SE 3XA3: SRS

April 11, 2022

Team Information:

Team Number	Name	MACID
L03 G07	Qianlin Chen	chenq84
	Jiacheng Wu	wuj187
	Tingyu Shi	shit19

Table 1: Revision History

Date	**Developer(s)**	**Change**
January 26, 2022	All team members	Initial Document
February 9, 2022	Qianlin Chen	Functional Requirements
February 10, 2022	Jiacheng Wu	Nonfunctional Requirements, Project Drivers, Project Constraints
February 10, 2022	Tingyu Shi	Nonfunctional Requirements, Project Issues
April 8, 2022	All team members	Revise Document

Contents

1 Project Drivers **6**
1.1 The purpose of the Project . . . 6
1.1.1 The User Business or Background of the Project Effort . . . 6
1.1.2 Goals of the Project . . . 6
1.2 Stakeholders . . . 6
1.2.1 The Client . . . 6
1.2.2 The Customer . . . 6
1.2.3 Other Stakeholders . . . 6
1.3 Users of the Product . . . 6
1.3.1 The Hands-On Users of the Product . . . 6
1.3.2 Priorities Assigned to Users . . . 7
1.3.3 User Participation . . . 7

2 Project Constraints **7**
2.1 Mandated Constraints . . . 7
2.1.1 Solution Constraints . . . 7
2.1.2 Implementation Environment of the Current System . . . 7
2.1.3 Partner or Collaboration Applications . . . 7
2.1.4 Off-the-Shelf Software . . . 7
2.1.5 Anticipated Workplace Environment . . . 8
2.1.6 Schedule Constraints . . . 8
2.1.7 Budge Constraints . . . 8
2.2 Naming Conventions and Definitions . . . 8
2.3 Relevant Facts and Assumptions . . . 8
2.3.1 Facts . . . 8
2.3.2 Assumptions . . . 8

3 Functional Requirements **9**
3.1 The Scope of Work and Product . . . 9
3.1.1 The Context of the Work . . . 9
3.1.2 Work Partitioning . . . 11
3.1.3 Individual Project Use Cases . . . 12
3.2 Functional Requirements . . . 14
3.2.1 System Viewpoint . . . 14
3.2.2 User Viewpoint . . . 15

4 Nonfunctional Requirements **16**
4.1 Look and Feel Requirements . . . 16
4.1.1 Appearance Requirements . . . 16
4.1.2 Style Requirements . . . 16
4.2 Usability and Humanity Requirements . . . 16
4.2.1 Ease of Use Requirements . . . 16
4.2.2 Personalization and Internationalization Requirements . . . 16
4.2.3 Learning Requirements . . . 16
4.2.4 Understandability and Politeness Requirements . . . 16
4.2.5 Accessibility Requirements . . . 16

4.3 Performance Requirements . . . 17
4.3.1 Speed and Latency Requirements . . . 17
4.3.2 Safety-Critical Requirements . . . 17
4.3.3 Precision of Accuracy Requirements . . . 17
4.3.4 Reliability and Availability Requirements . . . 17
4.3.5 Robustness and Fault-Tolerance Requirements . . . 17
4.3.6 Capacity Requirements . . . 17
4.3.7 Scalability or Extensibility Requirements . . . 17
4.3.8 Longevity Requirements . . . 17
4.4 Oerational and Environment Requirements . . . 17
4.4.1 Expected Physical Environment . . . 17
4.4.2 Requirements for Interfacing with Adjacent Systems . . . 18
4.4.3 Productization Requirements . . . 18
4.4.4 Release Requirements . . . 18
4.5 Maintainability and Support Requirements . . . 18
4.5.1 Maintenance Requirements . . . 18
4.5.2 Supportability Requirements . . . 18
4.5.3 Adaptability . . . 18
4.6 Security Requirements . . . 18
4.6.1 Access Requirements . . . 18
4.6.2 Integrity Requirements . . . 19
4.6.3 Privacy Requirements . . . 19
4.6.4 Audit Requirements . . . 19
4.6.5 Immunity Requirements . . . 19
4.7 Cultural and Political Requirements . . . 19
4.7.1 Cultural Requirements . . . 19
4.7.2 Political Requirements . . . 19
4.8 Legal Requirements . . . 19
4.8.1 Compliance Requirements . . . 19
4.8.2 Standards Requirements . . . 19
4.9 Health and Safety Requirements . . . 19

5 Project Issues **20**
5.1 Open Issues . . . 20
5.2 Off-the-Shelf Solutions . . . 20
5.2.1 Ready-Made-Product . . . 20
5.2.2 Reusable Components . . . 20
5.2.3 Products That Can Be Copied . . . 20
5.3 New Problems . . . 20
5.3.1 Effects on the Current Environment . . . 20
5.3.2 Effects on the Installed Systems . . . 20
5.3.3 Potential User Problems . . . 20
5.3.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product . . . 21
5.3.5 Follow-Up Problems . . . 21
5.4 Tasks . . . 21
5.4.1 Project Planning . . . 21

