

# Progetto

Requisiti del software

[manuel.comparetti@iet.unipi.it](mailto:manuel.comparetti@iet.unipi.it)

# Requisiti del software

Riferimento: Ian Sommerville, *Software Engineering (8° edition)*

Il capitolo 6 del libro, su cui si basa il presente materiale, è disponibile online:

<http://www.comp.lancs.ac.uk/computing/resources/lanS/SE8/SampleChapters/index.html>

# Obiettivi

- Requisiti utente/requisiti sistema
- Requisiti funzionali/non funzionali
- Come organizzare i requisiti del software in un documento

# Forma di un requisito

- La forma in cui un requisito viene espresso può essere una frase in linguaggio naturale (di “alto livello”) che descrive una funzionalità del sistema o un vincolo a cui il sistema deve sottostare, ma anche una descrizione in un linguaggio formale (algoritmo, formula matematica etc)
- Questo è dovuto al fatto che gli scopi per cui i requisiti vengono specificati sono molteplici
  - Base per contrattare le funzionalità del software tra fornitore e committente
  - Base per “specificare” in maniera univoca il contratto tra fornitore e committente
  - Base per il documento di progetto (l’implementazione delle funzionalità stesse)

# Livello di dettaglio dei requisiti

## ■ Requisiti utente.

- Usualmente sono scritti in linguaggio naturale e diagrammi e descrivono ad alto livello i servizi forniti dal sistema; sono scritti per essere compresi dal cliente.

## ■ Requisiti del software:

- Documento strutturato che descrive dettagliatamente e univocamente le funzionalità del sistema; sono il punto di partenza per il documento di progetto e hanno valore legale di “contratto”.

# Esempio

## User requirement definition

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

## System requirements specification

- 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.

# Destinatari delle varie tipologie

## ■ Requisiti utente:

- Management
- Utenti

## ■ Requisiti del software:

- Utenti (o chi scrive il manuale utente)
- Software architects
- Sviluppatori (...)

# Requisiti : classificazione

- I requisiti di un sistema software si possono suddividere (logicamente e nel documento) in due tipologie fondamentali:
- Requisiti funzionali.
  - Esprimono, con vari livelli di dettaglio (utente, sistema) “cosa deve fare il sistema”. Esprimono le funzionalità del sistema in termini di relazioni tra dati di ingresso e dati di uscita.
- Requisiti non funzionali.
  - Sono tutti gli altri requisiti, che possono esprimere caratteristiche di qualità e/o vincoli sulle funzionalità del sistema. Ad esempio:
    - Qualità: performance (vincoli sui tempi di risposta di un sistema).
    - Qualità: usabilità.
    - Vincoli: occupazione di memoria.

# Requisiti : classificazione (ii)

- Requisiti funzionali e requisiti non funzionali:
  - Spesso la distinzione tra i due tipi di requisiti può non essere immediata, inoltre è possibile che il vincolo espresso da un requisito non funzionale possa tradursi in requisiti funzionali ad un livello di dettaglio maggiore (..).
  - Più in generale tra i requisiti devono comparire, se necessari, la documentazione di interfacce esterne (ovvero sistemi con cui il prodotto si interfaccia, es, database, software legacy, dispositivi di i/o etc), requisiti logici di un database esterno oltre ai suddetti vincoli di design e alle caratteristiche di qualità.
  - Nota: alcuni libri di testo e standard (tra cui Sommerville) individuano una terza tipologia (trasversale) di requisito, i cosiddetti requisiti relativi al dominio. Tali requisiti sono una conseguenza del particolare dominio applicativo, più che di un obiettivo dell'utente o un vincolo del sistema, e riflettono caratteristiche del dominio applicativo stesso (...).

# Requisiti funzionali

- Descrivono le funzionalità, i servizi offerti dal sw.
- Dipendono del tipo di software, dal tipo di utenti e dal tipo di sistema su cui il software è eseguito.
- I requisiti funzionali “utente” possono essere frasi ad alto livello in linguaggio naturale.
- Un successivo raffinamento consiste nella specifica dei requisiti del software, che prevedono un livello di dettaglio maggiore.

