UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

Ingegneria del Software – Requirement Engineering

Prof. Sergio Di Martino

Outline

- Il concetto di Ciclo di Vita del Software
- L'Ingegneria dei Requisiti
- Le diverse Tipologie di Requisiti Software
  - Requisiti Utente vs. Requisiti di Sistema
  - Requisiti Funzionali vs. Requisiti Non Funzionali

Il Ciclo di Vita del Software

Processo

- "*Un processo è un particolare metodo per fare qualcosa costituito da una sequenza di passi che coinvolgono attività, vincoli e risorse*" (Pfleeger)
- "*Processo: una particolare metodologia operativa che nella tecnica definisce le singole operazioni fondamentali per ottenere un prodotto industriale*" (Zingarelli)
- "*Processo software: un metodo per sviluppare del software*" (Sommerville)

Processo/Ciclo di vita del software

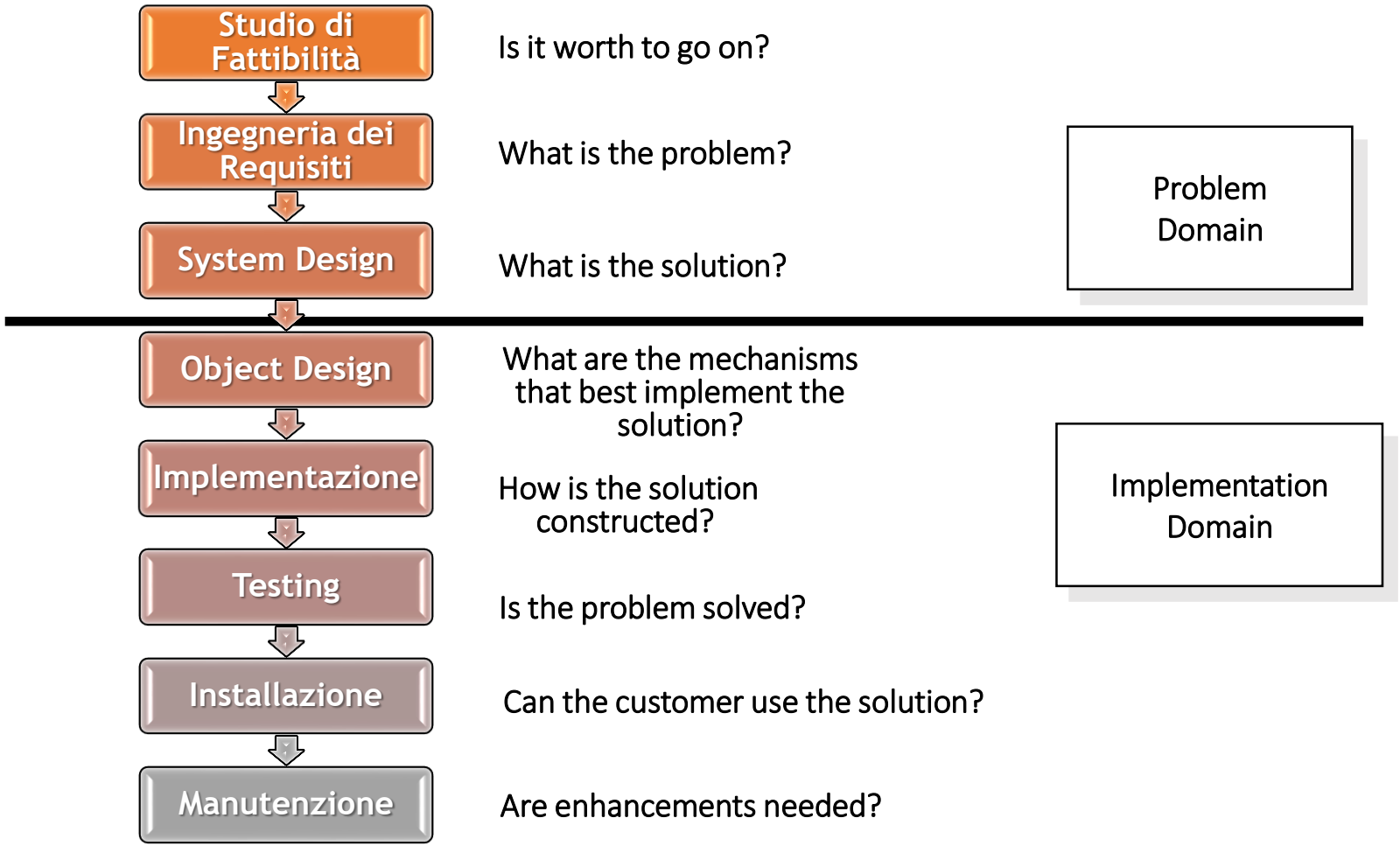
- Insieme organizzato di attività che sovrintendono alla costruzione del software da parte del team di sviluppo utilizzando metodi, tecniche, metodologie e strumenti.
- È suddiviso in fasi, che vanno dalla nascita fino alla dismissione (o morte) del software.
  - Per un parallelo con la biologia, è noto come ***ciclo di vita del software***
- Molti autori usano i termini ***processo*** *[di sviluppo del] software* e ***ciclo di vita*** *del software* come sinonimi.

