Requirements

Team 10: Hard G For GIFs

Dragos Stoican

Rhys Milling

Samuel Plane

Quentin Rothman

Bowen Lyu

Jack Gerhard

a)

To define requirements we decided upon using IEEE's 12207.1 [2] standards as a base and also followed their terminology. As a supplement, we also used guru99's article [1] to help us clarify the system requirements. to assist us in defining the system requirements in particular.

The requirements defined here were based on a few factors. We first analysed the assessment product brief and used specific features as a base. Then we expanded on them, discussing as a group how to implement this, ensuring that each requirement was feasible and achievable.

We also organised two meetings with the customer so they could review our proposed requirements and understand how we were going to develop the system. This allowed us to reduce the chance of misinterpretations, clarify what the end product was, and resolve any early issues we had.

We have sorted our requirements, into three separate tables:

1. User requirements are high-level abstractions, designed to be presentable to our customers and non-technical people in order to help us develop our functional requirements.
2. Functional requirements, a subset of system requirements we split up the user requirements into a more concrete implementation. This is the "business logic" of the game. Defining the behaviour (inputs and outputs) and how everything interacts together.
3. Non-functional requirements, also a part of system requirements, focusing more on the quality of the game. Not completing these requirements would result in a game that would be unsuitable for the customer.

To represent this data we have chosen to present through tables each focusing on different groups of requirements. The tables all follow a similar layout, the ID column uniquely identifies every requirement, with a person-friendly identifier and a number as a short representation (1st number representing the type i.e. functional / non-functional, etc. The 2nd number represents the object to which a group of requirements relates to i.e. boat. Finally, the 3rd represents the position in the sub-table to help us identify each requirement in the list.

The priority column helps us decide what will come first during the implementation, each requirement is ranked on the following scale:
- "Low", this requirement would be implemented for the final release build, but not in the earlier beta builds.
- "Medium", this requirement would be implemented on our second build (v0.2) as it isn't a fundamental requirement which others build off, but it should be done before the deadline as it is still a core requirement, ensuring the implementation is to a good standard and any issues are ironed out.
- "High", needs to be implemented in our first stable version (v0.1). A lot of other requirements depend on this being present.

b) User Requirements

ID	Description	Test	Assumptions	Priority
Boat				
UR_UNIQUE_ BO ATS 1.0.0	In all the different boats, they must all be unique in their own ways. Acceleration, Speed, Maneuverability, and Robustness. Create boats based on these.	Boats can be assigned different attributes Measurable by comparing how much the boat movement after: player interaction when hitting obstacles The boats look visually distinct	There is a base boat Boats have attributes to control their characteristics Boats can be represented by an image	Mediu m (v0.2)

UR_DAMAGE 1.0.1	Colliding with obstacles damages the boat	When a boat collides, the stamina should be reduced	Boat, movement, and obstacles have been implemented	Medium (v0.2)
UR_CONTROLS 1.0.2	The boat responds to the controls inputted by the user in a comfortable way	Boat's course will be adjusted when a user uses the specific key bindings	A base boat has been implemented and is currently stationary	High (v0.1)
Map				
UR_MAP 1.1.0	The map through which players race includes obstacles	Must be long enough in length to cover an entire leg Features obstacles which will reduce the player's stamina	Need to make a background graphics to represent the map	High (v0.1)
UR_LANE 1.1.1	Stay in your lane, or you get a penalty of X seconds for the amount of time you spend out of it	Deliberately move the boat to the other lane, and check if a warning is given. Check to see if the penalty is different for when the time spent in it changes.	Map has already been implemented	Medium (v0.2)
UI / HUD				
UR_HUD 1.2.0	Have a clear UI that is easy to understand and see what corresponds to what	Displays the timer, leg, position, stamina, and robustness	Boats with variables to store their statistics have been implemented	Medium (v0.2)
UR_MENU 1.2.1	The game has a main	Displays game logo,	Screens have been	Medium

