

## Progettazione e sviluppo del software e dei sistemi informativi



## Sommario

- Il ciclo di vita del software
- Strutturazione dei progetti software
- Modelli descrittivi per la progettazione di sistemi informativi
- Pianificazione, gestione e controllo nei progetti di sviluppo dei sistemi informativi
- Scelta e integrazione del software standard (o package applicativi)
- La qualità del software e dei sistemi



2

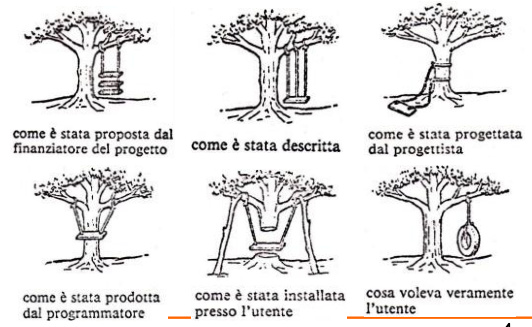
## Ingegneria del software

- Settore della disciplina dei sistemi informativi, dedicato allo studio delle metodologie, delle tecniche e degli strumenti utilizzati nella produzione industriale del software visto come processo di collaborazione tra analisti, programmatori e utenti finali
- Affronta le problematiche di tipo manageriale, organizzativo e metodologico per permettere che il lavoro di analisti e progettisti possa essere condotto con la maggiore efficacia, avvalendosi di tecniche e modi di procedere sperimentati in contesti eterogenei



3

## Perché Ingegneria del Software ? Un esempio di idea di progetto...



4

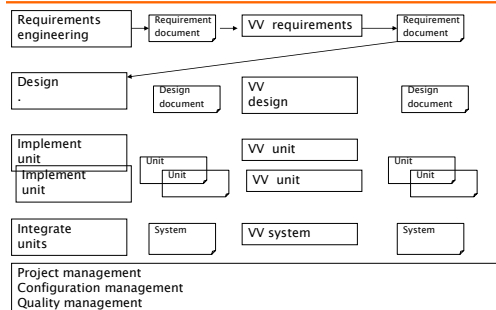
## Software life cycle

- Insieme delle fasi che si susseguono, dal momento in cui il software viene concepito, progettato, realizzato, alla sua messa in opera e manutenzione, sino alla sua dismissione
- Un progetto di sviluppo software segue sempre un modello di ciclo di vita



5

## Fasi in Progetto Software



6

## Fase di pianificazione (o studio di fattibilità)

- Si stabiliscono gli obiettivi del sistema informativo da sviluppare
- In un primo studio preliminare si analizza la fattibilità del progetto sotto il profilo tecnico (possibilità di utilizzo delle risorse esistenti) ed economico (stima costi / benefici)

SoftEng

7

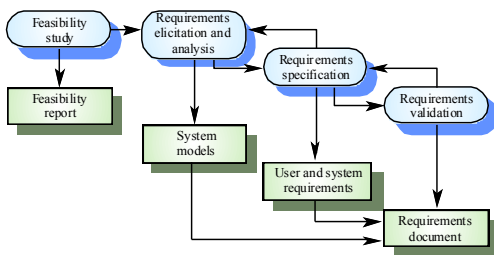
## Fase di analisi dei requisiti

- Serve a individuare le aspettative dell'utente finale in relazione al prodotto da realizzare attraverso la cosiddetta **analisi dei requisiti**.
- Spesso è utile eseguire a priori un'analisi dei processi aziendali.
- Sulla scorta di tali ricerche e di un'eventuale analisi delle aree di criticità si elabora un progetto di massima del software.

SoftEng

8

## Processo di Ingegneria dei requisiti



SoftEng

9

## Fase di analisi: aspetti funzionali

Sono orientati a determinare:

- l'area funzionale che il nuovo software/sistema deve supportare
- le modalità con cui il sistema informativo deve eseguire le funzionalità per cui è predisposto
- i modelli di organizzazione dei dati cui le diverse procedure dovranno avere accesso
- gli input e gli output del sistema

SoftEng

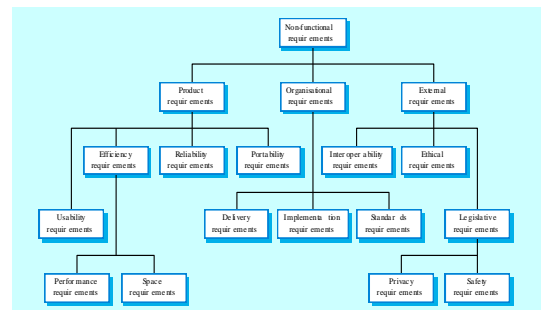
## Fase di analisi: aspetti qualitativi

Sono orientati a determinare i cosiddetti requisiti NON FUNZIONALI:

