Project Management dei Progetti Software

Obiettivi della lezione

- Il Project Management
- Tecniche per la stima dei costi sw
- Misurare le dimensioni di un progetto sw
  - Linee di codice
  - Function Point
- Stima Algoritmica
  - COCOMO

Due definizioni

- **Progetto**: impresa temporanea intrapresa per raggiungere un particolare scopo
- **Project Management**: applicazione di conoscenze, abilità, tecniche e strumenti alle attività di progetto allo scopo di soddisfarne i requisiti

Il Project Manager

- Il Project Manager (PM, o responsabile di progetto) controlla la pianificazione ed il budget di progetto
- Durante tutto il ciclo di vita deve controllare **costi** e **risorse** di un progetto:
  - **Inizialmente** per verificare la fattibilità del progetto
  - **Durante il progetto** per controllare che le risorse non vengano sprecate ed i budget vengano rispettati
  - **Alla fine** per confrontare preventivo e consuntivo
- Stimare il costo (specie del software) è un'arte
  - Occorre creare un **database dei costi** per supportare le stime correnti e future

Project Managers

- Anni ’50 fino agli ’80 – *Implementatori*
  - Ambito ristretto di attività.
  - Focalizzato principalmente sugli aspetti tecnici.
  - Abilità ristrette: budgeting, scheduling e interpretazione specifiche
- Oggi – *Decisori tattici*
  - Equivalenza tra Projects management e Business management.
  - Abilità personali più ampie, pressione elevata sulle prestazioni
  - Maggiore autorità e responsabilità.
  - Compenso aumentato, spesso legato ai risultati e non agli sforzi.

SPMP: Sw Process Management Plan

- IEEE Standard 1058.1
  - Largamente usato
  - Adatto per ogni tipo di sw
- Determina le unità di lavoro
- Stima delle risorse
- Calcolo del budget necessario
- Costruzione di un piano temporale dettagliato

IEEE SPMP

1. Introduction
   - 1.1 Project Overview
   - 1.2 Project Deliverables
   - 1.3 Evolution of the Software Project Management Plan
   - 1.4 Reference Materials
   - 1.5 Definitions and Acronyms
2. Project Organization
   - 2.1 Process Model
   - 2.2 Organizational Structure
   - 2.3 Organizational Boundaries and Interfaces
   - 2.4 Project Responsibilities
3. Managerial Process
   - 3.1 Management Objectives and Priorities
   - 3.2 Assumptions, Dependencies, and Constraints
   - 3.3 Risk Management
   - 3.4 Monitoring and Controlling Mechanisms
   - 3.5 Staffing Plan
4. Technical Process
   - 4.1 Methods, Tools, and Techniques
   - 4.2 Software Documentation
   - 4.3 Project Support Functions
5. Work Packages, Schedule, and Budget
   - 5.1 Work Packages
   - 5.2 Dependencies
   - 5.3 Resources Requirements
   - 5.4 Budget and Resource Allocation
   - 5.5 Schedule

Additional components

Project Management Handbook

- PMBOK è una pubblicazione del PMI che descrive la conoscenza e i metodi necessari ai PM “nella maggior parte dei progetti”
- Viene aggiornato periodicamente
- Struttura il Project Management definendo un certo numero di aree

Aree del PMBOK

- Project Integration Management
- Project Scope Management
- Project Time Management
- Project Cost Management
- Project Quality Management
- Project Human Resources Management
- Project Communication Management
- Project Risk Management
- Project Procurement Management

La gestione del tempo di progetto

- Definizione delle attività
- Sequenziamento delle attività
- Stima delle risorse per le attività
- Stima della durata delle attività
- Schedulazione delle attività
- Controllo delle attività

Grafici Gantt e PERT

- Il grafico Gantt permette di definire la durata - e sotto certe forme i ruoli - delle attività (ma non le loro dipendenze)
- La definizione delle dipendenze tra le attività si ottiene con un grafico PERT

Grafico Gantt

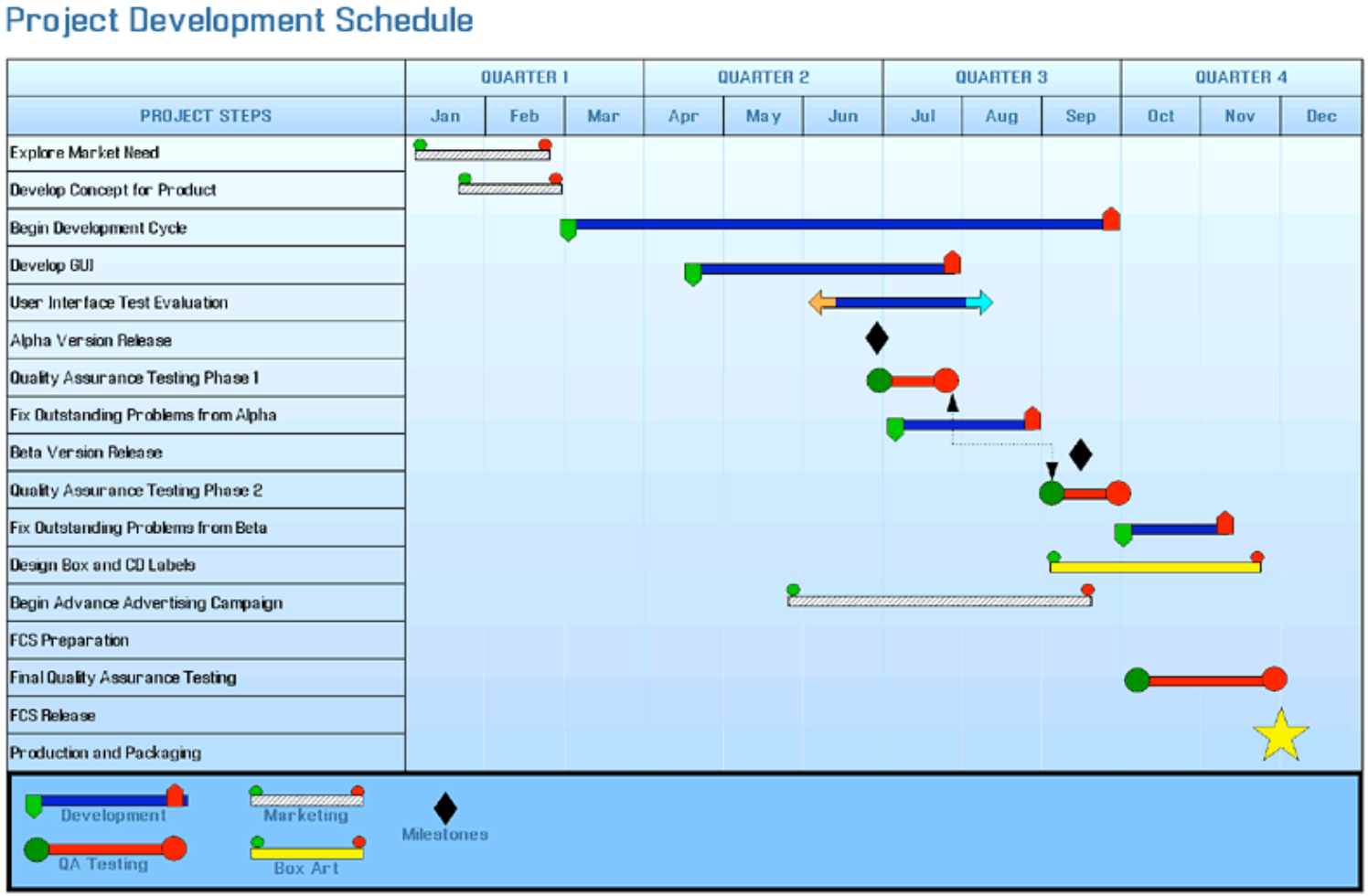
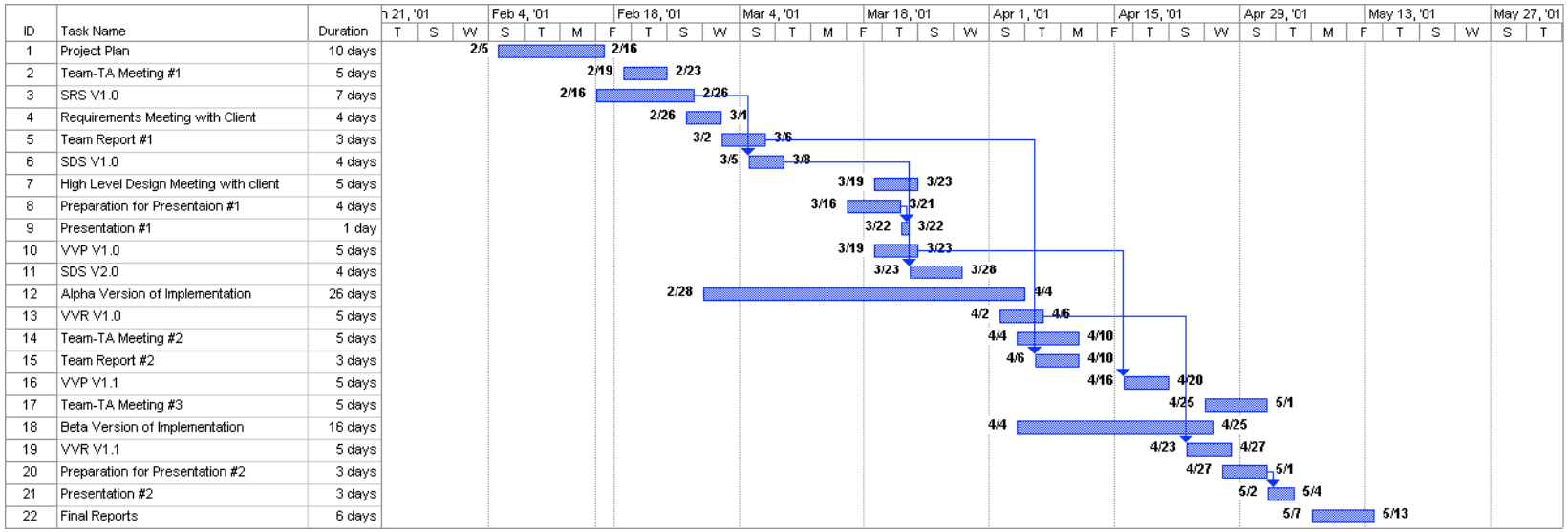


