SE 3XA3: Software Requirements Specification Space Shooter

Team #105, Space Shooter
Hongzhao Tan, tanh10
Nishanth Raveendran, raveendn
Dananjay Prabaharan, prabahad

April 7, 2020

Contents

List of Tables

List of Figures

Date	Version	Notes
02/09/2020	1.0	Initial Software Requirements Specification
03/13/2020	1.1	Requirements Update
04/02/2020	1.2	Rev1 Update - Updated Functional and Non-Functional Requirements

Table 1: **Revision History**

1 Project Drivers

1.1 The Purpose of the Project

The purpose of the project is to provide a fun and refreshing gaming experience to people who are missing classic games from years ago. The reference game in its current form is quick and fun, but the development team is trying to revamp the game by adding a clean user interface, new game-play features, and cleaning up the existing code to ensure easy maintenance. The new game features include increased mobility, new enemy types, and more power-ups. All of this is to create more of a challenge and keep the player more engaged.

1.2 The Stakeholders

1.2.1 The Client

The clients of this project consists of the 3XA3 Software Engineering Practice and Experience course teaching assistants, and the professor, Dr. Bokhari. The primary concern for the clients is to see that the outcome of the project is an enhanced and upgraded version of the original reference game.

1.2.2 The Customers

The target demographic of this game is aimed at people who are looking for a casual gaming experience. This includes adults who miss the nostalgic feel of older games from previous eras should find comfort in playing Space Shooter. Additionally, kids who want to have fun playing a video game are included as a customer. Generally, the consumer base consists of anyone that has access to a Windows system that is looking for a fun and quick way to entertain themselves.

1.2.3 Other Stakeholders

There are many other stakeholders involved in this project. The development team is a major stakeholder, as they are responsible for upgrading the original reference game. The development team has their own desires for what the overall outcome should be, some of which include technical aspects such as efficiency, run time, and code cleanliness. The general public is a stakeholder since they will be exposed to the game and make up the consumer base. The desires of the general public are heavily considered in determining the features added, and user experience updates.

1.3 Mandated Constraints

The main mandated constraints for this project involve the due dates for project deliverables. The final version of the project must be complete with documentation by April 6, 2020. The project must not be made from scratch - it must have a reference material and be a redesign of it. This means the final project will have a similar design and functionality of the original, but also additional unique features. The project must be able to run on Windows systems and be played with a keyboard.

1.4 Naming Conventions and Terminology

For this project, the development team will be using the camel case naming standards. Variables, classes, and files will be named with descriptive names to provide a little understanding of the purpose of them.

1.5 Relevant Facts and Assumptions

1.5.1 Relevant Facts

The original space shooter reference game consists of one Python file with 560 lines of code. The game is based on and inspired by the original Nintendo game, "Space Invaders".

1.5.2 Assumptions

The project team is making a few assumptions on user characteristics, as we do not have access to research and conduct surveys for our user base. The game will be available on Windows systems, so it is assumed that the user has a working keyboard to play the game. It is also assumed that the user is familiar with common symbols used in games, such as the health bar and power ups.

2 Functional Requirements

2.1 The Scope of the Work and the Product

The scope of the Space Shooter project is to revamp the existing game with additional unique features such as further direction control capabilities and more interactive objects. The development team also plans on fixing the existing project using common coding practices by modularizing the code and adding more functions.

2.1.1 The Context of the Work

The Space Shooter game is an arcade game where the user must survive in its spaceship for as long as possible while avoiding getting hit by upcoming meteors and aliens. The player can control the spaceship by moving left, right, up and down on the screen and shooting bullets. The intended audience of the Space Shooter project are kids and adults who miss the nostalgic feel of older arcade games.