- la configurazione dell'interfaccia utente
- le aspettative relative ai tempi di risposta
- l'affidabilità del sistema

SoftEng

## Requisiti Non-Funzionali



## Fase di analisi: aspetti economici

---

Sono orientati a determinare:

- i costi di esercizio
- i costi di manutenzione
- i benefici (risparmi) che si potranno eventualmente ottenere

SoftEng

---

## Fase di analisi: specifica dei requisiti

---

- Documento che sintetizza i problemi e le esigenze degli utenti in modo chiaro e univoco
- Contiene le funzionalità che il sistema dovrà avere, le prestazioni, l'ambiente di utilizzo, le interfacce esterne (con utenti, altro software, hardware), gli eventuali vincoli di progetto (tempi, risorse ecc.), i requisiti di qualità

SoftEng

---

## Fase di analisi: specifiche del processo di sviluppo

---

Si focalizza l'attenzione su:

- aspetti funzionali (necessità di collaborazione tra le singole aree operative o studio degli impatti organizzativi derivanti dall'introduzione del nuovo sistema)
- aspetti qualitativi (documentazione del programma, test da realizzare sul software)
- aspetti economici (costi di sviluppo, durata di quest'ultimo, risorse necessarie, risorse disponibili, stima dei possibili benefici)

SoftEng

---

## Fase di progettazione

---

- Ha l'obiettivo primario di individuare le funzioni che costituiscono un processo, le loro relazioni e i dati necessari alla loro realizzazione
- Inoltre, si studiano le modalità di produzione, utilizzazione, aggiornamento, cancellazione e scambio di dati rilevanti nell'ambito delle singole funzioni
- Due possibili approcci: **progettazione tradizionale (strutturata)** e **progettazione object oriented**

SoftEng

---

## Approcci alla progettazione

---

- **Progettazione tradizionale (strutturata):** la realtà aziendale o il singolo processo vengono analizzati attraverso il modello dei dati e quello delle funzioni
- **Progettazione object oriented:** dati e funzioni della realtà aziendale vengono riuniti in un unico modello oggetto
- In entrambi gli approcci si parte dalle specifiche di programma per realizzare il **progetto logico** e il **progetto fisico**

SoftEng

---

## Progetto logico (Architettura)

---

- Vengono identificate le componenti (**moduli**) del sistema e le connessioni fra esse
- Un modulo è una componente dedicata a svolgere una specifica funzione
- Ogni modulo è costituito da un'interfaccia (la parte visibile dall'esterno) e dalla logica applicativa (la parte "interna" del modulo)
- Un sistema è, quindi, composto da vari moduli che interagiscono fra loro
- Il prodotto del progetto logico è una descrizione dettagliata dei compiti che ogni modulo deve svolgere (cosa) e del modo in cui i vari moduli comunicano fra di loro
- Nulla viene detto sul come i vari moduli svolgano i loro compiti

SoftEng

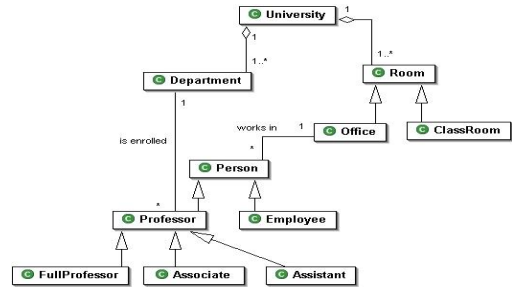
---

## Design

- Nelle fasi di progettazione si utilizzano strumenti di modellazione delle componenti software che saranno realizzate
- Linguaggio Standard UML (Unified Modeling Language)
  - Class Diagram
  - Sequence Diagram
  - State Charts
  - Activity Diagram
  - Use Case Diagram

