

PowerEnJoy

Project Plan Document

A.Y 2016/2017

Francesco Tinarelli(matr:806146)

Marco Wenzel(matr:878021)

Versione1.1

**Index**

1) Introduction------------------------------------------------------------------3

1.1) Purpose and Scope ------------------------------------------------3

1.3) List of abbreviations----------------------------------------------3

2) Project size, cost and effort estimation---------------------------------4

2.1) Overview-------------------------------------------------------------4

2.2) Size estimation: Function Points ------------------------------4

2.3) Internal Logic Files(ILFs)--------------------------------------5

2.4) External Logic Files(ELFs)-------------------------------------6

2.5) External inputs(EIs)---------------------------------------------6

2.6) Internal Logic Files(ILFs)--------------------------------------7

2.7) External inputs(EIs)---------------------------------------------8

2.8) Overall estimation -----------------------------------------------8

2.9) Cost and effort estimation: COCOMO II--------------------9

2.10) Scale factor-------------------------------------------------------9

2.11) Cost driver-------------------------------------------------------11

2.12) Effort estimation ----------------------------------------------20

2.12) Duration estimation-------------------------------------------20

3) Schedule---------------------------------------------------------------------21

4) Project risk------------------------------------------------------------------22

5) Appendix--------------------------------------------------------------------23

5.1) References----------------------------------------------------------23

5.2) Hours of work----------------------- ------------------------------23

1. Introduction

1.1 Purpose and scope

The project plan document has the main purpose to analyze the cost and effort estimation of PowerEnJoy, to schedule, after the estimation, how to divide the resources we have for completing the work in time, and finally to analyze the possible risk that during the works can occur.

The document is divided in three parts: the first parts focus on the cost and effort estimation. This estimation is done using the function points method and COCOMO.

The second part focus on the scheduling of the work. We divide the work in tasks and allocate these tasks to our resources.

The third and last part focus on risk that we could face during the development of PowerEnJoy. We will provide the probability of face that risk and the possible solution.

1.2 List of abbreviations

• **PP**: Project Plan.

• **FP**: Function Points.

• **ILF**: Internal logic file.

• **ELF**: External logic file.

• **EI**: External Input.

• **EO**: External Output.

• **EQ**: External Inquiries.

• **DBMS**: Database Management System.

• **DD:** Design Document.

1. Project size, cost and effort estimation

2.1. Overview

In this section, we will provide an estimation of the cost and the size of PowerEnJoy project.

This chapter will be divide into two parts. The first part belongs to the estimation of the size of the project. This estimation is done with a function points approach.

The second part belongs to estimation of the cost. This estimation is done with the COCOMO approach.

2.2. Size estimation: Function Points

The function points method provides an estimation of the size of the project, using the number of functionalities that the project provides to the client.

The estimation is done using the following tables. The tables belong to a statistical analysis.

For Internal Logic Files and External Logic 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+ |
| 0-1 | Low | Low | Avg |
| 2-4 | Low | Avg | High |
| 4+ | Avg | High | High |

For External Input

|  |  |  |  |
| --- | --- | --- | --- |
|  | Data elements | | |
| File Types | 1-4 | 5-15 | 16+ |
| 0-1 | Low | Low | Avg |
| 2-3 | Low | Avg | High |
| 3+ | Avg | High | High |

UFP Complexity Weights

|  |  |  |  |
| --- | --- | --- | --- |
|  | Complexity Weights | | |
| Function Types | Low | Average | High |
| Internal Logic Files | 7 | 10 | 15 |
| External Logic Files | 5 | 7 | 10 |
| External Inputs  External Outputs  External Inquiries | 3  4  3 | 4  5  4 | 6  7  6 |

2.3 Internal Logic Files(ILFs)

PowerEnJoy must store a lot of users because we think that a lot of people will use this service, and the information of the users are lots such as name, surname, password, e-mail, address, bank code. So, complexity is high.

For the car, we must store the availability and the actual position, because that two continue to change we deduce an average complexity.

We must store the reservation that is not simple because there to store the car, the user and date, and the right reservation hours for the timer of the reservation, so complexity is high.

Finally, we must store the bill of the use of the car. We need to store the possibly charge or discount, and the total amount of the ride so we deduce an average complexity.

This is the final table for the internal logic files:

|  |  |  |
| --- | --- | --- |
| ILFs | Complexity | FPs |
| User information | High | 15 |
| Car information | Avg | 10 |
| Reservation | Low | 15 |
| Bill | Avg | 10 |
| Total |  | 50 |

2.4 External Logic Files(ELFs)

For the External Logic Files, we should store only the information that the external agency for the payment give to us about the last nonpayment of the ride. So, the complexity is quite easy but on average because we need to store and change only a boolean but every user has a different one.

This is the final table for the external logic files:

|  |  |  |
| --- | --- | --- |
| ELFs | Complexity | FPs |
| Payment information | Avg | 15 |
| Total |  | 15 |

2.5 External inputs(EIs)

There are a lot of input that arrive from the different type of client of the application.

