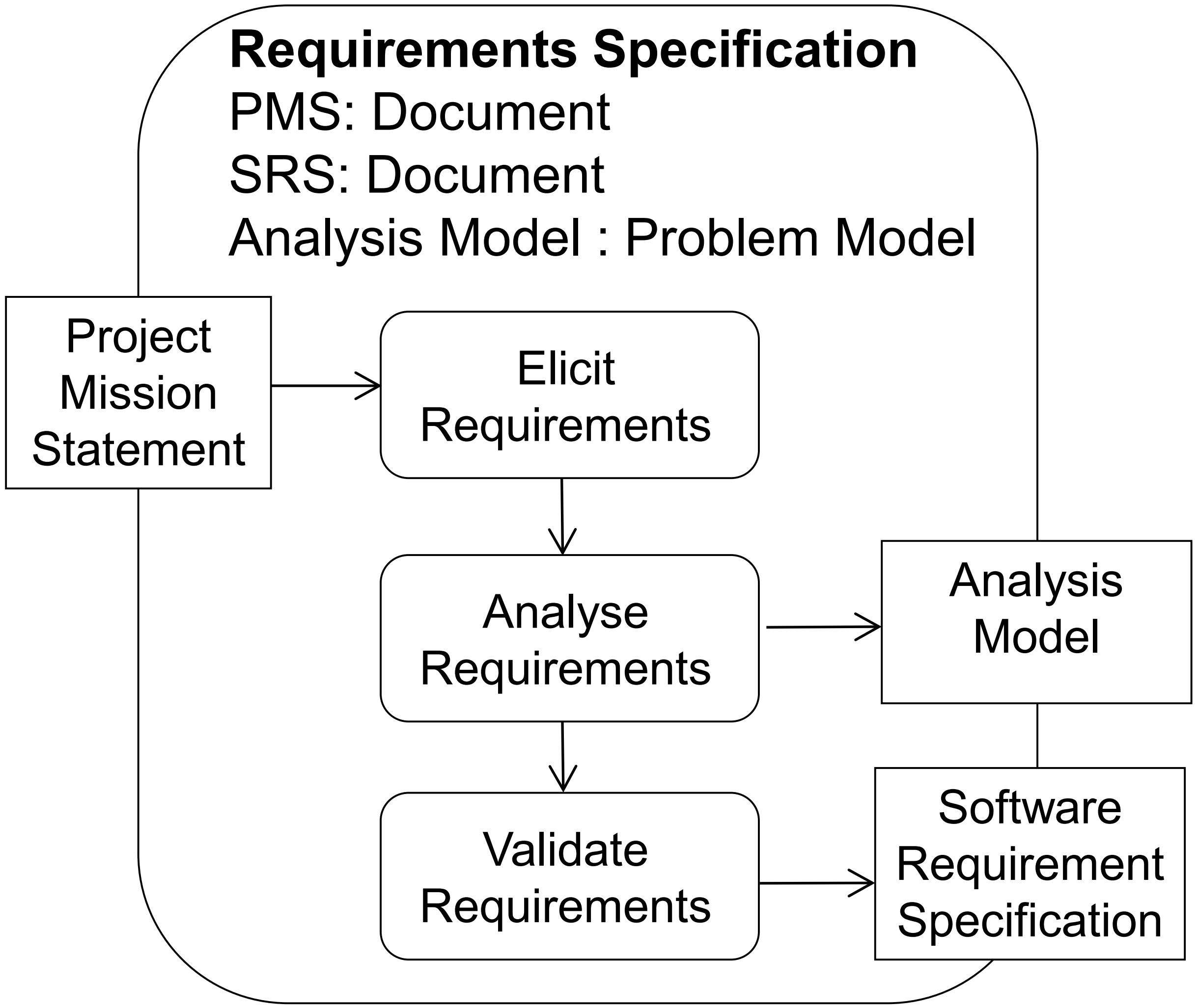
Software Engineering

CE2006/CZ2006

Software Requirements Specification

Requirements Specification Activities


Requirements Specification
PMS: Document
SRS: Document
Analysis Model : Problem Model
Project Mission Statement
Elicit Requirements
Analyse Requirements
Validate Requirements
Analysis Model
Software Requirement Specification

Software Requirements Specification is

1. The output of the Requirements Elicitation and Requirements Analysis.
2. An organization’s understanding (in writing) of a customer or potential client’s system requirements and dependencies *at a particular point in time* (usually) prior to any actual design or development work.
3. A two-way insurance policy that assures that both the client and the organization understand the other’s requirements from that perspective at a given point in time.
4. A statement in precise and clear language describing those functions and capabilities a software system must provide, as well as stating any required constraints by which the system must abide.
5. A blueprint for completing a project with as little cost growth as possible.
6. Detailed functional and non-functional requirements. it DOES NOT contain any design suggestions.