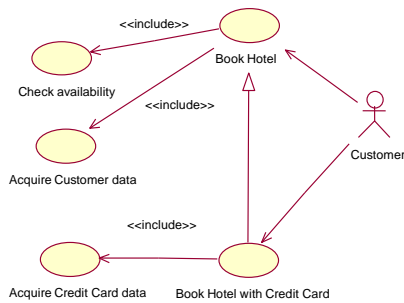
SoftEng

## Esempio Class Diagram



SoftEng

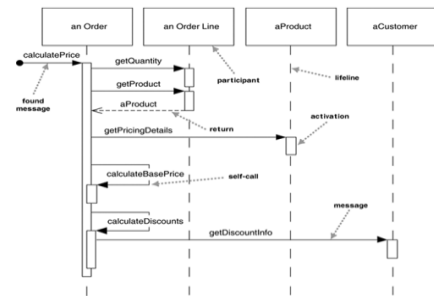
## Esempio Use-Case Diagram



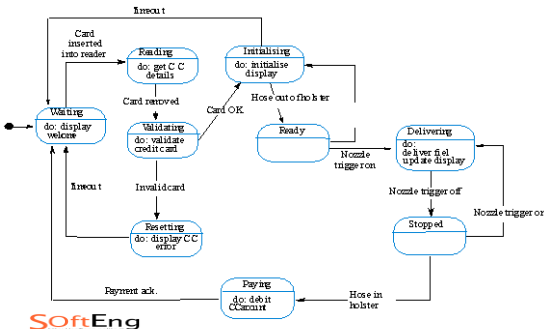
SoftEng

## Esempio Sequence Diagram

Figure 4.1. A sequence diagram for centralized control

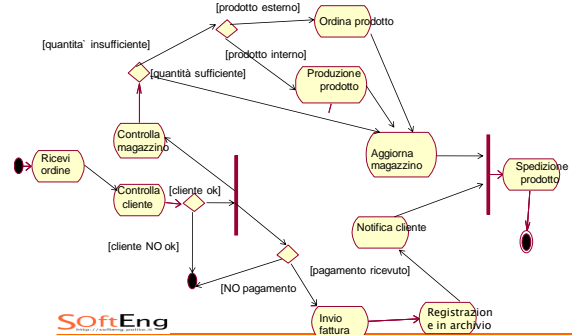


## Esempio State-chart



SoftEng

## Esempio Activity Diagram



SoftEng

## Progetto fisico

---

- È orientato a definire
  - Le caratteristiche dell'ambiente hardware e del software del nuovo sistema
  - La struttura generale (componenti) del sistema informativo
  - I moduli di programma con i quali vengono eseguite le varie procedure aziendali
  - La sequenza con la quale i singoli moduli di programma dovranno essere elaborati
  - La struttura logica dei dati dell'applicazione
  - La struttura fisica dei dati e dei file di dati
  - I primi test di prova

SoftEng

---

## Fase di implementazione (programmazione)

---

- Serve a specificare nei minimi dettagli il progetto del sistema fino ai singoli comandi nel linguaggio di programmazione prescelto e in particolare
  - schemi di dati (descrizione della struttura dei dati, dei file o dei database)
  - cicli di programma o funzioni rappresentati da elementi strutturali del programma sotto forma di un diagramma strutturale
  - interfacce utente

SoftEng

---

## CASE

---

- *Computer Aided Software Engineering*, strumenti che coadiuvano la programmazione con compilatori, interpreti, linker, editor, generatori di maschere, generatori di codici
- Forniscono ad analisti e programmatori supporti per controllare e gestire, da un punto di vista tecnico e organizzativo, la produzione del software

SoftEng

---

## Prototyping

---

- Si propone di sviluppare un progetto informatico creando, il più rapidamente possibile, una versione eseguibile del sistema informativo
- Non si esegue un'analisi dettagliata del progetto
- Nello sviluppo viene coinvolto il più possibile l'utente finale
- Si mira a eliminare le difficoltà di comunicazione tra specialisti informatici e utenti aziendali

SoftEng

---

## Critiche al prototyping

---

- Vengono trascurate esigenze di strutturazione sotto il profilo ingegneristico
- Poiché spesso alla prototipazione segue lo sviluppo a cascata, i costi complessivi dello sviluppo risultano elevati

SoftEng

---

## System test

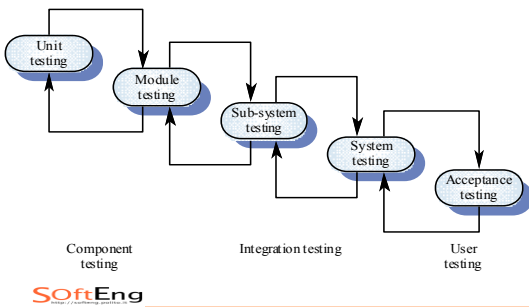
---

- Passo del processo di implementazione mediante il quale vengono verificati l'intera applicazione e i singoli sottosistemi che la compongono.

