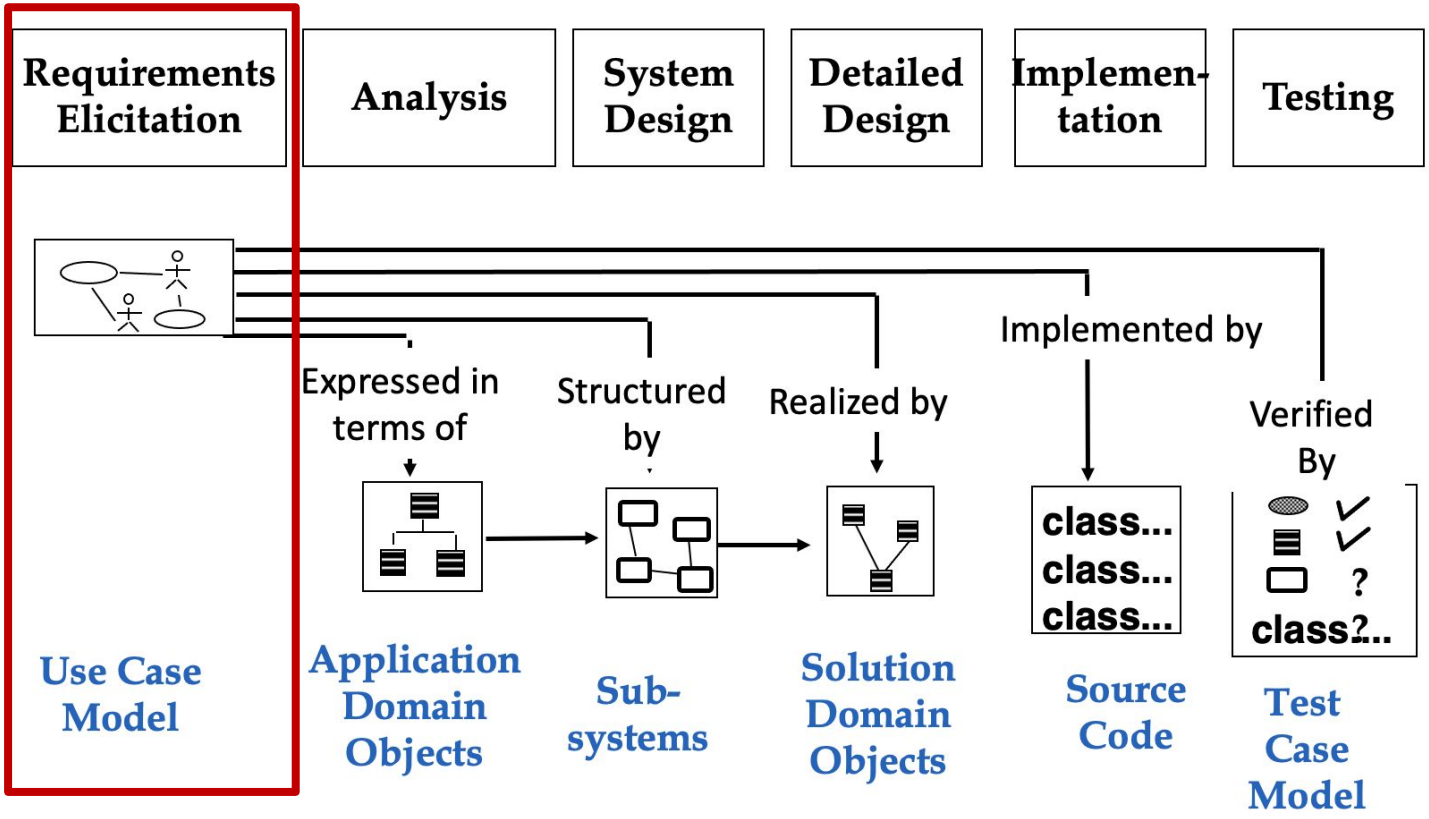
12. Analisi dei requisiti

Prof. Paola Barra
a.a. 2023/2024

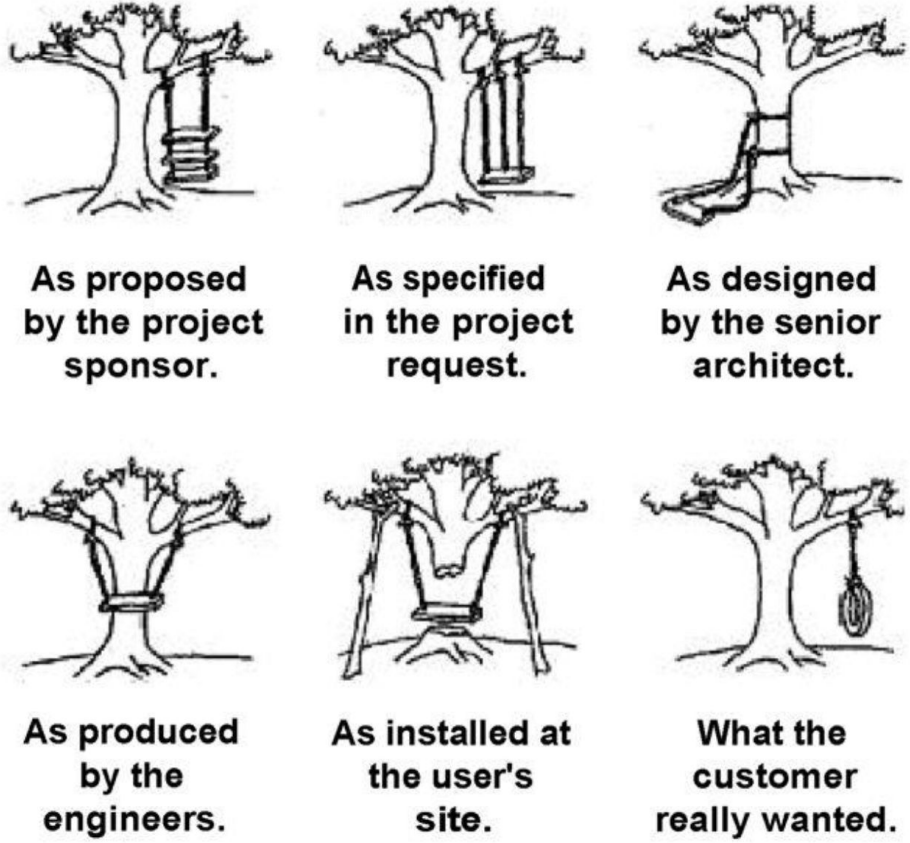
Attività del ciclo di vita del software



La gestione dei requisiti sono l’anello debole dello sviluppo del software

“L’avete fatta l’analisi dei requisiti?”... ma sì certo ci mancherebbe

Un articolo dell’aprile 1998 (Cutter IT journal) documenta la crescita esponenziale delle cause legali tra clienti e fornitori del sistema software, incentrate il più delle volte sulla poca chiarezza dei requisiti specificati a livello contrattuale, e sulle possibilità di una loro interpretazione divergente da parte delle diverse parti in causa.



Perchè i progetti falliscono?

Un celebre studio effettualto dallo Standish Group nel 1994, su un campione di 8000 progetti riporta risultati sconfortanti.

16% successi

53% fallimenti parziali

31% fallimenti completi

I figura ci sono i primi 8 fattori dei fallimenti, 5 dei quali riguardano i requisiti, 3 a problematiche di gestione e nessuno a problemi tecnici.

perché i progetti falliscono

Requisiti incompleti	13,1%
Mancato coinvolgimento utente	12,4%
Mancanza risorse	10,6%
Attese irrealistiche	9,9%
Mancanza supporto direzione	9,3%
Cambiamento requisiti	8,7%
Mancanza pianificazione	8,1%
Non serviva più	7,5%

studio Standish Group 1994 (su 8.000 progetti)

Software Crisis

Il prestigioso mensile "Scientific American" gridava in copertina, pochi anni orsono, la "Software Crisis".

