# Requisiti Software

- **Requisiti software** (software requirements): descrizione dei servizi che un sistema software deve fornire, insieme ai vincoli da rispettare sia in fase di sviluppo che durante la fase di operatività del software
- Def. IEEE Std 610.12 (1990):
  - (A) A condition or capability needed by a user to solve a problem or achieve an objective
  - (B) 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 document.
  - (C) A documented representation of a condition or capability as in definition (A) or (B).

# Requisiti software (2)

- I requisiti vengono generati applicando un processo di **ingegneria dei requisiti** (requirements engineering)
- ***Requirements abstraction*** (Davis, 1993)

  "*If a company wishes to let a contract for a large software development project, it must define its **needs** in a sufficiently abstract way that a solution is not pre-defined. The requirements must be written so that several contractors can bid for the contract, offering, perhaps, different ways of meeting the client organisation's needs. Once a contract has been awarded, the contractor must write a **system definition** for the client in more detail so that the client understands and can validate what the software will do. Both of these documents may be called the **requirements document** for the system*"

# Tipi di requisiti

- **Requisiti utente** (*user requirements*):
  - descrizione in linguaggio naturale, con eventuale aggiunta di diagrammi, dei servizi che il sistema deve fornire e dei vincoli operativi
  - sono scritti per (e con) il cliente
- **Requisiti di sistema** (*system requirements*):
  - specificati mediante la stesura di un documento strutturato che descrive in modo dettagliato i servizi che il sistema software deve fornire
  - il documento risultante costituisce un "contratto" tra cliente e fornitore

