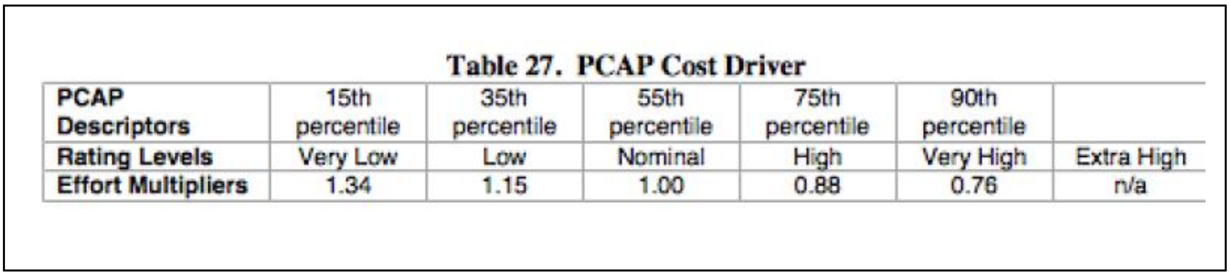


Table 27. PCAP Cost Driver

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

1.5.2.11 Application Experience

This is our first experience for project like this and so this value is set to low.

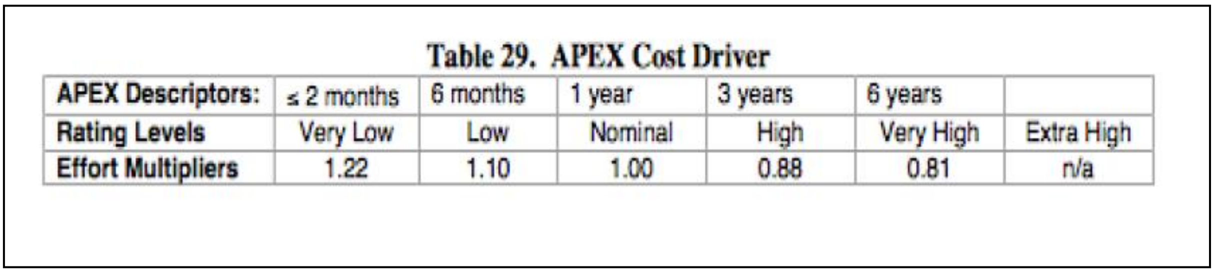


Table 29. APEX Cost Driver

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

1.5.2.12 Platform Experience

Our knowledge about databases, user interfaces, server platforms, logical components is about 1 year. So we set this value to nominal.

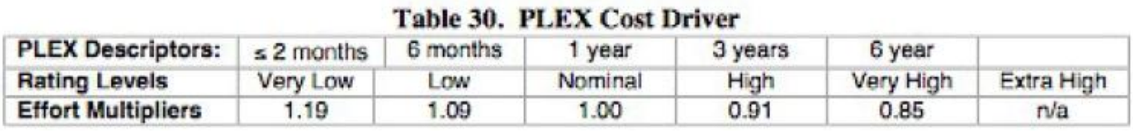


Table 30. PLEX Cost Driver

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

1.5.2.13 Language and Tool Experience

We have with java a previous experience of 2 months developing the project of SW1,but we don't have experience in android,WP,iOS development or in server architecture development. So we set this value to very low.

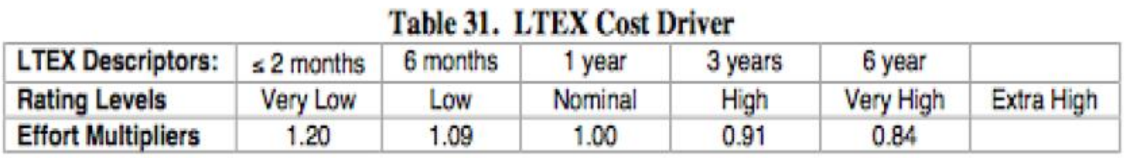


Table 31. LTEX Cost Driver

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	

1.5.2.14 Personnel Continuity

Since we suppose that we are only two that will develop the software and will work always together (supposing we are the only two employees of a software house) we will set this value to Very High.