Processo software: Standard IEEE 610.12-1990

- *«Sofware development process: The process by which user needs are translated into a software product. The process involves translating user needs into software requirements, transforming the software requirements into design, implementing the design in code, testing the code, and sometimes, installing and checking out the software for operational use».*
  - *Note: These activities may overlap or be performed iteratively.*

Macropassi fondamentali del processo software

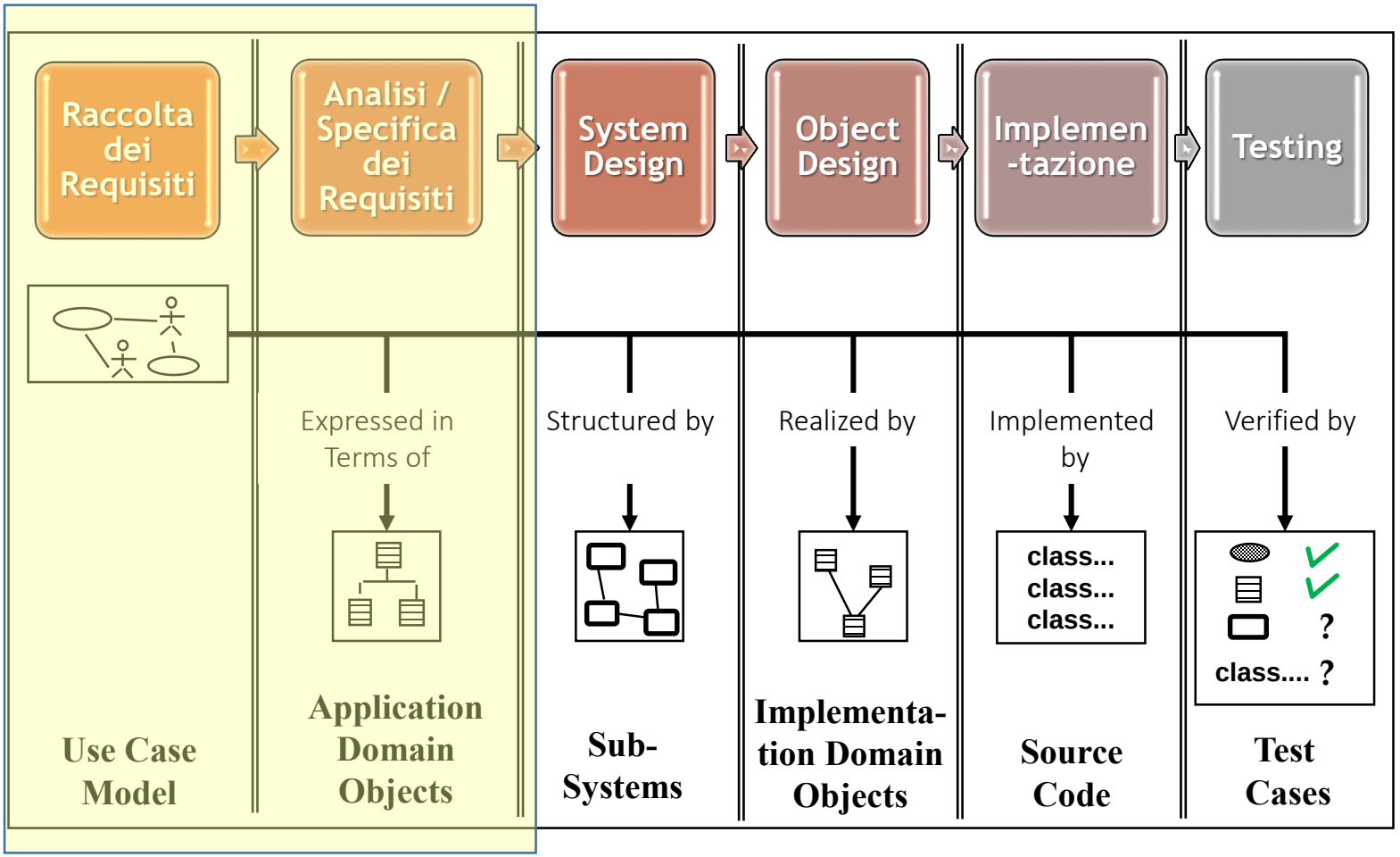
Attività richieste nel processo di sviluppo software



Studio di fattibilità

- Valutazione preliminare di costi e benefici
- Varia a seconda della relazione committente/produttore
- Obiettivo
  - Stabilire se avviare il progetto, individuare le possibili opzioni e le scelte più adeguate, valutare le risorse umane e finanziarie necessarie
  - Si conclude con un'*offerta* al Cliente
- Output: documento di fattibilità
  - definizione preliminare del problema
  - scenari - strategie alternative di soluzione
  - costi, tempi, modalità di sviluppo per ogni alternativa