5.4.2 Planning of the Development Phases . . . 21
5.5 Migration to the New Product . . . 21
5.6 Risks . . . 21
5.7 Costs . . . 22
5.8 User Documentation and Training . . . 22
5.8.1 Documentation . . . 22
5.8.2 Training . . . 22
5.9 Waiting Room . . . 22
5.10 Ideas for Solutions . . . 22

6 Reference **23**

List of Tables

1 Revision History . . . 2
2 Work Partitioning Table . . . 11

List of Figures

1 OLD Context Diagram . . . 9
2 NEW Context Diagram . . . 10
3 OLD Use Case Diagram . . . 12
4 NEW Use Case Diagram . . . 13

1 Project Drivers

1.1 The purpose of the Project

1.1.1 The User Business or Background of the Project Effort

Space Invaders is a popular game on the internet and the corresponding open source project is also loved by people on GitHub. We see the potential of this game and we found that there are some improvements can be done with this game. If we change some outdated features of this game, it should be really fun. After our project is delivered, players can have better experience with the new version of Space Invaders game.

1.1.2 Goals of the Project

This project is to redevelop a game called Space Invaders with some new features. The aim of this project is to produce an interesting, addictive and multi-player aircraft shooting game with a good graphic quality. We expect a service goal that over 80 % of the players will rate this game over 3/5. Another goal is to practice software engineering principles during the process of development.

1.2 Stakeholders

1.2.1 The Client

The clients of this project are Dr. Asghar Bokhari and TAs for course 3XA3. The clients want to redevelop this game since they want to add some features to this game and let developers to practice software engineering principles. Dr. Asghar Bokhari and TAs will evaluate the delivery of final product and the process of development.

1.2.2 The Customer

The customers of this project are the players who are interested in playing game Space Invaders. As long as players can meet the software and hardware requirements, for example, using Windows, MacOS or Linux, then he or she can be the customer.

1.2.3 Other Stakeholders

- Members of group 7 are stakeholders of this project. Team members will be responsible for writing documents, designing modules, implementing and testing.
- Other software developers are also stakeholders since this project is public on GitLab and other software engineers can also redevelop it.

1.3 Users of the Product

1.3.1 The Hands-On Users of the Product

This game is for anyone who knows the basic knowledge of how to operate personal computers. The player's role is to give input to manipulate the aircraft to move or attack. The way to give input is by pressing the keyboard. This game is suitable for people from all different age groups.

1.3.2 Priorities Assigned to Users

- Key users: The oriented audience of the game are video game lovers who want to spend time playing video games.
- Secondary users: Students who want to kill time playing games or learn to do a project of a simple game.
- Unimportant users: Kids under 12 who want to gain hand-eye coordination and problem solving skills.

1.3.3 User Participation

The players should spend at least 10 mins on this game and shall have basic knowledge of operating the computers. We want to hear the advice from users about the following perspective:

- Graphic Design
- Difficulty level of this game (We do not want our game to be too difficult to make players lose interests to play it)
- Game Items design
- Overall experience

2 Project Constraints

2.1 Mandated Constraints

2.1.1 Solution Constraints

- The project will be implemented by python since the old version is done by python. It is time saving to do it in python.
- Pygame library will be used in implementation since it is a suitable library to use for game development in python.
- The solution must follow proper software engineering principles.

2.1.2 Implementation Environment of the Current System

The project will be done in Windows and Linux(MacOS), which means the product will be able to run on those environments.

2.1.3 Partner or Collaboration Applications

N/A

2.1.4 Off-the-Shelf Software

N/A

2.1.5 Anticipated Workplace Environment

N/A

2.1.6 Schedule Constraints

The first version of project must be finished before Mar 22, 2022 because the project will be first demonstrated on that day. The final product should be done before April 12, 2022.

2.1.7 Budge Constraints

N/A

2.2 Naming Conventions and Definitions

- Aircraft: A player will control the aircraft to play the game. The player will give inputs to system to manipulate the aircraft and the aircraft will react accordingly.
- Monster: Monsters are the opponents(bad guys) of players. A player has to eliminate all of the monsters in order to win the game.
- Health points: It stands for the health of the aircraft. The aircraft will lose one health point once it gets hit by bullets from monsters.
- Ammo item: If the aircraft shoots an Ammo item, ~~the aircraft's ammo type will be upgraded to a more powerful one.~~ the system will allow the aircraft to shoot one more bullet at the same time.
- Health item: If the aircraft shoots a Health item, the aircraft will gain one more health point.
- Bomb item: If the aircraft shoots a Bomb item, the bomb will explode and do damage to a big range of the area in the monster matrix.