* **Registration:** is a complex operation, because involve a lot of information and so we need to do a lot of query.
* **Login:** is a simple operation because needs only the control of the e-mail and the password.
* **Password recovery:** we need to control the existence of the user and if is the right one. We will send again the password, so we need to do not simple operation and we deduce a high complexity.
* **Reserve car:** Complex input indeed we need to pass in a lot of component, so is a high complexity.
* **Unlock car:** Simple operation because we need only to control the user and the relative car reserved.
* **View reservation and payment history:** Simple operation that involve only the database of the application.
* **Send data sensor of the car:** Simple operation because is only a sending of the sensor that are in the car.

This is the final table for the External inputs:

|  |  |  |
| --- | --- | --- |
| EIs | Complexity | FPs |
| Registration  Login  Password recovery  Reserve car  Unlock car  Send data sensor  Payment/reservation history | High  Low  Avg  High  Low  Low  Low | 6  3  6  6  3  3  3 |
| Total |  | 30 |

2.6 External Outputs(EOs)

The application must provide these outputs for the client

* Send the bill to the client, this operation is quite simple because have only to send an e-mail to the correspondent user, so complexity is low.
* Send the fee for the reservation lost. This operation is like the send bill one so, complexity is low.
* Open the car. This operation is on average complexity because the system must control the reservation information, the position of the car and the mobile that want to open the car, and control also if the time of the reservation is finish.
* Notify the user if he hasn`t paid the last ride so, until he doesn`t pay it, he can`t reserve another one car. This action is only a notification via e-mail, so is a low complexity action.
* Notify the user if the he has paid the last ride, and so he can restart to reserve a new car, like the before one, is a low complexity action for the same reason.
* Notify the bill to the external agency. The external agency for the payment and the application has a direct link to each other, so the complexity for this action is low.

This is the final table for the External input:

|  |  |  |
| --- | --- | --- |
| EOs | Complexity | FPs |
| Send bill  Fee lost reservation  Open car  Notify not pay  Notify restart reserve  Bill to external agency | Low  Avg  Low  Low  Low  Low | 4  5  4  4  4  4 |
| Total |  | 25 |

2.7 External Inquiries(EQs)

The only external inquiries that the client can perform, is a request for the payment and reservation history. This action is quite simple to do, because have only to perform a query to the database of the user that asks for his history. So, complexity is low.

|  |  |  |
| --- | --- | --- |
| EQs | Complexity | FPs |
| Payment history | Low | 3 |
| Total |  | 3 |

2.8 Overall estimation

The following table summarizes the results of our estimation activity:

|  |  |
| --- | --- |
| Function Types | Value |
| Internal Logic Files | 50 |
| External Logic Files | 15 |
| External Inputs  External Outputs  External Inquiries  Total | 30  25  3  123 |

We use JEE for the platform development:

SLOC=123\*46=5.658

With an upper bound of

SLOC=113\*67=8.241

2.9 Cost and effort estimation: COCOMO II

For estimating cost and effort, we use the Cocomo II method, statistical approach.

First step for arriving to an effort estimation is to understand in which case we are:

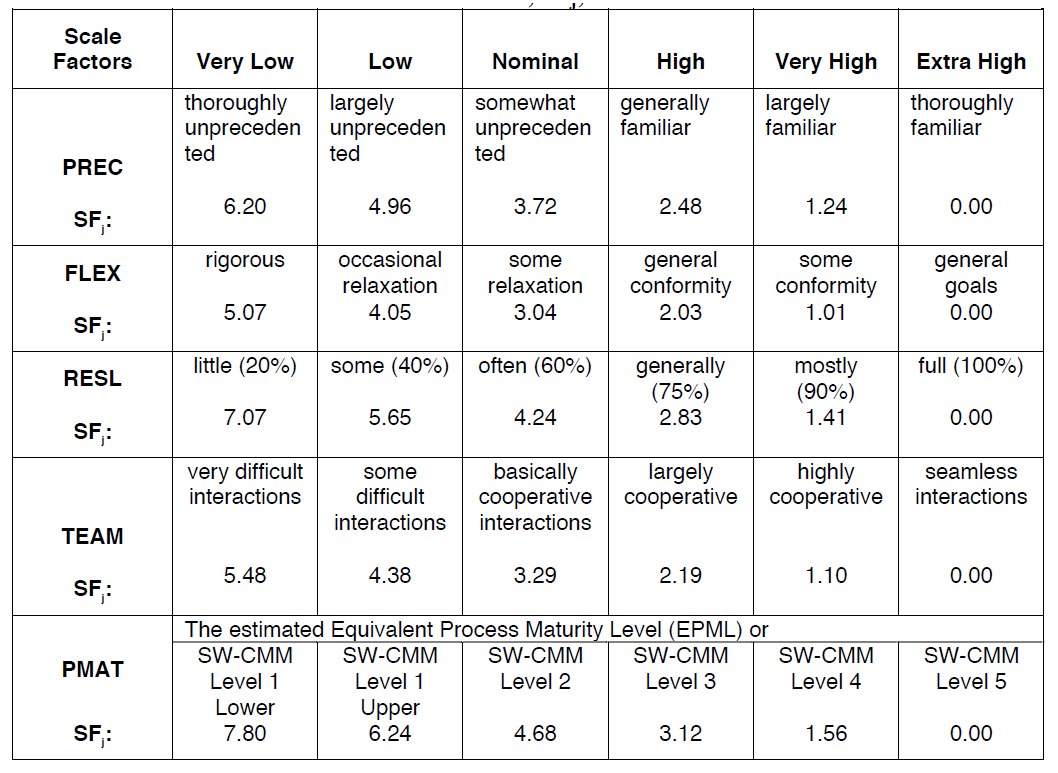
Post-Architecture: when we are extending an existing product line (we have already detailed information on cost).

