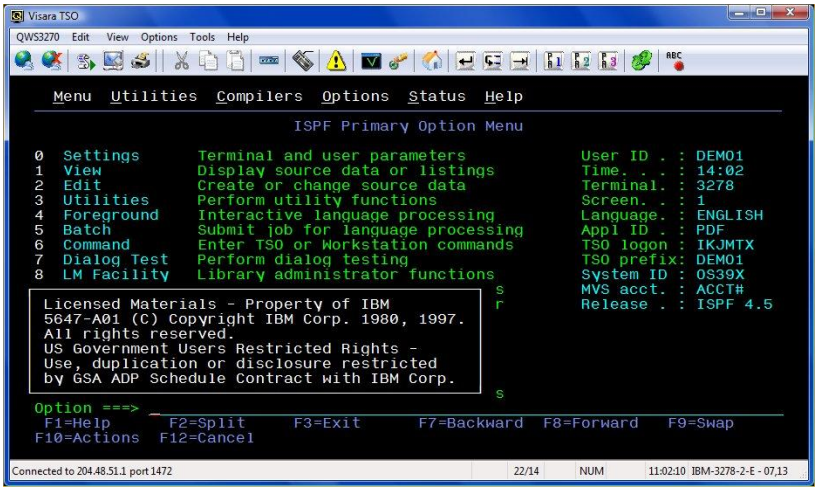
Architetture applicative

- MainFrame Batch
- MainFrame CICS (3270)
- Distribuito puro (PC non integrati)
- Client – Server
- Multi-Tiers
- SOA



Web Services

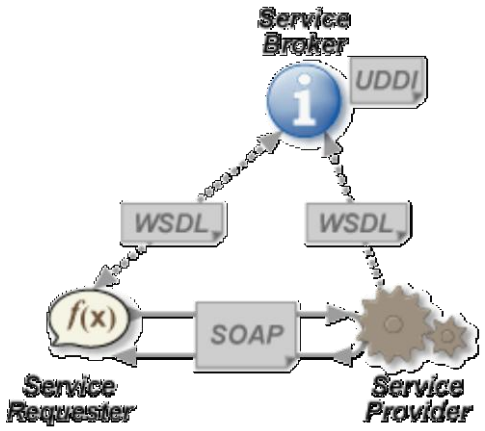
Definiti da World Wide Web Consortium (W3C)

Un Web Service (servizio web) è un sistema software progettato per realizzare l'interoperabilità tra diversi sistemi collegati da una medesima rete (sia locale che “Wide”).

Elemento fondamentale di un Web Service è quella di offrire un'interfaccia software (descritta in un formato standard (WSDL)) utilizzando la quale altri sistemi possono interagire con il Web Service stesso, tramite le funzioni descritte nell'interfaccia stessa, utilizzando "messaggi" racchiusi in una "busta".

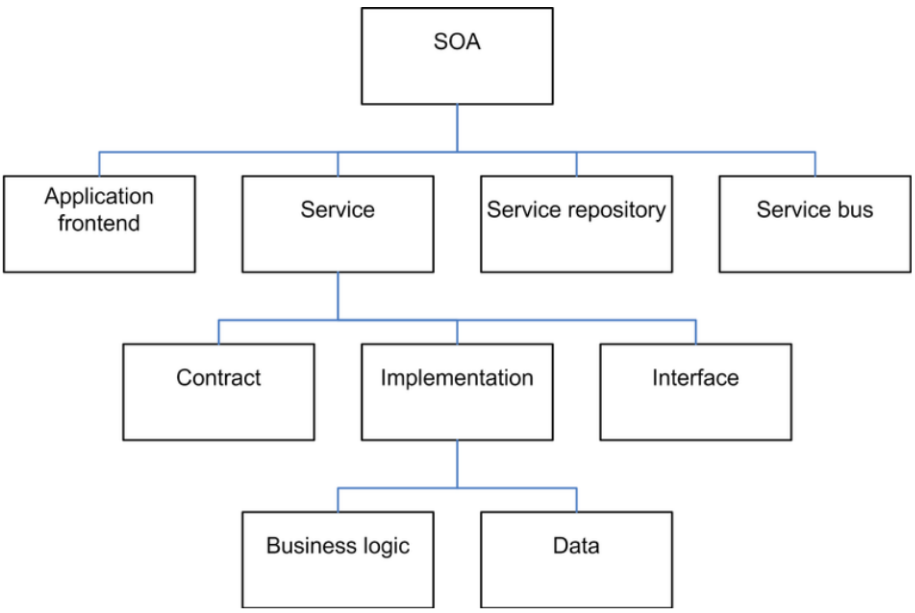
Tali messaggi sono trasportati tramite il protocollo HTTP e scritti in formato XML.

