

Politecnico di Milano  
Computer Science and Engineering

Project of Software Engineering 2

***Project  
Plan***

Authors:

Antonio Iannacci - 854157

Daniele Romanini - 854732

Federico Seri - 854032

Reference Professor: Mirandola Raffaella

# **TABLE OF CONTENT**

## **1. Introduction**

## **2. Function Points**

- 2.1. Internal Logic Files
- 2.2. External Interface Files
- 2.3. External Inputs
- 2.4. External Outputs
- 2.5. External Inquiry

## **3. COCOMO II**

- 3.1. Introduction
- 3.2. Scale drivers
- 3.3. Cost drivers
  - 3.3.1. Product factors
  - 3.3.2. Platform factors
  - 3.3.3. Personnel factors
  - 3.3.4. Project factors
- 3.4. Results

## **4. Tasks and Schedule**

- 4.1. Main tasks
- 4.2. RASD – tasks and schedule
- 4.3. DD – tasks and schedule
- 4.4. ITPD – tasks and schedule
- 4.5. PP – tasks and schedule
- 4.6. Implementation – tasks and schedule
- 4.7. Integration Test – tasks and schedule

## **5. Resources Allocation**

- 5.1. RASD – resources allocation
- 5.2. DD – resources allocation
- 5.3. ITPD – resources allocation
- 5.4. PP – resources allocation
- 5.5. Implementation - resources allocation
- 5.6. Integration Test – tasks and schedule

## **6. Risks**

- 6.1. Project risks
- 6.2. Technical risks
- 6.3. Business risks

## **7. Reference Documents**

# 1.Introduction

The purpose of this document is to calculate the effort, the time and allocate resources necessary to the development of the application named "myTaxiService".

Moreover, risks linked to development and deployment, with associated proposed recovery actions, are identified.

## 2.Function Points

The calculus of Function Points (FP) is "a technique to assess the effort needed to design and develop custom software applications" (A. Albrech IBM Technical Journal).

The basic assumption of this technique is that the effort required to develop a software product depends on its functionalities. Function Point are used there to calculate SLOC (Source Line of Code).

The Albrecht's method identifies and count the number of function types.

They represent the "external representation" of an application, that is, its functionality.

Function Types:

- Internal Logic Files (ILF)

- External Interface Files (EIF)

- External Input (EI)

- External Outputs (EO)

- External Inquiry (EQ)

To estimate the weight of the functions, the following parameters are used:

- Data Element Type (DET): a DET is a unique user recognizable, non-recursive (non-repetitive) field.

- Record Element Type (RET): a RET is user recognizable sub group of data elements within an ILF or an EIF.

- File Types Referenced (FTR): a FTR is a file type referenced by a transaction. An FTR must also be an internal logical file or external interface file.

## 2.1. Internal Logic Files (ILF)

*Reference Table:*

	DET		
	1-19	20-50	51+
RET			
1	Low (7)	Low (7)	Average (10)
2-5	Low (7)	Average (10)	High (15)
6+	Average (10)	High (15)	High (15)

ILF	RET	DET	Weight	Function Points
User	4	14	Low	7
Taxi Driver	2	4	Low	7
Call (Fast/Booked)	4	19	Low	7
Zone	2	12	Low	7
Queue Manager	1	1	Low	7
Admin	1	1	Low	7
Shared Call	2	30	Average	10
Shared Set	1	1	Low	7
User Taxi Sharing	3	14	Low	7
			<b>Total:</b>	69

Person (not considered because it is an abstract class):

- DET: name + surname
- RET: 1

Credit Card (it does not exist without a Bean User associated. Not considered alone):

- DET: code + cvv + owner\_name + expiration\_date
- RET: Credit Card

Bean User (it does not exist without a User associated. Not considered alone):

- DET: email + phone number + address + city + zip code + password + gender + Creditcard\_Data (4)
- RET: Bean User, Credit Card

User:

- DET: Person\_Data (2) + BeanUser\_Data (11) + id\_Call
- RET: Person + Bean user + User

Taxi-Driver:

- DET: Person\_Data (2) + code + status + id\_Call
- RET: Person + Taxi Driver

Bean Position (it does not exist without a Call associated. Not considered alone):

- DET: street\_name + house\_number + nation + city
- RET: Bean\_Position

Bean Time (it does not exist without a Call associated. Not considered alone):

- DET: year + month + day + hour + minute
- RET: Bean Time

Path (it does not exist without a Call associated. Not considered alone):

- DET: street\_name + start\_number + end\_number + number\_passengers + BeanTime\_Data (5)
- RET: Path, Bean Time

Call (abstract class):

- DET: BeanPosition\_data (4) + BeanTime\_data (2) + Path\_data (9) + id\_User + id\_TaxiDriver + numberPassenger + delayMinutes
- RET: Call + Bean Time + Bean Position + Path

Fast Call/Booked Call:

- The information contained in these two classes are the same of Call from an ILF viewpoint.  
Thus, in order to calculate DET and RET, it is sufficient to refer only to Call (on the table above it is possible to see DET and RET for Call).

