PowerEnJoy
Project Plan Document
Software Engineering 2 project

POLITECNICO
MILANO 1863

Authors:
Arcari Leonardo
Bertoglio Riccardo
Galimberti Andrea

January 22, 2017

Document version: 1.0

Contents

1 Introduction

1.1 Revision History

- 22/01/2017 - Version 1.0

1.2 Purpose

The purpose of this document is to describe how the PowerEnjoy project has to be carried out in regards to managing cost, time, resources and risks.

It first provides an estimation of the size of the project code by making use of the Function Points and an estimation of the cost and the required effort by using the COCOMO II model.

A schedule is also estimated for all the activities concerning the project, and the available personnel resources are properly allocated based on the schedule and the resources' availability.

Finally, risk management is thoroughly assessed to have a complete view of which are the main known and predictable risks to the positive outcome of the project, and how to mitigate them through appropriate strategies.

1.3 Scope

The required product is a digital management system for PowerEnJoy, a car sharing service employing exclusively electric vehicles.

The system has to provide functionalities to support both the users of the service and the employees of the company who interact with customers and cars.

Typical functions of a car-sharing service have to be supported, such as reserving a car and finding a parking area where to plug it in at the end of the rent.

A critical issue that has to be considered is the management of the electric vehicles, which have to be continuously recharged as they get used by customers and have to be distributed in a quite uniform way around the town; hence, the user has to be incentivized through discounts to recharge the car in power plugs-equipped parking areas, don't leave the car with a low battery level and park the car where there are few other available vehicles.

The product also has to provide a way to attract more potential customers to the service, therefore sharing the car with other passengers will be incentivized through appropriate discounts.

To achieve such functionalities, the system shall provide applications to its users, both customers and employees.

1.4 Reference Documents

- PowerEnjoy Assignment Document
- PowerEnjoy Requirements Analysis and Specifications Document
- PowerEnjoy Design Document
- PowerEnjoy Integration Testing Document
- Cocomo II Model Definition Manual, version 2.1, freely available at `http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf`

- IFPUG Function Point Counting Practices Manual, release 4.1.1
- Function Point Languages Table, version 5.0, freely available at `http://www.qsm.com/resources/function-point-languages-table`
- myTaxiService Project Plan Document example by Casati, Castelli

2 Size Estimation: Function Points

2.1 Internal Logic Files (ILF)

- Customer: 3 RETs, 16 DETs
  - ID
  - Personal data: First name, Last name, Date of birth, Place of birth, Gender, Residence, Fiscal code, ID card number, Driving license number, Mobile phone number
  - Data to access the personal area: Email address, Password
  - Payment details: Credit card number, Security code, Expiration date
- Car: 1 RET, 7 DETs
  - ID, Licence plate, Battery level, Locked, Available, GPS Location, *Safe area
- Reservation: 1 RET, 5 DETs
  - ID, Timestamp, Expiration, *Customer, *Car
- Safe area: 1 RET, 6 DETs
  - ID, Location, Number of spots, Number of plugs, Available spots, Available plugs
- City area: 2 tables, one containing all zones with their identifiers and the other containing geographic coordinates pairs stored as (latitude, longitude). 1 RET, 3 DETs
- Rental: 1 RET, 6 DETs
  - ID, Fee, Number of passengers, Duration, Kilometers traveled, *Reservation
- Assistance request: 1 RET, 7 DETs
  - ID, *Assistance service operator, *Road operator, *Car, Description, Position, Type, State
- Assistance service operator: 2 RETs, 13 DETs
  - ID
  - Personal data: First name, Last name, Date of birth, Place of birth, Gender, Residence, Fiscal code, ID card number, Mobile phone number
  - Data to access the system: Email address, Password
  - Level of privileges
- Road operator: 2 RETs, 13 DETs
  - ID
  - Personal data: First name, Last name, Date of birth, Place of birth, Gender, Residence, Fiscal code, ID card number, Driving license number, Mobile phone number
  - Data to access the system: Email address, Password

ILF	Complexity	FPs
Customer	Low	7
Car	Low	7
Reservation	Low	7
Safe area	Low	7
City area	Low	7
Rental	Low	7
Assistance request	Low	7
Assistance service operator	Low	7
Road operator	Low	7
Total		63

2.2 External Logic Files (ELF)

EIF	Complexity	FPs
Map service	Average	10
Payment service	Low	7
Total		17

2.3 External Inputs (EI)

Here follows a list of External Inputs divided by the component that generates them.

All users:

- Login/Logout
  - 1 FTR
    - Customer
  - 4 DETs
    - Email address, Password
    - response message
    - GO button

Customer:

- Register to the service
  - 1 FTR
    - Customer
  - 18 DETs
    - All fields of the Customer (16 fields)

        - response message
        - REGISTER button

- Reserve a car
    - 1 FTR
        - Reservation
    - 4 DETs
        - Customer ID, Car ID
        - response message
        - RESERVE button
- Unlock a car
    - 2 FTRs
        - Reservation, Car
    - 2 DETs
        - car ID
        - UNLOCK button
- Rental end
    - 1 FTR
        - Rental
    - 6 DETs
        - ID, Fee, Number of passengers, Duration, Kilometers traveled
        - RENTAL END button

Assistance service operator:

- Open an Assistance request
    - 1 FTR
        - Assistance request
    - 9 DETs
        - ID, Assistance service operator, Road operator, Car, Description, Position, Type, State
        - OPEN button
- Unlock car
    - 2 FTRs
        - Car, Assistance service operator (to check for the level of privileges)
    - 3 DETs
        - Car ID, Assistance service operator ID

        - UNLOCK button

Road operator:

- Accept an Assistance request
    - 1 FTR
        - Assistance request
    - 3 DETs:
        - Road operator ID, Assistance request ID
        - ACCEPT button
- Unlock car
    - 1 FTR
        - Car
    - 3 DETs
        - Road operator ID, Car ID
        - UNLOCK button

Car:

- Car status update
    - 2 FTRs
        - Car, Rental
    - 10 DETs
        - Car ID, Battery level, Locked, Available, GPS Location, Rental ID, Fee, Number of passengers, Duration, Kilometers traveled

EI	Complexity	FPs
All users		
Login	Low	3
Logout	Low	3
Customer		
Register to the service	Average	4
Reserve a car	Low	3
Unlock a car	Low	3
Rental end	Low	3
Assistance service operator		
Open an Assistance request	Low	3
Unlock car	Low	3
Road operator		
Accept an Assistance request	Low	3
Unlock car	Low	3
Car		
Car status update	Average	4
Total		35

2.4 External Inquiries (EQ)

Here follows a list of External Inquiries divided by the component that receives them.

Customer:

- View parked cars in a safe area
  - 1 FTR
    - Car
  - 7 DETs
    - Enter: Safe area ID
    - Exit: Location, Number of spots, Number of plugs, Available spots, Available plugs
    - SAFE AREA icon
- View reservation details
  - 1 FTR
    - Reservation
  - 6 DETs
    - Enter: Customer ID
    - Exit: Reservation ID, Timestamp, Expiration, Car ID

    - * RESERVATION swipe

- View rental details
    - 1 FTR
        - * Rental
    - 7 DETs
        - * Enter: Customer ID
        - * Exit: Rental ID, Fee, Number of passengers, Duration, Kilometers traveled
        - * RENTAL swipe

Assistance service operator:

- View all Assistance requests
    - 1 FTR
        - * Assistance request
    - 9 DETs
        - * Exit: Assistance request ID, Assistance service operator ID, Road operator ID, Car ID, Description, Position, Type, State
        - * VIEW ASSISTANCE REQUESTS button

EQ	Complexity	FPs
	Customer	
View parked cars in a safe area	Low	3
View reservation details	Low	3
View rental details	Low	3
	Assistance service operator	
View all Assistance requests	Low	3
Total		12

2.5 External Outputs (EO)

Here follows a list of External Outputs divided by the component that receives them.