SoftEng

---

## Processo di testing



31

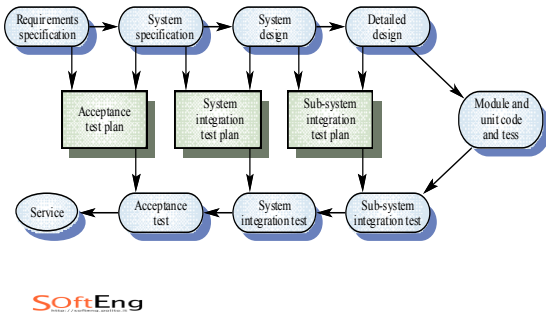
## Livelli di Testing

- Unit testing
  - Test di singoli moduli (procedure, classi, componenti)
- Integration testing
  - Test di vari moduli via via integrati
- System testing
  - Test del sistema completo. Test di funzionalita' e proprieta' non funzionali
- Acceptance testing
  - Come sopra, eseguito dal cliente, su dati/piattaforma del cliente

SoftEng

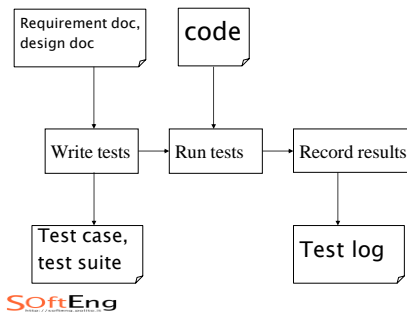
32

## Testing

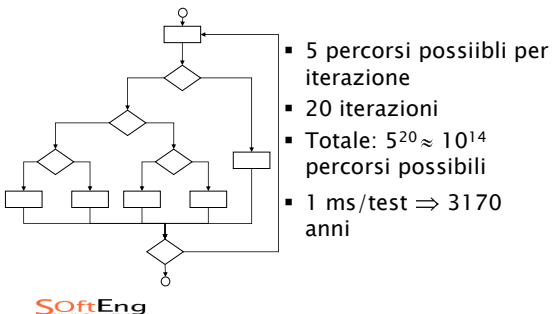


33

## Testing



## Test Esaustivo...

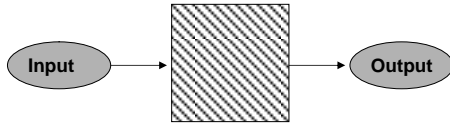


## Test Esaustivo...

- Test Esaustivo non è fattibile
- Obiettivo è trovare difetti, non dimostrare che sistema ne è privo...
- Obiettivo è garantire un accettabile livello di confidenza nel software

SoftEng

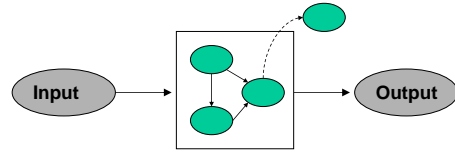
## Black Box testing



- Vede software come interfaccia di cui bisogna capire il comportamento
- Si può agire classificando vari tipi di input
- Serve specifica per test mirati

SoftEng

## White Box testing

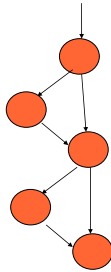


- Serve conoscenza del codice sorgente
- Permette di testare una singola funzione in modo molto preciso

SoftEng

## Control Flow Graph

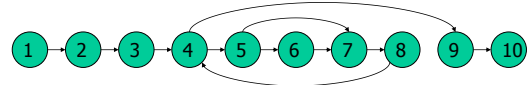
- Ogni porzione di codice può essere rappresentata con un grafo in cui ogni nodo è un'istruzione



SoftEng

## Esempio codice -> CFG

```
1 float homeworkAverage(float[] scores) {
2   float min = 99999;
3   float total = 0;
4   for (int i = 0; i < scores.length; i++){
5     if (scores[i] < min)
6       min = scores[i];
7     total += scores[i];
8   }
9   total = total - min;
10  return total / (scores.length - 1);
}
```



SoftEng

## Node coverage

- Scelgo un insieme di testcase in modo da visitare almeno una volta ogni nodo del grafo CFG
  - Cioè ogni istruzione è eseguita almeno una volta
- Scelgo il minimo numero di test-case per soddisfare node coverage

SoftEng

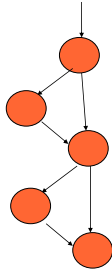
## Branch coverage

- Scelgo un insieme di testcase in modo da visitare almeno una volta ogni arco del grafo CFG
  - Cioè ogni istruzione è eseguita almeno una volta
- Ottenere Branch coverage implica ottenere Node coverage
- Scelgo il minimo numero di test-case per soddisfare branch coverage

SoftEng

## Path coverage

- Scelgo test-case in modo che ogni percorso (path) del grafo sia eseguito almeno una volta
- Es: 4 percorsi in grafo -->
- In caso di iterazioni si pone un limite max N di cicli
  - e si parla di path-N coverage



SoftEng

## Fase di collaudo e installazione

- Dopo aver verificato se il programma soddisfa tutti i requisiti tecnico/funzionali
- Il collaudo può dimostrare con certezza la presenza di errori, non l'assenza (a meno di fare tutte le prove possibili, approccio economicamente insostenibile)
- Viene, inoltre, intensificato l'addestramento degli utenti finali