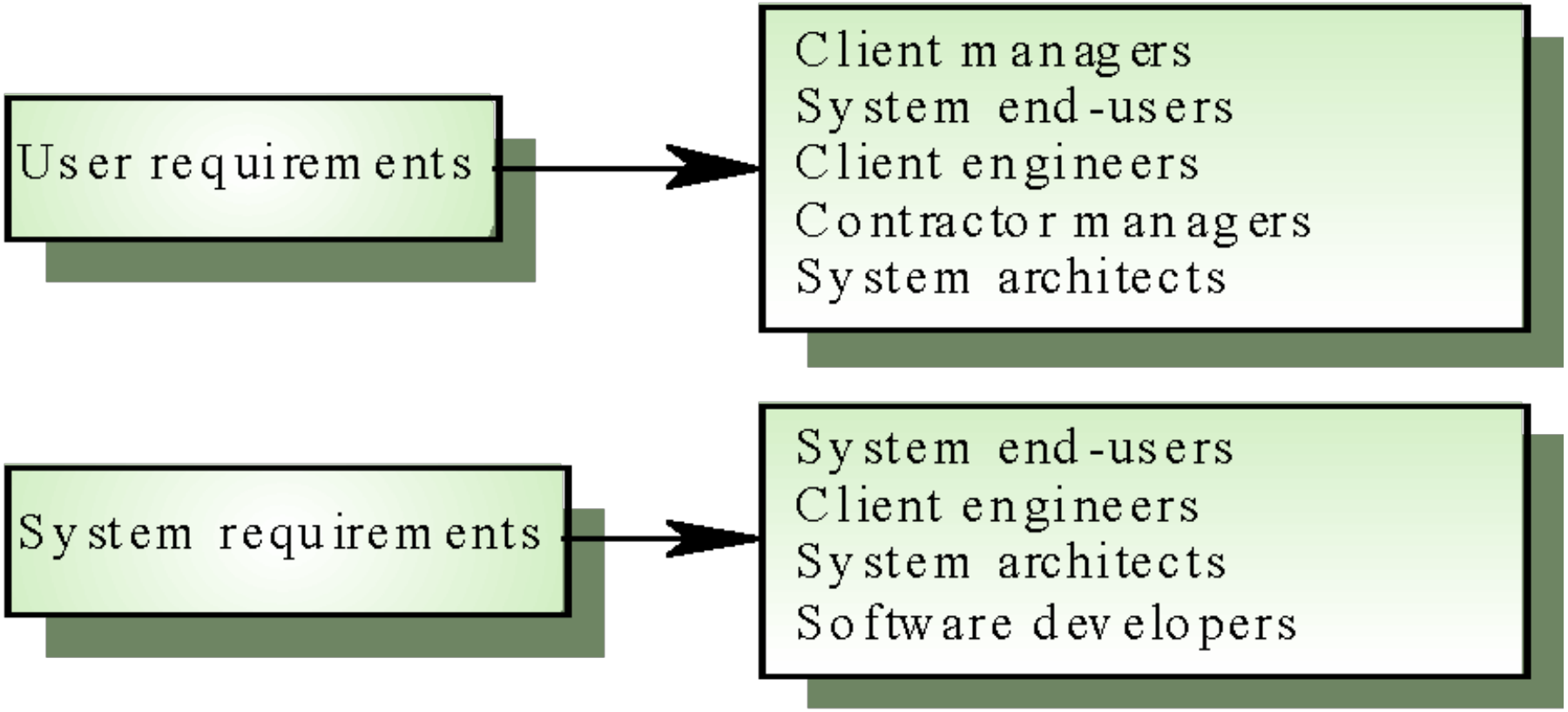
# Definizione dei termini

- **cliente** (*customer*, *client*)
  la persona od organizzazione che paga per la fornitura di un prodotto software
- **fornitore** (*supplier*, *contractor*)
  la persona od organizzazione che produce software per il cliente
- **utente finale** (*end-user*)
  la persona che interagisce direttamente con il prodotto software. Non corrisponde necessariamente al cliente

# Esempi di requisiti

- **Requisito utente**
  1. Il sistema software deve fornire un mezzo per rappresentare e visualizzare file esterni generati da altri tool
- **Requisito di sistema**
  1.1 L'utente deve avere la possibilità di definire il tipo dei file esterni
  1.2 Ad ogni tipo di file esterno deve essere associato il tool che lo ha generato
  1.3 Ogni tipo di file esterno deve essere rappresentato mediante una specifica icona sullo schermo
  1.4 L'utente deve avere la possibilità di definire l'icona che rappresenta il tipo di file esterno
  1.5 Quando l'utente seleziona un'icona che rappresenta un file esterno, deve poter essere eseguito il tool in grado di visualizzare il file

# Chi legge i requisiti?



# Categorie di requisiti

- **Requisiti funzionali**

  descrivono le funzionalità del sistema software, in termini di *servizi* che il sistema software deve fornire, di come il sistema software *reagisce* a specifici tipi di input e di come si *comporta* in situazioni particolari

  **Es.1** Il sistema software deve fornire un appropriato visualizzatore per i documenti memorizzati

  **Es.2** L'utente deve essere in grado di effettuare ricerche sia sull'intero insieme di basi di dati che su un loro sottoinsieme

  **Es.3** Ad ogni nuovo ordine deve essere associato un identificatore unico (Order_ID)

# Categorie di requisiti (2)

- **Requisiti non funzionali**

  descrivono le *proprietà* del sistema software in relazione a determinati servizi o funzioni e possono anche essere relativi al *processo*:

  - caratteristiche di *efficienza, affidabilità, safety,* ecc.
  - caratteristiche del *processo di sviluppo* (standard di processo, uso di ambienti CASE, linguaggi di programmazione, metodi di sviluppo, ecc.)
  - caratteristiche *esterne* (interoperabilità con sistemi di altre organizzazioni, vincoli legislativi, ecc.)

  **Es.1** Il tempo di risposta del sistema all'inserimento della password utente deve essere inferiore a 10 sec

  **Es.2** I documenti di progetto (*deliverable*) devono essere conformi allo standard XYZ-ABC-12345

  **Es.3** Il sistema software non deve rilasciare ai suoi operatori nessuna informazione personale relativa ai clienti, tranne nominativo e identificatore

