****

Data Creazione: 04/12/2015

Data ultima modifica: 13/07/2015

**Configuration Management**

**Versione 4.1**

**Elisa Antolli**

**Alice Culaon**

**Diego Pillon**

jiojij

INDICE

1. TABELLA DELLE REVISIONI 3

2. SCOPO DEL DOCUMENTO 3

3. AUTORI DEL DOCUMENTO 3

4. GLOSSARIO 4

5. MEMBRI DEL TEAM 4

6. RESPONSABILITA’ DI PROGETTO 5

7. COMUNICAZIONI 6

8. L’ AMBIENTE, I TOOLS E L’ INFRA-STRUTTURA 6

9. GESTIONE DOCUMENTI 7

10. MODELLO DI SVILUPPO 14

11. CICLO DI VITA 15

12. LA GESTIONE DEI CAMBIAMENTI 17

13. STILE DEL CODICE 19

APPENDICE A:   
Code Conventions for the JavaTM Programming Language 20

1. TABELLA DELLE REVISIONI

|  |  |  |  |
| --- | --- | --- | --- |
| **Rev./Ver.** | **Data** | **Descrizione** | **Autore** |
| 1/1.0 | 01-07-2015 | Creazione del Documento; definizione di alcune parti | Alice Culaon |
|  |  |  |  |
| **Tot. Rev.** |  | **Versione corrente 1.0** |  |

1. SCOPO DEL DOCUMENTO

L'analisi dei costi è una stima iniziale di quando l’intero sistema andrà a costare. L’analisi ha lo scopo di stabilire un riferimento per stimare i costi che l'intero progetto richiederà al committente.

Si andranno ad analizzare quindi, i fattori che influenzano il costo dell’intero sistema, valutando la dimensione e la complessità del software, i fattori umani impiegati, la stabilità dei requisiti e l'ambiente di sviluppo per poter giungere ad una stima del prezzo di vendita da fornire al committente.

In questo documento verranno analizzate due stime dei costi differenti, avendo così la possibilità di confrontare i risultati per avere una stima più precisa dell'effettivo costo finale del progetto.

Le due stime sono di diversa natura tra loro: una è di tipo algoritmico, mentre l’altra sfrutta le esperienze di progetti passati, per poter fare una stima dei costi.

Il primo modello preso in considerazione è il modello COCOMO (COnstructive Cost Model) un modello matematico espresso da più formule che forniscono lo sforzo e il tempo necessario per lo svolgimento delle attività dal progetto fino al test di accettazione che prevede un'analisi dettagliata del progetto, basandosi su parametri e valori numerici.

Il secondo modello scelto è il modello “giudizio degli esperti”, nel quale uno o più esperti di sviluppo del software, usano la loro esperienza per prevedere il costo del sistema. E’ una metodologia molto poco costosa per il team, ma richiede una buona esperienza da parte degli esperti.

GLOSSARIO

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Termine** | | **Descrizione** | | **Pagine** | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  | |  | |  | |
|  |  | |  | |  |
| **Termini del glossario: 7** | | | | | |

1. il modello cocomo

Il modello COCOMO (COnstructive COst Model) è un modello matematico basato su una procedura algoritmica progettato per seguire un modello di sviluppo a cascata, in particolare seguendo le seguenti fasi:

1. Pianificazione ed analisi dei requisiti

2. Progetto dell'applicazione

3. Sviluppo

4. Integrazione e test

COCOMO permette di calcolare il costo derivandolo dall’impegno (mesi persona) neccessario allo sviluppo che si ricava da una funzione che lo correla alla dimensione del sw da sviluppare.

Tale modello è un modello matematico espresso da più formule che forniscono lo sforzo e il tempo necessario per lo svolgimento delle attività dal progetto fino al test di accettazione. I costi per la pianificazione e l'analisi dei requisiti sono calcolati a parte.

* 1. motivazione

La scelta del modello di stima è stata fatta per vedere se la stima eseguita con due diverse metodologie, tornavano più o meno lo stesso risultato. Questo modello scelto è considerato un modello statistico ed analitico, mentre il secondo si basa solo sull’esperienza dei membri del team. Il modello COCOMO quindi, l’abbiamo scelto per verificare che i due risultati non si discostino di molto l’uno dall’altro, utilizzando formule matematiche e tabelle, permettendo dunque, di arrivare ad una stima dei costi attendibile.

* 1. stima dell’effort

Il modello COCOMO calcola lo sforzo di sviluppo del software come funzione della grandezza del programma espresso in KDSLOC e su degli indici dei costi.

Il costo quindi, è stimato da una funzione matematica degli attributi di prodotto, progetto e processi i quali valori vengono stimati dai product manager.

