**BIBBIA**

**INGEGNERIA DEL SOFTWARE**

Davide Capricano

d.capricano@studenti.unisa.it

**Capitolo 1 - Introduzione al Software**

**Definizione di software**

Secondo lo standard IEEE, il software è l'insieme dei programmi e procedure e documentazione compresi in un sistema computerizzato.

**Storia del software**

Nel corso dei decenni il software ha subito una evoluzione significativa. Dapprima esso era arte, ossia prodotto da una singola persona per utilizzo privato. Poi è diventato artigianato, ossia applicazioni sviluppate da produttori specializzati su una richiesta specifica del cliente.

Infine, è diventato industriale, ossia prodotto per un mercato più o meno grande che sia, che ha causato un grande aumento in complessità, dimensioni e richiesta.

**1.1 Programma e Prodotto**

Non sono assolutamente da confondere programma e prodotto software.

In un programma, l'autore è anche l'utente, tipicamente non documentato e con pochi test.

Un prodotto software è usato da persone diverse da chi lo ha sviluppato ed è di tipo industriale e serve un approccio formale allo sviluppo e costa circa 10x rispetto al corrispettivo programma, non è presente inoltre economia di scala.

La manutenzione di un prodotto software prende il significato di espansione, cambiamento, evoluzione, mentre in un prodotto hardware ciò non è possibile.

Il prodotto software si divide a sua volta in:

* Prodotto generico:

Sistemi stand-alone da distribuire sul mercato generale dove l’utente paga la licenza d’uso e non il prodotto

* Prodotti specifici:

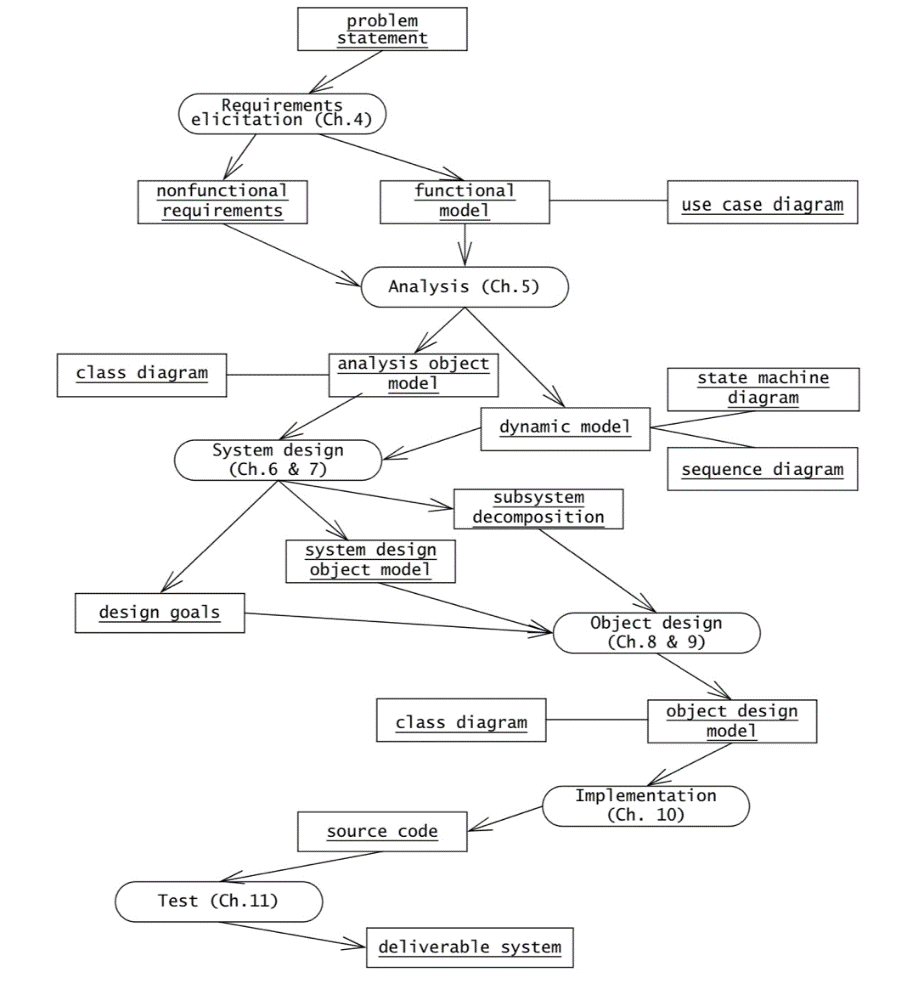
Sistemi commissionati da un fruitore o macrosistema specifico. Il fruitore compra il software nel suo insieme

**Capitolo 2 - Il ciclo di vita del software**

**2.1 Il CVS**

Il processo software può essere formale, semi-formale o informale, ed è definito come l'insieme organizzato di attività che sovrintendono alla costruzione del prodotto e comprende: metodi, tecniche, metodologie e strumenti

Il ciclo di vita del software è il periodo di tempo che inizia quando il prodotto software è concepito e termina quando il prodotto non è più disponibile per l'uso. Il ciclo di vita del software in genere comprende:

* una fase di ideazione e studio di fattibilità
* una fase di analisi dei requisiti
* una fase di progettazione e scomposizione in moduli
* fase di implementazione
* fase di programmazione e test di unità
* Integrazione e test di sistema, rilascio di alfa-test e beta-test
* deployment
* manutenzione
* talvolta, fase di pensionamento

**2.2 - Modelli di ciclo di vita del software**

Un modello del ciclo di vita del software (CVS) è una caratterizzazione descrittiva o prescrittiva di come un sistema software viene o dovrebbe essere sviluppato.