Early Design: when we don’t have clear information on the architecture of the system.

We decide to follow Post-Architecture method They have an assign document with a specification of what the customer wants and this document will be redact after RASD, DD and ITPD document so we have a clearly information about the product.

2.10 Scale factor

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



Precedentedness: this product is the first one that we have projected so we don’t have experience in project development, the values will be low.

Development Flexibility: we can considerate this factor in two aspects:

* Need for software conformance with pre-established requirements: we have an assignment document with several implicit specifications and from this we create a Rasd document and define explicit requirements, so we have some requirements to achieve, the value of this sub-factor will be nominal.
* Need for software conformance with external interface specifications: in the project, will be necessary a communication among several different type of clients so the external interface will be well-defined for each type of client, providing specific service in base of client type, hence the value will be nominal.

The value of FLEX is based on these two parameters will be nominal.

Risk resolution: the risk plan that will be performed in the next chapter will be quite extensive and cover major aspects, the value will be high.

Team Cohesion: the team is formed by two persons that have already worked together for three projects, so the parameters constitute this factor:

* Consistency of stakeholder objectives and cultures.
* Willingness of stakeholders to accommodate other stakeholders’ objectives: there is always a constructive discussion about main important point of the project.
* Stakeholder teambuilding to achieve shared vision and commitments.

All this parameter will be very high value; hence team cohesion will be very high.

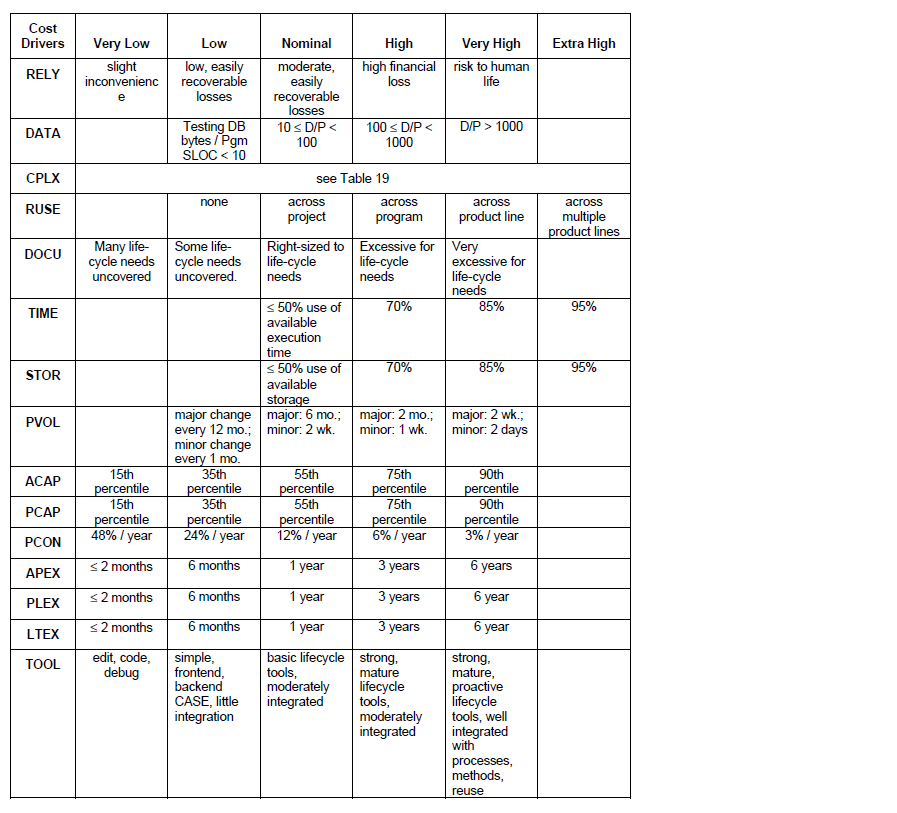
Process Maturity: for define the level of our project we use CMMI, this is the first our project so in based of our experience we think that our specification can reach level 3 of CMMI certification, this set of document (RASD, DD, ITPD and PP) have the achieve of avoiding inconsistency and incoherency among processes that provide same or different service.