Ciclo di Vita del Software



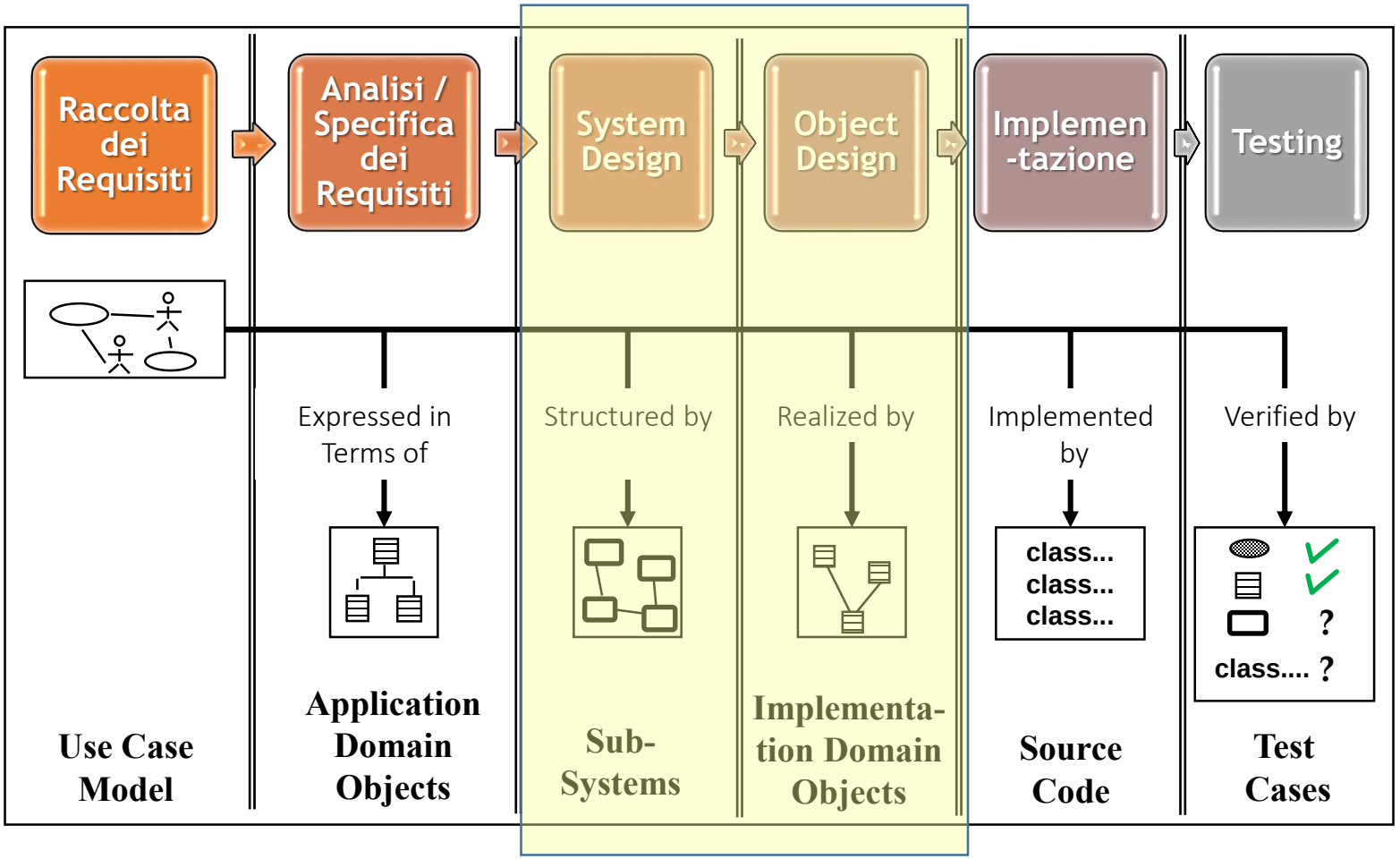
Raccolta/Analisi/Specifica dei requisiti

- Analisi completa dei bisogni dell'utente e dominio del problema
- Coinvolgimento di committente e ingegneri del SW
- Obiettivo
  - Descrivere le caratteristiche di qualità che l'applicazione deve soddisfare

