12. FONDAMENTI DI INGEGNERIA DEL SOFTWARE

PREMESSA

La produzione del software non può essere affidata *all'improvvisazione*: sapere scrivere algoritmi e programmi non garantisce di per se l'avere un software di *buona qualità.*
La *produzione artigianale* del software non sempre è garanzia di efficienza e di qualità: per ottenerle è necessario utilizzare **metodi di produzione** che possono essere applicati solo da esperti del settore.

A partire dagli ani 60 si è avuta una rivoluzione nell'ambito della produzione del software attraverso l'introduzione di nuove tecniche e metodologie in grado di affrontare e risolvere i problemi aventi le seguenti finalità:
- la **formalizzazione** dei metodi di *sviluppo* e di *documentazione* del software;
- la **realizzazione** di prodotti software di *buona qualità* ed a *costi contenuti*;
- la **riduzione** della *complessità* della tecnologia;
- la **facilità di utilizzo** dei programmi da parte dell'utente.

In Germania nel **1968** nel corso di una conferenza della NATO si pose l'attenzione sull'esigenza di disporre di metodi mutuati dall'industria per la produzione del software. Nacque così l'**ingegneria del software** (ossia **Software Engineering**) che applicava al software ed alla sua produzione i principi dell'ingegneria che permettevano la pianificazione accurata ed il controllo rigoroso di tutto il processo di sviluppo.

PROGETTAZIONE DI UN SISTEMA

Per sviluppare un sistema ed ottenere un prodotto finito occorre un **progetto**.
Progettare un sistema significa:
- studiare il progetto ossia comprendere il suo ambito di realizzazione, gli obiettivi che devono essere raggiunti, la sua fattibilità;
- analizzare i requisiti del progetto per comprenderne le caratteristiche e redigere un piano di lavoro;
- specificare il progetto ossia definire l'architettura del prodotto da realizzare utilizzando modelli;
- realizzare il progetto ossia "costruire" il prodotto.

