

Requirements Templates

SE3A04 – Tutorial

Jason Jaskolka

Department of Computing and Software
Faculty of Engineering
McMaster University
Hamilton, Ontario, Canada
jaskolj@mcmaster.ca

September 25, 2012

Outline

- ① Requirements Documents
- ② Requirements Document Templates
- ③ IEEE/Volere Hybrid Template
 - Section 1: Introduction
 - Section 2: Overall Description
 - Section 3: Functional Requirements
 - Section 4: Non-Functional Requirements
- ④ Some Tips for Writing Requirements Documents
- ⑤ References
- ⑥ Questions

What is a Requirements Document?

Definition (Software Requirements Specification (SRS))

A **software requirements specification (SRS)** or **requirements document** is a complete description of the behaviour of a system to be developed. It may include a set of use cases that describe interaction between the system and the user/environment.

Requirements Document Templates

- There are several different formats or templates suggested for requirements documents by different groups
 - **Example:** IEEE, Volere, etc.
- We are recommending a hybrid template
 - IEEE for context and scope
 - IEEE for functional requirements
 - Volere for non-functional requirements

Section 1: Introduction

- This section of the SRS should provide an overview of the entire SRS
- It should contain the following subsections:
 - 1 Purpose
 - 2 Scope
 - 3 Definitions, Acronyms, and Abbreviations
 - 4 References
 - 5 Overview

Section 1: Introduction

1.1 Purpose

- a) Delineate the purpose of the SRS
- b) Specify the intended audience for the SRS

1.2 Scope

- a) Identify the software product(s) to be produced by name (e.g., Host DBMS, Report Generator, etc.)
- b) Explain what the software product(s) will, and, if necessary, will not do
- c) Describe the application of the software being specified, including relevant benefits, objectives, and goals
- d) Be consistent with similar statements in higher-level specifications (e.g., the system requirements specification), if they exist

Section 1: Introduction

1.3 Definitions, Acronyms, and Abbreviations

- a) Provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS

1.4 References

- a) Provide a complete list of all documents referenced elsewhere in the SRS
- b) Identify each document by title, report number (if applicable), date, and publishing organization
- c) Specify the sources from which the references can be obtained

1.5 Overview

- a) Describe what the rest of the SRS contains
- b) Explain how the SRS is organized

Section 2: Overall Description

- This section of the SRS should describe the general factors that affect the product and its requirements
- It does not state specific requirements – it provides a background for those requirements and makes them easier to understand
- It usually contains the following subsections:
 - 1 Product Perspective
 - 2 Product Functions
 - 3 User Characteristics
 - 4 Constraints
 - 5 Assumptions and Dependencies
 - 6 Apportioning of Requirements

Section 2: Overall Description

2.1 Product Perspective

- a) Put the product into perspective with other related products, i.e., context
- b) If the product is independent and totally self-contained, it should be stated here
- c) If the SRS defines a product that is a component of a larger system, as frequently occurs, then this subsection should relate the requirements of that larger system to functionality of the software and should identify interfaces between that system and the software
- d) A block diagram showing the major components of the larger system, interconnections, and external interfaces can be helpful

Section 2: Overall Description

2.2 Product Functions

- a) Provide a summary of the major functions that the software will perform.
 - **Example:** An SRS for an accounting program may use this part to address customer account maintenance, customer statement, and invoice preparation without mentioning the vast amount of detail that each of those functions requires.
- b) Functions should be organized in a way that makes the list of functions understandable to the customer or to anyone else reading the document for the first time
- c) Textual or graphical methods can be used to show the different functions and their relationships
 - Such a diagram is not intended to show a design of a product, but simply shows the logical relationships among variables

Section 2: Overall Description

2.3 User Characteristics

- a) Describe those general characteristics of the intended users of the product including educational level, experience, and technical expertise
- b) Do not state specific requirements, but rather provide the reasons why certain specific requirements are later specified

2.4 Constraints

- a) Provide a general description of any other items that will limit the developer's options

Section 2: Overall Description

2.5 Assumptions and Dependencies

- a) List each of the factors that affect the requirements stated in the SRS
- b) These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS
 - **Example:** An assumption may 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.

