



POLITECNICO DI MILANO

SOFTWARE ENGINEERING 2 PROJECT
A.Y. 2015-16

MyTaxiService
Project Plan Document
Version 1.0

CASATI Fabrizio, 853195
CASTELLI Valerio, 853992

Referent professor: DI NITTO Elisabetta

January 27, 2016

Contents

1	Introduction	1
1.1	Revision History	1
1.2	Purpose and scope	1
1.3	Definitions, Acronyms, Abbreviations	2
1.3.1	Definitions	2
1.3.2	Acronyms	2
1.3.3	Abbreviations	2
1.4	Reference Documents	2
2	Project size, cost and effort estimation	3
2.1	Size estimation: function points	3
2.1.1	Internal Logic Files (ILFs)	4
2.1.2	External Logic Files (ELFs)	6
2.1.3	External Inputs (EIs)	6
2.1.4	External Inquiries (EQs)	8
2.1.5	External Outputs (EOs)	9
2.1.6	Overall estimation	9
2.2	Cost and effort estimation: COCOMO II	10
2.2.1	Scale Drivers	10
2.2.2	Cost Drivers	12
2.2.3	Effort equation	19
2.2.4	Schedule estimation	19
3	Schedule	21
4	Resource allocation	22
5	Risk management	23
6	Conclusions	24
	Appendix A Hours of work	25

Chapter 1

Introduction

1.1 Revision History

Version	Date	Author(s)	Summary
1.0	26/01/16	Valerio Castelli & Fabrizio Casati	Initial release

1.2 Purpose and scope

This document represents the Project Plan Document for myTaxiService.

Its main purpose is to analyze the expected complexity of myTaxiService and assist the project leader in the delicate phase of cost and effort estimation. This information can be subsequently used as a guidance to define the required budget, the resources allocation and the schedule of the activities.

In the first section, we're going to use the Function Points and CO-COMO approaches together to provide an estimate of the expected size of myTaxiService in terms of lines of code and of the cost/effort required to actually develop it.

In the second section, we'll reuse these figures to propose a possible schedule for the project that covers all activities from the requirements identification to the implementation and testing activities.

In the third section we're going to assign the different members of our development group to the various tasks.

Finally, we're going to elaborate on the possible risks that myTaxiService could face during the various phase of the project and provide some general conclusions.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

1.3.2 Acronyms

- FP: Function Points.
- ILF: Internal logic file
- ELF: External logic file.
- EI: External Input.
- EO: External Output.
- EQ: External Inquiries.
- DBMS: Database Management System.
- API: Application Programming Interface.
- ETA: Estimated Time of Arrival.
- UI: User Interface.
- GPS: Global Positioning System.

1.3.3 Abbreviations

1.4 Reference Documents

- Assignment document: Assignment 5 - Project Plan.pdf
- myTaxiService Requirement Analysis and Specification Document: RASD.pdf
- The Project Plan Example documents: Example of usage of FP and COCOMO for Assignment 5.pdf and Second example of usage of FP and COCOMO for Assignment 5.pdf
- The Function Points complexity evaluation tables.
- The COCOMO II Model Definition Manual (version 2.1, 1995 – 2000 Center for Software Engineering, USC).

Chapter 2

Project size, cost and effort estimation

This section is specifically focused on providing some estimations of the expected size, cost and required effort of myTaxiService.

For the size estimation part we will essentially use the Function Points approach, taking into account all the main functionalities of myTaxiService and estimating the correspondent amount of lines of code to be written in Java. This estimation will only take into account the parts of the project that concur to the implementation of the business logic and will disregard the aspects concerning the user interface.

For the cost and effort estimation we will instead rely on the COCOMO approach, using as in initial guidance the amount of lines of code computed with the FP approach.

2.1 Size estimation: function points

The Function Points approach provides an estimation of the size of a project taking as inputs the amount of functionalities to be developed and their complexity.