The result of our evaluation is the following:

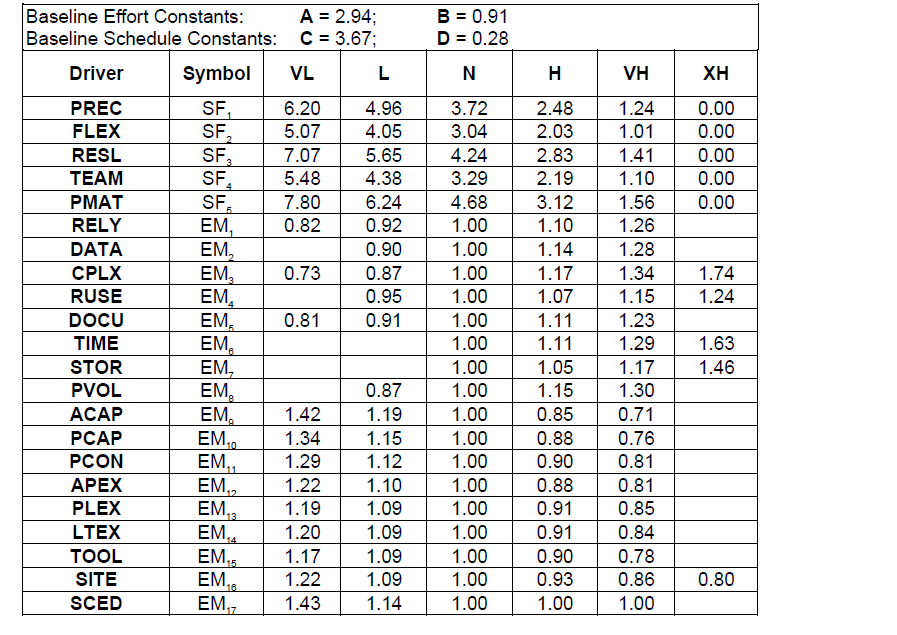
|  |  |  |
| --- | --- | --- |
| **Scale Driver** | **Factor** | **Value** |
| Precedentedness (PREC) | Low | 4.96 |
| Development flexibility(FLEX) | Nominal | 3.04 |
| Risk Resolution(RESL) | High | 2.83 |
| Team Cohesion(TEAM) | Very High | 1.10 |
| Process Maturity(PMAT) | Level 3 | 3.12 |
| **Total** |  | 15,05 |

2.11 Cost driver

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

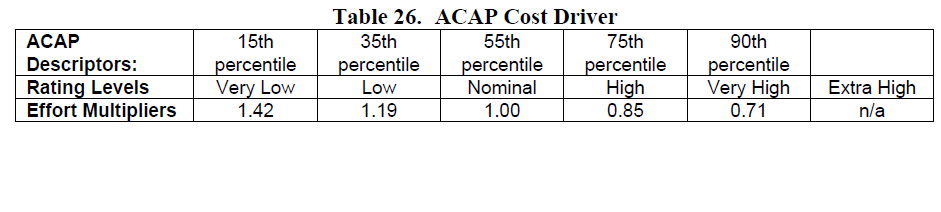


In the following table, we see the effort multiplier for each cost driver:



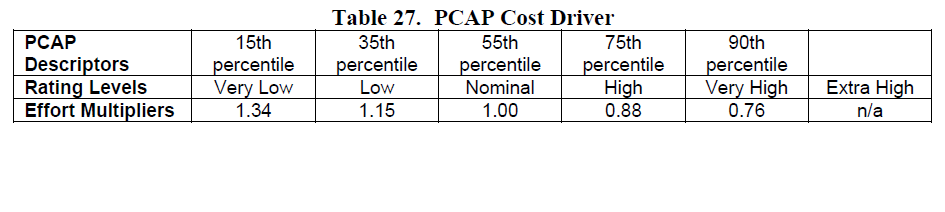
* Analyst Capability(ACAP):

The analysis of requirements that we have computed in RASD and DD document try to cover all possible aspects of the project moreover we insert “very high” in TEAM scale factor because there is a great communication among team members so the of ACAP will be high.



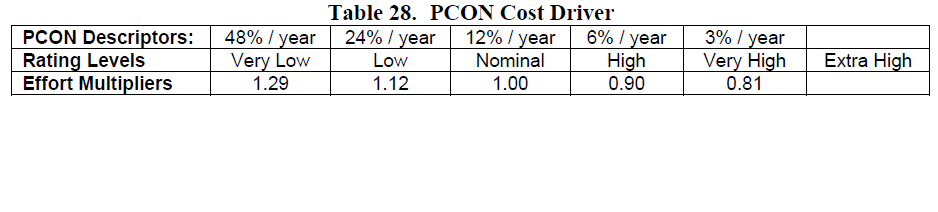
* Programmer Capability(PCAP):

This evaluation is based on the whole team capability in programming, for our team the capability is nominal because we have worked at only programming project so our experience is neither low nor high, PCAP will be nominal.



* Personnel Continuity (PCON):

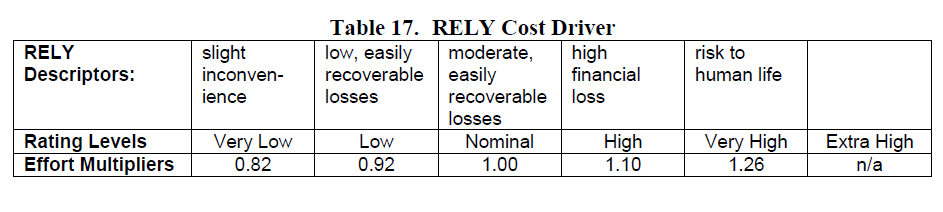
The time for this project is limited so we set the value of this parameters very low.