Zone:

- DET: code + name + id\_TaxiDriver + Path\_data(9)
- RET: Zone + Path

QueueManager:

- DET: id\_Zone
- RET: QueueManager

Admin:

- DET: id\_Admin
- RET: Admin

Shared Call:

- DET: Call\_data(20) + Path\_Data(10)
- RET: SharedCall + Call

Shared Set:

- DET: id\_SharedCall
- RET: Share

User Taxi Sharing:

- DET: User\_Data(14)
- RET: User Taxi Sharing + Person + User

## 2.2. External Interface Files (EIF)

*Reference Table:*

	DET		
	1-19	20-50	51+
RET			
1	Low (5)	Low (5)	Average (7)
2-5	Low (5)	Average (7)	High (10)
6+	Average (7)	High (10)	High (10)

EIF	RET	DET	Weight	Function Points
Distance	1	1	Low	5
Estimated Arrival Time	4	1	Low	5
Map images	1	1	Low	5
Coordinates	3	1	Low	5
Amount of money on credit card	1	1	Low	5
			<b>Total:</b>	25

Distance:

- DET: Number of kilometres
- RET: Distance

Estimated Arrival Time:

- DET: Year, Month, Day, Hour, Minutes
- RET: Estimated Arrival Time

Map images:

- DET: Image
- RET: Map images

Coordinates:

- DET: latitude, longitude, address
- RET: Coordinates

Amount of money on credit card:

- DET: Amount of money
- RET: Amount of money on credit card

## 2.3. External Inputs (EI)

*Reference Table:*

	DET		
	1-4	5-15	16+
FTR			
0-1	Low (3)	Low (3)	Average (5)
2-3	Low (3)	Average (5)	High (6)
4+	Average (5)	High (6)	High (6)

EI	FTR	DET	Weight	Function Points
Login:	5	6	High	6
Register:	3	24	High	6
Logout:	5	2	Average	5
Manage Zone:	2	26	High	6
Manage Street:	2	8	Average	5
Manage Taxi Driver:	3	12	Average	5
Manage Shift:	4	4	Average	5
Manage User profile:	3	28	High	6
Manage Call:	4	27	High	6
Manage Shared Call:	5	27	High	6
Update Taxi Driver status:	1	3	Low	3
Answer a Call:	3	4	Low	3
Update taxi driver location:	1	5	Low	3
Update passenger status:	3	22	High	6
			<b>Total:</b>	71



*Note:* With "Manage" we mean the following actions: Add, Change and Delete

Login:

- DET: username + password + 2\*Text Field + Activity Button + Confirmation message
- FTR: User + User Taxi Sharing + Taxi Driver + Admin + Person

Register:

- DET: Activity Button + BeanUser\_Data(11) + 11\*Text Field + Confirmation message
- FTR: Person + User + User Taxi Sharing

Logout:

- DET: Activity Button + Confirmation Message
- FTR: Person + User + Taxi Driver + User Taxi Sharing + Admin

Manage Zone:

- DET: Zone\_Data(12) + 12\*Text Field + Activity Button + Confirmation message
- FTR: Admin + Zone

Manage Street:

- DET: Street name + Start house number + End house number + 3\*Text Field + Activity Button + Confirmation Message
- FTR: Admin + Zone

Manage Taxi Driver:

- DET: TaxiDriver\_Data(5) + 5\*Text Field + Activity Button + Confirmation Message
- FTR: Person + Admin + Taxi Driver

Manage Shift:

- DET: Taxi Driver Shift + Text Field + Activity Button + Confirmation Message
- FTR: Person + Admin + TaxiDriver + Zone

Manage User profile:

- DET: Person\_Data(2) + BeanUser\_Data(11) + 13\*Text Field + Activity Button + Confirmation Message
- FTR: Person + User + User Taxi Sharing

Manage Call:

- DET: BeanPosition\_Data(4) + BeanTime\_Data(2) + Path\_Data(9) + 10\*Text Field + Activity Button + Confirmation Message
- FTR: Call + Person + User + User Taxi Sharing

Manage Shared Call:

- DET: BeanPosition\_Data(4) + BeanTime\_Data(2) + Path\_Data(9) + 10\*Text Field + Activity Button + Confirmation Message
- FTR: Call + Person + User + User Taxi Sharing + Shared Call

Update Taxi Driver status:

- DET: Status + Activity Button + Confirmation Message
- FTR: Taxi Driver

Answer a Call:

- DET: Activity Button + Confirmation Message + Zone Queue + ID Taxi Driver in Call
- FTR: Taxi Driver + Call + Zone

Update taxi driver location:

- DET: BeanPosition\_Data(4) + Taxi Driver Status
- FTR: Taxi Driver

Update passenger status:

- DET: Button status Passenger + Call\_Data(19) + Taxi Driver Status + Zone Queue
- FTR: Taxi Driver + Zone + Call

## 2.4. External Outputs (EO)

*Reference Table:*

	DET		
	1-5	6-19	20+
FTR			
0-1	Low (4)	Low (4)	Average (5)
2-3	Low (4)	Average (5)	High (7)
4+	Average (5)	High (7)	High (7)