The estimation is based on the usage of figures obtained through statistical analysis of real projects, which have been properly normalized and condensed in the following tables:

For Internal Logic Files and External Logic Files

	Data Elements		
<i>Record Elements</i>	<i>1-19</i>	<i>20-50</i>	<i>51+</i>
1	Low	Low	Avg
2-5	Low	Avg	High
6+	Avg	High	High

For External Output and External Inquiry

	Data Elements		
<i>File Types</i>	<i>1-5</i>	<i>6-19</i>	<i>20+</i>
0-1	Low	Low	Avg
2-3	Low	Avg	High
4+	Avg	High	High

For External Input

	Data Elements		
<i>File Types</i>	<i>1-4</i>	<i>5-15</i>	<i>16+</i>
0-1	Low	Low	Avg
2-3	Low	Avg	High
4+	Avg	High	High

UFP Complexity Weights

	Complexity Weight		
<i>Function Type</i>	<i>Low</i>	<i>Average</i>	<i>High</i>
Internal Logic Files	7	10	15
External Logic Files	5	7	10
External Inputs	3	4	6
External Outputs	4	5	7
External Inquiries	3	4	6

2.1.1 Internal Logic Files (ILFs)

myTaxiService relies on a number of ILFs to store the information it needs to offer the required functionalities. In the next few paragraphs, we'll analyze in detail the various ILFs we have identified.

First of all, the system has to store information about taxis and taxi drivers. These data are condensed in a single table that holds the first name, last name and birthdate of a taxi driver as Strings, together with his home address, SSN, email address and mobile phone number as contact information. It also stores the taxi driver's driving license, his taxi license, the taxi plate number and the taxi status (available, unavailable, on a ride, outside city) for convenience.

As for the zones, they are stores using a two-level structure. The first level of the structure holds the identifiers of all the zones, while a secondary table contains all the location coordinates (as <latitude, longitude> pairs) necessary to identify the vertices of the zone polygon.

The system furthermore needs a queue for each zone in which it can store the identifiers of the taxi drivers waiting in that zone and their relative

position in the queue, and similar structures to store the lists of taxi drivers who are unavailable, outside the city or currently on a ride. These data are stored on disk primarily to facilitate a fast recovery of the system in case of failure.

Reservations are stored in a dedicated table that holds all the information about the identifier of the passenger who booked them, the timestamp of the moment when the tuple is created, the date and time of the pickup, the origin and destination locations as addresses and the status (pending, being served, completed, cancelled).

Requests are also stored in a dedicated table with a similar structure, the only difference being the absence of the destination field and the presence of an extra field for the secret code.

The system has a dedicated table to store the passenger data. Each passenger is associated with an identifier, a name, surname and birthdate, the username, an email address and a mobile phone number. The passenger identifier is used as foreign key in a secondary table that stores his the personal settings.

Passengers and taxi drivers login information, that is username, password, identifier and user type are stored in a dedicated user table. Admin accounts are not stored in this table to achieve a greater level of security.

To keep track of who can access the administration services, information on the admin accounts is stored in a dedicated table. The main fields are name, surname, username, password and the id of the privilege level of the account, which is referencing a correspondent entry in a dedicated privileges table.

Finally, the system keeps a list of application and plugin identifiers that have access to the privileged API. Each identifier is associated with the contact information of the developer, a description of what the application or plugin does and the exhaustive list of methods that can be called.

SHOULD WE ADD THE STATISTICAL DATA HERE? OR KEEP IT AS A COMPUTED SET OF THINGS?

Using the previously defined tables, this is the count we obtain:

ILF	Complexity	FPS
Login data	Low	7
Passenger data	Low	7
Taxi drivers	Low	7
Zones	Low	7
Queues	Average	10
Reservations and requests	Low	7
API permissions	Average	10
Total		55

should we
add the
statistical
data here?
or keep it
as a com-
puted set
of things?

2.1.2 External Logic Files (ELFs)

The only external data source myTaxiService relies on is represented by the Mapping Service.

The interaction between the core system and the remote service provider happens through a RESTful API and data can be returned in JSON or XML format. The results have then to be processed before they can be used as part of our computation.

There are two main kind of interactions:

- Given the coordinates of two locations, get an estimate of the time that is necessary to drive from one to the other
- Given an address, get the correspondent pair of coordinates (reverse geocoding)

On the client side, the mapping service is also used to retrieve the graphical representation of the city map to be displayed on the smartphone of the taxi driver.