Customer:

- View near safe areas
    - 1 FTR
        - * Safe area
    - 4 DETs

    - Enter: Customer position, Customer destination
    - Exit: Safe Area ID, Location

- Money saving option
  - 1 FTR
    - Car
  - 8 DETs
    - Enter: Destination
    - Exit: Safe Area ID, Location, Number of spots, Number of plugs, Available spots, Available plugs
    - MONEY SAVING OPTION switch activated

Assistance operator:

- View list of faulty cars
  - 2 FTR
    - Car, Assistance request
  - 12 DETs
    - Exit: Car ID, Licence plate, Battery level, Locked, Available, GPS Location, Assistance Request ID, Description, Position, Type, State
    - VIEW button

Road operator:

- Receive an assistance request
  - 1 FTR
    - Assistance request
  - 6 DETs
    - Exit: Assistance Request ID, Car Licence plate, Description, Position, Type, State

EO	Complexity	FPs
Customer		
View safe areas	Low	4
Money saving option	Low	4
Assistance service operator		
View list of faulty cars	Average	5
Road operator		
Receive an assistance request	Low	4
Total		17

2.6 Overall Estimation

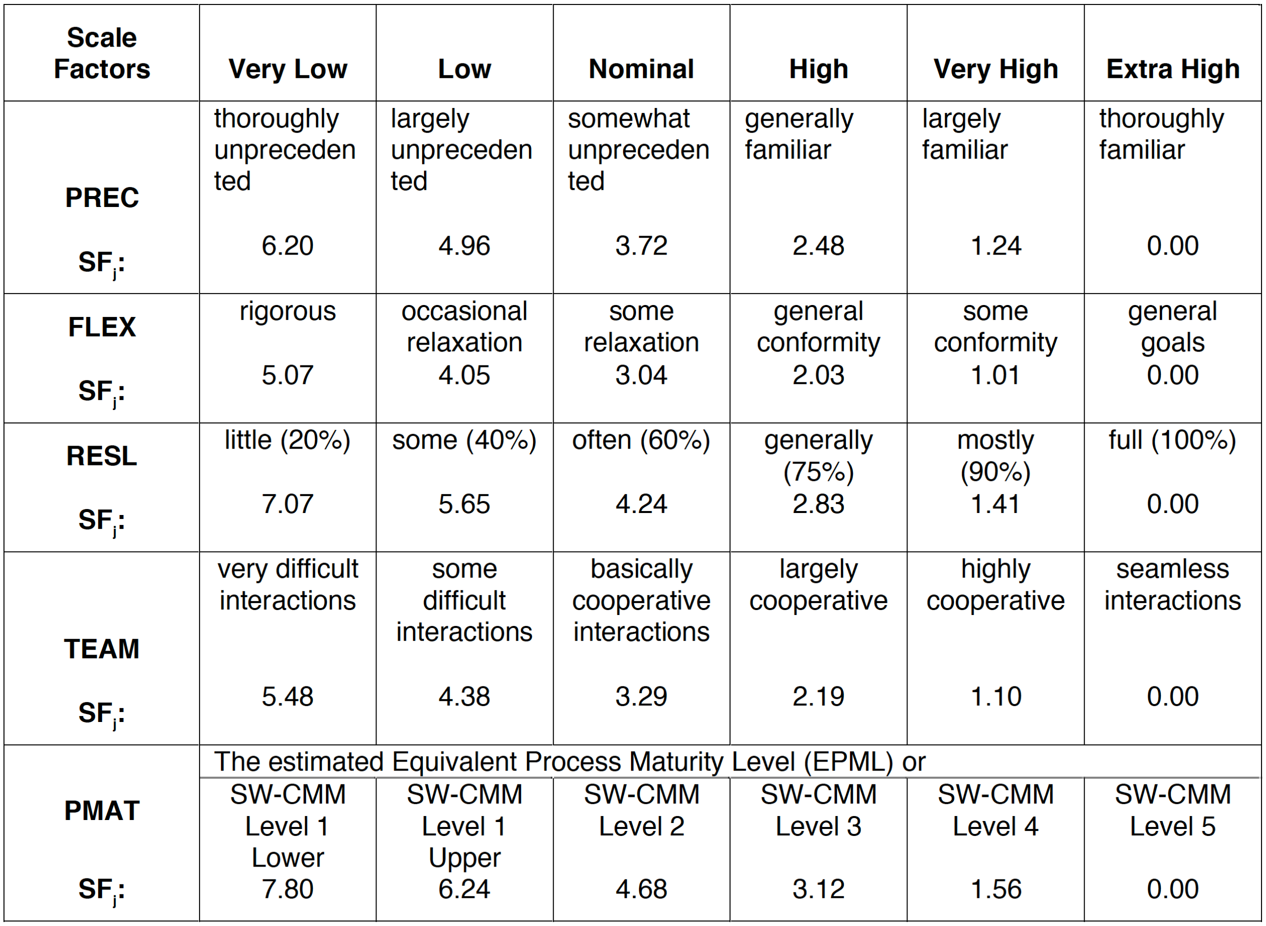
Function type	FP
ILF	63
EIF	17
EI	35
EO	17
EQ	12
Total	144

In order to compute the SLOC (Source Lines Of Code) we multiply the total number of function points by the AVC, i.e., an average number of lines of code specific for any programming language, in our case JavaEE.

$$SLOC_{min} = FP_{total} * AVC_{min} = 144 * 15 = 2160$$
$$SLOC_{avg} = FP_{total} * AVC_{avg} = 144 * 46 = 6624$$
$$SLOC_{max} = FP_{total} * AVC_{max} = 144 * 67 = 9648$$

3 Cost and effort estimation: COCOMO II

3.1 Scale Factors



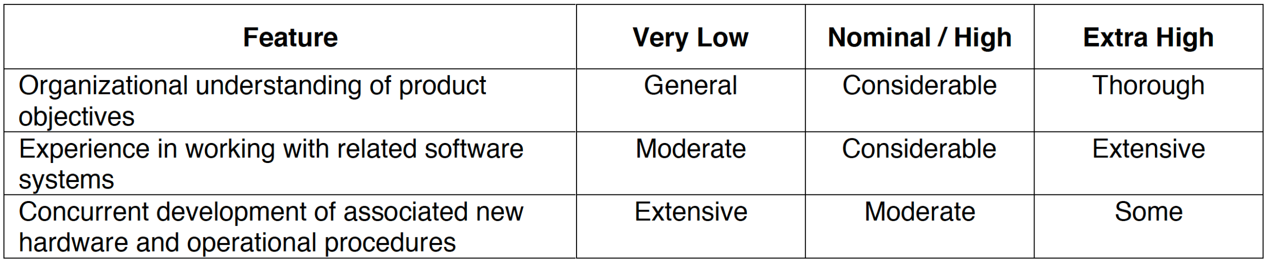
Scale Factors	Very Low	Low	Nominal	High	Very High	Extra High
PREC SF_j:	thoroughly unprecedented 6.20	largely unprecedented 4.96	somewhat unprecedented 3.72	generally familiar 2.48	largely familiar 1.24	thoroughly familiar 0.00
FLEX SF_j:	rigorous 5.07	occasional relaxation 4.05	some relaxation 3.04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

Figure 1: Scale Factor Values for COCOMO II Models

3.1.1 Precedentedness (PREC)

The PREC scale factor measures the similarity of the current project to previously developed ones.

In our case, PowerEnjoy is the first project of this kind for us, hence, although we have a good understanding of the goals, we practically start from scratch in its development; therefore, we set the corresponding rating level to *Low*.



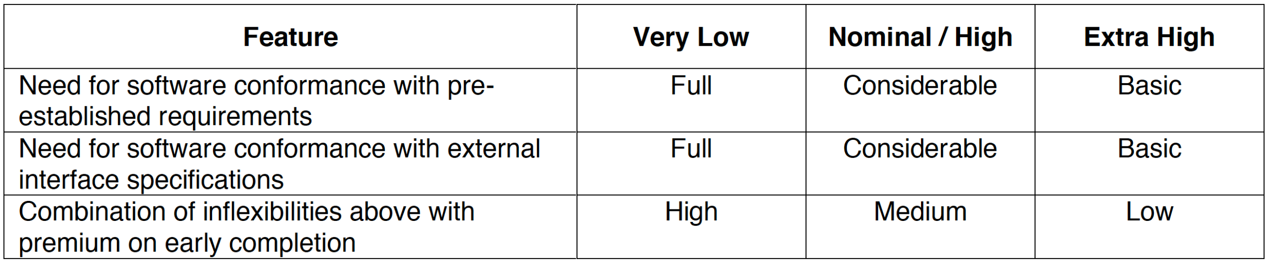
Feature	Very Low	Nominal / High	Extra High
Organizational understanding of product objectives	General	Considerable	Thorough
Experience in working with related software systems	Moderate	Considerable	Extensive
Concurrent development of associated new hardware and operational procedures	Extensive	Moderate	Some

Figure 2: PREC Scale Driver

3.1.2 Development Flexibility (FLEX)

This scale factor expresses the flexibility with regard to pre-established requirements and externale interface specifications.