CHE COSA?  **COME?** 

- Output:
  - documento di specifica dei requisiti

Ciclo di Vita del Software

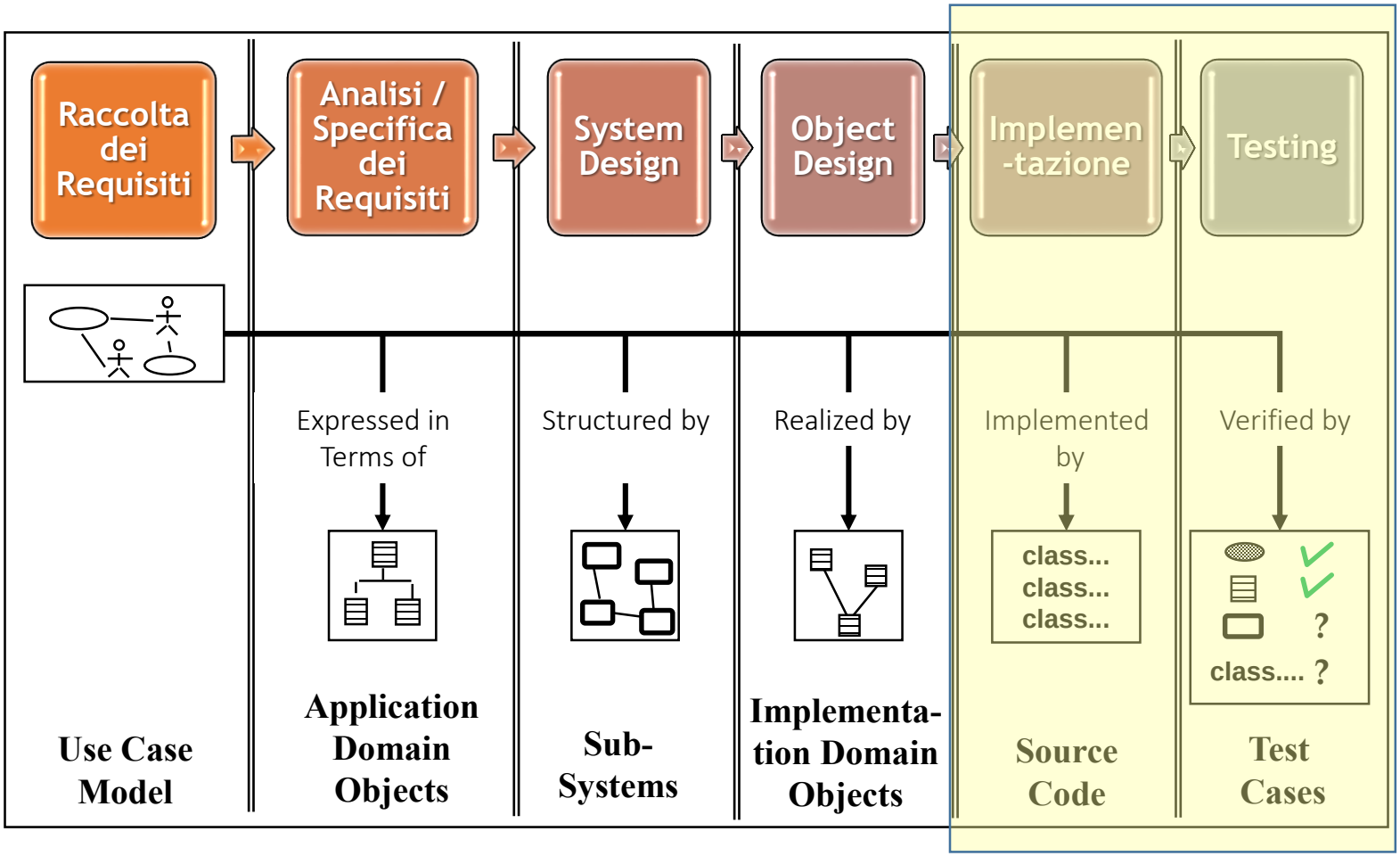


Progettazione
(o System/Object Design)

- Definizione di una struttura opportuna per il SW
- Scomposizione del sistema in componenti e moduli
  - allocazione delle funzionalità ai vari moduli
  - definizione delle relazioni fra i moduli
- Distinzione fra:
  - System Design: struttura modulare complessiva (componenti)
  - Object Design: dettagli interni a ciascuna componente
- Obiettivo

- Output: documento di specifica di progetto
  - possibile l'uso di linguaggi per la progettazione (UML)

Ciclo di Vita del Software



Fasi basse del processo

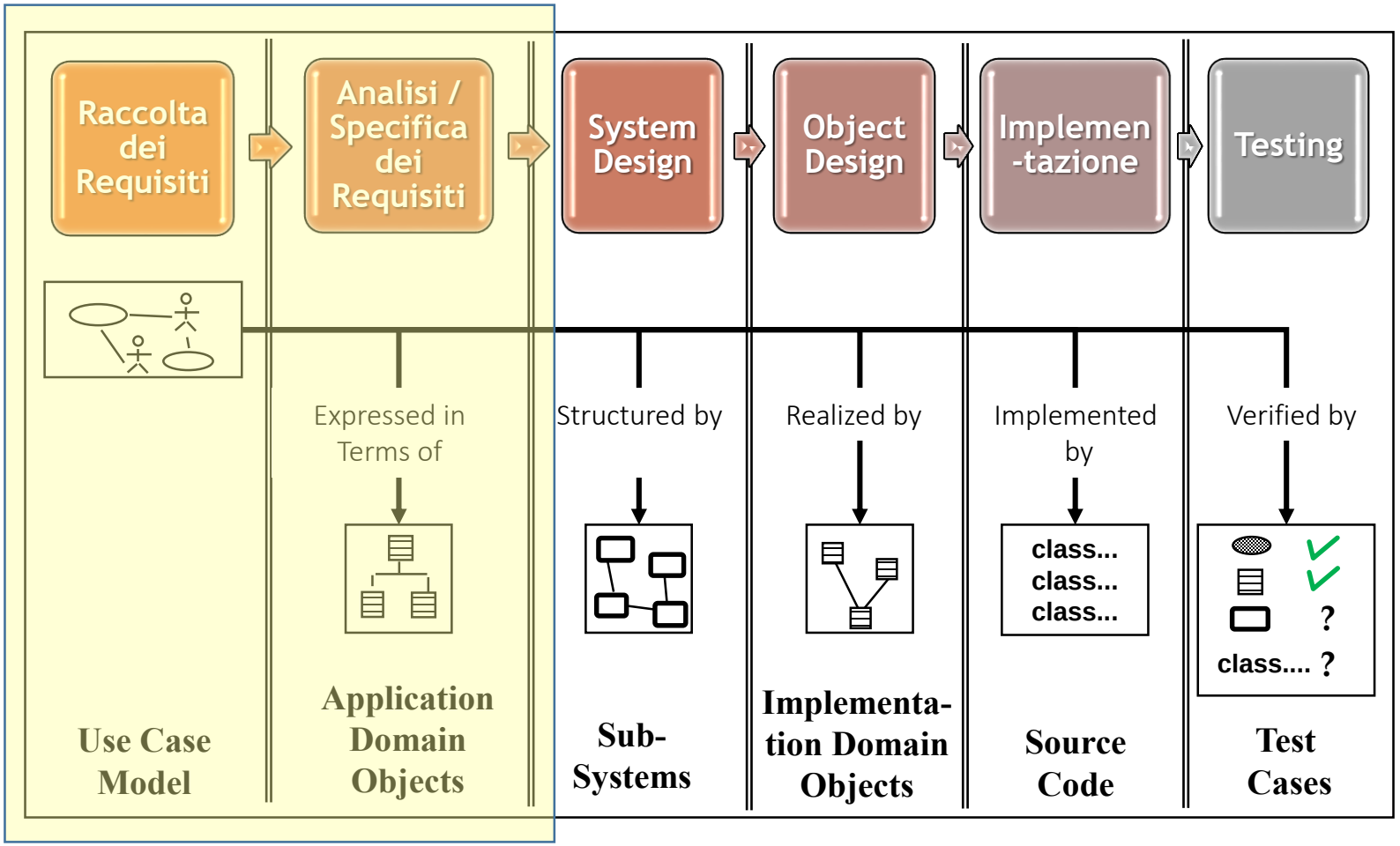
- **Implementazione**: ogni modulo viene codificato nel linguaggio scelto e testato in isolamento
- **Testing**
  - Composizione dei moduli nel sistema globale
  - Verifica del corretto funzionamento del sistema
  - Validazione che il sistema faccia ciò che vuole il cliente
- **Installazione**: distribuzione e gestione del software presso l'utenza
- **Manutenzione**: evoluzione del SW. Segue le esigenze dell'utenza. Comporta ulteriore sviluppo per cui racchiude in sé nuove iterazioni di tutte le precedenti fasi