Given the complexity of the interaction and the amount of data that is retrieved, it is reasonable to classify this logic file as a complex one.

ELF	Complexity	FPS
ETA computation	Low	10
Reverse geocoding	Low	10
Map data retrieval	Low	10
Total		30

2.1.3 External Inputs (EIs)

myTaxiService supports many kind of interactions with different categories of users.

We are now going to summarize the impact of the offered features, grouping them by user category. All users:

- Login/Logout: these are simple operations that involve only the account manager. They contribute 3 FPS each.

Passengers:

- Password retrieval: this operation has an average complexity, as it involves a number of steps in order to be sure the user is really entitled to retrieve his password. For this reason, it contributes 4 FPS.

- Change settings: this operation also has an average complexity, as the number of settings to be managed can be quite high. As such, it contributes 4 FPs.
- Request or reserve a taxi: these are both very complex operations that involve a large number of components. For this reason they account 6 FPs each.
- Delete a reservation: since this is straightforward operation, it yields 3 FPs.
- Register a new account: this operation also has an average complexity, as it involves a number of checks on the validity of the fields. As such, it contributes 4 FPs.
- View reservation history: since this is straightforward operation, it yields 3 FPs.

Administrators:

- Insert, delete and update zones: these are very complex operations that involve a great number of components. For this reason they account 6 FPs each.
- Insert, delete and update taxi drivers: as for the zones, the complexity of these operations is high, so they account another 6 FPs each.
- Request service statistics: as this operation involves some fairly complicated aggregate queries on the database, it can be considered complex and thus contributes 6FPs to the total amount.
- Grant and revoke privileges to an application or plugin: while these two specular operations involve a large number of fields that can be set, the impact on the database is quite limited. For this reason we think they have an average complexity and they should contribute 4 FPs each.

Taxi drivers:

- Accept, refuse and end ride: even though from a client point of view these operations seem trivial, the steps required to properly rearrange the taxi queues are quite complex. For this reason they account 6 FPs each.
- Set availability: this operation also involves a few rearrangements of the taxi queues, but has a smaller impact on the overall behavior of the system. For this reason it can be thought as having an average complexity and it contributes 4 FPs.

The final results are shown in the following table:

EI	Complexity	FPS
Login/Logout	Low	2x3
Password retrieval	Average	4
Change settings	Average	4
Request or reserve a taxi	High	2x4
Delete a reservation	Low	3
Register a new passenger account	Average	4
View reservation history	Low	3
Insert, delete and update zones	High	3x6
Insert, delete and update taxi drivers	High	3x6
Request service statistics	High	6
Grant and revoke app privileges	Average	2x4
Grant and revoke plugin privileges	Average	2x4
Accept, refuse and end ride	High	3x6
Set taxi availability	Average	4
Total		112

2.1.4 External Inquiries (EQs)

As specified by the FP guidelines, an inquiry is essentially a data retrieval request performed by an user.

myTaxiService supports a few interactions of this type that don't require complex computations:

- A taxi driver can retrieve his position in the queue of his current zone.
- A passenger can retrieve his reservation history.
- An administrator can retrieve the full list of taxi drivers, zones, passengers or approved applications and plugins that are making usage of the APIs.

All these operations can be though as fairly simple. The resulting table is the following:

EQ	Complexity	FPS
Retrieve taxi position in queue	Low	3
Retrieve passenger reservation history	Low	3
Retrieve list of taxi drivers	Low	3
Retrieve list of zones	Low	3
Retrieve list of passengers	Low	3
Retrieve list of approved applications	Low	3
Retrieve list of approved plugins	Low	3
Total		21

2.1.5 External Outputs (EOs)

As part of its normal behavior, myTaxiService occasionally needs to communicate with the user outside the context of an inquiry. These occasions are:

- Notify a taxi driver that he has been assigned to a request.
- Notify a passenger that his request has been accepted.
- Notify a passenger that his request has been dropped.
- Notify a taxi driver that his zone has changed.
- Notify a taxi driver that his position in the zone queue has changed.

