**INGEGNERIA DEI REQUISTI**

Specifica dei requisti -> stesura dei requisti + casi d’uso

Analisi dei requisiti: i requisiti me li danno gli utenti e ho successo in questa fase se ho capito bene quali sono; produco un documento che dice esattemente cosa devo fare non mi interesso in questa fase del come.

1. **DEFINIZIONE**

Processo di ricerca, analisi, documentazione e verifica dei requisti di sistema; i requisti di sistema si dividono a loro volta in:

* Funzionalità: ciò che il cliente richede come cosa deve fare il programma
* Vincoli: i vincoli che mi vengono dati in termini di tempo o altro.

L’ ingegneria dei requisiti non è solo una cosa tecnica, l’ aspetto umano è fondamentale, devo interagire con il commitente.

Modelli di sistema

Studio di Fattibilità

Deduzione e analisi dei requisti

Specifica dei requisti

Convalida dei requisiti

Rapporto di fattibilità

Requisti utente e di sistema

Documento diei Requisiti

**LEGENDA:**

1. STUDIO DI FATTIBILITÀ: capire cosa vuole la persona e se io riesco o meno a dare una soluzione al problema posto.
2. DEDUZIONE E ANALISI DEI REQUISTI: cerco di capire usando sistemi che già esistono ciò che mi chiede il committente, posso usare anche i prototipi per fare questo.
3. SPECIFICA DEI REQUISTI: qui posso descrivere i requisti esattamente come ho capito che sono dalle fasi precedenti.
4. CONVALIDA DEI REQUISTI: vado a vedere se i requisti ottentui dalla fase preccedente sono fattibili e conformi a quelli iniziali.
5. **COS’ È UN REQUISITO?**

* Descrizione ad alto livello di cosa deve fare il mio sistema
* Descrizione di un vincolo di sistema
* Descrizione dettagliata di una funzionalità

Nota: cambia il livello di astrazione

1. **TIPI DI REQUISTI:**
2. UTENTE: requisiti astratti di alto livello spesso espressi in linguaggio naturale, eventualmente con diagrammi, e permettono di spiegare al programmatore cosa si vuole, sono dati da chi non si occupa dell’implementazione, l’utente il manager e i collaboratori del manager.
3. DI SISTEMA: descrizione dettagliata delle funzionalità del sistema può essere anche in linguaggio semi-formale, vengono dati da coloro che si intendono degli aspetti informatici come sviluppatori e sistemisti.
4. **ALTRA CLASSIFICAZIONE**
5. FUNZIONALI: descrivono i servizi, elenco dei servizi che deve dare il sistema e come esso deve comportarsi, inoltre indicano anche le cose che il sistema non deve fare, possono essere sia utente che di sistema.
6. NON FUNZIONALI: vincoli sui servizi e eventuali vincoli sul processo di sviluppo, si applicano al sistema generale non al singolo servizio.
7. DI DOMINIO: raccontano le caratteristiche del dominio in cui ci stiamo muovendo, possono essere funzionali o non (es. bisogna seguire il protocollo x, o la legge y).

Nota: spesso e volentieri la distinzione non è così netta, è importante notare che non soddisfare un requisto non fuznionale potrebbe rendere il sistema inutilizzabile.

1. **REQUISTI FUNZIONALI PER LIVELLO DI ASTRAZIONE**
   1. L’ utente deve poter cercare gli appuntamenti nei reparti.
   2. Il sistema per ogni giorno per ogni reparto genera la lista dei pazienti che hanno un appuntament quel giorno.
   3. Ogni utente del sistema è identificato da login e password più quanto detto sopra

Nota: lo stesso requisti funzionale viene ripetuto a diversi livelli di astrazione in maniera crescente

1. **CARATTERISTICHE DEI REQUISTI:**
2. Precisi
3. Completi
4. Consistenti: i requisti devono avere un senso tra loro non può essere che qualcuno voglia tutte le possibile funzioni e vogli anche pagare poco.
5. **REQSUITI NON FUNZIONALI:**
6. Requisti del prodotto: raccontano reqsuiti come volecità, facilità d’uso e in generale cose che riguardano il Sw in sé
7. Reqsuiti organizzativi: presentano le condizioni di “contorno” come vincoli di qualità o procedura, non sono legati al Sw in sé
8. Requisti esterni: dipendono dai vincoli portati dal mondo esterno come i vincoli di legge.