- Web Service
- Interfaccia
- Messaggi
- SOAP
- HTTP
- XML



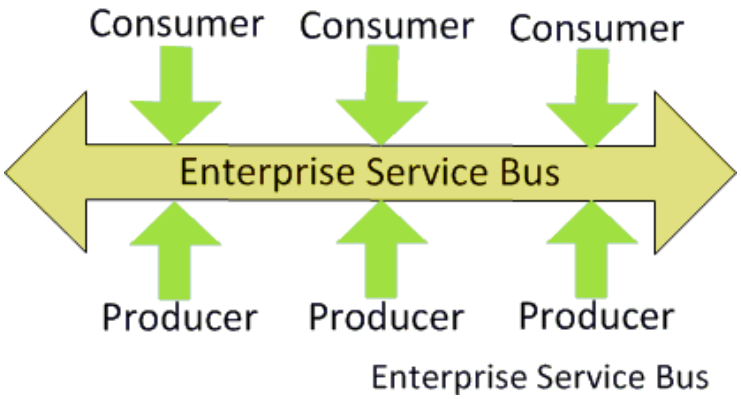
SOA

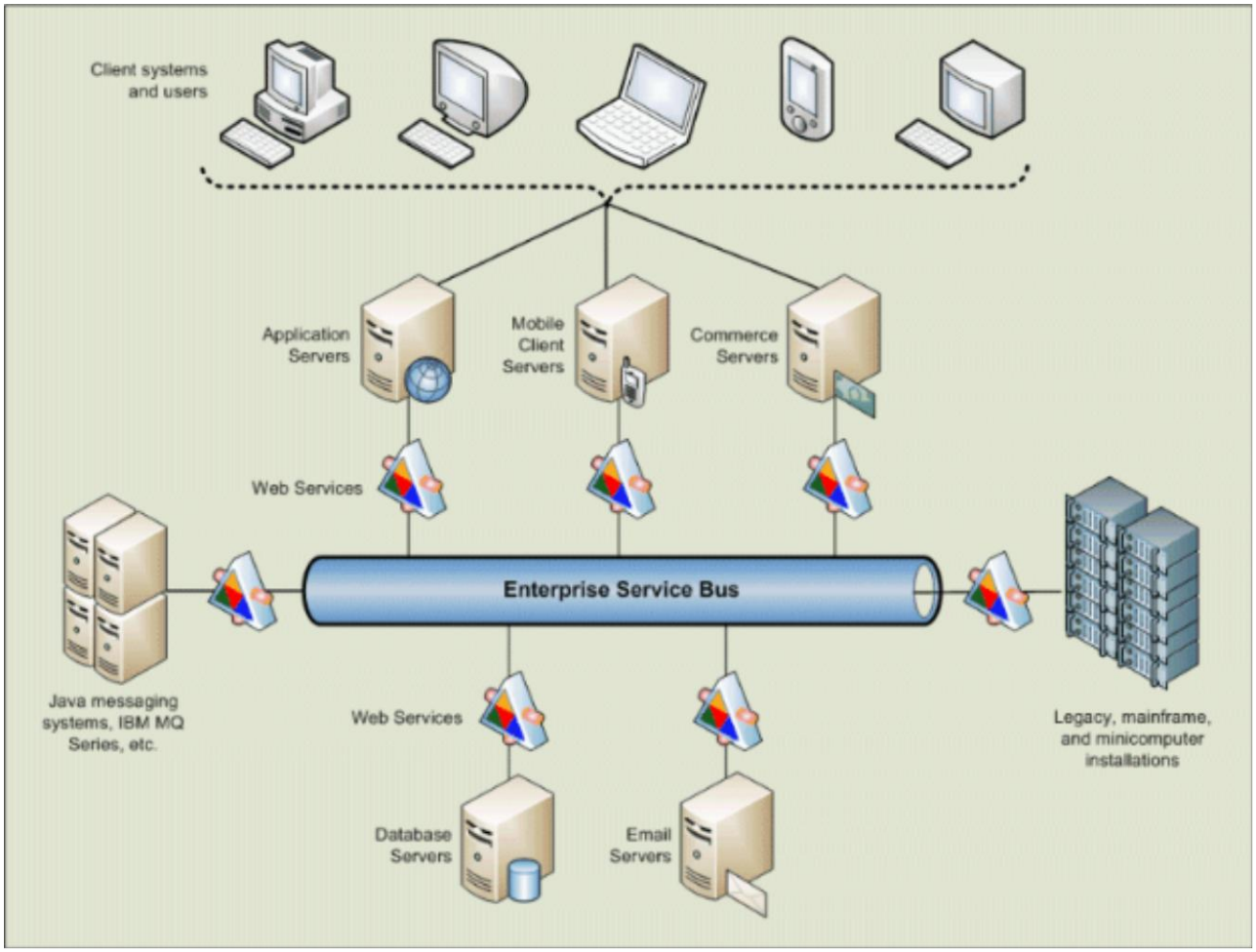
Service-Oriented Architecture (SOA) indica generalmente un'architettura software adatta a supportare l'uso di servizi Web per garantire l'interoperabilità tra diversi sistemi così da consentire l'utilizzo delle singole applicazioni come componenti del processo di business e soddisfare le richieste degli utenti in modo integrato e trasparente.