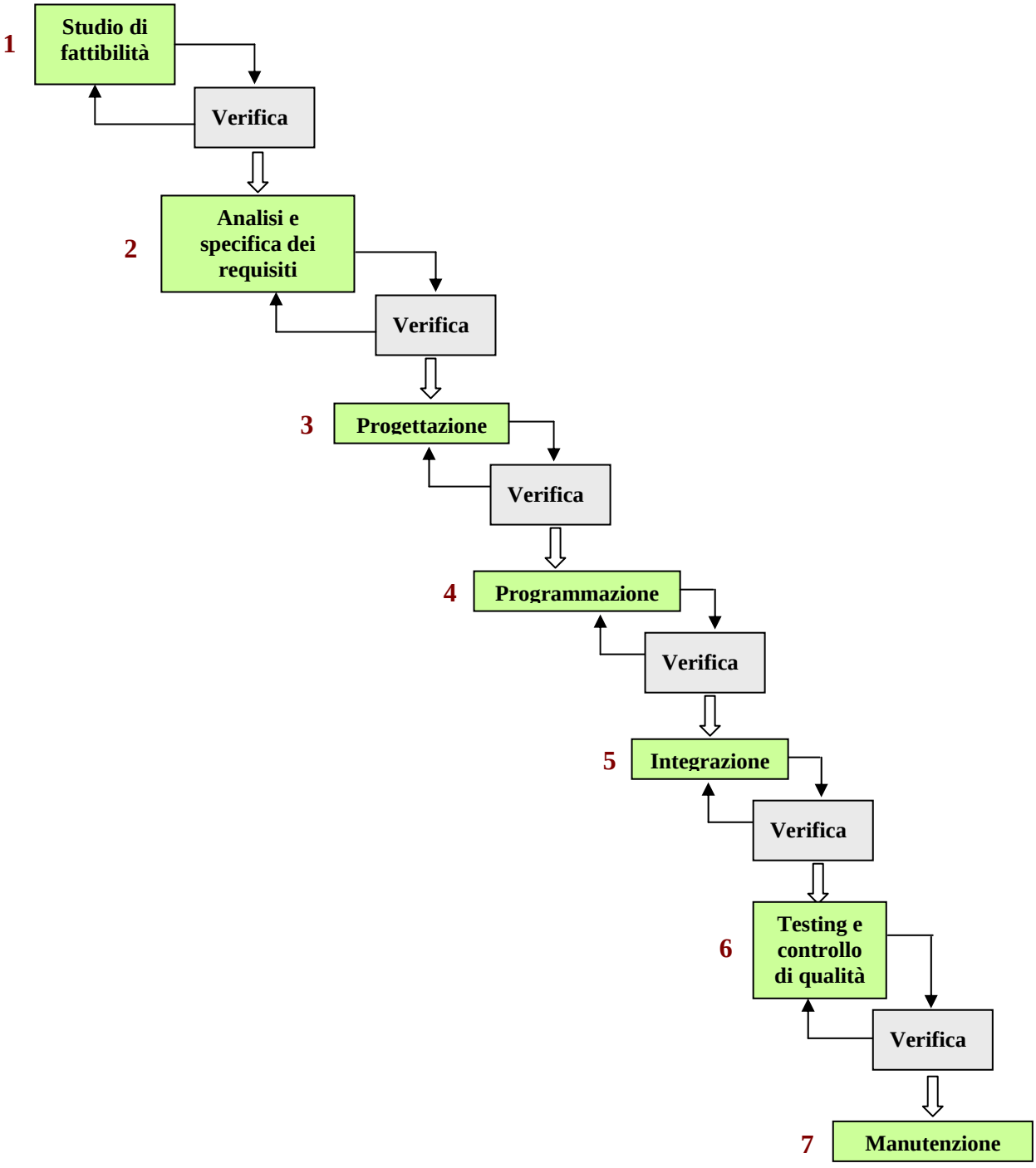
CICLO DI VITA DEL SOFTWARE

In informatica per quanto detto finora non si opera improvvisando ma seguendo schemi precisi in grado di assicurare una produzione di un buon software.
Per fare ciò occorre capire che cosa si intende realizzare, se è possibile realizzarlo, come realizzarlo e con quali strumenti. Solo alora si può passare alla realizzaione.

Riassumendo quindi qualsiasi attività di sviluppo software prevede:
1) un'attività di **studio del progetto**;
2) un'attività di **progetto vero e proprio**;
3) un'attività di **realizzazione del progetto**.

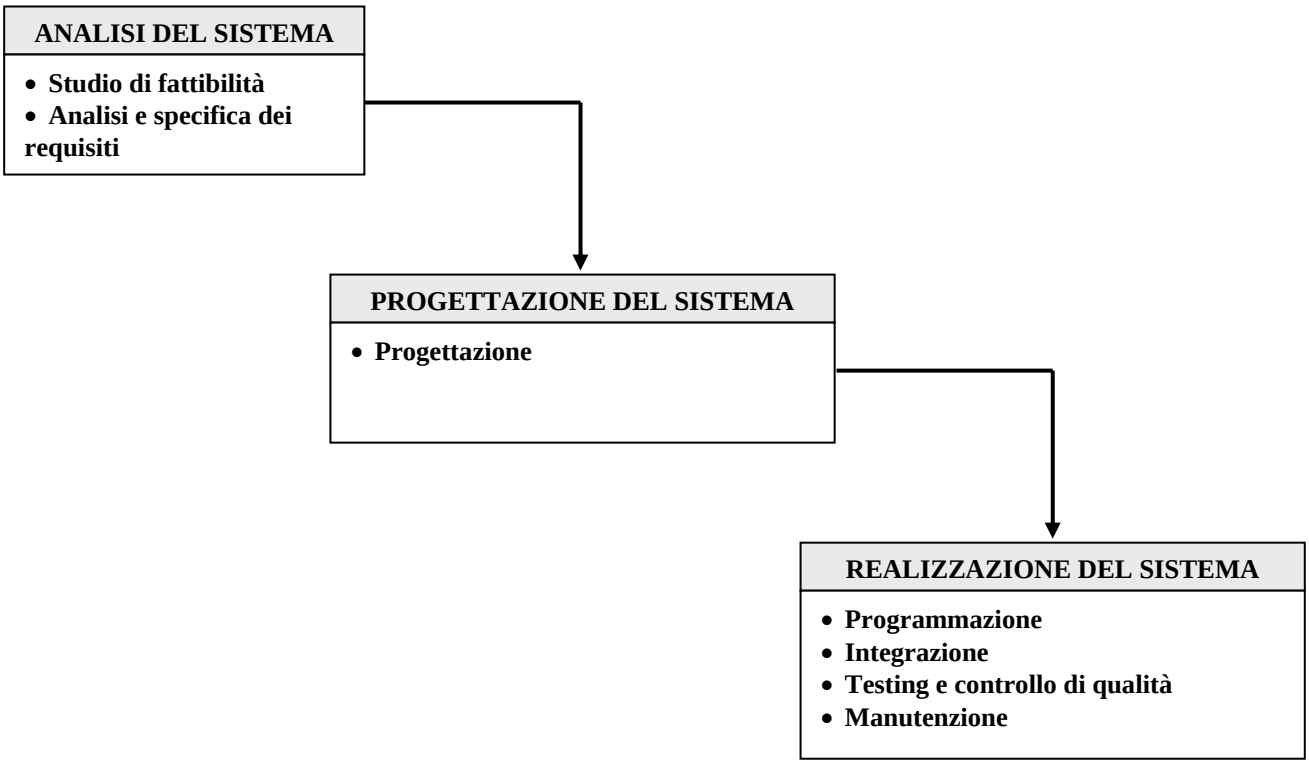
DEF: Il **ciclo di vita del software** (ossia **software life cycle**) è il processo scomposto in una *sequenza ordinata* di **fasi** cronologiche rigorosamente definite mediante il quale si giunge alla produzione di un pacchetto software