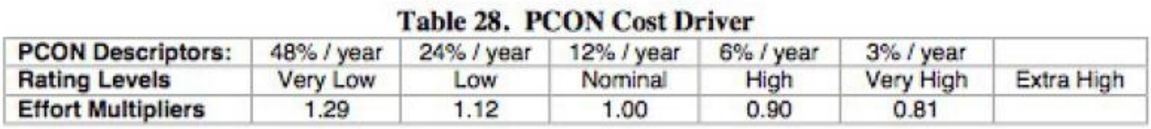


Table 28. PCON Cost Driver

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	

1.5.2.15 Usage of Software Tools

We will set this value to Very Low since as we said before we don’t have experience using the tools needed to build a big software like this one.

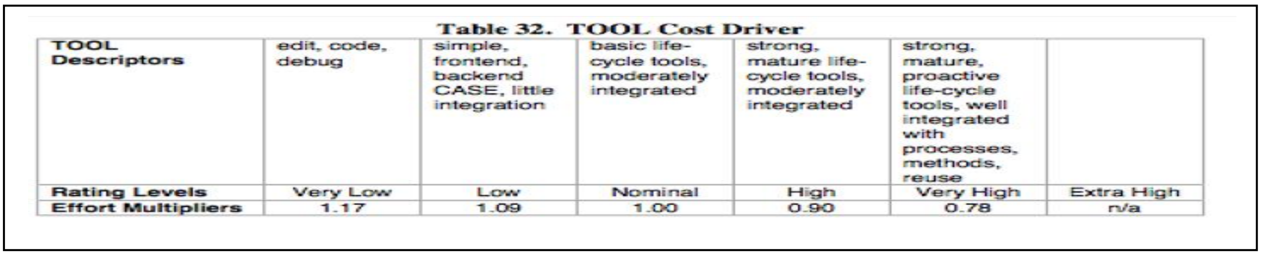


Table 32. TOOL Cost Driver

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

1.5.2.16 Multisite Development

Since we worked almost always together at the university we will set this value to ExtraHigh in fully collocated.

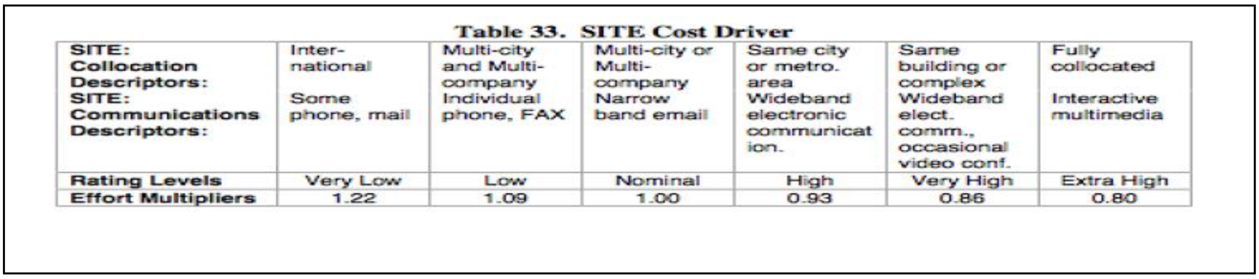


Table 33. SITE Cost Driver

SITE: Collocation Descriptors: SITE: Communications Descriptors:	Inter-national Some phone, mail	Multi-city and Multi-company Individual phone, FAX	Multi-city or Multi-company Narrow band email	Same city or metro. area Wideband electronic communication.	Same building or complex Wideband elect. comm., occasional video conf.	Fully collocated 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

1.5.2.17 Requirement Development Schedule

We used well our time for the project, but in the last period we had to work a lot in order tom complete all the functionalities of the project and so the value was set to high.