ESB

Un Enterprise Service Bus (ESB) è un'infrastruttura software che fornisce servizi di supporto ad architetture SOA complesse. Un ESB si basa su sistemi disparati, interconnessi con tecnologie eterogenee, e fornisce in maniera consistente servizi di orchestration, sicurezza, messaggistica, routing intelligente e trasformazioni, agendo come una dorsale attraverso la quale viaggiano servizi software e componenti applicativi.





Clustering

Un cluster (dall'inglese grappolo), è un insieme di computer connessi tramite una rete telematica.

Lo scopo di un cluster è quello di distribuire una elaborazione molto complessa o molto pesante tra i vari computer componenti il cluster.
L'importanza dei cluster è che l'intero cluster può essere visto dalle applicazioni client come un unico computer, non come la somma di tanti tra loro distinti.

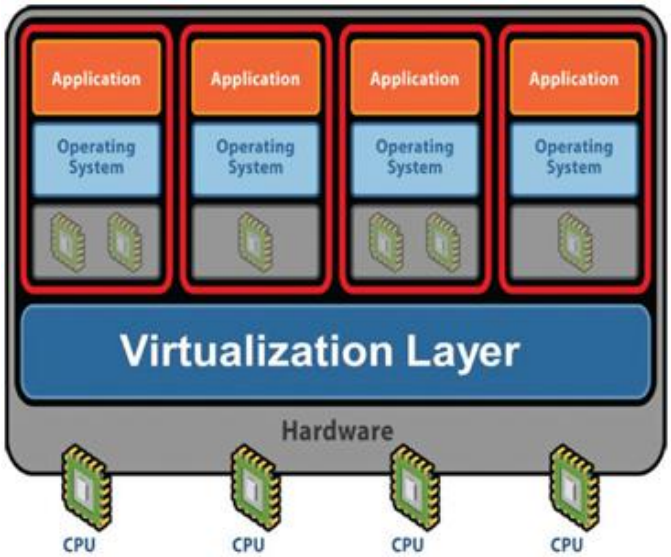
Esistono tre tipi di cluster:

- Fail-over
- Load balancing
- High Performance Computing

Virtualizzazione

Con il termine "virtualizzazione" si indica la possibilità di eseguire contemporaneamente più sistemi operativi virtuali su una singola macchina fisica, mantenendoli però, dal punto di vista logico, distinti.
Il sistema operativo "ospitante" (l'host) crea di fatto una sorta di hardware virtuale all'interno del quale vengono eseguiti i diversi sistemi operativi "ospiti" (i guest).

Di fatto la parte inferiore dello stack software è un sistema operativo ordinario che è installato direttamente sul server.
Sopra di questo, un layer di virtualizzazione gestisce il reindirizzamento e l'emulazione delle risorse fisiche rimappandole verso quelle virtuali.
Nei server virtuali sono installati dei sistemi operativi, ovvero i guest, i quali penseranno di avere tutta una macchina per sé, ignorando il fatto di essere in una macchina virtuale e quindi anche l'esistenza degli altri guest eseguiti sullo stesso hardware.

