Unit – IV

Software Requirements Analysis and Specification

4.1. Value of a Good SRS

- ✓ An SRS establishes the basis for agreement between the client and the supplier on what the software product will do.
  - i.e. it acts as the contract between the client (or the customer) and the developer (the supplier).
  - So, through SRS, the client clearly describes what he expects from the supplier, and the developer clearly understands what capabilities to build in the software.
- ✓ An SRS provides a reference for validation of the final product.
  - i.e. the SRS helps the client to determine whether the software meets the requirements.
  - Without SRS it is not possible for the client to check whether the software meets all the stated requirements in the SRS.
  - Similarly, without SRS it is not possible for the developer to convince the client that all the requirements have been fulfilled.
- ✓ A high-quality SRS is a prerequisite to high-quality software.
  - Many errors are made during the requirements phase. Good SRS can minimize changes and errors in the requirements.
  - Cost of fixing errors in requirements, design, coding, testing and maintenance increases exponentially.
- ✓ A high-quality SRS reduces the development cost.
  - Improving the quality of requirements substantially reduces the development cost.

4.2. Requirement Process

- The requirement process is the sequence of activities that need to be performed in the requirements phase in order to produce high quality SRS.
- The requirements process typically consists of three basic tasks:
  1. Problem or requirements analysis

2. Requirements specification and
3. Requirements validation

Problem or requirements analysis

- It often starts with a high-level problem statement.
- The basic purpose of this activity is to obtain a thorough understanding of what the software needs to provide.
- Frequently, during analysis, the analyst will have a series of meetings with the clients and end users.
- In the early meetings, the clients and end users will explain to the analyst about their work, their environment, and their needs as they perceive them.
- Any documents describing the work may be given along with the outputs of the existing methods of performing the tasks.
- In these early meetings, the analyst is basically the listener, absorbing the information provided.
- Once the analyst understands the system to some extent, he uses the next few meetings to seek clarifications of the parts he does not understand.
- In the final few meetings, the analyst essentially explains to the client what he understood about the system and cross checks the clients whether they are correct.

Requirements Specification

- After obtaining and analyzing the requirements, the next activity is to clearly specify the requirements.
- In this, we systematically organize the requirements.
- For many years requirements are specified using stories or scenarios or DFDs. But, today use case approach is followed to specify the requirements.
- Issues such as representation, specification languages, and tools are addressed during this activity.

Requirements validation

- It ensures that the SRS specifies all the requirements of the software and the SRS is of good quality.
- The requirements process ends with the production of a high quality SRS.

4.3. Requirement Specification

For many years requirements are specified using stories or scenarios. But, today use case approach is followed to specify the requirements. Before discussing the specification of requirements, let's see the following.

4.3.1 Desirable Characteristics of an SRS

To properly satisfy the basic goals, an SRS should have certain properties and should contain different types of requirements. Some of the desirable characteristics of an SRS are

1. **Correct** – The SRS is correct iff every requirement stated therein is one that the software shall meet. There is no tool or procedure that assures correctness.
2. **Complete** - The SRS is complete if everything the software is supposed to do and the responses of the software to all classes of input data are specified in the SRS.
3. **Unambiguous** - the SRS is unambiguous if and only if every requirement stated has one and only one interpretation (or meaning).
4. **Verifiable** – the SRS is verifiable if and only if every stated requirement is verifiable. A requirement is verifiable if there exists some cost-effective process that can check whether the final software meets that requirement.
5. **Consistent** – the SRS is consistent if there is no requirement that conflicts with another. For example, suppose a requirement states that an event e is to occur before another event f. But then another set of requirements states (directly or indirectly by transitivity) that event f should occur before event e. Such a requirement is said to be inconsistent requirement.

6. **Ranked for importance and/or stability** – the SRS is ranked for importance and/or stability if each requirement in it has an identifier which indicates the importance or stability of the requirement. Typically, all requirements are not equally important. In mission control applications, some requirements may be considered as more important and some requirements are considered as less important.

4.3.2 Components of an SRS

The basic issues an SRS must address are:

1. **Functionality** – specifies the functional requirements of the system. Functional requirement represents the service provided by the system. For example, the functional requirements of the library system are
   - Maintain books information
   - Maintain members information
   - Search book
   - Issue book
   - Return book etc.

   Each functional requirement transforms a set of input data to corresponding output data.

