

POLITECNICO DI MILANO

*Computer Science and Engineering*

**Project of Software Engineering 2: “*myTaxiService*”**

**Project Plan Document**

***Author***: Andrea Maioli (mat. 852429)

***Reference Professor***: Mirandola Raffaela

Sommario

[**1. Introduction** 2](#_Toc441334373)

[**2. Function Points** 2](#_Toc441334374)

[**3. COCOMO** 2](#_Toc441334375)

[**4. Tasks and Schedule** 2](#_Toc441334376)

[**5. Resources allocation** 2](#_Toc441334377)

[**6. Risks** 2](#_Toc441334378)

**1. Introduction**

This document has the scope of evaluate and identify the time and resources necessary to the development of the myTaxiService application.

TODOS: COCOMO II restyle (Chapter for every sub-driver)

Verify RASD and other document for “style”

**2. Function Points**

* 1. **Rating Tables**

|  |  |  |  |
| --- | --- | --- | --- |
| **Internal Logic Files** | | | |
| **Record Elements** | **Data Elements** | | |
|  | 1-19 | 20-50 | 51+ |
| 1 | Low (7) | Low (7) | Average (10) |
| 2-5 | Low (7) | Average (10) | High (15) |
| 6+ | Average (10) | High (15) | High (15) |

|  |  |  |  |
| --- | --- | --- | --- |
| **External Interface Files** | | | |
| **Record Elements** | **Data Elements** | | |
|  | 1-19 | 20-50 | 51+ |
| 1 | Low (5) | Low (5) | Average (7) |
| 2-5 | Low (5) | Average (7) | High (10) |
| 6+ | Average (7) | High (10) | High (10) |

|  |  |  |  |
| --- | --- | --- | --- |
| **External Inputs** | | | |
| **File Types** | **Data Elements** | | |
|  | 1-4 | 5-15 | 16+ |
| 0-1 | Low (3) | Low (3) | Average (5) |
| 2-3 | Low (3) | Average (5) | High (6) |
| 4+ | Average (5) | High (6) | High (6) |

|  |  |  |  |
| --- | --- | --- | --- |
| **External Outputs** | | | |
| **File Types** | **Data Elements** | | |
|  | 1-5 | 6-19 | 20+ |
| 0-1 | Low (4) | Low (4) | Average (5) |
| 2-3 | Low (4) | Average (5) | High (7) |
| 4+ | Average (5) | High (7) | High (7) |

|  |  |  |  |
| --- | --- | --- | --- |
| **External Inquiries** | | | |
| **File Types** | **Data Elements** | | |
|  | 1-5 | 6-19 | 20+ |
| 0-1 | Low (3) | Low (3) | Average (4) |
| 2-3 | Low (3) | Average (4) | High (6) |
| 4+ | Average (4) | High (6) | High (6) |

* 1. **Element Analysis**
     1. **Internal Logic Files**

The **Internal Logic Files** (**ILF**) are homogeneous set of data used and managed by the application, and in the **myTaxiService** application are:

|  |  |  |
| --- | --- | --- |
| **ILF** | **Data Elements** | **Record Elements** |
| User  (Not considered) | first name, last name, email, telephone, password, birthdate, address, city, zipcode, gender | User |
| Passenger | User’s data elements, request\_id | User, Passenger |
| Driver | User’s data elements, status, queue position, queue\_id, latitude, longitude, maxPassengers, workStartedAt, acceptedRequests, declinedRequests, currentRequestID | User, Driver |
| Operator | User’s data elements | User, Operator |
| Administrator | User’s data elements, special\_code | User, Administrator |
| Queue  (Not considered) | containedTaxis | Queue |
| Area | Queue’s Data Elements, name, startingLatitude, startingLongitude, endingLatitude, endingLongitude | Area, Queue |
| Request | passenger\_id, address, city, zipCode, taxi\_id, status, eta, numberOfPassengers, isFromACall, endLatitude, endLongitude, operator\_id | Request |

*User* and *Queue* are not considered as Internal Logic Files because they does not exists “logically” without being considered with another element. They are reported in this table only for completeness.