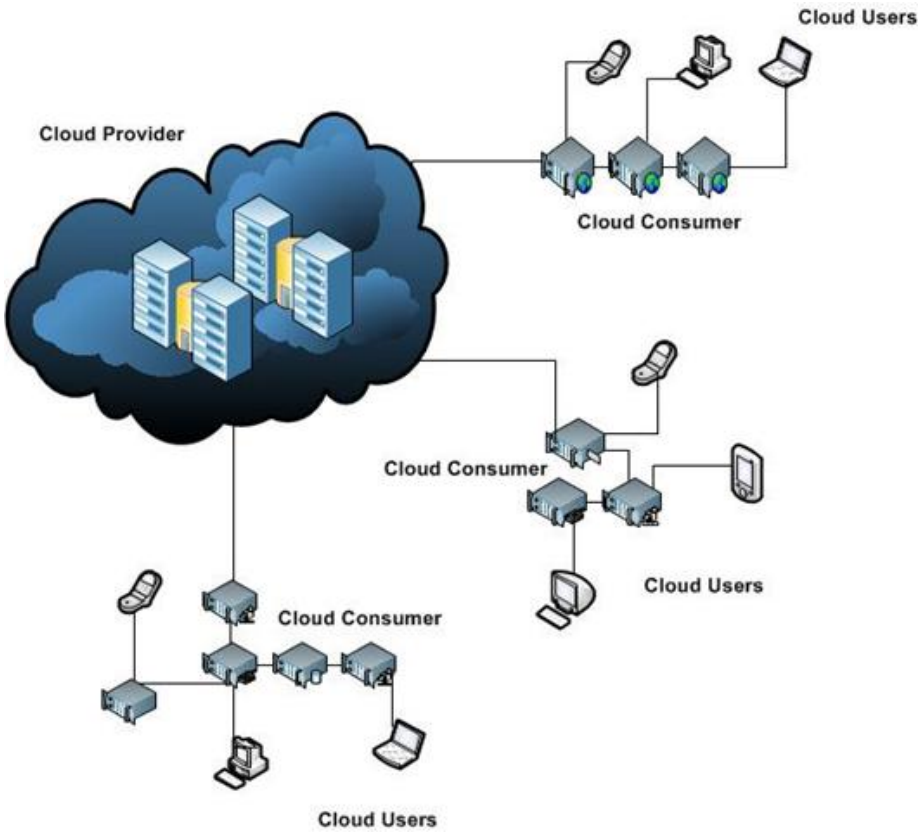


Cloud computing

Il Cloud Computing è stato definito come "un modello per abilitare in modo conveniente l'accesso a servizi funzionali attraverso insiemi di risorse condivise di calcolo che permettono di predisporre e fornire rapidamente applicazioni con un minimo sforzo di gestione e configurazione".

L'ecosistema del cloud computing prevede i seguenti attori:

- **Cloud Providers.**
  forniscono la piattaforma hardware e l'infrastruttura di servizi e software di base
- **Cloud Consumers.**
  realizzano applicazioni e servizi accessibili ed utilizzabili via cluod
- **Cloud Users.**
  utilizzano le applicazioni ed i servizi forniti dai Consumer per realizzare le proprie attività di business