Grafico Gantt con deriverables



ID	Task Name	Duration
1	Project Plan	10 days
2	Team-TA Meeting #1	5 days
3	SRS V1.0	7 days
4	Requirements Meeting with Client	4 days
5	Team Report #1	3 days
6	SDS V1.0	4 days
7	High Level Design Meeting with client	5 days
8	Preparation for Presentaion #1	4 days
9	Presentation #1	1 day
10	VVP V1.0	5 days
11	SDS V2.0	4 days
12	Alpha Version of Implementation	26 days
13	VVR V1.0	5 days
14	Team-TA Meeting #2	5 days
15	Team Report #2	3 days
16	VVP V1.1	5 days
17	Team-TA Meeting #3	5 days
18	Beta Version of Implementation	16 days
19	VVR V1.1	5 days
20	Preparation for Presentation #2	3 days
21	Presentation #2	3 days
22	Final Reports	6 days

Grafico PERT
(Program Evaluation Review Technique)

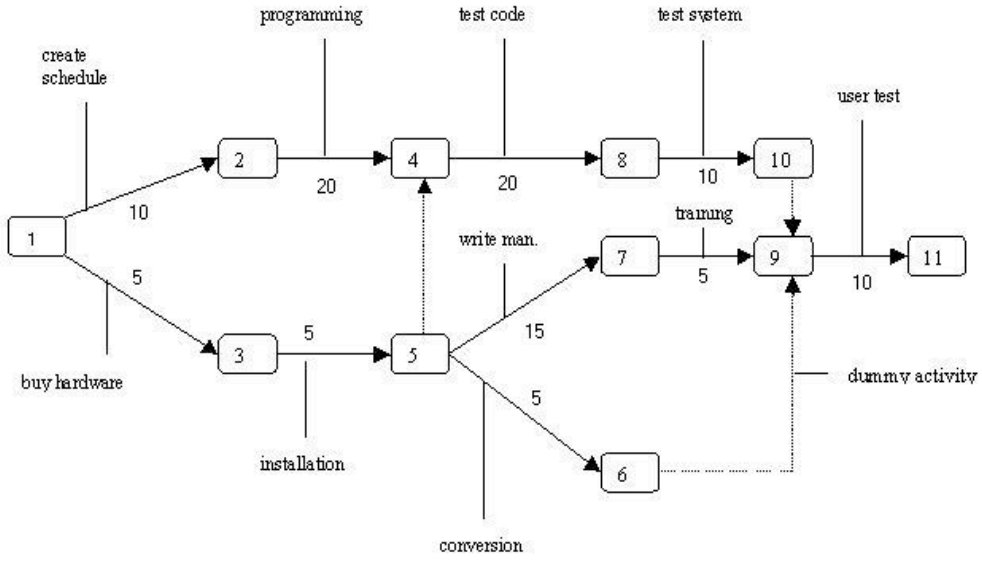


Fig. 1:
PERT Chart

* Numbered rectangles are nodes and represent events or milestones.
* Directional arrows represent dependent tasks that must be completed sequentially.
* Diverging arrow directions (e.g. 1-2 & 1-3) indicate possibly concurrent tasks
* Dotted lines indicate dependent tasks that do not require resources.

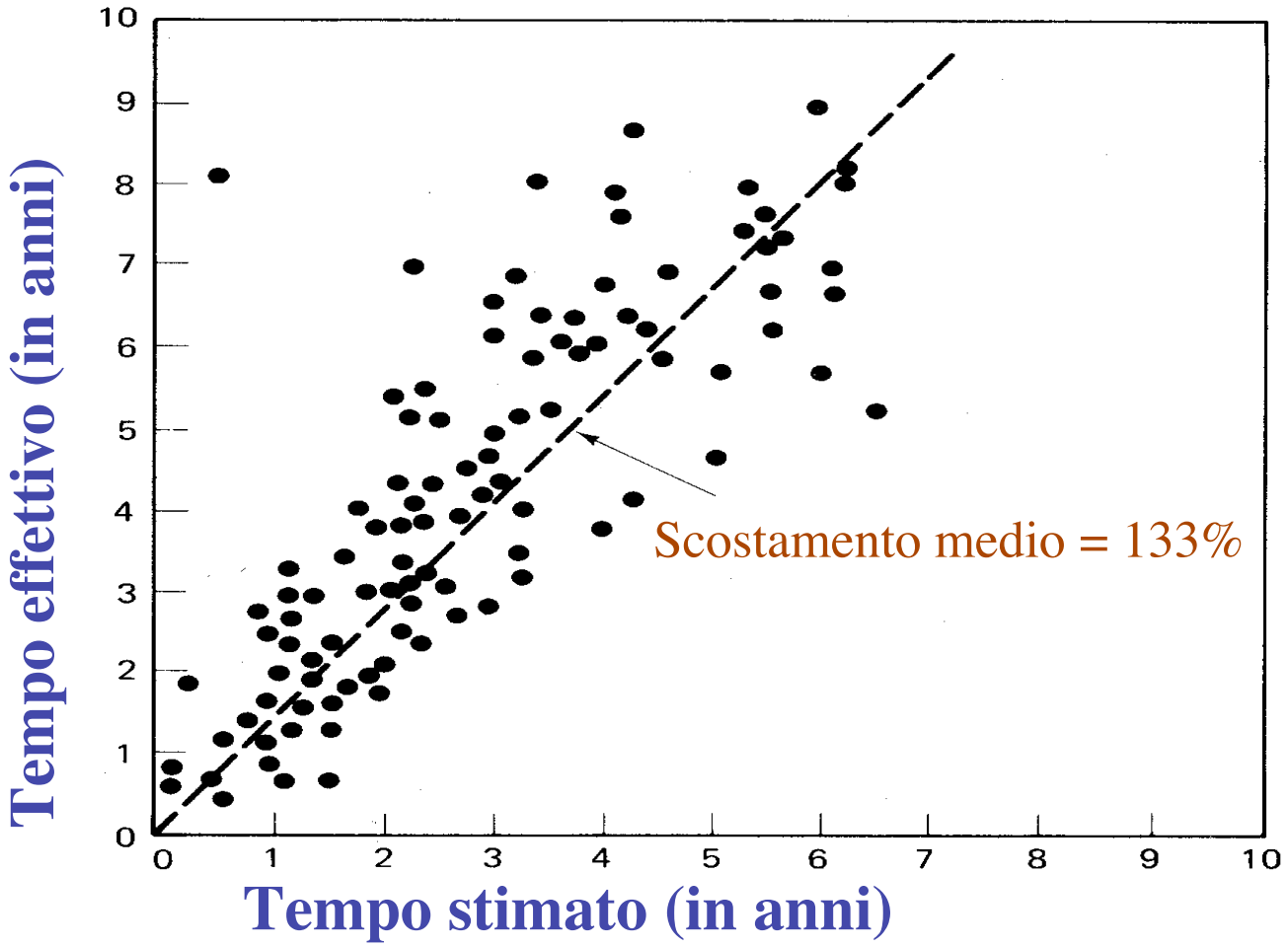
Costruire e analizzare un PERT

- Identificare le attività e le scadenze
- Determinare la sequenza delle attività
- Costruire il grafo
- Stimare il tempo necessario per ciascuna attività
- Determinare il *cammino critico*
- Aggiornare il grafico PERT mentre il progetto va avanti

Project Cost Management

- I progetti informatici di solito vengono sottostimati sia in tempo che in risorse
- Secondo uno studio del 1995 le risorse a consuntivo erano in media il 189% delle stime preventive; nel 2001 c'era stato in media un miglioramento al 145%

Valutare l'errore di stima



Fallimenti

Dalle prime pagine di quotidiani americani:

- "Computer Bumbling Costs the State $1 Billion: the state of California had a series of expensive IT project failures in the late 1990s, costing taxpayers nearly $1 billion… "ironic that the state which leads in creation of computers is the state most behind in using computer technology to improve state services".
- "…The Internal Revenue Service (IRS) managed a series of project failures that cost taxpayers over $50 billion a year—roughly as much money as the annual net profit of the entire computer industry."
- "…Connecticut General Life Insurance Co. sued PeopleSoft over an aborted installation of a finance system."

Cos'è il costo di un progetto?