Nota: per i requsiti non funzionali mi servono delle misurazioni (es. cosa vuol dire veloce? 1 s 1ms?) e un singolo requisito non funzionale èuò generare più requisti funzionali. Sono importanti anche i requisti di interfaccia che sarebbe come i sistemi si interacciano tra loro non è l’interfaccia utente.

1. **REQUISITI DI DOMINIO**

Dipendono dal dominio dell’applicazione, spesso sono specificati da chi mi da il lavoro e ciò li rende talvolta diffcili da comprendere perché espressi in un linguaggio tecnico proprio del commitente, inoltre può essere che elncandoli il commitente tralasci alcune cose dandole per scontate quando in realtà per una persona che non appartiene a quel settore potrebbero non esserlo, anche questi possono dare vita a più reqsuiti funzionali.

1. **DOCUMENTO DEI REQSUITI**

Decisi tutti i reqsuiti è opportuno produrre un documento dei reqsuiti, che sarà la dichiarazione ufficale di quanto deve essere sviluppato, da notare che esso non è un documento di progettazione, presenta solo il cosa fare non il come farlo. In tale documento sono presenti:

* Requisiti utente: reaccontano reqsuiti funzionali e non ad alto livello, descrivono il comportamento esterno del sistema, è importante evitare le caratteristiche di progettazione a questo livello, si deve usare un linguaggio semplice, comprensibile, non tecnico, potendo si sfruttano inguaggi grafici (casi d’uso e linguaggi di sicurezza), inoltre è importante cercare di standardizzare la raccolta dei reqsuiti in modo da non dimenticare nulla, importante sapere cosa deve esserci nel Sw e cosa sarebbe meglio ci fosse (obbligatorio o no).
* Reqsuiti di sistema: versione espansa dei reqsuiti utente, sono più dettagliati e sono usati come punto di partenza per lo sviluppo del Sw, dovrebbero anch essi essere privi di dettagli di implementazione anche se spesso e volentieri ci finiscono lo stesso. Uso sempre il linguaggio naturale ma cerco un compromesso tra linguaggio tecnico e non.

1. **A CHI È RIVOLTO IL DOCUMENTO DEI REQSUITI:**

* Clienti: coloro che mi hanno assunto e che useranno il Sw
* Manager: quelli che gestisono la parte contrattuale
* Sviluppatori: quelli che faranno l’implementazione (non è detto che debba farla io)
* Sviluppatori di test: coloro che penseranno a creare i test per validare il mio sistema
* Addetti alla manutenzione

1. **STRUTTURA GENERALE DEL DOCUMENTO DEI REQSUITI (IEEE/ANSI 830-1998)**
2. Parte introduttiva: conterrà abbreviazioni usate, riferimenti, e una descrizione del documento
3. Descrizione genrale: descrizione generale del prodotto, vincoli di dominio
4. Requisiti specifici: tutti quelli sopracitati funzionali e non.
5. Appendici
6. Indice
7. **STRUTTURA ALTERNATIVA (Più usata):**
8. Prefazione: racconto i potenziali lettori del documento, descrivo anche la “storia” della versione corrente cioè perché ho fatto una nuova versione e cosa è cambiato.
9. Introduzione: descrizione generale del sistema, descrivo in breve le funzionalità, eventtuali interazioni con altri sistemi e eventuale collocazione del sistema dal punto di vista organizzativo (perché ci è stato chiesto e che miglorie porterà all’azienda).
10. Glossario: descrivo i termini tecnici (e non) usati nel documento, devo evitare il più possbile di dare cose per scontate.
11. Definizione dei reqsuiti utente: servizi fonriti agli utenti, basso livello di dettaglio, uso di diagrammi (casi d’uso) e linguaggio naturale.
12. Architettura del sistema: descrizione ad alto livello dell’architettura del sistema, questa parte è rivolta in particolare allo sviluppatore.
13. Specifica dei reqsuiti di Sistema: descrizione dettagliata dei reqsuiti funzionali e non.
14. Modelli del Sistema: descrizione delle componeneti, altamente opzionale, uso diagrammi a oggetto e di flusso.
15. Evoluzione del sistema: si possono prevedere eventuali modifiche per componeneti Hw, o possono essere fatte per necessità.
16. Appendici: requisti Hw, legacy system, informazioni necessarie, elementi introdotti.
17. Indice