* + 1. **External Interface Files**

The **External Interface Files** (**EIF**) are homogeneous set of data used by the application but generated and maintained by other applications, and in **myTaxiService** application are:

|  |  |  |
| --- | --- | --- |
| **EIF** | **Data Elements** | **Record Elements** |
| Map Data | image | Map Data |
| Coordinates | Latitude, longitude, address | Coordinates |
| ETA | arrivalTime, remainingTime | ETA |
| SMS | sendStatus | SMS |

* + 1. **External Inputs**

The **External Inputs** (**EI**) are elementary operation to elaborate data coming from the external environment, and in the **myTaxiService** application are:

|  |  |  |
| --- | --- | --- |
| **EI** | **Data Elements** | **File Types** |
| Passenger Registration | User data elements, repeat password, activity button, message | Passenger |
| Call Center Forced User Registration | User data elements (without password), activity button, message | Passenger, Operator |
| Login | email, password, code + activity button, message | Operator, Administrator, Passenger, Driver |
| Logout | activity button, message | Operator, Administrator, Passenger, Driver |
| Call Center Request | address, city, zipCode, numberOfPassengers, driver\_id (derived and stored in the request), eta (calculated and stored), activity button, message | Request, Operator, Passenger, Area, Driver, ETA |
| Passenger Request | numberOfPassengers, driver\_id, latitude, longitude, activity button, message | Request, Passenger, Area, Driver |
| Manage User Profile | User data elements, level, activity button, message | Administrator, Operator, Passenger, Driver |
| Delete User Profile | user\_id, request\_id, activity button, message | Administrator, Operator, Passenger, Driver, Request |
| Create Area | name, startlatitude, startlongitude, endlatitude, endlongitude, activity button, message | Administrator, Area |
| Manage Area | name, startlatitude, startlongitude, endlatitude, endlongitude, area\_id, activity button, message | Administrator, Area |
| Delete Area | area\_id, activity button, message | Administrator, Area |
| Update Driver Status | tatus, driver\_id, queue\_id, activity button, message | Driver, Area |
| End Request | request\_id, area\_id, queue\_id, driver\_id, activity button, message | Driver, Request, Area |
| Report exceptional event | driver\_id, request\_id, area\_id, queue\_id, newdriver\_id, passenger\_id, activity button, message | Driver, Request, Area, Passenger |
| Accept Request | driver\_id, passenger\_id, request\_id, queue\_id, activity button, message | Driver, Request, Passenger |
| Decline Request | driver\_id, request\_id, queue\_id, queueposition, newdriver\_id, status, activity button, message | Driver, Request, Queue |
| Report Passenger Status | status, passenger\_id, activity button, message | Driver, Passenger, Request |
| Update Driver Location | latitude, longitude, newarea\_id, newqueue\_id | Driver, Area |
| Set Request End Point | address, request\_id, activity button, message | Driver, Request |

* + 1. **External Outputs**

The **External Outputs** (**EO**) are elementary operation that generates data for the external environment (usually includes the elaboration of data from logic files), and in the **myTaxiService** application are:

|  |  |  |
| --- | --- | --- |
| **EO** | **Data Elements** | **File Types** |
| Request Notification | address, numberOfPassengers, activity button | Request, Driver, Coordinate |
| Request Update Notification | status, eta, driver\_id | Request, Driver, Passenger, ETA |
| SMS Notification with Admin Code | Code, telephone number | Administrator, SMS |
| SMS Notification for passenger not found | Message, telephone number, request\_id | Passenger, Request, SMS |

