SE 3XA3: Software Requirements Specification CraftMaster

Group 307, 3 Craftsmen
Hongqing Cao 400053625
Sida Wang 400072157
Weidong Yang 400065354

April 7, 2020

Contents

List of Tables

List of Figures

Date	Version	Notes
Jan 27	1.0	Team information
Feb 9	1.1	Edited Functional Requirements and Non-functional Requirements
Feb 9	1.2	Added Reference
Feb 9	1.3	Edited other sections
Feb 11	1.4	Renamed project
Mar 12	1.5	Modified Functional Requirements
Mar 24	1.6	Modified Functional Requirements for Demo Rev0
April 5	2.0	Completed for Rev1

Table 1: **Revision History**

This document describes the requirements oriented by business events. It includes a set of use cases that describe the user interactions in the scenarios triggered by business events and also lays out functional and non-functional requirements in detail. The format of this document follows the template provided by Dr.Bokhari and Thien Trandinh.

1 Project Drivers

1.1 The Purpose of the Project

1.1.1 The User Business or Background of the Project Effort

In the current gaming market, the age limitation on video games builds boundaries between children and adults. Most 3D games available in the market are not available to children since they contain violent content. It should be expected to see more video games that are available and beneficial to children and teenagers.

1.1.2 Goals of the Project

We want to provide an interesting and inspiring video game for children to unleash their creativity.

1.2 The Stakeholders

1.2.1 The Client

Since this project is considered as a reimplementation of an existing open-sourced software game, it cannot be invested and be used for financial purposes. Therefore, the client of this project will only be Dr.Bokhari and all the TAs in the SFWRENG 3XA3 course at the current stage. For further development, with the authorization from the original software designer, this software game project is expected to be deployed on PC Video Game Distribution Agencies such as Steam. By that time, the client will include Video Game Distribution Agencies.

1.2.2 The Customers

In general, the customers of this project will be PC gamers who have a passion for Sandbox games. Due to the simplicity and user-friendliness, the main customer group specifically refers to young gamers aged 7-13.

1.2.3 Other Stakeholders

Other stakeholders aside from the clients and customers are listed below:

- The software development team, including Sida Wang, Hongqing Cao, and Weidong Yang
- Legal domain experts in the domain of youth game design
- Safety domain experts
- Software game testers

1.3 Mandated Constraints

The project should be designed and developed conforming with the following constraints:

1.3.1 Solution Constraints

- **Description:** The product shall operate using either Windows or Linux OS.
  Rationale: The user uses either Windows or Linux OS.
  Fit Criterion: The executable file shall be approved running on either Windows or Linux OS by the development team.

1.3.2 Implementation Environment of the Current System

- The processor of the hardware system where the product is to be installed must be capable to run a simple 3D game.
- The product is developed using Python, so the development environment must have Python installed.
- The executable file of the product will be independent of Python, which means the OS that runs the game will not be required Python installed.

1.3.3 Partner or Collaborative Applications

- The GUI of the game will be generated by Pyglet. Therefore, the development environment will require Pyglet package installed.
- The executable file will be generated by Pyinstaller. Hence, the development environment will require Pyinstaller package installed too.

1.3.4 Off-the-Shelf Software

- The development of this product shall follow and refine the original design, which is Fogleman's Minecraft software program[1](which is refered to as **original project** in the following content).

1.3.5 Anticipated Workplace Environment

- The sound experience will be negatively affected if the workplace is noisy.

1.3.6 Schedule Constraints

- The project must be completed within a three-month period starting from February to April.

