Requirements Changes Introduction

Note that where requirement numbers are referenced throughout this introduction, they refer to the requirement numbers from our previous (Assessment 1) requirements document, as after changes to the requirements document were carried out, the requirements were renumbered.

Design and implementation decisions that were too specific to be included in the system requirements have been removed. The requirements will therefore now represent a clearer picture of *what* the system should do and not *how* it should be done, which is covered in other areas of the documentation. This should be an appealing quality when other groups are choosing our game, as it will be easier to identify what the system still needs and what has already been achieved, with respect to the requirements.

- Requirement UR2.2 was removed as it contained specific implementation details – which keys to use for control - however were not suitable for a requirements document. Requirements FR1, FR1.1 and FR6 were removed for the same reason.
- FR12-FR12.3 were also removed, as they concerned sound usage, which is too design-focused for a requirements document. However, the intention to include sound was still included as a non-functional requirement.

The Non-Functional requirements section has been expanded. This section was previously fairly limited with only three requirement points; therefore, it did not present a thorough idea of our intentions for how the final game would look and play. In order to result in a final product which matches our intentions for the style and feel of the game, additional requirements about the GUI, gameplay and story were added.

- Requirements NF4 – NF6 were added.

A problem with our previous requirements was that not enough justification was provided. To better help readers of the documentation understand our thought process and reasons behind our requirements elicitation, justification has been added to major requirements and requirements where a decision was required/alternatives may have been viable. Reference to the brief (‘Scenario’) given to us by the client is made in these justifications to show that our requirements are consistent with the customer needs.

- Justification referring to the customer brief was added to FR7, FR8 and FR9.

Additional requirements and one-off changes – some further decisions were made during the implementation (weapon characteristics, guard enemies taking damage, additional enemies etc). These are therefore changed in the requirements so that confusion and incorrect requirements compared to our implementation are avoided.

- FR10.1 and FR10.2 were altered as enemies were altered in the implementation.
- Added FR11.2 to reflect the fact that points for killing enemies was added in the implementation.

Finally, the document was further organised – the functional requirements layout was slightly unclear at times and so to rectify this, it was organised into separate ‘Menus’ and ‘Gameplay’ sections. Additionally, some small grammatical errors were rectified, but the meanings of the requirements in these cases were left unchanged.

Updated Requirements

Changes to this document are shown in blue text.

2.4 User requirements

UR1 Menus & Navigation
UR1.1 (M) The user will navigate the menus using mouse inputs. In this section, the assumption was made that menus were required to navigate the game, despite not specifically being asked for in the scenario given. We decided to implement menus to make the game easy and user-friendly to navigate.

UR1.1.1 (M) The user will be able to exit the main menu.

UR1.1.2 (M) The user will be able to return to the main menu from secondary menus.

UR1.2 (M) The user will be able to start a new game.
UR1.3 (B) The user will be able to save the progress of their current game.

UR1.3.1 (B) The user will be able to load a previously saved game.

UR1.4 (M) The user will be able to see their current score and objective title.
UR1.5 (M) The user will be able to view the game instructions.
UR1.6 (M) The user will be able to exit the game during gameplay.
UR1.7 (M) The user will be able to continue the game after failing an objective.

- The user will be able to quit to the main menu after failing an objective.
- User will be able to pause /resume in-game.

UR1.8 (O) User should be able to turn music on/off.
UR1.9 (O) User will be able to view their achievements.

UR2 Gameplay
UR2.1 (M) The user will complete two kinds of objectives:

- Overworld objectives.
  - The user will reach a specified location on the University of York campus map.
  - The user will avoid obstacles by moving the duck.
- Location-specific objectives.
  - The user will reach the end (goal) of a map room.
  - The user will avoid or kill enemies.
  - The user will use the duck's skills to complete the map.

UR2.1.2 (M) The user will avoid obstacles and kill enemies to help complete objectives.

UR2.1.3 (M) The user will earn points whenever they complete an objective or kill an enemy.
UR2.2 (O) The user will be able to gain achievements.

2.5 Systems Requirements
2.5.1 Functional Requirements

Menus

FR1 (UR1) The system will display the main menu when the user starts the game or when exiting the game:

FR1.1 (UR1.2) Start New Game:

- The user will be prompted to give a name to their character. This will help distinguish different save files.
- Starting a new game will start the game from the predefined starting point.
- When the game is saved it will be stored locally.

FR1.2 (UR1.3) Load Saved Game:

- Prompt user to choose from three saved file slot.

        - The slots will show: name, health, stamina, points, location and save time.
    - Game will be loaded from the save file when selected and start from last saved state so player can quickly get into the game.

FR1.3 (UR1.9) View Achievements- There will be a list of achievements that the player can get by completing certain objectives in game. They will be able to see how many they have to still get. This will encourage the player to keep playing.
FR1.4 (UR1.8) Sound - turn off/on switch
FR1.5 (UR1.5) Instructions - display the instructions on user controls and aims for the game.
FR1.6 (UR1.1.1) Exit - closes the window and stops the program.

FR2 (UR1.7) The system will display the game over menu to the user when they lose a round of the game by losing all their health. They will be allowed to select:
FR2.1 Continue - restart a round from the beginning
FR2.2 Quit game - return to main menu

FR3 (UR1) The system will display the in-game options menu when a button in the top right corner of the screen is pressed at any time during the game which includes:
FR3.1 'Current Score: ', followed by the user's current score.
FR3.2: 'Quest: ', followed by the title of the current quest to be completed by the user.
FR3.3 Save game button - will display the in-game save menu to the user so they can save at any point. The game overall will not be short so having a save option allows the player to take a break.
FR3.4 View Achievements button.
FR3.5 Sound switch.
FR3.6 Instructions button - To help remind the player of controls/gameplay.
FR3.7 Exit game button - take the user to the main menu.