EO	FTR	DET	Weight	Function Points
Call notification:	2	11	Average	5
Call request notification:	3	10	Average	5
Update Map:	2	6	Average	5
Notify Shift to Taxi Driver	1	6	Low	4
			<b>Total:</b>	19

Call notification:

- DET: BeanTime\_Data(5) + BeanPosition\_Data(4) + taxiAvailableConfirm +Message
- FTR: Call + User

Call request notification:

- DET: BeanTime\_Data(5) + BeanPosition\_Data(4) + Message
- FTR: Call + Taxi Driver + Zone

Update Map:

- DET: BeanPosition\_Data(4) + Map Images + Message
- FTR: Bean Position + Map Images

Notify Shift to Taxi Driver:

- DET: BeanTime\_Data(5) + Message
- FTR: Taxi Driver

## 2.5. External Inquiry (EQ)

*Reference Table:*

	<b>DET</b>		
	1-5	6-19	20+
<b>FTR</b>			
0-1	Low (3)	Low (3)	Average (4)
2-3	Low (3)	Average (4)	High (6)
4+	Average (4)	High (6)	High (6)

<b>EQ</b>	<b>FTR</b>	<b>DET</b>	<b>Weight</b>	<b>Function Points</b>
Show Zones	3	17	Low	3
Show Streets	2	9	Average	4
Show Calls	1	19	Average	4
Show User information	2	14	Average	4
Show Taxi Driver Information	2	5	Low	3
			<b>Total:</b>	18

Show Zones:

- DET: Zone\_Data(12) + TaxiDriver\_Data(5)
- FTR: Zone + Admin + Taxi Driver

Show Streets of a zone:

- DET: Path\_Data(9)
- FTR: Zone + Admin

Show Calls:

- DET: Call\_Data(19)
- FTR: Call

Show User information:

- DET: User\_Data(14)
- FTR: Person + User

Show Taxi Driver Information:

- DET: TaxiDriver\_Data(5)
- FTR: Person + Taxi Driver

## 2.6. Total Number of Function Points:

Function Point Type	Total Weight
<b>ILF</b>	69
<b>EIF</b>	25
<b>EI</b>	71
<b>EO</b>	19
<b>EQ</b>	18
<b>Total:</b>	<b>202</b>

The unadjusted function points is 202. Assuming that the project should be written in Java the correspondent multiplier (according to COCOMO manual for Java code) is 53. Thus the SLOC (Source Lines Of Code) results:

$$\text{UFP} * 53 = 202 * 53 = \mathbf{10706}$$

## 3. COCOMO II

### 3.1. Introduction

In order to estimate parameters such as the number of months required to complete the software project and the number of required people to develop the software and, we used a mathematical model called COCOMO, abbreviation for COConstructive COst MOdel.

All the tables used in this analysis have been taken from COCOMO II Model Definition Manual at:

[http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII\\_modelman2000.0.pdf](http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf)

### 3.2. Scale Drivers

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
<b>PREC SF<sub>I</sub></b>	Thoroughly unprecedented 6.20	Largely unprecedented 4.96	Somewhat unprecedented 3.72	Generally familiar 2.48	Largely familiar 1.24	Thoroughly familiar 0.00
<b>FLEX SF<sub>I</sub></b>	Rigorous 5.07	Occasional relaxation 4.05	Some relaxation 3.04	General conformity 2.03	Some conformity 1.01	General goals 0.00
<b>RESL SF<sub>I</sub></b>	Little (20%) 7.07	Some (40%) 5.65	Often (60%) 4.24	Generally (75%) 2.83	Mostly (90%) 1.41	Full (100%) 0.00
<b>TEAM SF<sub>I</sub></b>	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
<b>PMAT SF<sub>I</sub></b>	SW-CMM Level 1 Lower 7.80	SW-CMM Level 1 Upper 6.24	SW-CMM Level 2 4.68	SW-CMM Level 3 3.12	SW-CMM Level 4 1.56	SW-CMM Level 5 0.00

Five types of Scale Drivers are defined:

- **Precedentedness (PREC):**

It reflects the previous experience that the team has with this type of project. Given that this is first experience for the team with this kind of framework, this value is low.

- **Development flexibility (FLEX):**

It reflects the degree of flexibility in the development process. The client gave to the team only the mainly functionalities that the system has to perform without going too much in details, thanks to it this value is high.

- **Risk resolution (RESL):**

It reflects the extent of risk analysis carried out. Team implemented security access and duplicated the main infrastructures in order to handle dangerous situations, so this value is high.

- **Team cohesion (TEAM):**

It reflects how well the development team know each other and work together. The team members already known each other so the communication problems were very limited, this value is high.

- **Process maturity (PMAT):**

It reflects the process maturity of the organisation. 18 Key Process Area (KPA's) in the SEI Capability Model is used in order to assign a value to this Scale Driver. The goals of the problem have achieved so this value is high.

In the following table all the values assigned to the Drivers are resumed:

Scale Driver	Factor	Value
Precedentedness	Low	4.96
Development Flexibility	High	2.03
Risk Resolution	High	2.83
Team Cohesion	High	2.19
Process Maturity	High	3.12
<b>Total</b>	$\sum_{i=0}^n SD(i)$	<b>15.13</b>

### 3.3. Cost Drivers

In this paragraph, we report the main cost drivers divided into categories.