I modelli devono avere le seguenti caratteristiche:

- descrizione dell'organizzazione

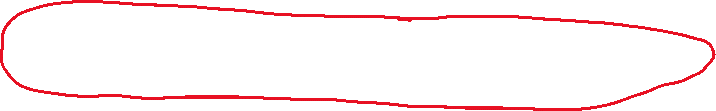
- linee guida

- definizione e scrittura manuale e diagrammi

- determinazione e classificazione dei metodi e strumenti più usati

**2.2.1 - fasi del CVS**

Le fasi generali di molti CVS sono:



**2.2.2 - Waterfall / A cascata**

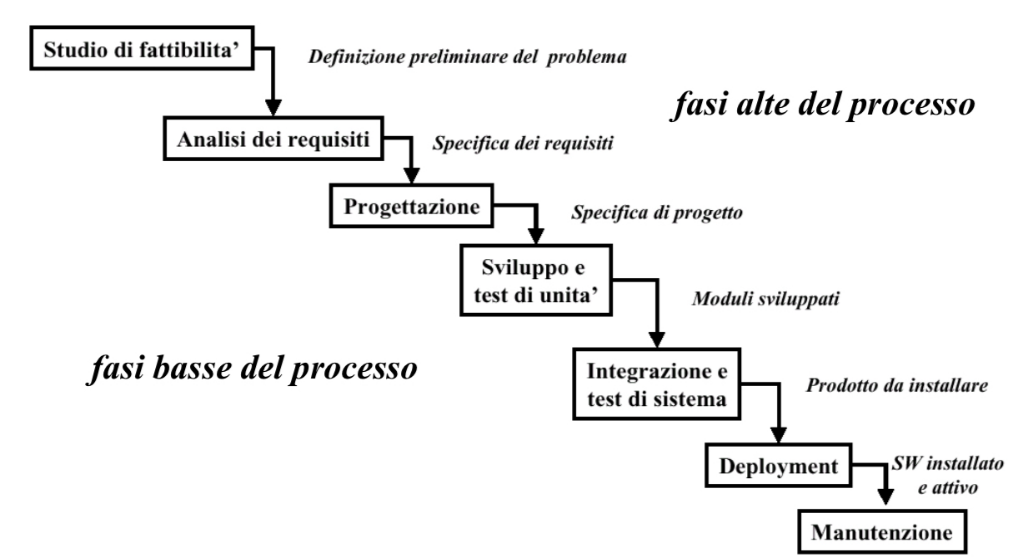
Era un modello popolare negli anni 70, avente un approccio ingegneristico derivante dall'industria manifatturiera. Prevedeva la progettazione, la produzione dei un prototipo e la messa in produzione.

PRO:

Definizione di concetti utili successivamente utilizzati ed espansi in altri modelli, la semplicità di gestione del progetto e comprensione.

CONTRO:

Mancanza di interazione col cliente, data dalla struttura modulare ma poco interconnessa delle fasi (chiusa la progettazione non si modificava). Il cliente era a corrente solo all’inizio e alla fine del progetto.



**2.2.3 – V&V e Feedback**

Ci sono stati molti modelli che hanno provato a correggere il Waterfall. Per esempio, il modello V&V/Feedback che prevede un controllo alla fine di ogni fase. La natura di questo controllo può essere di diversi tipi (verification, validation o entrambi) e se uno o più di questi controlli falliscono, si torna al passo precedente per rivederlo.

La verification è la valutazione della verità della corrispondenza tra il prodotto e la sua specifica, mentre la validation consiste nella appropriatezza di un prodotto rispetto alla sua missione operativa. Il concetto di feedback può essere generalizzato, ed esso può far ritorno ad una fase precedente arbitraria e non necessariamente a quella precedente.

Nel modello a V, le fasi di sinistra sono collegate alle fasi di destra. I test eseguiti in una fase a destra vengono pianificati nella corrispondente fase a sinistra. Se si trova un errore in una fase a destra (es. testing di sistema) si rivedono le attività della fase di sinistra corrispondente. È possibile iterare migliorando requisiti, progetto e codice.

**2.3 – I principi dell’ingegneria del software**

* Trasformazioni formali:

Lo sviluppo viene visto come una sequenza di passi che trasformano formalmente una specifica in una implementazione. Ogni passo è composto da descrizioni formali e può dare origine in modo quasi automatico al successivo ed è difficilmente applicabile a contesti reali ma può essere utile a sviluppo di sistemi orientati alla safety. Inoltre, costa di più ed è difficile realizzare una interfaccia utente.

* Sviluppo a componenti:

Si basa sullo sviluppo di componenti già esistenti (off the shelf). Utile per sistemi sufficientemente generici, ma man mano che aumenta la specificità di un sistema, diminuisce la possibilità di usare componenti già esistenti.

* Sviluppo evolutivo:

Questa tipologia di sviluppo del software prevede la rivisitazione di fasi precedenti, anche molto precedenti, del progetto quando necessario. Un approccio del genere