Project **P**lan
myTaxiService

Luca Nanni (850113) Giacomo Servadei (854819)

January 24, 2016

Contents

List of Figures

List of Tables

Chapter 1

Estimations of project size, cost and effort

1.1 *Function Points* approach

With this quantitative technique we can estimate the *project size* in terms of *function points.* Function points are a unit of measure of software size, and they are used to estimate the complexity (in terms of *functionalities*) of a software to be implemented, independently of the technology that will be used to implement it.

Since we are interested in the *functionalities* of our software, we reference directly to our Requirements Analysis and Specification Document (RASD) in order to retrieve them.

1.1.1 Short description of FP

The counting of function points is based of a combination of different characteristic of the software, both internal (regarding the internal structure of the software) and external (reqarding the interaction of the system with the user or other systems).
With this in mind we can classify the following types of function points

- *Internal logic file* (ILF): the data managed internally by the application
- *External interface file* (EIF): data used by the application but generated/maintained by other applications
- *External input* (EI): elaboration of data coming from the external environment
- *External output* (EO): generation of data for the external environment
- *External inquiry* (EQ): input/output operation that do not need significant elaboration of data from the ILF

We will use the following table (1.1) in order to retrieve the *weights* for each function type in the $\{$*Simple,Medium,Complex*$\}$ case

Function Types	Weight		
	Simple	Medium	Complex
N. Inputs	3	4	6
N. Outputs	4	5	7
N. Inquiry	3	4	6
N. ILF	7	10	15
N. EIF	5	7	10

Figure 1.1: Function points weight table

1.1.2 Calculation for *myTaxiService*

ILF	Complexity	Rationale
Requests	Simple	The requests of a passenger are to be stored, and they present a very simple structure (name of passenger, date of execution, location, ecc...)
Reservations	Simple	Similar to requests with some more attributes
Passenger accounts	Medium	Managing of all the data of a passenger account
TaxiDriver accounts	Simple	The accounts of taxi drivers are much simpler than the one of passengers. They consist simply in a username and password and the id of the taxi.
Taxi zones	Complex	The representation of taxi zone implies some geometry a probably the files containing them will be pretty complex
Taxi Queue	Medium	A taxi queue is composed by a taxi zone and the set of available taxi drivers that are in it.
Total FP	**56**	

Table 1.1: ILF calculation

EIF	Complexity	Rationale
GoogleMaps API	Medium	The system uses the external service offered by Google in order to parse the locations and to transform them into geographic/geometric objects. This task is estimated not to be trivial
Total FP	**7**	

Table 1.2: EIF calculation

External Input	Complexity	Rationale
Login of passenger	Simple	From the requirements
Logout of passenger	Simple	””
Registration of a passenger	Simple	””
Deletion of passenger account	Medium	Differently from the other basic operations on the account of a passenger, this one involves the elimination of all the requests and reservations.
Login of taxidriver	Simple	From the requirements
Logout of taxidriver	Simple	””
Make a request	Medium	Not an easy operation: it involves the storing of the request, the validation of the input and the parsing of the locations, ecc...
Make a reservation	Complex	This operation involves a lot of components in our system and probably is the most demanding one.
Setting of availability of taxi driver	Medium	It involves the setting of the zone and the pop/push into a taxi queue
Accepting/Refusing a ride	Medium	It involves the re-forwarding of the request/reservation and the notification to the passenger in case of problems.
Total FP	**37**	

Table 1.3: External Input calculation

External Output	Complexity	Rationale
Sending of a request of a ride to a taxi driver	Medium	We need to retrieve the correct taxi driver and send him all the data regarding the ride
Acknowledgment of reservation to a passenger	Simple	It only require the generation of a string on the base of the outcome of the reservation process.
Sending to the passenger the meeting data of a request	Medium	It involves the calculation of the estimated waiting time
Total FP	**14**	

Table 1.4: External Output calculation

External Inquiry	Complexity	Rationale
Visualize the list of reservations of a passenger	Simple	From the requirements
Total FP		**3**

Table 1.5: External Inquiry calculation

Finally we do a resume, calculating the total number of function points.

