

**Software Engineering 2: “PowerEnJoy”**

**Project Plan (V. 1.0)**

Authors:

Sergio CAPRARA

Soheil GHANBARI

Erica TINTI

Milan, Italy

22/01/2017

**Table of contents**

[1. Introduction 3](#_Toc472754548)

[1.1 Revision History 3](#_Toc472754549)

[1.2 Purpose and Scope 3](#_Toc472754550)

[1.3 Definitions and Abbreviations 3](#_Toc472754551)

[1.4 Reference Documents 4](#_Toc472754552)

[2. Project size, cost, and effort estimation 5](#_Toc472754553)

[2.1 Size estimation: function points 5](#_Toc472754554)

[2.1.1 External Inputs 6](#_Toc472754555)

[2.1.2 External Outputs 7](#_Toc472754556)

[2.1.3 External Inquiries 8](#_Toc472754557)

[2.1.4 Internal Logic Files 9](#_Toc472754558)

[2.1.5 External Interface Files 10](#_Toc472754559)

[2.1.6 Overall estimation 10](#_Toc472754560)

[2.2 Cost and effort estimation: COCOMO II 11](#_Toc472754561)

[2.2.1 Scale Drivers 11](#_Toc472754562)

[2.2.2 Cost Drivers 12](#_Toc472754563)

[3. Schedule 19](#_Toc472754564)

[4. Resource allocation 20](#_Toc472754565)

[5. Risk management 21](#_Toc472754566)

[6. Hours of work 22](#_Toc472754567)

# Introduction

## Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Authors | Description |
| 1.0 | 22/01/2017 | S. Caprara, S. Ghanbari, E. Tinti | First release |

## Purpose and Scope

In this document, we are providing details on how the components described in the Design Document will be integrated. To ensure that the interaction between them will give the expected results, we are choosing the method to follow and we are keeping in mind that the Integration Test of a component will be done after having Unit Tested it.

In the following chapters, you will find detailed descriptions of the tests and the name of the tools to be used.

## Definitions and Abbreviations

* **User:** the person registered to the system and allowed to access to its functions.
* **Operator:** a person with technical skills, that fixes car issues.
* **App:** short term used to define a mobile application.
* **Power Plug:** a column with one or more electricity socket where it is possible to charge the car.
* **Safe Area** (or Parking Area): a parking area with parking shared with all the other divers and not especially reserved to PowerEnjoy.
* **Special Parking Area** (or Power Station): a parking area reserved exclusively to PowerEnjoy cars where, for each parking space there is a Power Plug where it is possible to charge a car.
* **Car:** PowerEnjoy car.
* **Reservation:** the relation between a user and a car, that allows the user to start using the car. The reservation guarantees that no one else can reserve and use the reserved car till the end of the rental.
* **DB:** database, the collection of system data.
* **DAO:** Data Access Object.
* **Pojo:** Plain Old Java Object. Object having only getter and setter methods.
* **FP:** Function Point
* **UFP:** Unadjusted Function Point
* **LOC:** Lines of code
* **KSLOC:** Kilo Source Line of code

## Reference Documents

The documents used as a reference to provide the design document are:

* Assignments AA 2016-2017.pdf
* Project planning example document.pdf
* RASD\_PowerEnjoy\_Caprara\_Ghanbari\_Tinti
* DesignDocument\_PowerEnjoy\_Caprara\_Ghanbari\_Tinti
* TestPlan\_PowerEnjoy\_Caprara\_Ghanbari\_Tinti\_v1.0.pdf
* COCOMO II – Model Definition Manual

# Project size, cost, and effort estimation

This section of the document provides

## Size estimation: function points

The size of the project we are working on, will be estimated using the Function Point Analysis approach. This technique is based on counting the number of functionalities in a software project, assigning a weight to each functionality according to its Function type. From the result of this calculation, called Unadjusted Function Point (UFP), we will derive the number of lines of code (LOC) that will give us an idea of the size of the project.

