**Software Requirements Specification Template**

The following annotated template shall be used to complete the Software Requirements Specification (SRS). The instructor must approve any modifications to the overall structure of this document.

**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 upon the IEEE Guide to Software Requirements Specification (ANSI/IEEE Std. 830-1984).

User Stories:

|  |  |
| --- | --- |
| Identifier: | User Story |
| ST-1 | As the player, I can customize my character |
| ST-2 | As the player, I can control where my character goes |
| ST-3 | As the player, I can pick up items |
| ST-4 | As the player, I can choose to fight or flee a battle |
| ST-5 | As the player, I can save the game |

<PROJECT BUBBAS>

Software Requirements Specification

<0.1>

<3/27/­2020>

<Dominic Ferraro>

<Team Leader, Programmer>

<Cameron Germano>

<Graphic Designer, Programmer>

<Dennis Lupin>

<Programmer>

**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** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

1. Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

1.4 Overview 1

2. General Description 1

2.1 Product Perspective 1

2.2 Product Functions 2

2.3 Users and Characteristics 2

2.4 General Constraints 2

2.5 Assumptions and Dependencies 2

2.6 Operating Environment 2

3. Specific Requirements 2

3.1 External Interface Requirements 3

3.1.1 User Interfaces 3

3.1.2 Hardware Interfaces 3

3.1.3 Software Interfaces 3

3.1.4 Communications Interfaces 3

3.2 Functional Requirements 3

3.2.1 <Functional Requirement or Feature #1> 3

3.2.2 <Functional Requirement or Feature #2> 3

3.3 Use Cases 4

3.3.1 Use Case #1 4

3.3.2 Use Case #2 4

3.4 Non-Functional Requirements 4

3.5.1 Performance 5

3.5.2 Reliability 5

3.5.3 Availability 5

3.5.4 Security 5

3.5.5 Maintainability 5

3.5.6 Portability 5

3.5 Design Constraints 5

3.6 Logical Database Requirements 5

3.7 Other Requirements 5

4. Analysis Models 5

4.1 Sequence Diagrams 6

5. Change Management Process 6

A. Appendices 6

A.1 Appendix 1 7

A.2 Appendix 2 7

**List of Figures**

[Figure 1: Data Flow Diagram Example 1 6](#_Toc504395184)

[Figure 2 Data Flow Diagram Example 2 6](#_Toc504395185)

**List of Tables**

[Table 1: Use case 1 4](#_Toc504395186)

# 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.>

## 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 hour 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 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. This section does not state specific requirements; it only makes those requirements easier to understand.>

## 2.1 Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

## 2.2 Product Functions

<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate is often effective.>

## 2.3 Users and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

## 2.4 General Constraints

<This subsection of the SRS should provide a general description of any other items that will limit the designer/developer’s options for designing/developing the system.>

## 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.>

## 2.6 Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

# 3. Specific Requirements

<This will be the largest and most important section of the SRS. The customer requirements are embodied within Section 2 (functions), but this section will give the D-requirements that are used to guide the project’s software design, implementation, and testing.

Each requirement in this section should be:

* Correct
* Traceable (both forward and backward to prior/future artifacts)
* Unambiguous
* Verifiable (i.e., testable)
* Prioritized (with respect to importance and/or stability)
* Complete
* Consistent (with other requirements)
* Uniquely identifiable (usually via numbering like 3.4.5.6)

Attention should be paid to the carefully organize the requirements presented in this section so that they may easily accessed and understood. Furthermore, this SRS is not the software design document, therefore one should avoid the tendency to over-constrain (and therefore design) the software project within this SRS.>

## 3.1 External Interface Requirements.

### 3.1.1 User Interfaces

### 3.1.2 Hardware Interfaces

### 3.1.3 Software Interfaces

### 3.1.4 Communications Interfaces

## 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 <Character Movement>

#### 3.2.1.1 Description and Priority

The player will have complete control over how they shall move their character between areas on the map and rooms inside the areas. How the character maneuvers a room has an impact on which fights they will become engaged in. The priority for Character Movement is High.

#### 3.2.1.2 Stimulus/Response Sequences

The user will input movement using either a keyboard or mouse which will activate an action listener to determine where they wished to move.

#### 3.2.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

The world map and combat rooms must have squares that are available for the character to move to. If the player tries to move to a square that is not available for movement, such as a wall, they will not be able to proceed into that square.

##### 

### 3.2.2 <Unique Areas >

#### 3.2.2.1 Description and Priority

Each area the player explores on their adventure will be uniquely designed. No two areas will be the same. The areas will contain items for the character to interact with as well as numerous enemies to fight. The priority for Unique Areas is Low.

