POLITECNICO DI MILANO
SCUOLA DI INGEGNERIA DELL'INFORMAZIONE
CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

Progetto di Ingegneria del Software 2

TravelDream

Parte V: Project Planning

(A posteriori)

Responsabile:
Prof. Raffaella Mirandola

Progetto di:
Piazza Enrico Matricola n.760240
Polvara Riccardo Matricola n.817572
Zamperetti Niccolò Matricola n.814656

ANNO ACCADEMICO 2013-2014

Indice

Capitolo 1

Introduzione

1.1 Scopo del documento.

In questo documento viene svolta una analisi a posteriori del progetto. Viene simulata la fase di Project Planning, normalmente svolta all'inizio del ciclo di vita del software, con l'intento di effettuare delle stime sulla dimensione del progetto e sul tempo richiesto per svilupparlo, per confrontarle poi con le i dati reali ottenuti alla fine dello sviluppo. A tal fine useremo:

1. Analisi dei FP (Functional Points): per ottenere una stima delle dimensioni del progetto in termini di FP, poi convertiti in KLOC (Kilo Lines Of Code) per confrontarlo con le dimensioni reali del progetto.
2. Stima secondo COCOMO (COnstructive COst MOdel): per ottenere una stima in termini di tempo (ore/uomo) e costo del progetto (usando anche i dati dell'analisi precedente), da confrontare con le ore effettivamente svolte, riportate nel Time Reporting dai componenti del gruppo.

Data la inusuale collocazione temporale del documento (a fine progetto) non ci addentreremo in ulteriori dettagli di pianificazione tipici dei deocumenti di project planning.

1.2 Fonti e Riferimenti

La principali fonti sulle quali ci siamo basati per l'analisi, sono le lezioni stesse di Ingegneria del Software 2 e relativo libro di testo. Per la relazione tra (U)FP (Unadjasted FP) e KLOC abbiamo usato dati provenienti da "COCOMO II - Model Definition Manual" del Center for Software Engineering, USC, reperibile al link:

- http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf

(In ogni caso verranno riportati i dati provenienti da questo manuale laddove necessario)

Capitolo 2

Analisi di (U)FP e KLOC

2.1 UFP

L'analisi di UFP prevede la suddivisione dell'applicazione in diverse funzionalità (a loro volta distinte in 5 categorie), e il seguente assegnamento di un determinato numero di FP a ciascuna di esse secondo determinati pesi.

Table 3. UFP Complexity Weights

Function Type	Complexity-Weight Low	Average	High
Internal Logical Files	7	10	15
External Interfaces Files	5	7	10
External Inputs	3	4	6
External Outputs	4	5	7
External Inquiries	3	4	6

Figure 2.1: Tabella FP/Peso

Le 5 categorie previste possono così essere descritte:

- Internal Logic File (ILP): set di dati omogeneo usato e gestito dall'applicazione.
- External Interface File (EIF): set di dati omogeneo usato dall'applicazione ma generato e gestito da altre applicazioni.
- External Input: operazione elementare per elaborare dati provenienti dall'ambiente esterno (all'applicazione)
- External Output: operazione elementare che genera dati per l'ambiente esterno.
- External Inquiry: operazione elementare che coinvolge Input e Output (senza significativa elaborazione di dati da files di logica).

La valutazione del livello di complessità può essere guidata dalla seguente tabella:

Table 2. FP Counting Weights

For Internal Logical Files and External Interface Files

	Data Elements		
Record Elements	**1 - 19**	**20 - 50**	**51+**
1	Low	Low	Avg.
2 - 5	Low	Avg.	High
6+	Avg.	High	High

For External Output and External Inquiry

	Data Elements		
File Types	**1 - 5**	**6 - 19**	**20+**
0 or 1	Low	Low	Avg.
2 - 3	Low	Avg.	High
4+	Avg.	High	High