Problemi nel processo di sviluppo del software

- E’ qualcosa di altamente intellettuale e creativo, basato su giudizi delle persone
  - Non è possibile (almeno con i sistemi attuali) automatizzarlo
- I requisiti sono complessi e ambigui
  - Il cliente non sa bene, dall’inizio, cosa deve fare il software
- I requisiti sono variabili
  - Cambiamenti tecnologici, organizzativi, etc…
- Modifiche frequenti sono difficili da gestire
  - Difficile stimare i costi ed identificare cosa consegnare al cliente

L’Ingegneria dei Requisiti

Ciclo di Vita del Software



Requisiti

- Def. (IEEE Glossary) requirement.
  1. *A condition or capability needed by a user to solve a problem or achieve an objective.*
  2. *A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.*
  3. *A documented representation of a condition or capability as in (1)(2).*
- Ingegneria dei requisiti: Definire i requisiti del sistema di interesse.
  - Raccolta dei requisiti
  - Analisi dei requisiti

Requisiti software

- Requisiti di un sistema software:

  Cosa deve fare il sistema + Vincoli operativi

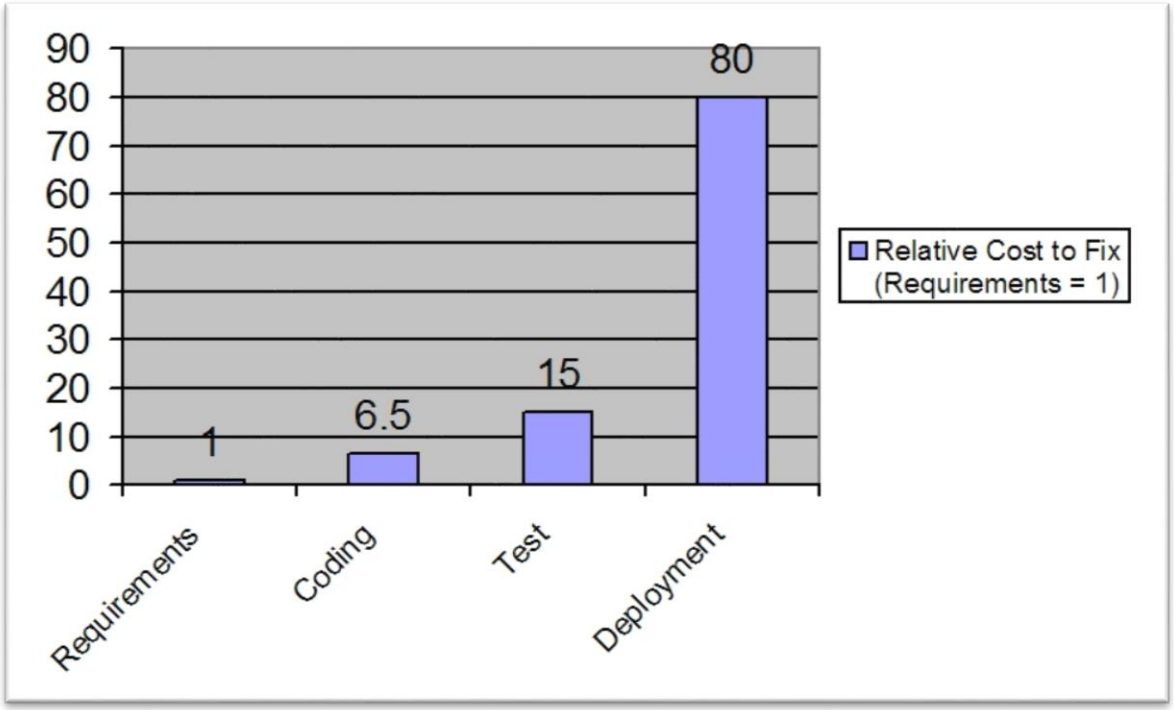
- Esempi:
  - Il sistema dovrà archiviare i dati di di una biblioteca, in particolare dati relativi ai libri, ai giornali, le riviste, video, nastri audio e CD-ROM.
  - Il sistema dovrà permettere agli utenti di fare delle ricerche per titolo, autore, o ISBN.
  - L'interfaccia al sistema dovrà essere realizzata come un'app per iOS e Android.
  - Il sistema dovrà gestire almeno 20 transazioni per secondo
  - Il sistema dovrà riconoscere l'utente attraverso una smart-card

Requirement Engineering

- L’ingegneria dei requisiti comprende tutte le attività che si occupano di requisiti:
  - Raccolta dei requisiti
  - Analisi dei requisiti
  - Specifica / Documentazione dei requisiti
  - Verifica e Validazione dei requisiti

Fred Brook's

- "*The most difficult part of building a software system is to decide, precisely, what must be built.*
  No other part of the work can undermine so badly the resulting software if not done correctly. No other part is so difficult to fix later."