# Categorie di requisiti (3)

- **Requisiti di dominio**

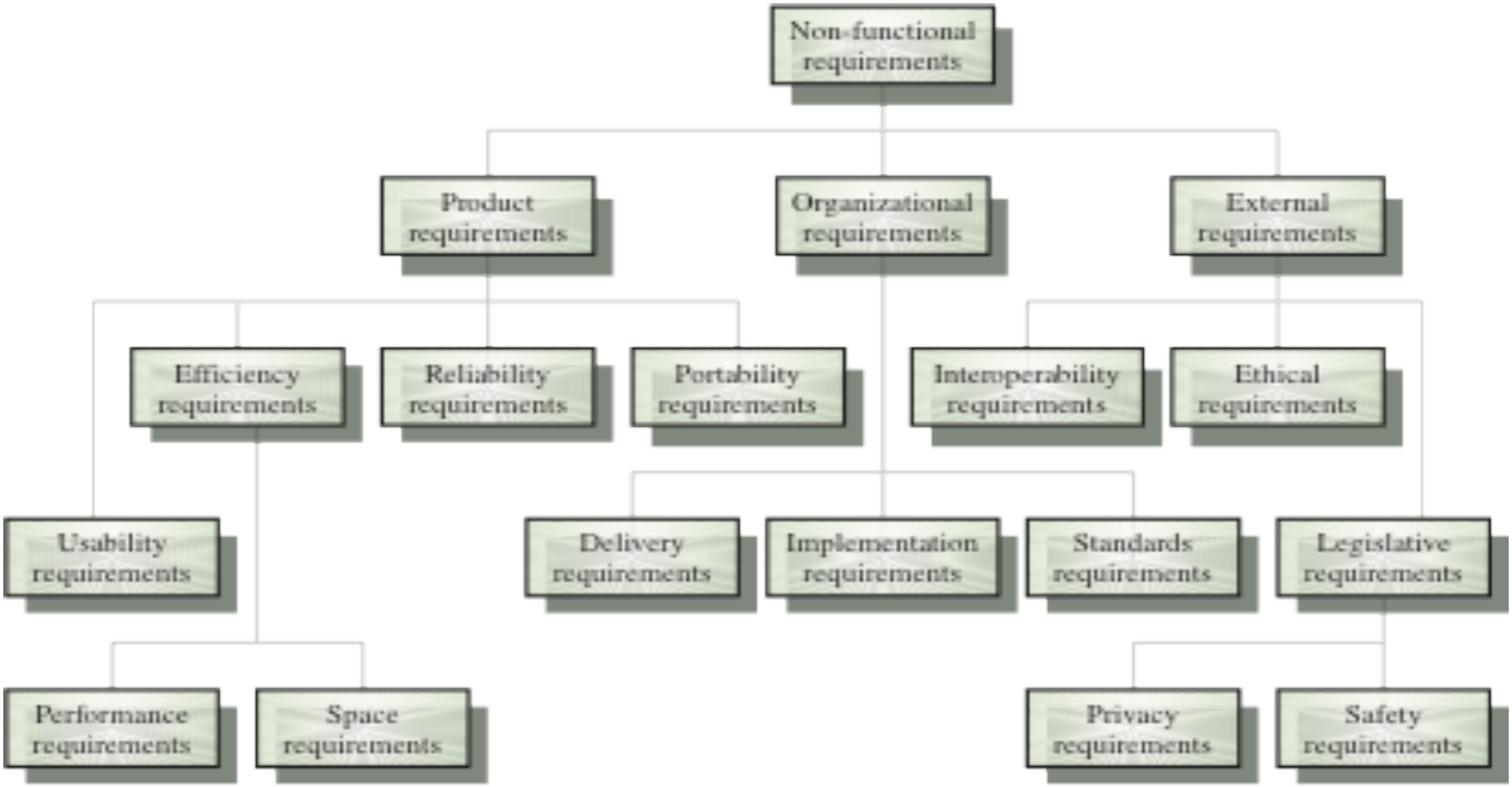
  requisiti derivati dal dominio applicativo del sistema software piuttosto che da necessità dettate dagli utenti

  - requisiti funzionali, nuovi o adattatati, relativi al particolare dominio applicativo
  - requisiti non funzionali, nuovi o adattati, relativi a standard esistenti o a procedure e regolamenti da applicare

  **Es.1** I documenti di rendiconto contabile, secondo la normativa XYZ.03, devono essere stampati alla ricezione e cancellati immediatamente

