Risk Assessment and Mitigation

Team: No

Team Members:

- Jack Burman
- Sam Churchill
- C Lloyd
- Sam Ralph
- Rebecca Wardle
- Jiacheng Wu

Previous Group

Team: Vega

Team Members:

- Chloe Wardle
- George Grasham
- Lewis McKenzie
- Matthew Rogan
- Haopeng Zhu
- Benjamin White

Requirements

a) Write a succinct introduction explaining how requirements were elicited and negotiated, and why they are presented as they are. Your submission should evidence research into requirements specification and presentation (4 marks, ≤ 1 page).

b) Give a systematic and appropriately-formatted statement of requirements, including, for each requirement, a note of any relevant environmental assumptions, associated risks, or alternatives (16 marks, ≤ 3 pages).

The decision to adopt an agile approach towards development influenced the way that the requirements elicitation began. The first two weeks were spent discussing individual ideas about the game and deciding the allocation of roles. The decision to defer the initial customer meeting was to ensure that the group was to promote creativity among the team, collectively generating sufficient ideas for the implementation of the game.

During our meeting with the stakeholder, we held a closed interview, asking a set of predetermined questions which provided us with an overview of the specific user requirements allowing us to adapt our design to ensure the client's needs were met. The responses to our questions were recorded verbatim and made available to all members of the team via a shared document stored on Google Drive. Thereby, the absence of any team members would still be fully aware of the requirements prior to any further design tasks.

Following the second customer-meeting, requirements were systematically logged and grouped based upon similar dependencies and rated against their priority during implementation. We understood that due to the timescale of the project, we needed to adopt an agile method as requirements could change quickly and we needed to be able to facilitate those changes without getting lost in the organisation of our requirements document.

We mainly followed a predictive life cycle, taking the initial time to construct a comprehensive requirements document including identifiable constraints such as time, budget, varying levels of technical experience. Furthermore, due to the nature of the project we did not have the time to produce prototypes, to return to the customer, readjust the requirements and perhaps do this multiple times as would be expected in a typical Agile process. Therefore, we wanted to utilise the time with our customer to ensure that provided us with a broad view of what was expected, including any additional unprecedented requirements.

Initially, once the requirements document had been compiled – every individual requirement was logged in tabular format, each assigned a unique ID prefaced with 'FR' denoting a functional requirement and similarly 'NFR' for a non-functional requirement (Appendix X). However, both (Keith, 2010) and (Sommerville, 2016) both placed emphasis on the benefits of incorporating stories into the requirements document. The overall presentation of the software requirements specification was based upon the IEEE standard 803-1993. We decided to group functional requirements by corresponding system features to make it easy to understand for both the client and among members of the team.

IEEE 803-1998

- Constraints
- User classes and characteristics
- Design and implementation constraints
- User documentation

System Features

1. Character Control: UR_CONTROL

1.1. Description and Priority

The actions and movements of Auber are controlled by the user.

1.2. Stimulus/ Response Sequences

The user will control Auber using the directional keys on a standard keyboard. A mouse is required to select a room to teleport to.

1.3. Functional Requirements

FR_PLAYSTYLE: The game must be real-time (not turn based) and single player.
FR_MOVEMENT: The movements and actions of Auber can be controlled using a keyboard and mouse.
FR_CONTROL_SELECT: The user can select whether to control the game using a mouse or a keyboard.

2. Non-Playable Characters: UR_NPC

2.1. Description and Priority

There must be eight infiltrators that must be able to navigate the map independently from user control. In

addition to this, there should be a significant number of decorative NPCs to conceal the infiltrator's identity. The infiltrators must have three distinct abilities which impact the main player, either by causing damage to Auber or hindering gameplay.

2.2. Stimulus/ Response Sequences

As Auber traverses the spaceship, the NPC have been programmed to navigate towards a crucial system to sabotage. Auber will need to walk up to a suspect, press the 'A' key which will initiate the arrest mechanism. The NPC will stop its traversal and begin to follow Auber who will need to travel to the nearest teleport where the NPC will be transferred to the jail.

2.3. Functional Requirements

FR_NPC_AI: A* Pathfinding is implemented to ensure that NPCs follow a logical path to the nearest system to infiltrate.

3. Arresting Mechanism: UR_ARREST

3.1. Description and Priority

The main character will be required to locate and arrest infiltrators onboard the spaceship. There is no time-limit, but the main character will be required to arrest eight infiltrators before 15 crucial systems onboard the ship are sabotaged.

3.2. Stimulus/ Response Sequences

The user will control Auber using the directional keys on a standard keyboard. A mouse is also required to select a room to teleport to.

3.3. Functional Requirements

FR_ARRESTMECH: Approaching an infiltrator and pressing the action key (A) will cause them to follow the lead character. The main character will need to take the infiltrator to the teleportation pad, select jail from the drop-down menu to send the infiltrator to the jail.

FR_PRISON: Once an infiltrator is sent to prison, they cannot return to the game.

4. Systems: UR_SYSTEMS

4.1. Description and Priority

The infiltrators mission is to sabotage fifteen systems located throughout the spaceship before all eight infiltrators have been arrested.

4.2. Stimulus/ Response Sequences

To begin sabotaging a system, an infiltrator must be near a system that has not already been sabotaged. After a short waiting time, the system will be sabotaged and the list of the sabotaged systems located on the HUD will be updated.