Registriamo continui miglioramenti di rapporto costi-prestazioni nel campo dell'hardware, ma il mondo del software non ce la fa a tenere il passo, e a fornire **soluzioni accettabili in termini di contenuti funzionali**, di livello di **affidabilità e prestazioni**, di **rispetto delle date di consegna** e dei **costi** da sostenere per lo sviluppo. L'elemento critico, più ancora che lo sviluppo di nuove applicazioni, è la cosiddetta **manutenzione evolutiva**. Le esigenze del business cambiano (e crescono) con un ritmo troppo accelerato, in questi periodi di concorrenza e continua innovazione, perché si riesca a far evolvere i sistemi "legacy" esistenti in tempo utile per supportare i nuovi prodotti o i nuovi servizi: con il risultato che l'informatica viene vista come una palla al piede, piuttosto che un'opportunità, e con un crescente grado di incomprensioni e risentimenti tra "utenza" aziendale e progettisti.

I requisiti vanno gestiti

Ma cosa si intende esattamente con gestione dei requisiti? Anzi, precisamente, cos'è un requisito? E' una caratteristica del sistema, richiesta al progettista dal committente (o da un altro interlocutore interessato), come necessaria per raggiungere i propri obiettivi.

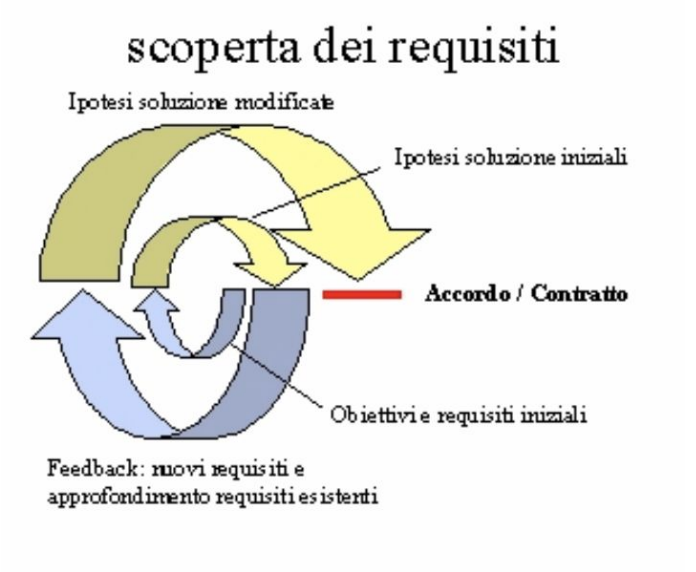
Formulando i requisiti, il committente esprime una serie di vincoli, chiarificando il modo in cui gli obiettivi dovranno essere soddisfatti dal sistema. A sua volta il progettista, dopo aver analizzato i requisiti ricevuti, può formulare una o più ipotesi di soluzione, tra loro diverse per caratteristiche, costi e tempi di realizzazione, ma comunque in grado di rispondere, in tutto o in parte, ai requisiti espressi. Tra le soluzioni proposte, il committente sceglierà quella migliore (dal suo punto di vista) in termini di rapporto tra costi e benefici, e stipulerà un accordo (o un contratto) con i progettisti affinché la realizzino.

