SE 3XA3: Software Requirements Specification
Snake Game Remake

Team #: 302, Team Name: 404
Student: Shunbo Cui cuis13
Student: Xiangxin Kong kongx9
Student: Shuo Zhang zhans18

February 9, 2020

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
Feb 9,2020	1.0	creating the document
Date 2	1.1	Notes

This document describes the requirements for .... The template for the Software Requirements Specification (SRS) is a subset of the Volere template (**?**). If you make further modifications to the template, you should explicitly state what modifications were made.

1 Project Drivers

1.1 The Purpose of the Project

The purpose of the software application, Snake-game-remake, is to provide an enjoyable and challenging gaming experience for the user. It is inspired by the games Snake-Java-2D-game in that it focuses on the growth of a snake character through consuming objects throughout the in-game environment. The snake will be a dynamic entity that moves automatically with a constant speed, unless it gains the effect from specific items. The player can adjust the direction to left or right. The player has to operate the snake to avoid collision with the walls and barriers in the map, while also collecting the food items in the map to achieve a higher score. In order to improve the complexity of the game, more in-game functions will be provided such as random potions and trap items.

1.2 The Stakeholders

1.2.1 The Client

The client of our product is an external entity who is acting as the final reviewer of our game. They are an entity interested in remaking the "snake game" for the general public.

1.2.2 The Customers

The customer is the general public, especially teenagers, who have little leisure time but great interest in video games.

1.3 Mandated Constraints

The game is only intended for use on Java virtual machines run on windows or OS X. The only input methods allowed are with a cursor (mouse/trackpad) and a keyboard.

1.4 Relevant Facts and Assumptions

It is assumed that the target demographic has access to traditional computers with a mouse and keyboard.It is assumed that Windows and Mac OS X will be available for both the developers and users of the game.

2 Functional Requirements

2.1 The Scope of the Work and the Product

2.1.1 The Context of the Work

Some important functions are absent in this game such as leader board and restart option when game is over. Also, the choices of maps for this game are limited and the original design for the interface is rough and simple. We are going to reconstruct the user interface to enhance user experience. More gaming systems such as utility items will be introduced to the game to improve the complexity of the game. Our game is a lightweight Java game on PC which can be played by all ages especially teenagers. All the additional functions will be implemented in individual Java modules with high cohesion.

2.1.2 Work Partitioning

Table 2: Work Partitioning

Event Name	Input	Output	Summary
Snake Game Creation	Developer code	Java Virtual Machine	Recreate a Java based game implementation that works on Java Virtual Machine.
Snake Game Audio	Audio File	Audio output device	Adding more sound effects into the game.
Snake Game Sprite	Developer graphics and code	Java Virtual Machine	Adding more functions of the spirits and objects to the game.
Snake Game Collisions	Developer code	Java Virtual Machine	Creating hit detection and ending screen.
Snake Game Final Edits	Developer code	Java Virtual Machine	Finishing edits for the whole project.

2.1.3 Individual Product Use Cases

The product is designed to be used by those teenagers who enjoy video game experience. As a successful game on Nokia phones in the 1990s, it requires both quick reactions and forethought. Our product will provide improved experience of challenging themselves to get a higher score on the leaderboard. As an open-source project on public Git repository, our code also provides educational support to the developers who are interested in similar projects.

2.2 Functional Requirements

FR1. The program will be able to run on a Java virtual machine.

FR2. The game will show the main title screen when the program is successfully started.

FR3. The system will adjust the speed of the snake and the maximum number of in-game items when the user selects different difficulties on the title screen.

FR4. The player will be able to choose the game map they want to play on the title screen.

FR5. When the start game button on the title screen is pressed, a new game will be started and the gaming screen will be shown.

FR6. At the start state of the game, a snake with length of 2 and default direction to the right will be generated in the middle of the map.

FR7. The snake will start moving when the player presses the space button and a sound effect will be played.

FR8. A food item will be generated at a random empty location inside the border of the map after the player pushes the space button to start.

FR9. When the player presses the pause button, the pause screen will be shown.

FR10. When the player presses the pause button on the pause state, the playing screen will be shown again.

FR11. On the pause screen, a list of buttons including go back to the title screen and restart game will be shown.

FR12. Every time when the snake's head reaches the existing food item, the length of the snake will increase by 1 at the tail and the score will increase, also the sound effect will be played.

FR13. The food collected by the snake will be removed from the map and a new food item will be generated at random empty locations at the same time.

FR14. The current length, effect and score should be always shown on the gaming screen.

FR15. After a set time period, a new effect item will be generated on a random empty location on the map until the maximum number of items is reached.

FR16. When the snake reaches the effect item, the item is removed from the map and the corresponding effect will be applied on the snake; if there is already effect on the snake, it will be replaced with the new one.

FR17. When the snake's head hits the border wall, the barrier in the map or the snake body, the game will be stopped and a game-over screen will be shown.