* + 1. **External Inquiries**

The **External Inquiries** (**EQ**) are elementary operation that involves input and output (usually does not include significant elaboration of data from logic files), and in the **myTaxiService** application are:

|  |  |  |
| --- | --- | --- |
| **EQ** | **File Types** | **Data Elements** |
| View User Information | User’s Data Elements, code | Passenger, Administrator, Operator, Driver |
| View Area Information | Area’s Data Elements | Area |
| View Active Request Information | Request’s Data Elements | Request, Passenger, Driver |
| View Queue Information | position | Area, Driver |

* 1. **Function Point Assignment**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Internal Logic Files** | | | | |
| **ILF** | **Record Elements** | **Data Elements** | **Weight** | **Function Points** |
| Passenger | 2 | 11 | Low | 7 |
| Driver | 2 | 20 | Average | 10 |
| Operator | 2 | 10 | Low | 7 |
| Administrator | 2 | 11 | Low | 7 |
| Area | 2 | 6 | Low | 7 |
| Request | 1 | 12 | Low | 7 |
|  |  |  |  |  |
| **Total:** | | | | 45 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **External Interface Files** | | | | |
| **EIF** | **Record Elements** | **Data Elements** | **Weight** | **Function Points** |
| Map Data | 1 | 1 | Low | 5 |
| Coordinate | 1 | 3 | Low | 5 |
| ETA | 1 | 2 | Low | 5 |
| SMS | 1 | 1 | Low | 5 |
|  |  |  |  |  |
| **Total:** | | | | 20 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **External Inputs** | | | | |
| **EI** | **File Types** | **Data Elements** | **Weight** | **Function Points** |
| Passenger Registration | 1 | 14 | Low | 3 |
| Call Center Forced User Registration | 2 | 11 | Average | 5 |
| Login | 4 | 5 | High | 6 |
| Logout | 4 | 2 | Average | 5 |
| Call Center Request | 6 | 8 | High | 6 |
| Passenger Request | 4 | 6 | High | 6 |
| Manage User Profile | 4 | 13 | High | 6 |
| Delete User Profile | 5 | 4 | Average | 5 |
| Create Area | 2 | 7 | Average | 5 |
| Manage Area | 2 | 8 | Average | 5 |
| Delete Area | 2 | 3 | Low | 3 |
| Update Driver Status | 2 | 5 | Average | 5 |
| End Request | 3 | 6 | Average | 5 |
| Report exceptional event | 4 | 8 | High | 6 |
| Accept Request | 3 | 6 | Average | 5 |
| Decline Request | 3 | 8 | Average | 5 |
| Report Passenger Status | 3 | 4 | Low | 3 |
| Update Driver Location | 2 | 4 | Low | 3 |
| Set Request End Point | 2 | 4 | Low | 3 |
|  |  |  |  |  |
| **Total:** | | | | 90 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **External Outputs** | | | | |
| **EO** | **File Types** | **Data Elements** | **Weight** | **Function Points** |
| Request Notification | 3 | 3 | Low | 4 |
| Request Update Notification | 4 | 3 | Average | 5 |
| SMS Notification with Admin Code | 2 | 2 | Low | 4 |
| SMS Notification for passenger not found | 3 | 3 | Low | 4 |
|  |  |  |  |  |
| **Total:** | | | | 17 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **External Inquiries** | | | | |
| **EQ** | **File Types** | **Data Elements** | **Weight** | **Function Points** |
| View User Information | 4 | 11 | High | 6 |
| View Area Information | 1 | 6 | Low | 3 |
| View Active Request Information | 3 | 12 | Average | 4 |
| View Queue Information | 2 | 1 | Low | 3 |
|  |  |  |  |  |
| **Total:** | | | | 16 |

* 1. **Final Results**

|  |  |
| --- | --- |
| **FP Type** | **FP Count** |
| ILF | 46 |
| EIF | 20 |
| EI | 90 |
| EO | 17 |
| EQ | 16 |
|  |  |
| **Total:** | 189 |

Considering that the multiplier for Java Code is 53, the total number of **Source Line Of Code** (**SLOC**) is *10017*.

