**Experiment No. - 8**

**Name-** Bhavesh Kewalramani

**Roll No.-** A-25

**Section-**  A

**Semester-** 6th

**Shift-** 1st

**Aim:**

To study the IEEE SRS standard and prepare SRS for the identified systems conceptualization.

**Theory:**

## What Is a Software Requirements Specification (SRS) Document?

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill all stakeholders (business, users) needs.

An SRS can be simply summarized into four Ds:

* [Define your product's purpose.](https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document#purpose)
* [Describe what you're building.](https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document#describe)
* [Detail the requirements.](https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document#detail)
* [Deliver it for approval.](https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document#approve)

We want to DEFINE the purpose of our product, DESCRIBE what we are building, DETAIL the individual requirements, and DELIVER it for approval. A good SRS document will define everything from how software will interact when embedded in hardware to the expectations when connected to other software. An even better SRS documents also account for real-life users and human interaction.

The best SRS documents define how the software will interact when embedded in hardware — or when connected to other software. Good SRS documents also account for real-life users.

### Why Use an SRS Document?

An SRS gives you a complete picture of your entire project. It provides a single source of truth that every team involved in development will follow. It is your plan of action and keeps all your teams — from development to maintenance — on the same page (no pun intended).

This layout not only keeps your teams in sync but it also ensures that each requirement is hit. It can ultimately help you make vital decisions on your product’s lifecycle, such as when to retire an obsolete feature.

The time it takes to write an SRS is given back in the development phase. It allows for better understanding or your product, team, and the time it will take to complete.

### Software Requirements Specification vs. System Requirements Specification

A **software requirements specification (SRS)** includes in-depth descriptions of the software that will be developed.

A **system requirements specification (SyRS)** collects information on the requirements for a system.

“Software” and “system” are sometimes used interchangeably as SRS. But, a software requirement specification provides greater detail than a system requirements specification.

## How to Write an SRS Document

Writing an SRS document is important. But it isn’t always easy to do.

Here are five steps you can follow to write an effective SRS document.

### 1. Define the Purpose With an Outline (Or Use an SRS Template)

Your first step is to create an outline for your software requirements specification. This may be something you create yourself. Or you may use an existing SRS template.

If you’re creating this yourself, here’s what your outline might look like:

**1. Introduction**

           1.1 Purpose

           1.2 Intended Audience

           1.3 Intended Use

           1.4 Scope

           1.5 Definitions and Acronyms

**2. Overall Description**

           2.1 User Needs

           2.2 Assumptions and Dependencies

**3. System Features and Requirements**

            3.1 Functional Requirements

            3.2 External Interface Requirements

            3.3 System Features

            3.4 Nonfunctional Requirements

This is a basic outline and yours may contain more (or fewer) items. Now that you have an outline, lets fill in the blanks.

### 2. Define your Product’s Purpose

This introduction is very important as it sets expectations that we will hit throughout the SRS.   
Some items to keep in mind when defining this purpose include:

#### **Intended Audience and Intended Use**

Define who in your organization will have access to the SRS and how they should use it. This may include developers, testers, and project managers. It could also include stakeholders in other departments, including leadership teams, sales, and marketing. Defining this now will lead to less work in the future.

#### **Product Scope**

What are the benefits, objectives, and goals we intend to have for this product? This should relate to overall business goals, especially if teams outside of development will have access to the SRS.

#### **Definitions and Acronyms**

It’s important to define the risks in the project. What could go wrong? How do me mitigate these risks? Who is in charge of these risk items?  
For example, if the failure of a medical device would cause slight injury, that is one level of risk. Taking into account the occurrence level and the severity, we can come up with a strategy to mitigate this risk.

### 3. Describe What You Will Build

Your next step is to give a description of what you’re going to build. Is it a new product? Is it an add-on to a product you’ve already created? Is this going to integrate with another product? Why is this needed? Who is it for?

Understanding these questions on the front end makes creating the product much easier for all involved.

#### **User Needs**

Describe who will use the product and how. Understanding the user of the product and their needs is a critical part of the process.

Who will be using the product? Are they a primary or secondary user? Do you need to know about the purchaser of the product as well as the end user? In medical devices, you will also need to know the needs of the patient.

#### **Assumptions and Dependencies**

What are we assuming will be true? Understating and laying out these assumptions ahead of time will help with headaches later. Are we assuming current technology? Are we basing this on a Windows framework?  We need to take stock of these assumptions to better understand when our product would fail or not operate perfectly.

Finally, you should note if your project is dependent on any external factors. Are we reusing a bit of software from a previous project? This new project would then depend on that operating correctly and should be included.

### 4. Detail Your Specific Requirements

In order for your development team to meet the requirements properly, we MUST include as much detail as possible. This can feel overwhelming but becomes easier as you break down your requirements into categories. Some common categories are:

#### **Functional Requirements**

Functional requirements are essential to your product because, as they state, they provide some sort of functionality.

Asking yourself the question “does this add to my tool’s functionality?” Or “What function does this provide?” can help with this process. Within Medical devices especially, these functional requirements may have a subset of risks and requirements.

You may also have requirements that outline how your software will interact with other tools, which brings us to external interface requirements.

#### **External Interface Requirements**

External interface requirements are specific types of functional requirements. These are especially important when working with embedded systems. They outline how your product will interface with other components.

There are several types of interfaces you may have requirements for, including:

* User
* Hardware
* Software
* Communications

#### **System Features**

System features are types of functional requirements. These are features that are required in order for a system to function.

#### **Other Nonfunctional Requirements**

Nonfunctional requirements can be just as important as functional ones.

These include:

* Performance
* Safety
* Security
* Quality

The importance of this type of requirement may vary depending on your industry. In the medical device industry, there are often regulations that require the tracking and accounting of safety.

[*IEEE*](https://ieeexplore.ieee.org/document/278253) also provides guidance for writing software requirements specifications, if you’re a member.

### 5. Deliver for Approval

We made it! After completing the SRS, you’ll need to get it approved by key stakeholders. This will require everyone to review the latest version of the document.

## Writing an SRS in Microsoft Word vs. Requirement Software

You can write your software requirement specification in Microsoft Word. A smart way to do this is to create an SRS template that you can use as a starting point for every project.

However, even with a template, writing an SRS this way can be a painstaking process. And if a requirement changes, your SRS can fall easily out-of-date. Plus, there can be versioning issues with requirements documents in Word.

**Conclusion:**

SRS **helps the customers to define their needs with accuracy**, while it helps the development team understand what the customers need in terms of development. Investing time in writing the SRS document will lead to the successful development of the software the customers need.