Una volta realizzato il sistema, la conformità ai requisiti concordati costituirà il criterio per l'accettazione del prodotto da parte del committente (ed è qui che il più delle volte emergono i conflitti di interpretazione che incrinano i rapporti tra le parti in causa).

Come nascono i requisiti?

In una situazione ideale, il committente comunica i requisiti alla partenza del progetto, e il compito dei progettisti è soltanto quello di acquisirli e di comprenderli. Nel mondo reale, invece, la **scoperta** dei requisiti è un'attività faticosa, e comporta una serie ripetuta di interazioni e discussioni tra gli attori coinvolti.

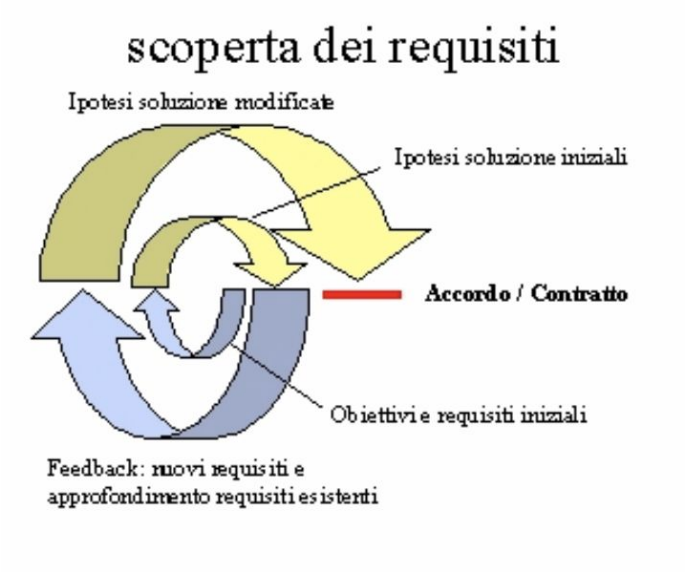
Il committente ha chiari i propri obiettivi di business, ma, ovviamente, non è quasi mai in grado di trasmettere ai progettisti un elenco di requisiti completo e dettagliato sino al punto da costituire un punto di partenza sufficiente per la progettazione del sistema. Il compito del progettista diventa quindi quello di aiutare il committente (e gli altri interlocutori interessati al sistema) a chiarire progressivamente tutti gli aspetti del problema, attraverso **interviste**, **analisi degli scenari concreti di operatività**, evidenziazione dei rischi, e soprattutto proposte preliminari di possibili soluzioni. L'obiettivo in questa fase è stimolare il committente, rendendolo consapevole delle diverse possibilità di soluzione, in modo da permettergli di prendere posizione chiarificando i requisiti espressi inizialmente, ed eventualmente aggiungendone altri. In pratica, permettergli di esprimere **feedback** sulle ipotesi prospettate dai progettisti.



Come nascono i requisiti?

Un fattore importante in questo contesto è la messa in luce dei conflitti tra requisiti.

Il conflitto tra requisiti è sempre presente, salvo casi eccezionali. Lo è quando i requisiti vengono espressi da un unico committente, che desidererebbe un sistema il più completo ed efficiente possibile, rilasciato nel più breve tempo possibile, con i costi minori possibili. E naturalmente lo è ancora di più quando i requisiti possono essere espressi da vari interlocutori (altre funzioni aziendali interessate, utilizzatori finali). Il compito del progettista in questo ambito è quello di portare allo scoperto il conflitto al più presto, permettendo al committente di prenderne coscienza e di decidere nel merito.