1.3.7 Budget Constraints

- None

1.4 Naming Conventions and Terminology

The naming conventions and terminology section will aid readers from different backgrounds to clearly understand the content of this document. The naming conventions and terminologies used in this document are listed below:

- **TA:** Teaching Assistant
- **OS:** Operating System
- **GUI:** Graphical User Interface, which allows the user to interact and visualize the program by graphics instead of text

- **PC Gamers:** The individuals who play games either on a desktop or on a laptop
- **Sandbox Games:** A type of game that allows player to create, modify, and destroy the environment
- **Python:** The programming language that is used for the development of this project
- **Pyglet:** The Python library for the design of graphical user interface
- **Pyinstaller:** The tool for containing a Python application and generating an executable file to that application.
- **Player/Gamer:** The person who controls the PC to play the game
- **Character:** The fictional character who is controllable by the player/gamer(not visible to the player).
- **3D Game:** A game in three dimensions.
- **Video Game Distribution Agencies:** The third-party agencies that deliver video game content as digital information.

1.5 Relevant Facts and Assumptions

1.5.1 Facts

- The original project is 902 lines of Python code.
- The project is built with Python and its GUI library Pyglet.
- Tests feedback will come from the assigned game testers.
- This is not a profit-oriented project unless the financial purpose is approved and authorized by the original project owner.
- The executable file will be built by Pyinstaller and requires up to 2GB PC memory.

1.5.2 Assumptions

- To compile the source code in the terminal, Python 3.7 and Pyglet package will be required to be installed on the OS.
- To open the game by clicking the executable file icon, no specific software is required to be installed.
- The executable file generated by Pyinstaller will keep all the features from the compiled application.
- The Pyglet library will provide sufficient functionalities for the reimplementation.
- **User Characteristics:**
- The user is capable to play 3D video games with no dizziness.
- The user is passionate about 3D Sandbox games.
- The user has a PC that can run a simple 3D game.
- The user has knowledge of Python compilation and OS operations in the case of running the program in the terminal.

2 Functional Requirements

2.1 The Scope of the Work and the Product

2.1.1 The Context of the Work

CraftMaster will be developed based on Python 3.7 as programming language and Pyglet library as GUI framework. It will use open-source Minecraft textures and open-source arcade musics and sounds as multimedia support. CraftMaster is also a reimplementation of Fogleman's original project[1]. The original project is built within one module. The development team of CraftMaster will apply advanced algorithms, software design principles and software architectural design patterns to optimize the quality of the software program. After the development, the team will generate executable game files using Pyinstaller and there will be two versions of the game that can be supported on Windows and Linux OS. The context diagram shown below graphically shows contribution of each component to the game design and development.