SoftEng

44

## Fase di manutenzione

- Normalmente si estende per tutta la vita del sistema
- Vengono apportate modifiche e adattamenti e si provvede a eliminare errori non rilevati nel test o nel collaudo di sistema
- Spesso le modifiche sono dettate da cambiamenti nell'esigenze dell'utente, da aggiornamenti legislativi o da variazioni nell'architettura del sistema
- A essa può essere imputato oltre il 50% delle spese affrontate per l'intero ciclo di vita del software (TCO, *Total Cost of Ownership*)

SoftEng

45

## Processi di Sviluppo Software Development Lifecycle



## Lifecycles

- Gli standard di processo elencano processi e attività senza specificarne l'ordine
- I cicli vita definiscono
  - un ordine per processi e attività (o fasi)
  - criteri per passare da una fase ad un'altra
- A livello aziendale / di progetto occorre specializzare

SoftEng

47

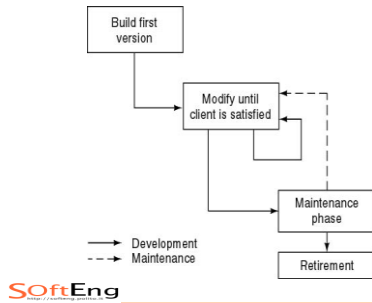
## Sw Development Lifecycles

- Build and Fix: un non-modello
  - Attività non identificate né organizzate
  - Progetti non gestiti
- Modelli organizzati
  - Cascata: fasi sequenziali
  - Incrementali: realizzazione in più passi
  - Evolutivi: modelli completamente ciclici
    - Prototyping
  - Spirale
  - Metodi formali
  - Riuso
  - Unified Process
  - Extreme Programming

SoftEng

48

## Build and fix (code and go)



SoftEng

49

## Build and fix

- No requisiti, no design
- Non scala a progetti grandi
- Verifiche su requisiti e design non fattibili se non a posteriori

SoftEng

50

## Cascata (waterfall)

- Definito nel 1970 da Royce
- Successione di fasi sequenziali
  - Impossibilità di ritornare a fasi precedenti
  - In caso di eventi eccezionali il processo riparte
- Documentazione
  - Ogni fase produce “documenti” che concretizzano la fase
  - I documenti sono necessari per la fase successiva
  - Modello “document driven”

SoftEng

51

## Modello a cascata

- Il processo di sviluppo di un sistema informativo o di un suo sottosistema (procedura o classi di procedure) è suddiviso in una sequenza di fasi
- Ogni fase deve essere terminata prima di passare a quella successiva (non si ritorna indietro) e l'output da essa generato andrà a costituire l'input della fase seguente
- È possibile effettuare controlli di qualità sui singoli risultati parziali

SoftEng

52

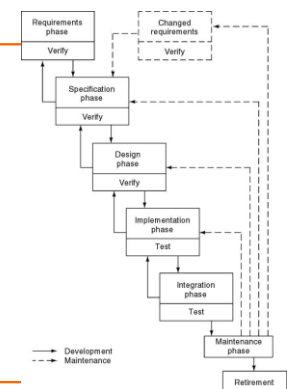
## Critiche al modello di sviluppo a cascata

- È un modello rigido che si fonda su due assunti discutibili
  1. Gli utenti sono in grado di esprimere esattamente le loro esigenze e, di conseguenza, è possibile definire in fase di analisi iniziale tutte le funzionalità che il software deve realizzare (immutabilità dell'analisi)
  2. È possibile progettare l'intero sistema prima di aver scritto una sola riga di codice (immutabilità del progetto)

SoftEng

53

## Waterfall



SoftEng

54

## Caratteristiche delle fasi

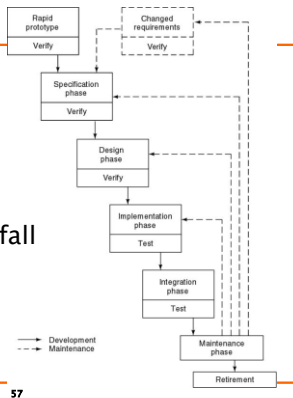
- Le fasi sono descritte in termini di:
  - Contenuti e struttura dei documenti
  - Responsabilità e ruoli coinvolti
  - Scadenze di consegna dei documenti
- Dipendenze causali e temporali
- Riferimento per l'identificazione delle attività

## Problemi

- Mancanza di flessibilità
  - Rigorosa sequenzialità delle fasi
  - Non prevede cambiamenti nei requisiti
    - Requisiti raramente chiari
    - Requisiti non si riescono a congelare per il tempo richiesto a implementare (mesi, anni)
  - Genera molta manutenzione
  - Burocratico e poco realistico