2.3 Relevant Facts and Assumptions

2.3.1 Facts

- The original project has 700 lines of code.
- The original project has soundtracks, but they will be removed because the soundtracks may make people uncomfortable.

2.3.2 Assumptions

- Users should have basic equipments to play the game, including a laptop, a mouse and a keyboard.
- Users should know how to operate a PC.
- Users should understand basic English.

3 Functional Requirements

3.1 The Scope of Work and Product

3.1.1 The Context of the Work

The original Space Invaders has poor implementation documentation and is not a fully functional game platform for the user to interact with to win the game. Not only does Space Invaders fails to deliver a user-friendly game but it also fails to meet the software engineering design principles. Space Invaders is not ideal for a developer to understand the game's ~~file structure~~ architecture ~~and working software~~ as it lacks documentation and modular design in the project. Therefore, our new Space Invaders has the following goals:

- Redesign the game following software engineering principles and process for the benefit of other developers.
- Give the user an interactive and engaging game with an understandable winning criteria and functionality

The following is a Work Context Diagram:

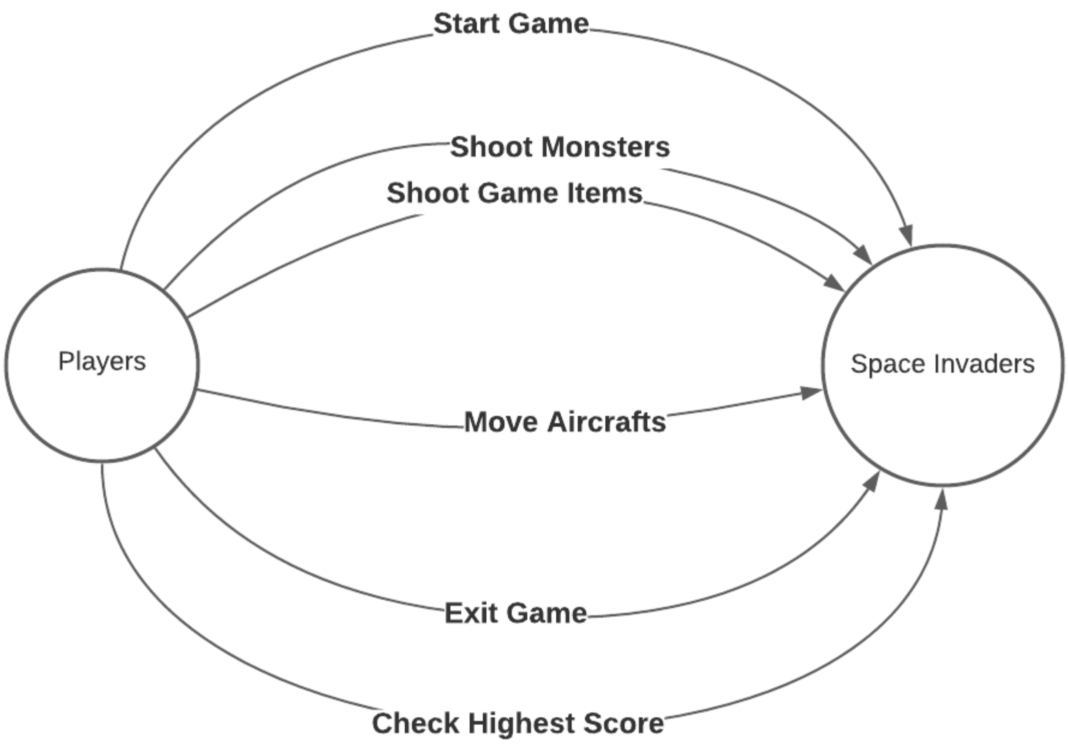