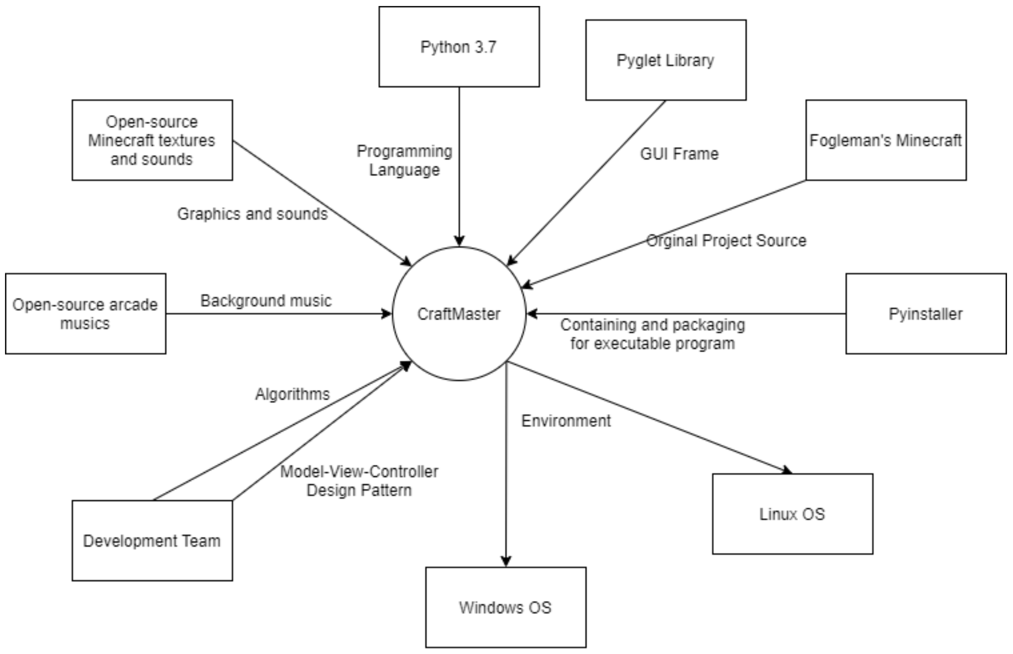

Figure 1: **Context Diagram**

2.1.2 Work Partitioning

Event Name	**Input and Output**	**Summary**
Player opens the game	Game Open Request(IN) Game GUI with main menu rendered(OUT)	The system opens the GUI and loads the main menu when the player requests to open the game
Player opens the game menu	Game Menu Request(IN) Game GUI with game menu rendered(OUT)	The system updates the GUI and loads the game menu when the player clicks on the game menu button on the main menu

Player opens the setting menu	Setting Menu Request(IN) Game GUI with setting menu rendered(OUT)	The system updates the GUI and loads the setting menu when the player clicks on the setting menu button on the main menu
Player quits the game	Game Quit Request(IN) Game GUI closes(OUT)	The system terminates when the player clicks on the quit button on the main menu
Player starts new game	New Game Request(IN) Game GUI with New Game Scene rendered(OUT)	The system updates the GUI and loads a new game scene(world) and initialize the game character when the player clicks on the new game button on the game menu
Player starts saved game	Load Saved Game Request(IN) Saved Game Selection(IN) Game GUI with Saved Game Scene rendered(OUT)	The system updates the GUI and loads a saved game scene(world) and initialize the game character when the system receives load saved game request and saved game selection
Player changes Day-Night Mode	Day-Night Mode Switch Request(IN) Day-Night Change(OUT)	The system changes the Day-Night Mode setting when the player requests a Day-Night Mode change on the setting menu or on the ESC menu
Player moves the character	Movement Keyboard input(IN) Character Movement(OUT)	The system moves the character according to the movement keyboard input initiated by the player

Player builds a block	Block Build mouse input(IN) Block Type selection keyboard input(IN) New Block(OUT)	A new block is built by the system at a certain place when the system receives Block Build mouse input and Block Type selection keyboard input initiated by the player
Player destroys a block	Block Destroy mouse input(IN) Block Remove(OUT)	A certain block is destroyed by the system when the system receives Block Destroy mouse input initiated by the player
Player changes the direction of the character	Direction Change mouse input(IN) New Direction(OUT)	The system changes the direction of the character when the system receives Direction Change mouse input initiated by the player
Player lets the character jump	Jump keyboard input(IN) Character Jump(OUT)	The system lets the character jump when the system receives Jump keyboard input initiated by the player
Player enables the character to fly	Flying Mode Toggle keyboard input(IN) Flying Mode Activation(OUT)	The system activates the flying mode when the system receives Flying Mode Toggle keyboard input initiated by the player
Player opens the ESC menu	ESC Menu keyboard input(IN) Game GUI with ESC menu rendered(OUT)	The system updates the GUI and loads the ESC menu when the system receives ESC Menu keyboard input initiated by the player

Player saves the game scene	Game Saving Request(IN) Saving Spot Selection(IN) Saved Game Data File(OUT)	The system saves the game scene when the player requests to save the game and select a saving spot