#### 3.3.1. Product Factors

This kind of cost drivers account for variation in the effort required to develop software caused by characteristics of the product under development.

##### Required Software Reliability (RELY)

This is the measure of the extent to which the software must perform its intended function over a period of time.

In case of failure (for example a request is not handled), the consequences of a not handled call could cause inconvenience to the client, for this reason this value is **nominal**.

RELY Descriptors:	Slight inconvenience	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

##### Data Base Size (DATA)

This cost driver attempts to capture the effect large test data requirements have on product development. The rating is determined by calculating D/P, the ratio of bytes in the testing database to SLOC in the program.

Data Descriptors:		Testing DB bytes/Pgm SLOC<10	$10 \leq D/P \leq 100$	$100 \leq D/P \leq 1000$	$D/P \geq 1000$	
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

##### Product Complexity (CPLX)

Complexity is divided into 5 areas: control operations, computational operations, device-dependent operations, data management operations and user interface management operations. This value is set to **high** according to the Component Complexity Ratings of COCOMO II rating scale.

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

### Developed for Reusability (RUSE)

This cost driver accounts for the additional effort needed to construct components intended for reuse on current or future projects.

According to the RUSE Cost Driver of COCOMO II rating scale, this value is set to **high** because the team divided the system in components making each component as independent as possible in order to reuse them in other applications projects.

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

### Documentation Match to Life-Cycle Needs (DOCU)

In COCOMO II, the rating scale for the DOCU cost driver is evaluated in terms of the suitability of the project's documentation to its life-cycle needs. This value is set to **nominal** because the mainly phases of the project has been reported into the RASD or into the DD which follow a pre-defined standard and their sections are consistent with each other.

DOCU Descriptors:	Many life-cycle needs uncovered	Some life-cycle need 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



### 3.3.2. Platform Factors

The platform refers to the target-machine complex of hardware and infrastructure software.

#### Execution Time Constraint (TIME)

It is a measure of the execution time constraint imposed upon a software system. The rating expresses the available execution time expected to be used by the system or subsystem consuming the executing time resource.

This value is set to **nominal**.

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

#### Main Storage Constraint (STOR)

It represents the degree of main storage constraint imposed on a software system or subsystem. This value is set to **nominal** because the system uses less than 50% of the available storage.

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

#### Platform Volatility (PVOL)

It is used to mean the complex of hardware and software product calls. The platform includes any compilers or assemblers supporting the development of the software system. This value is set to **low** because the platform should not be often changed, every 12 months is more than sufficient.

PVLO Descriptors:		Major change every 12 mo.; Minor change every 1 mo.	Major: 6 mo.; Minor: 2 weeks	Major: 2 mo.; Minor: 1 weeks	Major: 2 weeks; 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

### 3.3.3. Personnel Factors

These Cost Drivers are for rating the development team's capability and experience and not the single individual.

#### Analyst Capability (ACAP)

Analysts are personnel who work on requirements, high-level design and detailed design. The major attributes that should be considered in this rating are analysis and design ability, efficiency and thoroughness, and the ability to communicate and cooperate.

The team spent a lot of time in analyzing the requirements and in specifying all the ambiguities present in the description of the project. For these reasons, this value is set to **high**.

ACAP Descriptors:	15 <sup>th</sup> percentile	35 <sup>th</sup> percentile	55 <sup>th</sup> percentile	75 <sup>th</sup> percentile	90 <sup>th</sup> 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

#### Programmer Capability (PCAP)

It is evaluated according to the capability of the programmers as team rather than individuals. Major factors are ability, efficiency, thoroughness and the ability to communicate and cooperate. This value is set to **high** because the team did not have big cooperative and communication problems.

PCAP Descriptors:	15 <sup>th</sup> percentile	35 <sup>th</sup> percentile	55 <sup>th</sup> percentile	75 <sup>th</sup> percentile	90 <sup>th</sup> 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

#### Personnel Continuity (PCON)

It is evaluated in terms of the project's annual personnel turnover. This value is set to **extra high** because there is not any turnover.

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	n/a

### Applications Experience (APEX)

This Cost Driver is dependent on the level of applications experience of the project team developing the software system or subsystem. This is the first time that the team works with the Java EE framework so this value is set to **very low**.

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

### Platform Experience (PLEX)

The Post-Architecture model broadens the productivity influence of platform experience by recognizing the importance of understanding the use of more powerful platforms including database, networking and distributed middleware capabilities. The team has knowledge in handling database, client-server architecture and user interface derived from the last project's design. This value is set to **nominal**.

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

### Language and Tool Experience (LTEX)

It is a measure of the level of programming language and software tool experience of the project team developing the software system or subsystem. This value is set to **low** because it is less than 6 months that the team studies these new language and tools.

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

### 3.3.4. Project Factors

Project factors account for influences on the estimated effort such as use of modern software tools, location of the development team, and compression of the project schedule.