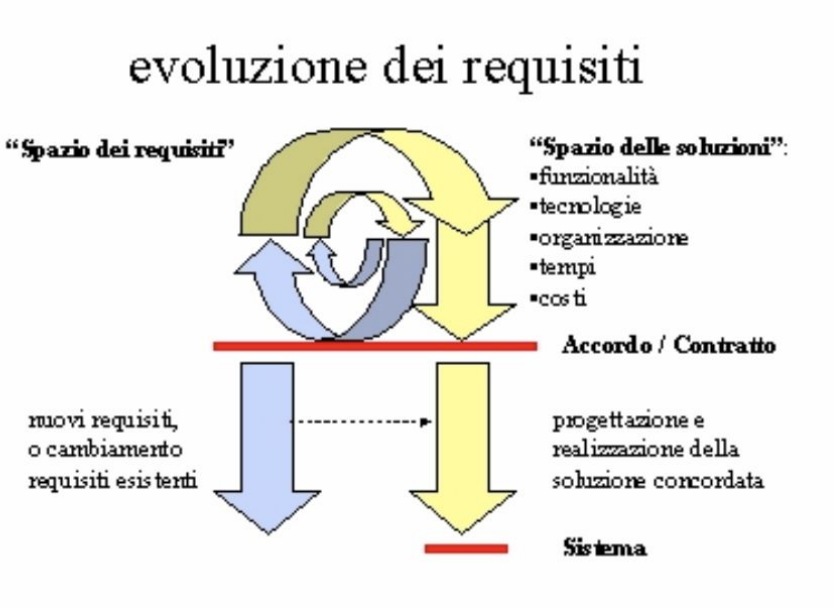


Quando nascono i requisiti?

Nel mondo ideale, i requisiti vengono definiti nella fase iniziale del progetto, permettono di definire un accordo, dopodiché non vengono più cambiati (il famoso "congelamento delle specifiche").

E' questa la situazione ipotizzata dal processo "a cascata" (waterfall), che è alla base della maggior parte delle metodologie di sviluppo ufficialmente presenti, ma raramente utilizzate davvero, nei settori informatici delle aziende.

Nel mondo reale, i requisiti possono nascere, o cambiare, anche dopo il raggiungimento dell'accordo (o la stipula del contratto) tra committenti e progettisti, mentre sono in corso le attività di realizzazione. Innovazioni legislative, cambiamenti negli scenari di mercato o nelle strategie aziendali, nuove opportunità: sono diversi i fattori che possono determinare un cambiamento di requisiti, anche nei casi in cui l'indagine iniziale sia stata veramente esaustiva.



Evoluzione dei requisiti

A fronte di un cambiamento di requisiti in corso d'opera, due sono le strade possibili. La prima, meno dolorosa ma non sempre praticabile, consiste nel rimandare la presa in carico del nuovo requisito a una release successiva. La seconda, nel rinegoziare i termini dell'accordo precedentemente raggiunto, rivedendone i contenuti e/o i costi e/o i tempi definiti.

Una cosa, comunque, è certa: i requisiti possono nascere o cambiare in ogni momento, dalle fasi iniziali di un progetto a quelle realizzative, e certamente dopo che il sistema è stato rilasciato. E' quindi necessario che il presidio sull'evoluzione dei requisiti venga effettuato in modo continuativo, durante l'intero ciclo di vita del sistema.