**3. COCOMO II**

Until now, only one person has developed the project. In order to have a more realistic situation that corresponds to the reality, two developers with the same level of knowledge of the currently employed developer will be hired to help the development phase.

**3.1. Software Scale Drivers**

* + 1. **Scale Factor Weight Table [TODO: better name]**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Scale Factors** | **Very Low** | **Low** | **Nominal** | **High** | **Very High** | **Extra High** |
| **PREC SFj:** | thoroughly unpreceden ted  6.20 | largely unpreceden ted  4.96 | somewhat unpreceden ted  3.72 | generally familiar  2.48 | largely familiar  1.24 | thoroughly familiar  0.00 |
| **FLEX**  **SFj:** | rigorous 5.07 | occasional relaxation 4.05 | some relaxation 3.04 | general conformity 2.03 | some conformity 1.01 | general goals 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 SFj:** | 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 SFj:** | 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 |

* + 1. **Scale Factor Explanation [TODO: better name]**

**Precedentedness** (**PREC**): Reflects the previous experience of the organization with this type of project. Very low means no previous experience, extra high means that the organization is completely familiar with this application domain.

*It is set to “Low” because there is no enough experience with the used technology and the design skills achieved until now are not enough. Also, the most of the notions and technology used for this project are new.*

**Development** **Flexibility** (**FLEX**): Reflects the degree of flexibility in the development process. Very low means a prescribed process is set; Extra high means that the client only sets general goals.

*It is set to “High” because there are some goals and some general key-point defined, but there is also a good level of flexibility.*

**Architecture** **/** **Risk** **Resolution** (**RESL**): Reflects the extent of risk analysis carried out. Very low means little analysis, extra high means a complete a thorough risk analysis.

*Due to the analysis done in the Risks chapter, it is set to “High”.*

**Team** **Cohesion** (**TEAM**): Reflects how well the development team know each other and work together. Very low means very difficult interactions, extra high means an integrated and effective team with no communication problems.

*The hired developers will have a good level of experience in working in team, so it can be considered “very high”.*

**Process** **Maturity** (**PMAT**): Reflects the process maturity of the organization. The computation of this value depends on the CMM Maturity Questionnaire but an estimate can be achieved by subtracting the CMM process maturity level from 5.

*By analyzing the description of each CMM level, the CMM level is 3 and the PMAT is set to “high”.*

* ***CMM level 3: “defined”.*** It is characteristic of processes at this level that there are sets of defined and documented standard processes established and subject to some degree of improvement over time. These standard processes are in place (i.e., they are the AS-IS processes) and used to establish consistency of process performance across the organization.
  + 1. **Scale Factor Values**

|  |  |  |  |
| --- | --- | --- | --- |
| **Code** | **Name** | **Factor** | **Value** |
| PREC | Precedentedness | Low | 4.96 |
| FLEX | Development Flexibility | High | 2.03 |
| RESL | Architecture / Risk Resolution | High | 2.83 |
| TEAM | Team Cohesion | Very High | 1.10 |
| PMAT | Process Maturity | High | 3.12 |
|  |  |  |  |
|  | | | 1.0504 |

**3.2. Software Cost Drivers**

**Required Software Reliability** (**RELY**): This is the measure of the extent to which the software must perform its intended function over a period of time. If the effect of a software failure is only slight inconvenience then RELY is very low. If a failure would risk human life, then RELY is very high.

*Due to the easily recoverable losses, it is set to “Nominal”.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **RELY**  **Descriptors:** | slight inconven- ience | low, easily recoverable losses | moderate, easily recoverable losses | high financial loss | risk to human life |  |
| **Rating Levels** | Very Low | Low | Nominal | High | Very High | Extra High |
| **Effort Multipliers** | 0.82 | 0.92 | 1.00 | 1.10 | 1.26 | n/a |

**Data** **Base** **Size** (**DATA**): This cost driver attempts to capture the effect large test data requirements have on product development. The rating is determined by calculating D/P, the ratio of bytes in the testing database to SLOC in the program. The reason the size of the database is important to consider, is because of the effort required to generate the test data that will be used to exercise the program.

