Requirements

Team 15
Team 14

Joe Wrieden
Benji Garment
Marcin Mleczko
Kingsley Edore
Abir Rizwanullah
Sal Ahmed

Babar Khan
George Rogers
Jacob Turner
James Crump
Chloe Remmer
Shijie Lin

Requirements Specification v0.5

Preface

This document is intended to be read by the System Engineers as well as the Stakeholder. The document is added to as new requirements arise (refer to version history for this document's various iterations).

- v0.1 Creation of the document, outlining the Introduction.
- v0.2 User, functional and non-functional requirements have been added but require revision.
- v0.3 System requirements have been revised. System Requirements added to as we checked back and found childless User Requirements. The additions could be revised.
- v0.4 Notes section in the User Requirements added to. This needs to be revised and linked to the Risk Assessment and Mitigation.
- v0.5 Added to and revised. Final version.

Introduction

Detailed below are the user requirements we gathered from the product brief and refined into system requirements in team-customer meetings between the development team and the client. The software is described in its entire scope; each requirement describes the game in enough detail to meet all of the client's needs. Requirements were elicited through several meetings with the Client, with original requirements expanded on during those sessions. The updated requirements will be further elicited in another meeting with the client scheduled for the: 29th February 2021.

The system itself is to be a single-player video game imitating the York Dragon Races, in which the player controls a boat in a top down view as it traverses a river in a race against other boats, controlled by the computer. The system will be written in Java and shall be available as a zip file, consisting of the code and an executable JAR file containing any external dependencies, and thus would be OS independent.

The following requirements are presented through natural language, without excessive technicalities so that it can be easily understood by the customer. They are written in a concise format so that confusion caused by ambiguity will be avoided, but have enough detail so software developers have enough guidance. Each record in the tables represents an individual requirement, referred to by its ID. This format works well as it provides an unambiguous way of defining requirements that can easily be understood by the customer as well as being a checklist for the software developers to refer back to when implementing changes. The ID's are useful to help to quickly refer and identify different requirements.

User Requirements

ID	Descriptions	Priority	Notes
UR_BOAT_UNIQUENESS	Every boat involved in the game must be unique in terms of speed, acceleration, maneuverability and robustness.	Shall	We should have every boat have its own strengths and weaknesses.
UR_DIFFICULTY_LEVEL	For each subsequent leg in the game, the difficulty of the game will increase. (This could be the rate at which obstacles appear during the race.)	Shall	The difficulty shouldn't be to the extent that the player loses hope in the game, nor so low that they become disinterested.
UR_PADDLERS_STAMINA_DECREASE	Paddlers in the teams get tired over time, so their boat's speed, acceleration and maneuverability decrease progressively during every leg.	Shall	The movement of the paddlers should still be of a visually engaging speed.
UR_PLAYER_PENALTY	Every boat must remain in its lane for the duration of the race. Leaving the lane may result in a penalty at the discretion of the chief race official.	Shall	The boundary detection shouldn't be consistent and not seem buggy to the player.
UR_OBSTACLES	Obstacles will be present in the game and teams may find these obstacles in the river during the race.	Shall	There may be so many obstacles that the player cannot get past without hitting them.
UR_OBSTACLE_COLLISION	When a boat collides against obstacles, this will progressively reduce the robustness of the boat, until it breaks down (resulting in the end of the game).	Shall	The system must spawn obstacles appropriately and not cause the player's avatar's health to be reduced to a great amount too easily.
UR_UX	The game should be playable and enjoyable. It should offer a pleasant user experience.	Shall	This is very subjective and not always easy to measure. Associated risks are R6 and R7.
UR_MOVEMENT	The user must be able to move left, forward and right.	Shall	The user must be informed beforehand how to move their avatar.
UR_POWERUPS	The user may be able to pick up powerups that will benefit them, this includes: Speed boost, health regeneration, shield, bomb and stamina regeneration	Shall	Power-Ups should be found randomly throughout the game
UR_RACE_T	The user must play three races	Shall	It must be possible for the player to