4.3. Functional Requirements

FR_SYSTEM: There will be 20 systems located throughout the map and the infiltrators are required to sabotage fifteen. To begin the sabotage, the infiltrators must be within close proximity of a system which will begin the sabotage sequence. Once a system has been sabotaged, it cannot be repaired.

FR_COOLDOWN: There will be a cooldown imposed on the infiltrators to make the game easier to win - the length of the cooldown will be determined by the average game length ideally lasting between 5-10 minutes.

5. Game Modes: UR_GAME_MODES

5.1. Description and Priority

In addition to the main gameplay element, the game should also feature a demo mode and a tutorial mode. There will also be a difficulty option.

5.2. Stimulus/ Response Sequences

The landing page of the game will include options to access the demo mode, where the game is able to play itself without any user input. There should also be a tutorial mode detailing the controls and aim of the game. In this page the user will be able to change the difficulty level of the game.

5.3. Functional Requirements

FR_DEMO: When selecting the demo mode, a pre-recorded video will play on a loop for between 30-40 seconds and will continue to loop until returning to the main-screen.

FR_TUTORIAL: When selecting the tutorial mode, a series of screens will detail the individual control elements and gameplay elements such as the arresting mechanism.

FR_MENU_SELECT: The menu select will act as the main homepage for the game, where the user can select options for gameplay, demo or tutorial modes.

FR_DIFFICULTY: In the options menu the user will have the choice of three difficulty settings for the game (easy, normal, hard) that affect the health of the systems. The default will be normal.

6. Fast-Travel Mechanism: UR_TELEPORT

6.1. Description and Priority

The main character will be able to quickly travel to different areas of the spaceship by walking onto a teleport pad and selecting a destination from a drop-down menu.

6.2. Stimulus/ Response Sequences

User will use the directional keys to move the main character. Once the character is on the teleportation pad, a drop-down menu will expand, and the user can select their destination. Once selected, the lead character will instantly teleport to that destination.

6.3. Functional Requirements

FR_TELEPORT: The lead character is able to walk onto a teleportation pad and select a location from a drop-down menu instantly moving the lead character to that room.

FR_ROOM_SELECT:

FR_HEALING: One of the rooms aboard the spaceship is the infirmary, which contains the healing pod where Auber can replenish their health, This room can be accessed either by walking, or by using a teleportation pad.

7. Head-Up Display: UR_HUD

7.1. Description and Priority

The game will feature a HUD detailing the main character's current health, mini-map and breakdown of the number of systems that have currently been sabotaged.

7.2. Stimulus/ Response Sequences

Aubers' current health status will be permanently displayed on the screen and will be depleted if attacked by one of the infiltrators and increased if he reaches the healing pod. The HUD will also include a mini-map showing the main characters' location. Finally, as systems are sabotaged throughout the spaceship the number and locations of sabotaged systems will be represented on-screen.

7.3. Functional Requirements

FR_HP: The lead character starts off with a full health-bar, which depletes if attacked by an infiltrator. The main character can replenish their health by standing on the healing pad.

FR_SABOTAGED: As systems are sabotaged, system names turn red and a counter is shown in the top left-hand corner.

FR_ARREST_COUNT: Once an infiltrator has been teleported to the jail, a counter shown within the HUD will update to reflect the number of infiltrators that have been arrested.

FR_MINIMAP: A minimap will also be included within the HUD, where the lead character can see his current location and < Confirm what this was meant to include >

8. Winning Condition: UR_END

8.1. Description and Priority

The game will end once 15 systems have been sabotaged, or once all eight infiltrators have been arrested.

8.2. Stimulus/ Response Sequences

Once either condition has been satisfied, the game will end and a final screen will show to the user signalling a win or loss.

8.3. Functional Requirements

FR_END: The end of the game will be decided once all fifteen systems have been sabotaged or Auber arrests all eight infiltrators.

9. Auber Power Ups: UR_POWERUPS

9.1. Description and Priority

Auber will have the opportunity to obtain five special power ups on the journey.

9.2. Stimulus/ Response Sequences

9.3. Functional Requirements

FR_POWERS: There will be five different power ups on the ship that Auber can choose to pick up.

10. Save Game State: UR_SAVE

10.1. Description and Priority

The user will be able to save their game state at any point and resume it later.

10.2. Stimulus/ Response Sequences

10.3. Functional Requirements

FR_SAVE:

External Interface Requirements

User Interface

Upon opening the game, the user will be taken to the main menu where they can select to begin a new game or initiate the

demo mode. The user will be required to click on the corresponding option to invoke gameplay. Once the game has begun, the camera will follow the lead character's movement across the map. As the game progresses, various attributes will be updated within the HUD such as the number of arrested infiltrators and sabotaged systems will be permanently displayed alongside the lead characters current health. The two screenshots below show the main menu and the user interface during gameplay including the HUD.

Hardware Interfaces

The game is expected to run as a desktop application rather than web or mobile based. The group has implemented the game using the LibGDX framework which by nature supports cross-compatibility among differing devices/operating systems. The suggested minimum system requirements is 4GB of memory. The game requires the use of a standard keyboard and mouse to control the actions and movements of the lead character.

Software Interfaces

LibGDX

Other Non-functional Requirements

NFR_GAMETIME

NFR_BRANDING

NFR_DOCUMENTATION