*The database size can be estimated with a value between 10 and 100 D/P, so it is set to “Nominal”*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

**Product** **Complexity** (**CPLX**): Complexity is divided into five areas: control operations, computational operations, device-dependent operations, data management operations, and user interface management operations. Using the following table, select the area or combination of areas that characterize the product or the component of the product you are rating. The complexity rating is the subjective weighted average of the selected area ratings.

*Considering all the areas and the selected effort for each area, the final rating is set to “Nominal”.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Control Operations** | **Computational Operations** | **Device- dependent Operations** | **Data Management Operations** | **User Interface Management Operations** |
| **Very Low** | Straight-line code with a few non-nested structured programming operators: DOs, CASEs, IF- THEN-ELSEs.  Simple module composition via procedure calls or simple scripts. | Evaluation of simple expressions: e.g., A=B+C\*(D- E) | Simple read, write statements with simple formats. | Simple arrays in main memory. Simple COTS- DB queries, updates. | Simple input forms, report generators. |
| **Low** | Straightforward nesting of structured programming operators.  Mostly simple predicates | Evaluation of moderate-level expressions: e.g., D=SQRT(B\*\*2- 4.\*A\*C) | No cognizance needed of particular processor or I/O device characteristics. I/O done at GET/PUT level. | Single file subsetting with no data structure changes, no edits, no intermediate files.  Moderately complex COTS- DB queries, updates. | Use of simple graphic user interface (GUI) builders. |
| **Nominal** | Mostly simple nesting. Some intermodule control.  Decision tables. Simple callbacks or message passing, including middleware- supported distributed processing | Use of standard math and statistical routines. Basic matrix/vector operations. | I/O processing includes device selection, status checking and error processing. | Multi-file input and single file output. Simple structural changes, simple edits. Complex COTS-DB  queries, updates. | Simple use of widget set. |
| **High** | Highly nested structured programming operators with many compound predicates.  Queue and stack control. Homogeneous, distributed processing.  Single processor soft real-time control. | Basic numerical analysis: multivariate interpolation, ordinary differential equations.  Basic truncation, round-off concerns. | Operations at physical I/O level (physical storage address translations; seeks, reads, etc.). Optimized I/O overlap. | Simple triggers activated by data stream contents.  Complex data restructuring. | Widget set development and extension. Simple voice I/O, multimedia. |
| **Very High** | Reentrant and recursive coding. Fixed- priority interrupt handling. Task synchronization, complex callbacks, heterogeneous distributed processing.  Single- processor hard real-time control. | Difficult but structured numerical analysis: near- singular matrix equations, partial differential equations.  Simple parallelization. | Routines for interrupt diagnosis, servicing, masking.  Communication line handling.  Performance- intensive embedded systems. | Distributed database coordination. Complex triggers. Search optimization. | Moderately complex 2D/3D, dynamic graphics, multimedia. |
| **Extra High** | Multiple resource scheduling with dynamically changing priorities.  Microcode-level control.  Distributed hard real-time control. | Difficult and unstructured numerical analysis: highly accurate analysis of noisy, stochastic data. Complex parallelization. | Device timing- dependent coding, micro- programmed operations.  Performance- critical embedded systems. | Highly coupled, dynamic relational and object structures.  Natural language data management. | Complex multimedia, virtual reality, natural language interface. |

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

**Developed** **for** **Reusability** (**RUSE**): This cost driver accounts for the additional effort needed to construct components intended for reuse on current or future projects. This effort is consumed with creating more generic design of software, more elaborate documentation, and more extensive testing to ensure components are ready for use in other applications.