Lo schema più tradizionale per rappresentare il ciclo di vita del software è il **modello a cascata** (ossia **waterfall model**)



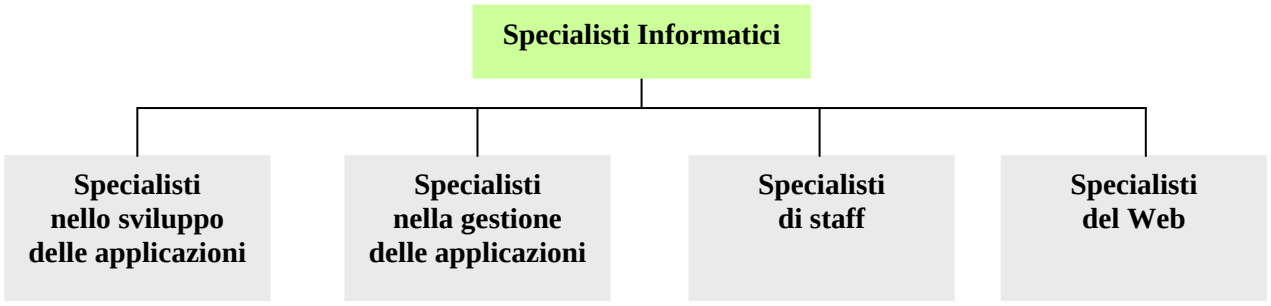
Le **fasi** che compongono il modello a cascata del ciclo di vita del software hanno le seguenti caratteristiche:
- la **parte finale** di ciascuna fase è caratterizzata dalla presenza di un processo di verifica e di controllo di validità. E' necessario poer eliminare gli errori commessi nello svolgimento della fase e per controllare la correttezza metodologica;
- la **messa a punto** di ciascuna fase interessa, in genere, solo la fase successiva.

In generale ogni fase produce un output che costituisce l'input della fase successiva.
Prima di analizzarle singolarmente ricordiamo che esse possono essere raggruppatenelle tre **macrofasi** seguenti sulle quali si basa l'intero processo di sviluppo:

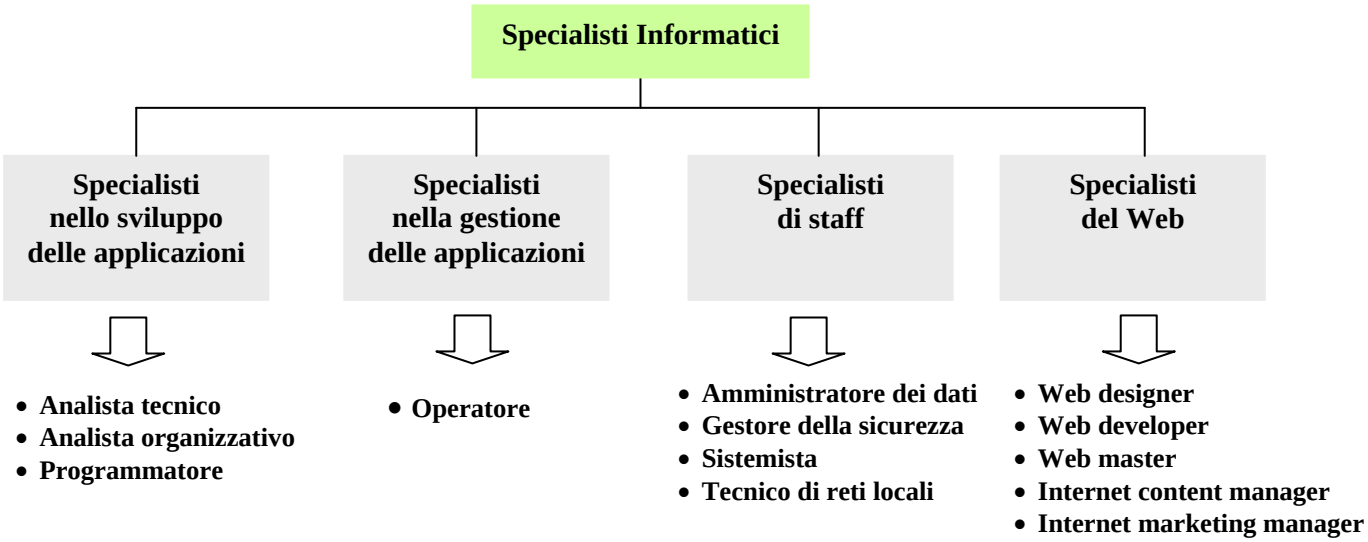


LE FIGURE PROFESSIONALI

La **funzione informatica** ossia quella particolare funzione che ha l'obiettivo di **produrre software** attraverso lo *studio*, la *progettazione* e la *realizzazione* di un *sistema complesso* è svolto da uno staff di specialisti.
Essi operano con ruoli e compiti che dipendono dal contesto dello sviluppo dell'intero sistema.
La classificazione degli **specialisti informatici** è riportata nel seguente diagramma:



Noi ora esamineremo solo alcune delle figure di specialisti esistenti ossia quelle che *più frequentemente interagiscono* in un processo informatico.



Specialisti nello sviluppo delle applicazioni
Sono le figure più antiche nella giovane storia della programmazione informatica:

- l'**analista organizzativo**: è la figura di confine tra l'utente e chi realizza il progetto. Il suo principale obiettivo è interpretare le esigenze del ***committente*** (ossia colui che ha ordinato l'esecuzione del progetto) e definire le ***specifiche funzionali*** (ossia cosa deve fare) del sistema.

- l'**analista tecnico**: porta avanti il progetto messo a punto dall'analista organizzativo ed ha il compito di formalizzare le direttive di quest'ultimo per tradurle poi in ***specifiche tecniche*** (ossia stende l'architettura logica e fisica del sistema).
A seconda della mansione e delle competenze l'analista tecnico può:
  - essere il responsabile dell'intera progettazione del sistema (**capo progetto**);
  - essere un semplice **progettista** di staff;
  - gestire anche la definizione degli algoritmi risolutivi (**analista-programmatore**).

- il **programmatore**: è lo specialista che si occupa dello *sviluppo degli algoritmi* e della loro *codifica*. Viene coinvolto nella fase di progettazione tecnica del sistema e cura tra l'altro anche:
  - la *documentazione* dei programmi al fine di garantirne una buona *manutenzione*;
  - la predisposizione dei dati per il *test* dei programmi.

Specialisti nella gestione delle applicazioni
Questo gruppo di specialisti si identifica essenzialmente in un'unica figura ossia:

- l'**operatore**: che si occupa dell'andamento del centro di elaborazione dati e segue globalmente l'esecuzione dei lavori.. Lo fa attraverso la pianificazione, l'organizzazione d il controllo di tutte le attività oltre che attraverso la gestione tecnica dei singoli dispositivi fisici.

Specialisti di staff
Pur non esistendo sin dall'inizio della storia dell'informatica, sono figure professionali che si sono imposte nel tempo come essenziali per la realizzazione di un progetto informatico. Tra gli altri

- l'**amministratore dei dati**: è una figura professionale per certi versi paragonabile all'analista organizzativo. Cura la progettazione dello ***schema concettuale dei dati*** ossia lo schema contenente la descrizione dei dati e dei loro vincoli che vengono individuati partendo dalle notizie fornite dall'utente e dall'analista tecnico.

- il **gestore della sicurezza**: si occupa della ***sicurezza del sistema informatico*** intesa sia come protezione del sistema sia come sicurezza di utilizzo e quindi come protezione dei dati introdotti.

- il **sistemista**: si occupa della gestione dell'hardware, del software di base e dell'installazione del sistema prodotto oltre che degli altri eventuali pacchetti di supporto ad esso.

Specialisti del Web
Sono figure professionali "giovanissime" nate dall'evoluzione delle nuove tecnologie informatiche, Internet, il WWW e la new economy molto richieste dalle aziende.

