ECCS 2411: Software Design

System Requirements Specification Template

(Adapted from Susan Mitchell and Michael Grasso)

General Instructions

1. Provide a cover page that includes the document name, product name, customer name, team name, team member names, and the current date.
2. Number the pages of the document.
3. Number and label all figures. Refer to the figures by number in the text.
4. All sections should have an introductory sentence or two.
5. Do not use vague words and phrases such as may, might, could, possibly, should, assumed to be, some, a little, and a lot. Use strong, definite words and phrases such as shall, will, will not, can, and cannot.
6. Watch your spelling, punctuation, and grammar. It is a reflection on your professionalism.

Be sure that your document is

- Complete - No information is missing
- Clear - Every sentence's meaning must be clear to all parties
- Consistent – The writing style and notation is consistent throughout the document and the document does not contradict itself
- Verifiable - All facts stated are verifiable

Remember that you are required to do a peer review of this document.

When you think you are done with the SRS, ask yourself, "Could someone who was not part of the development of this SRS write the corresponding System Design Document?"

[Put product name here]
System Requirements Specification

Table of Contents

Page

1. Introduction

   1.1 Purpose of This Document
   1.2 References
   1.3 Purpose of the Product
   1.4 Product Scope

2. Functional Requirements

3. Non-Functional Requirements

4. User Interface

5. Deliverables

6. Open Issues

Appendix A – Agreement Between Customer and Contractor

1. **Introduction**

1.1 Purpose of This Document

State the purpose of this document and specify the intended readership.

1.2 References

Provide a list of all applicable and referenced documents and other media (e.g., the Somerville text, UML references, documents provided by the customer, websites). For each reference, provide the title, author, publisher (if applicable), date, and URL (for websites).

1.3 Purpose of the Product

This section provides a short description of the user's work and the situation that triggered the need for the product. It describes the task(s) that the user wants to accomplish with the delivered product. It is the product justification.

1.4 Product Scope

This section identifies the boundary between the system under development and the outside world. That is, it identifies what is included in the system and what is not. Typically, a context diagram best describes the boundary. However, because the systems in this class are small, we will use a combination top-level use case and context diagram. In addition to referring the reader to the diagram, give a brief summary of how it illustrates the system's scope. Make sure to number the use cases in the diagram.

Use *The Unified Modeling Language(UML): A* reference is *UML Distilled*, by Martin Fowler.

2. **Functional Requirements**

Each functional requirement should be represented using a use case.

Refer the reader to the top-level use case/context diagram referred to in Section 1.4. In addition, include separate use case diagrams, where appropriate, for each of the top-level use cases.

In addition to the diagrams, every use case should be documented using the following use case specification format.

Number	< use case number >
Name	< use case name - a short active verb phrase >
Summary	< a brief summary of the use case >
Priority	< how critical this use case is to the customer (1 to 5, 5 being most critical >
Preconditions	< conditions that must be true before the use case trigger >
Postconditions	< conditions that will be true after the use case completes >
Primary Actor	< a role name for the primary actor >
Secondary Actors	< other systems that are relied upon to accomplish the use case >
Trigger	< the action that starts the use case >

Main Scenario	**Step**	**Action**
	1	< steps of the use case from trigger to goal delivery >
	2	< … >
	3	< … >
Extensions	**Step**	**Branching Action**
	1a	< condition causing branching > **:** < action or name of sub use case >
Open Issues	< list of issues awaiting decisions that affect the use case >	

(This template was adapted from Alistair Cockburn.)

Lastly, write the tests that will be used during system and acceptance testing to verify that each requirement has been met. Note that a single requirement may require multiple tests, so be thorough. It is also possible that a single test verifies more than one requirement. The goal is to come up with the minimum number of test cases that thoroughly test the system. Make sure that the test numbers correspond to the use case numbers.

3. **Non-Functional Requirements**

Decide on a standard format for the non-functional requirements (NFRs). Included in the format should be a unique number for each NFR, a priority (1 = lowest, 5 = highest), a clear, concise description, and the test(s) that will be used during system and acceptance testing to verify that the requirement has been met. Make sure that the test numbers correspond to the NFR numbers. Note that you must include a minimum of 10 NFRs specific to product requirements, organizational requirements, and external requirements.

4. **User Interface**

Simply put a statement such as

> See "User Interface Design Document for *your product name*."

here.

5. **Deliverables**

Provide a list of all deliverable items (that is, all artifacts that you will deliver to the customer). This list will include items such as the product itself (What format? Source code? Executable code? Object code?), documentation, and training resources (if any). Specify when (date) and in what format (e.g., hard copy, CD) each will be delivered. A tabular format works well for this section. We will assume that the deliverable items are as follows:

Hard copies of each of the following:

- Systems Requirement Specification
- System Design Document
- User Interface Design Document
- User Manual
- Administrator Manual
- Copies of all Biweekly Status Reports

A CD (or electronic copy in a ZIP file) containing the following:

- Systems Requirement Specification
- System Design Document
- User Interface Design Document
- User Manual
- Administrator Manual
- All source code
- The executable program
- Any other software required for installation and execution of the delivered program.

6. **Open Issues**

Issues that have been raised and do not yet have a conclusion. These issues will be addressed later in the development process.

Appendix A – Agreement Between Customer and Contractor

Place on a separate page. Describe what the customer and your team are agreeing to when all sign off on this document. [One paragraph] Include a statement that explains the procedure to be used in case there are future changes to the document. [One paragraph] Provide lines for typed names, signatures, and dates for each team member and the customer. Provide space for customer comments.

Appendix B – Team Review Sign-off

Place on a separate page. Provide a brief paragraph stating that all members of the team have reviewed the document and agree on its content and format. Provide lines for typed names, signatures, dates, and comments for each team member. The comment areas are to be used to state any minor points regarding the document that members may not agree with. Note that there cannot be any major points of contention.

Appendix C – Document Contributions

Identify how each member contributed to the creation of this document. Include what sections each member worked on and an estimate of the percentage of work they contributed. Remember that each team member <u>must</u> contribute to the writing (includes diagrams) for each document produced.