- Il **costo** è una *risorsa* impiegata o ceduta per raggiungere un *obiettivo* o per ottenere qualcosa in cambio
- Il costo si misura in *unità monetarie*($ o €)
- Il Project Cost Management definisce i processi necessari ad assicurare che il progetto si concluda con successo entro i termini di budget prestabiliti

Processi di Project Cost Management

1. *Resource planning*: determinare e pianificare le risorse da usare
2. *Cost estimating*: stimare costi e risorse necessarie ad un progetto
3. *Cost budgeting*: allocare il costo complessivo su compiti individuali per stabilire un riferimento per valutare le prestazioni
4. *Cost control*: controllare le modifiche al budget durante il progetto

Principi del Cost Management

- Molti dirigenti sono più esperti di finanza che di IT, quindi i PM di progetti IT devono familiarizzare col loro linguaggio
  - I profitti sono le entrate meno le spese
  - Il costo di ciclo di vita è la stima del costo di progetto più il costo della manutenzione del prodotto
  - L'analisi di flusso di cassa (cash flow) determina su base annua i costi stimati ed i benefici dati da un progetto
  - Costi e benefici possono essere tangibili o intangibili, diretti o indiretti

Resource Planning

- La natura del progetto ed il tipo di organizzazione che lo intraprende influenzano la pianificazione delle risorse
- Problematiche tipiche:
  - Quanto saranno difficili certi compiti specifici?
  - Questo progetto ha qualcosa di speciale che possa influenzare la gestione delle risorse?
  - Qual'è la storia dell'organizzazione nell'intraprendere progetti analoghi?
  - L'organizzazione ha o può acquisire le persone, gli strumenti ed i materiali adeguati per eseguire il lavoro?

Stima dei costi delle risorse necessarie

Gran parte dei costi dei progetti IT riguardano i costi delle risorse umane

Table 7-2: Maximum Departmental Headcounts by Year

DEPARTMENT	1994	1995	1996	1997	1998	TOTALS
Information Systems	24	31	35	13	13	116
Marketing Systems	3	3	3	3	3	15
Reservations	12	29	33	9	7	90
Contractors	2	3	1	0	0	6
Totals	41	66	72	25	23	227

Stimare i costi

- Un prodotto importante del Project Cost Management è una *stima dei costi*
- Esistono parecchi tipi di stima dei costi, e di corrispondenti tecniche e strumenti per calcolarli
- Occorre anche sviluppare un *piano di cost management* che descriva come il progetto possa gestire variazioni dei costi

Tecniche di stima dei costi

- Tre tecniche principali:
  - **top-down** (o analogica): uso del costo reale di un progetto analogo come base della stima del nuovo progetto
  - **bottom-up**: stima dei compiti individuali che compongono il progetto e loro somma per ottenere la stima totale
  - **parametrica**: stima basata sull'uso di un modello matematico che usa parametri di progetto (eg. COCOMO)

Problemi tipici con le stime dei costi IT

- L'attività di stima durante un grosso progetto software richiede sforzo notevole ed è un compito complesso
- Molte persone che effettuano stime hanno scarsa esperienza in tale attività: occorre formazione specifica
- Di solito si sottostima: è una tendenza molto diffusa
- Le revisioni di stima devono basarsi sulle domande "giuste", per evitare che le stime siano falsate
- Ai dirigenti occorre un numero per fare un offerta, non una vera stima. Poi è compito dei PM mediare con gli sponsor di un progetto per creare stime di costo realistiche

I componenti del costo del Sw

- I principali componenti di costo sw sono:
  - Costo dell'hardware di sviluppo
  - Costo del software di sviluppo
  - Costo delle risorse umane (**sforzo**)
  - **Durata** complessiva
- Lo *sforzo* necessita una stima di tipo predittivo e viene misurato in mesi-persona (man-month)

Alcune formule magiche

Per stimare rapidamente alcuni parametri chiave di un progetto, esistono alcune regole euristiche (Boehm) che aiutano a dare una stima "immediata" se è nota la misura dello **sforzo** in mesi-persona (mp)

- **Durata media di un progetto**:
  Durata = 2.5 * radice cubica di mp
- **Dimensione ottimale dello staff**:
  Dimensione_Staff = radice quadrata di mp
- **Durata minima di un progetto**:
  Durata_minima = 0.75 * radice cubica di mp

Strumenti Sw per il Cost Management

- I **fogli elettronici** si usano per resource planning, cost estimating, cost budgeting, e cost control
- Alcune aziende usano **applicazioni finanziarie** centralizzate per gestire le informazioni sui costi
- Gli strumenti per Project Management offrono di solito parecchie funzioni di **analisi dei costi**