	menu screen	buttons to navigate the game	implemented	(v0.2)
UR_CHANGE_RE SOLUTION 1.2.2	The game should offer the option to play in different resolutions	The UI elements location and sizes should be based on the screen width and	Game works in at least one resolution	Low (v1.0)

		height		
UR_DISPLAY INSTRUCTIONS 1.2.3	The game should display instructions about the input it is expecting from the user	Display text that explains what the player has to do next	Game works, and the input used for the game has been determined	Low (v1.0)
UR_CHANGE_ SE TTINGS 1.2.4	The user should be able to change multiple settings on the settings screen	The available options should include audio, contro,l and graphic settings	Game works with default settings	Low (v1.0)
Game				
UR_DIFFICUL TY 1.3.1	Each leg of the race becomes harder	AI stats, obstacle amount, global speed movement are increased	Map, AI, unique boats, legs	Low (v1.0)

Functional Requirements

ID	Description	Test	Alternatives	UR_ID
FR_STAMI NA 2.0.0	Each team gets tired and slower as the race progresses.	Reaction speed to user controls progressively gets worse	Boat speed is reduced.	1.0.0
FR_STATS 2.0.1	Define each stat and how they impact the boat and each other	Each stat that a boat has has an effect on its performance	Boats all act in the same way and races must be won through skill	1.0.0
FR_ASPECT 2.0.2	Each boat has a different aspect	Check each boat texture to ensure they are significantly different from one another	Use boats of different sizes with the same texture to differentiate between them	1.0.0
FR_OBSTAC LE S 2.0.3	Different obstacles appear on the course randomly.	Boat's robustness is decreased by the specific damage of the obstacle	Instead of every obstacle having unique damage amounts. Make it randomly generated	1.0.1 1.1.0

FR_PENAL TY 2.0.4	Get a penalty for time spend outside the lane	One of the boat's stats are modified as a penalty temporarily	Boat is frozen / on a cooldown before they can move again. Other players stats are improved temporarily	1.1.1

FR_LANE 2.0.5	Determine if a boat crosses out of its lane	The penalty system will engage on the person	A pop-up could occur to notify them to move back	1.1.1
FR_DAMA GE 2.0.6	Detect a collision between a boat and an obstacle	Boat's robustness is decreased	Sound effect when a collision occurs	1.0.1
FR_HEALTH BA R 2.0.7	Each boat has a health / robustness bar that decreases when obstacles are hit	Health bar is updated when a collision occurs	Update the robustness bar on a timer, rather than immediately after a collision	1.0.1 1.2.0
FR_MINIM AP 2.0.8	The player can see a minimap in one of the corners of the screen	Shows current location in the leg	Have the minimap just show upcoming terrain	1.1.0 1.2.0
FR_CONTR OL S 2.0.9	The player's movement should be based on the player's input	Boat's target position is updated to reflect the changed rotation	Use simple movement that doesn't involve the rotation of the boat	1.0.2
FR_VARIABL E_ CONTROLS 2.0.10	The player's ability to turn, as well as its speed, is based on stamina	Boat movement is gradual towards the desired location, factoring in current stats	Have a set time of movement, then a break needs to be taken	1.0.2

Non-Functional Requirements

ID	Description	Fit Criteria	UR_ID
NFR_FAST_CON TR OLS 3.0.0	When inputting the direction, the response from the game should be instant	In <0.5 seconds response to an input, if not lower	UR_BOAT_CONTROLS

NFR_FAST_TRAN SI TION 3.0.1	When switching between game states, response should be instant	In <0.5 seconds response to an input, if not lower	UR_MENU, UR_HUD

Constraint Requirements

We had to adhere to two restraints: our stakeholders, the customer, and The University of York Communications Office. We are required to market and sell our game to the stakeholders ensuring it is playable and enjoyable by the ENG1 team.

Bibliography

1

https://www.guru99.com/functional-vs-non-functional-requirements.html **2**

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=669648