OTAL	before it is decided whether they are eligible to compete in the final.		compete in the final.
UR_DIFFICULTY_BEFORE_GAME	The user may be able to choose different difficulty settings before the game, such as: easy, normal and hard	Shall	There will have to be a menu screen from which the player can select before the game actually starts.
UR_GAME_LENGTH	One whole game should last between 3 to 5 minutes.	Should	Each race should end in approximately one minute.
UR_GAME_END	The game must end if the boat is broken, the user finishes 3 races and does not get into the final or if the user finishes the final.	Shall	We need to make sure it is possible for all three outcomes to occur.
UR_SAVE_RELOAD	The user will be able to save a game during play which can then be reloaded at a later	Shall	Game should be able to accurately reload obstacles and boat at a similar position to where the game was saved
UR_SCREENS	The user should be able to see a starting and ending screen that will detail how the game is to be played and what place they game in the race respectively. A similar screen should be used for the user saving/reloading the game	Shall	Styling should be consistent and the start screen should contain enough information for the user to be able to play the game

Functional Requirements

ID	Description	User Requirements
FR_CHOOSING_UNIQUE_BOAT	The system must provide an option to choose a unique boat before the game starts.	UR_BOAT_UNIQUENESS
FR_DIFFICULTY_SELECTION	The system allows the user to select an (initial) game level of difficulty from easy, normal or hard.	UR_DIFFICULTY_BEFORE_GAME
FR_OBSTACLE_POWERUP_RATE	The system must spawn obstacles and power-ups appropriately, depending on raceNo and/or difficulty.	UR_DIFFICULTY_LEVEL UR_POWERUPS
FR_BOUNDARY_DETECTION	The system must be able to detect whether the user has left their individual lane and award an appropriate penalty	UR_PLAYER_PENALTY
FR_OBSTACLE_SPAWN	The system must not have obstacles spawn in a way that is impossible for the user to avoid colliding into them.	UR_OBSTACLES
FR_COLLISION_DETECTION	The system must be able to detect whether the user's boat has collided with an obstacle or power-up and respond	UR_OBSTACLE_COLLISION

	appropriately.	UR_POWERUPS
FR_HIT_DECREASED_BOAT_CONDITION	The system decreases the robustness of the boat if it has been hit by an obstacle.	UR_OBSTACLE_COLLISION
FR_INPUT_DETECTION	The system must be able to detect key presses and user selections/clicks and act accordingly	UR_MOVEMENT
FR_BOAT_BREAKAGE	The system must be able to detect whether the boat is broken.	UR_GAME_END
FR_GAME_DURATION	The system runs from between 3 to 5 minutes.	UR_GAME_LENGTH
FR_QUALIFIER_RACES	The system allows the user to play at least three boat races.	UR_RACE_TOTAL
FR_FINAL_RACE	The system allows the user to play in the championship finals race (on top of the other three), if they have performed well enough previously.	UR_RACE_TOTAL
FR_TIMED_DECREASED_BOAT_CONDITION	The system should have the boat's speed, acceleration and maneuverability decrease progressively over time throughout the race.	UR_PADDLERS_STAMINA_DECREASE
FR_POWERUP_EFFECTS	The system should detect which power-up has been collided with and adjust the boats attributes or influence the game according to the specific power-up	UR_POWERUPS
FR_SAVE_RELOAD	The system should 'save' the users game and be able to reload the game with the users boat having the same attributes as before and similar position in the race	UR_SAVE_RELOAD
FR_SCREEN_DISPLAY	The system should display an appropriate screen according to if the user has started/finished the race or wants to save/reload their game	UR_SCREENS
FR_INFORMATION_DISPLAY	The system must explain how the game is played and the different attributes, obstacles and power-ups	UR_SCREENS

Non-Functional Requirements

ID	Description	User Requirements	Fit Criteria
NFR_POSITIVE_UX	The system should maintain the player's engagement.	UR_UX	Throughout the game.
NFR_RELIABILITY	The system should be reliable when saving the users game	UR_SAVE_RELOAD	Accurately save: 99% of the time

Bibliography

- "Software Engineering", Ian Sommerville, Chapter 4

- Omar Elgabry’s Blog - “Requirements Engineering - Requirements Specification (Part 3)”
    - https://medium.com/omarelgabrys-blog/requirements-engineering-elicitation-analysis-part-5-2dd9cffafae8