*The reusability will be applied only to reuse components across the modules in this project, so it is set to “Nominal”.*

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

**Documentation Match to Lifecycle Needs** (**DOCU**): this cost driver is evaluated in terms of the suitability of the project’s documentation to its life-cycle needs. The rating scale goes from Very Low (many life-cycle needs uncovered) to Very High (very excessive for life-cycle needs).

*The suitability of the project’s documentation to its life-cycle needs is the right-sized, so it is set to “Nominal”.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

**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. The rating ranges from nominal, less than 50% of the execution time resource used, to extra high, 95% of the execution time resource is consumed.

*The estimated use of the execution time is below 50% of the available execution time, so it is set to “Nominal”.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TIME**  **Descriptors:** |  |  | 50% use of available execution time | 70% use of available execution time | 85% use of available execution time | 95% use of available execution time |
| **Rating Levels** | Very Low | Low | Nominal | High | Very High | Extra High |
| **Effort Multipliers** | n/a | n/a | 1.00 | 1.11 | 1.29 | 1.63 |

**Storage** **Constraint** (**STOR**): This rating represents the degree of main storage constraint imposed on a software system or subsystem.

*It is set to “Nominal” because the application will store only information in the database and the data usage will be very low.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **STOR**  **Descriptors:** |  |  | 50% use of available storage | 70% use of available storage | 85% use of available storage | 95% use of available storage |
| **Rating Levels** | Very Low | Low | Nominal | High | Very High | Extra High |
| **Effort Multipliers** | n/a | n/a | 1.00 | 1.05 | 1.17 | 1.46 |

**Platform** **Volatility** (**PVOL**): “Platform” is used here to mean the complex of hardware and software (OS, DBMS, etc.) the software product calls on to perform its tasks.

*It is set to “Low” because, for now, no changes to the platform are planned.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

**Analyst** **Capability** (**ACAP**): Analysts are personnel who work on requirements, high-level design and detailed design. The major attributes that should be considered in this rating are analysis and design ability, efficiency and thoroughness, and the ability to communicate and cooperate. The rating should not consider the level of experience of the analyst; that is rated with APEX, LTEX, and PLEX. Analyst teams that fall in the fifteenth percentile are rated very low and those that fall in the ninetieth percentile are rated as very high.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

**Programmer** **Capability** (**PCAP**): Evaluation should be based on 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, and the ability to communicate and cooperate. The experience of the programmer should not be considered here; it is rated with APEX, LTEX, and PLEX. A very low rated programmer team is in the fifteenth percentile and a very high rated programmer team is in the ninetieth percentile.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

**Personnel** **Continuity** (**PCON**): The rating scale for PCON is in terms of the project’s annual personnel turnover: from 3%, very high continuity, to 48%, very low continuity.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |  |

**Application** **Experience** (**APEX**): The rating for this cost driver (formerly labeled AEXP) is dependent on the level of applications experience of the project team developing the software system or subsystem. The ratings are defined in terms of the project team’s equivalent level of experience with this type of application. A very low rating is for application experience of less than 2 months. A very high rating is for experience of 6 years or more.

*It is set to “Low” due to a low level of experience for this kind of application.*

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

**Platform** **Experience** (**PLEX**): The Post-Architecture model broadens the productivity influence of platform experience by recognizing the importance of understanding the use of more powerful platforms, including more graphic user interface, database, networking, and distributed middleware capabilities.

*It is set to “Low” due to a low level of experience for this kind of platform.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

**Language and Toolset 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. Software development includes the use of tools that perform requirements and design representation and analysis, configuration management, document extraction, library management, program style and formatting, consistency checking, planning and control, etc. In addition to experience in the project’s programming language, experience on the project’s supporting tool set also affects development effort. A low rating is given for experience of less than 2 months. A very high rating is given for experience of 6 or more years.

*It is set to “Low” because there is a bit of experience with the Language and Toolset that have been grown during other java projects.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |  |

**Use** **of** **Software Tools** (**TOOL**): The tool rating ranges from simple edit and code, very low, to

integrated life-cycle management tools, very high.