Function types are the following:

* External Input, all operation that takes data in from the external environment
* External Output, all operation that sends data out
* External Inquiry, all operation involving both input and output
* Internal Logic Files, data used and managed by our application
* External Interface Files, data used by our system but generated by other applications

The tables used for the estimation are provided below.

|  |  |  |  |
| --- | --- | --- | --- |
| File Type Referenced | Data elements | | |
| 1-4 | 5-15 | > 15 |
| 0-1 | Low | Low | Avg |
| 2 | Low | Avg | High |
| 3 or more | Avg | High | High |

**Table 1: External Input**

|  |  |  |  |
| --- | --- | --- | --- |
| File Type Referenced | Data elements | | |
| 1-5 | 6-19 | > 19 |
| 0-1 | Low | Low | Avg |
| 2-3 | Low | Avg | High |
| 4 or more | Avg | High | High |

**Table 2: External Output and External Inquiries**

|  |  |  |  |
| --- | --- | --- | --- |
| Record Element Type | Data Elements | | |
| 1-19 | 20-50 | > 50 |
| 1 | Low | Low | Avg |
| 2-5 | Low | Avg | High |
| 6 or more | Avg | High | High |

**Table 3: Internal Logic Files and External Interface Files**

|  |  |  |  |
| --- | --- | --- | --- |
| Type of Component | Complexity of Components | | |
| Low | Average | High |
| External Inputs | 3 | 4 | 6 |
| External Outputs | 4 | 5 | 7 |
| External Inquiries | 3 | 4 | 6 |
| Internal Logic Files | 7 | 10 | 15 |
| External Interface Files | 5 | 7 | 10 |

**Table 4: Unadjusted Function Points**

### External Inputs

The **Login and Logout** (for both users and operators) are low complexity functions, because they involve just one file containing the information, so they get 3 FPs each.

**User functions**

**User Registration** is a function that can be classified as having an average complexity, because it deals with different data and because more than one file is referenced. This leads to 4 FPs.

The **Profile Update** function involves many fields that need to be updated on the DB, so it has a medium complexity.

The **Payment Method Update** operation requires the system to check if the new credit card is valid and it interacts with the DB for saving data. For this reasons the operation has a medium complexity, corresponding to 4 FPs.

The **Cancel Reservation** operation has a low complexity because it references two type of files but has few data. So, it has 3 FPs.

The **End of Renting** operation is a simple operation that involves a few fields, then we assign it a weight of 3.

**Operator functions**

The **Maintenance Request** is a simple operation, involving only one component. For this reason, it has low complexity and gets 3 FPs.

The **End of Maintenance** operation involves few components and has a low complexity, leading to 3 FPs.

|  |  |  |
| --- | --- | --- |
| EI | Complexity | FPs |
| Login | Low | 3x2 |
| Logout | Low | 3x2 |
| User Registration | Average | 4 |
| Profile Update | Average | 4 |
| Payment Method Update | Average | 4 |
| Cancel Reservation | Low | 3 |
| End of Renting | Low | 3 |
| Maintenance Request | Low | 3 |
| End of Maintenance | Low | 3 |
| Total | | 36 |

### External Outputs

The operation for **Registration Confirmation** has a low complexity, that corresponds to 4 FPs.

The **Fished Rent Feedback** operation involves many components and requires several calculations. Then we consider it as a medium complexity operation and we assign it a weight of 7.

The **Money Charge Notification** operation involves many components, interacts with the db and performs several calculations. Therefore, we consider it as a complex operation and we assign it a weight of 7.

|  |  |  |
| --- | --- | --- |
| EO | Complexity | FPs |
| Registration Confirmation | Low | 4 |
| Finished Rent Feedback | High | 7 |
| Total Amount Notification | High | 7 |
| Total | | 18 |

### External Inquiries

**User functions**

**Car Lookup** operation involves more than one component and contains many data, so its complexity is medium, corresponding to 4 FPs.

The **Car Information** is a simple operation to retrieve car details. It references just one component and contains few data. Because of this, the complexity is low and corresponds to 3 FPs.

The **Reserve Car** operation has a low complexity, so it gets 3 FPs.

**Car Unlock** operation has a medium complexity because it involves different components and requires many interactions. It corresponds to 4 FPs.

**View Parking Areas** and **View Special Parking Areas** are simple operations, involving only one component each. Their complexity is low, corresponding to 3 FPs each.

The **View Profile Info** functionality is a simple operation. It involves many fields but only some query is needed. We assigned a weight of 3.