  **Es.2** L'interfaccia utente per l'accesso al database magazzino deve essere conforme allo standard ZX.01

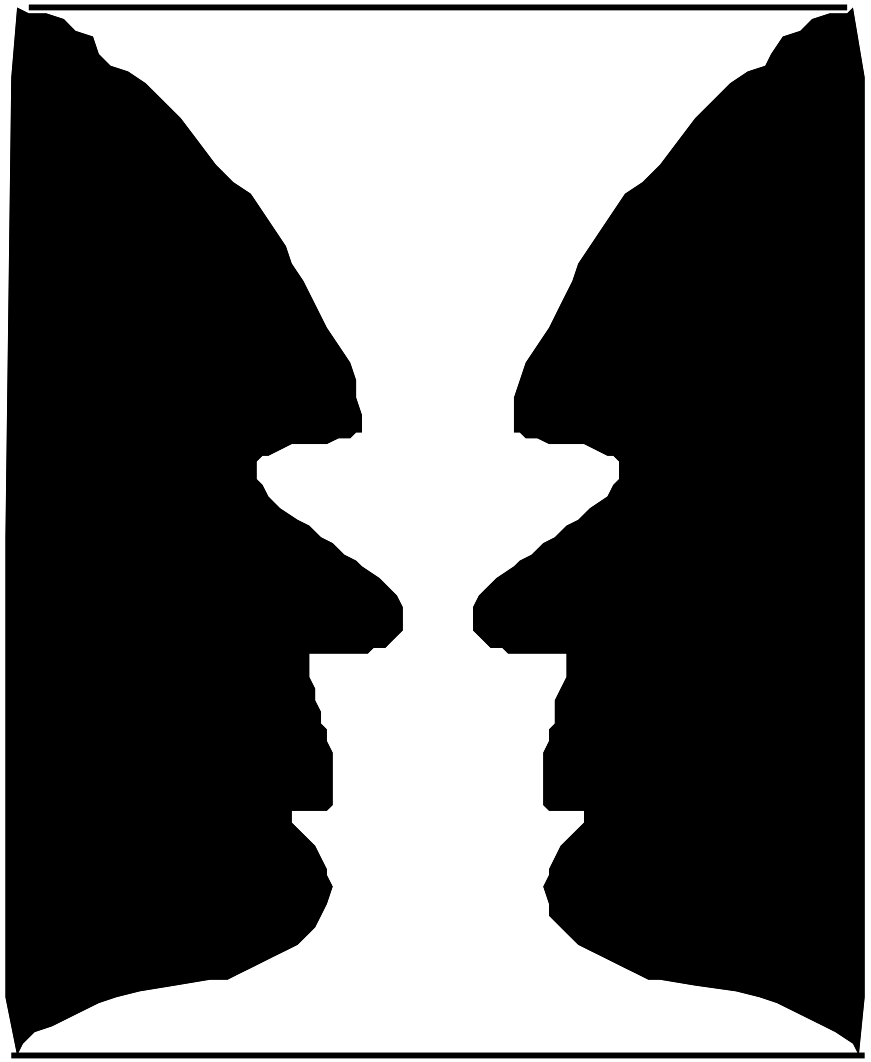
# Classificazione
# *requisiti non funzionali*



# Problemi con i requisiti software

**Ambiguità**

**Cosa vedete?**



# Problemi con i requisiti software (2)

- **Ambiguità**: requisiti interpretabili in modo differente

  *Esempio 1:* specificare un tempo senza fornire il riferimento al fuso orario (in un applicazione che gestisce chiamate intercontinentali)

  *Esempio 2*: significato di "appropriato visualizzatore"

  - *Interpretazione utente*: visualizzatore specifico per ogni tipo di documento
  - *Interpretazione sviluppatore*: generico visualizzatore di testo che mostri il contenuto del documento

- **Incompletezza**: i requisiti non includono la descrizione di tutte le caratteristiche richieste
- **Inconsistenza**: conflitti o contraddizioni nella descrizione delle caratteristiche del sistema

  *Esempio*

  - *Req 1*: ogni form di input non deve contenere più di 5 campi editabili dall'utente
  - *Req 2*: nella form di input relativa all'inserimento dei dati anagrafici l'utente deve introdurre i seguenti dati: nome, cognome, anno di nascita, luogo di nascita, indirizzo, telefono, fax, e-mail

# Verificabilità dei requisiti

- I requisiti non funzionali espressi in modo generico dall'utente (es. il sistema software deve essere *easy-to-use*) possono risultare non quantificabili e difficili da verificare

- E' quindi necessario esprimere i requisiti non funzionali usando una misura determinata che permetta di verificare quantitativamente se il requisito verrà soddisfatto dal sistema software

# Esempi di misure per requisiti

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

# Requisiti utente

- Descrivono requisiti funzionali e non funzionali, espressi in modo da risultare comprensibili agli utenti del sistema sprovvisti di conoscenze tecniche
- I requisiti utenti sono generalmente espressi in *linguaggio naturale*, tenendo in considerazione alcune linee guida:
  - usare un formato standard per tutti i requisiti
  - usare il linguaggio naturale in modo consistente (es. uso di "*deve*" per requisiti necessari e "*dovrebbe*" per requisiti desiderabili)
  - evidenziare le parti fondamentali di un requisito
  - evitare l'uso di termini tecnici

# Esempio di requisito utente

**3.5.1 Adding nodes to a design**

3.5.1.1 **The editor shall provide a facility for users to add nodes of a specified type to their design.**

3.5.1.2 The sequence of actions to add a node should be as follows:

1. The user should select the type of node to be added.
2. The user should move the cursor to the approximate node position in the diagram and indicate that the node symbol should be added at that point.
3. The user should then drag the node symbol to its final position.