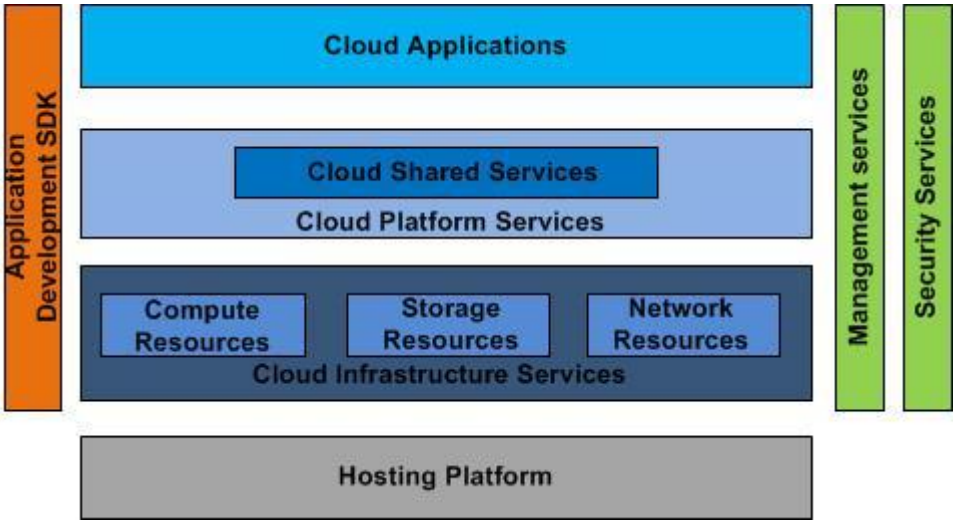
Cloud Characteristics

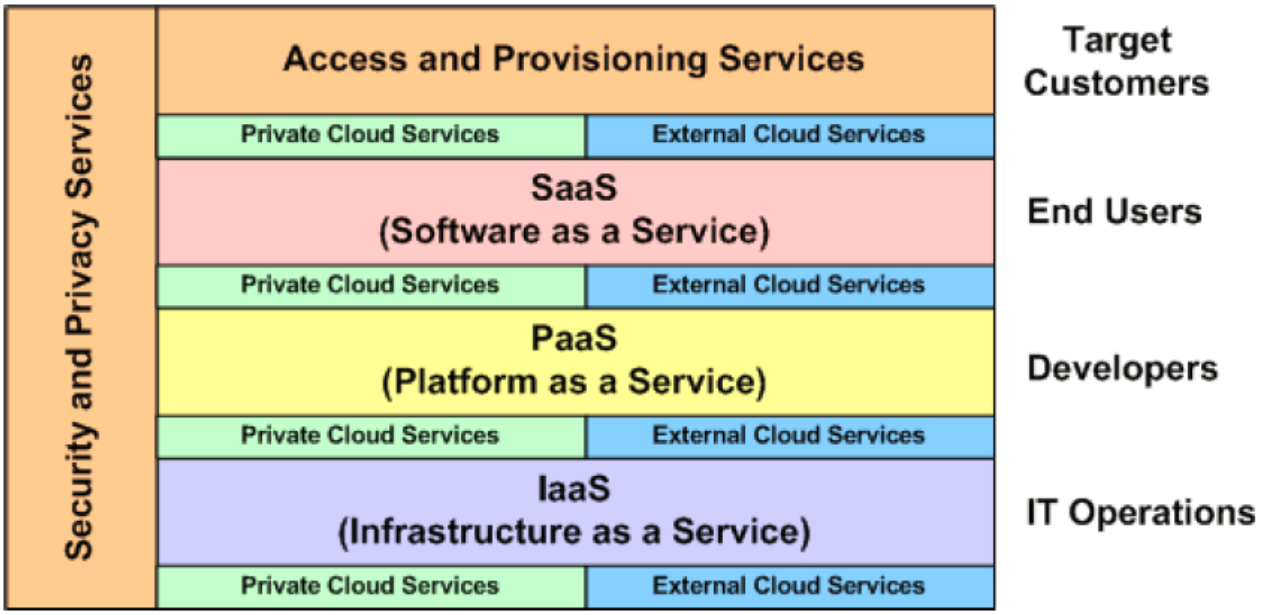
On demand-self-service is one of the cloud characteristics. It enables a consumer to provision computing resources as needed automatically, without intervention on the part of

the cloud solution provider. From now on we will use the term cloud to mean cloud computing.

Also a **broad network access** allows for computing resources to be available through standard mechanisms such as HTTP and SOAP protocols. This is to enable access by a variety of clients such as laptops, mobile phones and other devices.

Another important characteristic is **resource pooling**. The cloud resources are pooled to serve multiple consumers using a multi-tenant model where resources are assigned based on consumer demand.





Progettazione di sistemi di basi di dati

I rischi della progettazione dei sistemi informatici

As proposed by the project sponsor

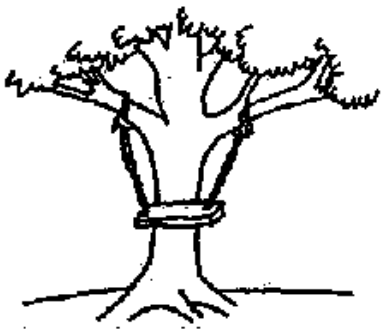
(proposta dal committente)

As specified in the project request

(specificata nei requisiti)

As designed by the senior analyst

(progettata dall'analista)

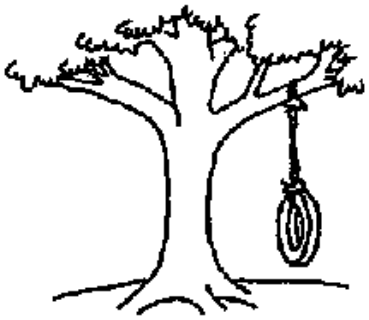


As produced by the programmers

(prodotta dai programmatori)

As installed at the user's site

(installata presso l'utente)



What the user wanted

(ciò che l'utente voleva)

Progettazione di Basi di Dati

Ciclo di vita di un sistema informativo

•**Idea**: identificazione di una necessità da soddisfare e del come soddisfarla

•**Studio di fattibilità**: verifica dell'effettiva realizzabilità dell'idea, valutazione costi e priorità

•**Raccolta e analisi dei requisiti**: raccolta dei desideri e delle necessità a cui va incontro il sistema e studio delle sue proprietà

•**Progettazione**: di dati e funzioni
•**Realizzazione**: codifica delle funzioni progettate e creazione dell'infrastruttura applicativa
•**Validazione e collaudo**: verifica sperimentale del funzionamento del sistema
•**Funzionamento**: vita operativa del sistema
•**Evoluzione**: trasformazione del sistema sulla base delle nuove esigenze (interne od esterne)

Centralità dei dati e necessità della loro modellazione in una fase precoce.