**Operator functions**

The **View Car Maintenance List** is a simple operation and it involves only one simple object. For this reason, we assign a low complexity weight of 3.

The **View Car Details** operation involves many fields because it has to retrieve all the technical information about the car. Then, it has a medium complexity weight of 4.

|  |  |  |
| --- | --- | --- |
| EQ | Complexity | FPs |
| Car Lookup | Average | 4 |
| Car Information | Low | 3 |
| Reserve Car | Low | 3 |
| Car Unlock | Average | 4 |
| View Parking Areas | Low | 3 |
| View Special Parking Areas | Low | 3 |
| View Profile Info | Low | 3 |
| View Car Maintenance List | Low | 3 |
| View Car Details | Average | 4 |
| Total | | 30 |

### Internal Logic Files

**Car Info** element is composed of many fields, organised in 2 data type groups, and is included in many transactions. We apply a medium weight of 10.

**User** element has many fields, organised in 3 or more data type groups, so we assign an average complexity weight of 10.

**Parking Area** is a simple element with few fields and interactions. Thus, it has a complexity weight of 7.

**Special Parking Area** is a simple element with only few fields and interactions, so we apply a weight of 7, even if its fields belong to 2 groups.

**Operator** is a simple element with few fields and simple interactions with cars. We apply a weight of 7.

|  |  |  |
| --- | --- | --- |
| ILF | Complexity | FPs |
| Car Info | Average | 10 |
| User | Average | 10 |
| Parking Area | Low | 7 |
| Special Parking Area | Low | 7 |
| Operator | Low | 7 |
| Total | | 41 |

### External Interface Files

Our system interacts with external components for retrieving some information. We can identify them as:

* driving licence validating system: this ;
* geolocation and mapping service;
* payment system;
* physical car internal system.

|  |  |  |
| --- | --- | --- |
| EIF | Complexity | FPs |
| Driving Licence | Low | 5 |
| Geolocation and Mapping | Average | 7 |
| Payment | Average | 7 |
| Physical Car | High | 10 |
| Total | | 29 |

### Overall estimation

The results of the complexity estimation are provided in the following table.

|  |  |
| --- | --- |
| Type of Component | FPs |
| External Inputs | 36 |
| External Outputs | 18 |
| External Inquiries | 30 |
| Internal Logic Files | 41 |
| External Interface Files | 29 |
| Total | 154 |

Using this result, we can estimate the amount of total code lines in our applications.

## Cost and effort estimation: COCOMO II

The cost and effort estimation is made using the COCOMO II method.

### Scale Drivers

We provide here the table containing figures used in the evaluation. It is based on the following scale drivers:

* Precedentedness (PREC): the value depends on the experience the team has on projects similar to the current one.
* Development flexibility (FLEX): represents the possibility to make changes to the project based on the strictness of the external requirements.
* Risk resolution (RESL): consists in the capacity to solve problems and is strictly related to the risk analysis provided in this document.
* Team cohesion (TEAM): its value represents the capacity of the team members to work together and cooperate
* Process maturity (PMAT): states the level of maturity reached in the process.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Scale Factors | Very Low | Low | Nominal | High | Very High | Extra High |
| PREC | thoroughly  unprecedented | largely  unprecedented | somewhat  unprecedented | generally  familiar | largely  familiar | thoroughly familiar |
| SFj | 6.20 | 4.96 | 3.72 | 2.48 | 1.24 | 0.00 |
| FLEX | rigorous | occasional  relaxation | some  relaxation | general  conformity | some  conformity | general goals |
| SFj | 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%) |
| SFj | 7.07 | 5.65 | 4.24 | 2.83 | 1.41 | 0.00 |
| TEAM | very difficult  interactions | some  difficult  interactions | basically  cooperative  interactions | largely  cooperative | highly  cooperative | seamless  interactions |
| SFj | 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 |
| SFj | 7.80 | 6.24 | 4.68 | 3.12 | 1.56 | 0.00 |

For our project, we will consider a low **PREC** factor (4.96) because we don’t have the experience on a similar project, except for some technical skills on technologies.

Since the project will be developed from scratch and is limited only by the use of some external API, we consider a Very high factor for **FLEX** (1.01).