#### Use of Software Tools (TOOL)

The team used tools as Mockito, Arquillian and JMeter for integration tests, Github for the repository management and Eclipse for the code's implementation. This value is set to **low** according to the previous considerations.

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

#### Multisite Development (SITE)

This Cost Driver involves the assessment and judgement-based averaging of site collocation and communication support. This application refers to a service restricted to a city and it is possible to access the service through wideband electronic communication, so this value is set to **high**.

SITE: Collocation Descriptors:	International	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	0.83	0.80

### Required Development Schedule (SCED)

This rating measures the schedule constraint imposed on the project team developing the software. This value is set to **nominal** because the team has respected the nominal schedule without stretching-out or accelerating with respect to a nominal schedule.

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

In the following table all the results regarding **Software Cost Drivers** are resumed.

Scale Driver	Factor	Value
Required Software Reliability	Nominal	1
Data Base Size	Nominal	1
Product Complexity	High	1.17
Required Reusability	High	1.07
Documentation match to life-cycle	Nominal	1
Execution Time Constraint	Nominal	1
Main Storage Constraint	Nominal	1
Platform Volatility	Low	0.87
Analyst Capability	High	0.85
Programmer Capability	High	0.88
Personnel Continuity	Extra High	0.81
Applications Experience	Very Low	1.22
Platform Experience	Nominal	1
Language and Tool Experience	Low	1.09
Use of Software Tools	Low	1.09
Multisite Development	High	0.93
Required Development Schedule	Nominal	1
EAF (Effort Adjustment Factor):	$\prod_i CD(i)$	0,8896

### 3.4. Results

With COCOMO II it is possible to estimate the required effort for the realization of the project, using the equation below.

KSLOC = estimated lines of code using the FP analysis

E = exponent derived from Scale Drivers

EAF = product of all the cost drivers

SE = Schedule Equation Exponent derived from the Scale Drivers.

**KSLOC = 10.706**

**E** =  $0.91 + 0.01 \times \sum_i SD(i) = 0.91 + 0.01 \times 15.13 = \mathbf{1.0613}$

**EAF** =  $\prod_i CD(i) = \mathbf{0.8896}$

**SE** =  $0.28 + 0.2 \times (E - 0.91) = 0.28 + 0.2 \times (1.0613 - 0.91) = \mathbf{0.31026}$

**Effort** =  $2.94 \times EAF \times KSLOC^E = 2.94 \times 0.8896 \times 10.706^{1.0613} =$

**32.38 Person/Month**

**Duration** =  $3.67 \times Effort^{SE} = 3.67 \times 32.38^{0.31026} = \mathbf{10.7955 month}$

**N<sub>People</sub>** =  $\frac{Effort}{Duration} = \frac{32.38}{10.7955} = \mathbf{2.99 people}$

The number of people needed is **3**, as it is in the reality, and the expected duration of project is **11 months**.

## 4. Tasks and Schedule

### 4.1. Main Tasks

The main tasks that constitute the project are:

<b>Task name</b>	<b>Start date</b>	<b>Deadline</b>
Rasd	15/10/2015	06/11/2015
Design Document (DD)	08/11/2015	04/12/2015
Integration Test Plan Document (ITPD)	07/01/2016	21/01/2016
Project Plan (PP)	22/01/2016	02/02/2016
Final presentation	04/02/2016	22/02/2016
Implementation	01/03/2016	15/11/2016
Integration Test	17/11/2016	31/01/2017

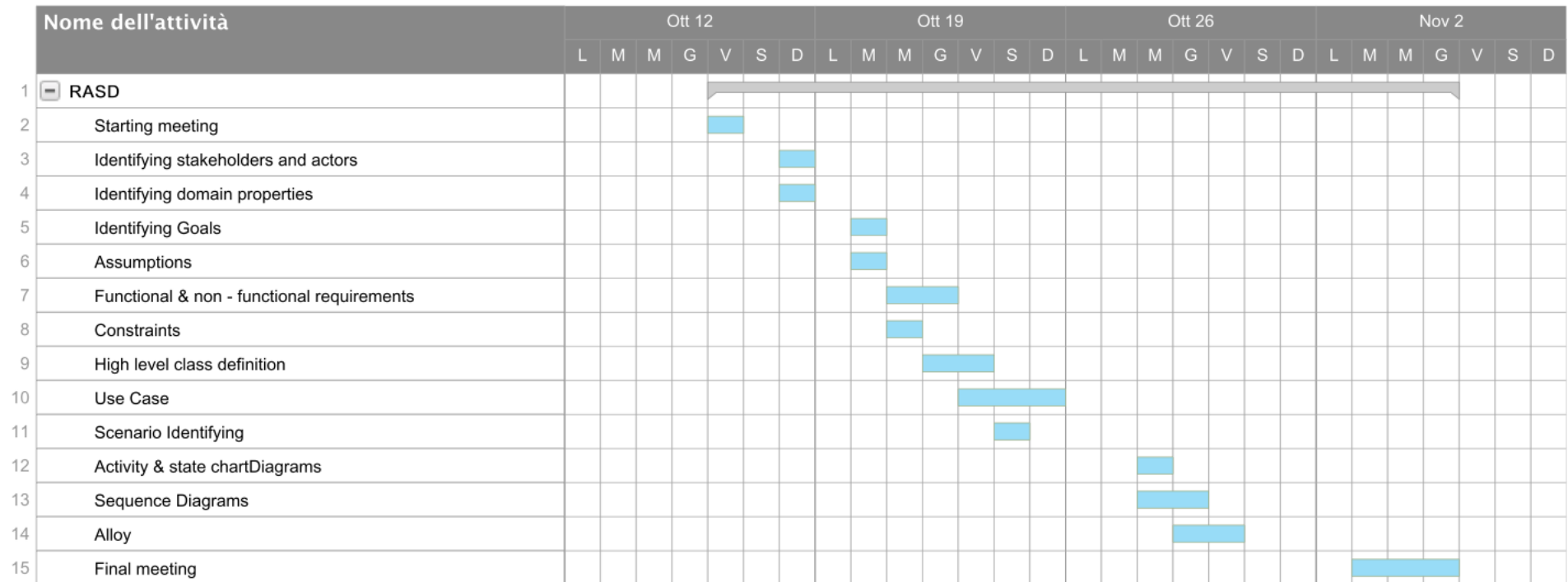
The first five tasks are given by the assignment.