2.6 Apportioning of Requirements

- a) Identify requirements that may be delayed until future versions of the system

Section 3: Functional Requirements

- This section of the SRS should contain all of the software requirements to 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

Section 3: Functional Requirements

- You normally have two options for organizing your functional requirements:
 - 1 Organize first by **business events**, then by **viewpoints**
 - 2 Organize first by **viewpoints**, then by **business events**
- Choose the one which makes the most sense

Section 3: Functional Requirements

Example (Organization By Business Events)

Consider a grocery store.

BE1. Customer places item on conveyor

VP1.1 Customer

VP1.2 Cashier

VP1.3 ...

BE2. Cashier Scans Item

VP2.1 Customer

VP2.2 Cashier

VP2.3 ...

Section 3: Functional Requirements

Example (Organization By Viewpoints)

Consider a grocery store.

VP1. Customer

BE1.1 Place item on conveyor

BE1.2 Redeem coupon

BE1.3 ...

VP2. Cashier

BE2.1 Scan item

BE2.2 Enter coupon code

BE2.3 ...

Section 4: Non-Functional Requirements

- ① Look and Feel Requirements
 - ① Appearance Requirements
 - ② Style Requirements

Example

LF1. The product shall incorporate the company logo and colours.

Section 4: Non-Functional Requirements

- ② Usability and Humanity Requirements
 - ① Ease of Use Requirements
 - ② Personalization and Internationalization Requirements
 - ③ Learning Requirements
 - ④ Understandability and Politeness Requirements
 - ⑤ Accessibility Requirements

Example

UR1. The product shall be easy to use on the first attempt by a member of the public with training.

Section 4: Non-Functional Requirements

③ Performance Requirements

- ① Speed and Latency Requirements
- ② Safety-Critical Requirements
- ③ Precision or Accuracy Requirements
- ④ Reliability and Availability Requirements
- ⑤ Robustness or Fault-Tolerance Requirements
- ⑥ Capacity Requirements
- ⑦ Scalability or Extensibility Requirements
- ⑧ Longevity Requirements

Example

PR1. The product shall identify whether an aircraft is hostile or friendly within 0.25 seconds.

Section 4: Non-Functional Requirements

- ④ Operational and Environmental Requirements
 - ① Expected Physical Environment
 - ② Requirements for Interfacing with Adjacent Systems
 - ③ Productization Requirements
 - ④ Release Requirements

Example

OE1. The product shall be used in and around trucks at night and during rainstorms, snow, and freezing conditions.

Section 4: Non-Functional Requirements

- ⑤ Maintainability and Support Requirements
 - ① Maintenance Requirements
 - ② Supportability Requirements
 - ③ Adaptability Requirements

Example

MS1. The product shall be readily portable to Linux.

Section 4: Non-Functional Requirements

- ⑥ Security Requirements
 - ① Access Requirements
 - ② Integrity Requirements
 - ③ Privacy Requirements
 - ④ Audit Requirements
 - ⑤ Immunity Requirements

Example

SR1. The product shall ensure that its air temperature data corresponds to the data transmitted by the weather station.

Section 4: Non-Functional Requirements

7 Cultural and Political Requirements

- 1 Cultural Requirements
- 2 Political Requirements

Example

CP1. The product shall use British spelling.

Section 4: Non-Functional Requirements

- ⑧ Legal Requirements
 - ① Compliance Requirements
 - ② Standards Requirements

Example

LR1. The product shall comply with the Accessibility for Ontarians with Disabilities Act.

Some Tips for Writing Requirements Documents

- Be sure to include all sections of the template in your document regardless whether you have something to write for each or not
 - If you do not have anything to write in a section, indicate this by the **N/A**, **void**, **none**, etc.
- Uniquely number each of your requirements for easy identification and cross-referencing
- Highlight terms that are defined in Section 1.3 (**Definitions, Acronyms, and Abbreviations**) with **bold**, *italic* or underline

Some Tips for Writing Requirements Documents

Special Note!

For Deliverable 1, please highlight, in some fashion, all (you may have more than one) creative and innovative features.

Your creative and innovative features will generally be described in Section 2.2 (**Product Functions**), but it will depend on the type of creative or innovative features you are including.

References



IEEE-SA Standards Board

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

Institute of Electrical and Electronics Engineers, October 1998.



Suzanne Robertson and James Robertson.

Mastering the Requirements Process, 2nd Edition

Addison-Wesley, 2006.

Questions

- Questions?