All these operations can be thought as fairly simple. The resulting table is the following:

EQ	Complexity	FPS
Taxi request assignment notification	Low	4
Request accepted notification	Low	4
Request dropped notification	Low	4
Zone changed notification	Low	4
Position in the queue changed notification	Low	4
Total		20

2.1.6 Overall estimation

The following table summarizes the results of our estimation activity:

Function Type	Value
Internal Logic Files	55
External Logic Files	30
External Inputs	112
External Inquiries	21
External Outputs	20
Total	238

Considering Java Enterprise Edition as a development platform and disregarding the aspects concerning the implementation of the mobile applications (which can be thought as pure presentation with no business logic), we can estimate the total number of lines of code.

Depending on the conversion rate, we have a lower bound of:

$$\text{SLOC} = 238 * 46 = 10948$$

and an upper bound of

$$\text{SLOC} = 238 * 67 = 15946$$

2.2 Cost and effort estimation: COCOMO II

In this section we're going to use the COCOMO II approach to estimate the cost and effort needed to develop myTaxiService.

2.2.1 Scale Drivers

In order to evaluate the values of the scale drivers, we refer to the following official COCOMO II table:

Scale Factor values, SF_j , for COCOMO II Models

Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprece-dented	largely unprece-dented	somewhat unprece-dented	generally familiar	largely fa-miliar	thoroughly familiar
SF_j	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	rigorous	occasional relaxation	some relaxation	general confor-mity	some con-formity	general goals
SF_j	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 _j	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	very diffi- cult inter- actions	some diffi- cult inter- actions	basically coop- erative interac- tions	largely co- operative	highly co- operative	seamless interac- tions
SF _j	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	Level 1 Lower	Level 1 Upper	Level 2	Level 3	Level 4	Level 5
SF _j	7.80	6.24	4.68	3.12	1.56	0.00

A brief description for each scale driver:

- Precedentedness: it reflects the previous experience of our team with the development of large scale projects. Since we are not expert in the field, this value will be low.
- Development flexibility: it reflects the degree of flexibility in the development process with respect to the external specification and requirements. Since there are very strict requirements on the functionalities but nothing specific is stated as for the technology to be used, this value will be low.
- Risk resolution: reflects the level of awareness and reactivity with respect to risks. The risk analysis we performed is quite extensive, so the value will be set to very high.
- Team cohesion: it's an indicator of how well the team members know each other and work together in a cooperative way. For our team, the value is very high.
- Process maturity: although we had some problems during the development of the project, the goals have been successfully achieved, so this value is set to high.

The results of our evaluation is the following:

Scale Driver	Factor	Value
Precedented (PREC)	Low	4.96
Development flexibility (FLEX)	Low	4.05
Risk resolution (RESL)	Very high	1.41
Team cohesion (TEAM)	Very high	1.10
Process maturity (PMAT)	Very high	3.12
Total		14,64

2.2.2 Cost Drivers

- Required Software Reliability:

Since the system represents the only way to get taxis in the city, a malfunctioning could lead to important financial losses. For this reason, the RELY cost driver is set to high.

RELY Cost Drivers

RELY De- scriptors	slightly inconve- nience	easily re- coverable losses	moderate recov- erable losses	high fi- nancial loss	risk to hu- man life	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	0.82	0.92	1.00	1.10	1.26	n/a

- Database size:

This measure considers the effective dimension of our database. We don't have the ultimate answer, but our estimation given the tables and fields we have is to reach a 3GB database. Since it is distributed over 10.000-15.000 SLOC, the ratio D/P is between 209 and 314, resulting in the DATA cost driver being high.

DATA Cost Drivers

DATA De- scriptors		Testing DB bytes/pgm SLOC < 10	$10 \leq D/P \leq 100$	$100 \leq P \leq 1000$	$DP > 1000$	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	n/a	0.90	1.00	1.14	1.28	n/a

- Product complexity:

Set to very high according to the COCOMO II rating scale.

CPLX Cost Driver

Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort multipliers	0.73	0.87	1.00	1.17	1.34	1.74

- Required reusability:

In our case, the reusability requirements are limited in scope to the project itself, so the RUSE cost driver is set to nominal.