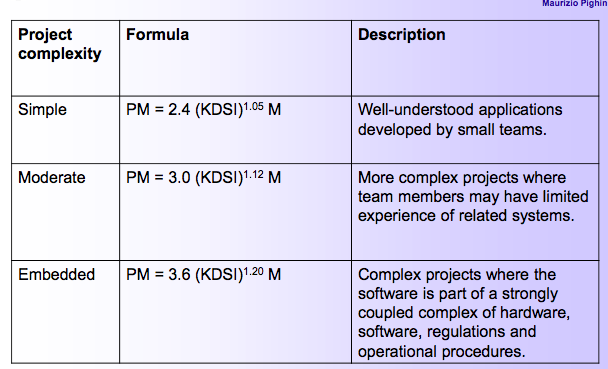
Questo modello calcola la stima dei costi seguendo la seguente formula:

**Effort= A\*SizeB \*M**

I parametri utilizzati nella formula sono:

* Effort: è il risultato della stima, indica la quantità di lavoro richiesta per completare il progetto calcolata in PM (mesi-persona).
* A: è una costante dipendente dall’organizzazione, si riferisce alla dimensione del team, dalla sua esperienza e dalla struttura dell’azienda.
* Size: la dimensione in KDSI (o KDSLOC), indica la lunghezza delle linee di codice del progetto.
* B: riflette la sproporzione dello sforzo per i progetti più grandi.

Le variabili A e B possono cambiare valore a seconda della dimensione del progetto come mostrato nella figura seguente.

****

**3.3 KDSLOC (Kilo Delivered Source Lines Of Code)**

Questo parametro non è ancora noto in questa fase in quanto non è ancora iniziata la stesura del codice; esso sarà conosciuto solamente alla fine del progetto. Per risolvere questo problema, viene diviso il sistema in diverse funzionalità che soddisfano i requisiti: ciò viene fatto attraverso i Function Points che rappresentano appunto le funzioni principali del progetto. Il risultato del calcolo del loro calcolo, restituirà quindi una stima della dimensione in Function Points, la quale verrà successivamente convertita il KDSLOC da inserire successivamente nella formula di calcolo dell'Effort.

Per effettuare la conversione useremo la seguente formula:

**KDSLOC = (FP \* LangIndex) / 100**

Riguardo al calcolo del Funcion Point, lo descriveremo nel paragrafo seguente, mentre il parametro LangIndex, è ricavato dalle tabelle di riferimento di Caper Jones (tabella sottostante), da cui si ricava una stima del numero di righe di codice per Function Point a seconda del linguaggio di programmazione che si intende utilizzare.



**3.4. FUNCTION POINTS**

I Function Points vengono calcolati attraverso la seguente formula:

**FP = UFP \* TCF**

I parametri utilizzati nella formula sono:

* FP: Function Points, il valore utilizzato per stimare la dimensione del sistema.
* UFP: Unadjusted Funcion Points, il valore dei fattori di peso divisi in 5 indici.
* TCF: Tecnical Complexity Factors, valore dei fattori di aggiustamento del peso calcolato dai Function Points.

Per calcolare il valore dei Function Points, vanno seguiti i seguenti 5 passi:

1. Identificazione e classificazione di 5 indici

2. Definizione della complessità di ogni indice

3. Calcolo dell'Unadjusted Function Points

4. Calcolo del Technical Complexity Factor

5. Calcolo dei Function Points

**3.4.1** **UNADJUSTED FUNCTION POINTS**

Il calcolo dell'Unadjusted Function Points avviene identificando nella descrizione dell'applicazione che si vuole implementare, ovvero nella specifica funzionale del sistema, gli elementi appartenenti alle seguenti classi:

- **External Input (EI):** dati in Input, cioè informazioni e/o dati forniti dall'utente al sistema, come ad esempio inserimento, modifica ed eliminazione di dati.

- **External Output (EO):** dati di Output, cioè informazioni e/o dati forniti all'utente dall'applicazione riguardo eventi, è il risultato ricavato dopo elaborazioni.

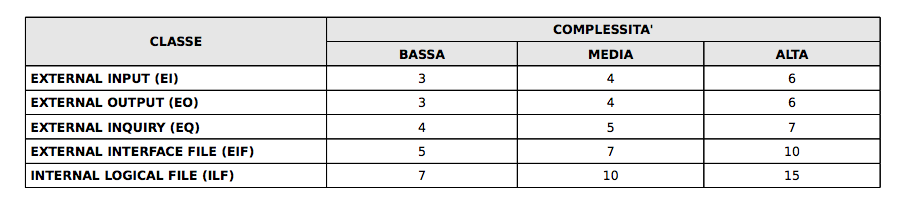
- **External Inquiry (EQ):** Interrogazioni, cioè sequenze interattive di richieste-risposte, come visualizzazione, esportazione e stampa di dati.