Table 2: **Business Events**

2.1.3 Individual Product Use Cases(PUC)

PUC No.	**Event Name**	**Trigger**	**Preconditions**	**Procedure**	**Outcome**
1	Open the game	The game executable file is clicked	The game executable is downloaded and installed	1. The player double left-clicks on the game executable file	The GUI is loaded with the main menu rendered
2	Open the game menu	The game menu button is clicked	The GUI is at main menu	1. The player clicks on the game menu button on the main menu	The GUI is updated with the game menu rendered
3	Open the setting menu	The setting menu button is clicked	The GUI is at main menu	1. The player clicks on the setting menu button on the main menu	The GUI is updated with the setting menu rendered
4	Quit the game	The quit button is clicked	The GUI is at main menu	1. The player clicks on the quit button on the main menu	The program terminates and the GUI closes
5	Start a new game scene	The new game button is clicked	The GUI is at game menu	1. The player clicks on the new game button on the game menu	The system updates the GUI with a new game scene(world) and initializes the character

6	Load a saved game scene	The load game button is clicked	The GUI is at game menu	1. The player clicks on the load game button on the game menu 2. The player select a saved game by clicking on the game number button (e.g. Game One)	The system updates the GUI with a saved game scene(world) and initializes the character
7	Change the Day-Night Mode	The Day-Night Mode switch is clicked	The GUI is at setting menu or at ESC menu	1. The player clicks on Day-Night Mode switch	The system changes the Day-Night Mode
8	Move the character	The key(s) on the keyboard that controls the movement is pressed	The game scene(world) is loaded and the character is initialized	1. The player presses on the movement key on the keyboard	The system moves the character to a short distance
9	Build a block	The key(s) on the keyboard that controls the build type selection and the key on the mouse that controls the build operation are clicked	The game scene(world) is loaded and the character is initialized	1. The player presses on the key(s) on the keyboard that controls the build type selection to select a block type to build 2. The player points the crosshair on the surface of a block 3. The player clicks on the key on the mouse that controls the build operation	The system places a certain type of block next to the one that is pointed to by the crosshair

10	Destroy a block	The key on the mouse that controls the destroy operation is clicked	The game scene(world) is loaded and the character is initialized	1. The player points the crosshair on the surface of a block that is to be destroyed 2. The player clicks on the key on the mouse that controls the destroy operation	The system destroys the block that is pointed by the crosshair
11	Change the direction of the character	The mouse is moved	The game scene(world) is loaded and the character is initialized	1. The player moves the mouse	The system changes the direction of the character according to the movement of the mouse
12	Let the character jump	The key on the keyboard that controls the jump operation is pressed	The game scene(world) is loaded and the character is initialized	1. The player presses on the key on the keyboard that controls the jump operation	The system lets the character jump once
13	Enable the character to fly	The key on the keyboard that controls flying mode toggle is clicked	The game scene(world) is loaded and the character is initialized	1. The player presses on the key on the keyboard that controls flying mode toggle	The system activates the flying mode
14	Open the ESC menu	The ESC key on the keyboard is clicked	The game scene(world) is loaded and the character is initialized	1. The player clicks on the ESC key on the keyboard	The GUI is loaded with ESC menu rendered

15	Save the game scene	The game saving button on the ESC menu is clicked and the game saving spot is selected	The GUI is at ESC menu	1. The player clicks on the game saving button on the keyboard 2. The player selects a game saving spot for it	The system saves the game scene at the selected game saving spot

Table 3: **Individual Product Use Cases**

2.2 Functional Requirements

- **FR1:** The software game shall start(open the GUI)when the player clicks on the icon of the game.
  Rationale: To enable the player to start the game.
  Criterion: The player clicks on the icon of the game and the game starts(GUI opens).
  Related PUC: 1

- **FR2:** The software game shall load the main menu of the game on the GUI when the game starts.
  Rationale: To enable the player to visualize the main menu.
  Criterion: The GUI is rendered with the main menu when the game starts.
  Related PUC: 1

- **FR2.1:** The main menu should have buttons linked to two sub-menus, game menu and setting menu, the sub-menus should have buttons linked to the main menu.
  Rationale: To provide the bidirectional references between the main menu and its sub-menus.
  Criterion: The sub-menus can be directed from the main menu and vice versa.
  Related PUC: 2,3