# Esempio: LIBSYS system

- Un sistema informativo che offre all'utente un'unica interfaccia per accedere a una insieme di database di pubblicazioni presenti in varie librerie
- Gli utenti possono effettuare una ricerca degli articoli, scaricarne una copia e stamparli

## Esempio(ii):Maggiore livello di dettaglio

- L'utente deve poter effettuare una ricerca sull'intero insieme dei database disponibili o poterne specificare un sottoinsieme
- Il sistema deve fornire all'utente un visualizzatore appropriato (viewer ...) per permettere la consultazione dei documenti presenti
- Ad ogni ordine deve essere associato un identificatore univoco (ORDER\_ID) che l'utente deve poter copiare sull'area di memoria dell'account (??): “Every order shall be allocated a unique identifier (ORDER\_ID), which the user shall be able to copy to the account's permanent storage area.” (...)

# Imprecisione

- L'imprecisione è inevitabile laddove venga utilizzato il linguaggio naturale (..)
- Se i requisiti non sono specificati in maniera precisa possono sorgere diversi problemi (ambiguità)
- Sia gli sviluppatori che gli utenti possono attribuire significati diversi al requisito
- “visualizzatore appropriato” ??:
  - Per ogni tipo di documento viene fornito un reader che lo interpreti
  - Deve essere fornito un visualizzatore che interpreti ogni formato di documento (...)

# Caratteristiche di qualità dei requisiti

- Correttezza
- Non ambiguità
- Completezza
- Consistenza
- Importanza
- Stabilità
- Verificabilità
- Modificabilità
- Tracciabilità
- Comprensibilità
- Fattibilità
- Livello di dettaglio adeguato

## Qualità dei Requisiti (ii)

- **Correttezza:** i requisiti che sono implementati devono riflettere il comportamento atteso
- **Non ambiguità:** il requisito deve avere solo una possibile interpretazione
- **Completezza:** tutti gli elementi importanti che sono rilevanti per l'utente devono essere considerati

# Qualità dei requisiti (iii)

- **Consistenza:** i requisiti devono essere consistenti verso loro stessi e verso i *vincoli*
- **Valutati per importanza:** ogni requisito deve essere valutato in termini di importanza, cioè di quanto esso è essenziale per il successo del progetto
- **Stabilità:** deve essere specificata la possibilità o meno di dover/poter cambiare il requisito
- **Verificabilità:** tutti i requisiti (funzionali e non) devono essere verificabili, cioè esiste un processo per controllare se il requisito è soddisfatto o no

# Qualità dei requisiti (iv)

- **Modificabilità:** alcuni requisiti possono dover essere modificati
- **Tracciabilità:** si deve tenere traccia delle informazioni che via via si introducono:
  - Traccia dei requisiti, nei diversi livelli di dettaglio,
  - Data una qualsiasi funzionalità implementata, si deve poter risalire al relativo requisito/caso d'uso,
  - Se si introducono modifiche ad un requisito, si deve poter risalire alla funzionalità implementata (nel progetto)
- **Fattibilità:** tutti i requisiti devono essere implementati con le risorse, la tecnologia e il budget disponibili
- **Giusto livello di dettaglio:** l'informazione contenuta nel requisito permette di ottenere la giusta comprensione per dar luogo all'implementazione