Il dato è più stabile delle funzioni (ma non immutabile)

Ciclo di analisi e progettazione dei dati

- **ANALISI** (CHE COSA)
  - raccolta dei requisiti
  - progettazione concettuale
    - schema concettuale
- **PROGETTAZIONE** (COME)
  - progettazione logica
    - schema logico
  - progettazione fisica
    - schema fisico

Modalità di progettazione e di realizzazione

Progettazione

Strategie di progetto in generale:

- Top-down
- Bottom-up
- Inside-out
- Prototipale (non su libro)
- Ciclica (non su libro)
- eXtreame Programming (XP - non su libro)

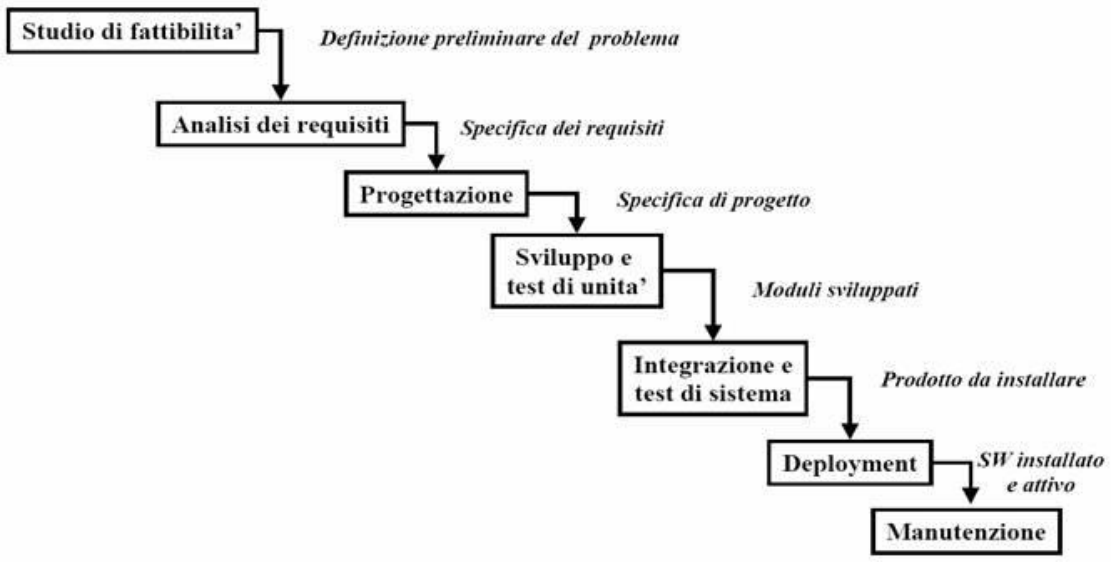
Per la progettazione dei dati non si possono adottare soluzioni prototipali, ne tanto meno XP.

Per i dati la strategia è tipicamente top-down ciclica, cioè fasi di progettazione top-down da un riesame della correttezza e coerenza delle definizioni top sulla base dei risultati down.

Top-down

Modello più anziano, detto anche "a cascata".
Presuppone la possibilità di completare una fase senza più necessità di riesame sulla base delle attività successive, quindi:

- perfetta conoscenza del dominio
- stabilità dei requisiti
- nessun effetto retroattivo tra dettaglio e generale (il dettaglio non nasconde nulla che non sia già definito a livello generale)



Bottom-up

Dopo una fase di “break-down” in cui il sistema viene spezzettato in elementi sempre più piccoli non ulteriormente scomponibili, si analizzano gli atomi per poi procedere ad aggregazioni successive fino ad ottenere il modello completo del sistema.

Aggregando i sistemi parziali si possono individuare:

- T1 - oggetti che hanno tutte le caratteristiche in comune (unificazione di entità)
- T2 - oggetti di sotto-sistemi diversi che hanno un legame (relazione)
- T3 - oggetti di sotto-sistemi diversi che sono casi particolari di un concetto più generale (generalizzazione)
- T4 e T5 - serie di attributi ricorrenti nei vari sotto-sistemi che possono essere aggregati in un'unica entità o relazione

E' un approccio da chimica-molecolare, che presuppone:

- la possibilità d'individuare subito tutte le componenti del sistema
- che il totale non abbia proprietà che trascendono la somma delle parti (problema della vita: come la si spiega partendo dalla sola chimica molecolare ?)

E' una ottima strategia per affrontare sistemi di grossissime dimensioni in cui un singolo analista non può essere in grado di dominare ogni dettaglio, ma l'integrazione dei modelli è sempre molto complessa ed è soggetta alla capacità di mediazione e di contrattazione tra i vari analisti.