Requirement Engineering (2)

- La raccolta dei requisiti è spesso considerata l'attività più difficile, perché richiede la collaborazione tra più gruppi di partecipanti con differenti background.
  - **Stakeholders**: "insieme dei soggetti che hanno un interesse nei confronti di un'organizzazione e che con il loro comportamento possono influenzarne l'attività".

    →
  - tutte le persone in qualche modo interessate alla messa in opera del sistema
  - Il cliente e gli utenti finali sono esperti nel loro dominio e hanno una idea generale (spesso vaga) di cosa il sistema debba fare, e poca (o nulla) esperienza nello sviluppo del software
  - Gli sviluppatori hanno esperienza nel produrre sistemi software, ma hanno una conoscenza limitata del dominio di applicazione (ambiente degli utenti finali)

Requirement Engineering (3)

- Tutti gli stakeholders comunicano tra di loro per definire il sistema da realizzare.
  - Fallimenti nella comunicazione (tra i diversi domini) portano a un sistema difficile da usare o che non supporta le funzionalità richieste
- Gli errori introdotti in questa fase sono difficili (e costosi) da risolvere perché sono scoperti nelle ultime fasi del processo di sviluppo del software
- Rischi possibili:
  - una funzionalità che il sistema dovrebbe supportare non è specificata;
  - funzionalità incorrette o obsolete;
  - interfacce utenti poco intuitive e difficile da usare.

Tipologie di Requisiti

Tipi di Requisiti

- I Requisiti SW possono essere classificati secondo due diversi punti di vista:
  - Livello di Dettaglio
    - Requisiti Utente
    - Requisiti di Sistema
  - Tipo di Requisito rappresentato
    - Requisiti Funzionali
    - Requisiti Non Funzionali
    - Requisiti di Dominio

Livello di dettaglio

- Possono espressi a vari livelli di astrazione e formalismo, dando luogo a 2 tipologie di requisiti:
- **Requisiti Utente**
  - Descrivono i servizi richiesti al sistema (comportamento osservabile dall'esterno) e i vincoli operativi del sistema.
  - Scritti in linguaggio naturale.
  - Il punto di vista è quello del Cliente, che li sottopone a un possibile Sviluppatore per ottenere una offerta (requisiti aperti a soluzioni alternative)
- **Requisiti di Sistema**
  - Formulazione dettagliata, strutturata, di servizi e vincoli.
  - Scritti in linguaggio naturale, notazioni semi-formali, linguaggi formali
  - Il punto di vista è quello dello Sviluppatore, che li puo' usare anche per il contratto con il Cliente (requisiti piu' restrittivi)

Un req. utente ➔ molti req. di sistema

Requirements definition (UN Requisito Utente)

1. The software must provide a means of representing and accessing external files created by other tools.

Requirements specification (diventa PIU' requisiti di Sistema)

1.1 The user should be provided with facilities to define the type of external files.
1.2 Each external file type may have an associated tool which may be applied to the file.
1.3 Each external file type may be represented as a specific icon on the user's display.
1.4 Facilities should be provided for the icon representing an external file type to be defined by the user.
1.5 When a user selects an icon representing an external file, the effect of that selection is to apply the tool associated with the type of the external file to the file represented by the selected icon.

Tipi di requisiti

- I **Requisiti Funzionali** descrivono i servizi, o funzioni, offerti dal sistema (normalmente attivati da user-inputs)
  - “Quando l’utente richiede la visualizzazione dell’estratto conto… allora il sistema deve …”
- I **Requisiti Non-Funzionali** descrivono vincoli sui servizi offerti dal sistema, e sullo stesso processo di sviluppo
  - “La visualizzazione dell’estratto conto deve avvenire entro 4 secondi dalla sua richiesta”
- I **Requisiti di Dominio** (funzionali e non-funzionali) riflettono caratteristiche generali del dominio applicativo
  - “L’accesso alla cassa continua da parte dell’addetto bancario al rifornimento deve avvenire secondo le consuete procedure di sicurezza a doppia-chiave”

Requisiti Funzionali

- Descrivono le interazioni tra il sistema e il suo ambiente indipendentemente dalla sua implementazione (l'ambiente include l'utente e ogni altro sistema esterno)
- ESEMPIO
  - Il sistema POS NextGen è un'applicazione informatica che deve permettere di registrare le vendite ed in pagamenti in negozi e supermercati.
  - Il sistema dovrà permettere ad un utente che vuole pagare con PagoBancomat di inserire il PIN su un'apposita tastiera, dopo aver visualizzato su un display l'elenco degli articoli acquistati.
  - Il Display deve mostrare l'elenco degli articolo acquistati, con il relativo prezzo, un eventuale sconto applicato, ed il costo complessivo dell'intera spesa.
  - …