*Rationale*: The user is the best person to decide where to position a node on the diagram. This approach gives the user direct control over node type selection and positioning.

*Specification*: ECLIPSE/WS/Tools/DE/FS/Section 3.5.1

# Requisiti di sistema (specifiche)

- Specifiche più dettagliate dei requisiti utente
- Sono usati come base per il progetto software
- Possono essere espressi facendo uso di notazioni differenti

| Notation | Description |
|---|---|
| ***Structured natural language*** | This approach depends on defining standard forms or templates to express the requirements specification. |
| ***Program description languages (PDL)*** | This approach uses a language like a programming language (*PDL*, *Program Description Language*) but with more abstract features to specify the requirements by defining an operational model of the system. |
| ***Graphical notations*** | A graphical language, supplemented by text annotations is used to define the functional requirements for the system. The graphical language is used to define *system models* |
| ***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. |

# Esempio di requisito di sistema

- basato su *form* in linguaggio naturale

*ECLIPSE/Workstation/Tools/DE/FS/3.5.1*

| | |
|---|---|
| **Function** | Add node |
| **Description** | Adds a node to an existing design. The user selects the type of node, and its position. When added to the design, the node becomes the current selection. The user chooses the node position by moving the cursor to the area where the node is added. |
| **Inputs** | Node type, Node position, Design identifier. |
| **Source** | Node type and Node position are input by the user, Design identifier from the database. |
| **Outputs** | Design identifier. |
| **Destination** | The design database. The design is committed to the database on completion of the operation. |
| **Requires** | Design graph rooted at input design identifier. |
| **Pre-condition** | The design is open and displayed on the user's screen. |
| **Post-condition** | The design is unchanged apart from the addition of a node of the specified type at the given position. |
| **Side-effects** | None |
| **Definition**: | ECLIPSE/Workstation/Tools/DE/RD/3.5.1 |

# Esempio di requisito di sistema (2)

- basato su *PDL* (*Java-like*)

```
class ATM {
 // declarations here
 public static void main (String args[]) throws InvalidCard {
    try {
       thisCard.read () ;  // may throw InvalidCard exception
       pin = KeyPad.readPin () ; attempts = 1 ;
       while ( !thisCard.pin.equals (pin) & attempts < 4 )
          { pin = KeyPad.readPin () ;  attempts = attempts + 1 ;
          }
          if (!thisCard.pin.equals (pin))
            throw new InvalidCard ("Bad PIN");
       thisBalance = thisCard.getBalance () ;
       do {  Screen.prompt (" Please select a service ") ;
          service = Screen.touchKey () ;
          switch (service) {
            case Services.withdrawalWithReceipt:
                          receiptRequired = true ;

            .........
```