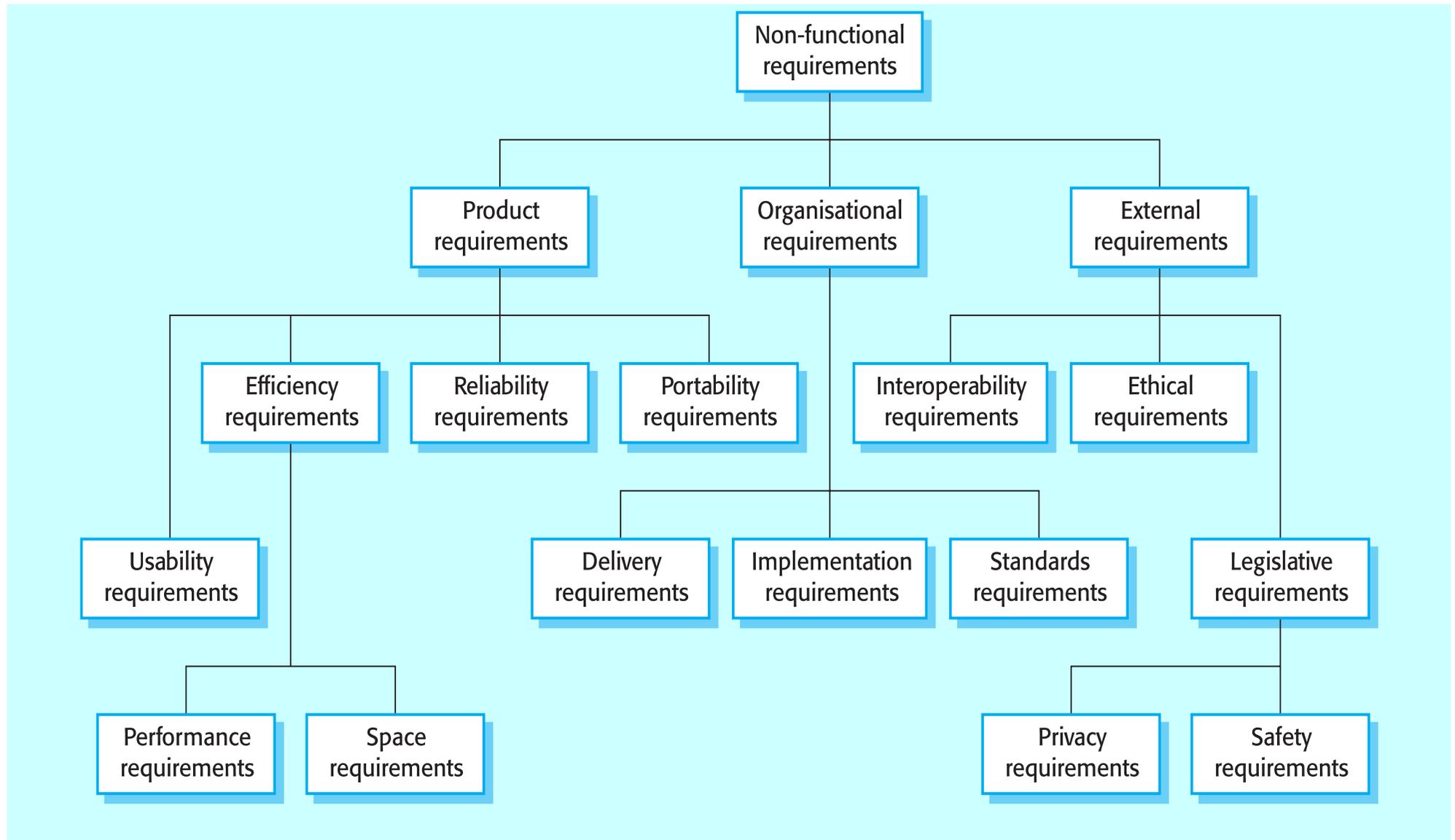
# Requisiti non funzionali

- I requisiti non funzionali specificano proprietà del sistema e vincoli, come ad esempio la reliability, i tempi di risposta e l'occupazione di memoria. Altri vincoli possono derivare dai dispositivi di i/o e dalla rappresentazione dei dati all'interfaccia con altri sistemi.
- Si noti come in generale particolari vincoli possono essere imposti sul processo di produzione più che sul sistema. Ad esempio l'utilizzo di un particolare strumento CASE o di un particolare linguaggio o tecnologia. Spesso tali vincoli rispecchiano una ragione economica più che tecnologica.
- I requisiti non funzionali possono essere critici per il corretto funzionamento del sistema. Il mancato soddisfacimento di un requisito non funzionale può rendere il sistema inutilizzabile (e rappresentare un costo per il produttore).