For External Input

	Data Elements		
File Types	**1 - 4**	**5 - 15**	**16+**
0 or 1	Low	Low	Avg.
2 - 3	Low	Avg.	High
3+	Avg.	High	High

Figure 2.2: Tabella Peso/Record coinvolti

Per l'applicazione TravelDream abbiamo quindi individuato queste funzionalità principali:

- ILF: l'applicazione dovrà mantenere informazioni su Voli, Hotel, Escursioni, Utenti e Pacchetti. Ciascuna di queste entità può essere vista come semplice, poichè necessità di un numero ristretto di attributi. Quindi 5x7 = 35 FP.
- EIF: l'applicazione usa dati gestiti da applicazioni esterne, quindi non ha EIF.
- External Input: le principali funzionalità di input individuate sono
  - Login/Logout/Registrazione: considerabili semplici, quindi 3x3 = 9FP.
  - Aggiungere un pacchetto al carrello(anche su invito di partecipazione)/rimuoverlo(pagandolo o togliendolo dal carrello): tutte operazioni che coinvolgono Utente e Pacchetto Personalizzato. Possono comunque essere considerati semplici, quindi 2x3=6FP.

  - Modifica di un pacchetto personalizzato (cliente)/predefinito (impiegato): coinvolgendo sia Pacchetti che singoli componenti possono essere considerate complesse. Quindi 2x6=12FP.
  - Aggiunta/rimozione/regalo elemento in Giftlist: Pur coinvolgendo più entities in realtà può essere gestita in maniera semplice con dei semplici flag (come di fatti è stato implementato realmente) quindi possono essere considerate semplici, e volendo anche gestite da una sola funzionalità. 1x3=3FP
  - Aggiunta singolo componente (impiegato): possono essere considerate semplici perchè coinvolgono una sola entity, quindi 1x3=3FP
  - Aggiunta pacchetto predefinito: coinvolgendo le entities dei singoli componenti può essere considerata di media complessità, quindi 1x4=4FP.
  - Rimozione di un singolo componente (impiegato): dovendo eliminare anche tutti i pacchetti predefiniti che coinvolgono quel componente, può essere considerata media, quindi 1x4=4FP
  - Rimozione Pacchetto Predefinito (impiegato): non prevede l'eliminazione anche dei singoli componenti, quindi è considerabile semplice, 1x3=3FP.

- External Output: l'applicazione non prevede la produzione di particolari Output autonomi, quindi non ha External outputs.
- External Inquiry: il sistema permette all'utente di richiedere informazioni su:
  - pacchetti nel proprio carrello/storico e loro dettagli: considerabili semplici, 1x3=3FP.
  - Giftlist e stato dei componenti in lista: considerabile semplice, 1x3=3FP.
  - l'elenco dei pacchetti predefiniti (operazione possibile anche per l'impiegato): semplice, 1x3=3FP.

In totale quindi risultano 115 (U)FP.

2.2 Passaggio da UFP a KLOC

Per tale passaggio abbiamo usato la seguente tabella:

Table 4. UFP to SLOC Conversion Ratios

Language	Default SLOC / UFP	Language	Default SLOC / UFP
Access	38	Jovial	107
Ada 83	71	Lisp	64
Ada 95	49	Machine Code	640
AI Shell	49	Modula 2	80
APL	32	Pascal	91
Assembly - Basic	320	PERL	27
Assembly - Macro	213	PowerBuilder	16
Basic - ANSI	64	Prolog	64
Basic - Compiled	91	Query – Default	13
Basic - Visual	32	Report Generator	80
C	128	Second Generation Language	107
C++	55	Simulation – Default	46
Cobol (ANSI 85)	91	Spreadsheet	6
Database – Default	40	Third Generation Language	80
Fifth Generation Language	4	Unix Shell Scripts	107
First Generation Language	320	USR_1	1
Forth	64	USR_2	1
Fortran 77	107	USR_3	1
Fortran 95	71	USR_4	1
Fourth Generation Language	20	USR_5	1
High Level Language	64	Visual Basic 5.0	29
HTML 3.0	15	Visual C++	34
Java	53		

Figure 2.3: Relazione UFP/SLOC

Quindi 115 x 53 = 6095 SLOC.