Eempio di tool per PM

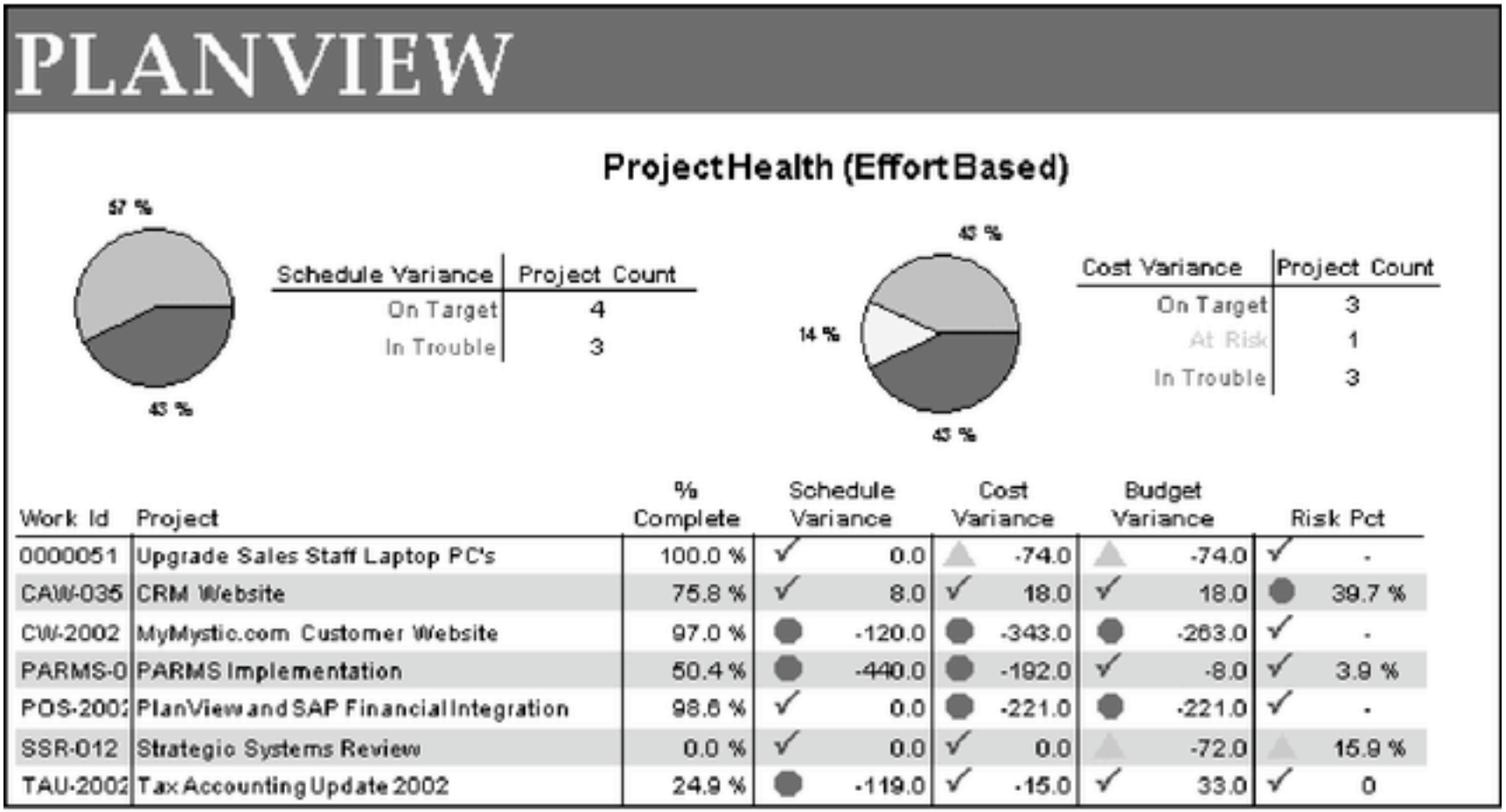
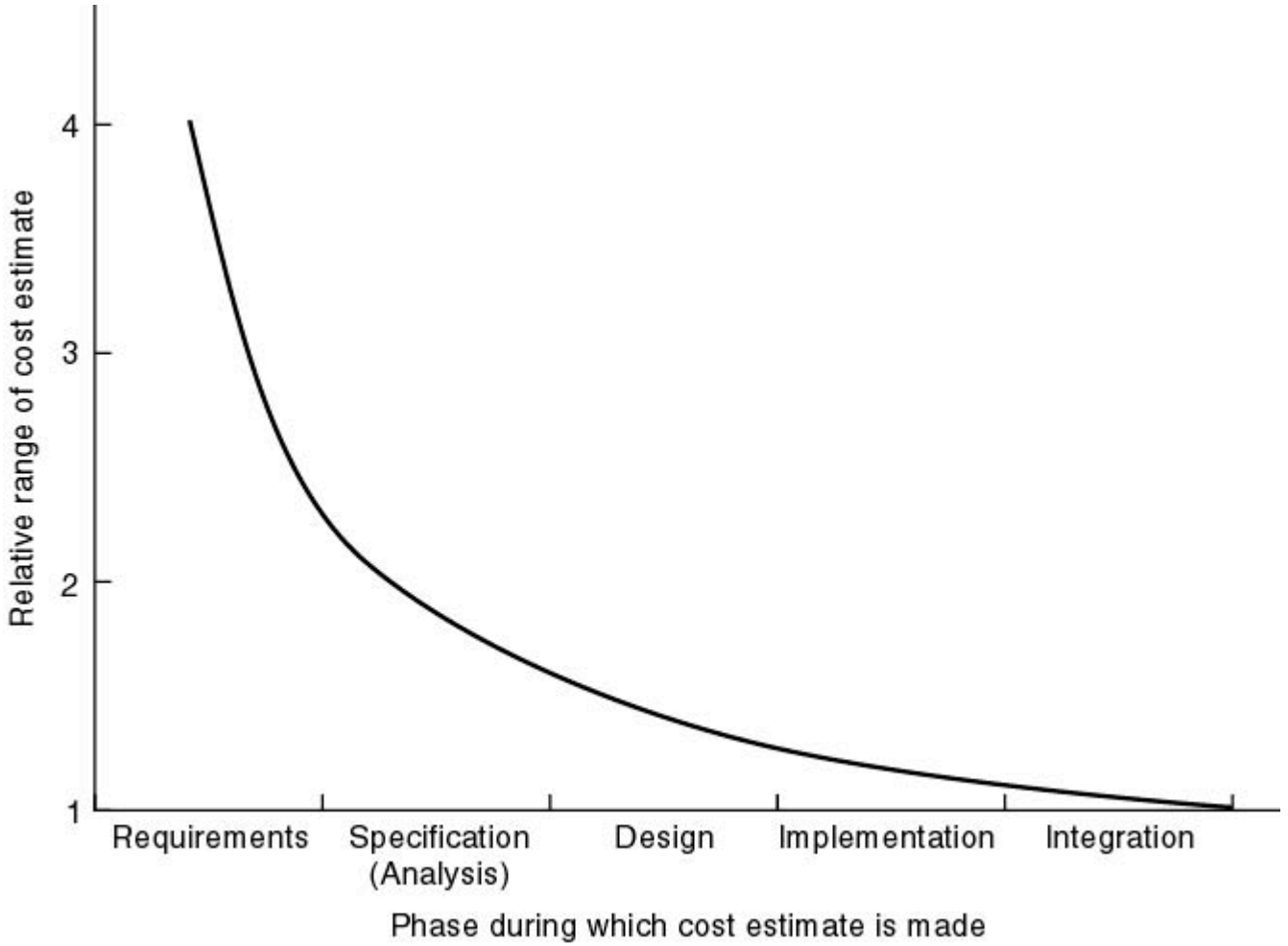
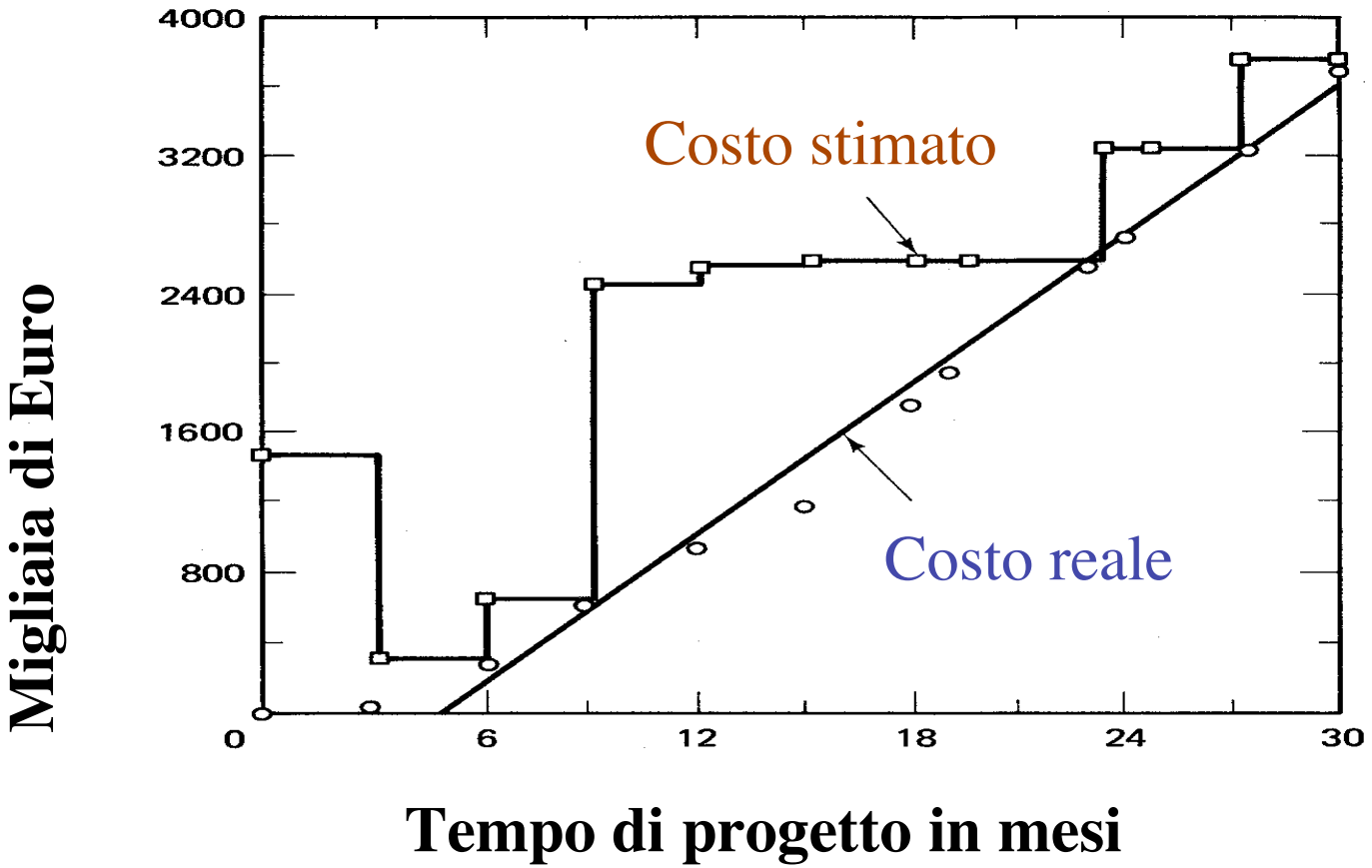


Figure 7-4. ProjectHealth Screen from PlanView Software

La stima migliora durante il processo?



Esempio di stime di costi



Tecniche di Stima dei Costi

- Il giudizio di esperti (metodo Delphi):
  - Vengono consultati esperti sulle tecniche di stima del software, i quali stimano indipendentemente il costo del progetto e reiterano finché non si trovano in accordo
- Stima per analogia:
  - Si applica quando sono già stati completati altri progetti con caratteristiche simili.
  - Il costo viene valutato in analogia al progetto già finito

Tecniche di Stima dei Costi

- Legge di Parkinson:
  - La previsione di durata e sforzo aumenta fino a coprire il tempo e le persone disponibili
  - Il costo è quindi determinato dalle risorse disponibili piuttosto che dal bisogno oggettivo
- Pricing to Win:
  - Il costo del software viene stimato in base alla spesa che il cliente è disposto a sostenere per il progetto.

Tecniche di Stima dei Costi

Modelli parametrici (o algoritmici):

- Il costo è stimato come una funzione matematica di attributi i cui valori sono determinati dal manager di progetto
- La funzione deriva da uno studio di precedenti progetti (regressione)
- Attributi chiave: quelli che hanno un impatto significativo sul costo

Tecniche di Stima dei Costi

- Uso di strumenti di intelligenza artificiale e apprendimento automatica
  - Reti neurali, Case Based Reasoning, Fuzzy logic
  - Il sistema impara a stimare usando un insieme di dati da progetti già finiti (detto training set)
  - La stima non è intellegibile, può essere inaccurata se il training set è piccolo

Tecniche di Stima dei Costi

- In molti progetti alcune tecniche possono essere combinate
- Se si predicono costi radicalmente differenti allora bisogna cercare più informazioni e le stime vanno ripetute fino a quando non convergono.
- L'accuratezza della stima deve crescere con il progredire del progetto.

Misurare le dimensioni del Progetto

- La ***dimensione*** del progetto è il primo coefficiente di costo in molti modelli
- Esistono due misure principali per stimare la dimensione del software:
  - **Le linee di codice**
  - **I punti funzione (Function point)**
- Entrambe hanno bisogno di dati storici per poter essere "calibrate" all'organizzazione che le usa

Linee di Codice

- La misurazione più comune della dimensione di un progetto software è il numero di linee di codice *sorgente* (LOC).
- LOC può tener conto delle linee vuote o dei commenti (CLOC).
- In generale si ha: LOC = NCLOC + CLOC, cioè i commenti si contano

Esempio: stima LOC (da Pressman)

• Interfaccia utente	2.300
• Gestione dati bidimensionali	5.300
• Gestione dati tridimensionali	6.800
• Gestione del db	3.350
• Visualizzazione grafica	4.950
• Controllo dispositivi	2.100
• Moduli di analisi del progetto	8.400
Totale LOC stimate	33.200

Produttività “storica” per sistemi di questo tipo = 620 LOC/pm.