Taking into account the duration of the project resulting from COCOMO II equations, we have established possible deadlines for the actual implementation and integration test.

Each main task has been divided into sub-tasks. Following, tasks division and schedules are reported (the Final Presentation division is not reported, because there is not any significant sub-tasks).



## 4.2. RASD – tasks and schedule



### 4.3. DD – tasks and schedule


[illegible]

#### 4.4. ITPD – tasks and schedule

After having decided (together) the sequence of integration, each member of the group provided to plan the integration of a subsystems and respective components, writing also the stubs and the drivers required for the integration:

[illegible]

## 4.5. PP – tasks and schedule

Nome dell'attività	Gen 18							Gen 25							Feb 1						
	L	M	M	G	V	S	D	L	M	M	G	V	S	D	L	M	M	G	V	S	D
 Project Plan																					
Starting meeting																					
Function points																					
COCOMO II																					
Tasks definition																					
Tasks design																					
Resource allocation																					
Risks identification																					
Final meeting																					

## 4.6. Implementation – tasks and schedule

In middle term meetings, there will be presentation, discussion and revision about the work of each team member and, eventually, implementation with the other team members.

## Implementation - part 1


[illegible]

[illegible]

## Implementation - part 3

[illegible]

#### 4.7. Integration Test – tasks and schedule

Nome dell'attività	Gen 18							Gen 25							Feb 1						
	L	M	M	G	V	S	D	L	M	M	G	V	S	D	L	M	M	G	V	S	D
 Project Plan																					
Starting meeting																					
Function points																					
COCOMO II																					
Tasks definition																					
Tasks design																					
Resource allocation																					
Risks identification																					
Final meeting																					



## 5. Resources allocation

### 5.1. RASD – resources allocation

05-nov-15			
04-nov-15			
03-nov-15			
02-nov-15			
01-nov-15			
31-ott-15			
30-ott-15			
29-ott-15			
28-ott-15			
27-ott-15			
26-ott-15			
25-ott-15			
24-ott-15			
23-ott-15			
22-ott-15			
21-ott-15			
20-ott-15			
19-ott-15			
18-ott-15			
17-ott-15			
16-ott-15			
RASD	Antonio	Daniele	Federico

## DD – resources allocation

DD	3-dic-15			
	2-dic-15			
	1-dic-15			
	30-nov-15			
	29-nov-15			
	28-nov-15			
	27-nov-15			
	26-nov-15			
	25-nov-15			
	24-nov-15			
	23-nov-15			
	22-nov-15			
	21-nov-15			
	20-nov-15			
19-nov-15				
18-nov-15				
17-nov-15				
16-nov-15				
15-nov-15				
14-nov-15				
13-nov-15				
12-nov-15				
11-nov-15				
10-nov-15				
9-nov-15				
Antonio				
Daniele				
Federico				

## 5.2. ITPD – resources allocation

ITDP	20-gen-16			
	19-gen-16			
	18-gen-16			
	17-gen-16			
	16-gen-16			
	15-gen-16			
	14-gen-16			
ITDP	13-gen-16			
	12-gen-16			
	11-gen-16			
	10-gen-16			
ITDP	9-gen-16			
	8-gen-16			

### 5.3. PP – resources allocation

	1-feb-16	
	31-gen-16	
	30-gen-16	
	29-gen-16	
	28-gen-16	
	27-gen-16	
	26-gen-16	
	25-gen-16	
	24-gen-16	
<b>PP</b>	<b>Antonio</b>	
	<b>Daniele</b>	
	<b>Federico</b>	