- **FR2.1.1:** The game menu should have buttons that support the functionality of playing new game(loading a new game scene) and playing

saved game(loading a saved game scene).
Rationale: To provide functionalities of playing new game and playing saved game.
Criterion: The player can start a new game or start a saved game from the game menu.
Related PUC: 5,6

- **FR2.1.2:** The setting menu should support the functionality of Day-Night mode change.
  Rationale: To provide the feature of Day-Night Mode.
  Criterion: The player can switch the Day-Night Mode on the setting menu.
  Related PUC: 7

- **FR2.2:** The main menu should have a button that supports the functionality of game quit and the software game shall terminate(close the GUI) when the player clicks on it.
  Rationale: To enable the player to quit the game.
  Criterion: The player can quit the game by clicking on the quit game button.
  Related PUC: 4

- **FR3:** The software game shall load the character of the game on the GUI and place it on the ground when the game scene is loaded.
  Rationale: To enable the player to control the character and play the game.
  Criterion: The player clicks on start new game button or load saved game button, the character will be initialized on the ground.
  Related PUC: 5,6

- **FR4:** The GUI shall place a crosshair at the center when the game scene is loaded.
  Rationale: To utilize the player with a crosshair, which supports the block build and destroy operation.
  Criterion: The player clicks on the icon of the game and can see a crosshair placed at the center of the GUI.
  Related PUC: 9,10

- **FR4.1:** The crosshair shall be kept at the center of the GUI starting from the game is loaded until the game ends.

Rationale: To ensure the stability of the position of the crosshair.
Criterion: The player plays the game for thirty minutes, the crosshair always keeps its position at the center of the GUI.
Related PUC: 9,10

- **FR5:** The software game shall move the character when the player press "W", "A", "S", and "D" keys on the keyboard.
  Rationale: To enable the movement of the character.
  Criterion: The player clicks on a certain movement key and the character moves accordingly on the GUI.
  Related PUC: 8

- **FR5.1:** The software game shall move the character forward when the player presses the "W" key on the keyboard.
  Rationale: To enable the forward movement of the character.
  Criterion: The player clicks on the "W" key on the keyboard and the character moves forward on the GUI.
  Related PUC: 8

- **FR5.2:** The software game shall move the character to the left when the player presses the "A" key on the keyboard.
  Rationale: To enable the left movement of the character.
  Criterion: The player clicks on the "A" key on the keyboard and the character moves to the left on the GUI.
  Related PUC: 8

- **FR5.3:** The software game shall move the character backward when the player presses the "S" key on the keyboard.
  Rationale: To enable the backward movement of the character.
  Criterion: The player clicks on the "S" key on the keyboard and the character moves backward on the GUI.
  Related PUC: 8

- **FR5.4:** The software game shall move the character to the right when the player presses the "D" key on the keyboard.
  Rationale: To enable the right movement of the character.
  Criterion: The player clicks on the "D" key on the keyboard and the character moves to the right on the GUI.
  Related PUC: 8

- **FR6:** The software game shall change the direction of the character according to the mouse movement controlled by the player.
  Rationale: To enable the direction change of the character.
  Criterion: The player moves the mouse and the direction of the character changes accordingly.
  Related PUC: 11

- **FR7:** The software game shall let the character jump when the player presses the space key on the keyboard.
  Rationale: To enable the jump action of the character.
  Criterion: The player clicks on the space key on the keyboard and the character jumps once.
  Related PUC: 12

- **FR8:** The software game shall toggle the flying mode when the player presses the tab key on the keyboard.
  Rationale: To enable the fly action of the character.
  Criterion: The player clicks on the tab key on the keyboard when the flying mode is off. And after that, the player can control the character to fly with "W" and "S" keys on the keyboard.
  Related PUC: 13

- **FR8.1:** The software game shall enable the flying mode when the flying mode is off and the player presses the tab key on the keyboard.
  Rationale: To ensure the correct functionality of the flying mode toggle.
  Criterion: The player clicks on the tab key when the flying mode is off, and the flying mode is turned on.
  Related PUC: 13

- **FR8.2:** The software game shall disable the flying mode when the flying mode is on and the player presses the tab key on the keyboard.
  Rationale: To ensure the correct functionality of the flying mode toggle.
  Criterion: The player clicks on the tab key when the flying mode is on, and the flying mode is turned off.
  Related PUC: 13

- **FR8.3:** The software game shall allow the player to control the character to fly using "W" and "S" keys when the flying mode is on.