2.1.2 Work Partitioning

Event Name	Input/Output	Summary
Destroy Asteroid/Aliens	Spaceship shoots multiple bullets to respective alien (out)	The user destroys an upcoming obstacle by shooting a bullet to the asteroid/alien. Some objects may be required to be hit more than others in order to be destroyed.
Spaceship Destruction	Spaceship loses all health (out)	The user's spaceship is destroyed in the game when it loses all its health after being hit multiple times by incoming obstacles.
Spaceship Movement	User enters in directional keyboard input (in)	The spaceship moves according to the user's input on the keyboard (right arrow for moving right, left arrow for moving left, and etc)

2.1.3 Individual Product Use Cases

The primary use case of the product is for the game to be played with. The user would have a fun experience when playing the game and have a nostalgic feeling of the 1980s.

2.2 Functional Requirements

F1: The game is intended to run offline on a Windows or MacOS operating system.

F2: The game shall have 4 main states: "Pre-Game State", "Playing Game State" and "Paused State".

F3: The main-menu would be in the "Pre-Game State". The game shall display the main menu when first launched by the user. The main menu would display three options: Play, Instructions, and Quit.

F4: If the user selects the Play option, the game shall proceed to the "Playing Game State" where the user plays the main game in.

F5: If the user selects the Instruction option, the game shall proceed to the Instruction Screen.

F6: As the user is in the "Playing Game State", the game shall redirect to the "Paused State" if the user clicks the pause button during the game.

F7: If the unpaused button is clicked in the "Paused State", the game shall resume back to the "Playing State" where the user left their progress in.

F8: The "Playing Game State" shall always contain the score, number of lives and health bar at the top.

F9: The score in the "Playing Game State" shall always start at 0 for each new game. The score should increase by destroying meteors and aliens.

F10: The health bar in the "Playing Game State" shall always restart to 100/100 after each new game.

F11: If the spaceship is hit by an upcoming meteor from the top of the screen, the spaceship's health shall decrease by an amount proportional to the meteor radius.

F12: If the spaceship is hit by an upcoming alien from the top of the screen, the spaceship's health shall decrease by 20 percent.

F13: The spaceship shall have the ability to move in 4 directions: up, down, left, and right.

F14: The spaceship shall have the ability to shoot upcoming obstacles with bullets. The spaceship would have unlimited bullets with strengths determined by its power-ups.

F15: If the spaceship collides with the "Faster Shooting" power-up, the spaceship shall shoot bullets twice the speed for 5 seconds.

F16: If the spaceship collides with the "Double Shooting" power-up, the spaceship shall shoot two bullets at a time for 5 seconds.

F17: If the spaceship collides with the "Shield" power-up, the spaceship gain a boost of 30% of health.

F18: If the spaceship collides with the "Missle" power-up, the spaceship begins shooting missles for 5 seconds. The missles shall continue colliding with upcoming obstacles even if it had already hit one in the past, unlike the bullets.

F19: If the spaceship contains 0/100 health during the "Playing Game State", the game shall redirect to the "Pre-Game State" where it displays the main-menu.

3 Non-functional Requirements

3.1 Look and Feel Requirements

NF1: Appearance Requirements
After starting, the product shall generate a multicolored space ship at the bottom middle of the user interface with a background of an "space" scene. The generated space ship shall be colored to be well-marked and able to keep user's attention on it while using the product.

Space Shooter shall generate bright colored laser from the space ship upon users' requests. Space Shooter shall generate solid colored meteorites. The product shall display an animation of explosion upon collisions between certain in-game objects.

The colors of the background shall not cover up color of any in-game object. All text in Space Shooter shall be written in reasonably sized fonts. The program shall be sized to properly fit on majority types of electronic devices' screen.

NF2: Style Requirements
Space Shooter shall generate meteorites in a pace that the meteorites can make the users feel intense atmosphere while not making the user interface too cluttered.

3.2 Usability and Humanity Requirements

NF3: Ease of Use Requirements
The product shall be able to be used by any person with appendages

NF4: Ease of Learning Requirements
The user shall be able to operate a keyboard and a mouse and read English.

3.3 Performance Requirements

NF5: Speed Requirements
The product shall respond to users' requests sent to the system immediately by human perception.

NF6: Safety Critical Requirements
The product shall not do harm to users' data and device.

NF7: Reliability and Availability Requirements
The product shall be available whenever it is installed on users' local devices at any time when the devices' random access memory is under maximum load. The product's defect rate shall be less than 1 failure per 200 hours of operation.