# Classificazione dei requisiti non funzionali

- **Requisiti sul prodotto**
  - Specificano vincoli sul comportamento del sistema (ad. esempio tempi di risposta, reliability, etc)
- **Requisiti del processo**
  - Sono una conseguenza di vincoli sul processo di produzione o di regole organizzative (ad. esempio gli standard di produzione, i vincoli sull'implementazione etc.)
- **Requisiti “esterni”**
  - Derivano da fattori esterni al sistema e al processo, come ad esempio requisiti di interoperabilità o legislativi. Si pensi ad esempio alla privacy per i sistemi informativi o a requisiti legali di ergonomia di un sistema

# Tassonomia



# Esempi di requisiti non funzionali

## ■ Sul prodotto.

- 8.1 L'interfaccia utente per LIBSYS dovrà essere implementata in HTML semplice senza frame o applet java (attenzione! vincola l'implementazione...).

## ■ Sul processo.

- 9.3.2 Il processo di sviluppo del software e i documenti prodotti nelle varie fasi (deliverables...) devono essere conformi allo standard definito in XYZCo-SP-STAN-95.

## ■ External requirement.

- 7.6.5 Il sistema non deve rendere noto agli operatori nessuna informazione personale degli utenti escluso nome e codice utente (privacy, richiesto dalla legge o dalle policy della softwarehouse).

# Dagli scopi dell'utente ai requisiti

- Può essere difficile definire precisamente i requisiti non funzionali, e renderli quindi difficili da verificare
- Scopo
  - L'"intenzione" dell'utente rischia di essere generica
- Un requisito non funzionale verificabile
  - E' un'affermazione che usa una qualche misura che può essere testata in maniera oggettiva
- E' comunque utile corredare l'introduzione di un requisito (questo vale anche per i requisiti funzionali) allo scopo iniziale dell'utente che lo ha generato. Questo può essere utile agli sviluppatori per capire le intenzioni dell'utente (ad esempio nel caso di requisiti di usabilità )

# Esempi

## ■ Scopo generico dell'utente.

- Il sistema deve essere “semplice da usare” (*da chi?*) (*Ad esempio da un utilizzatore “esperto”*) (*che vuol dire esperto?*) E deve essere organizzato in maniera (*l'interfaccia utente*) tale da minimizzare gli errori dell'utente.

## ■ Corrispondente requisito “verificabile.”

- Utenti *esperti* devono essere in grado di utilizzare tutte le funzionalità del sistema dopo due ore di training. Dopo tale periodo di training il numero medio di errori effettuati dall'utente non deve superare i due al giorno. ( ... ).

# Metriche per i requisiti non funzionali

---

<b>Property</b>	<b>Measure</b>
Speed	Processed transactions/second User/Event response time Screen refresh time
Size	M Bytes Number of ROM 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 <u>statements</u> Number of target systems

---

# Note

- Reliability
  - “continuity of correct service”\*
  - mean time to failure, mean time to repair -> mean time between failures
- Availability
  - “readiness for correct service”\*
  - “percentuale di tempo” in cui il sistema funziona correttamente (es. 24/7, 96.7% )
- ... alcuni testi considerano l’availability come una metrica di reliability, altri le raggruppano nella “dependability” (“affidabilità”), assieme con manutenibilità, sicurezza e riservatezza

\*J.C. Laprie. *"Dependable Computing and Fault Tolerance: Concepts and terminology"*

# Note (2)

- Safety (“sicurezza”).
  - Assenza di danni all’utente o all’ambiente circostante in conseguenza dell’utilizzo del prodotto.
- Security (“riservatezza”).
  - Garanzia dell’accesso al sistema solo ad utenti autorizzati.
  - Confidenzialità: garanzia dell’accesso alle informazioni (ad alcune informazioni) solo ad utenti autorizzati.
  - Protezione da modifiche al comportamento del sistema in quanto tale e alle informazioni che custodisce. Tali modifiche possono essere causate da comportamenti non autorizzati/previsti (malintenzionati o accidentali).