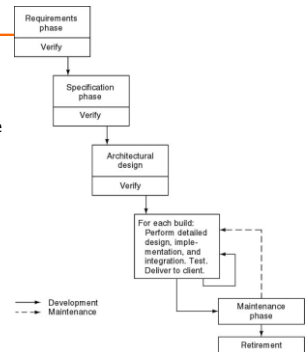
## Prototyping + waterfall

- Prototipo quick and dirty per validare /esplorare i requisiti
- in seguito waterfall



## Incrementale

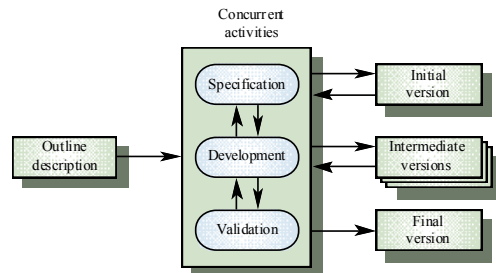
- Decomporre la realizzazione del sistema in incrementi
  - Ritardare la realizzazione delle componenti che dipendono criticamente da fattori esterni (tecnologie, hardware sperimentale, ecc)
- Le iterazioni sono pianificate
- Realizzare incremento, integrare



## Evolutivo (Evolutionary)

- I requisiti non vengono fissati, ma possono cambiare.
- In genere associato a iterazioni
- Si può sviluppare prototipo usa-e-getta quando i requisiti sono molto poco chiari

## Evolutivo



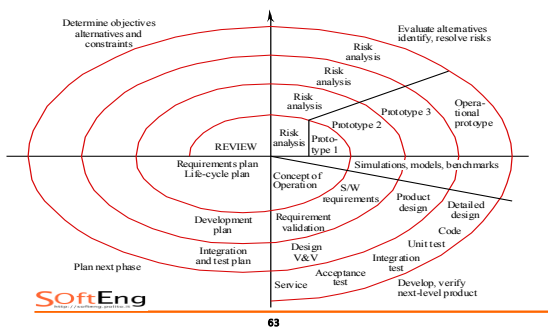
## Evolutivo

- Problemi
  - Poca visibilita' sul processo
  - Design di alto livello
  - Richieste competenze specifiche (ad es. Linguaggi di rapid prototyping)
  - Prototyping, tendenza a non gettare il prototipo
  - Puo' porre problemi contrattuali
    - Agreement su effort e non su funzioni / turn key system

## A spirale

- Processo come spirale invece che sequenza con backtracking
- Ogni loop e' una fase
- Non ci sono fasi predefinite come specifica o design, si sceglie il loop che serve
- Enfasi su raccolta e analisi dei rischi

## A spirale



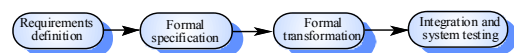
## Quattro quadranti

- Obiettivi
  - Definire gli obiettivi per la fase
- Rischi
  - Identificazione dei rischi per la fase
  - Analisi dei rischi (effetto e gravita)
  - Mitigazione, definizione di attivita' per ridurre o eliminare rischi
- Sviluppo e validazione
  - Sviluppo secondo uno qualsiasi dei cicli vita visti
- Planning
  - Post mortem di fase precedente, planning di fase seguente

## Processo basato su metodi formali

- Basato su trasformazione successiva di una specifica matematica in programma eseguibile
- I formalismi matematici permettono di eseguire prove di correttezza (model checking)
- Le trasformazioni mantengono la correttezza
  - Verification e' immediata

## Metodi formali



## Processo basato su metodi formali

- Problemi
  - Formalismi richiedono know how spinto e raramente disponibile
  - Alcune parti del sistema (es user interface) non si riescono a specificare
  - In ogni caso il customer raramente comprende la specifica, quindi la validazione dei requisiti è difficile
- Applicabilità
  - Parti specifiche, critiche (safety, security) e piccole di sistemi.

SoftEng

67

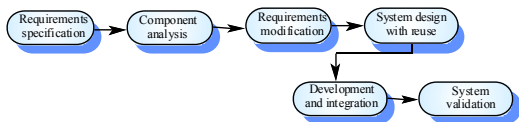
## Processo basato sul riuso

- Sviluppo basato su integrazione di componenti preesistenti:
  - COTS (Commercial-off-the-shelf)
  - oppure OSS (Open Source Software)
- Fasi
  - Analisi e ricerca dei componenti
  - Definizione/adattamento dei requisiti
    - Compromesso desideri – componenti disponibili
  - Design di sistema
  - Sviluppo e integrazione
- Approccio sempre piu' seguito ma ancora poco formalizzato

SoftEng

68

## Processo basato sul riuso



SoftEng

69

## XP

- Definire con il cliente 'storie', o funzionalita' richieste
- Scegliere una o piu' storie
  - realizzabili in settimane
- Dividere storie in tasks (= build)
- Scrivere test cases per task
- Pair programming
- Design for now, refactoring