5.4. Implementation - resources allocation

5-apr-16			
4-apr-16			
3-apr-16			
2-apr-16			
1-apr-16			
31-mar-16			
30-mar-16			
29-mar-16			
28-mar-16			
27-mar-16			
26-mar-16			
25-mar-16			
24-mar-16			
23-mar-16			
22-mar-16			
21-mar-16			
20-mar-16			
19-mar-16			
18-mar-16			
17-mar-16			
16-mar-16			
15-mar-16			
14-mar-16			
13-mar-16			
12-mar-16			
11-mar-16			
10-mar-16			
9-mar-16			
8-mar-16			
7-mar-16			
6-mar-16			
5-mar-16			
4-mar-16			
3-mar-16			
Implem	Antonio	Daniele	Federico

9-mag-16			
8-mag-16			
7-mag-16			
6-mag-16			
5-mag-16			
4-mag-16			
3-mag-16			
2-mag-16			
1-mag-16			
30-apr-16			
29-apr-16			
28-apr-16			
27-apr-16			
26-apr-16			
25-apr-16			
24-apr-16			
23-apr-16			
22-apr-16			
21-apr-16			
20-apr-16			
19-apr-16			
18-apr-16			
17-apr-16			
16-apr-16			
15-apr-16			
14-apr-16			
13-apr-16			
12-apr-16			
11-apr-16			
10-apr-16			
9-apr-16			
8-apr-16			
7-apr-16			
6-apr-16			
Implem	Antonio	Daniele	Federico

12-giu-16			
11-giu-16			
10-giu-16			
9-giu-16			
8-giu-16			
7-giu-16			
6-giu-16			
5-giu-16			
4-giu-16			
3-giu-16			
2-giu-16			
1-giu-16			
31-mag-16			
30-mag-16			
29-mag-16			
28-mag-16			
27-mag-16			
26-mag-16			
25-mag-16			
24-mag-16			
23-mag-16			
22-mag-16			
21-mag-16			
20-mag-16			
19-mag-16			
18-mag-16			
17-mag-16			
16-mag-16			
15-mag-16			
14-mag-16			
13-mag-16			
12-mag-16			
11-mag-16			
10-mag-16			
Implem	Antonio	Daniele	Federico

16-lug-16			
15-lug-16			
14-lug-16			
13-lug-16			
12-lug-16			
11-lug-16			
10-lug-16			
9-lug-16			
8-lug-16			
7-lug-16			
6-lug-16			
5-lug-16			
4-lug-16			
3-lug-16			
2-lug-16			
1-lug-16			
30-giu-16			
29-giu-16			
28-giu-16			
27-giu-16			
26-giu-16			
25-giu-16			
24-giu-16			
23-giu-16			
22-giu-16			
21-giu-16			
20-giu-16			
19-giu-16			
18-giu-16			
17-giu-16			
16-giu-16			
15-giu-16			
14-giu-16			
13-giu-16			
Implem	Antonio	Daniele	Federico

19-ago-16			
18-ago-16			
17-ago-16			
16-ago-16			
15-ago-16			
14-ago-16			
13-ago-16			
12-ago-16			
11-ago-16			
10-ago-16			
9-ago-16			
8-ago-16			
7-ago-16			
6-ago-16			
5-ago-16			
4-ago-16			
3-ago-16			
2-ago-16			
1-ago-16			
31-lug-16			
30-lug-16			
29-lug-16			
28-lug-16			
27-lug-16			
26-lug-16			
25-lug-16			
24-lug-16			
23-lug-16			
22-lug-16			
21-lug-16			
20-lug-16			
19-lug-16			
18-lug-16			
17-lug-16			
Implem	Antonio	Daniele	Federico

22-set-16			
21-set-16			
20-set-16			
19-set-16			
18-set-16			
17-set-16			
16-set-16			
15-set-16			
14-set-16			
13-set-16			
12-set-16			
11-set-16			
10-set-16			
9-set-16			
8-set-16			
7-set-16			
6-set-16			
5-set-16			
4-set-16			
3-set-16			
2-set-16			
1-set-16			
31-ago-16			
30-ago-16			
29-ago-16			
28-ago-16			
27-ago-16			
26-ago-16			
25-ago-16			
24-ago-16			
23-ago-16			
22-ago-16			
21-ago-16			
20-ago-16			
Implem	Antonio	Daniele	Federico

26-ott-16			
25-ott-16			
24-ott-16			
23-ott-16			
22-ott-16			
21-ott-16			
20-ott-16			
19-ott-16			
18-ott-16			
17-ott-16			
16-ott-16			
15-ott-16			
14-ott-16			
13-ott-16			
12-ott-16			
11-ott-16			
10-ott-16			
9-ott-16			
8-ott-16			
7-ott-16			
6-ott-16			
5-ott-16			
4-ott-16			
3-ott-16			
2-ott-16			
1-ott-16			
30-set-16			
29-set-16			
28-set-16			
27-set-16			
26-set-16			
25-set-16			
24-set-16			
23-set-16			
Implem	Antonio	Daniele	Federico

10-nov-16			
9-nov-16			
8-nov-16			
7-nov-16			
6-nov-16			
5-nov-16			
4-nov-16			
3-nov-16			
2-nov-16			
1-nov-16			
31-ott-16			
30-ott-16			
29-ott-16			
28-ott-16			
27-ott-16			
Implem	Antonio		
	Daniele		
	Federico		

