Project Report

Mite Ristovski Dushica Stojkoska

February 2, 2016

Software Engineering 2
Politecnico di Milano

version 1.0

Contents

1 Introduction

In this document is described in details the project plan of the developing and management process of MyTaxiService software product. Moreover, the following activities are described

- In the first part of the document, the focus is on project size estimation, by applying Function Points approach.
- In the next part, the effort and cost of the project are estimated, by applying COCOMO approach.
- Furthermore are presented tasks of the project, their scheduling and allocation of resources.
- The last part of this document is focused on the management of the possible risks.

2 Function Point Approach

Function Points approach is a method for estimating the size of the software product and its complexity, based on the functionalities the product provides. All functional requirements are discussed in details in the RASD document for MyTaxiService application. The functionalities are grouped in:

– Internal Logic Files (ILFs)

– External Interface Files (ELFs)

– External Inputs

– External Outputs

– External Inquiry

In the following table are presented the points given to each functionality and relative complexity:

Function Types	**Simple**	**Medium**	**Complex**
ILFs	7	10	15
EIFs	5	7	10
External Input	3	4	6
External Output	3	5	7
External Inquiry	3	4	6

Table 1: Function Points weight

In order to compute the number of Function Points, we need to compute the weighted sum of function types using the coefficients from the table above. The amount of points appropriate for each functionality, based on its complexity, is listed below:

- **Internal Logic Files:** The system stores information about the following data:

- Client;
- Taxi driver;
- Notifications;
- Requests;
- Reservations;
- Distribution of taxis.

Each of these entities has medium structure, because they are composed of a number of fields. Thus, we have 6 x 10 = 60 FPs.

- **External Interface Files:**

The system features only one operation in this category, and that is the external service for tracking location of the taxis. This information results in one entity, so this structure will be of a simple complexity. Thus, we have 1 x 5 = 5 FPs.

- **External Inputs:** The software product allows the user to:
  - Login/Logout;
  - Register;
  - Update profile;
  - Create request/reservation;
  - Taxi drivers confirm/decline a request;

Loging in or out, registering to the system, and updating user's profile are simple operations. Creating requests or reservations, and confirming or declining taxi call are operations of complex weight, because they involve many other operations. Thus, we have 3 x 3 + 2 x 6 = 21 FPs.

- External Outputs:

The system notifies the users that the ride is accepted, along with informing them about the code of the taxi and time to wait. This operation is of complex weight, so we have 1 x 7 = 7 FPs.

- External Inquiry: The application allows to its users to request to see:

- Reservation;
- Request;
- User's profile.

These are operations of medium complexity, so we have: 3 x 4 = 12 FPs.

Total number of FPs: The total number adds up to 105 FPs. With this value we can estimate the number of lines of code.

On the table found at this http://www.qsm.com/resources/function-point-languages-table, we choose the parametar 53.

Finaly, we get 53 x 105 = 5656 SLOC.

3 COCOMO 2 Approach

3.1 Scale Drivers

Scale Factors	**Very Low**	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
PREC	thoroughly unprecedented [6.2]	largely unprecedented [4.96]	somewhat unprecedented [3.72]	generally familiar [2.48]	largely familiar [1.24]	thoroughly familiar [0.0]
FLEX	rigorous [5.07]	occasional relaxation [4.05]	some relaxation [3.04]	general conformity [2.03]	some conformity [1.01]	general goals [0.0]
RESL	little (20%) [7.07]	some (40%) [5.65]	often (60%) [4.24]	generally (75%) [2.83]	mostly (90%) [1.41]	full (100%) [0.0]
TEAM	very difficult interactions [5.48]	some difficult interactions [4.38]	basically cooperative interactions [3.29]	largely cooperative [2.19]	highly cooperative [1.1]	seamless interactions [0.0]
PMAT	The estimated Equivalent Process Maturity Level (EPML) or					
	SW-CMM Level 1 Lower [7.8]	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 [0.0]

Table 2: Scale Factor Values, SFj, for COCOMO II Models [1]

These values are evaluated according to the following description:

Precedentedness [PREC] it reflects the previous experience that we had with this kind of projects. Since for us this was the first experience using this framework and these developments methodologies, this value will be low.

Development flexibility [FLEX] it reflects the degree of flexibility in the development process. The professors set the general specifications without going too much in detail, for this reason this value will be high.

Risk resolution [RESL] it reflects the extent of risk analysis carried out. Thanks to filters, and security access mostly of the risks were eliminated then this value will be very high.