Costo del lavoro =€8000 /mese, quindi ogni LOC costa €13.

La stima di costo totale è €431,000 mentre la stima di sforzo è 54 pm

Stime quantitative: LOC

- KLOC = Migliaia di **linee di codice** sorgente
- Metriche:
  - $ per KLOC
  - errori o difetti per KLOC
  - LOC per mese/persona
  - pagine di documentazione per KLOC
  - Errori per mese-persona
  - $/pagina di documentazione
- Il codice è il prodotto tangibile del processo di sviluppo, ed esiste già parecchia letteratura in proposito
- Dipende dal linguaggio di programmazione
- Penalizza (sottovaluta la produttività) programmi scritti bene e concisi

Aspetti critici delle stime dimensionali

1. Difficile stimare la dimensione in LOC
2. La stima LOC non tiene conto della diversa complessità e potenza delle istruzioni (di uno stesso linguaggio o di linguaggi diversi)
3. Difficile definire in modo preciso il criterio di conteggio (istruzioni spezzate su più righe, più istruzioni su una riga...)
4. Una maggior produttività in LOC potrebbe comportare più codice di cattiva qualità?

Function Point [Albrecht]

- La metrica Function Point misura le funzionalità di un sistema.

"The original objective of the Function Points work was to define a measure [that would] help managers analyze application development and maintenance work and highlight productivity improvement opportunities."

"The Function Points method measures the equivalent functions of end-user applications regardless of the language, technology, or development environment used to create the application."

Le metriche funzionali

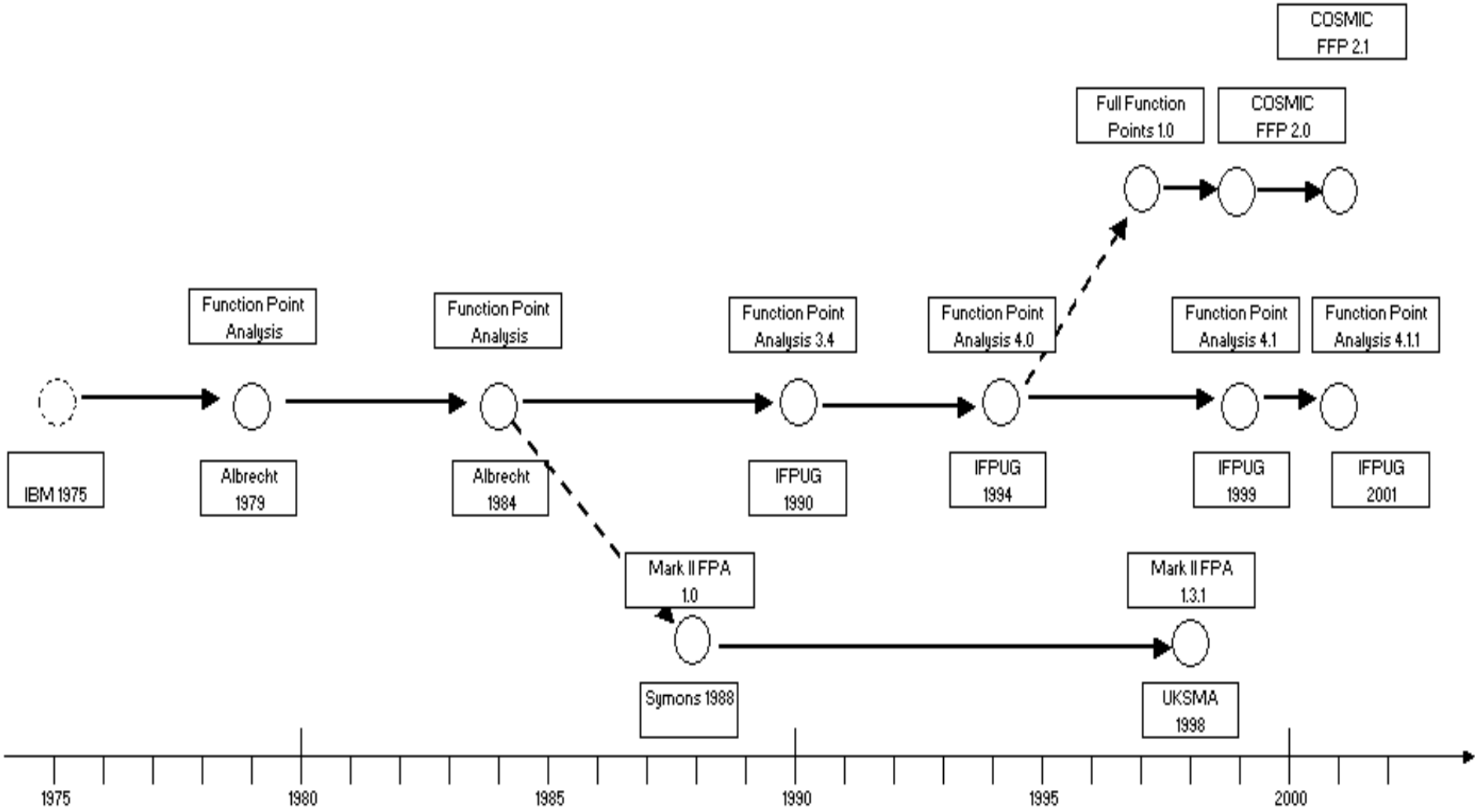
- FP-IFPUG, nasce negli USA
- COSMIC-FFP (ISO/IEC 19761:2003)
- MkII FPA (**ISO**/IEC 20968) nasce in UK
- NESMA (**ISO**/IEC 24570) nasce in Olanda

Sono metodi funzionali di “prima generazione” progettati 10-20 anni fa per software business

Si usano per misure di produttività, stime, benchmarking

Storia di FP

- 1979: introdotti da Alan Albrecht
- 1984: primi manuali
- 1986: IFPUG
- 1994: Counting Practices Manual 4.0
- 1999: Counting Practices Manual 4.1
- 2003 Standard ISO/IEC 20926:2003


COSMIC
FFP 2.1
Full Function
Points 1.0
COSMIC
FFP 2.0
Function Point
Analysis
Function Point
Analysis
Function Point
Analysis 3.4
Function Point
Analysis 4.0
Function Point
Analysis 4.1
Function Point
Analysis 4.1.1
IBM 1975
Albrecht
1979
Albrecht
1984
IFPUG
1990
IFPUG
1994
IFPUG
1999
IFPUG
2001
Mark II FPA
1.0
Mark II FPA
1.3.1
Symons 1988
UKSMA
1998
1975
1980
1985
1990
1995
2000

Calcolo di FP

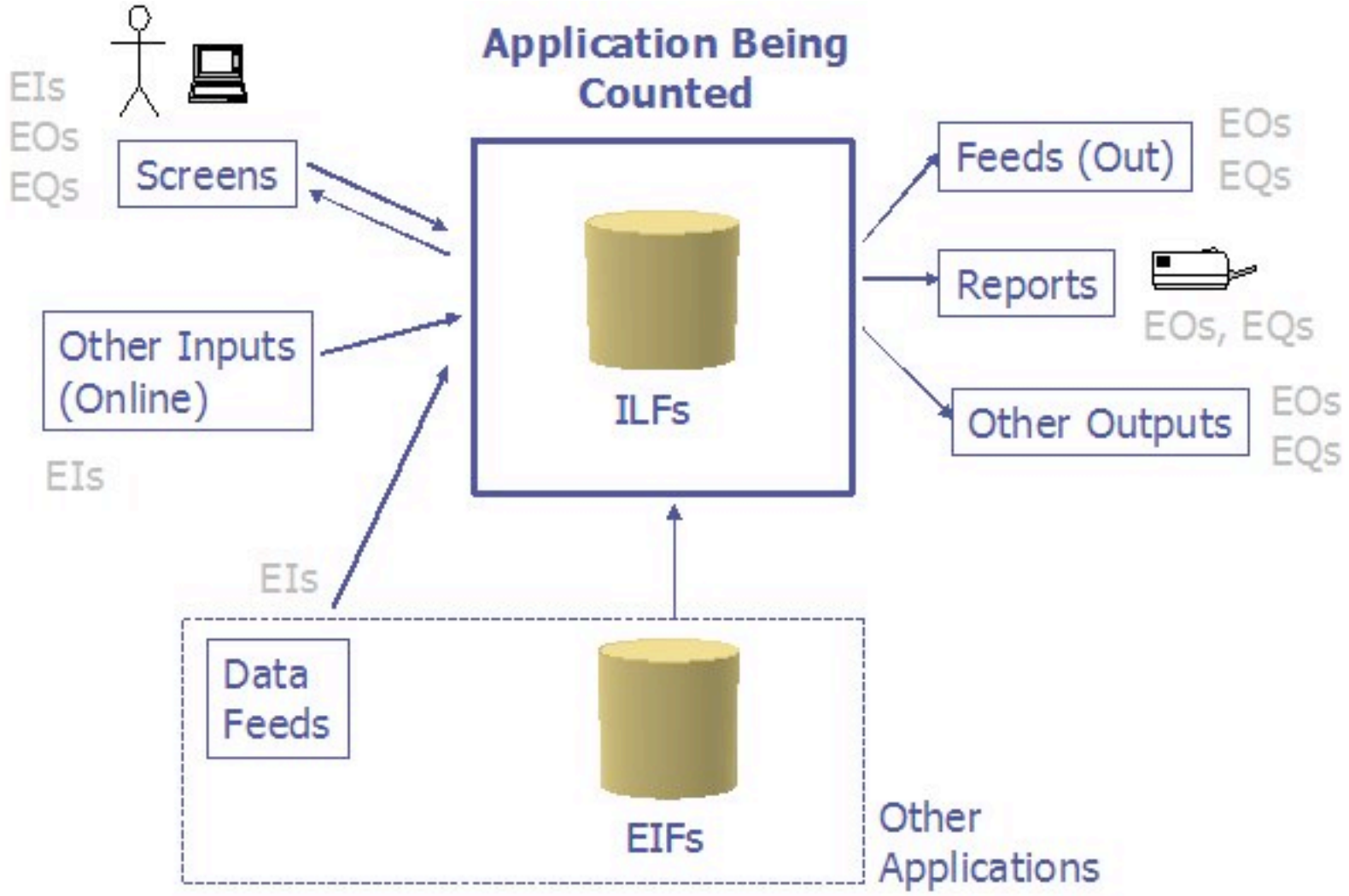
1 Si descrive l'applicazione che si vuole implementare.