- il **Web designer**: si occupa della ***ideazione*** e dello ***sviluppo*** di tutto ciò che riguarda l'aspetto grafico di un sito web, da un punto di vista sia tecnico sia creativo. Gestisce la struttura della navigazione, l'impaginazione, le immagini, l'audio, il video ed ogni altro eventuale contenuto multimediale.

- il **Web developer**: si occupa della ***realizzazione delle applicazioni*** per Internet e le Intranet. Deve conoscere i linguaggi di formato come l'HTML, i prodotti di web publishing (come Dreamweaver, Flash, etc.) e linguaggi di programmazione specifici orientati al web (Java, Javascript, PHP, etc.).

- il **Web master**: si occupa della ***gestione tecnica del sito.*** Coordina il lavoro dei web developer e deve possedere forti competenze tecnologiche.

- l'**Internet content manager**: è il progettista ossia l***'ideatore del prodotto multimediale***. E' la figura che più di tutte le altre deve saper congiungere sapere umanistico ed altre competenze tecnologiche.

- l'**Internet marketing manager**: si occupa dello sviluppo progettuale e delle redazioni dei ***piani* di *impresa e di marketing***, a livello sia strategico sia operativo e della gestione e promozione delle azioni di marketing sulla Rete, partendo dall'analisi della clientela sino a giungere alla gestione dei sistemi di vendita online.

DETTAGLIO DELLE FASI DEL CICLO DI VITA DEL SOFTWARE

1 Lo studio di fattibilità

Lo *studio di fattibilità (o progettazione preliminare)* è la fase iniziale dell'intero ciclo di sviluppo. Essa permette di stabilire gli obiettivi da raggiungere e di verificare se è possibile realizzarli.

Per avere informazioni dettagliate indispensabili per avviare un progetto informatico:
- si devono raccogliere i "desiderata" utente per mezzo di una **intervista conoscitiva** che consenta all'analista organizzativo di sapere cosa va informatizzato;
- si devono effettuare delle **analisi preliminari** dei vari pacchetti software già presenti sul mercato per valutarne pregi e difetti;
- si devono valutare le possibili soluzioni ideate determinandone **costi e benefici** al fine di scegliere quella più conveniente.

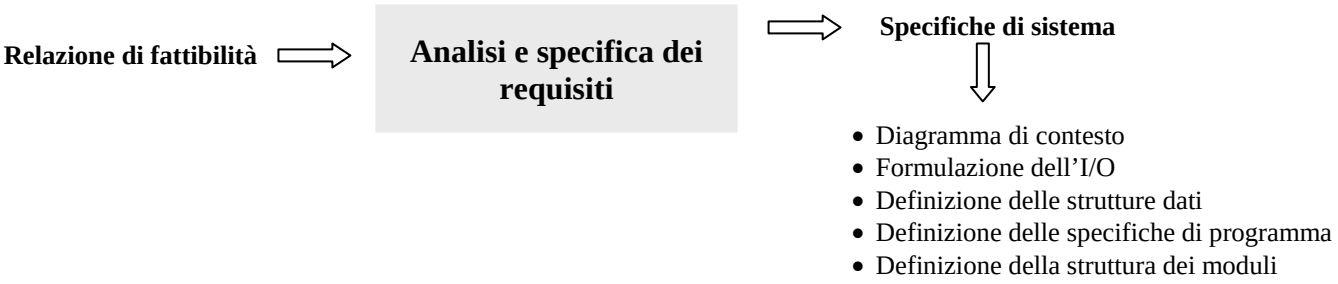
Terminata questa attività di analisi viene redatto un documento chiamato **relazione di fattibilità** che contiene la *formalizzazione* della proposta di sviluppo e la documentazione del progetto.
La sua eventuale accettazione da parte del cliente permette l'avvio della realizzazione dello stesso.

Nonostante questa fase sia una fase preliminare è importante affrontare fin da ora il problema delle **spese di manutenzione** che l'utente dovrà sostenere periodicamente. Va infatti fatto notare fin da ora che i costi di produzione sono in generale molto più bassi di quelli di modifica.
Infatti spesso anche gli eventuali costi aggiuntivi dovuti al prolungamento della fase di analisi e progettazione a causa della una scarsa precisione con cui sono stati comunicati dall'utente i dettagli, vengono caricati e quindi recuperati sui costi di manutenzione.
Ecco perché in definitiva tali costi finiscono con l'avere una incidenza notevole sul costo globale del prodotto.