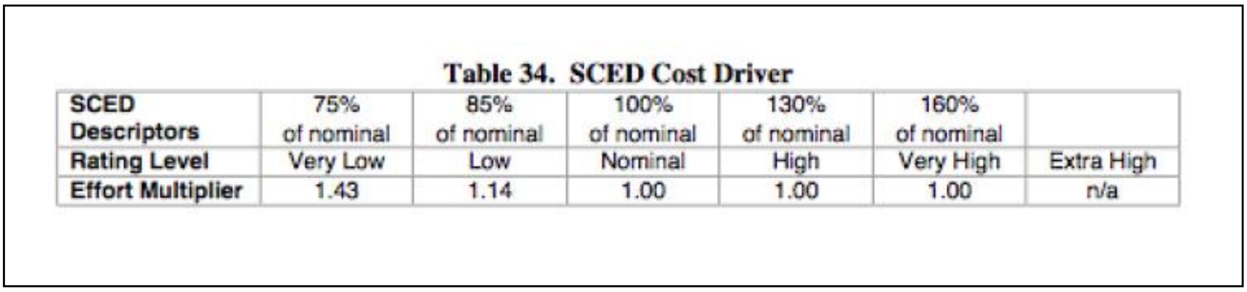


Table 34. SCED Cost Driver

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

Cost Driver	Factor	Value
Required Software Reliability	Low	0.92
Data Base Size	High	1.28
Product Complexity	High	1.34
Required Reusability	High	1.07
Documentation match	Nominal	1
Execution Time Constraint	Low	n/a
Main Storage Constraint	Low	n/a

Platform Volatility	Very Low	n/a
Analyst Capability	Low	1.19
Programmer Capability	Nominal	1.00
Application Experience	Low	1.10
Platform Experience	Nominal	1
Language and Tool Experience	Very Low	1.20
Personnel continuità	Very High	0.81
Usage of Software tools	Very Low	1.17
Multisite Development	Extra High	0.80
Required Development schedule	High	1
Product:		2,01

1.6 Effort

Effort = A*EAF*(KSLOC)^E = 288,09 PM

A = 2,94(constant for COCOMO 2.000)

EAF = 2,01(product)

KSLOC = 34,240

E=1,1019 calculated as: (sum of values in table=19,19)*0.01 +0.91

1.7 Schedule Estimation

Duration = 3,67*(Effort)^F = 22.22 Month

F=0,28 + 0,2(E-B) =0,28 + 0,2(1,1019-0,91) = 0,318

NumberOfPeople = Effort/Duration = 288,09/22.22=12,96 (almost 13 people)

Considerations on the result

1. We noticed that the effort required to develop this system is very high,but we think that it could be less if the COCOMO model considered that for developing mobile applications cross-platform languages and IDEs can be used and therefore a lot of code can be reused in developing the same application for different OS (for the Taxi Driver mobile application and the Customer Mobile App). Instead in the FP calculation we simply multiplied by 3 the Total FP of each one of the two mobile apps as they had to be developed for each OS starting from scratch. Finally the COCOMO doesn't take in account that we could use JEE framework or other frameworks to develop the central system and therefore simplify the work .

2. Since the COCOMO model is a non linear model we can't say that since we are only two and the total effort is 288,09 PM,the time required for us to develop is :

   $T_{required}$= TotalEffort/Duration = 288,09/2 = 144 Months (12 years)

   This result, as said before, is not correct for COCOMO hypothesis and model structure and also it can't be a meaningful estimation since 12 years to develop a product,considering that the market and IT technologies change very quickly,we think that developing a software in that time (even in the hypothesis of correctness of the estimation of the time required) is pointless,because even if we started today in 12 years the technologies we are using today could be easily changed or not used anymore and this will lead to the project failure.

2.Tasks

Based on the consiederations we said before,we will consider that the required number of people needed are 15 and the need time is 24 months. This is an approximation for the estimated values of the COCOMO model considering that anyway it can't make a certain prevision on the future.
Since we are 2 in the project team,we will suppose that we hired 13 people to work in the development,testing and code inspection tasks,instead the RASD,DESIGN, Integration Testing and ProjectPlan is made only by ourselves.