#### 3.2.2.2 Stimulus/Response Sequences

An area will be loaded when it is chosen by the player to move to. All available areas will be displayed on the player’s map.

#### 3.2.2.3 Functional Requirements

Multiple areas will need to be designed by the team for each area the player is able to travel too. The areas will be populated with environmental features, enemies, items, and objectives.

### 3.2.3 <Game Saving>

#### 3.2.3.1 Description and Priority

The player has the option to save them game whenever they wish to, but they must be out of combat. Upon starting up the game they can load up their previous save. To avoid “save scumming”, the practice of loading previous saves to undo decisions, the game will only have on save file, which will be overwritten each time they decide to save the game.

#### 3.2.3.2 Stimulus/Response Sequences

In order to save the game, the player must open up the player menu and proceed to select the “Save Game” option. The game will then save the current game file and overwrite the previous save file.

#### 3.2.3.3 Functional Requirements

Saving the game must overwrite the previous save file. The player will be prompted when the game has finished saving and that they should not exit the game whilst it is saving.

### 3.2.4 <Turn Based Combat>

#### **3.2.1.1 Description and Priority**

This gameplay aspect is fundamental to the game. Good turn based combat is the backbone of a turn based RPG. Therefore, this is a high priority requirement. In combat, the player will have the option to perform various, use items, or flee a battle. We will spend much time developing and improving the combat, as boring gameplay could cause the player to become uninterested.

#### **3.2.4.2 Stimulus/Response Sequences**

Approaching an enemy or entering some unspecified locations will trigger a battle.

#### **3.2.4.3 Functional Requirements**

A combat user interface is necessary for each battle. This user interface will show the current status of each character’s health, power points, and mana. It also contains options for the player to perform in battle. If the player attempts to perform an option that is unavailable, they must be told to make another selection.

### 3.2.5 <Character Selection>

#### **3.2.5.1 Description and Priority**

The player is the hero of this story. At the beginning of the game, they will be able to customize their own character, choosing their race, gender, and class. For race, players may pick between a dwarf, elf, or human. Classes include a warrior, rogue, and mage. The priority is high as these choices come at the very beginning and will impact the whole game.

#### **3.2.5.2 Stimulus/Response Sequences**

Upon starting a new game, the player will be brought to the character selection.

#### **3.2.5.3 Functional Requirements**

Different player sprites will need to be made for a combination of each race, gender, and class. Additionally, we will have to provide the player with different stats based on their selection. Players must be given a description of each race and class when deciding which to choose, so they are aware if the impact of their decisions.

### 3.2.6 <Stat Customization>

#### **3.2.6.1 Description and Priority**

The player is prompted to choose which areas to increase their strength upon leveling up. Whichever areas the players choose will increase by a set number of skill points.

#### **3.2.6.2 Stimulus/Response Sequences**

Upon earning enough experience points to level up a character and completing the current battle, the player will be prompted to choose which stat to increase for that character.

#### **3.2.6.3 Functional Requirements**

<We will need to make an interface for stat customization that will be shown at the end of a battle when a character levels up. The user must be able to see all current stats and how much each stat will be increased by if they were to select it.>

### 3.2.7 <Boss Battles>

#### **3.2.7.4 Description and Priority**

Bosses will be much stronger than regular foes and each boss battle will be much more intricate than a standard one. Bosses will changeup their strategies and attacks as the battle progresses, getting increasingly difficult. In some cases, the player will even need to interact with the environment to defeat the boss. Players will not be able to flee from a boss battle.

#### **3.2.7.4 Stimulus/Response Sequences**

Upon entering a boss room the player will be locked into the battle. It will be obvious when the player is approaching one of these rooms, and they will be advised to safe beforehand.

#### **3.2.7.4 Functional Requirements**

We will need to implement the same combat system, but remove the feature to flee the battle. We must also implement advanced features to increase the complexity of the battle, such as multiple targets on the boss or other elements to attack.

### 3.2.8 < Decision Making >

#### **3.2.8.1 Description and Priority**

The user can make decisions throughout the game where they can choose what character they play as, how they attack, where they move, and what items they use and when they use them. Priority level: Medium.

#### **3.2.8.2 Stimulus/Response Sequences**

Through the use of valid inputs, the user can choose what happens in the game, such as where to move.

