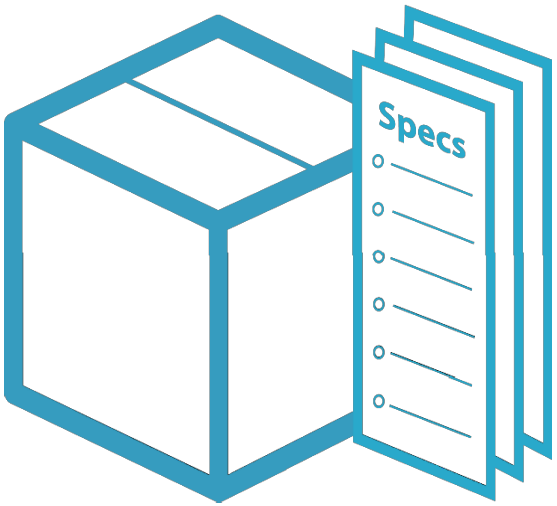
Unit 2.4
Software Requirement

Specification

Requirement Analysis & Specification

Requirements Engineering Tasks cont.

5 Specification



- Create analysis model
- It may be written document, set of graphical models, formal mathematical model, collection of user scenarios, prototype or collection of these
- SRS (Software Requirement Specification) is a document that is created when a detailed description of all aspects of software to build must be specified before starting of project

Software Requirements Specification

- Software Requirement Specification (SRS is a document that completely describes what the proposed software should do without describing how software will do it.
- It contains:
  - a complete information description
  - a detailed functional description
  - a representation of system behaviour
  - an indication of performance requirements and design constraints
  - appropriate validation criteria
  - other information suitable to requirements
- SRS is also helping the clients to understand their own needs.

Characteristics of a Good SRS

- SRS should be accurate, complete, efficient, and of high quality
  - so that it does not affect the entire project plan.
- An SRS is said to be of high quality when the developer and user easily understand the prepared document.
- Characteristics of a Good SRS:
  - Correct
    - SRS is correct when all user requirements are stated in the requirements document.
    - Note that there is no specified tool or procedure to assure the correctness of SRS.
  - Unambiguous
    - SRS is unambiguous when every stated requirement has only one interpretation.

Characteristics of a Good SRS (Cont…)

- Complete
  - SRS is complete when the requirements clearly define what the software is required to do.
- Ranked for Importance/Stability
  - Al requirements are not equally important, hence each requirement is identified to make differences among other requirements.
  - Stability implies the probability of changes in the requirement in future.
- Modifiable
  - The requirements of the user can change, hence requirements document should be created in such a manner that those changes can be modified easily.
- Traceable
  - SRS is traceable when the source of each requirement is clear and facilitates the reference of each requirement in future.

Characteristics of a Good SRS (Cont…)

- Verifiable
  - SRS is verifiable when the specified requirements can be verified with a cost-effective process to check whether the final software meets those requirements.
- Consistent
  - SRS is consistent when the subsets of individual requirements defined do not conflict with each other.

Standard Template for writing SRS

- Front Page

  Software Requirements Specification

  for

  <Project> Version

  <no.> Prepared by

  <author>

  <organization>

  <date created>
- Table of Contents
- Revision History

Standard Template for writing SRS (Cont…)

1. Introduction
    - 1.1 Purpose
    - 1.2 Document Conventions
    - 1.3 Intended Audience and Reading Suggestions
    - 1.4 Project Scope
    - 1.5 References
2. Overall Description
    - 2.1 Product Perspective
    - 2.2 Product Features
    - 2.3 User Classes and Characteristics
    - 2.4 Operating Environment
    - 2.5 Design and Implementation Constraints

Standard Template for writing SRS (Cont…)

2.6 User Documentation

2.7 Assumptions and Dependencies

3. System Features

3.1 System Feature 1

3.2 System Feature 2 (and so on)

4. External Interface Requirements

4.1 User Interfaces

4.2 Hardware Interfaces

4.3 Software Interfaces

4.4 Communications Interfaces

Standard Template for writing SRS(Cont…)

5. Other Nonfunctional Requirements
    5.1 Performance Requirements
    5.2 Safety Requirements
    5.3 Security Requirements
    5.4 Software Quality Attributes
6. Other Requirements

Appendix A: Glossary
Appendix B: Analysis Models
Appendix C: Issues List

Problems Without SRS

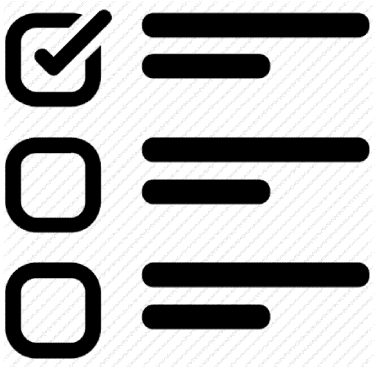
- Without developing the SRS document, the system would not be properly implemented according to customer needs.
- Software developers would not know whether what they are developing is what exactly is required by the customer.
- Without SRS, it will be very difficult for the maintenance engineers to understand the functionality of the system.
- It will be very difficult for user document writers to write the users' manuals properly without understanding the SRS.

Summary

- Requirements Engineering
  - Requirements Engineering Tasks
  - Eliciting Requirements
  - Collaborative Requirements Gathering
  - Quality Function Deployment
  - Usage Scenarios
  - Elicitation Work Products
- Requirements Analysis Model
  - Use-Case Diagram
  - Class Diagram
  - State Diagram
  - Data Flow Diagram
- Negotiating Requirements
- Functional and non-functional requirements
- SRS

Requirements Engineering Tasks cont.

6 Validation



- Ensure quality of requirements
- Review the requirements specification for errors, ambiguities, omissions (absence) and conflicts

Validation. The work products produced as a consequence of requirements engineering are assessed for quality during a validation step. Requirements validation examines the specification to ensure

•that all software requirements have been stated unambiguously;

•that inconsistencies, omissions, and errors have been detected and corrected;

•that the work products conform to the standards established for the process, the project, and the product

Requirements Engineering Tasks

7 Requirements Management

- It is a set of activities to identify, control & trace requirements & changes to requirements (Umbrella Activities) at any time as the project proceeds.

Requirements management is the process of collecting, analyzing, refining, and prioritizing product requirements and then planning for their delivery. Big Word Requirement Change Management and relationship of change in requirement (Impact).

Requirements management does not end with product release. From that point on, the data coming in about the application's acceptability is gathered and fed into the Investigation phase of the next generation or release. Thus the process begins again.

Requirements Analysis is Hard

"The hardest single part of building a software system is deciding what to build. No part of the work so cripples the resulting system if done wrong."

"Fred" Brooks Jr. is an American computer architect, software engineer, and computer scientist, best known for managing the development of IBM's System/360 family of computers

"The seeds of major software disasters are usually sown in the first three months of commencing the software project."

Capers Jones is an American specialist in software engineering methodologies