# Esempio di requisito di sistema (3)

- specifica di interfaccia basata su *PDL*

```
interface PrintServer {

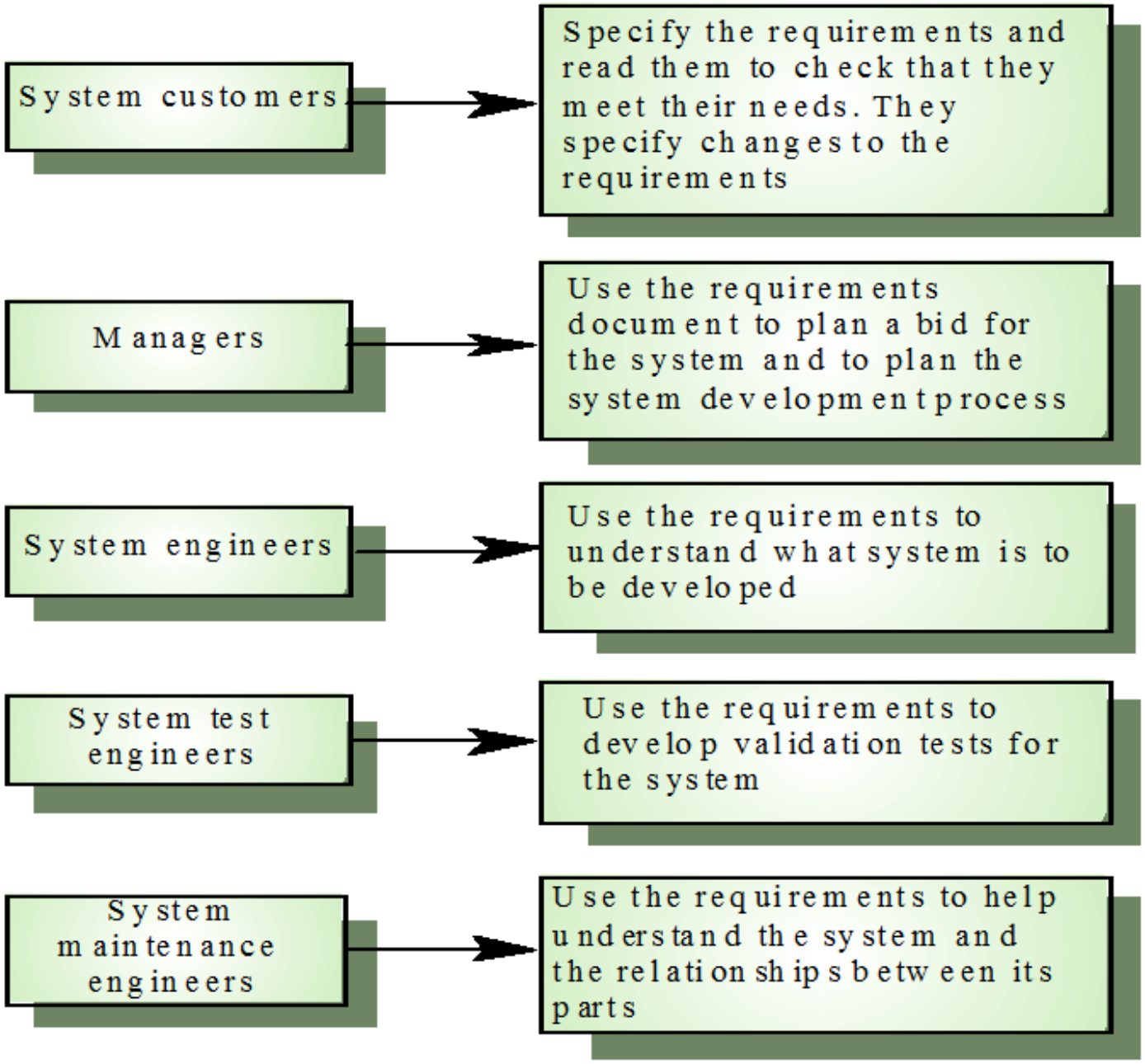
// defines an abstract printer server
// requires: interface Printer, interface PrintDoc
// provides: initialize, print, displayPrintQueue, cancelPrintJob, switchPrinter

    void initialize ( Printer p ) ;
    void print ( Printer p, PrintDoc d ) ;
    void displayPrintQueue ( Printer p ) ;
    void cancelPrintJob (Printer p, PrintDoc d) ;
    void switchPrinter (Printer p1, Printer p2, PrintDoc d) ;
} //PrintServer
```

# Il documento di *analisi dei requisiti* (o documento di *specifica*)

- Documento ufficiale che descrive in dettaglio le caratteristiche del sistema da sviluppare
- Include sia la definizione dei requisiti che la loro specifica
- Descrive **COSA** il sistema deve fornire (*dominio del problema*) e non **COME** il sistema deve essere sviluppato (*dominio della soluzione*)

# Utenti del documento di analisi dei requisiti



# Struttura del documento di specifica

Basata sullo standard **IEEE 830-1998**
(*IEEE Recommended Practice for Software Requirements Specifications*) pag 1/2

**Preface**

expected readership, version history, changes summary

**Introduction**

purpose, brief description of the system, interaction with other systems, scope within the business context

**Glossary**

definition of technical terms used in the document

**User requirements definition**

functional and non-functional user requirements

**System architecture**

high-level overview of the system components

**System requirements specification**

functional and non-functional system requirements

# Struttura del documento di specifica

Basata sullo standard **IEEE 830-1998**
(*IEEE Recommended Practice for Software Requirements Specifications*) pag 2/2

**System models**

description of the relationships between the system components and the system and its environment

**System evolution**

assumptions on which the system is based and anticipated changes (hardware evolution, user needs changes, etc.)

**Appendices**

specific information related to the application which is being developed (ex. HW and DB descriptions)

**Index**

table of contents, alphabetic index, list of diagrams, etc.