   For example, consider the functional requirement *Search Book* in a library system;

   Functional Requirement: Search Book

   Input: author's name

   Output: details of books by that author and the locations of these books in the library
2. **Performance** – represents the non-functional requirements of the system.

   For library system, the non-functional requirements might be
   - Library system should respond to a query in less than 5 seconds
   - It shall operate with zero down time
   - It shall contain well written user manual

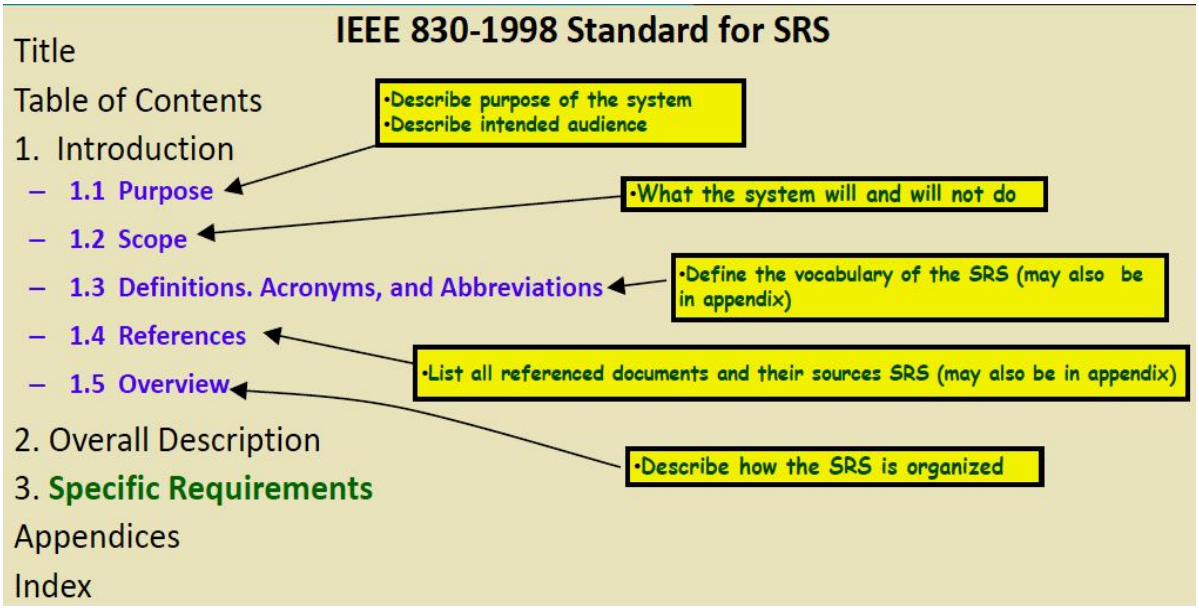
- It shall allow upto 100 users to remotely connect to the system etc.

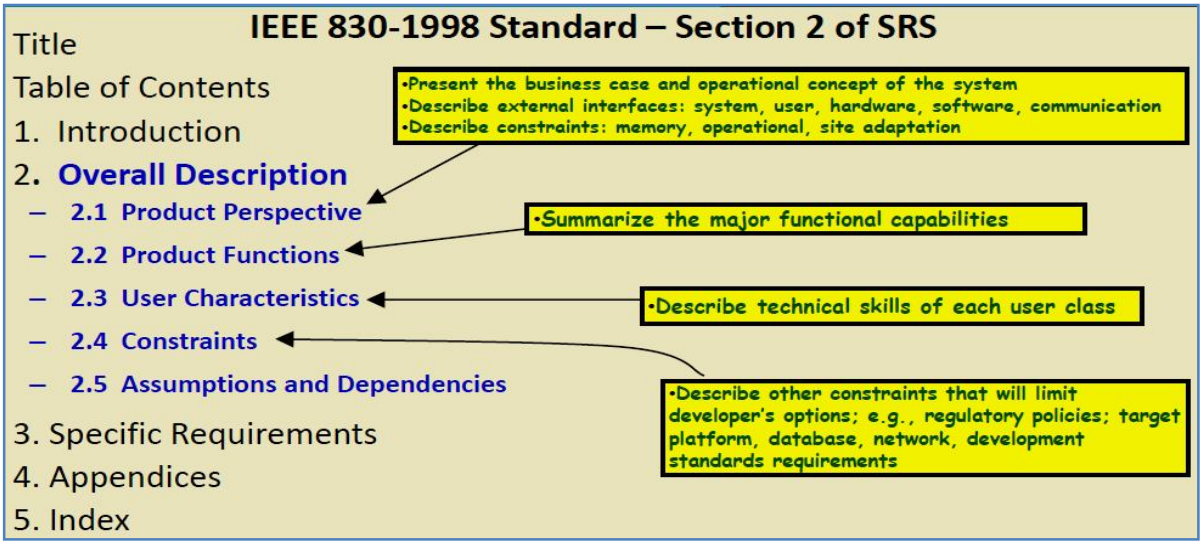
3. **Design constraints** – represents constraints imposed on the system For library system, the design constraints might be
   - It shall provide user interface through standard web browsers
   - It shall use an open source RDBMS such as Postgres SQL
   - It shall be developed using Java language etc.
4. **External interfaces** – It represents how does the software interacts with the people, the system's hardware, other hardware and other software. Examples of external interfaces are; GUI screens, file formats etc.