* Required Software Reliability(RELY):

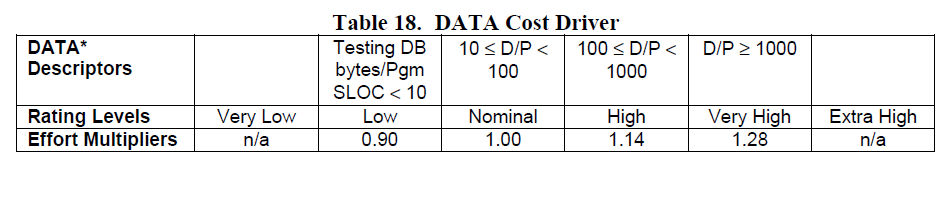
The reliability of the services is very important point for this project, a user should be performing a reservation when he wants, the major services like reservation and unlocking car must be available at any time.

If there is a software failure maybe, we can loss the client so the RELY values is set to high.



* Database size(DATA):

Database test should be quite large because our database should be memorizing all personal user data and providing the history payment data the database capture also all ride data for each user, DATA will be high.

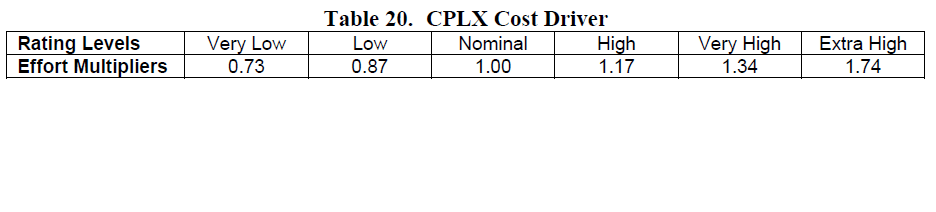


* Product Complexity(CPLX):

Complexity is divided into five points:

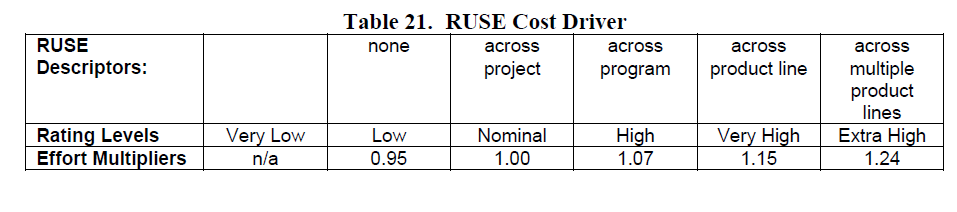
* + Control operations: the system must perform all request that clients send, only for this fact we must implement some control operation to control the incoming and outgoing stacks and the application must be supported for each type of device and system (Very-High).
  + Computational Operations: for develop the system we need to use structure like matrix and vector for controlling the queues and capture the user position for example (Nominal).
  + Device-dependent Operations: all operation I/O will be done with the simply methods GET or POST to capture the information that user sends (Low).
  + Data Management Operations: the database that we would build is a simple database containing with re-edit and change database structure is not contemplated, the DBMS manager create various type of queries (Low).
  + User Interface Management Operations: The various User interface is composed by GUI for providing the services (Low).

Through this analysis, the value of CPLX is Nominal.



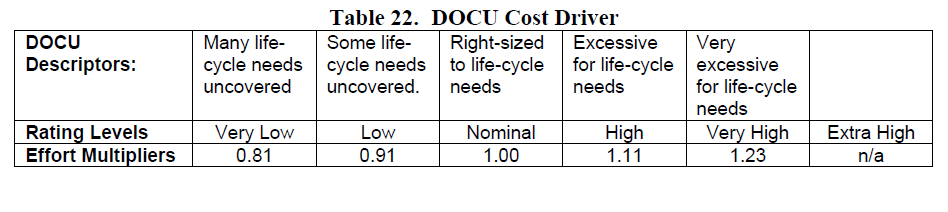
* Developed for Reusability(RUSE):

In the DD, we declare to use JEE for supporting our component architecture, one of the most advantage of using JEE is the code reusability because it’s organize in “Beans” and if the customer wants another service is enough to add a new bean and create an interaction among the other ones (Very High).



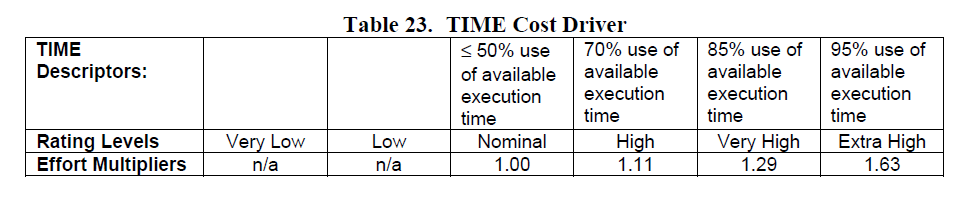
* Documentation match to life-cycle needs(DOCU):

The documentation covers all parts of the product life-cycle (Nominal) but in RUSE we set very high so the documentation will be excessive for life-cycle needs in this way we have a fully reusability(High).