# Interazione tra i requisiti

- In generale i vari vincoli espressi dai requisiti possono essere collegati tra di loro. Ad esempio la sicurezza o la robustezza possono influenzare i tempi di risposta o l'occupazione di memoria.
- Inoltre la presenza di vincoli non funzionali può condizionare i vincoli funzionali del sistema o tradursi in più vincoli funzionali a un livello di dettaglio ulteriore degli stessi. Ad esempio l'usabilità può richiedere funzionalità aggiuntive dell'interfaccia o limitarle. Ad esempio un requisito di sicurezza può prevedere una funzionalità di autenticazione nel sistema.
- In sistemi complessi è possibile che vari requisiti non funzionali entrino in conflitto tra di loro.

# Interazione tra i requisiti (esempio)

## Sistema di controllo di un razzo.

- Allo scopo di minimizzare il peso del razzo, il numero di chip a bordo deve essere minimizzato.
- Allo scopo di minimizzare il consumo energetico dell'elettronica di controllo, devono essere impiegati componenti a ridotto consumo di potenza.
- L'utilizzo di componenti a basso consumo di potenza può richiedere l'impiego di un numero maggiore di componenti (ad esempio per ridondanza in caso di non affidabilità). I due requisiti possono entrare in conflitto. E' importante quindi specificare l'ordine di priorità per i requisiti non funzionali.

# Requisiti di dominio

- Derivano dal dominio dell'applicazione e riflettono caratteristiche del sistema che derivano dal dominio di applicazione. Si tratta di una categoria trasversale di requisiti, che esprime l'origine di requisiti funzionali e non funzionali dal dominio dell'applicazione.
- Possono tradursi in requisiti funzionali aggiuntivi, vincoli aggiuntivi o richiedere elaborazione aggiuntiva.
- Determinano l'applicabilità del sistema al dominio per cui è stato progettato (è importante quindi conoscere il dominio dell'applicazione per individuarli in tempo).

# Esempio di requisiti di dominio (LIBSYS)

- Deve essere presente un'interfaccia utente standard a tutti i database basata su Z39.50.
- Per rispetto del copyright alcuni documenti devono essere distrutti (?) immediatamente dopo essere stati recuperati dal sistema. A seconda di quanto specificato dall'utente i documenti possono essere stampati localmente o inviati a una stampante di rete.

# Sistema di sicurezza per un treno

- Il sistema deve calcolare la decelerazione del treno come:

- $D_{\text{train}} = D_{\text{control}} + D_{\text{gradient}}$

con  $D_{\text{gradient}} = 9.81 \text{ m/s}^2 * \text{comp\_gradient} / \alpha$ .

Il valore di  $\alpha$  e di  $\text{comp\_gradient}$  sono noti per i diversi tipi di treni(...)

# Problemi relativi ai requisiti derivanti dal dominio

- **Comprensibilità.**
  - In fase di raccolta dei requisiti stessi, questi ultimi sono espressi nel linguaggio proprio del dominio dell'applicazione.
  - Tale linguaggio può non essere compreso dagli sviluppatori software del sistema.
- **Molti requisiti rischiano di essere considerati impliciti.**
  - da specialisti del dominio dell'applicazione (...) che danno per scontata parte della conoscenza che hanno accumulato sul dominio e non si rendono conto che molti requisiti che considerano impliciti in realtà non sono noti al resto del gruppo di lavoro (esempio un grafico può dare per scontate le equazioni che descrivono alcune curve o non esplicitarne dei parametri).
  - *Nota:* si consideri anche il fenomeno per cui esperti di un settore diverso dall'ingegneria informatica possono non rendersi conto delle reali capacità e dei limiti di un sistema software o di controllo.

# Requisiti a livello utente

- Devono descrivere i requisiti funzionali e non funzionali in modo da risultare comprensibili da utenti che non hanno conoscenza dei dettagli tecnologici e implementativi.
- Si definiscono tramite linguaggio naturale, tabelle e diagrammi (linguaggi universali).

