Software Requirements Specification

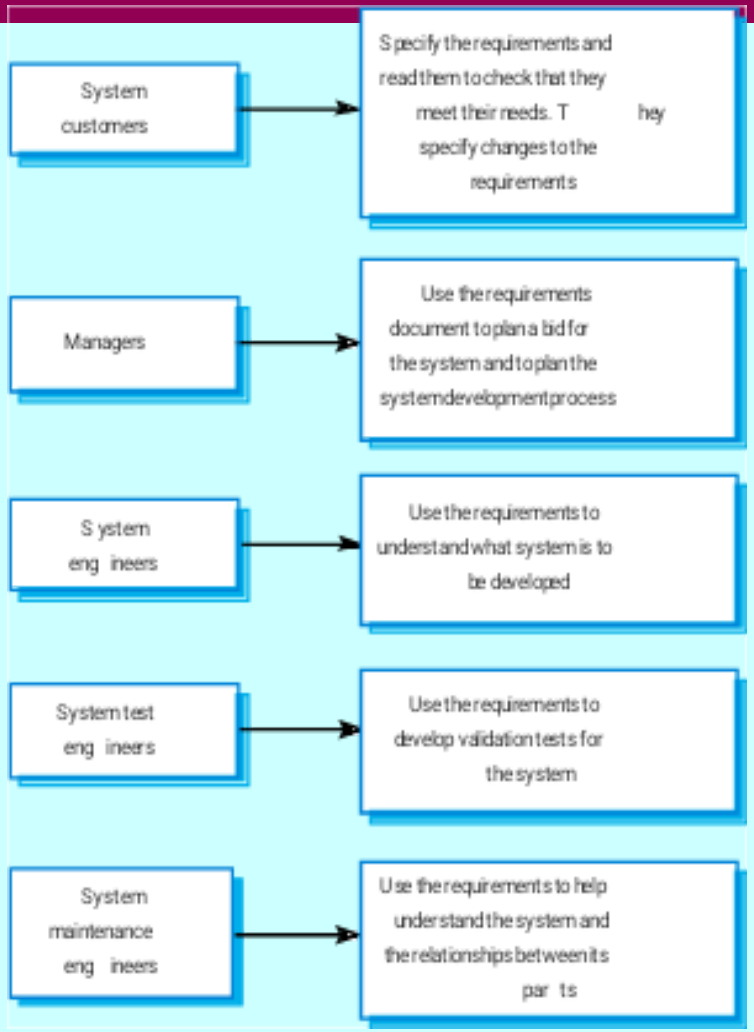
Software Requirements Specification: A Contract Document

- Requirements document is a reference document.
- SRS document is a contract between the development team and the customer.
  - Once the SRS document is approved by the customer,
    - any subsequent controversies are settled by referring the SRS document.

SW Requirements Specification

- Purpose of SRS
  - Interface (communication) between the Customer, Analyst, designers, system developers, testers, maintainers, ...
  - agreement between Purchaser and Supplier
  - firm foundation for the design phase
  - support system testing activities
  - support project management activities
  - controlling the evolution of overall system

Users of a requirements document



Example- ATM stakeholders

- Bank customers
- Bank managers
- Counter staff
- Database administrators
- Security managers
- Marketing department
- Hardware and software maintenance engineers
- Banking regulators
- Representatives of other banks

Problems of requirements analysis

- Stakeholders don’t know exactly, that what they really want.
- Stakeholders express requirements in their own terms.
- Different stakeholders may have conflicting requirements.
- Organisational and political factors may influence the system requirements.
- The requirements may change during the analysis process. New stakeholders may emerge and the business environment can also change.

SRS Document (CONT.)

- The SRS document is known as black-box specification:
  - the system is considered as a black box whose internal details are not known.
  - only its (system's) visible external (i.e. input/ output) behaviour is documented.

SRS Document (CONT.)

- SRS document concentrates on:
  - what needs to be done
  - carefully avoids the solution ("how to do") aspects.
- The SRS document serves as a contract
  - between development team and the customer.
  - Should be carefully written

SRS Document (CONT.)

- The requirements at the first stage:
  - written using end-user terminology.
  - later a formal requirement specification may be developed from it.