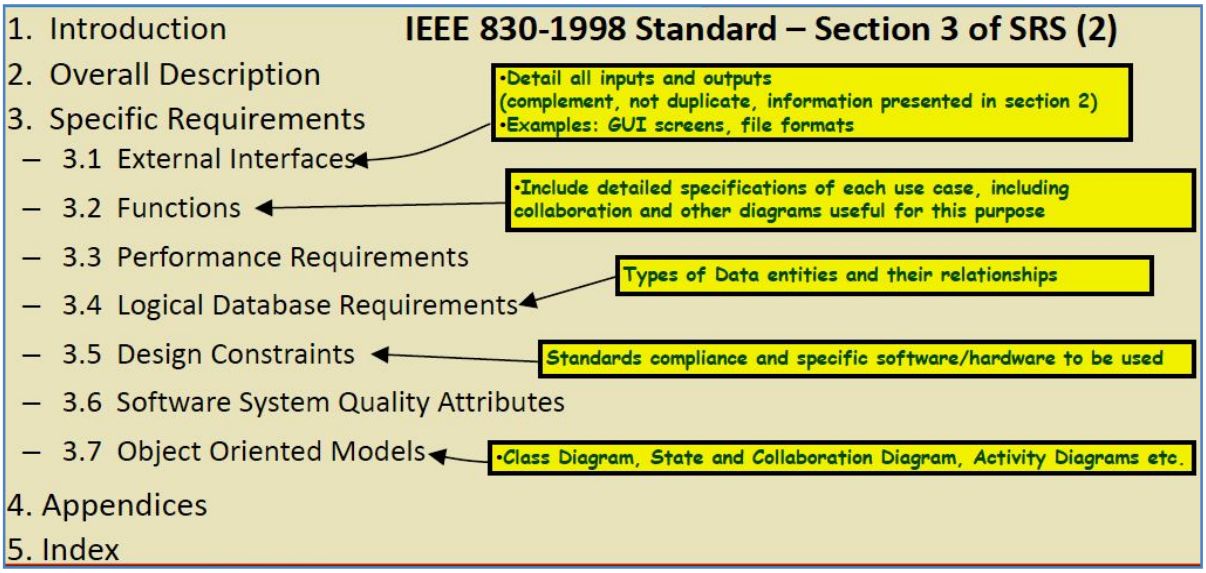
4.3.3 Structure of SRS Document

After collecting all data regarding the system to be developed,

- Remove all inconsistencies and anomalies from the requirements
- Systematically organize the requirements into a Software Requirements Specification (SRS) document.
- SRS document is known as black-box specification because it specifies what the system will do without specifying how it will do.
- Different templates are used by companies to specify SRS: Often variations of IEEE 830 standard is used for specifying SRS document.







The following figure shows an example SRS document.

Example Section 3 of SRS of Academic Administration Software

SPECIFIC REQUIREMENTS

3.1 Functional Requirements

3.1.1 Subject Registration

– The subject registration requirements are concerned with functions regarding subject registration which includes students selecting, adding, dropping, and changing a subject.

F-001:

– The system shall allow a student to register a subject.

F-002:

– It shall allow a student to drop a course.

F-003:

– It shall support checking how many students have already registered for a course.