2.1 Task Identification and duration

For the first 4 tasks we put the same duration that were stated in the project rule description at the start of the project. For the remaining time (87 weeks of the 96 Total) we supposed that the development would have taken 80% of that time (70 weeks) and that testing and code inspection the rest 20% (17 weeks)

Task name	Task duration
Project Plan	1 week
RASD	3 weeks

DESIGN	3 weeks
Integration Testing Plan	2 weeks
Development	70 weeks
Code Inspection And Testing	17 weeks

2.1.1 Development Tasks

In this section we will analyze the macro-development tasks. We made the estimation using this formula:

T_{task} = (LOCTask/TotalLOC) * DevelopmentTime (rounded)

Task name	**Task duration**
Backend	27 weeks
Taxi Driver mobile app (WP)	7 weeks
Taxi Driver mobile app (Android)	6 weeks
Taxi Driver mobile app (iOS)	6 weeks
Customer mobile app(WP)	8 weeks
Customer mobile app(Android)	6 weeks
Customer mobile app(iOS)	7 weeks
Web Client	3 weeks

In the next sub-section we will discuss on how the backend app development duration is splitted in the development of each single component. We don't split the client development since they are thin and are considered as a single component.0

2.1.1.1 Backend tasks

In this section we will analyze the development time needed to develop each single component of thebackend app. We made the estimation using this formula:

$T_{component}$ = (Estimated percentage of totalTime)* BackEndDevTime(rounded)

The estimated percentage is based on our evaluation of the complexity of that component

Component name	**Estimated Percentage**	**Task duration**
Api Manager	20	5 weeks
Reservation System	45	12 weeks
Notification System	5	1 week
Account Manager	5	1 week
Db Manager	5	1 week
Taxi Positioning System	5	1 week
Communication System	5	1 weeks
Taxi Driver System	10	3 weeks

3.Resources

This is the table that represents the allocation of the resources to the various tasks.

Date	15/10-22/10	22/10 -13/11	13/11 -4/12	4/12-16/12		
Name	1week	3 weeks	3 weeks	2 weeks	70 weeks	17 weeks
Riccardo	Project Plan	RASD	Design	Test Plan	Development	Integration Testing
Ivan	Project Plan	RASD	Design	Test Plan	Development	Integration Testing
Member 3					Development	Integration Testing
Member 4					Development	Integration Testing
Member 5					Development	Integration Testing
Member 6					Development	Integration Testing
Member 7					Development	Integration Testing
Member 8					Development	Integration Testing
Member 9					Development	Integration Testing
Member 10					Development	Integration Testing
Member 11					Development	Integration Testing
Member 12					Development	Integration Testing
Member 13					Development	Integration Testing

4.Risks

In this section we will talk about the risks that could rise in the different project phases.

- **Misunderstanding between us and the developers** that helped us in the passage between the design document phase and the development phase:
  This can be avoided if a proper explanation is given to the developers before starting to develop.
- **Misunderstanding between us and the clients** that committed to us this work:
  This can be avoided with a strict contact with the client and with a constant update of what we're doing to make sure it is exactly what the client wanted from us.
- **Misunderstanding between developers** when are writing parts of the same components: This could happen even if the models of the project in the design document were perfect,but to minimize this risk it is useful that when a developer develops a part of a component, another developer checks his work to understand if he did it correctly,doing so a common checking between developers. This will surely slow down the development but statistically it will avoid some problems that,if spotted late,it would cost too much to repair them.
- **The integration testing part is not done in the correct order**. The components are not integrated in the correct order as described in the testing document. Some components are not unit tested and the integration testing couldn't be done in a correct way:
  The integration testing plan must be read by all developers before starting testing.

Total hours:
Riccardo Giambona: 10 hours
Ivan Antozzi: 10 hours