It is evaluated as having a *Nominal* rating level, as one of the critical features of the system is interfacing with the car's hardware, hence there is a low flexibility for external interfaces, while there is very high flexibility for pre-established requirements and there is not any considerable premium on early completion.



Feature	Very Low	Nominal / High	Extra High
Need for software conformance with pre-established requirements	Full	Considerable	Basic
Need for software conformance with external interface specifications	Full	Considerable	Basic
Combination of inflexibilities above with premium on early completion	High	Medium	Low

Figure 3: FLEX Scale Driver

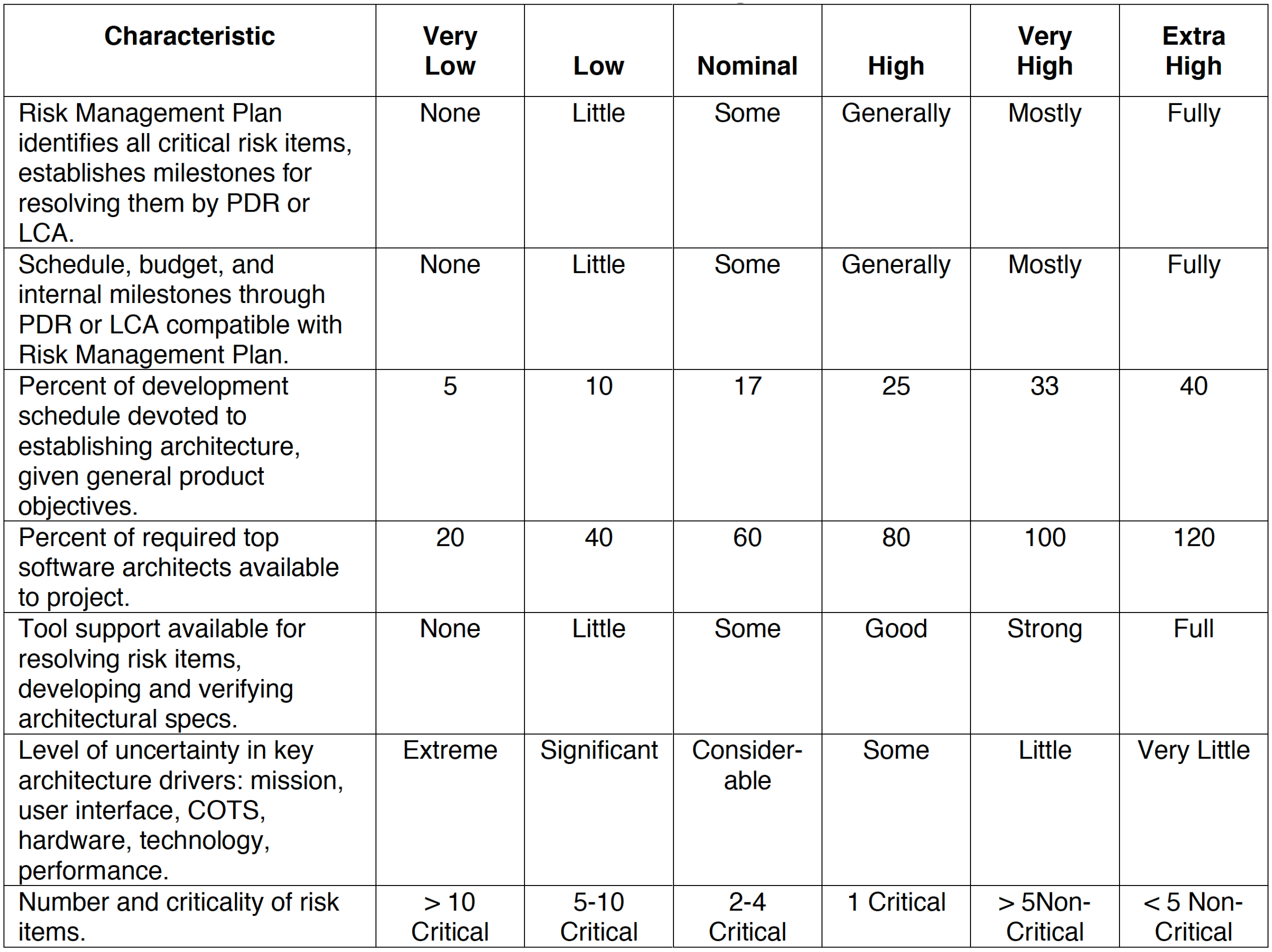
3.1.3 Architecture / Risk Resolution (RESL)

This rating measures the depth level of architecture definition and risk management .

For the PowerEnjoy project, both the architecture and possible risks were thoroughly studied, with very little uncertainty and, with regard to risks, defining appropriate strategies to deal with them.

Also, a quite large number of critical risks was found.

The RESL scale factor is assigned a *High* rating level.



Characteristic	Very Low	Low	Nominal	High	Very High	Extra High
Risk Management Plan identifies all critical risk items, establishes milestones for resolving them by PDR or LCA.	None	Little	Some	Generally	Mostly	Fully
Schedule, budget, and internal milestones through PDR or LCA compatible with Risk Management Plan.	None	Little	Some	Generally	Mostly	Fully
Percent of development schedule devoted to establishing architecture, given general product objectives.	5	10	17	25	33	40
Percent of required top software architects available to project.	20	40	60	80	100	120
Tool support available for resolving risk items, developing and verifying architectural specs.	None	Little	Some	Good	Strong	Full
Level of uncertainty in key architecture drivers: mission, user interface, COTS, hardware, technology, performance.	Extreme	Significant	Consider-able	Some	Little	Very Little
Number and criticality of risk items.	> 10 Critical	5-10 Critical	2-4 Critical	1 Critical	> 5Non-Critical	< 5 Non-Critical

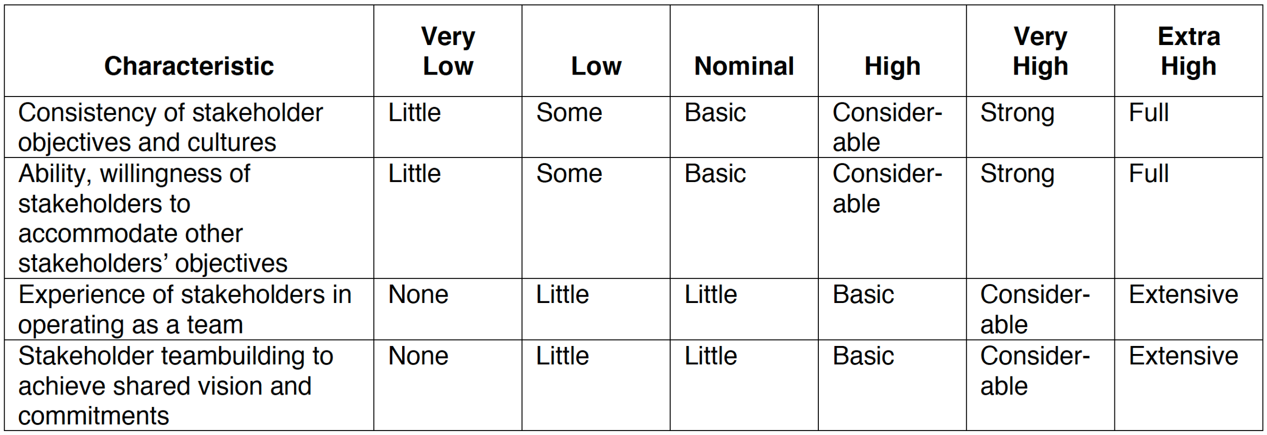
Figure 4: RESL Scale Driver

3.1.4 Team Cohesion (TEAM)

The TEAM scale factor accounts for the sources of project turbulence and entropy because of difficulties in synchronizing the project's stakeholders (users, customers, developers, maintainers, interfacers, etc.).

There was a constant feedback from the customer, facilitating work on the project developers' side, teamwork was proficient and there was a shared vision for how to proceed in all the steps of the project development.

Hence, the TEAM is assigned a *Very High* rating level.