Rationale: To enable the fly action of the character.
Criterion: The player clicks on the tab key on the keyboard when the flying mode is off. And after that, the player can control the character to fly with "W" and "S" keys on the keyboard.
Related PUC: 13

- **FR9:** The GUI shall show the outline of a block when the crosshair is pointing to the block.
Rationale: To enable the player to visualize the crosshair's functionality and also to support block build and destroy operations.
Criterion: The player moves mouse to point the crosshair to a block and can see the outline of it.
Related PUC: 9,10,11

- **FR10:** The software game shall remove a block when the player moves the mouse to point the crosshair to the block and left-clicks on it.
Rationale: To enable the block destroy operation.
Criterion: The player moves the mouse to point the crosshair to a block and left-clicks on it, the block will be removed.
Related PUC: 10

- **FR11:** The software game shall place a new block next to or on the top of an existing block when the player moves the mouse to point the crosshair to the existing block and right-clicks on it.
Rationale: To enable the block build operation.
Criterion: The player moves the mouse to point the crosshair to an existing block and right-clicks on it, the new block will be built.
Related PUC: 9

- **FR11.1:** The type of the block to be built shall be changed to the brick type when the player presses "1" on the keyboard.
Rationale: To enable the block type selection.
Criterion: The player clicks on the "1" key on the keyboard and moves the mouse to point the crosshair to an existing block and right-clicks on it, a new brick block will be built.
Related PUC: 9

- **FR11.2:** The type of the block to be built shall be changed to the grass type when the player presses "2" on the keyboard.

Rationale: To enable the block type selection.
Criterion: The player clicks on the "2" key on the keyboard and moves the mouse to point the crosshair to an existing block and right-clicks on it, a new grass block will be built.
Related PUC: 9

- **FR11.3:** The type of the block to be built shall be changed to the stone type when the player presses "3" on the keyboard.
  Rationale: To enable the block type selection.
  Criterion: The player clicks on the "3" key on the keyboard and moves the mouse to point the crosshair to an existing block and right-clicks on it, a new stone block will be built.
  Related PUC: 9

- **FR12:** The software game shall release the mouse movement from the GUI(stop the character direction changing) and show the ESC menu when the player presses "ESC" on the keyboard.
  Rationale: To enable the player to visualize the ESC menu.
  Criterion: The player clicks on the "ESC" key on the keyboard and the ESC menu will be shown.
  Related PUC: 14

- **FR12.1:** The ESC menu should be able to direct to the game menu.
  Rationale: To provide directional reference from ESC menu to the game menu.
  Criterion: The player clicks on the save game button on the ESC menu and it directs to the game menu.
  Related PUC: 14

- **FR12.2:** The ESC menu should support the functionalities of Day-Night mode change.
  Rationale: To provide the feature of Day-Night Mode.
  Criterion: The player can switch the Day-Night Mode on the ESC menu.
  Related PUC: 7

- **FR12.3:** The ESC menu should support the functionalities of game saving.
  Rationale: To enable the player to save the game.