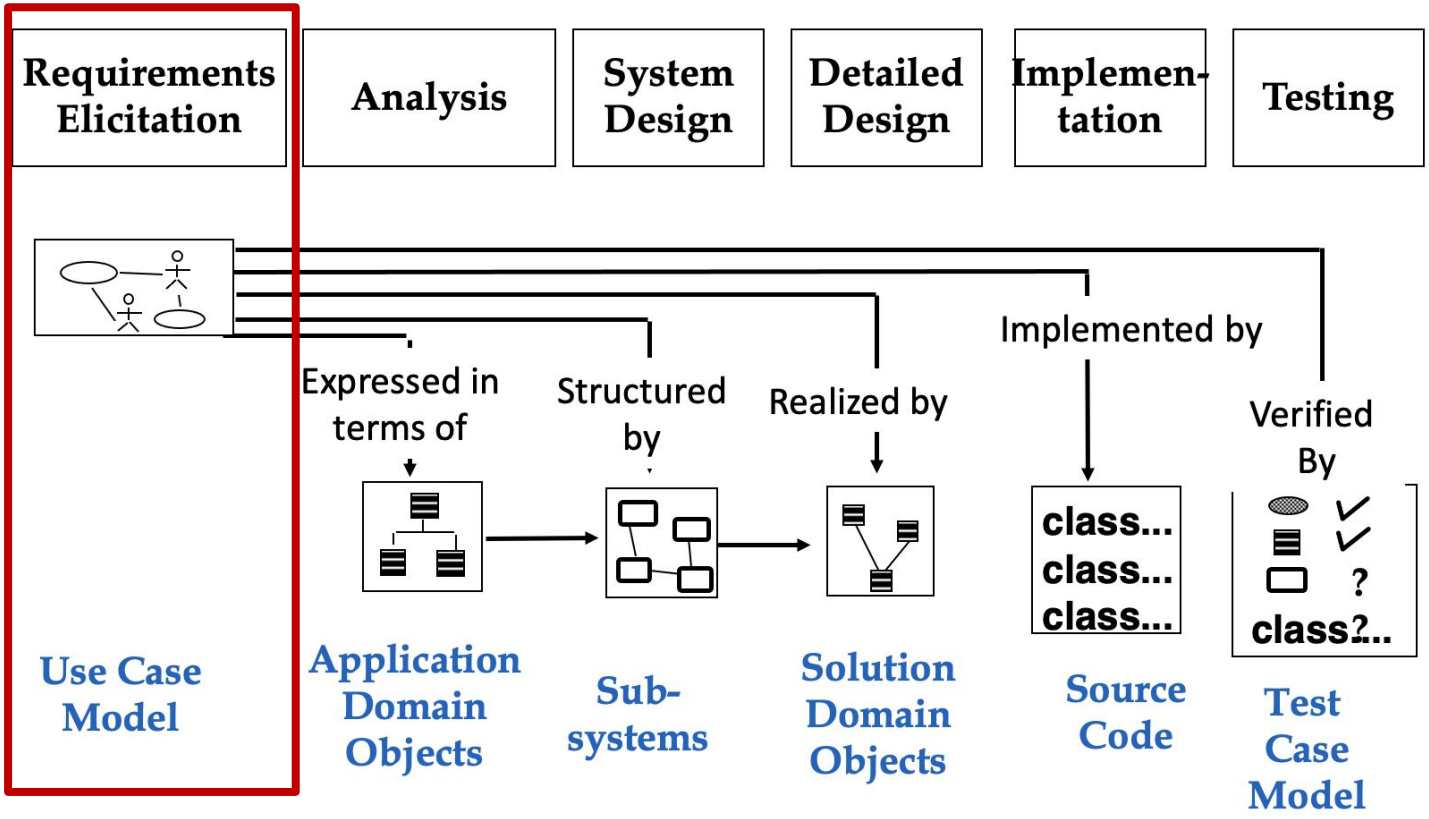
Il documento di specifica dei requisiti

Come vanno specificati i requisiti, in modo da costituire la base per un accordo effettivo? Le pratiche correnti di gestione dei requisiti raccomandano, prima di tutto, di raccogliere e gestire i requisiti (le richieste) in un ambito distinto da quello delle proposte di soluzione avanzate dai progettisti. E per la gestione dei requisiti sono nati, negli ultimi anni, numerosi strumenti dedicati (requirements management tools).

L'accordo vero e proprio dovrà essere raggiunto su una specifica soluzione, che risponda in modo adeguato ai requisiti espressi dal committente e dagli altri interlocutori interessati, e le cui caratteristiche siano comprensibili per tutte le parti in causa, senza ambiguità. In particolare, dovrà essere chiaro sia per il committente che per i progettisti quali siano i requisiti soddisfatti dalla soluzione concordata, e quali invece siano stati eliminati (a causa di conflitto) o rimandati a data da destinarsi (per contenere tempi e costi della soluzione concordata).

Scoperta dei requisiti

Attività del ciclo di vita del software



Studio di fattibilità

- Fase preliminare per stabilire l'opportunità o meno di realizzare il software
- Si basa su
  - Una descrizione sommaria del sistema software e delle necessità utente
  - Un'analisi per capire se ha senso realizzare il prodotto:
    - Cosa esiste già
    - Il prodotto venderà?
  - Un'analisi tecnica per capire se è realizzabile

Studio di fattibilità

- Si basa sulla valutazione dei costi e dei benefici di una possibile attività di produzione
- Fattibilità tecnologica
  - Strumenti per la realizzazione (software, librerie, …)
  - Soluzioni algoritmiche e architetturali
  - Hardware
  - Processo (prototipazione, progetti esplorativi, ricerca..)
- Aspetti economici e di mercato
  - Confronto tra il mercato attuale e quello futuro
  - Costo della produzione, redditività dell'investimento

Attività di analisi dei requisiti

- Studiare e definire il problema da risolvere
  - Per capire cosa deve essere realizzato
  - Per documentare cosa deve essere realizzato
  - Per negoziazioni committente/fornitore

cosa

non come

Quanto costa correggere un errore nei requisiti? [Boehm]

FASE	COSTO
Analisi Requisiti	1
Progettazione	5
Codifica	10
Test di unità	20
Test di accettazione	50
Operazione	100

Prodotto dell'attività di analisi dei requisiti

- Documento e/o modello analitico di
  - Descrizione del dominio
  - Descrizione dei requisiti
- Opzionalmente anche
  - Manuale utente
  - Casi di test
- (spesso in parallelo)

DOMINIO

- Dominio
  - Il campo di applicazione del prodotto

- Prima ancora di incontrare i committenti il team di analisti deve acquisire conoscenza del dominio di applicazione
  - Per poter porre le domande giuste domande: pensate al software per transazioni finanziarie
  - Per capire la differenza sottile che può esserci tra termini solo apparentemente sinonimi: pensate a supporto, montante, sostegno, trave in ingegneria civile

DOMINIO

- Per acquisire conoscenza e definire il dominio:
  - si costruisce un glossario
    - Una collezione di definizioni di termini rilevanti in un dominio specifico
    - il team di analisti lo costruisce mentre studia il dominio e poi si arricchisce via via che si incontrano nuovi termini
    - Può essere riusato in progetti successivi nello stesso dominio
  - Si definiscono un modello statico e un modello dinamico

- La conoscenza del dominio evolve dopo incontro con committenti, utenti, ecc

Requisito

- Requisito
  - Una condizione o una capacità necessaria a un utente per risolvere un problema [..]
  - Una condizione (capacità) che deve essere soddisfatta (posseduta) [..] da un sistema [..] per soddisfare un contratto [..]

[Glossario IEEE]

- In altre parole:
  - Una proprietà che deve essere garantita dal sistema per soddisfare una necessità di un utente
    - Funzionalità, qualità, ...