*The tool to be used corresponds to the descriptor of the rating level “Low”.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TOOL**  **Descriptors** | edit, code, debug | simple, frontend, backend CASE, little integration | basic life- cycle tools, moderately integrated | strong, mature life- cycle tools, moderately integrated | strong, mature, proactive life-cycle tools, well integrated with processes, methods, reuse |  |
| **Rating Levels** | Very Low | Low | Nominal | High | Very High | Extra High |
| **Effort Multipliers** | 1.17 | 1.09 | 1.00 | 0.90 | 0.78 | n/a |

**Multisite** **Development** (**SITE**): Determining its 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).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

**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. Accelerated schedules tend to produce more effort in the earlier phases to eliminate risks and refine the architecture, more effort in the later phases to accomplish more testing and documentation in parallel.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Code** | **Name** | **Factor** | **Value** |
| **Product** | | | |
| RELY | Required Software Reliability | Nominal | 1.00 |
| DATA | Data Base Size | Nominal | 1.00 |
| CPLX | Product Complexity | Nominal | 1.00 |
| RUSE | Developed for Reusability | Nominal | 1.00 |
| DOCU | Documentation Match to Lifecycle Needs | Nominal | 1.00 |
| **Platform** | | | |
| TIME | Execution Time Constraint | Nominal | 1.00 |
| STOR | Storage Constraint | Nominal | 1.00 |
| PVOL | Platform Volatility | Low | 0.87 |
| **Personnel** | | | |
| ACAP | Analyst Capability | Nominal | 1.00 |
| PCAP | Programmer Capability | High | 0.88 |
| PCON | Personnel Continuity | Very High | 0.81 |
| APEX | Application Experience | Very Low | 1.22 |
| PLEX | Platform Experience | Very Low | 1.19 |
| LTEX | Language and Toolset Experience | Low | 1.09 |
| **Project** | | | |
| TOOL | Use of Software Tools | Low | 1.09 |
| SITE | Multisite Development | Extra High | 0.80 |
| SCED | Required Development Schedule | Nominal | 1.00 |
|  |  |  |  |
| **:** | | | 0.8557 |

**3.3. Results**

(KSLOC: *Kilo Source Line Of Code*; lines of code estimated with the FP analysis, expressed in multiple of 103)

(EAF: *effort adjustment factor*)

(E: *Effort Applied*)

(SE: *Schedule Equation Exponent*)

1. **Tasks and Schedule**

**4.1. Project Tasks**

Until now, the provided Project’s Documentation is:

* Requirements Analysis and Specification Document
* Design Document
* Integration Test Plan Document

Even considering the various changes made to the provided documentation, it is useful to present the project to the stakeholders in order to verify that all their requests have been fulfilled. This will prevent modifications to the project documents (and project structure, too) during the implementation phase.

Considering the documentation provided until now, it is possible to identify three different tasks with their sub-tasks:

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Presentation Tasks** | | | |
| **Task ID** | **Task Description** | **Dependencies** | **Duration** |
| P.A | Creation of the slides. | - | 22 Days |
| P.B | Presentation of the slides to the stakeholders. | P.A | 1 Day |
| P.C | Apply requested changes, if any, to the current documentation | P.B | * 1. Days |