We will assign **RESL** a Nominal (4.24) factor, because of the presence of a detailed evaluation risks analysis.

To **TEAM** factor we assign a level of Very High (1.10) because stakeholders are collaborative and the developers know each other well.

The **PMAT** scale driver has a nominal rating (4.68), due to the current status of the development.

### Cost Drivers

**Required Software Reliability**

This parameter measures the effect produced by possible software failures.

In our case, we consider a nominal rating, because failures that may take place could produce a breakdown and it can be solved by refunding the client.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RELY Cost Drivers | | | | | | |
| RELY  Descriptors | slightly  inconvenience  losses | easily recoverable  losses | Moderate,  recoverable | high financial  loss | risk to human life |  |
| Rating Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | 0.82 | 0.92 | 1.00 | 1.10 | 1.26 | n/a |

**Data Base Size**

Data driver measures the relation between the Database size and the estimated SLOC. The Database size is helpful to determine the effort necessary to produce test data for testing the software.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| DATA Cost Drivers | | | | | | |
| DATA  Descriptors |  | D/P < 10 | 10 <= D/P <= 100 | 100 <= D/P <= 1000 | D/P > 1000 |  |
| Rating Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | n/a | 0.90 | 1.00 | 1.14 | 1.28 | n/a |

**Product Complexity**

This measure considers the complexity of the different type of operations performed by the system (control, computational, device-dependent, data management, and UI management operations).

In our system the complexity is nominal.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CPLX Cost Drivers | | | | | | |
| 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 |

**Developed for Reusability**

This parameter states the effort needed to produce code to be reused in the current or future projects.

In our case this is nominal, because the code is used only in the current project and its future evolutions.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RUSE Cost Drivers | | | | | | |
| 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 cost driver measures the level of detail of the documentation with respect to the life-cycle needs of the project.

The documentation we provide is strictly related to system needs and satisfies them without exceeding, so the complexity is nominal.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| DOCU Cost Drivers | | | | | | |
| 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 rating represents the percentage of available execution time that the system is expected to use on the resource. In other terms, it is the memory usage.

Our software has some background operations, other than standard operations, so memory usage will be high.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TIME Cost Drivers | | | | | | |
| TIME  Descriptors |  |  | <= 50%  use of  available  execution  time | 70% use of  available  execution  time | 85% use of  available  execution  time | 95% use of  available  execution  time |
| Rating Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | n/a | n/a | 1.00 | 1.11 | 1.29 | 1.63 |

**Main Storage Constraint**

This parameter is related to the amount of storage required for the software system with respect to hardware availability.

Our system can be easily stored in actual hardware, because it doesn’t exceed their capacity.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| STOR Cost Drivers | | | | | | |
| STOR  Descriptors |  |  | <= 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 Multipliers | n/a | n/a | 1.00 | 1.05 | 1.17 | 1.46 |

**Platform Volatility**

This cost driver refers to the regularity of platform changes, intended as hardware and software updates.

For our system, the rating is low, as changes would not be frequent, once the development will be finished and the product tested.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PVOL Cost Drivers | | | | | | |
| PVOL  Descriptors |  | Major change every 12 mo., minor change every 1 mo. | Major: 6mo.; minor: 2wk. | Major: 2mo., minor: 1wk. | Major: 2wk.; minor: 2 days |  |
| Rating Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | n/a | 0.87 | 1.00 | 1.15 | 1.30 | n/a |

**Analyst Capability**

This measure consists in the capability of analysts in providing good Analysis and Design and in their ability to cooperate.

The rating for this parameter is set to low, as the analysis that has been made may lack in some details, due to the poor experience of the analysts in working on project of this entity.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ACAP Cost Drivers | | | | | | |
| ACAP  Descriptors | 15th percentile | 35th percentile | 55th percentile | 75th percentile | 90th percentile |  |
| Rating Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | 1.42 | 1.19 | 1.00 | 0.85 | 0.71 | n/a |

**Programmer Capability**

Like in ACAP, this cost driver represents the ability of programmers in developing the system, cooperating, and working efficiently.

In our case, since programmers are skilled and have few years of experience we assign a High rating.

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

**Personnel Continuity**

This rating represents the personnel turnover.