Inside-out

Si procede dall'analisi completa di una porzione del sistema complessivo, per poi aggiungere altre parti da analizzare integrandole con quanto già modellato.

Adatta a sistemi che hanno un “core” (nucleo) molto ben definito e coerente, intorno al quale ruotano una serie di altri elementi lascamente (non strettamente) correlati al nucleo.

Ad esempio la progettazione di un intero sistema ferroviario si focalizza prima di tutto sul treno e le rotaie, poi sulle stazioni e le altre infrastrutture.

Completata l'analisi si procede poi all'implementazione del sistema completo.

Non facilita una visione globale ed è poco astratta.

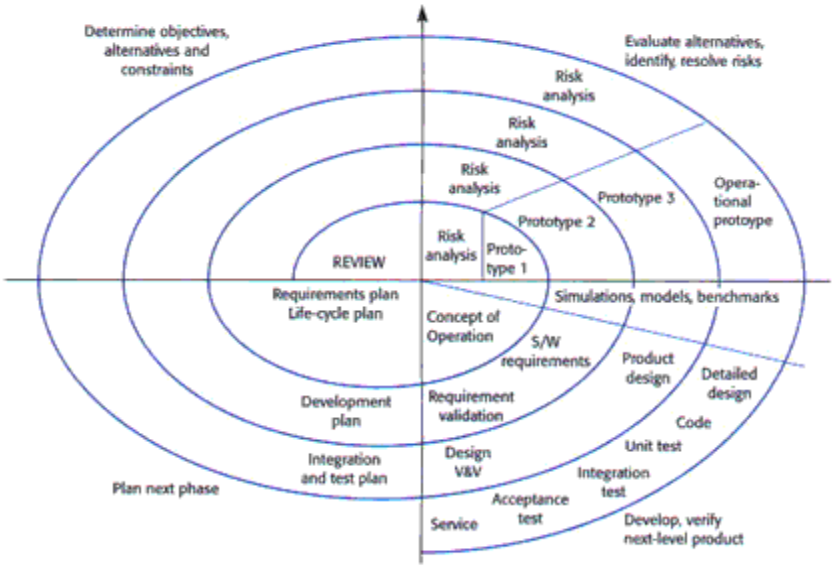
Prototipale (non su libro)

Quando il dominio non è noto, un approccio possibile è quello prototipale, che prevede la creazione appunto di un prototipo sul quale verificare i concetti di base e discutere le caratteristiche del sistema con l'utente, e poi sulla base di quanto raccolto procedere con una strategia classica di implementazione.

(Rischio di rilasciare in produzione prototipi fatti con lo scotch...)

Ciclica (non su libro)

Si procede come nell'Inside-out, ma spingendo oltre l'analisi fino all'implementazione completa della parte, per poi fare nuovi cicli di analisi e implementazione di parti sempre più ampie del sistema.



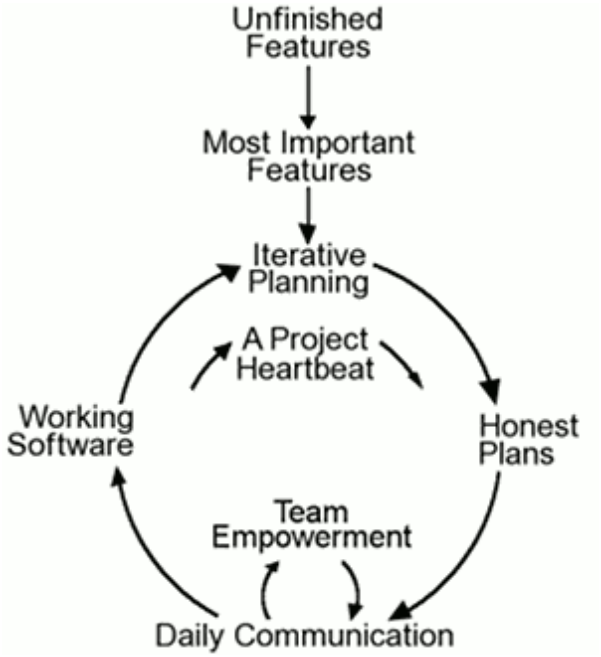
eXtreame Programming (XP - non su libro)

Il sistema è definito attraverso un insieme di test che permettono di stabilire se una funzione o un componente del sistema raggiunge o meno gli obiettivi assegnatigli.

Lo sviluppo, affidato a piccoli team di programmatori, è scomposto in iterazioni brevi con continue fasi di test e verifica.

Tra gli aspetti caratteristici dell'extreme programming vi sono la programmazione a più mani (generalmente in coppia), la verifica continua del programma durante lo sviluppo per mezzo di programmi di test e la frequente reingegnerizzazione del software, di solito in piccoli passi incrementali, senza dover rispettare fasi di sviluppo particolari.