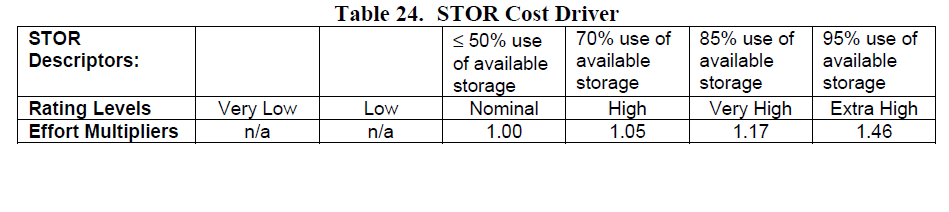
* Execution Time Constraint (TIME):

When the software system runs on server doesn’t occupy much space because there isn’t heavy process, the system just send/receive message (Nominal).



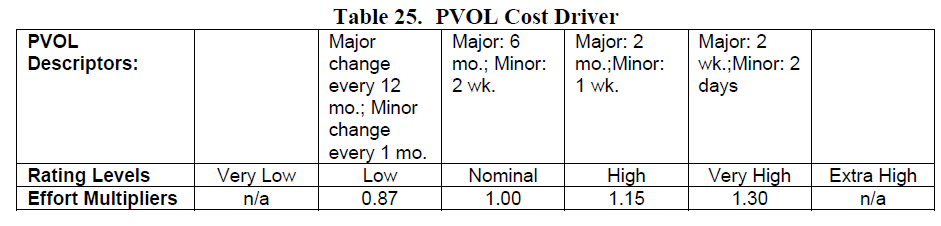
* Main storage constraint (STOR):

We think that the amount of storage usage is enough respect the availability of the hardware because when the software run occupy at most 50$ of available storage (Nominal).



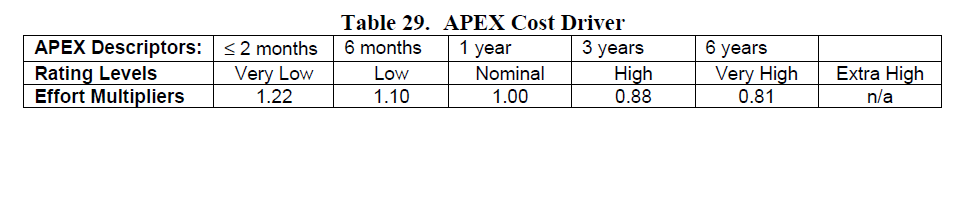
* Platform volatility (PVOL):

Various applications don’t change very often, even if the reusability of the code; maybe database maintenance every six months (Nominal)



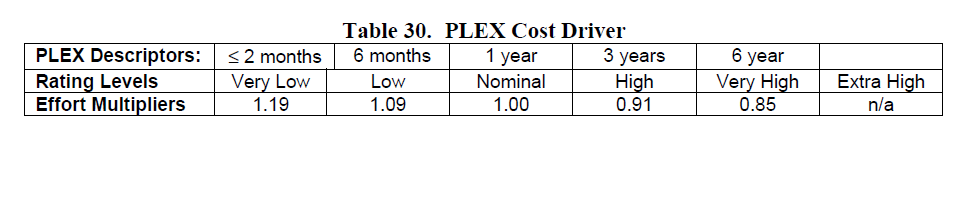
* Application Experience (APEX):

The team members work on a project only one time for six months and the focus of that project is quite different from this project so the experience value will be low.



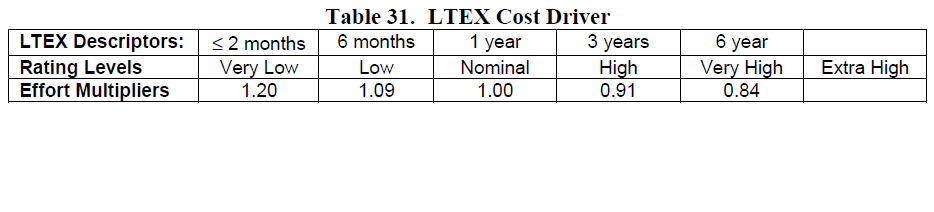
* Platform Experience (PLEX):

The team member never worked on any platform used in this project but they have a theory background on database and network platform and how these platforms are used hence the value of PLEX will be set to low.



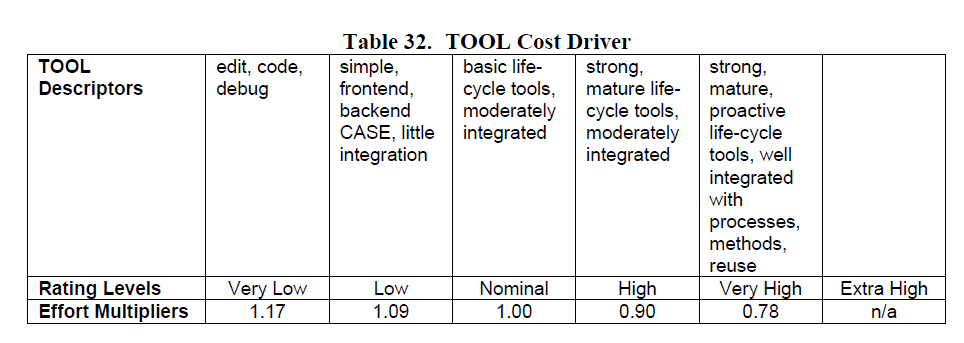
* Language and tool experience (LTEX):

The team members have a theory background on programming language, networking and database management moreover the team have just worked on a programming project the value set for LTEX will be Nominal.