Figure 1: OLD Context Diagram

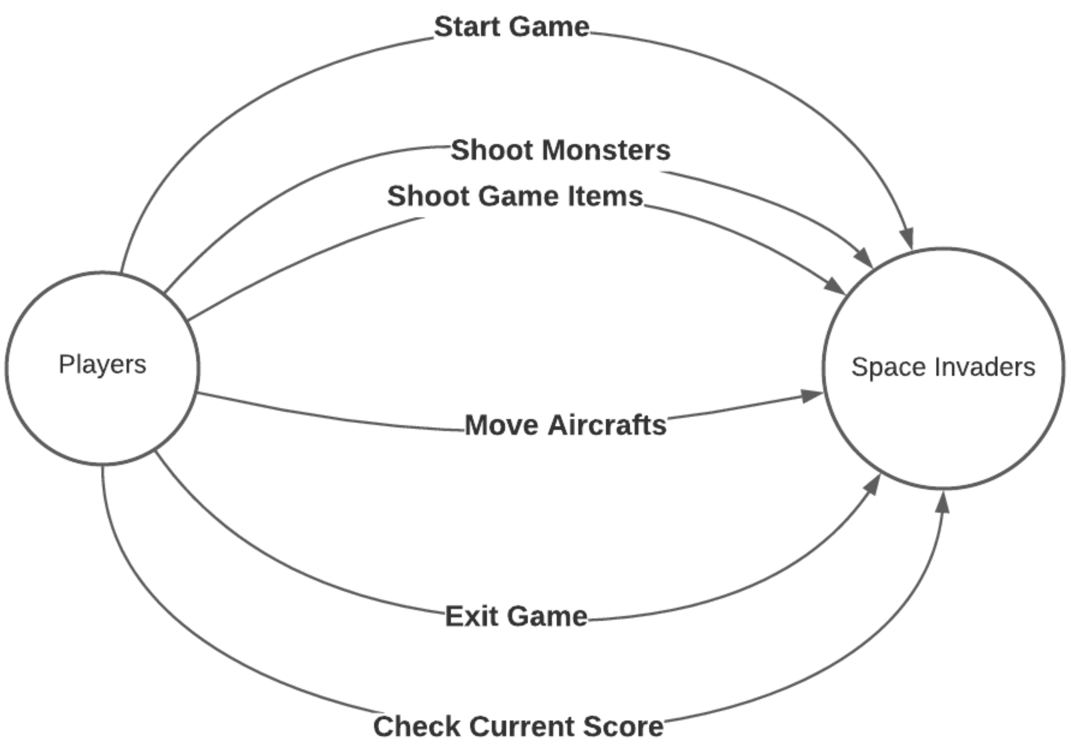

Figure 2: NEW Context Diagram

3.1.2 Work Partitioning

Business Event	**Inputs**	**Outputs**	**Summary**
Start Game	Keyboard (Players press ~~any~~ Enter/Return key to start)	Display instructions and let players to choose playing mode	• Creates game frame with monsters, game items, aircraft and obstacles. • Create five rounds of game • Presents two playing modes for players • Display game instructions
Shoot Monsters	Keyboard Input	Monster disappears in GUI	Based on the player's actions, aircraft will attack monsters which are controlled by the system.
Shoot Game Items	Keyboard Input	Game Items disappear in the monster matrix and display the corresponding changes due to game items	The players can shoot game items inside the monster matrix.
Move Aircraft	Keyboard Input	Aircraft changes position	Different keyboard input will represent aircraft moving left or right.
Exit Game	Keyboard Input	Exit game window	Players can exit game by clicking the cross icon on the up right corner.
Check ~~Highest~~ current game Score	N/A	Display ~~highest point~~ current game score on the screen	Players can check the ~~highest score~~ current game score on the screen.

Table 2: Work Partitioning Table

3.1.3 Individual Project Use Cases

The following is the use case diagram:

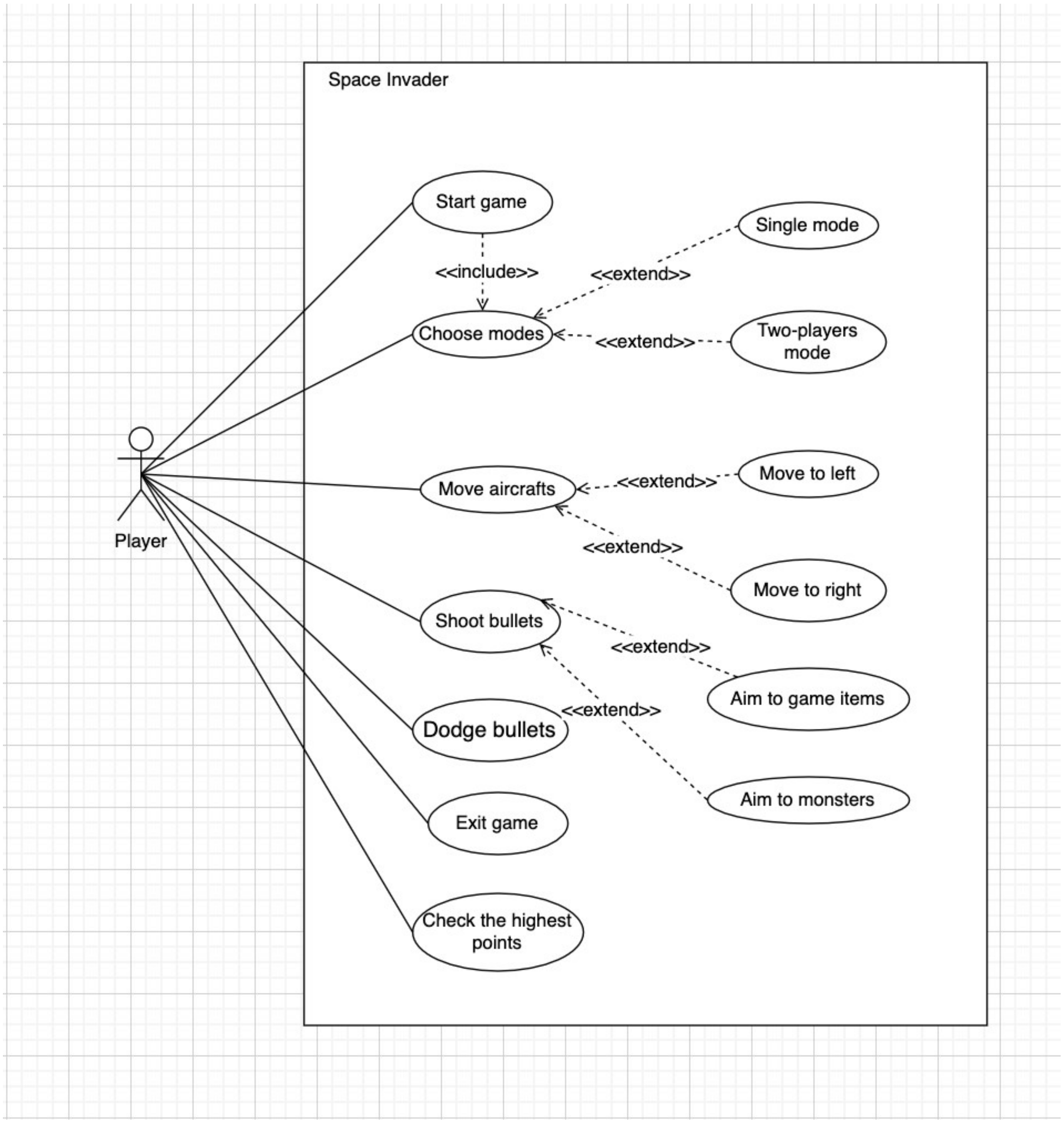

Figure 3: OLD Use Case Diagram

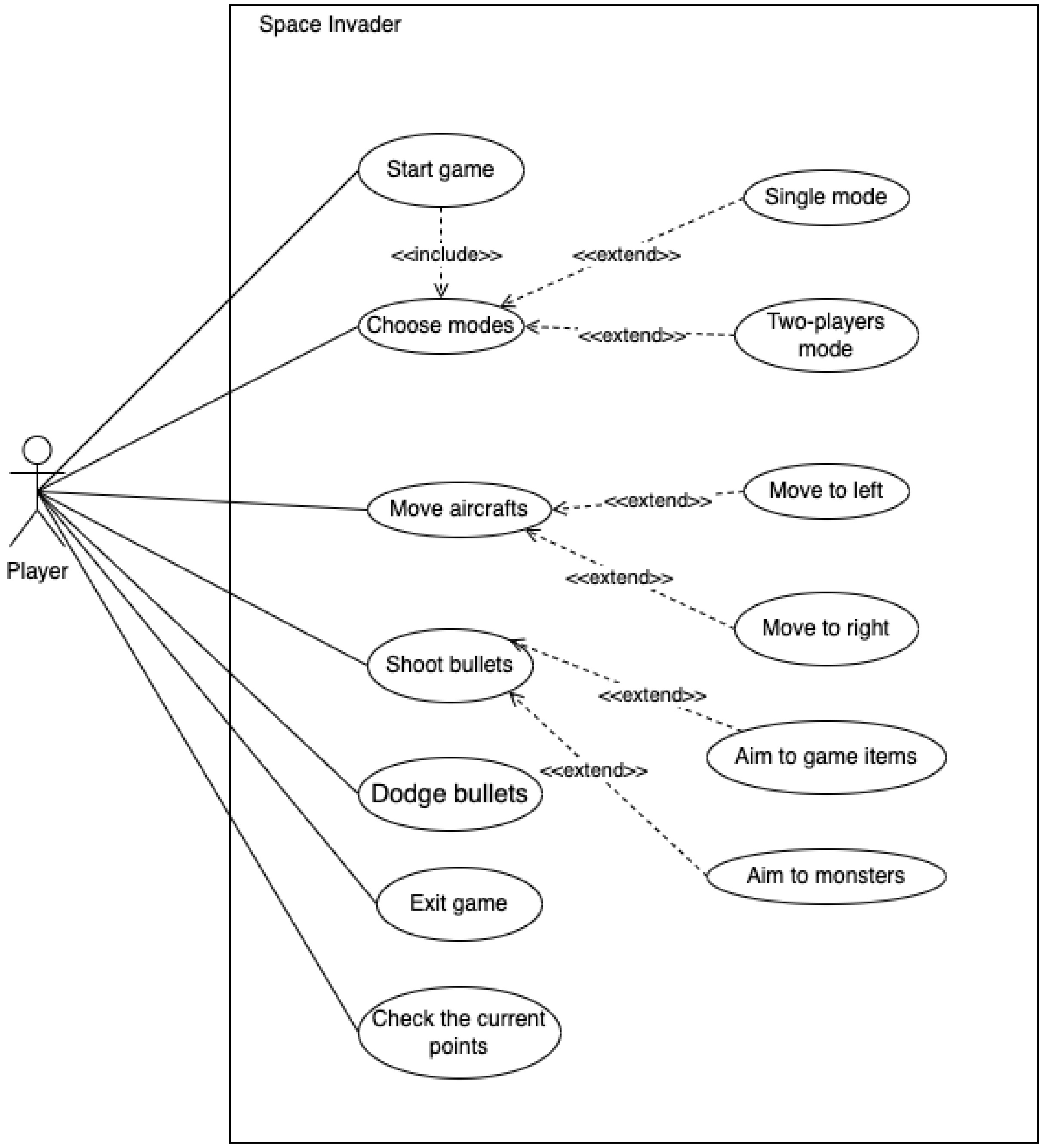

Figure 4: NEW Use Case Diagram

3.2 Functional Requirements

We will state functional requirements from system viewpoint and user viewpoint perspectives:

3.2.1 System Viewpoint

FR1: The system shall prompt players to choose single-player mode or two-players mode.

FR2: The system shall display the information about monsters' types ~~(green, blue and pink)~~ and corresponding points ~~(10, 20 and 30)~~ at the starting page.

FR3: The system shall display instructions about how to play to game after the starting page.

FR4: The system shall display the total scores for one player or two players, and the initial score should be zero.

FR5: The system shall display the ~~total five~~ health points for each player in different modes at the beginning.

FR6: The system shall display ~~a galaxy picture as~~ game background.

FR7: The system shall display monsters as a matrix for each round ~~(5 rows and 10 columns)~~.

FR8: The system shall display game items (bomb, ammo and health) randomly inside the monster matrix. For the details of these items, please refer to the naming convention.

FR9: The system shall display one aircraft in single-player mode and two aircraft in two-player mode.

FR10: The system shall display ~~four~~ obstacles between the aircraft and monsters.

FR11: The system shall display the final score at the termination of game.

FR12: The system shall display 'WIN!' once players pass five rounds.

FR13: The system shall display 'FAIL' in following situations:

- In single-player mode: The only one player loses five health points.
- In two-players mode: Both players lose five health points.

FR14: ~~The system shall display back to starting page after displaying 'WIN!' or 'FAIL' for allowing players to replay.~~

FR15: The system shall increase players' total scores once they hit the monsters.

FR16: The system shall increase players' total health points when they hit the health item and decrease health points when players' aircraft are shot by monsters.