Come acquisire informazioni sul dominio? Interviste

- I membri del team di analisti incontrano i membri dell'organizzazione del committente
- Si procede con delle interviste che possono essere
  - strutturate
  - non strutturate
- Entrambe presentano vantaggi e rischi
- Difficile condurre una buona intervista

Tecniche per acquisire conoscenza su dominio e requisiti

- Questionari a risposte multiple a tutti i membri rilevanti dell'organizzazione del committente
- Costruzione di prototipi
- Osservazione di futuri utenti al lavoro
- Studio di documenti

Tecniche per scoprire i requisiti

- Colmare il gap tra utente e sviluppatore
  - **Questionari**
    - Chiedere all'utente una lista di domande predefinite
    - Questionari a risposte multiple a tutti i membri rilevanti dell'organizzazione del committente
  - **Analisi dei compiti**
    - Osservare gli utenti nel loro ambiente di lavoro
  - **Scenari**
    - Descrivono l'uso del sistema come una serie di interazioni tra uno specifico utente e il sistema
  - **Casi d'uso**
    - Astrazioni che descrivono una classe di scenari

ALTRE TECNICHE

- Costruire dei prototipi
- Osservazione di futuri utenti al lavoro
- Studio di documenti

Obiettivo: scoperta dei requisiti

Focalizzata sulla descrizione dello **scopo** del sistema

Il cliente, gli sviluppatori e gli utenti **identificano un'area del problema** e **definiscono un sistema** che lo affronta

La suddetta definizione è chiamata **specifica dei requisiti** e funge da contratto tra il cliente e gli sviluppatori

La **specifica dei requisiti** è strutturata e formalizzata durante l'**analisi**

Definizione del problema

Descrive i requisiti, la decomposizione in sottosistemi e l'infrastruttura di comunicazione

Descrive la situazione corrente, le funzionalità da supportare e l'ambiente in cui il sistema sarà distribuito

Contiene informazioni per una comprensione reciproca (cliente-sviluppatore) del problema che il sistema deve risolvere

Non è una specifica né precisa né completa ma un sommario dei desiderata del cliente

Requisiti

- I requisiti sono descrizioni testuali delle funzionalità desiderate del sistema e delle loro proprietà. Divise in due categorie principali secondo l'acronimo FURPS
  - Functionality, Usability, Reliability (affidabilità), Performance, Supportability

- Requisiti Funzionali (F):
  - Una singola funzione del sistema
  - «*Uno studente può vedere tutti i corsi del semestre corrente nella propria lista di esami fondamentali e a scelta*»

- Requisiti non funzionali (URPS):
  - Aspetti del sistema non legati direttamente al suo comportamento funzionale
  - «*aspetto dell'interfaccia, tempo di risposta, problemi di sicurezza*»

Primo passo: identificazione del sistema

- E’ necessario rispondere a due domande:
  1. Come possiamo identificare lo scopo del sistema?
     - Quali sono i requisiti? Quali sono i vincoli?
  2. Cosa si trova all’interno del sistema e cosa al di fuori?
- La risposta è fornita durante la scoperta dei requisiti e l’analisi
- **Scoperta dei requisiti**
  - Definizione del sistema in termini compresi dal cliente e/o dagli utenti («specifica dei requisiti»)
- **Analisi**
  - Definizione del sistema nella «terminologia» dallo sviluppatore (specifica tecnica, «Modello di analisi»)
- **Processo dei requisiti** = scoperta dei requisiti + Analisi

Scenari

Scenari

Una descrizione sintetica di un evento o serie di azioni ed eventi

Una descrizione testuale dell'uso del sistema. La descrizione è scritta da un punto di vista dell'utente

Uno scenario può includere testo, video, immagini e racconti

- Solitamente, contiene dettagli del luogo di lavoro, situazioni sociali e vincoli sulle risorse

Ulteriori definizioni

Scenario

- «*Una descrizione narrativa di cosa fanno le persone e le loro esperienze nel momento in cui cercano di utilizzare i computer e le applicazioni*»
  - [M. Carroll, Scenario-Based Design, Wiley, 1995]

- Una descrizione informale, concreta e circoscritta di una singola caratteristica del sistema usato da un singolo attore
  - Lo scenario diventa la base d'interazione per un nuovo progetto e per meglio capire un nuovo progetto

Gli scenari nel processo software

- Gli scenari possono essere usati in modi diversi durante il ciclo di vita del software
  - **Scoperta dei requisiti**
    - Scenario «as-is», scenari visionari
  - **Test di accettazione del cliente**
    - Scenario di valutazione
  - **Distribuzione del sistema**
    - Scenari di training
- L'uso degli scenari dovrebbe essere iterativo
  - Ogni scenario dovrebbe essere considerato come un documento di lavoro da aggiornare e riorganizzare quando i requisiti, i criteri di accettazione del cliente o le situazioni di distribuzione cambiano

Sviluppo basato su scenari

Focalizzato su descrizioni complete e istanze particolari e non idee generiche ed astratte

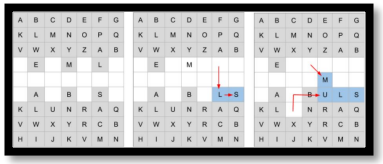
E’ guidato dal lavoro e non dalla tecnologia