[1] http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf

Team cohesion [TEAM] it reflects how well the development team know each other and work together. In our case we had some problems, in particular for the difference of working time and initial synchronization issues. Nonetheless, we overcame those difficulties by thoroughly describing guidelines and goals in our development process. Since this approach was successful the final value for this attribute is very high.

Process maturity [PMAT] this was evaluated around the 18 Key Process Area (KPAs) in the SEI Capability Model. Because of the goals were consistently achieved these values will be set to high, level 3.

The results can be summed up in the following table:

Scale Driver	Rating Level	Value
Precedentedness	Low	4.96
Development Flexibility	Low	4.05
Risk Resolution	High	2.83
Team Cohesion	Very High	1.1
Process Maturity	Nominal	4.68
Total		17.62

3.2 Cost Drivers

Because the project is not implemented in code all of the following choices are invented and try to be as realistic as possible. The values for the data are taken from the following COCOMO 2 manual http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf

Required Software Reliability (RELY) This is set to **high** rating level because a failure can cause high financial loss and the effort multiplier is 1.10.

Data Base Size (DATA) This cost driver attempts to capture the effect large test data requirements have on product development. Our choice for the test database is 1GB. The program size is 5656 SLOC so the fraction $\frac{D}{P} = 176.8$ and this value **high** rating level and effort multiplier of 1.14.

Product Complexity (CPLX) According to the *Component Complexity Ratings Levels*[Table 19] this product can be characterized as one with **very high** rating level and has effort multiplier 1.34.

Developed for Reusability (RUSE) This product is build such that large portion of the code can be reused in similar projects. According to this RUSE rating can be set **nominal** and the value for this is 1.0.

Documentation Match to Life-Cycle Needs (DOCU) This will have **nominal** rating level since it will be right sized to the life-cycle needs and large portion of the product is already described in the RASD and DD documents. The effort multiplier for nominal rating is 1.0.

Analyst Capability (ACAP) 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. Since we have spend reasonable effort in preparation of RASD and DD we can consider ourselves as in the 75th percentile which has **high** rating and multiplier of 0.85.

Programmer Capability (PCAP) Since Major factors which should be considered in the rating are ability, efficiency and thoroughness, and the ability to communicate and cooperate. Because experience of the programmer should not be considered here and we have successfully cooperated during all of the phases of this project we can consider **high** rating and the multiplier is 0.88.

Applications Experience (APEX) The rating for this cost driver is dependent on the level of applications experience of the project team developing the software system or subsystem. Since we have no particular working experience but during studies we have created different some applications involving different technologies and we are in university education for 5 years we can rate this as **high** which has multiplier 0.88.

Platform Experience (PLEX) Because our knowledge of enterprise frameworks such as Java EE is limited, but we have some experience with databases and other frameworks we can rate this as **nominal** with multiplier 1.0.

Language and Tool Experience (LTEX) This relates to the previous and because we consider Java EE as framework where the system is developed we rate LTEX as **nominal** with multiplier 1.0.

Use of Software Tools (TOOL) We suppose that the NetBeans IDE is used for development, Maven for dependency management, Arquillian and JUnit at different levels for testing and Git for the source code management. This are all mature tools and are well integrated so can rate TOOL as **high** with multiplier 0.99.

Multisite Development (SITE) We are fully collocated as we worked together Slack which very efficient tool for communication between developers and we rate this as **extra high** with multiplier 0.8.

Required Development Schedule (SCED) Here we consider that accelerated schedules. Since 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. But still this may not be reliable because of inexperience some unpredictable delays can occur so we can rate SCED as **very high** which has multiplier 1.0.

Cost Driver	Rating Level	Effort Multiplier
RELY	High	1.10
DATA	High	1.14
CPLX	Very High	1.34
RUSE	Nominal	1.0
DOCU	Nominal	1.0
ACAP	High	0.85
PCAP	High	0.88
APEX	High	0.88
PLEX	Nominal	1.0
LTEX	Nominal	1.0
TOOL	High	0.99
SITE	Extra High	0.8
SCED	Extra High	1.0
Product		0.87

Results

3.3 The Effort-Equation

The effort estimation equation is the following

$$Effort = A * EAF * KSLOC^E$$

which is measured in person-months(PM). $A = 2.94$ for COCOMO 2. EAF is the effort adjustment factor derived from Cost Drivers and here $EAF = 0.87$ and $KSLOC = 5.656.E$ is derived from Scale Drivers and can be calculated as follows $E = B + 0.01 * \sum_{k=1}^{5} SF_k$ where SF is the *scale factor* and $B = 0.91$ for COCOMO 2. So $E = 0.91 + 0.01 * 17.62 = 1.0862$. The solution to the effort equation is the following $Effort = 2.94 * 0.87 * 5.656^{1.0862} = 16.797(PM)$.