2 Analisi e specifica dei requisiti

Se l'esito dello studio di fattibilità è stato positivo (ossia si è avuto l'OK dell'utente ed il progetto può essere realizzato), lo sviluppo del progetto entra nel vivo attraverso la fase di *analisi e specifica dei requisiti (o progettazione concettuale).*
In questa fase, fondamentale dell'intero progetto, si analizza accuratamente il problema da risolvere e si valuta quale obiettivo si vuole raggiungere al fine di individuare una serie di procedimenti che permettano di ottenerlo.
Tale fase è anche la fase più critica dell'intero progetto in quanto una specifica dei requisiti parziale o incompleta potrebbe comportare gravi problemi al team di progetto (con eventuali penali e/o mancato guadagno).
La formalizzazione dei requisiti da sviluppare è svolta sia dall'analista organizzativo sia dall'analista tecnico che insieme provvedono alla stesura di un documento chiamato **specifiche di sistema** all'interno del quale sono contenute tutte quelle specifiche in grado di mettere in evidenza i **requisiti funzionali** del prodotto software da sviluppare

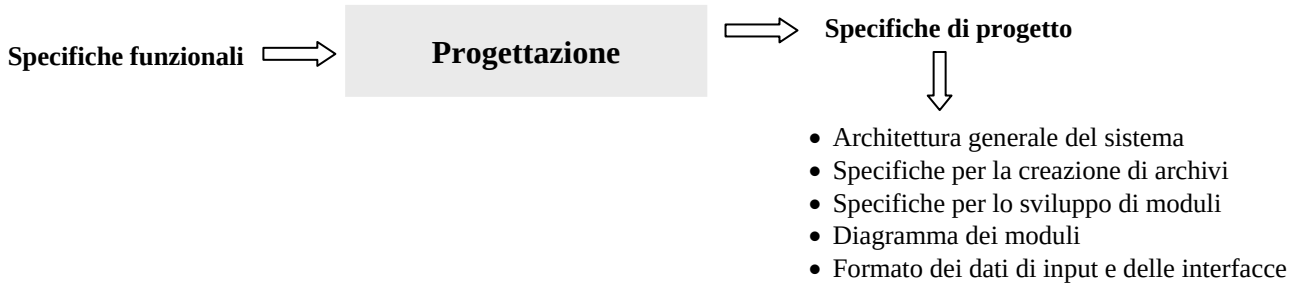
In particolare le specifiche di sistema contengono:
- la *descrizione del sistema* così come dedotto dallo studio di fattibilità;
- la descrizione di eventuali *vincoli informatici e vincoli prodotti dall'ambiente* ai quali il progetto deve sottostare (esempio di vincolo informatico può essere rappresentato dalla configurazione minima e massima dell'hardware sul quale il prodotto dovrà "girare");
- la *rappresentazione grafica dei modelli* relativi ala soluzione scelta.



In particolare le **specifiche di programma** descrivono la rappresentazione del progetto e contengono le soluzioni di massima dei vari algoritmi senza entrare in dettagli tecnici.

3 La progettazione

E' senza dubbio la fase più delicata dell'intero ciclo di vita del software.
L'obiettivo della progettazione è quello di illustrare dettagliatamente i *requisiti* del sistema informatico. In questa fase si passa *dal cosa fare al come farlo.*



Il prodotto di questa fase non è ancora una procedura automatizzata ossia non è ancora un programma. E' l'ultimo stadio di avanzamento del processo prima della reale implementazione.
Alla fine di tale fase i progettisti redigeranno un documento detto **specifiche di progetto** che deve contenere:
- gli *obiettivi principali* del progetto;
- l'*elenco delle risorse* e dei *documenti* utilizzati;
- la *descrizione dell'architettura generale* del progetto (in particolare l'architettura delle strutture dati utilizzate);
- la *descrizione dei moduli* che compongono il progetto;
- le *direttive per lo svolgimento del test*;
- la *guida all'installazione del prodotto* software ottenuto.

4 La programmazione

In questa fase si passa all'effettiva concretizzazione del cosa fare stabilito nella fase precedente.
Avviene quindi la costruzione e l'implementazione di una serie di programmi per l'elaboratore che costituiscono l'intero sistema.
Programmare significa non solo codificare e testare i singoli moduli (**test unitario**), ma anche creare gli archivi occorrenti nonché documentare opportunamente attraverso l'uso mirato di commenti il codice scritto, al fine di rendere più agevole il lavoro di manutenzione in caso di errori.