Si parte in genere da una descrizione del sistema (specifica) in genere di natura funzionale

Function Point

2. Individuare nella descrizione i seguenti elementi:
   - External Input: informazioni, dati forniti dall'utente (es. nome di file, scelte di menù, campi di input)
   - External Output: informazioni, dati forniti all'utente dell'applicazione (es. report, messaggi d'errore)
   - External Inquiry: sequenze interattive di richieste – risposte
   - External Interface File: interfacce con altri sistemi informativi esterni
   - Internal Logical File: file principali logici gestiti nel sistema

Cosa “conta” chi usa i FP



Function Point

3 Classificare gli elementi individuali in base alla complessità, usando la seguente tabella

	Weighting Factor		
Item	Simple	Average	Complex
External inputs	3	4	6
External outputs	4	5	7
External inquiries	3	4	6
External files	7	10	15
Internal files	5	7	10

Function Point

4. Contare gli elementi di ciascun tipo per ciascuna complessità ed applicare la formula:

UFC= Sum [i=1_5] (Sum [j=1_3] (elemento i,j * peso i,j))

Dove i = tipo elemento

j = complessità (bassa media alta)

UFC = Unajdjusted Function Count

Function Point

5. Moltiplicare UFC per il fattore di complessità tecnica dell'applicazione (TFC)

Calcolo TFC (fattore di complessità tecnica):

$TFC = 0.65+0.01* S_{[i=1-14]} F_i$

F_i assume un valore tra 0 (irrilevante) e 5 (massimo)

F1	Reliable back-up and recovery	F2	Data communications
F3	Distributed functions	F4	Performance
F5	Heavily used configuration	F6	Online data entry
F7	Operational ease	F8	Online update
F9	Complex interface	F10	Complex processing
F11	Reusability	F12	Installation ease
F13	Multiple sites	F14	Facilitate change

Function point

TFC risulta sempre nell'intervallo 0.65 (facile) e 1.35 (difficile)

Alla fine la formula risulta:

FP = UFC * TFC (function point)

Esempio

La descrizione funzionale di un sistema contiene

6	“average” inputs	x	4	=	24
6	“complex” outputs	x	7	=	42
2	“average” files	x	10	=	20
3	“simple” inquiries	x	2	=	6
2	“complex” interfaces	x	10	=	20
	Unadjusted FC			=	112

Esempio (continua)

Data communications	3
Distributed processing	2
On-line processing	4
Complex internal processing	5
Multiple sites	3
Adjustment factor	17

Adjustment computation:

UFC x [.65+(Project.complexity x 0.01)]
= 112 x [.65+(17x0.01)] = 92 Function Points

Esempio (continua)

Se si sa, per esempio a causa di esperienze passate che il numero medio di FP per mese-persona è pari a 18, allora si può fare la stima che segue:

$$92 \text{ F.P.} \div 18 \text{ F.P./mp} = 5.1 \text{ mp}$$

Se lo stipendio medio mensile per lo sviluppatore è di is €6.500, allora il costo [dello sforzo] del progetto è

$$5.1\text{mp} \times €6.500 = €33.150$$

Calibrazione

- Il conteggio degli FP si basa su giudizi soggettivi, quindi persone diverse possono raggiungere risultati diversi

- Quando si introduce la FPA in una organizzazione, è necessaria una fase di **calibrazione**, usando il sw sviluppato in passato come base del sistema di conteggio

Problemi

- I “logical files” sono difficili da definire
- Riconoscimento esplicito dei dati da elaborare
- Pesi insufficienti alla complessità interna
- Validità dei pesi e consistenza della loro applicazione
- La produttività in termini di FP sembra funzione della dimensione del progetto
- Gli FP sono stati estensivamente usati con sistem iinformativi; scarsa esperienza con sistemi client-server o sistemi embedded

Backfiring (LOC -> FP)

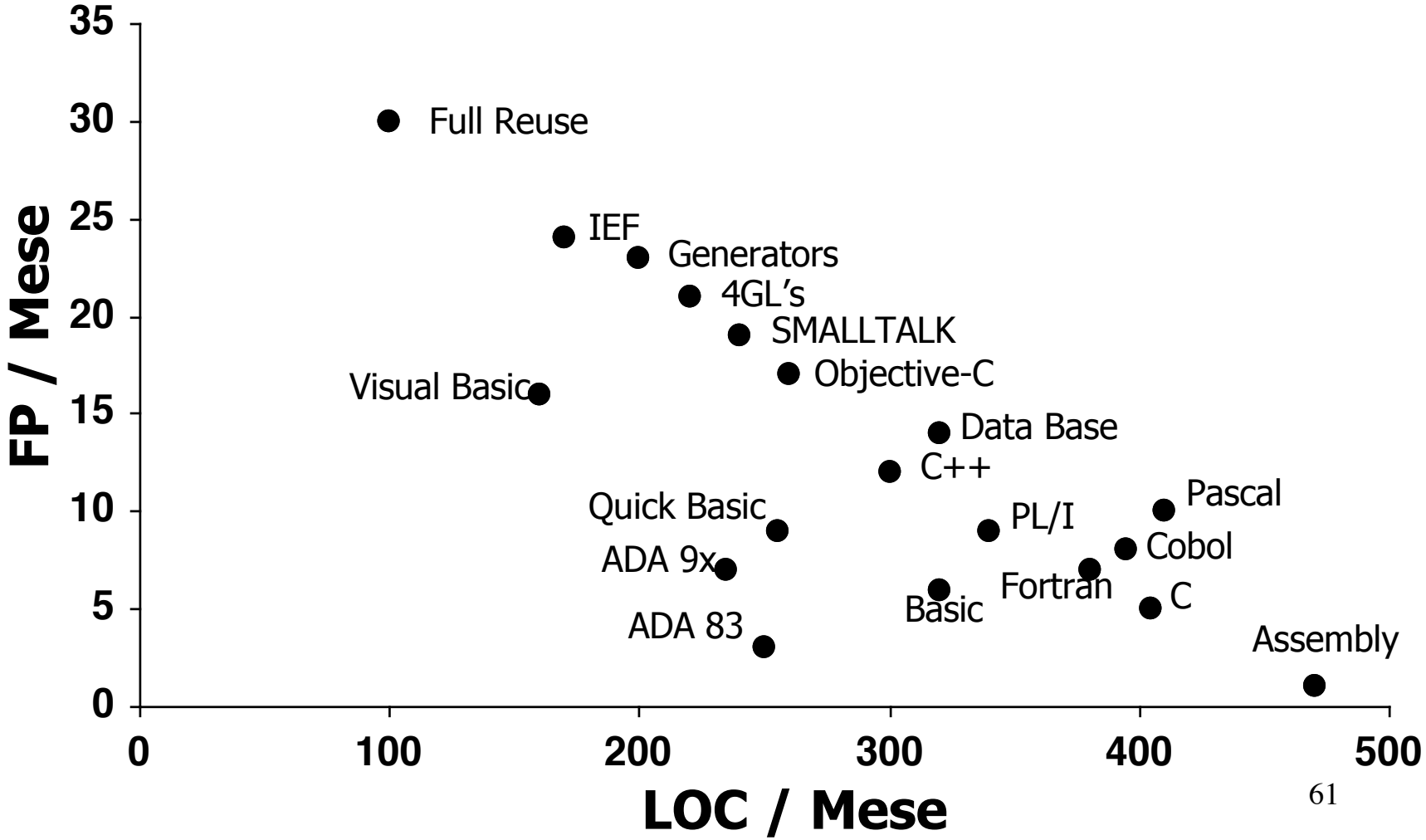
- E’ possibile confrontare LOC e FP usando la tecnica del backfiring, che permette di convertire LOC in FP e viceversa
- Si basa sulla nozione di *livello di astrazione* dei linguaggi di programmazione

LOC vs FP

Linguaggio di programmazione	LOC / FP
Assembler	320
C	128
Cobol / Fortran	105
Pascal	90
Ada	70
C++	55
Java / VisualBasic	35
HTML-3	15

Capers Jones, *Programming Languages Table*, 1996

LOC vs FP (Produttività)