E' un particolare metodo per lo sviluppo del software che coinvolge quanto più possibile il committente, ottenendo in tal modo una elevata reattività alle sue richieste.



Feedback a scala fine

- Pair Programming - significa che tutto il codice viene prodotto da due programmatori che lavorano insieme su una sola workstation.
- Planning Game - è una riunione di pianificazione che avviene una volta per iterazione, tipicamente una volta a settimana.
- Test-driven Development - i test unitari vengono scritti prima di scrivere il codice. Test funzionali e unitari.
- Whole Team - in XP, il "cliente" non è colui che paga il conto, ma la persona che realmente utilizza il sistema. Il cliente deve essere presente e disponibile a verificare (sono consigliate riunioni settimanali).

Processo continuo

- Continuous Integration - Integrare continuamente i cambiamenti al codice eviterà ritardi più avanti nel ciclo del progetto, causati da problemi d'integrazione.
- Refactoring o Design Improvement - riscrivere il codice senza alterarne le funzionalità esterne, cambiando l'architettura, in modo da renderlo più semplice e generico.
- Small Releases - consegna del software avviene tramite frequenti rilasci di funzionalità che creano del valore concreto.

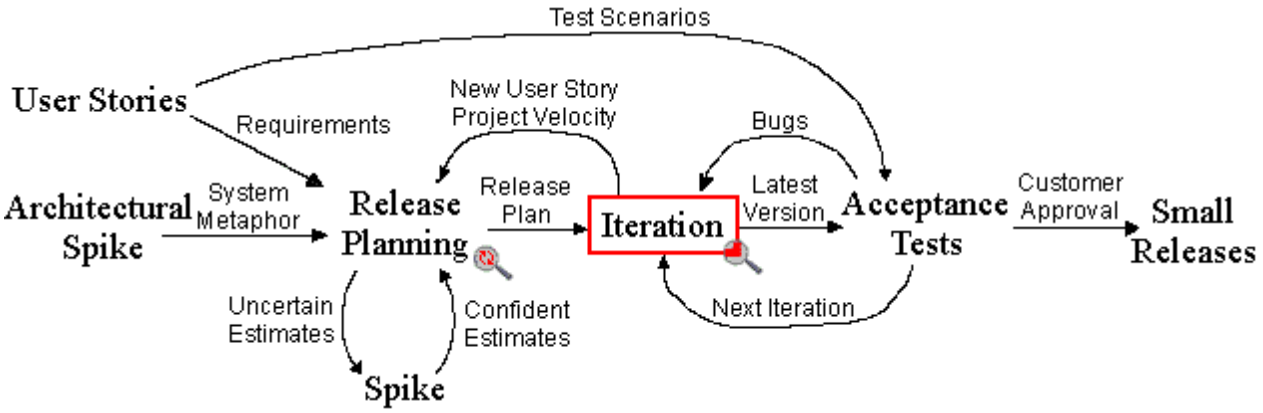
Comprensione condivisa

- Coding Standards - Scegliere ed utilizzare un preciso standard di scrittura del codice. Questo rappresenta un insieme di regole concordate, che l'intero team di sviluppo accetta di rispettare nel corso del progetto.

- Collective Code Ownership - significa che ognuno è responsabile di tutto il codice; quindi contribuisce alla stesura chiunque sia coinvolto nel progetto.
- Simple Design - i programmatori dovrebbero utilizzare un approccio del tipo "semplice è meglio" alla progettazione software. Progettare al minimo e con il cliente.
- System Metaphor - descrivere il sistema con una metafora, anche per la descrizione formale. Questa può essere considerata come una storia che ognuno - clienti, programmtori, e manager - può raccontare circa il funzionamento del sistema.

Benessere dei programmatori

- Sustainable Pace - il concetto è che i programmatori o gli sviluppatori software non dovrebbero lavorare più di 40 ore di lavoro settimanali.



Considerazioni sui costi e sugli aspetti di progettazione, realizzazione e produzione per sistemi: manifatturieri, informatici, di definizione dei dati.

- *Prodotti manufatturieri: peso preponderante sulla fase di produzione di serie (anche se aumentano sempre più i costi di progettazione)*
- *Prodotti informatici: peso quasi paritetico tra progettazione e produzione del primo singolo manufatto (costo di produzione di serie asintoticamente nullo)*
- *Modello dati: peso quasi completamente sulla parte progettuale (il processo di produzione è quasi sempre automatizzabile con strumenti CASE e comunque il prodotto è il risultato della pregettazione)*