RUSE Cost Driver

RUSE Descriptors		None	Across project	Across program	Across product line	Across multiple product lines
Rating level	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:

This parameter describes the relationship between the documentation and the application requirements. In our case, every need of the product life-cycle is already foreseen in the documentation, so the DOCU cost driver is set to nominal.

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 level	Very low	Low	Nominal	High	Very High	Extra High
Effort multipliers	0.81	0.91	1.00	1.11	1.23	n/a

- Execution time constraint:

This parameter describes the expected amount of CPU usage with respect to the computational capabilities of the hardware. As myTaxiService is a quite complex software, our expectance is that its CPU usage will be very high.

TIME Cost Driver

TIME De- scriptors			$\leq 50\%$ use of available execution time	70% use of available execution time	85% use of available execution time	95% use of available execution time
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	n/a	n/a	1.00	1.11	1.29	1.63

- Storage constraint:

This parameter describes the expected amount of storage usage with respect to the availability of the hardware. As current disk drives can easily contain several terabytes of storage, this value is set to nominal.

STOR Cost Driver

STOR De- scriptors			$\leq 50\%$ use of available storage	70% use of available storage	85% use of available storage	95% use of available storage
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	n/a	n/a	1.00	1.05	1.17	1.46

- Platform Volatility:

For what concerns the core system, we don't expect our fundamental platforms to change very often. However, the client applications may require at least a major release once every six months to be aligned with the development cycle of the main mobile operating systems. For this reason, this parameter is set to nominal.

PVOL Cost Driver

PVOL De- scriptors		Major change every 12 mo., minor change every 1 mo.	Major: 6mo; minor: 2wk.	Major: 2mo, minor: 1wk	Major: 2wk; mi- nor: 2 days	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	n/a	0.87	1.00	1.15	1.30	n/a

- Analyst Capability:

We think the analysis of the problem has been conducted in a thorough and complete way with respect to a potential real world implementation. For this reason, this parameter is set to high.

ACAP Cost Driver

ACAP De- scriptors	15th per- centile	35th per- centile	55th per- centile	75th per- centile	90th per- centile	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	1.42	1.19	1.00	0.85	0.71	n/a

- Programmer Capability:

We have not implemented the project, so this parameter is just an estimation; however we are fairly in our programming abilities, so we'll set this parameter to high.

PCAP Cost Driver

PCAP De- scriptors	15th per- centile	35th per- centile	55th per- centile	75th per- centile	90th per- centile	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	1.34	1.15	1.00	0.88	0.76	n/a

- Application Experience:

We have some experience in the development of Java applications, but we never tackled a Java EE system of this kind. For this reason we're going to set this parameter to low.

APEX Cost Driver

APEX De- scriptors	\leq 2 months	6 months	1 year	3 years	6 years	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	1.22	1.10	1.00	0.88	0.81	n/a

- Platform Experience:

We don't have any experience with the Java EE platform, but we have some previous experience with databases, user interfaces and server side development. For this reason, we're going to set this parameter to nominal.

PLEX Cost Driver

PLEX De- scriptors	\leq 2 months	6 months	1 year	3 years	6 years	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	1.19	1.09	1.00	0.91	0.85	n/a

- Language and Tool Experience:

We don't have any experience with the Java EE platform, but we have some previous experience with databases, user interfaces and server side development. We are also knowledgeable of the development environment, so we're going to set this parameter to nominal.

LTEX Cost Driver

LTEX De- scriptors	\leq 2 months	6 months	1 year	3 years	6 years	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	1.20	1.09	1.00	0.91	0.84	n/a

- Personnel continuity:

This parameter is quite relevant in our case, since the time we can spend on this project is limited. For this reason, this parameter is set to very low.

PCON Cost Driver

PCON De- scriptors	48% / year	24% / year	12% / year	6% / year	3% / year	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	1.29	1.12	1.00	0.90	0.81	n/a

- Usage of Software Tools:

Our application environment is complete and well integrated, so we'll set this parameter as high.