Purpose of the SRS is to

1. Give the customer assurance that the development organization understands the issues or problems to be solved and the software behaviour necessary to address those problems.
2. Decomposes the problem into component parts.
3. Serve as an input to the design specification so must contain sufficient detail in the functional requirements so that a design solution can be devised.
4. Serve as the parent or master document for testing and validation strategies that will be applied to the requirements for verification in the testing stage.

How and Who should write the SRS?

1. It should be written in **natural language**, in an **unambiguous** manner that may also include diagrams as necessary.

2. At the same time the detailed functional and non-functional **requirements need to be defined and understood** by the development team.

3. **Development team** members (programmers and managers) are **seldom the best writers of formal documents.** However, they often have to write it!

4. It is good practice to involve a **technical writer** with the development team in the drafting of the SRS because they are usually better at assessing and planning documentation projects and better meet customer document needs.

Contents of the SRS

1. Each organisation will have its own template for an SRS.

2. There are key content that must be addressed in an SRS

3. There is an IEEE standard

   830-1998 - IEEE Recommended Practice for Software Requirements Specifications.

   This is a document that you can download from IEEEXplore.

Contents of the SRS

1) Product Description
    a) Purpose of the System (mission statement)
    b) Scope of the System
    c) Users and Stakeholders
    d) Assumptions and Constraints
2) Functional Requirements
3) Non-Functional Requirements
4) Interface Requirements
    a) User
    b) Hardware
    c) Software
5) Data Dictionary

See SRS Template on NTULearn course website.

Contents of the SRS

Give Project Mission Statement and state the purpose of the system in terms of what it does from a high level perspective. This should not be a long statement. 100-200 words should be sufficient.

Identify the system and the software to which this document applies, including, as applicable, identification number(s),title(s)

1) **Product Description**
   a) **Purpose of the System**
   b) **Scope of the System**
   c) **Users and Stakeholders**
   d) **Assumptions and Constraints**

Identify all the people and organisations who have an interest in the system. Users, clients, funding bodies, standards bodies, ……..

Include any information that may affect the SRS. E.g. assuming that any external interfaces are fully specified and will not change, regulations that have to be met……

Contents of the SRS

2) Functional Requirements

State the functional requirements in precise numbered statements and use

- *Activity Diagrams*
- *Use Case Diagrams with Use Case Descriptions*
- *Class Diagrams*
- *Sequence Diagrams*
- *Communication Diagrams*
- *State Machine Diagrams and*
- *Prototype UI as appropriate*

Contents of the SRS

3) Non-Functional Requirements

State the non-functional requirements in precise numbered statements covering any necessary

- *Performance requirements*
- *Any standards requirements*
- *Reliability*
- *Availability*
- *Security*
- *Maintainability*
- *Portability*

Contents of the SRS

e.g. required screen formats, page or window layouts, content of any reports or menus, or availability of programmable function keys

4) Interface Requirements

a) **User**

b) **Hardware**

c) **Software**

e.g. number of ports, instruction sets, etc.

e.g., interfaces to a data management system, an operating system, a mathematical package …..

Contents of the SRS

5) Data Dictionary

Unambiguously defines all terms, phrases and abbreviations used in the SRS

Language quality characteristics of an SRS

1. **Correct.** Each requirement must accurately describe the functionality to be delivered. Only user representatives can determine the correctness of user requirements, which is why it is essential to include them in inspections of the requirements.

2. **Feasible.** It must be possible to implement each requirement within the known capabilities and limitations of the system and its environment.

3. **Necessary.** Each requirement should document something the customers really need or something that is required for conformance to an external requirement, an external interface, or a standard. If you cannot identify the origin of the requirement, perhaps the requirement is not really necessary.

4. **Unambiguous.** The reader of a requirement statement must be able to draw only one interpretation of it. Also, multiple readers of a requirement must arrive at the same interpretation.

5. **Verifiable.** See whether you can devise tests or use other verification approaches, such as inspection or demonstration, to determine whether each requirement is properly implemented in the product.

Language quality characteristics of an SRS

6. **Complete.** No requirements or necessary information should be missing. It is hard to spot missing requirements because they aren't there. Organize the requirements hierarchically in the SRS to help reviewers understand the structure of the functionality described, so it will be easier for them to tell if something is missing.

7. **Consistent.** Consistent requirements do not conflict with other software requirements or with higher level (system or business) requirements. Disagreements among requirements must be resolved before development can proceed.

8. **Modifiable.** You must be able to revise the SRS when necessary and maintain a history of changes made to each requirement. This means that each requirement be uniquely labeled and expressed separately from other requirements so you can refer to it unambiguously.

9. **Traceable.** You should be able to link each software requirement to its source, which could be a higher-level system requirement, a use case, or a voice-of-the-customer statement.