It has a very high rating, as we wouldn’t have a high turnover percentage in our project.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PCON Cost Drivers | | | | | | |
| PCON  Descriptors | 48% / year | 24% / year | 12% / year | 6% / year | 3% / year |  |
| Rating Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | 1.29 | 1.12 | 1.00 | 0.90 | 0.81 | n/a |

**Applications Experience**

The APEX cost driver states the experience of the team in developing the same type of application required by the current project.

This rating is nominal, as the programmers have few experience on such type of projects.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| APEX Cost Drivers | | | | | | |
| APEX  Descriptors | <= 2  months | 6 months | 1 year | 3 years | 6 years |  |
| Rating Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | 1.22 | 1.10 | 1.00 | 0.88 | 0.81 | n/a |

**Platform Experience**

Same as APEX, experience is measured by this cost driver, but with respect to platform. So, it consists in the experience in working on the same type of platform.

Having worked on other projects based on the same kind of platforms, we can consider for the whole team a high rating of this cost driver.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PLEX Cost Drivers | | | | | | |
| PLEX  Descriptors | <= 2  months | 6 months | 1 year | 3 years | 6 years |  |
| Rating Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | 1.19 | 1.09 | 1.00 | 0.91 | 0.85 | n/a |

**Language and Tool Experience**

This parameter is related to the experience of the team in using the programming language and software tools required by the current project.

Same as the previous rating, we can consider a high rating for this parameter too.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| LTEX Cost Drivers | | | | | | |
| LTEX  Descriptors | <= 2  months | 6 months | 1 year | 3 years | 6 years |  |
| Rating Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | 1.20 | 1.09 | 1.00 | 0.91 | 0.84 | n/a |

**Use of Software Tools**

This rating refers to the level at which the software tool is used.

Because it is being used in an integrated environment, the rating is set to high.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TOOL Cost Drivers | | | | | | |
| 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 Level | Very Low | Low | Nominal | High | Very High | Extra High |
| Effort Multipliers | 1.17 | 1.09 | 1.00 | 0.90 | 0.78 | n/a |

**Multisite Development**

This cost driver is based on the site collocation and on the communication support.

As we live in the same city and use wideband communication, the rating can be set to high.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SITE Cost Drivers | | | | | | |
| SITE  Descriptors | Internatio-nal | 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 communica-tion | 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**

This parameter measures the percentage of schedule stretch-out or acceleration with respect to a nominal schedule for a project requiring a given amount of effort.

In our case, the schedule will consider the necessary time for the development and for resolving issues, if needed.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SCED Cost Drivers | | | | | | |
| 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 |

|  |  |  |
| --- | --- | --- |
| Cost Driver | Factor | Value |
| Required Software Reliability (RELY) | Nominal | 1.00 |
| Data Base Size (DATA) | High | 1.14 |
| Product Complexity (CPLX) | Very High | 1.00 |
| Developed for Reusability (RUSE) | Nominal | 1.00 |
| Documentation Match to Life-Cycle Needs (DOCU) | Nominal | 1.00 |
| Execution Time Constraint (TIME) | High | 1.11 |
| Main Storage Constraint (STOR) | Nominal | 1.00 |
| Platform Volatility (PVOL) | Low | 0.87 |
| Analyst Capability (ACAP) | High | 1.19 |
| Programmer Capability (PCAP) | High | 0.88 |
| Personnel Continuity (PCON) | Very High | 0.81 |
| Applications Experience (APEX) | High | 1.00 |
| Platform Experience (PLEX) | High | 0.91 |
| Language and Tool Experience (LTEX) | High | 0.91 |
| Use of Software Tools (TOOL) | High | 0.90 |
| Multisite Development (SITE) | High | 0.93 |
| Required Development Schedule (SCED) | Nominal | 1.00 |
| Total | | 0.6472 |

# Schedule

The following paragraphs contain the detail of the Test Cases defined in the previous chapter.

# Resource allocation

For supporting and automating Integration Tests we will use two testing tools: JUnit and Arquillian.

# Risk management

Integration tests should also verify the responses of the system in specific cases, such as

# Hours of work

To make this document we have spent:

* Sergio Caprara, 14 hours
* Soheil Ghanbari, 8 hours
* Erica Tinti, 14 hours