FR18. The player will be able to restart the game or go back to the title screen with the buttons on the game-over screen.

FR19. The player will be able to activate the color blind option with a button on the title screen.

FR20. The colors used in the game will be adjusted to be better visible by colourblind people when the color-blind mode is activated.

3 Non-functional Requirements

3.1 Look and Feel Requirements

LF1. The game will follow Java coding structure.

LF2. The game shall be two-dimensional in design.

LF3. The game should use colors that make players easily identify the objects in the game.

3.2 Usability and Humanity Requirements

UH1. Players shall be able to customize the color for the snake before playing.

UH2. There shall be a help page available for users to find the controls and descriptions of items.

UH3. The game shall provide the objective of the game on the homepage.

UH4. The game shall be accessible for deaf users.

UH5. The game shall be playable by users with at least a single fully functional hand.

3.3 Performance Requirements

3.3.1 Speed and Latency Requirements

PR1. Any valid user input shall receive an immediate response by the game in a manner instantaneous to the user's perception (under 50ms).

PR2. Game initialization upon opening the application shall be less than 10 seconds.

3.3.2 Precision or Accuracy Requirements

PR1. All stats of the snake must be an integer value.

3.3.3 Reliability and Availability Requirements

PR1. The software must be able to initialize and operate normally, not necessarily continuously, at any time.

3.4 Operational and Environmental Requirements

3.5 Maintainability and Support Requirements

3.5.1 Maintenance Requirements

MS1. The properties of game elements shall be updated appropriately to ensure game balance.

3.5.2 Supportability Requirements

MS1. The game will have feedback and support service.

3.5.3 Adaptability Requirements

MS1. The game shall be updated based on user feedback and feature suggestions.

3.6 Security Requirements

SR1. The product shall not interact with any personal user data.

SR2. User data shall be kept private.

3.7 Cultural Requirements

CR1. The system shall not show any religious imagery or messages.

3.8 Legal Requirements

LR1. This software shall comply with all national and federal software regulation laws.

LR2. This software shall comply with all relevant software standards.

LR3. This software shall comply with all relevant privacy acts.

LR4. Design shall follow Google Java Style Guide

3.9 Health and Safety Requirements

HS1. Cacophony shall not be selected as the game sound.

4 Project Issues

4.1 Open Issues

The renewal period of software and hardware shortens continuously. These changes may make the game no longer playable or run with bugs. Apart from that, many devices do not have a physical keyboard. Users of these devices may not be able to play this game.

4.2 Off-the-Shelf Solutions

Snake game is a very classic game that has been implemented in many languages. These products have the same core game rules game but different add-ons. Some of them add multiplayer options in the game. Some others focus on adding more scenarios, game modes or more items. A java based product called "Snake2D" can be used as the prototype of this project. Other products on GitLab can be treated as source of code.

4.3 New Problems

4.3.1 current environment

Although most of the snake games online are open source projects, there are still many snake games with copyrights. The developing team should be careful not to breach copyright laws.

4.3.2 existing users

Color-blind users may have a hard time identifying the snake and items in the game.

4.3.3 Limitations in implementation environment

The java virtual machine on the local machine of the developing team processes the code slowly. (long processing time).

4.3.4 problem from "solution"

None.

4.4 Tasks

The plan for the delivery of the project and each member's role in every delivery are covered by the Gantt chart located in the "project schedule" file in the project. The plan for the meeting is covered in "development plan". In the Gantt chart, phase one tasks are in light blue color that shows the developing team will be working on structuring the project. Phase two tasks are in red color that shows the team will be constructing and building tests for the project. Also, there will be changes in requirements in phase two. The final phase tasks are colored in yellow that shows the developing team has completed the coding part of the project and will be in progress in modifying documents and code for the final product.

4.5 Migration to the New Product

None.
This project will only be used in the device that supports java.

4.6 Risks

None. As a software product, there is no physical damage to users. However there might be a risk that some users will be addicted to the game.

4.7 Costs

As this project will be available on GitLab, there will be no cost for downloading the game. But in order to play this game, users need a device that supports java and is able to connect a keyboard.

4.8 User Documentation and Training

Instructions will be shown in the game. There is no need for a separate user documentation file to guide users. Users can get training by playing this game.

4.9 Waiting Room

A possible improvement of the original project multiplayer option can be added to the game to let two players compete for food and items. This will make the game more competitive and attractive to the target users of this product. Then, the game only has one music sound that is played throughout the game. Adding more sounds will also improve the user experience.

4.10 Ideas for Solutions

For the multiplayer option, the second snake feature can be introduced into the game interface with different control buttons and color from the first snake. The two snakes will compete on their scores gaining from eating food. They will also compete for items and avoid hitting on each other. For new sounds, they will be selected from online music records that have no copyright. Background music shall differ in different maps.

5 Appendix

This section has been added to the Volere template. This is where you can place additional information.

5.1 Symbolic Parameters

The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.