# Problemi del linguaggio naturale

- Mancanza di chiarezza.
  - E' difficile essere sufficientemente precisi senza compromettere la leggibilità dei documenti (sintassi schematica, ripetizioni, ridondanza).
- Confusione tra le tipologie di requisiti.
  - Requisiti funzionali e non funzionali rischiano di essere espressi assieme.
- Requisiti mescolati tra loro.
  - Requisiti differenti espressi nella stessa frase.

# LIBSYS

*4..5 LIBSYS deve comprendere un sistema di gestione finanziaria che mantenga traccia dei pagamenti effettuati dagli utenti del sistema. Gli amministratori del sistema possono configurare il sistema di gestione per abilitare automaticamente sconti per gli utenti selezionati.*

- Questo requisito include informazioni sia generiche che dettagliate

- Nel requisito viene introdotto il concetto di un sistema di gestione finanziaria incluso in LIBSYS

- Contemporaneamente però viene specificato un livello di dettaglio ulteriore, inutile in questa fase: gli amministratori (...) possono configurare il sistema, e in particolare abilitare automaticamente degli sconti (implicitamente hanno la facoltà di gestire categorie di utenti).

# Dalle spec. Di *Eclipse*: requisiti della griglia

**2.6 Grid facilities** To assist in the positioning of entities on a diagram, the user may turn on a grid in either centimetres or inches, via an option on the control panel. Initially, the grid is off. The grid may be turned on and off at any time during an editing session and can be toggled between inches and centimetres at any time. A grid option will be provided on the reduce-to-fit view but the number of grid lines shown will be reduced to avoid filling the smaller diagram with grid lines.

- Sono mescolati tipi diversi di requisiti
  - Requisito funzionale di alto livello (serve una griglia);
  - Requisito non funzionale (unità della griglia);
  - Requisito non funzionale di interfaccia (grid switching);

# Requisito strutturato

---

## 2.6.1 Grid facilities

**The editor shall provide a grid facility where a matrix of horizontal and vertical lines provide a background to the editor window.** This grid shall be a passive grid where the alignment of entities is the user's responsibility.

*Rationale:* A grid helps the user to create a tidy diagram with well-spaced entities. Although an active grid, where entities 'snap-to' grid lines can be useful, the positioning is imprecise. The user is the best person to decide where entities should be positioned.

*Specification:* ECLIPSE/WS/Tools/DE/FS Section 5.6

*Source:* Ray Wilson, Glasgow Office

---

# Linee guida per la scrittura dei requisiti utente

- Definire un formato standard e usarlo sempre.
- Usare le parole in maniera consistente. Ad. Es “Deve” (shall) individua sempre un requisito obbligatorio. “Può” (may) individua un requisito opzionale. Questa pratica può essere esplicitata.
- Evidenziare parole chiave o parti salienti del requisito. (La formattazione è importante, ed è importante che si utilizzi sempre la solita).
- Evitare l'uso di gergalità informatiche (se necessario inserire i termini utilizzati nel glossario): ad es. “settare”, “resettare”, “loggare”, “sloggarsi”, “uploadare”.

# Requisiti del software

- Specificano in maniera dettagliata e completa le funzionalità e i servizi offerti dal sistema e i requisiti non funzionali. Rappresentano il punto di partenza per il documento di progetto e come tali vanno scritti.
- Possono avere valore legale.

# Interazione tra DSR E DSP

- In teoria i requisiti rappresentano “cosa deve fare il sistema” e il progetto “come sistema lo fa.”
- Spesso, specie ad un elevato livello di dettaglio, i due aspetti possono confondersi.
  - Per strutturare i requisiti e renderli comprensibili può essere necessario introdurre un’architettura del sistema (specie in sistemi distribuiti).
  - L’interazione del sistema con altri sistemi esterni può vincolare il progetto.
  - Alcuni vincoli non funzionali possono riguardare l’implementazione (ad esempio vincoli tecnologici).