Criterion: The player can save the game on the ESC menu.
Related PUC: 15

- **FR13:** The software system shall be able to save up to two game scenes for the player to play at a later time.
Rationale: To provide sufficient capacity for game saving.
Criterion: The player can choose a spot(Game One or Game Two) to save the game on the ESC menu.
Related PUC: 15

- **FR14:** The software game shall play the background music when the game starts.
Rationale: To provide more entertainment.
Criterion: The player opens the game and the background music starts to play.
Related PUC: 1

- **FR15:** The software game shall play the sound effect when a block is being built or destroyed.
Rationale: To notify the player of the block build or destroy operation.
Criterion: The player destroys or builds a block and should be able to hear the sound effect.
Related PUC: 9,10

- **FR16:** The player shall not be able to move through blocks.
Rationale: To ensure the stability of the character movement.
Criterion: The player moves the character to a block and the character should not move through it.
Related PUC: 8

- **FR17:** The marble block shall not be built or destroyed.
Rationale: To support the special feature of the marble blocks.
Criterion: The player controls the character to attempt to destroy a marble block and the marble block cannot be destroyed.
Related PUC: 9,10

- **FR18:** The keys on the keyboard that are not assigned with any task should not trigger any event to the system or cause the system crash.
Rationale: To avoid the unexpected behaviour of the system caused by trivial keyboard inputs.

Criterion: The player opens the game and start a new game, click on a trivial key and the system does nothing to respond and also does not crash.
Related PUC: 8, 12, 13, 14

3 Non-functional Requirements

3.1 Look and Feel Requirements

3.1.1 Appearance Requirements

- **NFR1:** The software game shall be attractive to a teenage player.
Fit Criterion: **Survey 1** is distributed to the players and the result confirms that the players find the game is attractive to teenage players.

3.2 Usability and Humanity Requirements

3.2.1 Ease of Use Requirements

- **NFR2:** Players aged MIN_AGE+ shall be able to play the game with no difficulty.
Fit Criterion: **Survey 2** is distributed to the players and the result confirms that the players find that players aged MIN_AGE+ could play the game with no difficulty.

3.2.2 Personalization and Internationalization Requirements

- None

3.2.3 Learning Requirements

- **NFR3:** The learning curve of this game shall vary from LEARN_MIN to LEARN_MAX minutes depending on the age of the player.
Fit Criterion: **Survey 3** is distributed to the players and the result confirms that the players find that the learning curve of this game varies from LEARN_MIN to LEARN_MAX.

3.3 Performance Requirements

3.3.1 Speed and Latency Requirements

- **NFR4:** The response shall be fast enough to avoid interrupting the user's flow of thought.
  Fit Criterion: The software game responds in less than RESPONSE second for 99 percent of the interrogations. No response takes longer than FALSE_RESPONSE second.

3.3.2 Precision or Accuracy Requirements

- None

3.3.3 Reliability and Availability Requirements

- **NFR5:** The software game should be available for play HOURS per day, DAYS per year.
  Fit Criterion: 100 times access tests at different time will be applied and the result is 100 successful accesses.

- **NFR6:** The software game shall be able to keep running for a maximum of MAX_HOUR hours.
  Fit Criterion: A MAX_HOUR long time program duration test will be applied 3 times and the result is all pass.

3.4 Operational and Environmental Requirements

3.4.1 Requirements for Interfacing with Adjacent Systems

- **NFR7:** The software game shall not perform negative actions on other programs running on the same system.
  Fit Criterion: The software game will be tested with different types of other software programs running and it does not bring negative effects to other programs.

3.5 Maintainability and Support Requirements

3.5.1 Maintenance Requirements

- **NFR8:** An update must be applied to the software game when a bug is revealed.
  Fit Criterion: Every time there is a bug revealed, the team updates the product.

3.5.2 Supportability Requirements

- None

3.5.3 Adaptability Requirements

- **NFR9:** The software game shall be able to run on Windows and Linux OS.
  Fit Criterion: The software game will be tested on both a Windows OS and a Linux OS and the test cases all pass.

3.6 Security Requirements

3.6.1 Integrity Requirements

- **NFR10:** The software game shall protect itself from intentional abuse.
  Fit Criterion: Specific threat test cases will be made and the threats are prevented.