5 L'integrazione

In fase di integrazione tutti i moduli codificati e testati nella fase precedente vengono *uniti insieme* (*integrati*) ed il prodotto software così ottenuto viene testato nella sua totalità.
In questa fase viene eseguito il **test di integrazione** che ha lo scopo di valutare sia il funzionamento globale del sistema software sia la rispondenza delle interfacce di comunicazione tra i moduli componenti.
Poiché in generale i prodotti software sono composti da numerosi moduli è rischioso verificare il programma mediante una analisi globale. E' meglio analizzare l'integrazione software totale attraverso le integrazioni parziali modulo per modulo per poi verificare alla fine tutto l sistema (**test incrementale**).
Il lavoro di integrazione è svolto assieme dal gruppo di professionisti che si è occupato della progettazione e da quello che si è dedicato alla programmazione.

6 Testing e controllo di qualità

Per verificare la correttezza si sottopone il programma alla fase di **testing** mediante la quale il programma viene sottoposta ad una serie di **test funzionali.**
Testare un programma significa percorrere tutti i cammini che sono stati previsti al fine di verificare la correttezza di percorso di ciascuno (**copertura topologica**). Vanno anche previsti tutti i casi possibili compresi i casi limite al fine di valutare il comportamento del sistema in tutte le possibili situazioni **(copertura funzionale**).
Le attività previste per la fase di testing sono le seguenti:
- *preparazione dell'ambiente di prova* (o *di test*) analogo a quello in cui il software dovrà poi funzionare utilizzando le apparecchiature a disposizione dell'utente;
- *esecuzione delle prove* (o *dei casi di test*) al fine di verificare la rispondenza del software alle specifiche raccolte in fase di analisi;
- *certificazione delle prove* (o *dei test*) vale a dire verificare che le prove svolte siano coerenti con gli standard di mercato e con le specifiche di progetto.

Parliamo più dettagliatamente dell'*esecuzione delle prove* (o *dei casi di test*).
In genere i test vengono dapprima svolti dallo steso staff che ha implementato il sistema e vengono svolti in maniera scrupolosa anche per non danneggiare il nome e l'immagine dell'azienda produttrice: in tal caso si parla di **alfa test ossia α-test.**

Questa fase di testing può anche essere affidata ad un gruppo di aziende opportunamente scelte che testano e verificano i programma simulando situazioni reali. In caso di errori il programma viene immediatamente restituito alla casa produttrice che apporterà le modifiche necessarie. In questo caso si parla di **beta test o β-test.**
E' possibile anche coinvolgere l'utente stesso nella fase di test.
Per seguire il comportamento del sistema ed analizzare le varie risposte i risultati forniti dal testing vengono registrati su di un apposito documento detto **checklist** sul quale sono anche indicati i dati attesi dal programma

Una volta ottenuto il programma completo si da inizio al **controllo di qualità**.
In questa fase si verifica la **qualità del prodotto** ossia l'aderenza dei risultati ai bisogni e quindi la rispondenza del progetto alle esigenze manifestate dal committente.
Controllare la qualità di un prodotto consiste nel **misurare** le sue caratteristiche.
Far questo significa:
- stabilire quali devono essere le caratteristiche importanti da valutare ossia le **capacità del prodotto** *(o capability*);
- scindere le capacità del prodotto in **caratteristiche di dettaglio** (o *property*) oggettivamente misurabili;

La misurazione delle property rappresenta il *controllo di qualità del prodotto*., ma per giungere ad un prodotto di qualità occorre che anche le fai di produzione siano verificate: è quindi necessario anche un *controllo di qualità del processo*
La somma di queste due qualità porta alla **qualità totale** ossia a prodotti che soddisfano il cliente.

Affinché un prodotto software sia di buona qualità deve rispondere ad alcuni requisiti fondamentali (**attributi di qualità**) divenuti ormai standard.
Un software deve essere.
- **affidabile**: ossia privo di errori ed in grado di soddisfare completamente i requisiti richiesti in fase di analisi;
- **robusto**: ossia in grado di comportarsi correttamente anche di fronte a situazioni anomale (tipo inserimento dati errati o utilizzo di dispositivi periferici spenti)
- **semplice nell'utilizzo o user friendly**: ossia utilizzabile con successo anche da utenti non esperti;
- **sicuro**: ossia in grado di garantire la sicurezza delle informazioni sia dall'accesso di utenti esterni (*security*) sia dal punto di vista della sicurezza interna (*safety*) ossia capace di proteggere l'utente da errori accidentali e situazioni critiche (copie, cancellazioni accidentali, etc.);
- **efficiente**: ossia in grado di elaborare e conservare i dati nel modo più veloce possibile gestendo le risorse di memoria in modo altamente funzionale ossia senza sprechi;
- **comprensibile**: ossia in grado di esser appreso con facilità. Ciò dipende in gran parte dalla presenza di una documentazione comprensibile ed esaustiva fornita con il prodotto. Tale caratteristica riguarda anche gli specialisti di progetto in quanto un sistema comprensibile deve essere ottimamente documentato solo in questo modo sarà facilmente mantenibile.