Nota: Una cosa fondamentale è il linguaggio da usare e in che punti, in pratica:

Lingauggio Naturale(Strutturato)

Lingauggio Naturale(Strutturato)

**USO DI SPECIFICHE STRUTTURATE:** riduce le ambiguità e aumenta l’efficacia

Mi fa ridurre la variabilità nella descrizione dei requisti.

Casi d’uso, tabelle…

1. **PROCESSO ALTERNATIVO INGEGNERIA DEI REQUISITI:**

Convalida dei requisiti

Specifica dei requisti

Deduzione e analisi dei requisti

Studio di Fattibilità

A questo approccio classico all’ingegneria dei reqsuiti se ne contrappone un altro chimato **PROSPETTIVA A SPIRALE**:

**NON COMPLETO**

# Studio di fattbilità:

## Fase inziale

## Descrizione sommaria del sistema -> requisti preliminari (A contatto con gli stakeholders -> coloro che hanno a che fare con il sistema)

# Deduzione e anlisi dei reqsuiti:

## Scoperta dei reqsuiti: processo di raccolta di informazioni -> interazione con gli stakeholders vado a usare:

### Interviste: che possono essere formali o informali, aperte o chiuse

### Scenari: che possono essere esempi di interazione o dettagliati

### Casi d’uso: si basano sigli scenari uso UML.

### Etnografia:

#### Reqsuiti socialie organizzativi

#### Tecniche di osservazione -> req. Impliciti, processi reali vs. processi formali

#### Reqsuitiderivanti dal modo in cui le persone lavorano

#### Reqsuiti relativi alla cooperazione

#### Prototipazione: concentro l’attività nell’identificare i problemi da discutere, uso tecniche di osservazione.

## Classificazione e organizzazione

## Negoziazione e priorità

## Documentazione

# Convalida dei reqsuiti: dimostrare che i reqsuiti definiscono il sistema che il cliente vuole -> errori nei reqsuiti costano moltissimo.

## Varie attività di controllo:

### Di verificabilità

### Di validità

### Di consistenza -> ci sono reqsuiti in conflitto o in contraddizione?

### Di Completezza -> rispetto a quanto ci è stato chiesto

### Di realismo -> Posso implementare i reqsuiti richiesti in base a tecnologie e budget?

## Tecniche di Validazione:

### Revisione dei reqsuiti -> team di revisori

### Prototipazione -> creo un prototipo per fare delle verifiche

### Generazione dei test

### Verifico l’assenza nel documento dei reqsuiti di anomalie e omissioni:

#### Comprenisbilità -> verifico che il reqsuito sia comprensibile

#### Verificabilità

#### Tracciabilità

#### Adattabilità

## Gestione dei reqsuiti: controllare cambaimenti -> tengo traccia dei cambiamenti nei reqsuiti e del loro collegamento

## Evoluzione dei reqsuiti:

### Reqsuiti duraturi -> Stabili nel tempo

### Reqsuiti volatili -> Cambiano più facilmente

**TEMPO**

Modifica dei Reqsuiti

Reqsuiti inziali

Comprensione inizale del problema

Modifche nella descrizione del problema

## Pianificazione: è necessario sapere che i reqsuiti possono cambiare, devo usare politiche di tracciabilità e definire in che modo gestire la modifca dei reqsuiti:

### Identifcare i reqsuiti

### Gestire le modifche dei reqsuiti

### Gestire relazioni tra reqsuiti

1. **ESERCIZI USE-CASE**

Questi sono lasciati alle diapositive dove sono speigati.