FP type	Number of FP
ILF	56
EIF	7
External Input	37
External Output	14
External Inquiry	3
Total FP	**117**

Table 1.6: Total number of function points calculation

1.2 *COCOMO* approach

1.2.1 Source Lines of Code

Since we are planning to implement our system through the JEE environment, we use an average conversion factor of 46[1]. The *Source Lines of Code* (SLOC) is:

$$SLOC = 46 \times 117 = 5382$$

1.2.2 Scale Drivers

The scale drivers considered are[2]:

- *Precedentedness* (PREC): describe the previous experience of the organization with this type of project.
- *Development Flexibility* (FLEX): Degree of flexibility of the development process
- *Architecture/Risk resolution* (RESL): Extent of risk analysis carried out.
- *Team cohesion* (TEAM): how the development team know each other and work together
- *Process maturity* (PMAT): Process maturity of organization dependent on the CMM Maturity Questionnaire.

[1] Documented in `http://www.qsm.com/resources/function-point-languages-table`
[2] Taken from `http://sunset.usc.edu/research/COCOMOII/Docs/modelman.pdf`

In the following table[3] we describe the meanings of the different values that each scale driver can have.

Scale Factors (*Wi*)	Very Low	Low	Nominal	High	Very High	Extra High
PREC	thoroughly unprecedented	largely unprecedented	somewhat unprecedented	generally familiar	largely familiar	throughly familiar
FLEX	rigorous	occasional relaxation	some relaxation	general conformity	some conformity	general goals
RESL[a]	little (20%)	some (40%)	often (60%)	generally (75%)	mostly (90%)	full (100%)
TEAM	very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions
PMAT	Weighted average of "Yes" answers to CMM Maturity Questionnaire					

Table 6: Scale Factors for COCOMO II Early Design and Post-Architecture Models

[a] % significant module interfaces specified, % significant risks eliminated.

Figure 1.2: Scale drivers levels explanation

In the case of *myTaxiService* the following table describe the choice for each scale driver

Scale driver	Value
PREC	Very Low
FLEX	High
RESL	Low
TEAM	Nominal
PMAT	Nominal

Table 1.7: Scale drivers values

The scale drivers are used in order to calculate the scale exponent E, according to the following formula:

$$E = 0.91 + 0.01 \times \sum_{j=1}^{5} (SF_j)$$

where the values of SF_j are from the following table

[3] Taken from: `http://sunset.usc.edu/research/COCOMOII/Docs/modelman.pdf`

W(i)	Very Low	Low	Nominal	High	Very High	Extra High
Precedentedness	4.05	3.24	2.43	1.62	0.81	0.00
Development Flexibility	6.07	4.86	3.64	2.43	1.21	0.00
Architecture / Risk Resolution	4.22	3.38	2.53	1.69	0.84	0.00
Team Cohesion	4.94	3.95	2.97	1.98	0.99	0.00
Process Maturity	4.54	3.64	2.73	1.82	0.91	0.00

Table 1: Scale Factors

Figure 1.3: Scale factors

1.2.3 Cost drivers

The COCOMO II uses 17 different cost drivers. They are grouped into four categories: *Product, Platform, Personnel* and *Project*.

Product factor	**Description**
Required Software Reliability (RELY)	Extent to which the software must perform its intended function over a period of time
Data Base size (DATA)	This measure attempts to capture the affect large data requirements have on product development.
Product Complexity (CPLX)	Control operations, computational operations, device-dependent operations, data management operations, and user interface management operations.
Required Reusability (RUSE)	Effort needed to construct components intended for reuse on the current or future projects
Documentation match to life-cycle needs (DOCU)	Suitability of the project's documentation to its life-cycle needs