FR3.2 The system will pause the game when the in-game menu is displayed so the game state does not update during this time but will resume when the menu is closed.

FR4 (UR1.3) A save file will contain the following data:
- Player name
- Time/date of save
- Points
- Players health/stamina
- Location - area and obstacles states
- Current objective

Gameplay

FR5.1 The duck will have a health bar.
- Being collided into by enemies or enemy projectiles will reduce its health.
- Collecting health drops that are in the rounds will increase health.
- When it reaches 0 health, the user must restart the quest.

FR5.2 The duck will have a stamina bar.
- Will deplete when using powers.
- Recover over time.
    - This will create more of a challenge and strategy for what/when to use powers.

FR5.3 The duck's innate abilities will be:
- Peck attack.
- Waddling.
- Sprinting.
- Flying.

FR5.4 The duck will acquire these powers throughout the game:
- Quack cannon.
    - A projectile attack thrown from the duck's mouth.

        - Player begins with 10 ammo; 10 ammo can be held in each clip.
        - Will alert enemies to the duck when hit.
- Quack hack.
    - Emits a high frequency that hacks into a B.R.A.I.N so the duck can manipulate humans into opening doors.
    - To use the ability the duck will need to be behind the human.

FR6 (UR2.1) The quests will have two different forms of objectives, as is required in the customer's brief:

FR6.1 Main objective - save the humans.

- Navigate through pass enemies/traps/areas to reach the mind control emitter.
- Destroy the mind control emitter that controls the humans that is in each of the different locations.

FR6.2 Secondary objectives - will earn the player more points for a higher score at the end.

- The user will utilise timing, different movement speeds and walls/obstacles to avoid the enemies' lines of sight so they aren't detected.
- Defeat certain enemies.
- Achieve the hidden achievements.
- Complete the quest within the time limit.

FR7 (UR2.1) The game will display an overworld between locations. This overworld has two objectives. The use of varying objective between the overworld and map locations is to present a varied and engaging game to the user:

FR7.1 Traversing the overworld. The player will have to avoid randomly generated obstacles which are:

- Airplanes
- Flock of geese
- Thunderstorms

FR7.2 Complete objective on map within a university location.

FR8 (UR2.1) The user will only be able to enter one of the 8 university locations on the overworld map if it is a quest location by flying the duck there. The brief requires all locations to be within the University of York.

FR8.1 The locations are: Computer Science Building, Library, Sports Centre, Lakes, Ron Cooke Hub, Glasshouse, Heslington Hall and Halifax College.

FR8.2 Entering the location changes the screen to the location map.

FR8.3 The quest objectives will be displayed as text at the top of the screen when triggered. This is so the player knows what objectives need completing in order to complete the level.

FR8.4 Upon completing the objective, the user will finish the quest. Quest complete text will display and the player will exit back into the overworld where a new quest's location will appear, unless all quests have been completed.

FR9 (UR2.1) Enemy Obstacles - The assumption made in this section is that enemies are a suitable interpretation of the required obstacles from the scenario given. Enemies will have a cone of vision. If the player is spotted different enemies will act accordingly.

FR9.1 Normal geese:

- Chase after the player.
- They do one point of damage to the duck when they collide with it.
- They take three attacks from the duck to be destroyed.

FR9.2 Guard turrets:

- Are randomly generated and stay in a fixed place.
- Cannot be killed.
- Shoot constantly in a circle to provide projectile obstacles for the player to avoid

FR9.3 Humans:

- Cannot be killed by the player as one of the objectives is to save the humans.
- Quack hack can be used by the player on the humans.

FR10 (UR2.1.3) Points system:

FR10.1 The player's points will be displayed on screen during the game so they can see how well they are doing.

FR10.2 Upon killing an enemy the player earns points

- The player can earn extra points by killing multiple enemies in quick succession

FR10.3 Upon successfully completing a quest the player will earn points:

- Completing quests will result in the player earning points.
- Points will determine player's final score.

FR11 (UR1.7) Game Endings.

FR11.1 Loss:

- The player runs out of time - a cutscene is shown and the player's score is displayed on-screen.
- The player's health is zero- a cutscene is shown and their score is displayed along with the options to restart the quest or go to the main menu.

FR11.2 Win:

- The player completes all main objectives - an ending cutscene plays and the player's score is displayed.

2.5.2 Non-Functional Requirements

NFR1 Player needs to understand the game mechanics in 5 minutes. This makes sure the player can get immersed into the game quickly and not be too intimidated to play the game which will be good for open days when time is limited.

NFR2 Game will run on Java-supported university platforms on both Windows and Linux.

NFR3 Saving/Loading the game will take less than 10 seconds - so player can get into game quickly.

NFR4 The game should be fun to play. This is referred to in the scenario where it specifies that a game should be enjoyable.

NFR4.1 To achieve this, the game should:

- Be fast-paced and exciting – Our game will contain fast-moving enemies.
- Provide user rewards for completing stages - Which is done through the achievements system..
- Provide a range of enemies and objectives - There are multiple enemies, each with different abilities and the player has both primary and secondary objectives.
- Have an attractive GUI and suitable music/sound effects.

NF5 The game should challenge the player

NFR5.1 To achieve this, the game should:

- Increase in difficulty at later levels.
- Provide suitable amounts of ammo/health.

NF6 The game should present an interesting, engaging story to the user.

NFR 6.1 The game should present the story through cutscenes which play throughout the game