Requirements

Cohort 1 Group 7

Group Members:

Kyle Clifton (wnp512)

Bailey Eyers (rkh552)

Matt Hollyhead (fpk506)

William Martin (zjc524)

Max Pither (rkm538)

Enkhtuguldur Sarantsetseg (wbj512)

Ronny Watt (dvn513)

Requirement Elicitation

Single Statement of Need (SSON):

"A game where the user escapes from a university-themed maze with a time limit while avoiding obstacles and acquiring power-up items"

Initially we created a Single Statement of Need and collected and inferred requirements from the initial brief. While this meant our initial requirements were rough around the edges, we planned to interview the customer to get further details on the game. From there, we planned to change any requirements that needed changing, as well as adding some new ones if we deemed it necessary.

After the interview with the customer, in which we asked them a set of pre-planned questions to determine what they wanted the game to be like, we reviewed our initial requirements and found that we had covered most of the points that the customer wanted for the game. We added a few minor changes as well as some extra requirements to ensure the game was as close to the customer's description as possible.

To organise our requirements in a way that would make them easy for anyone unfamiliar with our game to understand, we decide to divide the requirements into 4 categories:

- User requirements - Features that the user would encounter or interact with during the game
- Functional requirements - What the system would need to fulfil the User requirements
- Non-functional requirements - Standards to measure the system's performance against such as reliability, timing and usability when fulfilling the Functional and requirements.
- Constraint requirements - Any constraints that we would need to follow during the development of the game

Each category of requirement would then get a table of requirements made up of 3 columns. The first column of every row was the requirement ID. Our requirement ID started with the table it was in, to indicate what category it was. Then we added a description of what its requirement was to the ID. The second column of the requirements table was a description of each requirement that the game would need to fulfil the brief. The third columns of the tables were different for some categories. The third column of the user and constraint requirements was the priority of each task, with the option of Shall, Should or May for each row while the third column of the Function and Non-functional tables was the User requirement that would be fulfilled if we fulfilled the requirement.Use case

To verify that our requirements were valid, we created a text-based use case of the user playing the game and assessed if the requirements would be satisfied in it. The use case was:

- Actor: User
- Precondition: The user has already loaded the game
- Trigger: The user starts the game
- Main success scenario:
    - The user moves around the maze
    - The user finds the way out the maze
    - The user escapes within the time limit
- Secondary scenarios:
    1) The user encounters positive events that help them
    2) The user encounters negative events that hinder them
    3) The user encounters hidden events that change the course of the game
    4) The user does not escape the maze within the time limit and the game automatically ends
- Success Postcondition The user receives a higher score at the end of the game
- Minimal Postcondition: The game ends and the user receive their score

User Requirements Table

ID	Description	Priority
UR_ESCAPE_MAZE	The player shall be able to escape from a university themed maze.	Shall
UR_POSITIVE_ EVENTS	The player shall encounter at least 3 visible events that provide beneficial effects.	Shall
UR_NEGATIVE_ EVENTS	The player shall encounter at least 5 visible events that hinder their progress.	Shall
UR_HIDDEN_ EVENTS	The player shall encounter at least 3 hidden events that alter the game in beneficial or detrimental ways.	Shall
UR_UI	The user interface shall be friendly to new users	Shall
UR_TIME_LIMIT	The player shall be able to complete the maze within a time limit of 5 minutes.	Shall
UR_PAUSE	The player shall be able to pause the game during play.	Shall
UR_SCORE	The player shall receive a score after completing the game.	Shall
UR_BOUNDARIES	There should not be areas of the map where it is unclear if they are explorable.	Shall
UR_THEME	The maze shall include clearly recognisable university-like features.	Shall
UR_DO_NOT_SAVE	The user shall be unable to access their progress through the game after they have completed it	Shall
UR_DIFFICULTY	The game should allow for varying difficulties, including an easy mode that always permits pausing.	Should
UR_HIDDEN_ EVENT_HINTS	The player could find hints that tell them the location of the hidden events, or clues as to what they are	May
UR_DEAN	There may be a feature based around a dean, or university staff member that shall inconvenience the player	May
UR_PROFESSOR	There shall be a feature where a professor will block a door until the player hands in homework	Shall
UR_AUDIO	The user may hear music that enhances their play experience	May
UR_LEADERBOARD	The top five scores are saved and able to be displayed	Shall
UR_ACHIEVEMENTS	The game will contain achievements for certain player behaviour	Shall
UR_PERFORMANCE	The game should be reliable on an average machine	Should