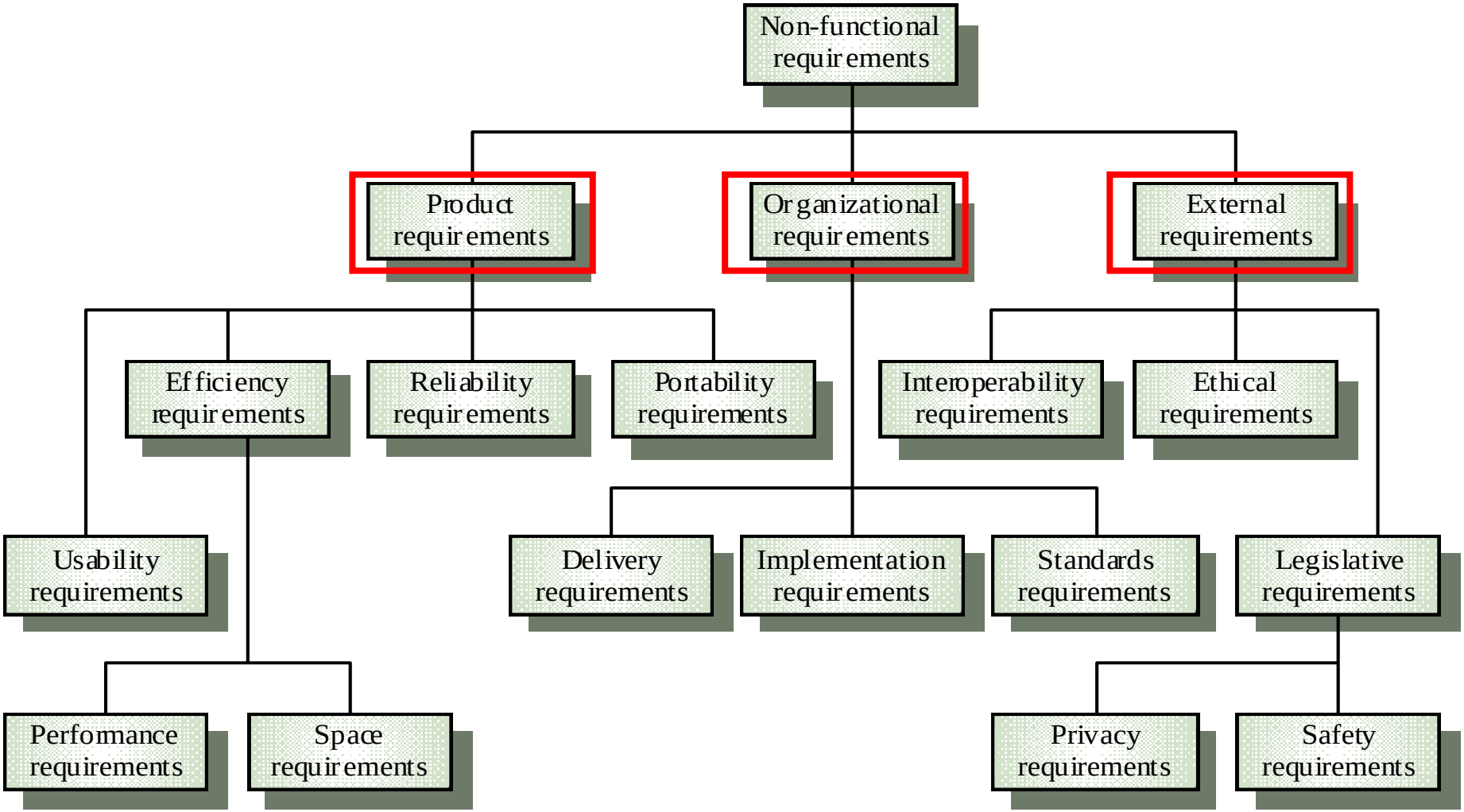
Qualità dei Requisiti Funzionali

- I requisiti funzionali devono essere:
  - Completi: devono indicare tutti i servizi richiesti dagli utenti
  - Coerenti: i requisiti non devono avere definizioni contraddittorie
- Per sistemi grossi è difficile ottenere requisiti completi e coerenti
  - I vari stakeholders hanno esigenze diverse, spesso in contrasto

Requisiti non funzionali

- Descrivono gli aspetti del sistema che non sono direttamente legati al comportamento (funzionalità) del sistema.
- ESEMPIO (POS NextGEN).
  - Risposta all'autorizzazione entro 30 secondi, nel 90% dei casi
  - Il sistema deve essere in grado di gestire diversi tipi di interfacce utente, quali un display multiriga, un web-browser, etc…
  - Qualora ci siano problemi di connettività, il sistema deve comunque registrare in locale il pagamento, in modo da permettere il normale svolgimento degli acquisti nel negozio.
- Includono una grande varietà di richieste che si riferiscono a diversi aspetti del sistema, dall'usabilità alle performance
  - Varie classificazioni (Sommerville, FURPS+, etc…)

Tipi di requisiti non-funzionali (Sommerville)



Il modello FURPS

- FURPS è un acronimo per la definizione dei requisiti
- FURPS classifica i Requisiti in:
  - Functionality ← Requisiti Funzionali
  - Usability = Usabilità
  - Reliability = Affidabilità
  - Performance
  - Supportability = Supportabilità

Requisiti Non Funzionali

Requisiti non Funzionali: Il modello FURPS

- Usabilità
  - facilità per l'utente di imparare ad usare il sistema, e capire il suo funzionamento. Includono:
    - convenzioni adottate per le interfacce utenti
    - portata dell'Help in linea
    - livello della documentazione utente.

- Affidabilità
  - capacità di un sistema o di una componente di fornire la funzione richiesta sotto certe condizioni e per un periodo di tempo. Includono:
    - un accettabile tempo medio di fallimento
    - l'abilità di scoprire specificati difetti
    - o di sostenere specificati attacchi alla sicurezza.