NF8: Capacity Requirements
The game shall not exceed uses' local devices' random access memory.

3.4 Operational and Environmental Requirements

NF9: Operational Requirements
Users shall need to install the external software of the product on their devices to run Space Shooter.

NF10: Expected Physical Environment
The Product shall be able to be used in anywhere there is a device with Space Shooter installed on it and enough random access memory to run the system, a keyboard and a mouse that can be used to control the space ship.

NF11: Expected Technological Environment
The product shall be used on many types of devices that can open a window big enough to fit in the user interface. The medium that will be carrying out the product's user interface will be the generated window on devices' screen.

3.5 Maintainability and Support Requirements

NF12: Maintainability Requirements
The product shall be properly modularized so that modification on a specific function of the product shall be easily implemented.

NF13: Support Environment
The product shall ensure the target clients are capable of running it on their devices.

3.6 Security Requirements

NF14: Access Requirements
The entire general public shall have access to the necessary infrastructure of the product.

NF15: Integrity Requirements
Source code of the product shall not be altered by any operation on the user interface (playing the game).
The product shall reject invalid user requests and inputs.

NF16: Privacy Requirements
The product shall not leak out any data outside itself from users' devices.

3.7 Cultural Requirements

NF17: Cultural Requirements
The product shall not contain any element(symbols, text, etc.) that could be considered as offensive to any culture.
The product shall be available in English.

3.8 Legal Requirements

NF18: Legal Requirements
The product shall not violate any law.
The product shall not contain any symbol, typeface and logo that requires license which the development team don't have.

3.9 Health and Safety Requirements

NF19: Health and Safety Requirements
The product shall not do any harm to human body
The product shall suggest the users to stop using itself when it has been continuously used for a long time.

4 Project Issues

4.1 Open Issues

Since technology in these days evolves really fast, the hardware and software in the future could be extraordinarily different from what we have now. It is hard to forecast whether the product will work fine in the future or not and how should the product react to the coming changes. In addition, the product has not been test on all main operating systems on electrical devices (PC, Mobile) and it is also not known if the product will work on new created operating systems that will be employed by majority of popularity in the future. Also, because the users' locals device will be hosting the product, the issues of the local machines would affect the product.

4.2 Off-the-Shelf Solutions

Since the original implementation is from an open source project, the development team could rely on it as a prototype and source of code.
Also, there are currently plenty of space shooter games in different languages which could provide the development team some ideas on implementing Space Shooter.

4.3 New Problems

Space Shooter is not likely to be the only software application in users' devices. Nevertheless, if the random access memory of user's device is not enough to run all of the application that is simultaneously working, the product could cause shut down to other running application in the device.
In addition, since there are many other implementation of the space shooter type games, the development team shall be careful on not to infringe on others' copyrights.

4.4 Tasks

The schedule of deliverables and tasks are all covered in the dynamic Gantt Chart of the development team which can be found from the following link: `https://gitlab.cas.mcmaster.ca/raveendn/space-shooter/blob/master/ProjectSchedule/Project%20Gantt%20Chart.gan`

4.5 Migration to the New Product

None

4.6 Risks

There is very little risk for the product to do harm to user's body physically, since it is a game. However, the development team still shall be careful on the risks that could injure the users indirectly such that the product shall not be too poorly optimized which could slow down or even overheating the user's device.

4.7 Costs

As long as the development tools and documentation tools remain free, and the original reference product keeps being open-source, there will be no cost for developing the product.

4.8 User Documentation and Training

The development team shall use the (""") format for docstring to give description to all of the modules, classes and functions in the source code of the product written with Python.

4.9 Waiting Room

In addition to the objects that could destroy the space ship in the original project, new enemy types would be added into the game. Powerups to the lasers and additional mobility would be also added to the in-game space ship.

4.10 Ideas for Solutions

The development team could define new Python classes for the new enemy types and powerups to the lasers, and modifying the original source code in the module for defining the movement of space ship to add mobility to the ship.

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.