FR17: ~~The game shall contain 5 rounds.~~

FR18: The game shall contain 5 rounds with different patterns of monster matrix. ~~The game shall set five rows of green monsters in round 1.~~

FR19: ~~The game shall set three rows of green monsters and two rows of blue monsters in round 2.~~

FR20: ~~The game shall set five rows of blue monsters in round 3.~~

FR21: ~~The game shall set one row of green monsters, three rows of blue monsters and one row of pink monsters in round 4.~~

FR22: ~~The game shall set five rows of pink monsters in round 5.~~

FR23: The system shall set monsters' movement be east-south-west under a constant velocity.

FR24: The system shall let monsters shoot bullets randomly.

FR25: The system shall let green monsters die after being shot once (from players' aricraft).

FR26: The system shall let blue monsters die after being shot twice (from players' aircraft).

FR27: The system shall let pink monsters die after being shot three times (from players' aircraft).

FR28: The system shall let a monster disappear if it dies.

FR29: The system shall reduce the area of obstacles if they are shot by monsters or aircraft.

FR30: The system shall set game items in a way that they can only be obtained by aircraft's bullets instead of monsters' bullets.

FR31: The system shall limit the amount of game items for each round.

FR32: The system shall provide bombs which can destroy four random monsters inside the matrix ~~the monsters around it (8 in total)~~ once it is shot by aircraft.

FR33: The system shall provide ammo game items to increase the number of aircraft bullets by 1. ~~only for the present round~~.

FR34: The system shall provide heart to increase aricraft's health points by one once it is shot by aircraft.

FR35 The system shall have random occurrence for each game item.

The following are some summaries about Functional requirements from system viewpoint:

- FR17 - FR21 describes how to set up monster matrix for each round.
- FR22 - FR27 describes monsters' behaviors.
- FR30 - FR35 describes game items.