Requisiti non Funzionali: Il modello FURPS

- Performance
  - riguardano attributi quantificabili del sistema come
  - tempo di risposta, quanto velocemente il sistema reagisce a input utenti
  - throughput, quanto lavoro il sistema riesce a realizzare entro un tempo specificato
  - disponibilità, il grado di accessibilità di una componente o del sistema quando è richiesta

- Supportabilità
  - Riguardano la semplicità di fare modifiche dopo il deployment. Includono:
    - adattabilità, l'abilità di cambiare il sistema per trattare concetti addizionali del dominio di applicazione
    - mantenibilità, l'abilità di cambiare il sistema per trattare nuove tecnologie e per far fronte a difetti

Requisiti di qualità per POS NextGEN

- Il testo deve essere visibile da un metro di distanza (Usabilità)
- Qualora ci siano problemi di connettività, il sistema deve comunque registrare in locale il pagamento, in modo da permettere il normale svolgimento degli acquisti nel negozio. (Affidabilità)
- Risposta all'autorizzazione entro 30 secondi, nel 90% dei casi (Performance)
- Deve essere possibile cambiare la lingua del testo visualizzato (Supportabilità / Usabilità)

Requisiti di dominio

- Si riferiscono a caratteristiche del sistema che derivano da caratteristiche generali del dominio applicativo

- Problemi:
  - A volte sono Requisiti impliciti: il Cliente li dà per scontati e non li esprime esplicitamente: ma lo Sviluppatore non lo sa…
  - A volte sono Requisiti espliciti ma oscuri: Il Cliente descrive requisiti utilizzando termini e concetti che lo Sviluppatore ignora…(➔ Glossario)

Requisiti di dominio – Esempi

- Gestione biblioteca
  - ‘There shall be a standard user interface to all databases which shall be based on the Z39.50 standard’
    - Lo Standard è noto al personale della biblioteca, non agli sviluppatori
- POS NextGEN
  - Il codice identificativo dell’articolo, letto mediante codice a barre, può essere basato sui seguenti schemi di codifica:
    - UPC
    - EAN
    - JAN
    - SKU

Difficoltà nella Requirement Elicitation

Verificabilità di requisiti non funzionali

- I Requisiti non funzionali dovrebbero sempre essere Verificabili
- Goal (non verificabile)
  - ‘Il sistema deve essere facile da usare per controllori esperti, e deve essere tale da minimizzare gli errori degli utenti’
- Requisito non-funzionale (verificabile)
  - ‘controllori esperti devono poter imparare a usare tutte le funzioni del sistema in max. 2 ore di apprendimento.
  - Dopo l’apprendimento, il controllore deve essere in grado di operare senza commettere piu’ di 2 errori al giorno’.

Quantificazione di requisiti non-funzionali

Property	**Measure**
Speed	Processed transactions/second User/Event response time Screen refresh time
Size	K Bytes Number of RAM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

Conflitti fra requisiti non-funzionali

- Esempio: Sistema di rilevamento pattume spaziale sullo Shuttle
  - Req.1 - System should fit into 4Mbytes of memory
  - Req.2 - System should be written in ADA

- Puo' risultare impossibile compilare un programma ADA con le funzionalià richieste, e che occupi solo 4Mbytes: uno dei requisiti va escluso

Validazione dei requisiti

- E’ un passo critico nel processo di sviluppo
- I requisiti sono continuamente validati da cliente e utenti
- La validazione dei requisiti richiede di controllare:
  - **Correttezza**: una specifica è corretta se rappresenta accuratamente il sistema che il cliente richiede e che gli sviluppatori intendono sviluppare;
  - **Completezza**: una specifica è completa se tutti i possibili scenari per il sistema sono descritti, incluso i comportamenti eccezionali
  - **Coerenza**: se i requisiti non si contraddicono tra di loro
  - **Chiarezza**: una specifica è chiara se non è possibile interpretare la specifica in due modi diversi

Validazione dei requisiti (2)

- **Realismo**: La specifica dei requisiti è realistica se può essere implementata tenendo conto dei vincoli
- **Verificabilità**: La specifica dei requisiti è verificabile se, una volta che il sistema è stato costruito, test ripetuti possono essere delineati per dimostrare che il sistema soddisfa i requisiti
  - Es: Il prodotto dovrebbe avere una buona interfaccia – Buono non definito
- **Tracciabilità**: Ogni funzione del sistema può essere individuata e ricondotta al corrispondente requisito funzionale.
  Include anche l'abilità di tracciare le dipendenze tra i requisiti, le funzioni del sistema, gli artefatti, incluso componenti, classi, metodi e attributi di oggetti
  - È cruciale per lo sviluppo di test e per valutare i cambiamenti

Tipi di raccolta dei requisiti

- Classificati sulla base della sorgente dei requisiti:
  - Greenfield Engineering
    - Lo sviluppo parte da zero, nessun sistema esiste in precedenza, così i requisiti sono “estratti” dagli utenti e dal cliente
    - É guidata dalla necessità dell’utente o dalle esigenze del mercato
  - Re-engineering
    - Un sistema esistente viene riprogettato a causa della disponibilità di nuove tecnologie o per estendere funzionalità del sistema
  - Interface Engineering
    - Per fornire i servizi di un sistema esistente in un nuovo ambiente
    - Vengono riprogettate le interfacce
    - Sistemi legacy lasciati inalterati, eccetto che per le interfacce
    - È guidata dall’introduzione di nuove tecnologie, e da nuove necessità del mercato