E’ sempre aperto (aggiornabile), non cerca di essere completo

E’ informale, non formale e rigoroso

Riguarda risultati previsti (immaginati), non risultati specifici

Tipi di scenari



Scenario As-is

- Descrive una situazione corrente. Usato comunemente in progetti di reingegnerizzazione. L'utente descrive il sistema
  - Esempio: descrizione del Gioco Letter-Chess

Scenario Visionario

- Descrive un sistema futuro
  - Esempio: l'home computer del futuro
- Spesso usato nei progetti di ingegneria greenfield o dell'interfaccia
  - Esempio: descrizione del gioco interattivo basato sul web tic-tac-toe

Scenario di valutazione

- Descrizione di compito dell'utente rispetto al quale deve essere valutato il sistema

Scenario di training

- Una descrizione delle istruzioni passo-passo che guidano un utente novizio nell'uso del sistem

Come troviamo gli scenari?

- **Non aspettatevi che il cliente sia generoso con le parole se il sistema non esiste**
  - Il cliente capisce il dominio dell'applicazione (dominio del problema), non il dominio della soluzione
- **Non attendetevi informazioni anche se il sistema esiste**
  - «ciò che è ovvio non ha bisogno di essere detto»
- **Impegnarsi con un approccio dialettico**
  - Aiutate il cliente a formulare i requisiti
  - Il cliente vi aiuta a capire i requisiti
  - I requisiti evolvono mentre si sviluppano gli scenari

Euristiche per trovare gli scenari

- **Ponete a voi stessi e al cliente le seguenti domande**:
  - Quali sono i compiti principali che il sistema deve eseguire?
  - Che dati l'attore dovrà creare, memorizzare, modificare, eliminare o aggiungere nel sistema?
  - Quali modifiche esterne l'attore deve informare il sistema?
  - Di quali cambiamenti o eventi dovrà essere informato l'attore del sistema?
- Tuttavia, non basatevi solo su domande e questionari
- Insistete sull'osservazione del compito da svolgere se il sistema già esiste (ingegneria dell'interfaccia o reingegnerizzazione)
  - Chiedete di parlare con l'utente finale, non solo il cliente
  - Aspettatevi resistenze e cercate di superarle

Esempio di scenario: warehouse on Fire

- Bob, guidando lungo la strada principale nella sua patrol, osserva del fumo proveniente da un magazzino. Il suo partner, Alice, notifica l'emergenza dall'auto
- Alice immette l'indirizzo dell'edificio nel suo portatile, una breve descrizione della sua ubicazione (angolo nord-ovest) ed un livello di emergenza.
- Alice conferma l'input ed attende l'accettazione.
- John, il Dispatcher, è allertato per l'emergenza da un beep dalla sua stazione di lavoro. Rivede le informazioni sottomesse da Alice e accetta il rapporto. Alloca un'unità dei vigili del fuoco ed invia la stima del tempo di arrivo (ETA) ad Alice.
- Alice riceve l'accettazione e l'ETA

Osservazioni sullo scenario “Warehouse on fire”

- E’ uno scenario concreto
  - Descrive una singola istanza di un rapporto di un incendio
  - Non descrive tutte le possibili situazioni in cui può essere riportato un incidente
- Attori partecipanti
  - Bob, Alice e John

Dopo che gli scenari sono formulati

- Trovare il caso d'uso nello scenario che specifica tutte le istanze di come riportare un incendio
  - Esempio dallo scenario Warehouse on Fire
    - «**Bob … nota del fumo provenire da un magazzino. Il suo partner, Alice, riporta l'emergenza dalla sua auto**»
  - «**Report Emergency**» è un candidato per un caso d'uso
- Descrivere ciascun caso d'uso con maggior dettaglio
  - Attori partecipanti
  - Descrivere le condizioni di entrata
  - Descrivere il flusso di eventi
  - Descrivere le condizioni di uscita
  - Descrivere le eccezioni
  - Descrivere i requisiti non funzionali
- L'insieme di tutti i casi d'uso è la base per il modello funzionale

Scoperta dei requisiti: difficoltà e sfide

- Comunicazione accurata tra il dominio ed il sistema
  - Le persone con background diversi devono collaborare per colmare il gap tra utenti e sviluppatori
    - I clienti e gli utenti hanno una conoscenza del **dominio dell'applicazione**
    - Gli sviluppatori hanno conoscenza del **dominio della soluzione**
- Identificazione di un sistema adatto (definizione dei contorni del sistema)
- Fornire una specifica non ambigua
- Escludere caratteristiche accidentali

Scoperta dei requisiti: difficoltà
Esempio di una specifica ambigua

- Durante un esperimento, fu diretto un raggio laser dalla terra verso uno specchio sullo Space Shuttle Discovery
- Il raggio laser si supponeva dovesse riflettersi indietro verso la cima di una montagna alta 3.055 metri
- L'operatore immise elevazione pari a «3.055»
- Il raggio di luce non toccò mai la cima della montagna

  Quale fu il problema?
- **Il computer interpretò il numero in miglia!**

Scoperta dei requisiti: difficoltà
Esempio di caratteristica accidentale

- **Dalle ultime notizie: il treno della metropolitana di Londra lascia la stazione senza conducente!**
- Cosa è successo?
  - Una porta passeggeri si era bloccata e non si chiuse
  - Il conducente lasciò il suo treno per chiudere la **porta passeggeri**
    - Lasciò la **porta conducente** aperta
    - Si basò sulla specifica che diceva che il treno non si muove se almeno una porta è aperta
  - Quando egli chiuse la porta passeggeri, il treno lasciò la stazione senza di lui
    - La porta del conducente non era considerata una porta passeggeri nel codice sorgente!