Software Requirements Specification (SRS)

- Defines the customer's requirements in terms of :
  - Functional (all required functions)
  - Non functional:
    - Performance (efficiency, load etc.)
    - External interfaces
    - Design constraints
- The SRS is the basis of bonding between the purchaser and supplier

Specification Principles

- Separates functionality from implementation
- Develops model of desired behavior of the system
- Define the environment in which system operates
- Specifications must be open to incompleteness
- Content & structure of a specifications should be open to change

What is **not included** in an SRS ?

- Project requirements
  - cost, delivery schedules, staffing, reporting procedures (All are included in a separate contract document and not in SRS)
- Design solutions
  - partitioning of SW into modules, choosing data structures
- Product assurance plans
  - Quality Assurance procedures, Configuration Management procedures, Verification & Validation procedures

Benefits of SRS

- Forces the users to consider their specific requirements **carefully.**
- **Enhances communication** between the Purchaser and System developers.
- Provides a **firm foundation** for the system design phase.
- Enables planning of validation, verification, and acceptance procedures.
- Enables project planning e.g. estimates of cost and time, resource scheduling (Feasibility, SPM activity)
- Usable till/during maintenance phase

Types of Requirements

- Functional requirements
- Non functional requirements
  - Performance requirements
  - Interface requirements
  - Design constraints
  - Other requirements

Functional Requirements

- Transformations (inputs, processing, outputs)
- Requirements for sequencing and parallelism (dynamic requirements)
- Data
  - Inputs and Outputs
  - Stored data
- Exception handling
- Nature of function: Mandatory/ Desirable/ Optional / Volatile / Stable

Performance Requirements

- Capacity
  - no. of simultaneous users, processing requirements for normal and peak loads, storage capacity, spare capacity. (e.g. bandwidth, os etc) (scalability)
- Response time,
- System priorities for users and functions,
- System efficiency,
- Availability and Fault recovery,
- Best case, average case and worst case analysis,
- Dead lines / maximum limits
- E.g. ATM, Defense applications, Medical applications, any web based application and RTS etc.

* *All these requirements should be stated in measurable terms so that they can be verified.*

External Interface Requirements

- User interfaces
  - E.g. if display terminal used, specify required screen formats, menus, report layouts, function keys
- Hardware interfaces
  - characteristics of the interface between the SW product and HW components of the system
- Software interfaces
  - specify the use (connectivity) of other SW products eg. OS, DBMS, other SW

Design Constraints

- SW design constraints
  - standards for design, coding, naming, etc.
  - SW interfaces (to OS, DBMS, other SW)
  - use a specific application package
  - constraints on program size, data size etc.
- HW design constraints
  - specific type of HW, reliability requirements
  - HW interfaces
  - requirements for spare capacity or spare performance

Design Constraints (contd..)

- User-interface design constraints
  - features of operator/user with details of working environment
  - any special features required

Other Requirements

- Security requirements
- Safety requirements
- Environmental aspects
- Reusability
- Training
- ...

SRS Standards

- ANSI/IEEE SRS Standard 830-1984
- BS 6719: 1986
- European Space Agency Standards (ESA PSS-05-0, Jan 1987)
- US DoD-Std-7935A
- ...

SRS Prototype Outline

1. Introduction
2. General description
3. Specific Requirements
4. Appendices

   Index

SRS Prototype Outline…

[IEEE SRS Standard]

1. Introduction

- 1.1 Purpose
- 1.2 Scope
- 1.3 Definitions, Acronyms and Abbreviations
- 1.4 References
- 1.5 Overview

SRS - Introduction Section

- Purpose
  - define the purpose of the particular SRS
  - specify the intended audience for the SRS
- Scope
  - identify the SW products to be produced by name
  - explain what the SW product will do, and if necessary, what it will not do
  - describe the application of the SW being specified. ie. benefits, objectives, goals as precisely as possible
- Overview
  - describe what the rest of the SRS contains

SRS Prototype Outline…

[IEEE SRS Standard]

2. General description

- 2.1 Product perspective
- 2.2 Product function summary
- 2.3 User characteristics
- 2.4 General constraints
- 2.5 Any Assumptions and dependencies

Product Perspective

- State whether the product is independent and totally self contained, or
- If the product is component of a larger system then:
  - describe the functions of each component of the larger system and identify interfaces
  - overview of the principal external interfaces of this product
  - overview of HW and peripheral equipment to be used
- Give a block diagram showing the major components of the product, interconnections,

Product Functions

- Provide a summary of functions the SW will perform
- The functions should be organized in such a way that they are understandable by the user

User Characteristics

- Describe the general characteristics of the eventual users of the product. (**such as educational level, experience and technical expertise**)