TOOL Cost Driver

TOOL De- scriptors	edit, code, debug	simple, frontend, backend CASE, little inte- gration	basic life-cycle tools, mod- erately integrated	strong, mature life-cycle tools, mod- erately integrated	strong, mature, proactive life-cycle tools, well integrated with pro- cesses, methods, reuse	
Rating level	Very low	Low	Nominal	High	Very High	Extra High
Effort mul- tipliers	1.17	1.09	1.00	0.90	0.78	n/a

- Multisite development:

Although we live in two different cities, we have collaborated relying hugely on wideband Internet services including social networks and emails. For this reason, we're going to set this parameter to very high.

SITE Cost Driver

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 level	Very low	Low	Nominal	High	Very High	Extra High
Effort multipliers	1.22	1.09	1.00	0.93	0.86	0.80

- Required development schedule:

Although our efforts were well distributed over the available development time, the definition of all the required documentation took a consistent amount of time, especially for the requirement analysis and the design phases. For this reason, this parameter is set to high.

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 multipliers	1.43	1.14	1.00	1.00	1.00	n/a

Overall, our results are expressed by the following table:

Cost Driver	Factor	Value
Required Software Reliability (RELY)	High	1.10
Database size (DATA)	High	1.14
Product complexity (CPLX)	Very high	1.34
Required Reusability (RUSE)	Nominal	1.00
Documentation match to life-cycle needs (DOCU)	Nominal	1.00
Execution Time Constraint (TIME)	Very high	1.29
Main storage constraint (STOR)	Nominal	1.00
Platform volatility (PVOL)	Nominal	1.00

Analyst capability (ACAP)	High	0.85
Programmer capability (PCAP)	High	0.88
Application Experience (APEX)	Low	1.10
Platform Experience (PLEX)	Nominal	1.00
Language and Tool Experience (LTEX)	Nominal	1.00
Personnel continuity (PCON)	Very low	1.12
Usage of Software Tools (TOOL)	High	0.90
Multisite development (SITE)	Very high	0.86
Required development schedule (SCED)	High	1.00
Total		1,54613

2.2.3 Effort equation

This final equation gives us the effort estimation measured in Person-Months (PM):

$$\text{Effort} = A * \text{EAF} * \text{KSLOC}^E$$

where:

$$\begin{aligned} A &= 2.94 \text{ (for COCOMO II)} \\ \text{EAF} &= \text{product of all cost drivers (1.54613)} \\ E &= \text{exponent derived from the scale drivers. It is computed as:} \\ &B + 0.01 * \sum_i SF[i] = B + 0.01 * 14.64 = 0.91 + 0.1464 = 1.0564 \\ &\text{in which B is equal to: 0.91 for COCOMO II.} \end{aligned}$$

With this parameters we can compute the effort value, which has a lower bound of:

$$\text{Effort} = A * \text{EAF} * \text{KSLOC}^E = 2.94 * 1.54613 * 10.948^{1.0564} = 56.957 \text{ PM} \approx 57 \text{ PM}$$

and an upper bound of:

$$\text{Effort} = A * \text{EAF} * \text{KSLOC}^E = 2.94 * 1.54613 * 15.946^{1.0564} = 84.737 \text{ PM} \approx 85 \text{ PM}$$

2.2.4 Schedule estimation

Regarding the final schedule, we are going to use the following formula:

$$\text{Duration} = 3.67 * \text{Effort}^E$$

As a lower bound, we consider

$$\begin{aligned} F &= 0.28 + 0.2 * (E - B) = 0.28 + 0.2 * 0.1464 = 0.30928 \\ \text{Effort} &= 56.957 \text{ PM} \\ \text{Duration} &= 3.67 * (56.957)^{0.30928} = 12.81 \text{ months} \end{aligned}$$

while as an upper bound, we consider

1	$F = 0.28 + 0.2 * (E - B) = 0.28 + 0.2 * 0.1464 = 0.30928$
	Effort = 84.737 PM
3	Duration = $3.67 * (56.957)^{0.30928} = 14.49$ months

which seem to be both reasonable estimates.

Chapter 3

Schedule

Chapter 4

Resource allocation

Chapter 5

Risk management

Chapter 6

Conclusions

Appendix A

Hours of work

To redact this document, we spent ?? hours per person.