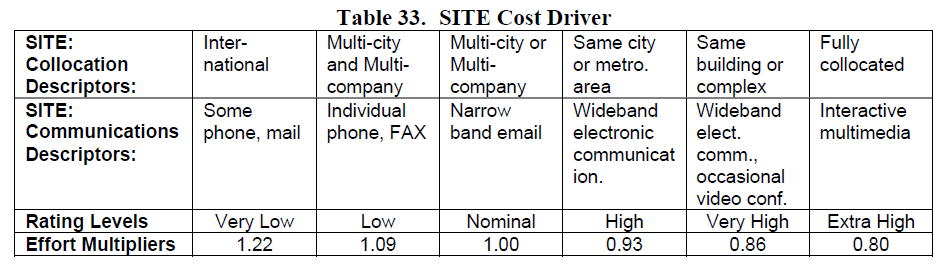
* Use of software tool (TOOL):

We use tools for the basic life-cycle of the project, this tools support the project development, the value of TOOL will be Nominal



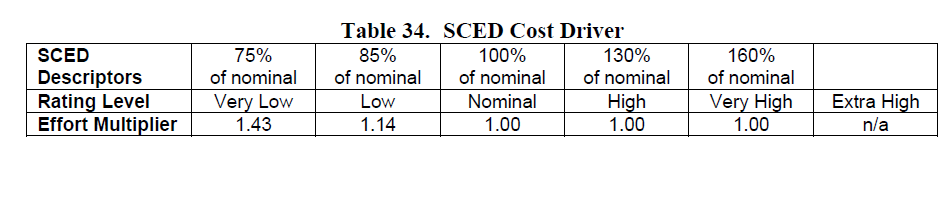
* Multisite Development (SITE):

The major part of the time the team is in the same building for discussing the project; so, the value of SITE will be very high.



* Requirement Development Schedule (SCED):

The deadline for this project is fixed and the range of time for redact each document is variable, even if we are well-balanced the work the part of requirements and the component was quite long and after these parts we must accelerate to accomplish the deadline, so the value of SCED is set to high.

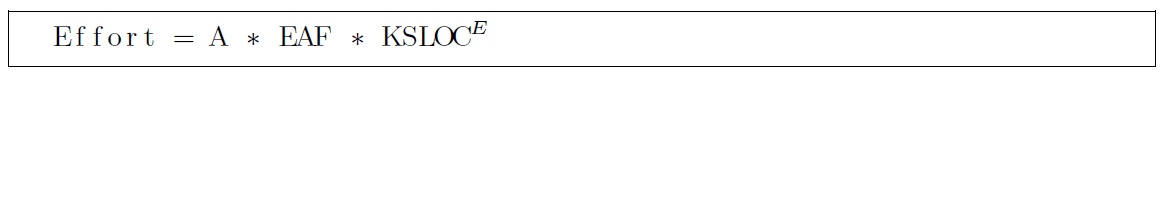


The following table resume the results of cost driver:

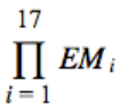
|  |  |  |
| --- | --- | --- |
| Cost Driver | Rank | Value |
| Analyst Capability(ACAP) | High | 0,85 |
| Program. Capability(PCAP) | Nominal | 1,00 |
| Person. Continuity(PCON) | Low | 1,12 |
| Required Software Reliability(RELY) | High | 1,10 |
| Database size(DATA) | High | 1,14 |
| Product Complexity(CPLX) | Nominal | 1,00 |
| Developed for Reusability(RUSE) | Very High | 1,15 |
| Documentation match to life-cycle needs(DOCU) | High | 1,11 |
| Execution Time Constraint (TIME) | Nominal | 1,00 |
| Main storage constraint (STOR) | Nominal | 1,00 |
| Platform volatility (PVOL) | Nominal | 1,00 |
| Application Experience (APEX) | Low | 1,10 |
| Platform Experience (PLEX) | Low | 1,09 |
| Language and tool experience (LTEX) | Nominal | 1,00 |
| Use of software tool (TOOL) | Nominal | 1,00 |
| Multisite Development (SITE) | Very High | 0,86 |
| Requirement Development Schedule (SCED) | High | 1,00 |
| **Total** |  | **1.5714** |

2.11 Effort estimation

The following formula permits us to calculate the effort estimation in terms of Person-Month:



A=2,94 in COCOMO II

Where EAF= =1.5714

E= exponent derived from the scale driver analysis = =0,91+0,01\*15,05=1,0605

B=0,91 in COCOMO II

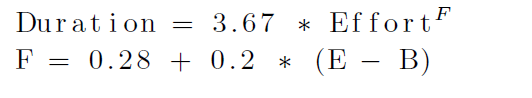
SLOC=5198=5,198KSLOC

So, with this parameter the effort estimation is:

Effort=2,94 \* 1,5714 \* 5.743KSLOC =26,532 PM=>27PM

2.12 Duration estimation

The following formula permits us to calculate the duration estimation:



F=0.28+0.2\*(1.0605-0.91) = 0.3101

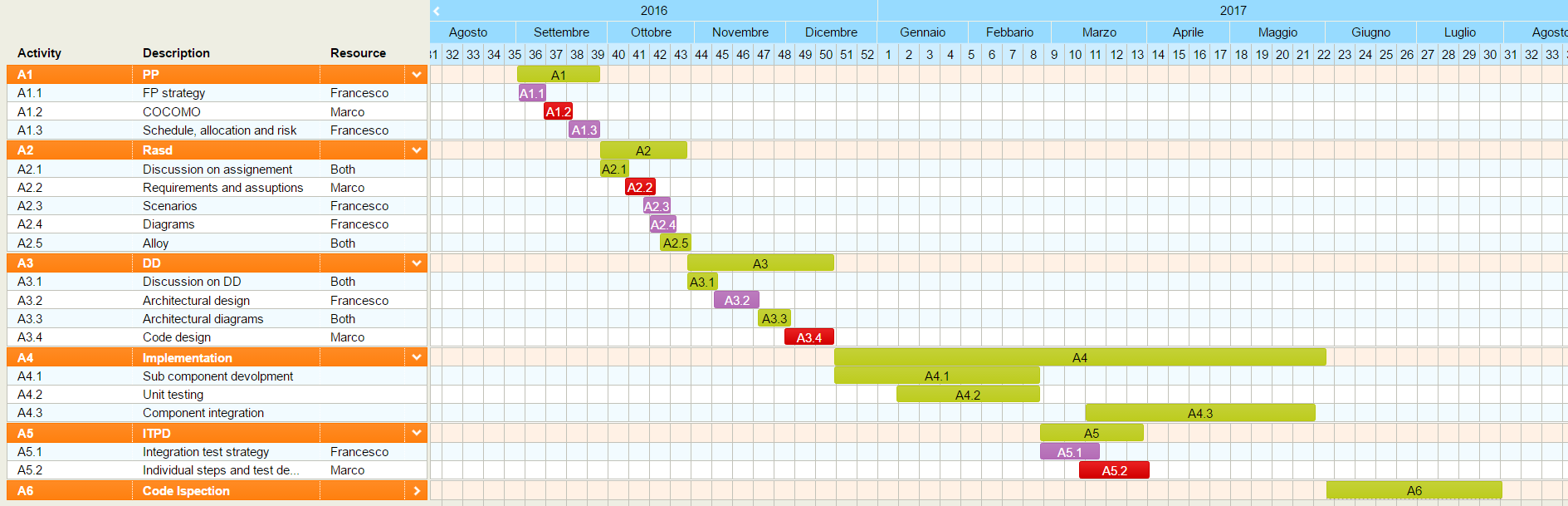
Duration= 3.67 \* 26.532 PM ^ 0.3101=10,15 months => 11months

1. Schedule

In this part, we provide a scheduling of the work that we have divided in task.

Every task has been assigned or to Francesco Tinarelli or to Marco Wenzel or to both. We take in consideration the time estimation of COCOMO: 11 months.

The following graphics represent the schedule.



1. Project risk

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Probability** | **Strategy** | **Effects** |
| The Client don`t accept the  requirements during the discussion after the ending of the Rasd document. | Moderate. The client can ask more than the requirements offers | Change the requirements analysis | Serious. We must redo the work on the requirements, time get shorter |
| The city change rules for the availability of some street in the center of the city | Moderate. It`s possible that the municipality decide to lock the traffic in some parts of the city for a better circulation. | Ask to the municipality for a normal circulation even in that part of the city. | Serious. We can lose client if we don`t find a solution with the municipality |
| Another company creates an electric car sharing system. | High. In this period a lot of car sharing system are springing up. | Add some functionalities that can improve the competition with the other company | Serious. We must improve functionalities for not losing client |
| Server belongs too slow, cause the lots of user | Low. The server can support a lot of client | Improve the server without create problem for the user. | Catastrophic. We must probably stop the service for same day due to fix the server and improve it. |
| Database can`t store other user, cause lots of him | Low. The database is  made for lots of user | Enlarge the database | Serious. We have to pay lots for enlarge the database |
| The client says that the architecture we wont to realize it`s too much expensive | Low. The client tells us the maximum price, if the client doesn’t change it we stay under that price | Change the architecture for stay under the established price | Catastrophic. We must redo all the previous work and reschedule all the work. Lots of time will be lost |

1. Appendix

5.1 References

* Project Management Basics + Advanced Dec. 1
* 5\_PPD\_rev11
* Tom's planner (<https://www.tomsplanner.com/>)

5.2 Hours of work

Francesco Tinarelli: 12 Hours

Marco Wenzel: 12 Hours