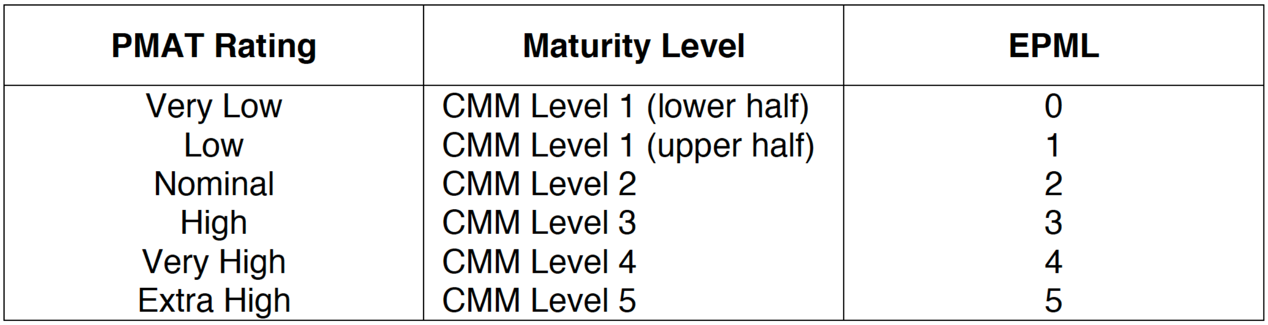
Characteristic	Very Low	Low	Nominal	High	Very High	Extra High
Consistency of stakeholder objectives and cultures	Little	Some	Basic	Consider-able	Strong	Full
Ability, willingness of stakeholders to accommodate other stakeholders' objectives	Little	Some	Basic	Consider-able	Strong	Full
Experience of stakeholders in operating as a team	None	Little	Little	Basic	Consider-able	Extensive
Stakeholder teambuilding to achieve shared vision and commitments	None	Little	Little	Basic	Consider-able	Extensive

Figure 5: TEAM Scale Driver

3.1.5 Process Maturity (PMAT)

The procedure for determining PMAT is organized around the Software Engineering Institute's Capability Maturity Model (CMM).

As we are not an already mature organization, but just three unexperienced students working on a one-time project, the PMAT scale factor is assigned a *Very Low* rating level.



PMAT Rating	Maturity Level	EPML
Very Low	CMM Level 1 (lower half)	0
Low	CMM Level 1 (upper half)	1
Nominal	CMM Level 2	2
High	CMM Level 3	3
Very High	CMM Level 4	4
Extra High	CMM Level 5	5

Figure 6: PMAT Scale Driver

3.2 Effort Multipliers: Post-Architecture Cost Drivers

The Post-Architecture Cost Drivers are divided in:

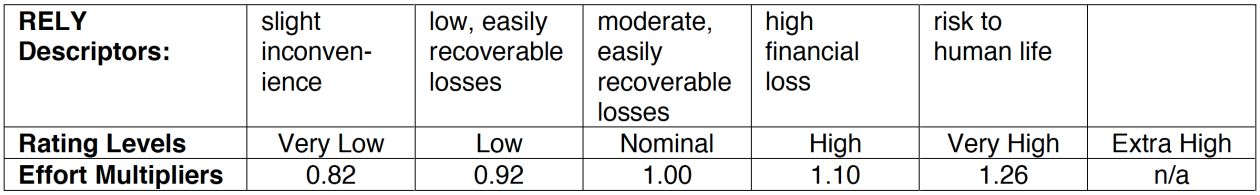
- Product Factors
- Platform Factors
- Personnel Factors
- Project Factors

Following are the Product Factors:

3.2.1 Requested Sotware Reliability (RELY)

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

The PowerEnjoy service is estimated to require an high amount of reliability, to guarantee the highest possible availability of the service for the end users, therefore we choose to assign the RELY driver a *Very High* rating level.



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

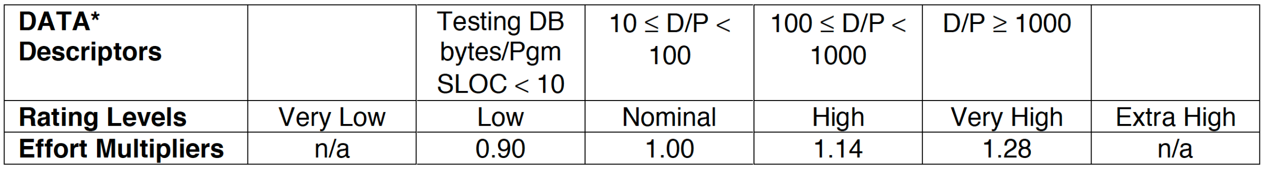
Figure 7: RELY Cost Driver

3.2.2 Data Base Size (DATA)

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

An estimate to the test database size needed for appropriate testing leads to an approximate value in the order of 10MB, while the estimated average value of SLOC is 6624 (from the Function Points estimation).

Hence the D/P ratio is equal to $D/P = 10000000/6624 = 1500$, and the DATA cost driver has a *Very High* rating level.



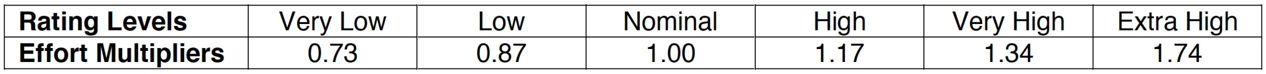
DATA* Descriptors		Testing DB bytes/Pgm SLOC < 10	10 ≤ D/P < 100	100 ≤ D/P < 1000	D/P ≥ 1000	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.90	1.00	1.14	1.28	n/a

Figure 8: DATA Cost Driver

3.2.3 Product Complexity (CPLX)

The CPLX cost driver measures the complexity of control operations, computational operations, device-dependent operations, data management operations, and user interface management operations.

Referring to Table 19 from the COCOMO II Model Definition Manual and computing a subjectively weighted average of the five different aspects, we identify a *High* CPLX rating level.



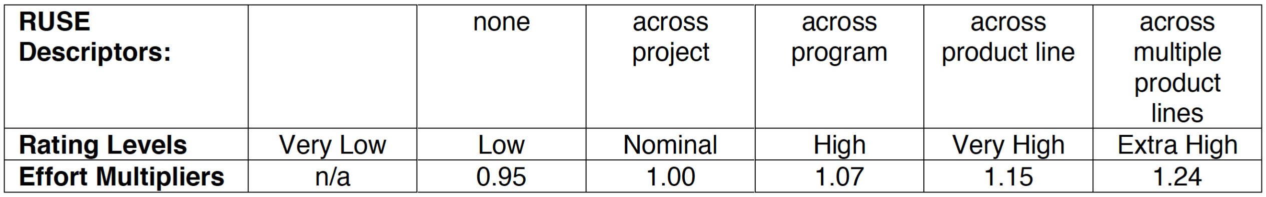
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.73	0.87	1.00	1.17	1.34	1.74

Figure 9: CPLX Cost Driver

3.2.4 Developed for Reusability (RUSE)

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

In this project the reusability will be limited to the scope of this project, because there won't be any other concurrent projects now or in the future to share modules with, hence we assign the RUSE driver a *Nominal* rating level.



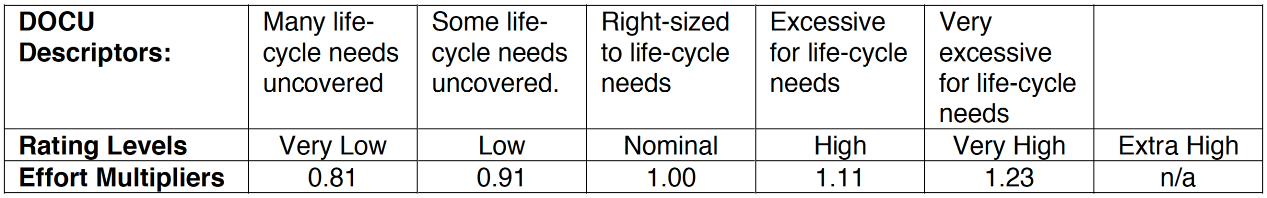
RUSE Descriptors:		none	across project	across program	across product line	across multiple product lines
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	n/a	0.95	1.00	1.07	1.15	1.24

Figure 10: RUSE Cost Driver

3.2.5 Documentation Match to Life-Cycle Needs (DOCU)

This cost driver measures the level of required documentation, and is evaluated in terms of the suitability of the project's documentation to its life-cycle needs.

It is evaluated as having a *Nominal* rating level, as every aspect of the project will be appropriately covered in the documentation.



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

Figure 11: DOCU Cost Driver

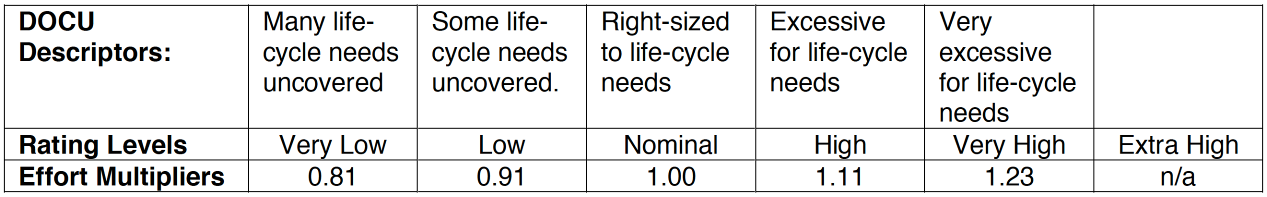
Following are the Platform Factors:

3.2.6 Execution Time Constraint (TIME)

This is a measure of the execution time constraint imposed upon a software system.

