POLITECNICO DI MILANO

Software engineering 2 project

TAXI SERVICE

PROJECT PLAN DOCUMENT

AUTHOR

Pallotta Simone

PROFESSOR

Raffaela Mirandola

Version 1.0

ACADEMIC YEAR 2015/2016

intentionally left blank

CONTENTS

Introduction

In this document we estimate the time and resources needed to develop “Taxi service” applications.

We will use “Functional Point approach” and results will be compared against the size of the project.

As a second step we will adopt COCOMO model to evaluate the effort and its results will be compared whit the time actually spent.

As we have two applications (mobile-client and server) we have estimated both using the same approach.

Project Size estimate with Function Point Approach

The Function Point estimation approach, is a technique used to assess the effort needed to design and develop an applications.

We have used this technique to evaluate the application size based on program characteristics.

The functionalities list has been obtained from the RASD document and for each one of them the realisation complexity has been evaluated.

The Function Types are:

- **Internal Logic File**: homogeneous set of data managed by the application.
- **External Interface File**: homogeneous set of data used by the application but generated by external application.
- **External input**: elementary operation to elaborate data coming form the external environment
- **External Output:** elementary operation that generates data for the external environment
- **External Inquiry:** elementary operation that involves input and output operations.

Mobile App:

Function Type	Complexity		
	Simple	Medium	Complex
Internal Logic File	6	10	15
External Interface File	4	7	11
External Input	5	7	9
External Output	3	4	7
External Inquiry	4	5	6

Internal Logic File	Keychain and user settings	2x6FP
External Interface File	JSON from server,	1x7FP
External Input	• Login/logout: these are complex operations • Sign up: this is a complex operation • Request a cab: this is a complex operation as involve only geolocation and user data; • Accept request: this is a complex operation as involve only user data; • View status ride: this is a complex operation as involve only geolocation; we consider these as complex operation because we also include all the UI code	2x9FP 1x9FP 1x9FP 1x9FP 1x9FP
External Output	JSON generated from the queue	1x7FP
External Inquiry	push notification	1x6FP
Total		86

Server Application:

Function Type	Complexity		
	Simple	Medium	Complex
Internal Logic File	6	10	15
External Interface File	4	8	11
External Input	5	6	7
External Output	3	4	7
External Inquiry	3	4	6

Internal Logic File	database tables	1x10FP
External Interface File	JSON from mobile app,	1x8FP
External Input	no data	0FP
External Output	JSON parsed and sent to mobile app	1x4FP
External Inquiry	no inquiry	0FP
Total		22

Then we need to convert the FP to lines of code.

We will be using the table found at this URL:

http://www.cs.bsu.edu/homepages/dmz/cs697/langtbl.htm

Number of lines of code for iOS version since we use Objective C

LOC = 86 * 27 = 2.236

Number of lines of code for Android version since we use JAVA

LOC = 86 * 53 = 4.558

For converting FP to lines code for server parts we have estimated roughly using Haskell value since there is no reference value for Ruby language.

Number of lines of code for server parts:

LOC = 22 * 38 = 912

Project Size estimate with COCOMO Approach

COCOMO, COnstructive COst MOdel, its a mathematical model used by software engineer to takes into account the characteristics of product, people and process for developing a software product.

We Consider a project with all "Nominal" Cost Drivers and Scale Drivers would have an EAF of 1.00 and exponent E of 1.0997. Following this formula

$$effort = 2.94 * EAF * (KSLOC)^{E}$$

we obtain:

Mobile app:

effort=$2.94*(1.0)*(2.236)^{1.0997}$ = 7,12 Person/Months

EAF: Effort Adjustment Factor derived from Cost Drivers.
E:Exponent derived from Scale Drivers.

Now we try to calculate the schedule estimation of project in month with the following formula:

$$Duration = 3.67 * (effort)^{F}$$

We consider an exponent E of 0.3179

Duration = $3.67 * (7,12)^{0.3179}$ = 1.86 Months
Now we can estimate the number of people needed to complete the project with the following formula:

$$N_{people} = effort/Duration$$

we obtain:

N_{people} = 7,12/1,86 = 4 people

For android version we can roughly assume the same parameters.

Server app:

effort=$2.94 * (1.0) * (2.236)^{1.0997}$ = 2,65 Person/Months

Duration = $3.67 * (2,65)^{0.3179}$ = 1,36 Months

N_{people} = 2,65/1,36 = 2 people

Task Identification and scheduling

Here, we report the list describing the project tasks and related scheduling:

- Requirements Analysis and Specifications Document:

  We started on October 25th 2015 describing the raw functionalities of the system paying attention in dividing them between mobile and server aspects.

  We spent 3 days listing the main functionalities and the following 4 days splitting them between mobile, server side and external services.

  Additional 3 days were spent drawing schemas and writing the document.

- Design Document

  We spent 4 days planning the database and the backbone of application with particular attention to the database logical structure.

  Additional 2 days were spent drawing schemas and writing the document.

- Development

  The first release is focused only on iOS version for mobile application, while the server implementation is complete.

  Efforts were split between mobile app and server app.

  Server app:

  a. database design: 2 days

  b. queue implementation in ruby: 4 days

  c. crontab settings: 1 days

  d. Restful calls/JSON formatting and parsing: 4 days

  e. APNS connection: 2 days

  f. SMS gateway: 1 days

  Mobile app:

  g. Backbone app: 3 days

  h. JSON management: 1 days

i. JSON parsing: 1 days

j. signup view: 2 days

k. signing view: 2 days

l. taxi status view: 2 days

m. accept request view: 3 days

n. request taxi view: 3 days

o. verification code view: 1 days

- Integration Test Plan Document

  3 days were spent drawing schemas and writing the document.

- Testing

  Server app:

  We started testing server functionalities preparing a full suite of terminal script containing curl commands.

  Mobile app:

  we based out testing using different approaches:

  - Assert
  - UNIT test
  - UI test
  - Continuos integration using test flight app

  Totally, we spent 8,5 days for testing

- Deployment

  Due to apple store acceptance policies we must add 5 days to the time required to build the app package, so we take 1 hour plus 5 days.

Resources allocation

- Requirements Analysis and Specifications Document:

  Every member of the group took part in requirement analysis.

- Design Document

  Every member of the group took part in design specification phases.

- Development

  as only one person was allocated to the project, allocation days corresponds to the effort of the single passes.

- Testing

  Every member of the group took part in testing phases.

- Deployment

  Every member of the group took part in deployment phases.