Processo dei requisiti

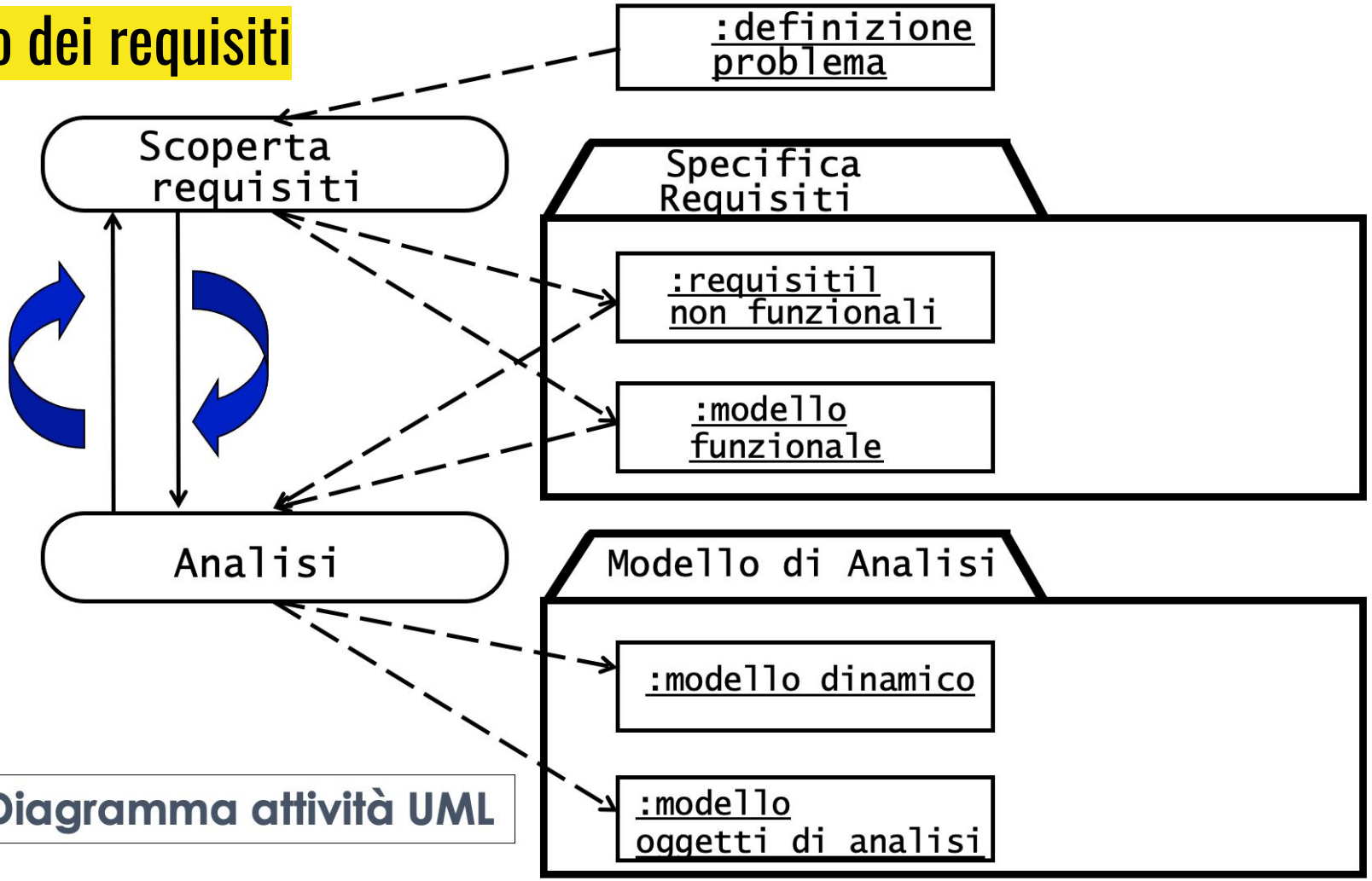


Diagramma attività UML

Specifica dei requisiti vs modello d'analisi

Entrambi sono modelli circoscritti ai requisiti del sistema dal punto di vista dell'utente

La **specifica dei requisiti** usa il linguaggio naturale (derivato dalla definizione del problema)

Il **modello di analisi** usa una notazione formale o semi-formale

Linguaggi di modellazione dei requisiti

- Linguaggio naturale
- Linguaggi grafici: UML, SysML, SA/SD
  - http://hepunx.rl.ac.uk/BFROOT/www/doc/workbook_kiwi/coding/OOvsSASD.html
- Linguaggi di specifica matematica: VDM (Metodo di definizione Vienne), Z, metodi formali
  - http://en.wikipedia.org/wiki/Formal_methods
  - http://en.wikipedia.org/wiki/Vienna_Development_Method
  - http://en.wikipedia.org/wiki/Z_notation