3.2.2 User Viewpoint

FR36: The game shall be able to start by ~~entering any keys.~~ pressing Enter/Return

FR37: The system shall be able to exit any time by clicking cross icon. ~~at up right corner.~~

FR38: The system shall allow the player to move the aircraft left or right by operating keyboards. ~~by clicking ← or → in the single player mode.~~

FR39: ~~The system shall allow player A to move the aircrafe left or right by click ← or → and let player B to move left or right by clicking A or D in the two-players mode.~~

FR40: The system shall let each player shoot the monsters by one bullet as default.

4 Nonfunctional Requirements

4.1 Look and Feel Requirements

4.1.1 Appearance Requirements

NFR1: The product shall have a good graphic quality.
Fit Criterion: FPS of the product shall be over 30 fps and the graphics should be over 8 bit.

NFR2: The product shall display all the ~~elements in game~~ game elements clearly enough to recognize.
Fit Criterion: 8 out of 10 random persons can recognize all the elements directly.

4.1.2 Style Requirements

NFR3: The game display should follow the style of minimalism.
Fit Criterion: 8 out of 10 random persons should think that the game display follows the style of minimalism.

NFR4: The mood of the graphics should be intense.
Fit Criterion: 8 out of 10 random persons should think the mood of graphics is intense.

4.2 Usability and Humanity Requirements

4.2.1 Ease of Use Requirements

NFR5: The product shall be easy for a 10-year-old child to operate.
Fit Criterion:Over 8 out 10 10-year old children assess this game as easy.

NFR6 : The product shall be used by people with no training.
Fit Criterion:Over 8 out 10 people with no training know how to play this game after 5 minutes of learning.

4.2.2 Personalization and Internationalization Requirements

N/A

4.2.3 Learning Requirements

NFR7: The product shall allow users to understand all the rules of the game quickly.
Fit Criterion: All sample users can learn all the rules of the game in 10 minutes.

4.2.4 Understandability and Politeness Requirements

N/A

4.2.5 Accessibility Requirements

N/A

4.3 Performance Requirements

4.3.1 Speed and Latency Requirements

NFR8: In selection menu interface, The system shall respond user inputs quickly.
Fit Criterion: In selection menu interface, the system shall respond user inputs within 1 second.

4.3.2 Safety-Critical Requirements

N/A

4.3.3 Precision of Accuracy Requirements

NFR9: In gaming interface, the system shall respond to user operating inputs quickly.
Fit Criterion: In gaming interface, the system shall respond user operating inputs within 1 second.

4.3.4 Reliability and Availability Requirements

NFR10: The product shall be available for long-term use.
Fit Criterion: The product shall be available for 24 hours per day, 365 days per year.

4.3.5 Robustness and Fault-Tolerance Requirements

N/A

4.3.6 Capacity Requirements

NFR11: The product shall run steadily when in two-players mode.
Fit Criterion: In two-players mode, the game runs normally for at least 2 hours.

4.3.7 Scalability or Extensibility Requirements

N/A

4.3.8 Longevity Requirements

N/A

4.4 Oerational and Environment Requirements

4.4.1 Expected Physical Environment

NFR12: The product shall run on regular PCs.
Fit Criterion: The game should run on different operating systems correctly 9 times out of 10.

4.4.2 Requirements for Interfacing with Adjacent Systems

NFR13: The product shall run without errors on the most popular operating systems.
Fit Criterion: The product can run normally on Windows, Linux and MacOS.

4.4.3 Productization Requirements

NFR14: ~~The product shall be distributed as a EXE file.~~
~~Fit Criterion: N/A~~

NFR15: The product shall be installed by an untrained user without complex instructions.
Fit Criterion: 8 out of 10 untrained users can install the game without any trouble.

4.4.4 Release Requirements

N/A

4.5 Maintainability and Support Requirements

4.5.1 Maintenance Requirements

NFR16: For any new features to be added, the implementation should be finished as soon as possible.
Fit Criterion: The MIS should be ready within one week since adding new features decisions are made. After that, coding and testing should be completed within one week. Given the size of our project, this timeline is feasible.

NFR17: All the code should be well documented.
Fit Criterion: The code should be documented using doxygen.

4.5.2 Supportability Requirements

NFR18: There must be a way that developers can communicate with players so that developers can receive the advice from players and offer support to players.
Fit Criterion: The repo of this game should remain public so that developers and players can communicate via GitLab.

4.5.3 Adaptability

NFR19: The product is expected to run under different operating systems.
Fit Criterion: The game should be able to be executed in Windows, Linux and MacOS.