#### **3.2.8.3 Functional Requirements**

All menus and movement capabilities must be working for the user to be able to make decisions within the game. If an invalid input is entered, the game should do nothing as it will only take valid inputs. If unsure about inputs, the user could pause the game and navigate to a help or controls screen.

3.2.9 < Item Features>

#### **3.2.9.1 Description and Priority**

   The user will be able to find various weapons, armor, and useable items throughput the playable world. All items can be switched or dropped at any time. Priority level: Medium.

#### **3.2.9.2 Stimulus/Response Sequences**

   The items must be found from there strategically place locations around the map.

#### **3.2.9.3 Functional Requirements**

            All items will need to be thought up and designed to look distinct and unique. Each item will need its own stats and effects that will affect the player when used or equipped.

3.2.10 < Character Menu>

#### **3.2.10.1 Description and Priority**

   The player can view their stats, what items they have acquired, what items are equipped, what weapons/spells are equipped, and what effects are active. While the character menu is active the game will pause itself until the menu is closed. Priority level: Medium.

#### **3.2.10.2 Stimulus/Response Sequences**

   Character menu screen can be visible at any time while in-game so long as the player is not in combat. A button already mapped out must be pressed to access the menu.

#### **3.2.10.3 Functional Requirements**

            The character menu screen must be access by the press of a button.

## 3.3 Use Cases

### 3.3.1 Use Case #1

|  |  |
| --- | --- |
| **Use Case Name** | Saving the game and Loading a save |
| **Reference** | Section 2.2.1 |
| **Trigger** | The user has either pressed the save game button or has selected the load game option at the main menu |
| **Precondition** | The user wants to save the game or load a previous save file |
| **Basic Path** | 1. Opens the menu 2. Clicks “Save Game” 3. Game saves 4. Opens the main menu 5. Clicks “Load Game” 6. Selects the desired save file |
| **Alternative Paths** | None |
| **Postcondition** | The game is either saved or the game save is loaded |
| **Exception Paths** | Player cannot save while in combat or with enemies nearby |
| **Other** | The save files are saved to a folder in the game files |

Table 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.).

### 3.5.1 Performance

### 3.5.2 Reliability

### 3.5.3 Availability

### 3.5.4 Security

<Example: The server on which the Online Journal resides will have its own security to prevent unauthorized *write*/*delete* access. There is no restriction on *read* access. The use of email by an Author or Reviewer is on the client systems and thus is external to the system. The PC on which the Article Manager resides will have its own security. Only the Editor will have physical access to the machine and the program on it. There is no special protection built into this system other than to provide the editor with *write* access to the Online Journal to publish an article.>

### 3.5.5 Maintainability

### 3.5.6 Portability

## 3.5 Design Constraints

<Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project. Example, the software is required to have a login screen based on company policies.>

## 3.6 Logical Database Requirements

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

## 3.7 Other Requirements

<Catchall section for any additional requirements that did not belong to the previous sections. If there are none, exclude this section>

# 4. Analysis Models

<List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.>

## Sequence Diagrams

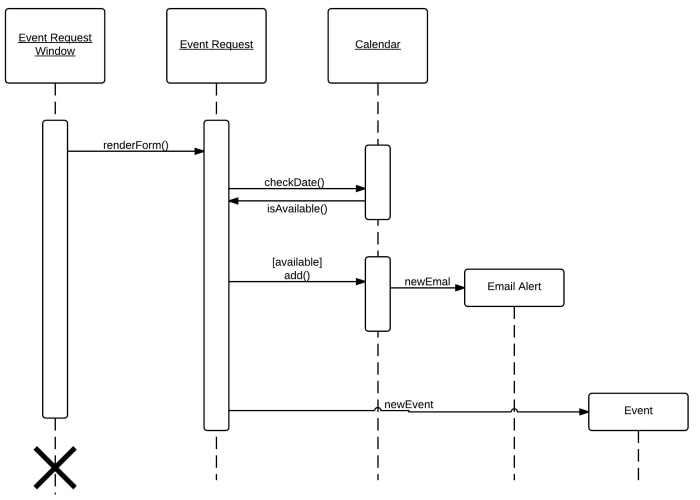


Figure 1: Data Flow Diagram Example 1

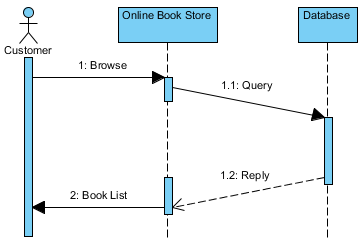


Figure 2 Data Flow Diagram Example 2

<At least one sequence diagram should be included for each requirement or use case.>

# 5. 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

# 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