Software Requirements Specification Template

CS 258 Software Engineering

January 2018

The following annotated template shall be used to complete the Software Requirements Specification (SRS) assignment of CS 258.

**Template Usage:**

Text contained within angle brackets (‘<’, ‘>’) shall be replaced by your project-specific information and/or details. For example, <Project Name> will be replaced with either ‘Smart Home’ or ‘Sensor Network’.

Italicized text is included to briefly annotate the purpose of each section within this template. This text should not appear in the final version of your submitted SRS.

This cover page is not a part of the final template and should be removed before your SRS is submitted.

**Acknowledgements:**

Sections of this document are based on the IEEE Guide to Software Requirements Specification (ANSI/IEEE Std. 830-1984). The SRS templates of Dr. OrestPilskalns (WSU, Vancover) and Jack Hagemeister (WSU, Pullman) have also been used as guides in developing this.

<Software Application For Interfacing and Controlling A High Resistance Meter and Temperature Controller>

Software Requirements Specification

Version 1.0

22-1-2018

SHASHANK GIRI

AYUSH MANTRI

NEHA NAGENDRA

GHANSHYAM BAIRWA

NANDESHWAR PANKAJ SIDDHARTH

Prepared for

CS 258 Software Engineering

Spring 2018

# 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> |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

Revision History ii

Document Approval ii

1. Introduction i

1.1 Purpose i

1.2 Scope i

1.3 Definitions, Acronyms, and Abbreviations i

1.4 References i

1.5 Overview i

2. General Description ii

2.1 Product Perspective ii

2.2 Product Functions ii

2.3 User Characteristics ii

2.4 General Constraints ii

2.5 Assumptions and Dependencies ii

3. Specific Requirements ii

3.1 External Interface Requirements iii

3.1.1 User Interfaces iii

3.1.2 Hardware Interfaces iii

3.1.3 Software Interfaces iii

3.1.4 Communications Interfaces iii

3.2 Functional Requirements iii

3.2.1 <Functional Requirement or Feature #1> iii

3.2.2 <Functional Requirement or Feature #2> iii

3.3 Classes / Objects iii

3.3.1 <Class / Object #1> iii

3.3.2 <Class / Object #2> iii

3.4 Non-Functional Requirements iii

3.4.1 Performance iv

3.4.2 Reliability iv

3.4.3 Availability iv

3.4.4 Security iv

3.4.5 Maintainability iv

3.4.6 Portability iv

3.5 Inverse Requirements iv

3.6 Design Constraints iv

3.7 Logical Database Requirements iv

3.8 Other Requirements iv

A. Appendices iv

A.1 Appendix 1 iv

A.2 Appendix 2 iv

# 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 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.*

## 1.4 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.*

## 1.5 Overview

*This subsection should:*

*(1) Describe what the rest of the SRS contains*

*(2) 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).*

## 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 will give the D-requirements that are used to guide the project’s software design, implementation, and testing.

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

All interaction with the user should be via the GUI designed on LabVIEW. Laboratory Virtual Instrument Engineering Workbench is a development environment for visual programming language from National Instruments. The ultimate goal is to display the graph between resistivity and temperature of a high resistance substance which is to be used as the working material. The GUI should be such that the user can control the experiment variables (like current, distance between the two probes, etc) directly by typing the desired values of these variables on the corresponding places. The application is dynamic in the sense that it displays the graph while the values are still being recorded. At the end of the experiment which usually lasts for hours the data recorded of Temperature vs Resistivity should be stored in a different spreadsheet for future references.

### 3.1.2 Hardware Interfaces-

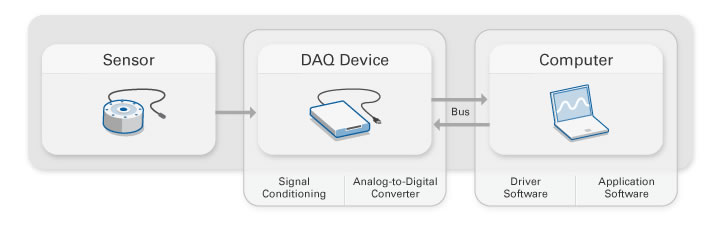
The two major hardware to be used and controlled from the application is Temperature controller and high resistance meter. The setup should be such that the values of the experiment variables can be controlled via the designed GUI or via the buttons on these two national instruments devices and it should be reflected in the other.

### 3.1.3 Software Interfaces

LabVIEW requires another software applicaltion called NI-MAX for controlling the drivers for easy connection between the device and the system. MAX installs automatically with NI-VISA version 2.5 or higher or NI-VXI version 3.0 or higher. MAX is available only for Win32-based operating systems.

### 3.1.4 Communications Interfaces

The major communication interface to be made is between the designed computer application and the hardware devices used in the experiment. The most common means of connecting the Lab VIEW application to devices is via a DAQ device, for example a USB cable, or WLAN cable. Some special kinds of DAQ which needs to be used for easy connectivity here is KUSB-488a cable and IEEE cable. LabVIEW makes it easy for making communication interfaces as it comes with specific drivers for specific DAQ devices. These drivers make I/O easy as only the address of the device connected is required.



## 3.2 Functional Requirements

This section describes specific features of the software project

### 3.2.1 Functional Requirement of designed GUI

3.2.1.1 Introduction

The main fundamental feature is that the LabVIEW GUI should be able to take input data from the temperature controller and high resistance meter accurately and also to write values to these devices accurately. To ensure this the particular driver of the instrument should be appropriately installed and correct address of the connected device should be used. Also the graph drawn between the temperature and resistivity should comply with input data.

3.2.1.2 Inputs

The major variables to control are the current (to be passed between the two probes) and temperature of the working piece.

1. Since the working piece has high resistance, the temperature should be strictly controlled and should not exceed the safe bounds (A safe bound for temperature used here is <300 K). Thus the application should be such that it should stop the experiment when this temperature is attained and after that should decrease the temperature and retrace the data recording process.
2. The current should also be appropriately controlled by the user and if the user is trying to run the experiment with higher current values, it should display an error message.
3. The voltage across the two probes of the working piece will be measured by voltmeter and this value should be read by the LabVIEW for further calculations and display.

3.2.1.3 Processing

The one value to be calculated here is resistivity. As all other values are pre-known or are provided by the user operating the application.

We have current (I), Voltage across the probes (V), and length (l) and area (A) of working piece. Resistivity=V\*A/I\*l;

All we need is now make a graph between this resistivity and temperature at regular interval of 5K.

3.2.1.4 Outputs

Once the experiment is completed the obtained values of temperature VS resistivity should be stored in a local spreadsheet for future reference. This can be achieved by file I/O capability of LabVIEW workbench.

3.2.1.5 Error Handling

The application should not behave abruptly after usage for sometime and the resistivity value should be an increasing function of temperature or else there is some problem either with connection or the material is not what it is expected to be.

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

…

## 3.3 Classes / Objects

### 3.3.1 <Class / Object #1>

3.3.1.1 Attributes

3.3.1.2 Functions

<Reference to functional requirements and/or use cases>

### 3.3.2 <Class / Object #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.

## 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.

# 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