4.6 Security Requirements

4.6.1 Access Requirements

NFR20: In two-players mode, player A can not have control of player B's aircraft and vice versa.
Fit Criterion :The control modules of two aircraft should be separated in module design.

4.6.2 Integrity Requirements

NRF21: ~~This game can record the highest score and players can not access to this file.~~
~~Fit Criterion: Highest score should be private so that it is inaccessible from outside.~~

4.6.3 Privacy Requirements

N/A

4.6.4 Audit Requirements

N/A

4.6.5 Immunity Requirements

N/A

4.7 Cultural and Political Requirements

4.7.1 Cultural Requirements

NFR22: The game shall not contain any offensive contents to religious and ethnic groups.
Fit Criterion: This game will not contain many words, therefore, developers should be carefully about pictures used in GUI.

4.7.2 Political Requirements

N/A

4.8 Legal Requirements

4.8.1 Compliance Requirements

NFR23: The system shall not break and Canadian Laws.
Fit Criterion: Game should be approved by some law institution.

4.8.2 Standards Requirements

N/A

4.9 Health and Safety Requirements

NFR24: ~~The game developers shall notify players to avoid game addiction.~~
~~Fit Criterion: Game shall display a message after the starting page to tell players to avoid game addiction.~~

5 Project Issues

5.1 Open Issues

Issue1. The original project has all the code written in just one `.py` file. The issue is to redesign the modules to achieve software engineering principles.

Issue2. Our team hopes that executable file can be executed on Windows, Linux and MacOS. We are still looking for ways to create executable files for MacOS and Linux.

Issue3. Since this game includes GUI, our team is still thinking about appreciate ways to test GUI.

5.2 Off-the-Shelf Solutions

5.2.1 Ready-Made-Product

Since our new Space Invaders game has some new features and will be developed using proper software principles, we can not find any existing solutions.

5.2.2 Reusable Components

`Pygame` library can be used for implementing games in python. For testing, we can use `Pytest` and `unittest` libraries.

5.2.3 Products That Can Be Copied

The original project is a similar product. This is the Link.

5.3 New Problems

5.3.1 Effects on the Current Environment

The new product will not affect the current implementation environment. However, our product should not do the following things:

- Bring Malware software to players' computers.
- Use too much computer resources when running.

5.3.2 Effects on the Installed Systems

If the user wants to run the source code to play the game, then required libraries should be installed first.

5.3.3 Potential User Problems

N/A

5.3.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

N/A

5.3.5 Follow-Up Problems

N/A

5.4 Tasks

5.4.1 Project Planning

The project schedule is created as a Gantt Chart which can be found using this Link

5.4.2 Planning of the Development Phases

Phase1: Listing all the functional and nonfunctional requirements. Phase 1 has already finished in this document.

Phase2: After having all the requirements, phase 2 will be designing modules and module interfaces.

Phase3: Coding according to MG and MIS developed in phase 2.

Phase4: Testing the final product.

5.5 Migration to the New Product

Our project is independent from the original project, there are no requirements for Migration to the New Product.

5.6 Risks

The following are some risks with this project:

R1: Implementation Difficulties:
This game will be implemented with python. The good news is that every team member has experience with this language. However, we need to use pygame module for implementation and only one team member has experience with this module. Therefore, other two team members should spend time to learn this module and familiarize with the API.

R2: Testing:
Game testing is usually harder compared with testing of other projects. The main difficulty here is to choose an appropriate testing method so that it can cover all the scenarios.

R3: Portability:
We believe that source code can be executed in different operating systems. However, the problem is to make sure that executable files can be executed in different operating systems.

R4: Design and Requirements Difficulties:

- Design Difficulty: The main difficulty in design process is designing appropriate modules. We need to make sure that each module is low coupling and high cohesion.
- Requirements Difficulty: The main difficulty for requirements engineering is to solve conflicts and make sure the final requirements document does not have any conflicts.

5.7 Costs

This project is based on an open project and all the libraries are free to use, so this project does not involve monetary cost. However, there are time costs, minimum 4 hours of weekly meeting are needed.

5.8 User Documentation and Training

5.8.1 Documentation

After the starting the page, there will be a page to show instructions of how to play the game.

5.8.2 Training

This game is quit easy to learning. No training is required.

5.9 Waiting Room

The following are some potential future features:

- The system shall allow two players to connect and play two-player mode via internet.
- Add more interesting game items.
- Introduce game currency. Maybe players can buy aircraft skins using the currency.

5.10 Ideas for Solutions

The following are some potential modules for the game implementation, this may change during the design process:

- Single Player Mode
- Two Players Mode
- Monster Matrix
- Aircraft
- Game items
- Obstacles
- Record Highest Score
- bullets
- Health Points

6 Reference

Robertson, James, and Suzanne Robertson. "Volere." Requirements Specification Templates (2000).