


Addis Ababa Institute of Technology
School of Information Technology and Engineering
Department of IT/SE Eng.

# <Project Name>

# Software Requirements Specification

Team Members

Advisors:
.

Date

Revision History

| Date | Description | Author | Comments |
|---|---|---|---|
| <date> | <Version 1> | <Your Name> | <First Revision> |
| | | | |
| | | | |
| | | | |

## Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

| Signature | Printed Name | Title | Date |
|---|---|---|---|
| | <Your Name> | Lead Software Eng. | |
| | A. David McKinnon | Instructor, CptS 322 | |
| | | | |

# Table of Contents

**DOCUMENT APPROVAL** ..... **II**

**LIST OF TABLES** ..... **IV**

**LIST OF FIGURES** ..... **V**

**DEFINITIONS, ACRONYMS, AND ABBREVIATIONS** ..... **VI**

**DECLARATION** ..... **VII**

**1. INTRODUCTION** ..... **1**

1.1 Purpose ..... 1
1.2 Scope ..... 1
1.3 Overview ..... 1

**2. GENERAL DESCRIPTION** ..... **1**

2.1 Product Perspective ..... 1
2.2 Product Functions ..... 1
2.3 User Characteristics ..... 2
2.4 General Constraints ..... 2

**3. SPECIFIC REQUIREMENTS** ..... **2**

3.1 External Interface Requirements ..... 3
*3.1.1 User Interfaces* ..... *3*
*3.1.2 Hardware Interfaces* ..... *4*
*3.1.3 Software Interfaces* ..... *4*
*3.1.4 Communications Interfaces* ..... *5*
*3.2 Functional Requirements* ..... *5*
*3.2.1 <Functional Requirement or Feature #1>* ..... *5*
*3.2.2 <Functional Requirement or Feature #2>* ..... *5*
3.3 Use Cases ..... 5
*3.3.1 Use Case #1* ..... *5*
*3.3.2 Use Case #2* ..... *5*
3.4 Non-Functional Requirements ..... 5
*3.4.1 Performance* ..... *5*
*3.4.2 Reliability* ..... *5*
*3.4.3 Availability* ..... *5*
*3.4.4 Security* ..... *5*
*3.4.5 Maintainability* ..... *5*
*3.4.6 Portability* ..... *5*
3.5 Inverse Requirements ..... 5
3.6 Design Constraints ..... 6
3.7 Logical Database Requirements ..... 6
3.8 Other Requirements ..... 6

**4. CHANGE MANAGEMENT PROCESS** ..... **6**

**REFERENCES** ..... **7**

**A. APPENDICES** ..... **8**

A.1 Appendix 1 ..... 8

A.2 Appendix 2.....8

## List of Tables

## List of figures

## Definitions, Acronyms, and Abbreviations

*This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.*

# DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included. We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

**Group NO Team Date:** xx-xx-2021

# 1. Introduction

*The introduction to the Software Requirement Specification (SRS) document should provide an overview of the complete SRS document. While writing this document please remember that this document should contain all of the information needed by a software engineer to adequately design and implement the software product described by the requirements listed in this document. (Note: the following subsection annotates are largely taken from the IEEE Guide to SRS).*

## 1.1 Purpose

*What is the purpose of this SRS and the (intended) audience for which it is written?*

## 1.2 Scope

*This subsection should:*
*(1) Identify the software product(s) to be produced by name; for example, Host DBMS, Report Generator, etc*
*(2) Explain what the software product(s) will, and, if necessary, will not do*
*(3) Describe the application of the software being specified. As a portion of this, it should:*
*(a) Describe all relevant benefits, objectives, and goals as precisely as possible. For example, to say that one goal is to provide effective reporting capabilities is not as good as saying parameter-driven, user-definable reports with a 2 h turnaround and on-line entry of user parameters.*
*(b) Be consistent with similar statements in higher-level specifications (for example, the System Requirement Specification) , if they exist.What is the scope of this software product.*