General Constraints

- Regulatory policies
- HW limitations
- Interfaces to other applications
- Parallel operation
- Audit functions
- Control functions
- Criticality of the application
- Safety and security considerations

SRS Prototype Outline…

[IEEE SRS Standard]

3. Specific Requirements

- Functional requirements
- External interface requirements
- Performance requirements
- Design constraints
- Attributes eg. security, availability, maintainability.
- Other requirements

Appendices

Index

Functional Requirements

- Introduction
  - describe purpose of the function and the approaches and techniques employed
- Inputs and Outputs
  - sources of inputs and destination of outputs
  - quantities, units of measure, ranges of valid inputs and outputs
  - timing

Functional Requirements

- Processing
  - validation of input data
  - exact sequence of operations
  - responses to abnormal situations
  - any methods (eg. equations, algorithms) to be used to transform inputs to outputs

External Interface Requirements

- User interfaces,
- Hardware interfaces,
- Software interfaces,
- Communications/interfaces between software and hardware
- Other requirements
  - database: frequency of use, accessing capabilities, static and dynamic organization, retention requirements for data
  - operations: periods of interactive and unattended operations, backup, recovery operations
  - site adaptation requirements

Appendices

- Not always necessary
- It may include:
  - sample I/O formats
  - DFD, ERD documents
  - results of user surveys, cost analysis studies
  - supporting documents to help readers of SRS

Characteristics of a **Good** SRS

- Unambiguous
- Complete
- Verifiable
- Consistent
- Modifiable
- Traceable
- Usable till/during the Operation and Maintenance phase

Examples of Requirements statements

- *The data set will contain an end of file character.*
- *The product should have a good human interface.*
- *The program shall never enter an infinite loop.*
- *The output of the program shall usually be given within 10 secs.*
- *The output of a program shall be given within 20secs of event X 60% of the time.*

7 Ambiguous

7 Non-verifiable

7 Non-verifiable

7 Non-verifiable

3 Verifiable

Examples of **Bad** SRS Documents

- Unstructured Specifications:
  - Narrative **essay** --- one of the worst types of specification document:
    - Difficult to change,
    - difficult to be precise,
    - scope for contradictions, etc.

Examples of **Bad** SRS Documents…

- Noise:
  - Presence of text containing information irrelevant to the problem. (**less imp things are given more emphasis**)
- Silence:
  - aspects important to proper solution of the problem are omitted. (**important things are not properly covered)**

Examples of Bad SRS Documents…

- Overspecification:
  - Addressing "**how to**" aspects
  - For example, "Library member names should be stored in a sorted descending order"
  - **Overspecification restricts the solution space for the designer**.
- Contradictions:
  - Contradictions might arise
    - **if the same thing described at several places in different ways.**

Examples of Bad SRS Documents

- Ambiguity:
  - Literary expressions
  - Unquantifiable aspects, e.g. “good user interface”
- Forward References:
  - References to aspects of problem
    - defined only later on in the text.
- Wishful Thinking:
  - Descriptions of aspects
    - for which realistic solutions will be hard to find.

Complete

- **All significant requirements should be included.**
- Definition of responses of the SW to all realizable classes of input data in all situations.
- Conformity to a standard.
- Full labeling and referencing of all figures, tables etc. and definition of all terms and units of measure

Verifiable / testable

- A requirement is verifiable if and only if there exists some finite cost effective process with which a person or machine can check that the SW meets the requirement.

Consistent

- No two requirements are in conflict

Modifiable

- Structure and style of SRS is such that changes to requirements can be made easily, completely and consistently.
  - SRS organisation -- table of contents, index, explicit cross-referencing
  - no redundancy

Traceable

- An SRS is traceable if the origin of each requirement is clear and it facilitates the referencing of each requirement in future.
- Backward traceability
  - requirement explicitly referencing its source in previous documents
- Forward traceability
  - each requirement has a unique name or reference number and it can be traced to design documents, program

SRS Review

- Formal Review done by Users, Developers, Managers, Operations personnel
- To verify that SRS confirms to the actual user requirements
- To detect defects early and correct them.

Sample SRS Checklist

- Are all HW resources defined ?
- Have response times been specfied for functions ?
- Have all the HW, external SW and data interfaces been defined ?
- Is each requirement testable ?
- Is the initial state of the system defined ?
- Are the responses to exceptional conditions specified ?
- Are possible future modifications specified ?