## 5.5. Integration Test – tasks and schedule

Int. test	21-dic-16		
	20-dic-16		
	19-dic-16		
	18-dic-16		
	17-dic-16		
	16-dic-16		
	15-dic-16		
	14-dic-16		
	13-dic-16		
	12-dic-16		
	11-dic-16		
	10-dic-16		
	9-dic-16		
	8-dic-16		
	7-dic-16		
	6-dic-16		
	5-dic-16		
	4-dic-16		
	3-dic-16		
	2-dic-16		
	1-dic-16		
30-nov-16			
29-nov-16			
28-nov-16			
27-nov-16			
26-nov-16			
25-nov-16			
24-nov-16			
23-nov-16			
22-nov-16			
21-nov-16			
20-nov-16			
19-nov-16			
18-nov-16			
17-nov-16			
	Antonio		
	Daniele		
	Federico		

25-gen-17	Antonio	Daniele	Federico
24-gen-17			
23-gen-17			
22-gen-17			
21-gen-17			
20-gen-17			
19-gen-17			
18-gen-17			
17-gen-17			
16-gen-17			
15-gen-17			
14-gen-17			
13-gen-17			
12-gen-17			
11-gen-17			
10-gen-17			
9-gen-17			
8-gen-17			
7-gen-17			
6-gen-17			
5-gen-17			
4-gen-17			
3-gen-17			
2-gen-17			
1-gen-17			
31-dic-16			
30-dic-16			
29-dic-16			
28-dic-16			
27-dic-16			
26-dic-16			
25-dic-16			
24-dic-16			
23-dic-16			
22-dic-16			
Int. test			

Int. test	27-gen-17	
	26-gen-17	
Antonio		
Daniele		
Federico		

## 6. Risks

### 6.1. Project risks

- Change of requirements: requirements may change during the development of the project
- Lack of experience of team-members: The team-members are not experienced using the necessary technology for the development. This can be significantly slow down the process.
- Lack of communication among team members: the team members do not communicate each other too much.
- Staff illness: team members are ill at critical times in the project
- Delays: the development can require more time than the estimated one.

<b>Risk</b>	<b>Probability</b>	<b>Effect</b>	<b>Strategy</b>
Change of requirements	Moderate	Serious	Derive traceability information to assess requirement change impact; maximize information hiding in the design
Lack of experience of team-members	Very high	Moderate	Looking for example and courses about the development of similar systems; ask to more expert people; organize more meeting among team-members to help each other
Lack of communication among team members	Low	Serious	Organize periodical meeting in which it should be define the actual work division
Staff illness	Moderate	Serious	Reorganize the team-work so that there is more overlap of work and people can understand each other's job
Delays	High	Moderate	Release a first version with the main functionality working; release new versions with other functionalities as soon as they are ready

## 6.2. Technical risks

- Security risk: the system is not very secure; external attacks happens
- Transaction from a previous system: the system developed must be introduced in the city and, if a previous system exists, it should replace the old one.
- Data loss: some data can be lost because of network failures, problems in transactions in database, or bad system configuration.
- Downtime: the service can go down due to server or network problems
- Integration test failures: some components may not pass the integration test phase
- Bad programming: risk of applying bad programming techniques (or also writing the code in a bad way, with bad indentation or without comment or documentation associated) increases proportionally to the dimension of the project.
- Database performance: the database used in the system cannot process as many transactions per second as expected

<b>Risk</b>	<b>Probability</b>	<b>Effect</b>	<b>Strategy</b>
Security risk	Moderate	Catastrophic	Increase system security (i.e.: hiring professional security engineers or entrust the management to a security consultant company)
Transaction from the previous system	Moderate	Moderate	Hiring a consulting company to integrate the old data
Data loss	Low	Catastrophic	Provide periodical back-up. Back-up should be maintained on a different data-system.
Downtime	Low	High	Provide more than one server to increase availability; in case of network problems, contact the Internet Service Provider
Integration test failures	Low	High	Systematically define the interfaces and the dependencies among components and subsystems
Bad programming:	Low	Moderate	Inspect periodically the code; read again the Design Document
Database performance	Moderate	Serious	Investigate the possibility of buying an higher-performance database



### 6.3. Business risks

- Budget risk: the allocated budget is not sufficient for complete the projects.
- Competitors: another company builds a similar product in less time and propose it to the government of the city.

<b>Risk</b>	<b>Probability</b>	<b>Effect</b>	<b>Strategy</b>
Budget risk	Low	Catastrophic	Better feasibility study and budget allocation
Competitors	Moderate	Severe	Maintining the product up-to-date; implement new desirable function (questionnaire among users)

## 7. Reference documents

### Assignment 5 – Project Plan

#### **RASD – DD -ITPD:**

<https://github.com/daler3/se2project/tree/master/Deliveries>

#### **Cocomo II Manual:**

[http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII\\_modelman2000.0.pdf](http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf)

#### **Function Point Analysis – Training course (David Longstreet):**

[http://www.softwaremetrics.com/Function%20Point%20Training%20Booklet%20New.p  
df](http://www.softwaremetrics.com/Function%20Point%20Training%20Booklet%20New.pdf)