Misure derivate dai FP

- **Produttività: Ore per FP**
  - Misura il numero di ore necessarie in media per sviluppare un FP.
  - Calcolo:
    - Ore totali di progetto / # FP consegnati
- **Produttività generale**
  - E' la produttività complessiva di gruppo di produzione IT
  - Calcolo:
    - FP totali / Sforzo totale del gruppo
- **Funzionalità consegnata & Funzionalità di sviluppo**
  - Funzionalità consegnate all'utente finale in relazione al tasso di produttività raggiunto per tali funzionalità
  - Calcolo:
    - Costo totale / # FP consegnati
    - Sforzo totale / # FP sviluppati internamente

Misure di qualità

- **Dimensione dei requisiti funzionali**
  - Misura il numero totale di funzioni richieste in FP
- **Completezza**
  - Misura la funzionalità consegnata rispetto a quella richiesta inizialmente
  - Calcolo: #FP consegnati / #FP richiesti
- **Produttività di consegna**
  - Produttività necessaria alla consegna entro un certo periodo di durata del progetto
  - Calcolo:
    - # FP totali / Durata dello sforzo

Misure di qualità

- **Efficienza nella rimozione di difetti**
  - # difetti trovati prima della consegna / # difetti totali
- **Densità dei difetti**
  - Numero di difetti identificati in una o più fasi del ciclo di vita in rapporto alla dimensione totale dell'applicazione
    - # difetti / # FP totali
- **Copertura dei casi di test**
  - Numero dei casi di test necessari ad un controllo accurato del progetto
    - # casi di test totali / # FP totali
- **Volume della documentazione**
  - Misura o stima delle pagine prodotte o previste come effetto dello sviluppo
    - # pagine / # FP totali

Misure finanziarie

- **Costo per function point**
  - Il costo medio della realizzazione di ciascun FP
    - Costo totale / # FP totali
- **Costo di una riparazione**
  - Costo di riparazione di applicazioni consegnate ed operative. E' una metrica di monitoraggio di nuove applicazioni
    - (Ore totali di riparazione * Costo_orario) / # FP rilasciate

Misure di manutenzione

- **Mantenibilità**
  - Misura del costo necessario per mantenere un'applicazione
    - Costo di manutenzione / # FP totali dell'applicazione
- **Affidabilità**
  - Numero dei fallimenti in rapporto alla dimensione dell'applicazione
    - # fallimenti / # FP totali dell'applicazione
- **Impegno di manutenzione**
  - Dimensione dell'applicazione rapportata al numero di risorse a tempo pieno necessarie per il suo supporto
    - # FP totali dell'applicazione / # persone a tempo pieno impiegate per la manutenzione

Misure di manutenzione

- **Tasso di accrescimento**
  - Crescita delle funzionalità di un'applicazione in un certo periodo.
    - # FP correnti / # FP originali
- **Dimensione del patrimonio**
  - Misura del patrimonio di FP di un'organizzazione
  - Si usa per scopi di budget in accordi di outsourcing
- **Valore di Backfire**
  - Numero di linee di codice moltiplicate per un fattore di complessità linguistica per derivare il numero totale di FP.
- **Tasso di stabilità**
  - Si usa per monitorare quanto effettivamente un'applicazione ha soddisfatto i suoi utenti (in base al numero di modifiche richieste nei primi 2-3 mesi di produzione oppure operatività)
    - # modifiche / # FP totali dell'applicazione

Stima Basata Su Modelli di Costo

- I modelli di costo permettono una stima rapida dello sforzo
- Questa prima stima è poi raffinata, più avanti nel ciclo di vita, mediante dei fattori (**cost driver**).
  - Il calcolo si basa sulla seguente funzione:
    - $E = A + B*S^C$

    dove E è lo sforzo (in Persone Mese),
    A, B, C sono costanti dipendenti dal progetto e dall'organizzazione che lo esegue,
    S è la dimensione del progetto stimata in LOC o FP.

Problemi dei costi del software

Mentre i costi hardware diminuiscono, i costi del software continuano ad aumentare, almeno in percentuale dei costi totali dei sistemi informativi.

I problemi di stima dei costi sw sono causati da:

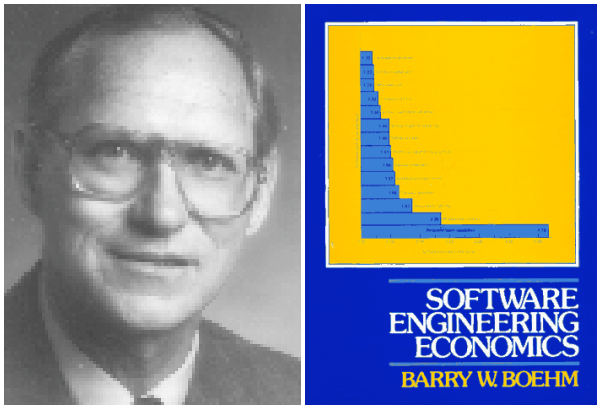
- incapacità di dimensionare accuratamente un progetto sw;
- incapacità di definire nel suo complesso il processo e l'ambiente operativo di un progetto;
- valutazione inadeguata del personale, in quantità e qualità;
- mancanza di requisiti di qualità per la stima di attività specifiche nell'ambito del progetto

Modelli dei costi del software

Esempi di modelli commerciali

- **COCOMO**
- COSTXPERT
- SLIM
- SEER
- Costar, REVIC, etc.

Modelli di costi sw: COCOMO



- Il **Constructive Cost Model** (**COCOMO**) è uno dei modelli più diffusi per fare stime nei progetti software
- COCOMO è descritto nel libro *Software Engineering Economics* di Barry Boehm, 1981
- COCOMO è un modello basato su regressione che considera vari parametri storici di progetto, pesati mediante una griglia di valutazione

Modelli di costi sw: COCOMO

- Il principale calcolo di COCOMO si basa sull'equazione dello Sforzo (Effort) per stimare il numero di mesi-persona necessari per un progetto

  # mp * costo_lavoro = Costo_Stimato

- La maggior parte delle altre grandezze stimate (per i requisiti, per la manutenzione, ecc.) vengono poi derivate da questa equazione

*Co*nstructive *Co*st *Mo*del

- Boehm costruì la prima versione di un modello di costo chiamato CoCoMo 1 nel 1981
- CoCoMo 1 è una collezione di tre modelli:
  - **Basic** (applicato all'inizio del ciclo di vita del progetto)
  - **Intermediate** (applicato dopo la specifica dei requisiti)
  - **Advance** (applicato al termine della fase di design)

CoCoMo

- I tre modelli hanno forma equazionale:

  Effort = $a*S^{b}*EAF$

  – Effort è lo sforzo in mesi-persona

  – EAF è il *coefficiente di assestamento*

  – S è la dimensione stimata del codice sorgente da consegnare, contata in migliaia di linee di codice (KLOC)

  – a e b sono dei coefficienti che dipendono dal tipo di progetto.

Tipi di Progetti

- **Organic mode** (progetto semplice, sviluppato in un piccolo team)
- **Semidetached mode** (progetto intermedio)
- **Embedded** (requisiti molto vincolanti e in campi non ben conosciuti)

Formule Per il Modello Base

Tipo Fase basic	A	B	EAF	FORMULA RISULTANTE
Organic	2.4	1.05	1	E = 2.4*S^1.05*1
Semi detached	3.0	1.12	1	E = 3.0*S^1.12*1
Embedded	3.6	1.20	1	E = 3.6 * S^1.20 * 1

Un esempio

Dimensione = 200 KLOC
Sforzo(in mp) = a * Dimensione ^{b}

Organic: Sforzo = 2.4*(200 $^{1.05}$) = 626 mesipersona
Semidetached: Sforzo = 3.0*(200 $^{1.12}$) = 1133 mp
Embedded: Sforzo = 3.6*(200 $^{1.20}$) = 2077 mp

Modello Intermedio

- Prende il precedente come base
- E' quello più usato
- Identifica un insieme di attributi che influenzano il costo (detti *cost driver*)
- Moltiplica il costo di base per un fattore che lo può accrescere o decrescere
- I valori reali rimangono entro il 20% dei valori stimati circa il 68% delle volte

Cost drivers

- Del personale
  - capacità ed esperienza degli analisti e dei programmatori
  - conoscenza del linguaggio di programmazione utilizzato
- Del prodotto
  - affidabilità richiesta
  - dimensione del DB
  - complessità del prodotto

Cost drivers

- Del computer
  - Tempo di esecuzione del programma
  - Limitazioni di memoria
- Del progetto
  - Tools software disponibili
  - Tempo di sviluppo del programma

Cost drivers

Cost Drivers	Rating					
	Very Low	Low	Nominal	High	Very High	Extra High
Product attributes						
Required software reliablity	0.75	0.88	1.00	1.15	1.40	
Database size		0.94	1.00	1.08	1.16	
Product complexity	0.70	0.85	1.00	1.15	1.30	1.65
Computer attributes						
Execution time constraint			1.00	1.11	1.30	1.66
Main storage constraint			1.00	1.06	1.21	1.56
Virtual machine volatility*		0.87	1.00	1.15	1.30	
Computer turnaround time		0.87	1.00	1.07	1.15	
Personnel attributes						
Analyst capabilities	1.46	1.19	1.00	0.86	0.71	
Applications experience	1.29	1.13	1.00	0.91	0.82	
Programmer capability	1.42	1.17	1.00	0.86	0.70	
Virtual machine experience*	1.21	1.10	1.00	0.90		
Programming language experience	1.14	1.07	1.00	0.95		
Project attributes						
Use of modern programming practices	1.24	1.10	1.00	0.91	0.82	
Use of software tools	1.24	1.10	1.00	0.91	0.83	
Required development schedule	1.23	1.08	1.00	1.04	1.10	

*For a given software product, the underlying virtual machine is the complex of hardware and software (operating system, database management system) it calls on to accomplish its task.