## 1.3 Overview

*This subsection should:*
*(1) Describe what the rest of the SRS contains and Explain how the SRS is organized.*

# 2. General Description

*This section of the SRS should describe the general factors that affect 'the product and its requirements. It should be made clear that this section does not state specific requirements; it only makes those requirements easier to understand.*

## 2.1 Product Perspective

*This subsection of the SRS puts the product into perspective with other related products or projects. (See the IEEE Guide to SRS for more details).*

## 2.2 Product Functions

*This subsection of the SRS should provide a summary of the functions that the software will perform.*

## 2.3 User Characteristics

*This subsection of the SRS should describe those general characteristics of the eventual users of the product that will affect the specific requirements. (See the IEEE Guide to SRS for more details).*

## 2.4 General Constraints

*This subsection of the SRS should provide a general description of any other items that will limit the developer's options for designing the system. (See the IEEE Guide to SRS for a partial list of possible general constraints).*
*Provide a general description of any other items that will limit the developer's options. These can include:*

*(1) Regulatory policies*
*(2) Hardware limitations (for example, signal timing requirements)*
*(3) Interface to other applications*
*(4) Parallel operation*
*(5) Audit functions*
*(6) Control functions*
*(7) Higher-order language requirements*
(8) *Signal handshake protocols (for example, XON-XOFF, ACK-NACK)*
(9) *Reliability requirements*
*(10) Criticality of the application*
*(11) Safety and security considerations*

*This section captures non-functional requirements in the customer's language. A more formal presentation of these will occur in section 3.*

## 2.5 Assumptions and Dependencies

*This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.*

# 3. Specific Requirements

*This section contains all the software requirements at a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. Throughout this section, every stated requirement should be externally perceivable by users, operators, or other external systems. These requirements should*

*include at a minimum a description of every input (stimulus) into the system, every output (response) from the system and all functions performed by the system in response to an input or in support of an output. The following principles apply:*

*(1) Specific requirements should be stated with all the characteristics of a good SRS*
- *correct*
- *unambiguous*
- *complete*
- *consistent*
- *ranked for importance and/or stability*
- *verifiable*
- *modifiable*
- *traceable*

*(2) Specific requirements should be cross-referenced to earlier documents that relate*
*(3) All requirements should be uniquely identifiable (usually via numbering like 3.1.2.3)*
*(4) Careful attention should be given to organizing the requirements to maximize readability (Several alternative organizations are given at end of document)*

*Before examining specific ways of organizing the requirements it is helpful to understand the various items that comprise requirements as described in the following subclasses. This section reiterates section 2, but is for developers not the customer. The customer buys in with section 2, the designers use section 3 to design and build the actual application.*

*Remember this is not design. Do not require specific software packages, etc unless the customer specifically requires them. Avoid over-constraining your design. Use proper terminology:*
*The system shall… A required, must have feature*
*The system should… A desired feature, but may be deferred til later*
*The system may… An optional, nice-to-have feature that may never make it to implementation.*

*Each requirement should be uniquely identified for traceability. Usually, they are numbered 3.1, 3.1.1, 3.1.2.1 etc. Each requirement should also be testable. Avoid imprecise statements like, "The system shall be easy to use" Well no kidding, what does that mean? Avoid "motherhood and apple pie" type statements, "The system shall be developed using good software engineering practice"*

*Avoid examples, This is a specification, a designer should be able to read this spec and build the system without bothering the customer again. Don't say things like, "The system shall accept configuration information such as name and address." The designer doesn't know if that is the only two data elements or if there are 200. List every piece of information that is required so the designers can build the right UI and data tables.*

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

*Specify:*

*(1) The logical characteristics of each interface between the software product and its users.*

*(2) All the aspects of optimizing the interface with the person who must use the system*

*This is a description of how the system will interact with its users. Is there a GUI, a command line or some other type of interface? Are there special interface requirements? If you are designing for the general student population for instance, what is the impact of ADA (American with Disabilities Act) on your interface?*