# Problemi del linguaggio naturale nei requisiti software

## ■ Ambiguità.

- Non è detto che le parole abbiano lo stesso significato per chi scrive e per chi legge.

## ■ Estrema flessibilità.

- La stessa cosa può essere descritta in molti modi diversi.

## ■ Modularità limitata.

- Il linguaggio naturale può non prestarsi a descrivere requisiti strutturati (ad esempio sequenze di controllo ramificate).

# Alternative

---

## Notation

## Description

Structured natural language

This approach depends on defining standard forms or templates to express the requirements specification.

Design description languages

This approach uses a language like a programming language but with more abstract features to specify the requirements by defining an operational model of the system. This approach is not now widely used although it can be useful for interface specifications.

Graphical notations

A graphical language, supplemented by text annotations is used to define the functional requirements for the system. An early example of such a graphical language was SADT. Now, use-case descriptions and sequence diagrams are commonly used.

Mathematical specifications

These are notations based on mathematical concepts such as finite-state machines or sets. These unambiguous specifications reduce the arguments between customer and contractor about system functionality. However, most customers don't understand formal specifications and are reluctant to accept it as a system contract.

---

# Form-based node specification

*Insulin Pump/Control Software/SRS/3.3.2*

**Function** Compute insulin dose: Safe sugar level

**Description** Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units.

**Inputs** Current sugar reading (r2), the previous two readings (r0 and r1)

**Source** Current sugar reading from sensor. Other readings from memory.

**Outputs** CompDose: the dose in insulin to be delivered

**Destination** Main control loop

**Action:** CompDose is zero if the sugar level is stable or falling or if the level is increasing but the rate of increase is decreasing. If the level is increasing and the rate of increase is increasing, then CompDose is computed by dividing the difference between the current sugar level and the previous level by 4 and rounding the result. If the result, is rounded to zero then CompDose is set to the minimum dose that can be delivered.

**Requires** Two previous readings so that the rate of change of sugar level can be computed.

**Pre-condition** The insulin reservoir contains at least the maximum allowed single dose of insulin..

**Post-condition** r0 is replaced by r1 then r1 is replaced by r2

**Side-effects** None

# Tabular specification

---

Condition	Action
Sugar level falling ( $r_2 < r_1$ )	CompDose = 0
Sugar level stable ( $r_2 = r_1$ )	CompDose = 0
Sugar level increasing and rate of increase decreasing ( $(r_2 - r_1) < (r_1 - r_0)$ )	CompDose = 0
Sugar level increasing and rate of increase stable or increasing. ( $(r_2 - r_1) > (r_1 - r_0)$ )	CompDose = round $((r_2 - r_1) / 4)$ If rounded result = 0 then CompDose = MinimumDose

---

# Linguaggi di specifica grafici

## ■ UML

- Use cases
- Sequence diagrams
- Class diagrams
- ...

# Il documento di specifica dei requisiti

- Il documento di specifica dei requisiti ha valore di riferimento ufficiale per ciò che si richiede agli sviluppatori (ancora->deve essere consultabile facilmente!).
- Deve includere i requisiti a livello utente e la specifica dei requisiti del sistema.
- Non è un documento di progetto. Per quanto possibile deve esprimere “cosa” deve fare il sistema e non “come”.

# Destinatari

- **Clienti:** (utilizzatori del sistema) specificano i requisiti desiderati, controllano che questi siano rispettati, propongono modifiche
- **Management:** utilizzano il documento per pianificare la produzione e quantificare i costi
- **Team di progettazione e sviluppo**
- **Team di testing**
- **Team di manutenzione** (per capire meglio il sistema e le relazioni tra le varie componenti ...)

# IEEE requirements standard (830 -1998 )

- Definisce (tra le altre cose) una struttura generica per il documento, da adattare allo specifico sistema da progettare
  - Introduzione (al documento)
  - Descrizione generale
  - Requisiti specifici
  - Appendici
  - Indice(i)