The rating is expressed in terms of the percentage of available execution time expected to be used by the system or subsystem consuming the execution time resource.

We expect the PowerEnjoy service to require a high level of the allocated CPU resources, hence we assign it a *High* rating level.



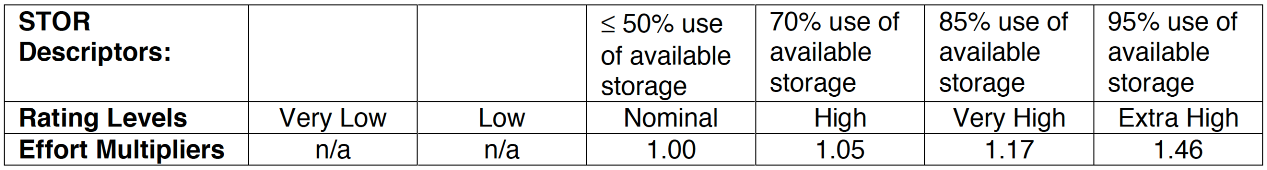
DOCU Descriptors:	Many life-cycle needs uncovered	Some life-cycle needs uncovered.	Right-sized to life-cycle needs	Excessive for life-cycle needs	Very excessive for life-cycle needs	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.81	0.91	1.00	1.11	1.23	n/a

Figure 12: TIME Cost Driver

3.2.7 Main Storage Constraint (STOR)

This rating represents the degree of main storage constraint imposed on a software system or subsystem.

We don't expect the developed service to require a significantly high amount of storage, so it is evaluated as having a *Nominal* rating level.



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

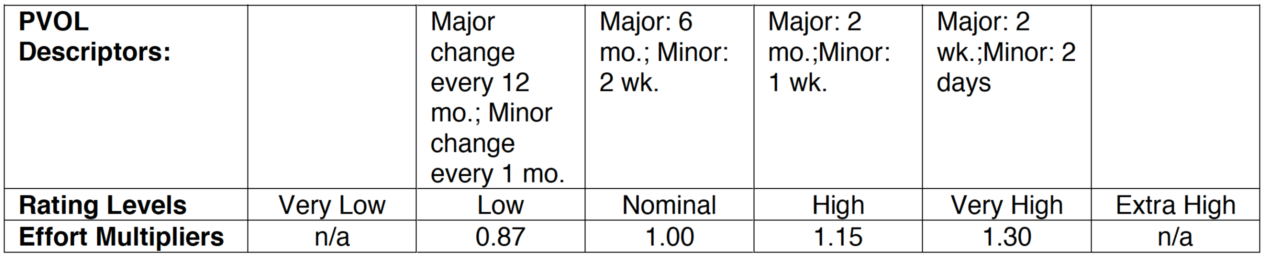
Figure 13: STOR Cost Driver

3.2.8 Platform Volatility (PVOL)

This cost driver represents how often there are significant changes in the complex of hardware and software (OS, DBMS, etc.) the software product calls on to perform its tasks.

We don't foresee any major changes more often than every 12 months, so to maintain especially on the server side a good system stability ,as its requirements and features aren't expected to change that fast, hence using an already established platform is deemed the best choice; a period of 12 months between any major changes to the platform also corresponds to the usual time spanning between major updates to the most common operating systems.

Therefore the PVOL cost driver is assigned a *Low* rating level.



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

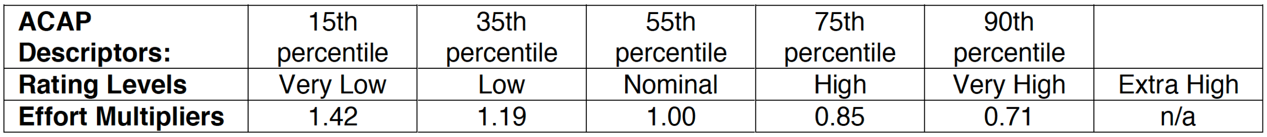
Figure 14: PVOL Cost Driver

Following are the Personnel Factors:

3.2.9 Analyst Capability (ACAP)

This cost driver evaluates the analysis and design ability, efficiency and thoroughness, and the ability to communicate and cooperate of the personnel who work on requirements, high-level design and detailed design.

We assign it a *Nominal* rating level, as both the Requirements Analysis and the Architecture Design have been dealt with a sufficient amount of precision and completeness.



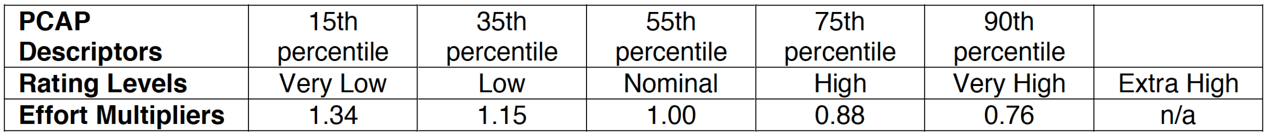
ACAP Descriptors:	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.42	1.19	1.00	0.85	0.71	n/a

Figure 15: ACAP Cost Driver

3.2.10 Programmer Capability (PCAP)

This cost driver measures the capability of the programmers as a team rather than as individuals; major factors which should be considered in the rating are ability, efficiency and thoroughness.

The PCAP driver is evaluated as having a *Nominal* rating level, as we estimate our programming abilities to be substantially average.



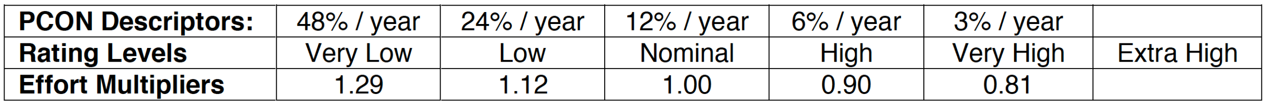
PCAP Descriptors	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.34	1.15	1.00	0.88	0.76	n/a

Figure 16: PCAP Cost Driver

3.2.11 Personnel Continuity (PCON)

This cost driver is measured in terms of the project's annual personnel turnover.

We evaluate it at a *Very High* rating level, as no one of the members is absolutely expected to abandon the project.



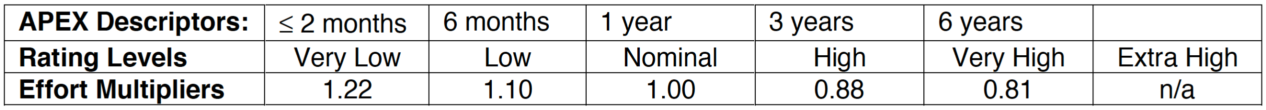
PCON Descriptors:	48% / year	24% / year	12% / year	6% / year	3% / year	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.29	1.12	1.00	0.90	0.81	

Figure 17: PCON Cost Driver

3.2.12 Application Experience (APEX)

This rating is dependent on the level of applications experience of the project team developing the software system or subsystem, and it is defined in terms of the project team's equivalent level of experience with this type of application.

We have been dealing with the JavaEE platform in these last months, and we all have developed a smaller-scale distributed Java application for the graduation project last year, so we deem appropriate to assign this cost driver a *Low* rating level.



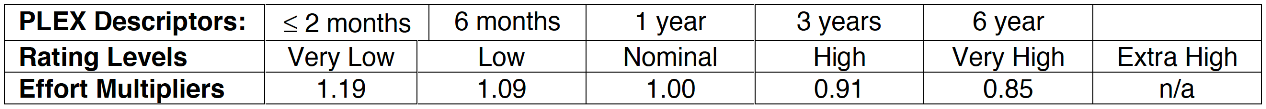
APEX Descriptors:	≤ 2 months	6 months	1 year	3 years	6 years	
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.22	1.10	1.00	0.88	0.81	n/a

Figure 18: APEX Cost Driver

3.2.13 Platform Experience (PLEX)

This cost driver measures the team's understanding of the use of more powerful platforms, including more graphic user interface, database, networking, and distributed middleware capabilities.

We don't have any significant experience for what concerns the platform on which we are going to build our service, therefore the PLEX cost driver is assigned a *Very Low* rating level.