Design Constraints (3.2)

3.2 Design Constraints

C-001:

– AAS shall provide user interface through standard web browsers.

C-002:

– AAS shall use an open source RDBMS such as Postgres SQL.

C-003:

– AAS shall be developed using the JAVA programming language

Non-functional requirements

3.3 Non-Functional Requirements

N-001:

– AAS shall respond to query in less than 5 seconds.

N-002:

– AAS shall operate with zero down time.

N-003:

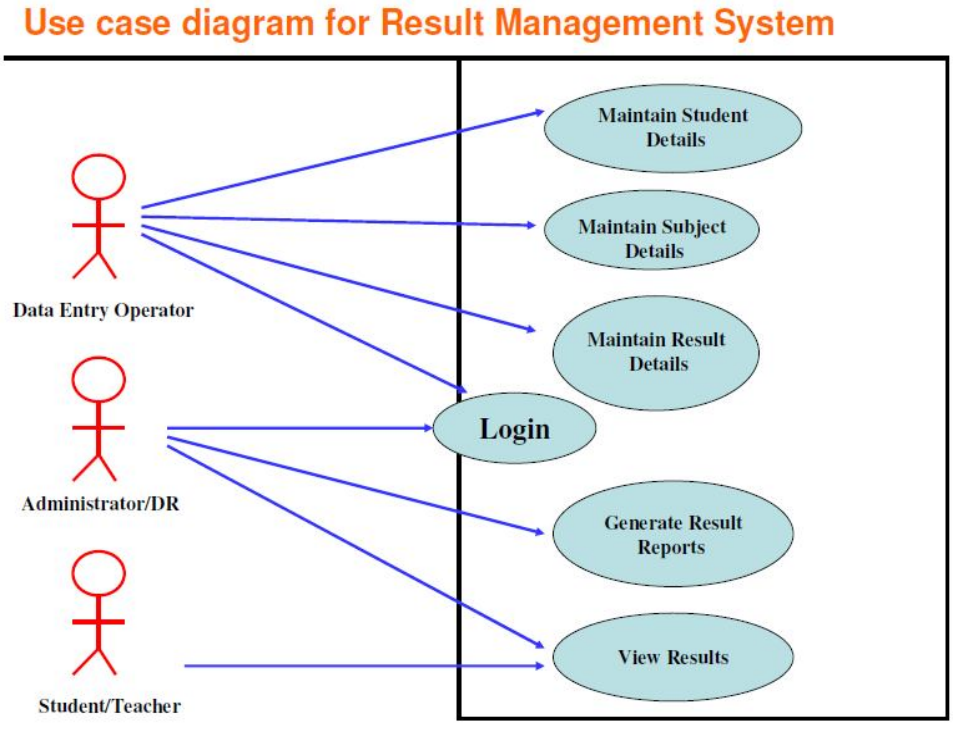
– AAS shall allow upto 100 users to remotely connect to the system.

N-004:

– The system will be accompanied by a well-written user manual.

4.4 Functional Specification with Use Cases

- Ivar Jacobson & others introduced Use Case approach for gathering & modelling requirements.
- Use case represent high level functional requirement of a system.
- For example, the use cases of ATM system are;
  - Withdraw money
  - Deposit money
  - Transfer money are
  - Mini statement
  - Check balance etc.
- An actor initiates a use case for achieving a goal.
- An actor can be a person or a machine or another software system that interacts with this system.
- A uses case diagram is used to show the behaviour of a system. For example, the following figure shows the use case diagram for Result Management System.



- Each use case is described by two types of scenarios.

- Main success scenario (or Basic flow)
- Exceptional scenarios (or Alternate flows)

- In addition to this, actor(s), preconditions and post conditions of the use case are also specified.

Example 4.1

Use Case – **Login**

1.1 **Actors:** 1. Data Entry Operator
2. Administrator/Deputy Registrar

1.2 **Pre conditions** – None

1.3 **Post conditions** - If the use case is successful, the actor is logged into the system. If not, the system state is unchanged.

1.4 **Basic Flow** - This use case starts when the actor wishes to login to the Result Management system.

i. System requests that the actor to enter user id and password.
ii. The actor enters user id & password.
iii. System validates user id & password, and if finds correct allow the actor to logs into the system.

1.5 **Alternate Flow**

1.5.1 If in the basic flow, the actor enters an invalid user id and/or password, the system displays an error message. The actor can choose to either return to the beginning of the basic flow or cancel the login, at that point, the use case ends.