3.6.2 Privacy Requirements

- None

3.7 Cultural Requirements

3.7.1 Cultural Requirements

- **NFR11:** The software game shall not be offensive to any religious or ethic groups.
  Fit Criterion: **Survey 4** is distributed to the players and the result confirms that the players find that this game is not offensive to any religious or ethic group.

3.7.2 Political Requirements

- None

3.8 Legal Requirements

3.8.1 Compliance Requirements

- **NFR12:** The product shall not violate the Digital Millennium Copyright Act[2].
  Fit Criterion: A legal expert will be asked to check if this game confirms the Digital Millennium Copyright Act and the result is meeting.

3.9 Health and Safety Requirements

- **NFR13:** The product shall not generate any mental or physical threat to the safety of the players.
  Fit Criterion: A safety domain expert will be asked to check if this game meets the safety standard and the result is meeting.

3.10 Robustness Requirements

- **NFR14:** Any key with no task assigned should not trigger any event to the system.
  Fit Criterion: The tester opens the game and clicks on a key on the keyboard that is not assigned any task, nothing happens to the system and the system does not crash.

4 Project Issues

4.1 Open Issues

Our investigation into whether Pyglet is the best fit GUI library for the design of this game product is still an open issue.

4.2 Off-the-Shelf Solutions

The original project designed and developed by Folgleman [1] will be considered as the candidate component for the development of this software game. Online open-source Minecraft textures and sounds will be considered as source elements to create graphics and sounds of the game.

4.3 New Problems

One potential problem of this software game could be that the player feels it is too similar to the well-known Minecraft game released by Microsoft. Another problem is that the further development for game features extension could be difficult due to the limitation of Pyglet.

4.4 Tasks

The project development plan can be found **HERE**.

4.5 Migration to the New Product

None

4.6 Risks

Due to time constraints, the development team might not be able to fully learn all the functionalities provides by Pyglet. Therefore, some features that are planned out might not be able to be fully implemented according to this requirement specification. This document will be revised in the case that some features or functionalities need to be removed.

4.7 Costs

Since all the sources(including textures, sounds, original program) and tools(including Python, Pyglet, Pyinstaller) are open-source, there will be no monetary cost for the development. The time cost of the development will be 3 months.

4.8 User Documentation and Training

A beginner's manual, also called **User Guide** will be added to the project as a guideline of how to install and play the game. The beginning manual will include all the basic operations that the player could do. With the help of the manual, the player will be able to learn the game.

4.9 Waiting Room

The toolbox feature is still under consideration.

4.10 Ideas for Solutions

None

5 Appendix

5.1 NFR Surveys

- **Survey 1:** The survey will ask for the attractiveness(a rank between 1-10), the average of the result should be above **AVG1** to pass the test.
- **Survey 2:** The survey will ask for the difficulty of the game(a rank between 1-10), the average of the result should be below **AVG2** to pass this test.
- **Survey 3:** The survey will ask for the learning time of the game, the average of the result should be between **LEARN_MIN** to **LEARN_MAX** minutes to pass this test.
- **Survey 4:** The survey will ask for the satisfaction of the cultural politeness of the game(a rank between 1-10), the average of the result should be above **AVG1** to pass this test.

5.2 Symbolic Parameters

- **MIN_AGE: 7**
- **HOURS: 24**
- **DAYS: 365**
- **MAX_HOUR: 10**
- **AVG1: 7**
- **AVG2: 3**
- **LEARN_MIN: 2**
- **LEARN_MAX: 30**
- **RESPONSE: 0.1**
- **FALSE_RESPONSE: 0.5**

References

[1] Fogleman, “fogleman/Minecraft,” GitHub, 16-Feb-2019. [Online]. Available: https://github.com/fogleman/Minecraft. [Accessed: 10-Feb-2020].

[2] ”Video Games and the law: Copyright, Trademark and Intellectual Property”, NewMediaRights, 2020. [Online]. Available: https://www.newmediarights.org/guide/legal/Video_Games_law_ Copyright_Trademark_Intellectual_Property. [Accessed: 06- Feb- 2020].