Functional Requirements Table		
ID	Description	Corresponding User Requirements
FR_MAZE	The system shall generate a university-themed maze that the player must escape from.	UR_ESCAPE_MAZE
FR_TIMER	The system shall provide a timer that tracks playtime and controls the time limit.	UR_TIME_LIMIT
FR_END_GAME	The system shall end the game when the player escapes, is caught by the Dean, or the timer expires.	UR_ESCAPE_MAZE
FR_PAUSE	The system shall fully pause all gameplay when the player pauses the game.	UR_PAUSE
FR_PAUSE_MENU	The system shall display a pause menu that provides appropriate options.	UR_UI, UR_PAUSE
FR_POSITIVE_EVENTS	The system shall include visible events that provide beneficial effects.	UR_POSITIVE_EVENTS
FR_NEGATIVE_EVENTS	The system shall include visible events that hinder progress.	UR_NEGATIVE_EVENTS
FR_HIDDEN_EVENTS	The system shall include hidden events that may help or hinder the user.	UR_HIDDEN_EVENTS
FR_DEAN	The system shall display and control Dean character that the player must avoid.	UR_DEAN
FR_PROFESSOR	The system shall display and control Professor character that the player must interact with in order to pass through a door.	UR_PROFESSOR
FR_WAY_OUT	The system shall provide an exit point to escape the maze.	UR_ESCAPE_MAZE
FR_SCORE	The system shall calculate the player's score, based on time to escape and interactions with events.	UR_SCORE, UR_POSITIVE_EVENTS, UR_NEGATIVE_EVENTS, UR_HIDDEN_EVENTS
FR_BOUNDARIES	The system shall enforce boundaries that the player cannot pass.	UR_BOUNDARIES
FR_EVENT_AMOUNTS	The system shall include a minimum of 5 negative event types, 3 positive event types, and 3 hidden event types.	UR_EVENT_AMOUNTS
FR_THEME	The system shall simulate a university-like environment featuring appropriate scenery and aesthetics.	UR_THEME
FR_EVENT_TRACKER	The system shall include counters that track the total number of each event type that the player has triggered.	UR_EVENT_AMOUNTS
FR_DIFFICULTY	The system shall provide at least a base difficulty where pausing is always enabled.	UR_DIFFICULTY
FR_DO_NOT_SAVE	The system shall not save the user's progress once they have completed the game session	UR_DO_NOT_SAVE
FR_LEADERBOARD	The system shall display a leaderboard with the top five scores achieved on a device	UR_LEADERBOARD

FR_ACHIEVEMENTS	The game shall have achievements that the user can receive when they display certain behaviour	UR_ACHIEVMENTS
FR_AUDIO	The game shall play appropriate audio to the player given different game events	UR_AUDIO

Non-Functional Requirements Table

ID	Description	User Requirements	Fit Criteria
NFR_PAUSE	The system shall pause within an acceptable time frame upon player request.	UR_PAUSE	<0.2 second after pressing pause
NFR_END_GAME	The system shall move the game to an end state immediately after an end condition (escape, capture, time expired) is reached.	UR_ESCAPE_MAZE, UR_DEAN	<3 seconds after escaping or encountering the dean
NFR_PERFORMANCE	The game should run smoothly on an average computer	UR_PERFORMANCE	0 crashes in test runs under normal conditions
NFR_TIMER	The system shall track and display the remaining time accurately,	UR_TIME_LIMIT	Timer reduces by 1 second per real time second
NFR_EVENT	The system shall trigger positive, negative or hidden events immediately upon event activation.	UR_POSITIVE_EVENTS, UR_NEGATIVE_EVENTS, UR_HIDDEN_EVENTS	<1 second delay after an event is initiated
NFR_SCORE	The system shall update the player's score in real time	UR_SCORE	<1 second after score changes
NFR_PRESERVE_ GAMESTATE	The system shall preserve and subsequently restore the current gamestate upon pause.	UR_PAUSE	Gamestate is successfully preserved.
NFR_DO_NOT_SAVE	The system shall not save any progress the player has made once the game has ended.	UR_DO_NOT_SAVE	Once the player completes the game, they cannot access the previous game
NFR_UI	Menu options will be clear and unambiguous	UR_UI	90% of players shall press the correct menu options first time

Constraint Requirements Table

ID	Description	Priority
CR_SPEND	The project should not exceed £50 in total development cost.	Should
CR_ENGINE	The game shall be implemented using an open source engine and open source tools.	May

CR_COPYRIGHT	All 3rd party assets and resources shall be open source, free of copyright, or otherwise morally and legally acceptable for use	Shall
CR_PLATFORM	The game shall run on Desktop only, on any Operating System without needing specialist hardware	Shall