Example 4.2

Use Case – **Withdraw Money (ATM System)**

1.1 **Actor** – Customer

1.2 **Pre conditions** –

1. ATM must be in a state ready to accept transactions
2. ATM must have at least some cash it can dispense
3. ATM must have at least some cash it can dispense

1.3 **Post conditions**

1. The current amount of cash in the user account is the amount before withdraw minus withdraw amount

2. A receipt was printed on the withdraw amount

1.4 **Basic Flow** – This use case starts when customer invokes withdraw money use case.

i. Customer inserts a Credit card into ATM
ii. System verifies the customer ID and status
iii. System asks for an operation type
iv. Customer chooses “Withdraw” operation
v. System asks for the withdraw amount
vi. Customer enters the cash amount
vii. System checks if withdraw amount is legal
viii. System dispenses the cash
ix. System deduces the withdraw amount from account
x. System prints a receipt
xi. Customer takes the cash and the receipt
xii. System ejects the cash card

1.5 **Alternate Flows**

1.5.1 Step 2: Customer authorization failed. Display an error message, cancel the transaction and eject the card.

1.5.2 Step 7: Customer has insufficient funds in its account. Display an error message, and go to step 5.

1.5.3 Step 7: Customer exceeds its legal amount. Display an error message, and go to step 5.

SE UNIT-IV
Assignment-Cum-Tutorial Questions

A) Objective Questions

1) What is the final outcome of requirements analysis and specification phase? [ ]

a) Drawing the data flow diagram b) The SRS document

c) Coding the project d) The user manual

2) Which of the following is not included in SRS document? [ ]

a) Functional requirements b)Non functional requirements

c) Goals of implementation d) User manual

3) As Software Manager, when you will decide the number of people required for a software project? [ ]

a) Before the scope is determined.

b) Before an estimate of the development effort is made

c) After an estimate of the development effort is made.

d) None of the above

4) Which of the following is not a 'concern' during the management of a software project? [ ]

a) Money d) Time

b) Product quality e)Project/product information

c) Product quantity

5) How does a software project manager need to act to minimize the risk of software failure? [ ]

a) Double the project team size

b) Request a large budget

c) Form a small software team

d) Track progress

e) Request for more period of time.

6) Which one of the following is a functional requirement [ ]

a) Maintainability

b) Portability

c) Robustness

d) None of the mentioned

7) The Software Requirement Specification(SRS) is said to be _____ if and only if no subset of individual requirements described in it conflict with each other. [ ]

a) Correct b) Consistent c) Unambiguous d) Verifiable

8) Which one of the following is NOT desired in a good software requirement specifications(SRS) document? [ ]

a) Functional requirements

b) Non-Functional requirements

c) Goals of implementations

d) Algorithm for software implementation.

9) When is the requirement specification activity carried out? [ ]

a) During requirements gathering activity

b) Before requirements analysis activity

c)Before requirements gathering activity

d) After requirements analysis activity

10) Which one of the following is not a requirements gathering technique?

a) Task analysis b) Scenario analysis [ ]

c) Form analysis d) SRS document review

SECTION-B

Descriptive Questions

1) Explain briefly **the value of good SRS and** the Requirements Engineering Process.

2) Give the Structure of Software Requirements Specification Document.

3) Design a SRS Document for Online Banking System?

4) Describe the Functional Specification Technique with use cases.

5) What is SRS? Discuss the characteristics of SRS.

6) Design a SRS Document for ATM System?

7) Design a SRS Document for Library Management System?

8) Briefly describe the functional specification with usecase with an example of auction system.

SECTION-C

C) Previous Gate Questions

1) Which one of the following is NOT desired in a good Software Requirement Specifications (SRS) document? **(GATE 2011)**

a) Functional Requirements

b) Non-Functional Requirements

c) Goals of Implementation

d) Algorithms for software implementation [ ]

2) A Software Requirements Specification (SRS) document should avoid discussing which one of the following? **(GATE 2015)**

a) User interface issues

b) Non-functional requirements

c) Design specification

d) Interfaces with third party software [ ]

3) Software requirement Specification(SRS) is also known as specification of: (**Nielit Scientist-2016**)

a) White box testing b) Grey box testing [ ]

c) Acceptance testing d) Black box testing