SoftEng

70

## Synch and stabilize (MS)

- Definire i requisiti, prioritizzare
- Dividere progetto in 3-4 build
- Ogni build un team, vari team in parallelo
- Per una build / team
  - ogni sera: synchronize (test case devono girare tutti)
  - a fine build: stabilize (build stabile puo' essere rilasciata)

SoftEng

71

## Unified Process

- Proposto nel 1999 da
  - Grady Booch
  - Ivar Jacobson
  - James Roumbaugh
- Caratteristiche
  - Guidato dai casi d'uso e dall'analisi dei rischi
  - Incentrato sull'architettura
  - Iterativo incrementale

SoftEng

72

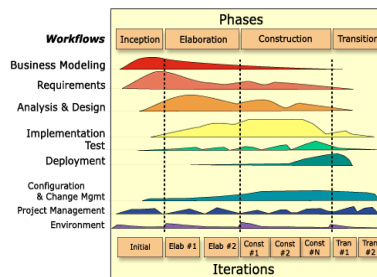
## Fasi di UP

- **Avvio**
  - Fattibilità; Analisi dei rischi; Requisiti essenziali per definire il contesto del sistema; Eventuale prototipo
- **Elaborazione**
  - Analisi dei requisiti; Analisi dei rischi; Sviluppo di un'architettura base; Piano per la fase di costruzione
- **Costruzione**
  - analisi, disegno, implementazione, testing
- **Transizione**
  - Beta testing, aggiustamento delle prestazioni, creazione di documentazione aggiuntiva, attività di formazione, guide utenti, creazione di un kit per la vendita

SoftEng

73

## UP



SoftEng

74

## Project Management



## Progetto

- Insieme di attività finalizzate al raggiungimento di un determinato obiettivo attraverso l'impiego di risorse umane, materiali, tecnologiche, temporali e finanziarie, nel rispetto di prefissati vincoli in termini di tempi, costi e qualità

SoftEng

## Project management

- Insieme di tutte le attività tecniche, organizzative e gestionali connesse con la realizzazione di qualunque tipologia di progetto
- **Project manager:** figura dotata di esperienza e background culturale tecnico e organizzativo che opera con ruoli di responsabilità nella realizzazione di un progetto

SoftEng

## Fasi della pianificazione di un progetto

- Identificazione delle aree aziendali coinvolte nel progetto
- Scelta dell'ambiente di sviluppo del software
- Definizione della sequenza di attività per lo sviluppo della procedura automatizzata
- Coordinamento delle risorse
- Determinazione della responsabilità del personale coinvolto e nella sua motivazione
- Individuazione di scadenze che consentano di verificare il progetto di sviluppo del software in termini di risultati finali e parziali raggiunti, nonché di rispetto dei tempi e dei costi previsti

SoftEng

## Struttura analitica di progetto

- Definisce una corretta interrelazione fra tutti le componenti (attività) elementari del progetto, per soddisfare la necessità di disporre di un procedimento ordinato e sistematico nell'avanzamento dei lavori
- L'analisi di tale struttura deve essere condotta con tutti gli attori coinvolti nella realizzazione del progetto, per raggiungere la piena condivisione della sua validità

SoftEng

## Piano strutturale del progetto

- Consente di evidenziare gli elementi oggetto di consegna al committente (*deliverable*) e i principali compiti funzionali da eseguire per la realizzazione di ciascuno di essi

SoftEng

## Work Breakdown Structure

- Procedimento sistematico e organizzato mediante il quale il progetto viene articolato in componenti elementari detti **workpackage**, che devono essere opportunamente pianificati, valutati, programmati e controllati anche attraverso la definizione di *milestone*
- Milestone: eventi particolare rilevanza (operativa, contrattuale ecc.), la cui mancata realizzazione potrebbe provocare significative criticità per l'intero progetto

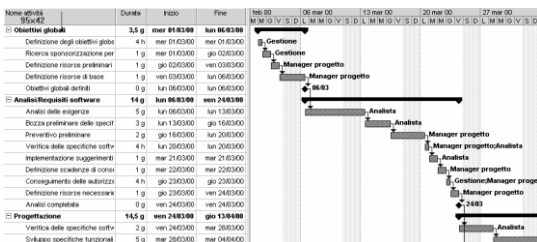
SoftEng

## Diagrammi di Gantt

- Diagrammi a barre che rappresentano in forma integrata e visuale l'evoluzione temporale complessiva del progetto e dell'interrelazione fra tutte, o le principali, attività in cui il progetto stesso è stato scomposto
- Per ciascuna attività possono essere specificate varie informazioni che la caratterizzano (data di inizio e di conclusione, durata prevista, risorse allocate,...)