Good words/phrases to use in an SRS

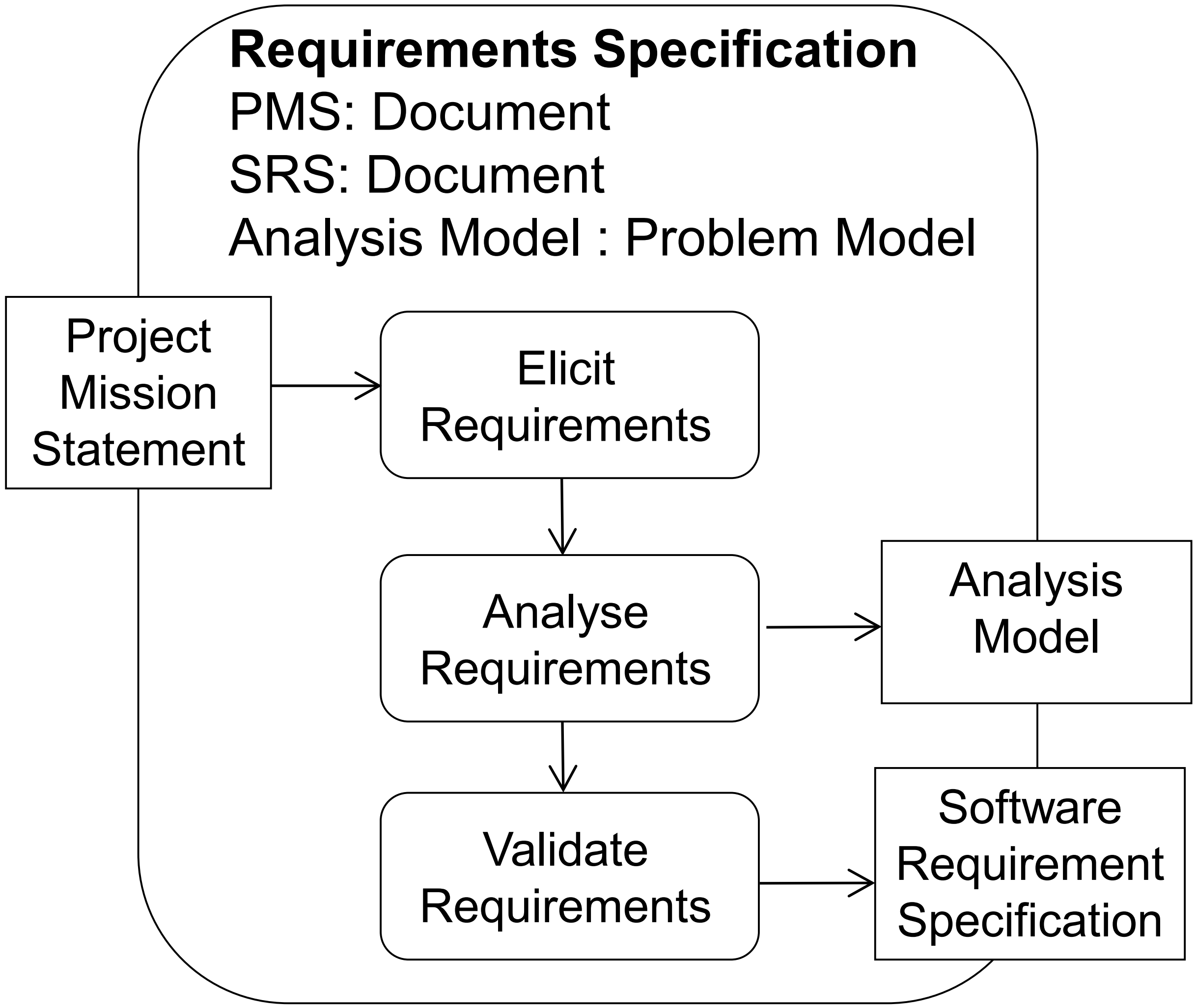
Shall	Used to dictate the provision of a functional capability.
Must or Must not	Most often used to establish performance requirement or constraints.
Is required to	Used as an imperative in SRS statements.
Are applicable	Used to include, by reference, standards, or other documentation as an addition to the requirement being specified.
Responsible for	Used as an imperative in SRSs that are written for systems with pre-defined architectures.
Will	Used to cite things that the operational or development environment is to provide to the capability being specified.

Words/phrases NOT to use in an SRS

Avoid these words/phrases like the following as they can create uncertainty

adequate
as a minimum
be capable of…
as applicable
as appropriate
effective
if possible
if practical
timely
normal

Summary: Requirements Specification Activities


Requirements Specification
PMS: Document
SRS: Document
Analysis Model : Problem Model
Project Mission Statement
Elicit Requirements
Analyse Requirements
Analysis Model
Validate Requirements
Software Requirement Specification