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

Figure 19: PLEX Cost Driver

3.2.14 Language and Tool Experience (LTEX)

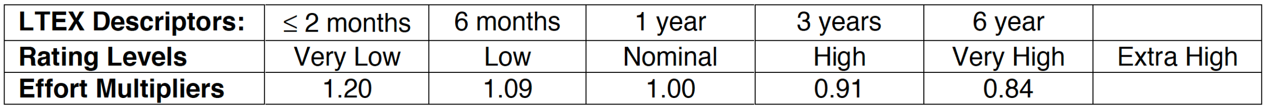
This is a measure of the level of programming language and software tool experience of the project team developing the software system or subsystem.

Considered tools are ones that perform requirements and design representation and analysis, configuration management, document extraction, library management, program style and formatting, consistency checking, planning and control.

We have some good experience on some tools (e.g. UML modeling of the system, Git versioning control of the project), while others are totally unknown.

For what concerns the JavaEE programming language, we have a sufficient understanding of how to exploit its features, also building on our good knowledge level of the Java (version 8) language.

The LTEX is then assigned a *Low* rating level.



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

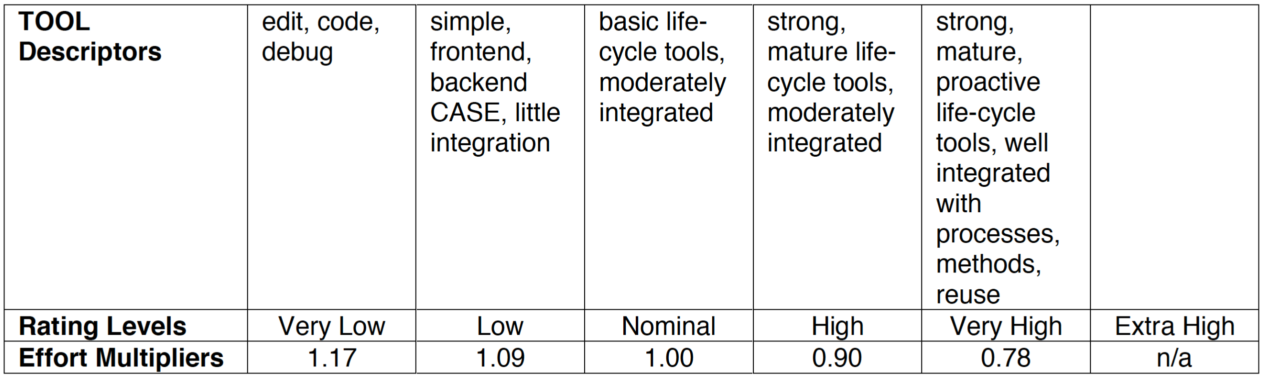
Figure 20: DOCU Cost Driver

Following are the Project Factors:

3.2.15 Use of Software Tools (TOOL)

This cost driver measures the extent of the functionalities provided by software tools for this project, with its rating ranging from simple edit and code, very low, to integrated life-cycle management tools, very high.

We consider the tools used for the design, development, testing and deployment of the PowerEnjoy service to be strong, mature and integrated enough to attain a *High* TOOL rating level.



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

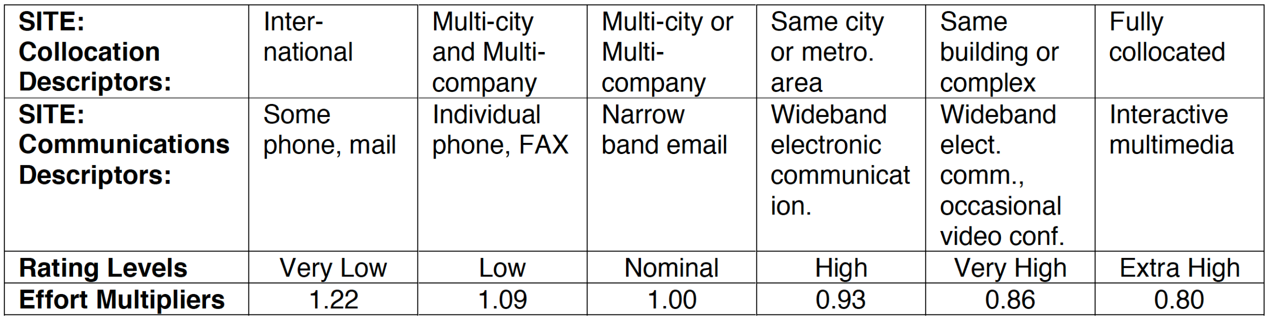
Figure 21: TOOL Cost Driver

3.2.16 Multisite Development (SITE)

This cost driver rating involves the assessment and judgement-based averaging of two factors: site collocation (from fully collocated to international distribution) and communication support (from surface mail and some phone access to full interactive multimedia).

In our case, we all live and work in the same city and we spend most of the time in the same campus, and a good team communication is provided by the use of Slack, streamlining all the discussions and also facilitating calls and video-calls.

Hence, the SITE cost driver is assigned a *Very High* rating level.



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

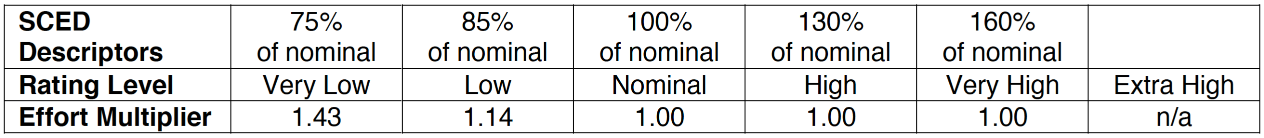
Figure 22: SITE Cost Driver

3.2.17 Required Development Schedule (SCED)

This rating measures the schedule constraint imposed on the project team developing the software.

The ratings are defined in terms of the percentage of schedule stretch-out or acceleration with respect to a nominal schedule for a project requiring a given amount of effort.

As we set to not either accellerate nor stretch-out significantly the schedule, we assume for the SCED cost driver a *Nominal* rating level.



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

Figure 23: SCED Cost Driver

3.3 Scale Factors and Cost Drivers Recap

Scale Factors	Rating Level	Effort Multiplier
PREC	Low	4.96
FLEX	Nominal	3.04
RESL	High	2.83
TEAM	Very High	1.10
PMAT	Very Low	7.80

Cost Driver	Rating Level	Effort Multiplier
RELY	Very High	1.39
DATA	Very High	1.28
CPLX	High	1.17
RUSE	Nominal	1.00
DOCU	Nominal	1.00
TIME	High	1.11
STOR	Nominal	1.00
PVOL	Low	0.87
ACAP	Nominal	1.00
PCAP	Nominal	1.00
PCON	Very High	0.81
APEX	Low	1.10
PLEX	Very Low	1.19
LTEX	Low	1.09
TOOL	High	0.90
SITE	Very High	0.86
SCED	Nominal	1.00

3.4 Effort Estimation

The required effort for the project can be estimated by computing this formula, which expresses the final result in terms of *PersonMonths (PM)*:

$PM = A * SIZE^{E} * \prod CD_{i}$

where:

$A = 2.94\,(for\,COCOMO\,II\,2000)$

$SIZE = estimated\,size\,expressed\,in\,KSLOC\,from\,Function\,Points$

$E = B + 0.01 * \sum SF_i$

$B = 0.91\,(for\,COCOMO\,II\,2000)$

$SF_i = i - th\,scale\,factor\,effort\,multiplier$

$CD_i = i - th\,cost\,driver\,effort\,multiplier\,excluding\,SCED$

Computing the formula with the values from the Function Points and COCOMO Analysis, we get the following average result:

$E = 0.91 + 0.01 * 19.73 = 1.1073$

$PM\,avg = 2.94 * 6.624^{1.1073} * 1.7982 = 42.89\,PM \cong 43\,PM$

with a lower bound of:

$PM\,min = 2.94 * 2.160^{1.1073} * 1.7982 = 12.40\,PM \cong 13\,PM$

and an upper bound of:

$PM\,max = 2.94 * 9.648^{1.1073} * 1.7982 = 65.05\,PM \cong 66\,PM$

3.5 Schedule Estimation

A schedule estimation for the project can be estimated by computing this formula, which expresses the final result in terms of *Months:*

$TDEV = C * PM^{D+0.2*(E-B)} * SCED$

where:

$B = 0.91\,(for\,COCOMO\,II\,2000)$

$C = 3.67\,(for\,COCOMO\,II\,2000)$

$D = 0.28\,(for\,COCOMO\,II\,2000)$

Computing the formula with the values from the COCOMO Analysis and the results of the Effort Estimation, we get the following average result:

$TDEV\,avg = 3.67 * 43^{0.28+0.2*(1.1073-0.91)} * 1.00 = 12.2\,months \cong 13\,months$

with a lower bound of:

$TDEV\,min = 3.67 * 13^{0.28+0.2*(1.1073-0.91)} * 1.00 = 8.32\,months \cong 9\,months$

and an upper bound of:

$TDEV\,max = 3.67 * 66^{0.28+0.2*(1.1073-0.91)} * 1.00 = 13.99\,months \cong 14\,months$

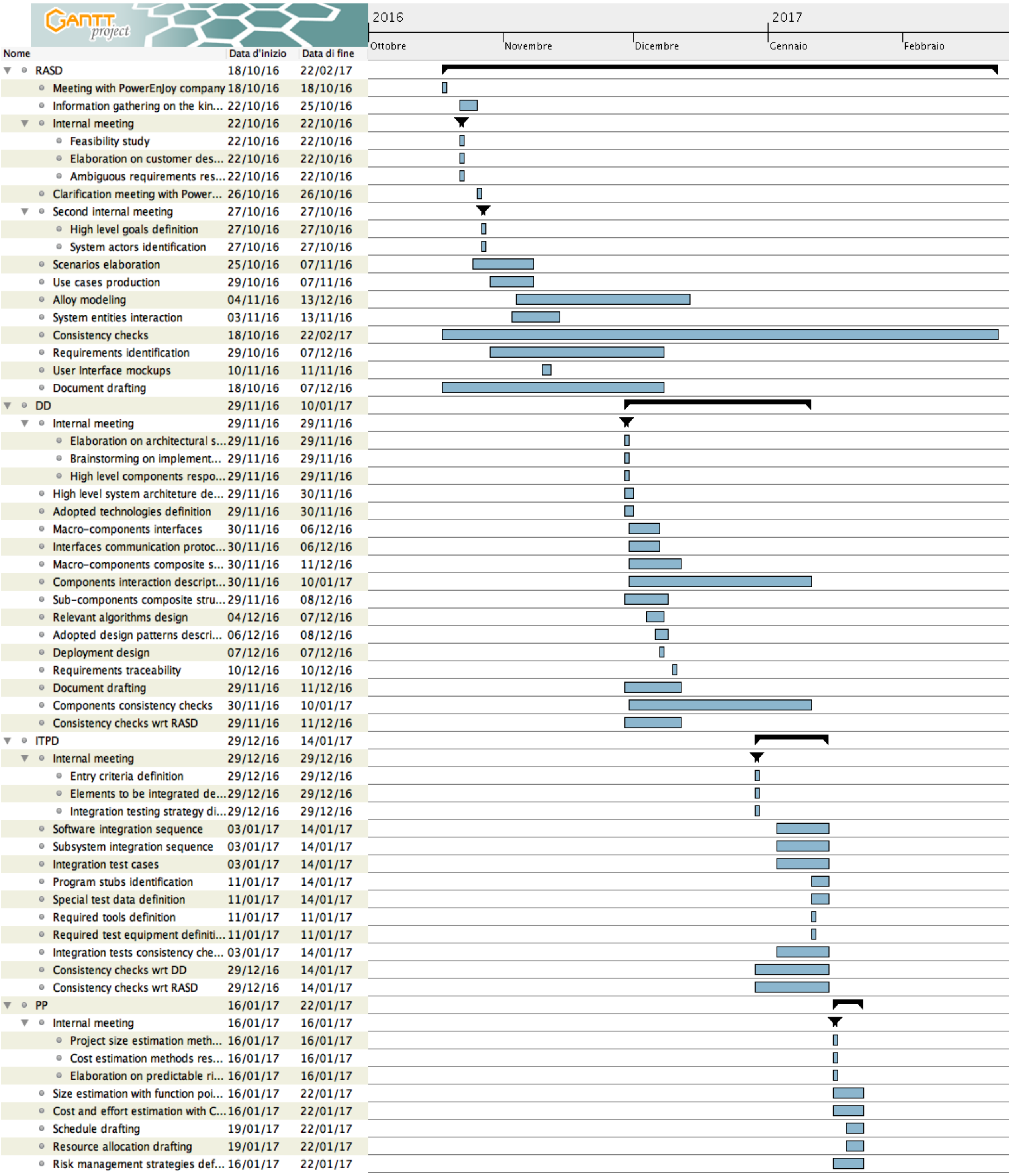
4 Schedule

In the following section the schedule for the project is presented. Some key points explanation is due.

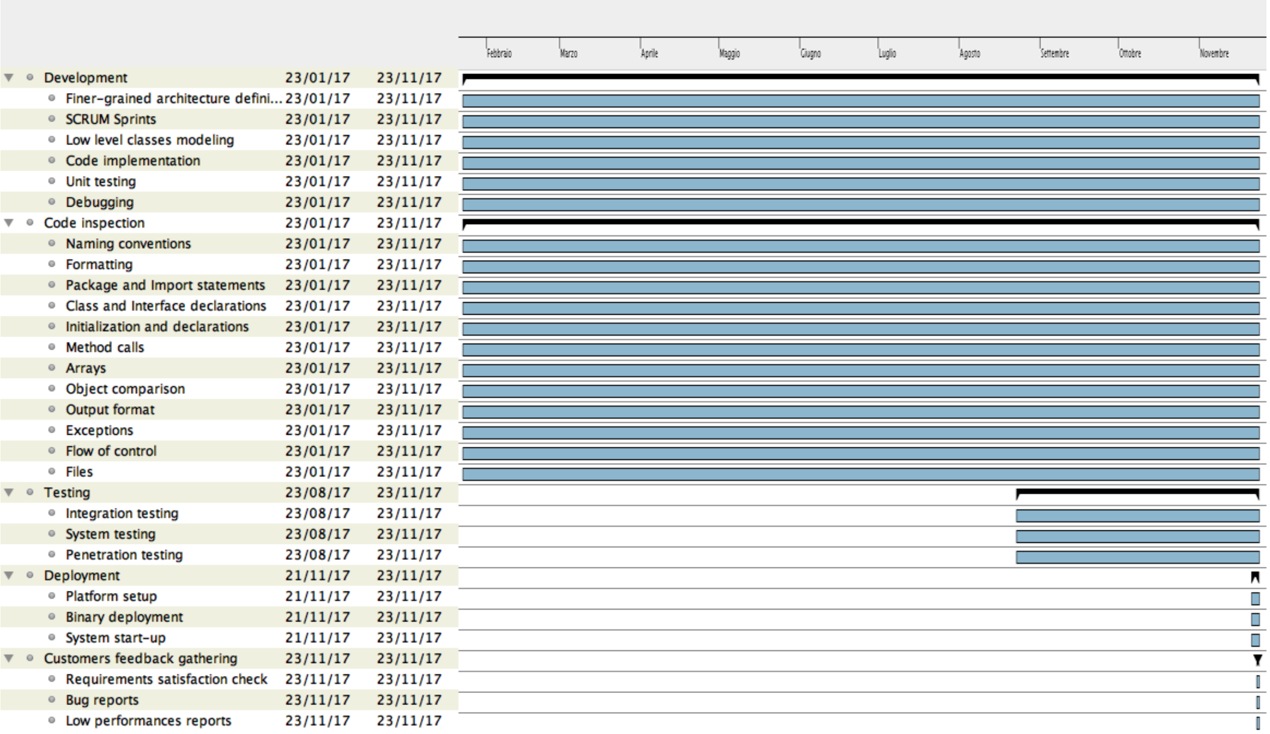
For sections concerning *RASD, DD, ITPD and PP* the schedule has been draft retrospectively, reconstructing the different tasks the team tackled. To keep consistency, time spans are defined referring to the commits of the project repository.

Following sections schedule, instead, a foresight is made based on the previously calculated metrics.

During the project phases, one of the attempt the team strived for was working in an agile fashion, setting up meetings every two weeks identifying the different tasks to be accomplished, splitting up the work and beginning the development. Consistency checks were periodically performed intra-section first and then against previously produced deliverables, updating them in case flaws were identified. This non-stop iteration allowed the team to achieve every time a deeper understanding of the project, building a common knowledge. Doing so, team members could revise other people work, pointing out flaws to be corrected.