3.4 The Schedule-Equation

The following equation will be used to predict the number of months complete the software product

$$Duration = 3.67 * Effort^{(D+0.2*(E-B))}$$

where $D = 0.28$, $B = 0.91$, and $E = 1.0862$ as calculated previously. So $Duration = 3.67 * 16.797^{0.31524} = 8.926 \approx 9$ months. From here we can also estimate the number of required people for the project using $N_{people} = \frac{Effort}{Duration} = \frac{16.797}{8.926} = 1.88 \approx 2$ people and this gives reasonable estimate given the KSLOC and the rating levels chosen for chosen for the scale and cost drivers.

4 Tasks, schedule and resources allocation

This project consists of five assignments:

- **RASD Document:** This document contains identified actors and scenarios for MyTaxiService software product. It also contains use cases, requirements, as well as UML nad Alloy tools are used for analysis of the software product.

  Submition deadline for this document was on: 6/11/2015.

- **Design Document:** This document contains more detailed description of the components of MyTaxiService system. Moreover, some main algorithms of the software product are described: for managing rides as being requested, and for distribution of unqueued taxis.

  Submition deadline for this document was on: 4/12/2015.

- **Code Inspection:** This document contains a systematic inspection of a computer source code. Several methods from the class EjbBundleValidator were assigned to our group for code inspection. More specificcally, the assigned methods from this class were: public void accept(EjbDescriptor ejb), private void validateConcurrencyMetadata(EjbDescriptor ejb), and private void validatePassivationConfiguration(EjbDescriptor ejb).

  Submition deadline for this document was on: 5/1/2016.

- **Integration Testing:** This document contains description of the tests that should be performed in order all components of MyTaxiService system can communicate correctly with each other.

  Submition deadline for this document was on: 21/1/2016.

- **Project Plan Report:** This document contains size and cost estimation of MyTaxiService software product, as well as overall description of the tasks of the project, how many hours each member had been working on every task, and risk management of the project.

  Submition deadline for this document was on: 2/2/2016.

The following table contains the hours of work each team member had spent for every task assignment of this project.

	RASD	**Design Document**	**Inspection Code**	**Integration Testing**	**Project Plan**
Mite Ristovski	30	25	25	13	5
Dushica Stojkoska	35	28	20	12	5

Table 3: Hours of work

The total number of hours of work spent for the whole project is 198 hours. If we assume that a person works for 40 hours per week, we get:

$$198 \text{ hours} / (40 \times 4) \text{ hours} = 1.2375 \text{ Person/Month}$$

5 Risks management

Identifying all possible risks that can occur is of a very crucial meaning for successful development of the whole project. The team should be well prepared - for every possible scenario, concise solution or plan of action should be developed.

In this table are presented the possible risks that can occur during project development, their probability of occurring, their impact, and suitable solution for every risk.

Risk	**Risk type**	**Probability**	**Impact**	**Solution**
Team member cannot work or finish task	Personnel	Low	Serious	The schedule should be flexible and this type of scenarios should be predicted in the overall period for development of the project
Team member quits	Personnel	Low	Serious	The schedule should be flexible and this type of scenarios should be predicted in the overall period for development of the project
Team member is added to the project and needs more time to be trained	Personnel	Low	Serious	The schedule should be flexible and this type of scenarios should be predicted in the overall period for development of the project
Team does not have enough skills for product development	Technical	Low	Catastrophic	The schedule should take into account this type of scenarios and provide time needed for training
Software is damaged or affected, or physical equipment is damaged	Technical	Low	Catastrophic	Having backup of all information, and having budget predicted for this type of scenarios
Software tools require more time to implement than expected, or are not productive as expected	Technical	Medium	Critical	Planning of the project should be very accurate, as well as the choice of software tools

Interfaces of external systems do not work properly	Technical	High	Serious	Changing the provider
Some functionality do not work properly	Technical	Medium	Catastrophic	Tests should ensure optimal and correct performance, as well as having well written and efficient algorithms
Overall development of the product is larger than estimated	Management	Medium	Not serious	Estimation and schedule for development of the project should be made very accurate and according to real time expectations and estimations

Table 4: Risks assessment