|  |  |  |  |
| --- | --- | --- | --- |
| **Development Tasks** | | | |
| **Task ID** | **Task Description** | **Dependencies** | **Duration** |
| D.A1 | Creation of the Development Database Tier | - | 2 |
| D.A2 | Creation of the DBMS tables | D.A1 | 3 |
| D.B | Creation of the Development Application Server | - | 2 |
| D.C | Creation of the Development Web Server | - | 2 |
| D.D | Creation of the Entity Beans (Queue, Area, Driver, Passenger, Administrator, Operator, Request and user) | D.A, D.B | 1 x bean |
| D.E1 | Creation of the Location Manager and declaration of the public methods that can be called by other components | D.A, D.B |  |
| D.E2 | Implementation of the Location Manager | D.A, D.B, D.D, D.E1 |  |
| D.E3 | JUnit tests for the Location Manager | D.E2 |  |
| D.F1 | Creation of the Account Manager and declaration of the public methods that can be called by other components | - |  |
| D.F2 | Implementation of the Account Manager | D.A, D.B, D.D,D.F1 |  |
| D.F3 | JUnit tests for the Account Manager | D.F2 |  |
| D.G1 | Creation of the Queue Manager and declaration of the public methods that can be called by other components | - |  |
| D.G2 | Implementation of the Queue Manager | D.A, D.B, D.D, D.G1 |  |
| D.G3 | JUnit tests for the Queue Manager | D.G2 |  |
| D.H1 | Creation of the Taxi Manager and declaration of the public methods that can be called by other components | - |  |
| D.H2 | Implementation of the Taxi Manager | D.A, D.B, D.D, D.E1, D.G1, D.H1 |  |
| D.H3 | JUnit tests for the Taxi Manager | D.H2 |  |
| D.I1 | Creation of the Request Manager and declaration of the public methods that can be called by other components | - |  |
| D.I2 | Implementation of the Request Manager | D.A, D.B, D.E1, D.F1, D.G1, H1, D.I1 |  |
| D.I3 | JUnit tests for the Request Manager | D.I2 |  |
| D.L1 | Declaration of the exposed methods by the RESTful API | - |  |
| D.L2 | Implementation of the Web Server Controller | D.C, D.E1, D.F1, D.G1, D.H1, D.I1, D.L1 |  |
| D.L3 | JUnit tests for the Web Server Controller | D.L2 |  |
| D.M | Creation of the xhtml pages needed by JSF | D.C |  |
| D.N | Creation of the Mobile Application | D.L1 |  |
| D.O | Perform Integration Testing as described in ITPD (40 tasks) | All the above tasks | 15 |
| D.P | Deployment | All the above tasks | 5 |

*NB: The declaration of the public methods of each component enables simultaneous work.*

**4.2. Project Schedule**

Given the results of the COCOMO analysis, the development phase should last 284 day. In order to be a little bit more flexible with the deadline, the development phase can last maximum 300 day. For the integration testing phase will be used 15 days and 5 more days will be used to deploy the entire system.

The deadline of the project are shown in this table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Starting Date** | **Deadline** | **Status** |
| RASD | 15/10/2015 | 06/11/2015 | Done |
| DD | 07/11/2015 | 04/12/2015 | Done |
| ITPD | 07/01/2016 | 21/01/2016 | Done |
| PPD | 21/01/2016 | 02/02/2016 | Done |
| Project Presentation | 02/02/2016 | 24/02/2016 | To Be Done |
| Project Presentation (P.C) | 24/02/2016 | 27/02/2016 | To Be Done |
| Implementation | 27/02/2016 | 22/12/2016 | To Be Done |
| Integration Testing | 23/12/2016 | 6/01/2017 | To Be Done |
| Deployment | 7/01/2017 | 11/01/2017 | To Be Done |

[TODO: move integration testing inside IMPLEMENTATION)

**5. Resources allocation**

Creare I diagrammi con I vari component

**6. Risks**

**6.1. Project Risks**

Delays -> High

Lack of communication -> Low

Requirements Change -> Very Low (not probable, presentation done for avoid that)

Lack of experience-> Very High

**6.2. Technical Risks**

Downtime: Very Low (it is redundant with EC2)

Data loss: Low (beneficts of EC2, but backup needed)

Data leaks & security problems: Low (https enabled by design. Two-step-auth for admin. Firewall properly configured )

Junk code: moderate (perform code analysis to prevent)

Integration testing failures: low (it is done as described in ITPD and when components are available)

**6.3. Business Risk**

Bankruptcy