SoftEng

## Diagramma di Gantt



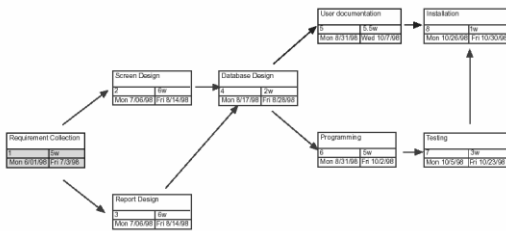
SoftEng

## PERT

- Program Evaluation & Review Technique
- Fondato sul concetto di "evento", che consiste nel raggiungimento di un determinato stato di completamento del progetto, cui è associata una "durata temporale"
- Assunti:
  - il progetto è costituito da un ben preciso insieme di compiti o attività, il cui completamento determina la conclusione del progetto
  - ciascun compito può iniziare e concludersi indipendentemente dagli altri, all'interno di una determinata sequenza
  - i compiti sono "ordinati", devono cioè essere eseguiti secondo una sequenza "tecnologica" coerente e ben definita

SoftEng

## Diagramma PERT



SoftEng

## Metodo per analogia

- La stima delle risorse necessarie viene realizzata attraverso un confronto per singoli fattori, tra intero processo di sviluppo (o specifiche attività) con altri progetti già conclusi per cui esista un valore definito dei costi sostenuti

SoftEng

## Function point

- Costituiscono una misura delle funzioni che l'applicazione fornisce all'utente
- Il metodo consiste nel conteggiare il numero di funzionalità (punti funzione) fornite all'utente dall'applicazione da misurare
- Le funzioni possono essere di 5 tipi (input esterni, output esterni, interrogazioni esterne, file logici interni, file esterni di interfaccia)

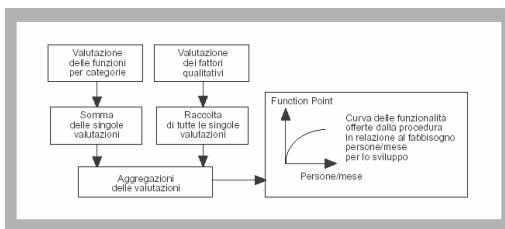
SoftEng

## Calcolo dei function point

- A ogni punto funzione viene assegnato un peso in base alla propria complessità
- Viene assegnato un peso anche all'applicazione nel suo complesso sulla base di una serie di fattori correttivi (general system characteristic) che tentano di catturare ulteriori elementi di complessità
- Al termine dell'analisi si ha un numero adimensionale che rappresenta la quantità di funzionalità offerte all'utente dell'applicazione misurata e che permette di stabilire le persone/mese necessarie allo sviluppo del progetto

SoftEng

## Rappresentazione dei punti funzione



SoftEng

## Package applicativo

- Procedura che è già stata analizzata, progettata e documentata e viene offerta a un'azienda che evita in tal modo di doverla sviluppare in proprio
- Presupposto per l'impiego di software standard è che le esigenze dell'azienda coincidano ampiamente con funzionalità offerte dal software standard disponibile sul mercato

SoftEng

## Vantaggi offerti dal software standard

---

- Costi di acquisto e personalizzazione inferiori rispetto alla realizzazione di software dedicato
- Tempi di implementazione relativamente brevi
- Contiene meno errori rispetto al software dedicato
- Consente di acquisire know-how gestionale e organizzativo non disponibile all'interno dell'azienda
- I package applicativi più diffusi facilitano l'integrazione interaziendale
- Le risorse EDP interne possono essere riallocate su compiti di particolare rilevanza strategica

SoftEng

---

## Svantaggi del software standard

---

- Spesso sussistono discrepanze tra i requisiti aziendali di tipo funzionale e organizzativo e la configurazione degli specifici programmi
- La piattaforma hardware aziendale può risultare incompatibile con lo specifico prodotto software
- All'interno dell'impresa viene sviluppato soltanto un limitato know-how EDP
- L'impresa può trovarsi involontariamente a dipendere dal fornitore di software

SoftEng

---

## Criteri di certificazione della qualità

---

- L'assicurazione della qualità nello sviluppo del software deve far sì che, tanto il processo di sviluppo, quanto il prodotto software soddisfino determinati requisiti (es. tempi di risposta)
- La normativa ISO 9000 presenta linee guida speciali per lo sviluppo del software
- Gli informatici sono stati particolarmente attivi nell'ambito dei Gruppi di Lavoro ISO e hanno preparato numerose Norme e Guide, alcune già emesse e in vigore, altre ancora in fase più o meno avanzata di elaborazione

SoftEng

---