Nome	Data d'inizio	Data di fine
RASD	18/10/16	22/02/17
Meeting with PowerEnJoy company	18/10/16	18/10/16
Information gathering on the kin...	22/10/16	25/10/16
Internal meeting	22/10/16	22/10/16
Feasibility study	22/10/16	22/10/16
Elaboration on customer des...	22/10/16	22/10/16
Ambiguous requirements res...	22/10/16	22/10/16
Clarification meeting with Power...	26/10/16	26/10/16
Second internal meeting	27/10/16	27/10/16
High level goals definition	27/10/16	27/10/16
System actors identification	27/10/16	27/10/16
Scenarios elaboration	25/10/16	07/11/16
Use cases production	29/10/16	07/11/16
Alloy modeling	04/11/16	13/12/16
System entities interaction	03/11/16	13/11/16
Consistency checks	18/10/16	22/02/17
Requirements identification	29/10/16	07/12/16
User Interface mockups	10/11/16	11/11/16
Document drafting	18/10/16	07/12/16
DD	29/11/16	10/01/17
Internal meeting	29/11/16	29/11/16
Elaboration on architectural s...	29/11/16	29/11/16
Brainstorming on implement...	29/11/16	29/11/16
High level components respo...	29/11/16	29/11/16
High level system architeture de...	29/11/16	30/11/16
Adopted technologies definition	29/11/16	30/11/16
Macro-components interfaces	30/11/16	06/12/16
Interfaces communication protoc...	30/11/16	06/12/16
Macro-components composite s...	30/11/16	11/12/16
Components interaction descript...	30/11/16	10/01/17
Sub-components composite stru...	29/11/16	08/12/16
Relevant algorithms design	04/12/16	07/12/16
Adopted design patterns descri...	06/12/16	08/12/16
Deployment design	07/12/16	07/12/16
Requirements traceability	10/12/16	10/12/16
Document drafting	29/11/16	11/12/16
Components consistency checks	30/11/16	10/01/17
Consistency checks wrt RASD	29/11/16	11/12/16
ITPD	29/12/16	14/01/17
Internal meeting	29/12/16	29/12/16
Entry criteria definition	29/12/16	29/12/16
Elements to be integrated de...	29/12/16	29/12/16
Integration testing strategy di...	29/12/16	29/12/16
Software integration sequence	03/01/17	14/01/17
Subsystem integration sequence	03/01/17	14/01/17
Integration test cases	03/01/17	14/01/17
Program stubs identification	11/01/17	14/01/17
Special test data definition	11/01/17	14/01/17
Required tools definition	11/01/17	11/01/17
Required test equipment definiti...	11/01/17	11/01/17
Integration tests consistency che...	03/01/17	14/01/17
Consistency checks wrt DD	29/12/16	14/01/17
Consistency checks wrt RASD	29/12/16	14/01/17
PP	16/01/17	22/01/17
Internal meeting	16/01/17	16/01/17
Project size estimation meth...	16/01/17	16/01/17
Cost estimation methods res...	16/01/17	16/01/17
Elaboration on predictable ri...	16/01/17	16/01/17
Size estimation with function poi...	16/01/17	22/01/17
Cost and effort estimation with C...	16/01/17	22/01/17
Schedule drafting	19/01/17	22/01/17
Resource allocation drafting	19/01/17	22/01/17
Risk management strategies def...	16/01/17	22/01/17



Development	23/01/17	23/11/17
Finer-grained architecture defini...	23/01/17	23/11/17
SCRUM Sprints	23/01/17	23/11/17
Low level classes modeling	23/01/17	23/11/17
Code implementation	23/01/17	23/11/17
Unit testing	23/01/17	23/11/17
Debugging	23/01/17	23/11/17
Code inspection	23/01/17	23/11/17
Naming conventions	23/01/17	23/11/17
Formatting	23/01/17	23/11/17
Package and Import statements	23/01/17	23/11/17
Class and Interface declarations	23/01/17	23/11/17
Initialization and declarations	23/01/17	23/11/17
Method calls	23/01/17	23/11/17
Arrays	23/01/17	23/11/17
Object comparison	23/01/17	23/11/17
Output format	23/01/17	23/11/17
Exceptions	23/01/17	23/11/17
Flow of control	23/01/17	23/11/17
Files	23/01/17	23/11/17
Testing	23/08/17	23/11/17
Integration testing	23/08/17	23/11/17
System testing	23/08/17	23/11/17
Penetration testing	23/08/17	23/11/17
Deployment	21/11/17	23/11/17
Platform setup	21/11/17	23/11/17
Binary deployment	21/11/17	23/11/17
System start-up	21/11/17	23/11/17
Customers feedback gathering	23/11/17	23/11/17
Requirements satisfaction check	23/11/17	23/11/17
Bug reports	23/11/17	23/11/17
Low performances reports	23/11/17	23/11/17

5 Resource Allocation

Because of the high level of communication among the team members, tasks were produced thanks to a collaboration of all of them. Therefore, a clear distinction of schedule jobs is not possible to be reconstructed. Refer to the previous schedule.

6 Risk Management

In the following section an elaboration on risk management is proposed, showing the proactive risk strategy the team is willing to follow.

Each risk is described by an ID, a risk description, a probability level of that risk happening (*Low, Moderate, High*) and a the impact it would have on the project (*Negligible, Marginal, Critical and Catastrophic*)

ID	Risk description	Probability	Effects
R1	Relevant people in the team are all ill at critical times in the project	Moderate	Critical
R2	Relevant people in the team lose interest and/or motivation in working on the project	High	Marginal
R3	Changes to the requirements that require significant design rework are proposed	Moderate	Critical
R4	Customer financial problems force reductions in the project budget	Low	Negligible
R5	Lack of coordination causes inconsistencies within the project design	Moderate	Critical
R6	Technology/human faults causes project data loss	Low	Catastrophic
R7	Dev teams of technologies the project is relying on shutdown their activities	Moderate	Critical
R8	Infrastructure owned by the company does not handle the throughput expected	High	Critical
R9	It is impossible to recruit staff with the skills required for the project	Moderate	Catastrophic
R10	Project deadlines cannot be hold	Moderate	Critical
R11	The marketplace is saturated by products similar to PowerEnJoy and the customer will ask frequently for more features to be added in order to be competitive	High	Critical
R12	Relevant people in the team lack of experience in the field	Moderate	Critical
R13	Process definition is ambiguous in describing some notions	Moderate	Marginal
R14	Presented law drafts would affect PowerEnJoy on the legal point of view	Low	Critical

R1 contingency plan Make every person in the team aware of the most important decisions on the project, so that they can keep elaborating on ill people work with average results.

R2 contingency plan Keep motivation high by reminding the main purposes of each step of the project, highlighting the personal skillset that can be developed by taking each task seriously.

R3 contingency plan In designing the system strive for a loosely coupled, highly modular architecture, already thinking it for future changes and expansion. Remember to always design towards interfaces.

R4 contingency plan No big deal. We do not make money anyway out of this.

R5 contingency plan Share every idea and decision about the system design. Communicate much and often. This way designing inconsistencies will eventually arise and get fixed sooner, requiring lower budget modifications.

R6 contingency plan Take advantage of versioning systems to able to recover from possible data loss. In addition, keep weekly backup on different storage devices.

R7 contingency plan Perform some research on activity status of technologies the project is relying on, contacting their development team. Operate a *make or buy* analysis before adopting a technology and in case of relying on an existing one consider the possibility of hiring its development team.

R8 contingency plan Prepare a document targeting the business division showing performed measurements and graphs to express the unfeasibility of some *non-functional requirements* to acquire better infrastructures.

R9 contingency plan Consider hiring less-qualified people and a team of consultancy to train them. Make the management division aware of this and of the necessary budget modification and time to market delay.

R10 contingency plan Implement critical features first, production-ready. Release the product unfinished and introduce remaining feature incrementally through updates. Make the development team aware of the concept of *continuous integration.*

R11 contingency plan Keep system design losely coupled and highly modular. Make it open to extension, in order to require less work when introducing new features the team was not aware of at design time.

R12 contingency plan Never be pretentious. In case of feeling unconfident with something perform research, learn best practices or hire experts of the field to achieve a deeper understanding and knowledge of the domain your are developing for.

R13 contingency plan In case ambiguity is not limited to low-level component scope, organize a meeting with software architects to sort it out, developing more detailed interfaces.

R14 contingency plan Re-negotiate the requirements with the customer to make them in accordance with the new laws.

7 Effort Spent

Arcari Leonardo

- 16/01 - 3h
- 20/01 - 3h
- 22/01 - 3h
- 22/01 - 2h

Bertoglio Riccardo

- 20/01 - 2:30h
- 21/01 - 5:30h

Galimberti Andrea

- 16/01 - 3h
- 17/01 - 1h
- 18/01 - 1h30m
- 19/01 - 2h
- 20/01 - 1h30m
- 21/01 - 3h