La **Certificazione di Qualità** è il riconoscimento formale che attesta che l'organizzazione (nel nostro caso la *software house*) ha svolto un insieme di attività progettate e documentate per garantire che i prodotti/servizi siano realizzati secondo determinati standard, nel rispetto di un piano che mira a garantire la qualità richiesta.
Il *prodotto* o il *servizio* certificato diventa così "di qualità" quando assicura al consumatore/utente di essere stato realizzato utilizzando risorse, prodotti e tecniche di realizzazione conformi agli sta

Organismi internazionali quali **l'ISO** (**I**nternational **O**rganization for **S**tandardization) e l'**IEEE** (**I**nstitute of **E**letrical and **E**lectronic **E**ngineers) hanno emanato specifiche e norme ufficiali che consentono di determinare la qualità.
Le norme sul **Sistema Qualità** che guidano alla certificazione sono state elaborate dal comitato tecnico 176 della ISO che nel 1987 ha emesso le prime norme sulla qualità le **ISO-9000**.
Le norme principali sono:
- la **ISO-9000**: assunta come norma di presentazione della serie;
- la **ISO-9001**: che indica le regole relative alla *progettazione*;
- la **ISO-9002**: che indica le regole relative alla *produzione*;
- la **ISO-9003**: che indica le regole relative alla *installazione ed ai collaudi*;
- la **ISO-9004-1**: che indica le norme guida della *conduzione aziendale* per la qualità;
- la **ISO-9004-2**: che si riferisce in particolar modo alla *gestione della qualità nei servizi*.;

7 Manutenzione

Con il termine manutenzione si intende l'insieme degli interventi che vengono attuati sul sistema software per garantirne l'efficienza. In origine si pensava che la manutenzione corrispondesse solo al bug fixing, ma oggi la situazione è più complessa; la manutenzione riguarda infatti ogni miglioramento del software e andrebbe indicata più precisamente come evoluzione del software. Ormai circa il 55%-60% dei costi dipende proprio dalla manutenzione.
Per analizzare questi costi occorre suddividere in manutenzione:

- **Correttiva**
- **Adattativa**
- **Perfettiva**

La **manutenzione correttiva**

- Elimina gli errori presenti sin dall'inizio
- Elimina gli errori introdotti da precedenti interventi di manutenzione
- Rappresenta circa il 20% del totale della manutenzione

La **manutenzione adeguativa o adattativa**:

- Modifiche a seguito di cambiamenti nell'ambiente
- Cambiamenti nell'Hardware, nel Sistema operativo, etc.
- Rappresenta circa il 20% del totale della manutenzione

La **manutenzione evolutiva o percettiva**

- Modifiche per migliorare le qualità del software
- Introduzione di nuove funzionalità
- Miglioramento delle funzionalità esistenti non richiesto in precedenza
- E' la parte più consistente della manutenzione (circa il 60% del totale)

La Documentazione

La **Documentazione**, pur non avendo un obiettivo specifico all'interno del ciclo di vita del software, rappresenta una fase trasversale a tutto il progetto.
Un prodotto scarsamente documentato non può essere considerato l'output di un buon progetto in quanto non garantisce il raggiungimento di molti attributi di qualità ed ostacola il lavoro di manutenzione.

La **preparazione ella documentazione** deve procedere in parallelo ad ogni fase del ciclo di vita.
Questo vale sia per la documentazione dei programmi, sia per la documentazione dell'utente: rimandare la stesura dei vari documenti è pericoloso in quanto si rischia di produrre documenti parziali assolutamente non in linea rispetto a quanto implementato.
La scarsezza o la non corrispondenza o la completa assenza di documentazione influiscono negativamente non solo sulla futura manutenzione, ma anche sull'utilizzo del prodotto da parte dell'utente.

Gli obiettivi della documentazione sono:
- fornire la **documentazione del sistema** ossia il materiale documentativi consultabile dagli esperti in fase di intervento sul prodotto;
- fornire la **documentazione per l'utente** ossia il materiale documentativi consultabile dall'utente nel corso della sua normale attività lavorativa. Tale documentazione è più frequentemente fornita in formato elettronico sottoforma di file (in genere PDF oppure DOC oppure HTML) su CD ma può anche essere fornita sottoforma di *guida on-line o help online.*