### 3.1.2 Hardware Interfaces

*Specify the logical characteristics of each interface between the software product and the hardware components of the system. This includes configuration characteristics. It also covers such matters as what devices are to be supported, how they are to be supported and protocols. This is not a description of hardware requirements in the sense that "This program must run on a Mac with 64M of RAM". This section is for detailing the actual hardware devices your application will interact with and control. For instance, if you are controlling X10 type home devices, what is the interface to those devices? Designers should be able to look at this and know what hardware they need to worry about in the design. Many business type applications will have no hardware interfaces. If none, just state "The system has no hardware interface requirements" If you just delete sections that are not applicable, then readers do not know if: a. this does not apply or b. you forgot to include the section in the first place.*

### 3.1.3 Software Interfaces

*Specify the use of other required software products and interfaces with other application systems. For each required software product, include:*

*(1) Name*

*(2) Mnemonic*

*(3) Specification number*

*(4) Version number*

*(5) Source*

*For each interface, provide:*

*(1) Discussion of the purpose of the interfacing software as related to this software product*

*(2) Definition of the interface in terms of message content and format*

*Here we document the APIs, versions of software that we do not have to write, but that our system has to use. For instance if your customer uses SQL Server 7 and you are required to use that, then you need to specify i.e.*
*2.1.4.1 Microsoft SQL Server 7. The system must use SQL Server as its database component. Communication with the DB is through ODBC connections. The system must provide SQL data table definintions to be provided to the company DBA for setup.*

*A key point to remember is that you do NOT want to specify software here that you think would be good to use. This is only for **customer-specified systems** that you **have** to interact with. Choosing SQL Server 7 as a DB without a customer requirement is a Design choice, not a requirement. This is a subtle but important point to writing good requirements and not over-constraining the design.*

### 3.1.4 Communications Interfaces

*Specify the various interfaces to communications such as local network protocols, etc. These are protocols you will need to directly interact with. If you happen to use web services transparently to your application then do not list it here. If you are using a custom protocol to communicate between systems, then document that protocol here so designers know what to design. If it is a standard protocol, you can reference an existing document or RFC.*

### 3.2 Functional Requirements

*This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.*

#### 3.2.1 <Functional Requirement or Feature #1>

3.2.1.1 Introduction
3.2.1.2 Inputs
3.2.1.3 Processing
3.2.1.4 Outputs
3.2.1.5 Error Handling

#### 3.2.2 <Functional Requirement or Feature #2>

…

## 3.3 Use Cases

#### 3.3.1 Use Case #1

#### 3.3.2 Use Case #2

…

## 3.4 Non-Functional Requirements

*Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the*

*following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).*

**3.4.1 Performance**

**3.4.2 Reliability**

**3.4.3 Availability**

**3.4.4 Security**

**3.4.5 Maintainability**

**3.4.6 Portability**

## 3.5 Inverse Requirements

*State any *useful* inverse requirements.*
***what the application does not do***

## 3.6 Design Constraints

*Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.*

## 3.7 Logical Database Requirements

*Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc.*

## 3.8 Other Requirements

*Catchall section for any additional requirements:*

***Training-related Requirements***

If there is any specific training that should be necessary for a user to begin using this application.

***Packaging Requirements***

How the system is packed (folder with code or in exe format or……) if there is any file for the purpose of help (README file) and Any If the is makefile for recompilation

***Legal Requirements***

Copyright laws and license agreements

# 4. Change Management Process

*Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.*

## References

*This subsection should:*
*(1) Provide a complete list of all documents referenced elsewhere in the SRS, or in a separate, specified document.*
*(2) Identify each document by title, report number - if applicable - date, and publishing organization.*
*(3) Specify the sources from which the references can be obtained.*
*This information may be provided by reference to an appendix or to another document.*

# A. Appendices

*Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS's overall set of requirements.*

*Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.*

## A.1 Appendix 1

## A.2 Appendix 2