- **External Interface File (EIF):** interfacce dell'applicazione, cioè interfacce con altri sistemi informativi esterni, ovvero altri programmi.

- **Internal Logical File (ILF):** file interni dell'applicazione, cioè i file logici principali gestiti nel sistema necessari per l'elaborazione dei dati e per garantire il corretto funzionamento del sistema.

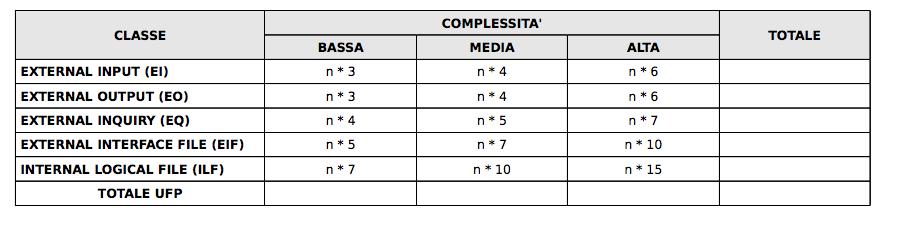
In seguito all’identificazione, ogni funzionalità del sistema viene collocata nell'apposita classe di appartenenza, a cui viene assegnato un peso di complessità del requisito, che può assumere valore basso, medio o alto in relazione alla seguente tabella:

(da inserire tabella) ad esempio:



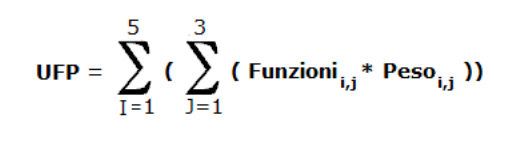
Al fine dell'analisi delle funzionalità, bisognerà contare gli elementi di ciascuna classe, vale a dire il numero di funzionalità di ogni classe, per ciascuna complessità riempiendo la tabella nel seguente modo:

(da inserire tabella)



Dove n indica il numero di funzionalità collocate in una specifica classe con una determinata complessità.

In seguito, Per calcolare il totale degli Unadjusted Function Points basterà applicare la seguente formula:



Dove I indica il tipo di classe e J indica la complessità (bassa, media, alta).

3.4.1 **TECHNICAL COMPLEXITY FACTORS (TFC)**

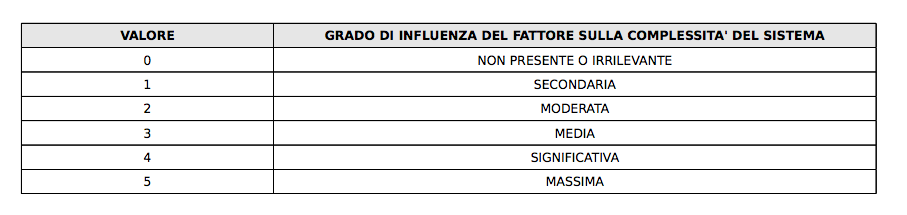
Il Technical Complexity Factors è il fattore di complessità tecnica del sistema, che serve per introdurre nel calcolo dei Function Points un valore di influenza rispetto alle sue caratteristiche generali.

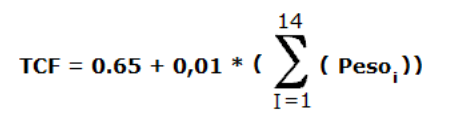
Il calcolo del Technical Complexity Factors avviene sulla base di 14 caratteristiche (o fattori) del sistema a cui ad ognuna viene assegnato un peso, che varia da 0 a 5, a seconda dell'influenza che quella specifica caratteristica ha sulla complessità del progetto.

Le caratteristiche del sistema che dovranno essere valutate e a cui dovranno essere assegnati dei pesi sono riportate nella seguente tabella:



Come accennato in precedenza, il valore del peso che può essere assegnato ad ogni caratteristic assume un valore compreso tra 0 e 5 in relazione alla seguente tabella:

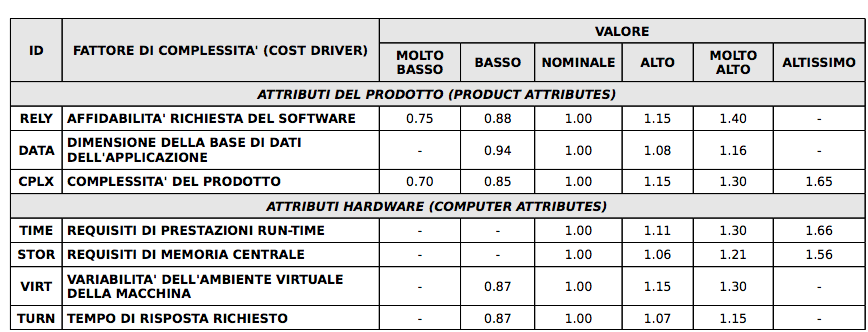
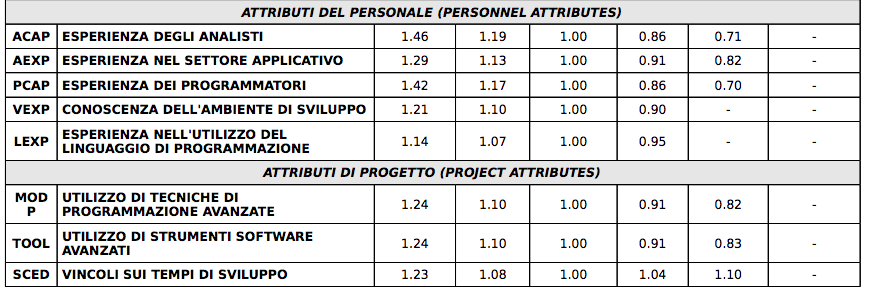


Infine, dopo aver riempito la tabella dei fattori con i relativi pesi, per calcolare il Technical Complexity Factors (TCF) si utilizza la seguente formula:

**3.4.2 FATTORE MOLTIPLICATIVO**

Il fattore moltiplicativo M indica il coefficiente di assestamento, vale a dire un valore che indica lo scostamento dal valore normale di un determinato attributo all'interno del sistema. Per calcolare questo fattore, si analizzano quattro categorie di attributi che influenzano il costo, che identificano diversi aspetti del sistema, e per ognuno di questi si hanno dei sotto-attributi con dei valori predefiniti che ne determinano la complessità in base all'importanza che hanno all'interno del software.

Gli attributi che influenzano il fattore moltiplicativo M sono 15 (detti Cost Driver) ed ognuno di essi ha un valore su una scala di 6 valori che varia da “Molto Basso” ad “Altissimo” come riportato nella seguente tabella:



Il fattore moltiplicativo M, verrà dato dalla moltiplicazione di tutti i parametri descritti nella tabella precedente, ed avrà la seguente forma:

**M = RELY \* DATA \* CPLX \* TIME \* STOR \* VIRT \* TURN \* ACAP \* AEXP \* PCAP \* VEXP \* LEXP \* MODP \* TOOL \* SCED**

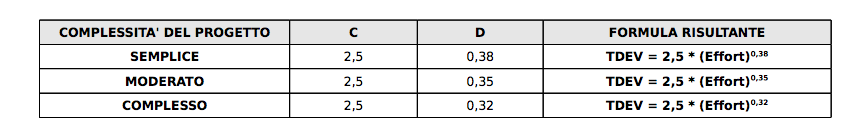
**3.5 STIMA DEL TEMPO DI SVILUPPO (TDEV)**

Una volta conosciuto il costo finale dello sforzo necessario per lo sviluppo del sistema software espresso in mesi-persona (PM), è possibile calcolare il tempo solare minimo di sviluppo TDEV (Development Time), calcolato tramite la seguente formula:

**TDEV = C \* Effort D**

In cui:

* **Effort:** è il risultato dello sforzo espresso in numero di persone/mese necessarie per lo sviluppo del progetto
* **C** e **D**: sono costanti legate alla complessità del progetto in relazione alla seguente tabella:



Una volta noto il numero di persone/mese ed il tempo necessario per lo sviluppo del progetto è possibile stimare il numero di programmatori da impiegare nello sviluppo del progetto tramite la seguente formula:

**N = Effort / TDEV**

4. **GIUDIZIO DEGLI ESPERTI**

Il secondo modello di stima preso in considerazione è il modello “Giudizio degli esperti” in cui vengono calcolarti tutti i costi che il team dovrà sostenere per lo sviluppo del progetto.

Nel calcolo vengono presi in considerazione le ore già lavorate e le ore da lavorare, in accordo con il cronogramma.

Nell’analisi dei costi vengono elencati tutti i possibili costi che l’azienda dovrà sostenere per poter sviluppare il sistema, partendo dai bisogni primari come ad esempio i computer, le stampanti… fino ai vari tool che vengono utilizzati nello sviluppo e i veri mezzi che il team utilizza per poter comunicare.

Nell’analisi sono stati calcolati anche i costi dell’affitto, dell’adsl ed inoltre è stata fatta un accurata analisi del calcolo delle imposte, tra cui IRPEF e INAIL.