L'analisi delle linee di codice sul progetto finale ha prodotto questi risultati (usando il tool CLOC):

```
Niccolos-MacBook:Downloads niccolozamperetti$ /Users/niccolozamperetti/Downloads/cloc-1.60-2.pl /Users/niccolozamperetti/Downloads/TravelDream
     119 text files.
     110 unique files.
     912 files ignored.

http://cloc.sourceforge.net v 1.60  T=1.09 s (83.3 files/s, 6520.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Java                            48           1183            233           3543
JavaServer Faces                25            430              0           1254
SQL                              1             31             44            184
XML                             12              4              5            155
Visualforce Component            4              0              0             37
Maven                            1              5              0             17
-------------------------------------------------------------------------------
SUM:                            91           1653            282           5190
-------------------------------------------------------------------------------
Niccolos-MacBook:Downloads niccolozamperetti$
```

Figure 2.4: Analisi CLOC

Per un totale quindi di 5190 SLOC = 5.190 KLOC.

La stima quindi è forse pessimistica, avendo previsto quasi il 20% in più di SLOC. Ma va comunque tenuto conto del fatto che un'analisi più precisa si sarebbe potuta fare usando gli Adjusted FP.

Capitolo 3

Stima di Costo con COCOMO

La stima del costo dell'applcazione con COCOMO, prevede l'utilizzo di formule dipendenti da alcuni parametri, a loro volta scalati sulla complessità dell'applicazione.

$$M = a_b S^{bb} \qquad T = c_b S^{db}$$

Tipo dell'applicazione	a_b	b_b	c_b	d_b
Organic mode	2,4	1,05	2,5	0,38
Semi-detached mode	3,0	1,12	2,5	0,35
Embedded mode	3,6	1,20	2,5	0,32

Tabella 3.1: Formule e Tabella COCOMO

Dove M rappresenta l'effort stimato, T il tempo stimato e S è la dimensione del progetto in KLOC.

Si può aggiungere anche la formula N=M/T per il numero di persone richieste per lo sviluppo dell'applicazione. Nel nostro caso, essendo N fissato a 3, calcoleremo la stima di M, e T lo ricaveremo da tale formula in base al valore stimato di M e N=3.

Il tipo dell'applicazione, sfruttando il riutilizzo di componenti già esistenti (offerti da JEE, Glassfish etc.) può essere considerata del tipo più semplice, cioè Organic.

In tal caso risulta:

- M = 2,4 x 5.190 ^ (1,05) = 13,5249...
- T = M / N = 4,50... [mesi]

I mesi-uomo sono pari a 20giorni-uomo, e 1 giorno-uomo è pari a 8ore-uomo. Quindi:

- T = 4,5 x 20 x 8 = 721,33 ore

Il nostro Time Reporting ammonta a 359 ore. Alle quali tuttavia vanno aggiunte le ore di Acceptance Testing (sul progetto di altri gruppi) e di stesura dei relativi documenti (oltre che anche l'attuale di project planning). Comunque, in questo caso, COCOMO risulta sovrastimare molto l'attuale quantità di tempo necessario.

Tuttavia prendendo in considerazione, anzichè l'intero S risultante da CLOC, solo le 3543 SLOC scritte effettivamente in Java dai componenti del gruppo, risulta:

- M = 2,4 x 3.543 ^ (1,05) = 9,0583...
- T = M / N = 3,019... [mesi]

I mesi-uomo sono pari a 20giorni-uomo, e 1 giorno-uomo è pari a 8ore-uomo. Quindi:

- T = 3,019 x 20 x 8 = 483,11 ore

Che è molto più vicino al reale tempo riscontrato nello sviluppo dell'applicazione.