Calcolo del fattore moltiplicativo

- I fattori moltiplicativi dovuti agli attributi hanno un valore che indica lo spostamento dal valore normale di quel determinato attributo
- E' determinato tenendo conto dei progetti passati
- Il prodotto della valutazione degli attributi rilevanti forma EAF

Esempio

Dimensione = 200 KLOC

Sforzo = a * Dimensione ^{b} * EAF

Cost drivers:

Low reliability	0.88
High product complexity	1.15
Low application experience	1.13
High programming language experience	0.95

EAF= 0.88 * 1.115 * 1.13* 0.95 = 1.086

Organic: Sforzo = 3.2*(200 $^{1.05}$) * 1.086= 906 mp

Semidetached: Sforzo = 3.0*(200 $^{1.12}$) * 1.086 = 1231 mp

Embedded: Sforzo = 3.6*(200 $^{1.20}$) * 1.086 = 2256 mp

Esempio 2

- Sw di comunicazione per trasferimento fondi (e.g. BANCOMAT)
- Embedded mode
- 10 KLOC

Cost Drivers	Situation	Rating	Effort Multiplier
Required software reliability	Serious financial consequences of software fault	High	1.15
Database size	20,000 bytes	Low	0.94
Product complexity	Communications processing	Very high	1.30
Execution time constraint	Will use 70% of available time	High	1.11
Main storage constraint	45K of 64K store (70%)	High	1.06
Virtrual machine volatility	Based on commercial microprocessor hardware	Nominal	1.00
Computer turnaround time	Two hour average turnaround time	Nominal	1.00
Analyst capabilities	Good senior analysts	High	0.86
Applications experience	Three years	Nominal	1.00
Programmer capability	Good senior programmers	High	0.86
Virtual machines experience	Six months	Low	1.10
Programming language experience	Twelve months	Nominal	1.00
Use of modern programming practices	Most techniques in use over one year	High	0.91
Use of software tools	At basic minicomputer tool level	Low	1.10
Required development schedule	Nine months	Nominal	1.00

Esempio 2

- Effort = 2.8 ´ $(10)^{1.20}$ = 44 pm
- Moltiplicatori di effort = 1.35
- Sforzo stimato reale 44 x 1.35 = 59 pm
- Questo indice (59 pm) si può usare in altre formule per calcolare o stimare
  - Costo in Euro
  - Piani di sviluppo
  - Distribuzioni di attività
  - Costi dell'hw
  - Costi di manutenzione annui
  - Ecc.

Modello Avanzato

- Questo metodo aggiunge una serie di cost driver **pesati** per **ciascuna** fase del ciclo di vita
- Le fasi del ciclo di vita usate in dettaglio da CoCoMo sono:
  - Requisiti e progettazione di alto livello
  - Progetto dettagliato
  - Codice e test
  - Integrazioni e test

Modello Avanzato

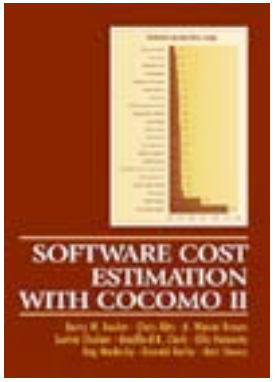
- Si usa il cost driver associato alla fase del ciclo di vita
- Si moltiplica questo al fattore moltiplicativo (EAF')
- $E = a*S^b*EAF'$

Modello avanzato

- RPD: Analisi dei requisiti e progettazione
- DD: progetto dettagliato
- CUT: codice e test
- IT : unificazione e test

Cost Driver	Rating	RPD	DD	CUT	IT
ACAP	Very Low	1.80	1.35	1.35	1.50
	Low	0.85	0.85	0.85	1.20
	Nominal	1.00	1.00	1.00	1.00
	High	0.75	0.90	0.90	0.85
	Very High	0.55	0.75	0.75	0.70

CoCoMo 2



- CoCoMo 2 è un rinnovato (2000) metodo di stima dei costi sw che supporta:
  - Software orientati agli oggetti.
  - Software creati con ciclo di vita a spirale o con modelli di processo evolutivi.
  - Software per e-commerce.

CoCoMo 2

- Il metodo COCOMO2 non usa i LOC come misura di dimensione del software, ma i FP.
- I FP vengono trasformati in LOC mediante la tabella di C. Jones.

Riferimenti

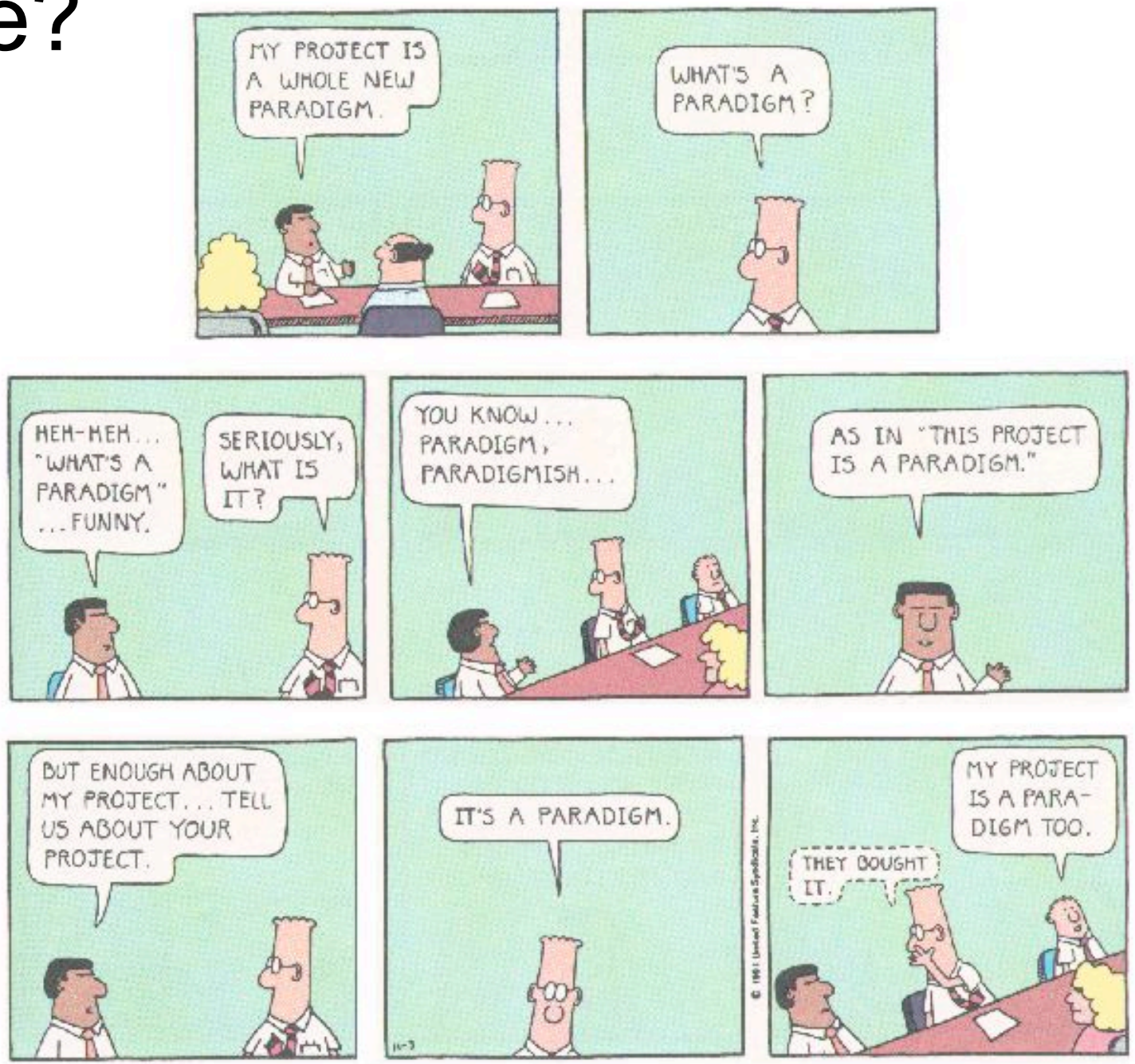
- IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans
- Chen, Boehm, Madachy e Valerdi, An Empirical Study of eServices Product UML Sizing Metrics, 2004

Siti

- www.pmi.org
- www.ifpug.org

- www.dwheeler.com/sloc/
- www.devdaily.com/FunctionPoints/FunctionPoints.shtml
- www.cosmic.com
- sunset.usc.edu/research/COCOMOII

Domande?


MY PROJECT IS A WHOLE NEW PARADIGM.
WHAT'S A PARADIGM?
HEH-HEH... "WHAT'S A PARADIGM" ...FUNNY.
SERIOUSLY, WHAT IS IT?
YOU KNOW... PARADIGM, PARADIGMISH...
AS IN "THIS PROJECT IS A PARADIGM."
BUT ENOUGH ABOUT MY PROJECT... TELL US ABOUT YOUR PROJECT.
IT'S A PARADIGM.
MY PROJECT IS A PARA-DIGM TOO.
THEY BOUGHT IT.