Table 1.8: Product cost drivers

Platform factor	**Description**
Execution Time Constraint (TIME)	Execution time constraint imposed upon a software system
Main Storage Constraint (STOR)	Degree of main storage constraint imposed on a software system or subsystem
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 encodes the change of platform over 12 months

Table 1.9: Platform cost drivers

Personnel factor	**Description**
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
Programmer Capability (PCAP)	Evaluation should be based on the capability of the programmers as a team rather than as individuals.
Applications Experience (AEXP)	The ratings are defined in terms of the project team’s equivalent level of experience with this type of application
Platform Experience (PEXP)	Understanding of the platform, both hardware and software
Language and Tool Experience (LTEX)	Measure of the level of programming language and software tool experience of the project team developing the software system or subsystem
Personnel Continuity (PCON)	Project’s annual personnel turnover

Table 1.10: Personnel cost drivers

Project factor	**Description**
Use of Software Tools (TOOL)	The tool rating ranges from simple edit and code, very low, to integrated lifecycle management tools, very high
Multisite Development (SITE)	Ability of distributed software development
Required Development Schedule (SCED)	Measures the schedule constraint imposed on the project team developing the software

Table 1.11: Project cost drivers

These cost drivers are evaluated through the following table[4]

	Very Low	Low	Nominal	High	Very High	Extra High
RELY	slight inconvenience	low, easily recoverable losses	moderate, easily recoverable losses	high financial loss	risk to human life	
DATA		DB bytes/ Pgm SLOC < 10	$10 \leq D/P < 100$	$100 \leq D/P < 1000$	$D/P \geq 1000$	
CPLX	see Table 20					
RUSE		none	across project	across program	across product line	across multiple product lines
DOCU	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	
TIME			$\leq$ 50% use of available execution time	70%	85%	95%
STOR			$\leq$ 50% use of available storage	70%	85%	95%
PVOL		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	
ACAP	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
PCAP	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile	
PCON	48% / year	24% / year	12% / year	6% / year	3% / year	
AEXP	$\leq$ 2 months	6 months	1 year	3 years	6 years	
PEXP	$\leq$ 2 months	6 months	1 year	3 years	6 year	
LTEX	$\leq$ 2 months	6 months	1 year	3 years	6 year	
TOOL	edit, code, debug	simple, frontend, backend CASE, little integration	basic lifecycle tools, moderately integrated	strong, mature lifecycle tools, moderately integrated	strong, mature, proactive lifecycle tools, well integrated with processes, methods, reuse	
SITE: Collocation	International	Multi-city and Multi-company	Multi-city or Multi-company	Same city or metro. area	Same building or complex	Fully collocated
SITE: Communications	Some phone, mail	Individual phone, FAX	Narrowband email	Wideband electronic communication.	Wideband elect. comm, occasional video conf.	Interactive multimedia
SCED	75% of nominal	85%	100%	130%	160%	

Table 21: Post-Architecture Cost Driver Rating Level Summary

Figure 1.4: Cost drivers levels explanation

[4]Taken from `http://sunset.usc.edu/research/COCOMOII/Docs/modelman.pdf`

In the case of *myTaxiService* are used the following values:

Cost driver	**Value**
RELY	Nominal
DATA	Nominal
CPLX	Nominal
RUSE	High
DOCU	High
TIME	Nominal
STOR	High
PVOL	Nominal
ACAP	High
PCAP	High
AEXP	Low
PEXP	Low
LTEX	High
PCON	Very High
TOOL	High
SITE	Low
SCED	High

Table 1.12: Values of cost drivers

We use this values in order to calculate the *effort adjustment factor* (EAF) like this:

$$EAF = \prod_{j=1}^{17} EM_j$$

where EM_j are the cost drivers factors derived from the following table:

Cost Driver	Rating					
	Very Low	Low	Nominal	High	Very High	Extra High
RELY	0.75	0.88	1.00	1.15	1.39	
DATA		0.93	1.00	1.09	1.19	
CPLX	0.75	0.88	1.00	1.15	1.30	1.66
RUSE		0.91	1.00	1.14	1.29	1.49
DOCU	0.89	0.95	1.00	1.06	1.13	
TIME			1.00	1.11	1.31	1.67
STOR			1.00	1.06	1.21	1.57
PVOL		0.87	1.00	1.15	1.30	
ACAP	1.50	1.22	1.00	0.83	0.67	
PCAP	1.37	1.16	1.00	0.87	0.74	
PCON	1.24	1.10	1.00	0.92	0.84	
AEXP	1.22	1.10	1.00	0.89	0.81	
PEXP	1.25	1.12	1.00	0.88	0.81	
LTEX	1.22	1.10	1.00	0.91	0.84	
TOOL	1.24	1.12	1.00	0.86	0.72	
SITE	1.25	1.10	1.00	0.92	0.84	0.78
SCED	1.29	1.10	1.00	1.00	1.00	

Figure 1.5: Cost drivers factors

1.2.4 Effort calculation

The *effort* (calculated in Person-Months) required is given by the following formula:

$$Effort = 2.94 \times EAF \times \left(\frac{SLOC}{1000}\right)^{E}$$

And in the particular case of *myTaxiService* it is estimated to be:

$$Effort_{myTaxiService} = 15.9 \text{ Person-Months}$$

1.2.5 Scheduling

The COCOMO II scheduling equation predicts the number of months required to complete a software project, and consists in the following formula:

$$Duration = 3.67 \times (Effort)^{SE}$$

$$SE = 0.28 + 0.2 \times (E - 0.91)$$

In the case of *myTaxiService* we estimate

$$Duration_{myTaxiService} = 11.9 \text{ Months}$$

1.2.6 Dimensioning of the team

Finally we compute how big should be the development team for the project.

$$People = \frac{Effort}{Duration}$$

and in the specific case of *myTaxiService* we have

$$People_{myTaxiService} = 1.36 \approx 2$$

1.2.7 Resume

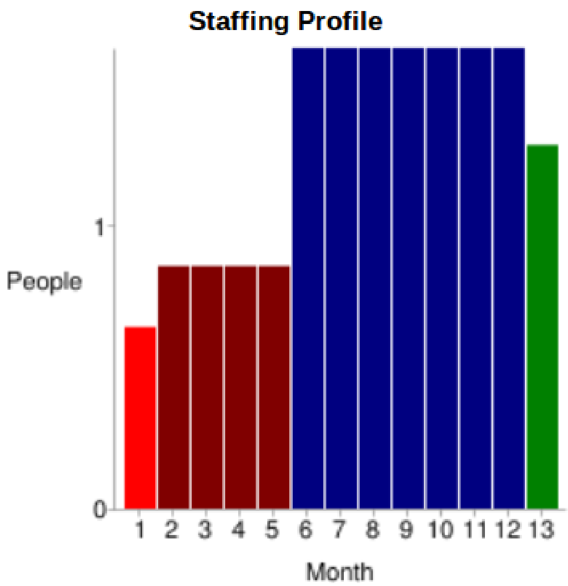
Software Development (Elaboration and Construction)

Effort = 15.9 Person-months
Schedule = 11.9 Months
Cost = $39716

Total Equivalent Size = 5382 SLOC

Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	1.0	1.5	0.6	$2383
Elaboration	3.8	4.5	0.9	$9532
Construction	12.1	7.4	1.6	$30184
Transition	1.9	1.5	1.3	$4766



Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.1	0.5	1.2	0.3
Environment/CM	0.1	0.3	0.6	0.1
Requirements	0.4	0.7	1.0	0.1
Design	0.2	1.4	1.9	0.1
Implementation	0.1	0.5	4.1	0.4
Assessment	0.1	0.4	2.9	0.5
Deployment	0.0	0.1	0.4	0.6

Figure 1.6: Resume

We supposed to pay our developers an average of 2500$ per month.

Chapter 2

Tasks of the project

Chapter 3

Resource allocation

Chapter 4

Risk analysis