Requirements

Group 11 - 11 Musketeers

Osama Azaz
Adam Dawtry
Tom Jackson
Brendan Liew
Holly Reed
Harry Ryan

2.(a)

To start the **requirements elicitation process**, we researched options and decided on the following:

- First we carried out a document analysis of the product brief. [1]
- Then had a meeting with the customer who is interested in trying to market and sell the game.
    - This was a semi-structured interview in which a combination of predefined and unplanned questions were asked [1].
    - It was useful to clarify some questions the team had about the product brief and ask if any additional requirements would be necessary.
- The next stage was to do a use case [WB.1] which is the scenario in which an actor interacts successfully with the system, in this case, successfully plays through the game.[2]
    - From this, we have picked out the requirements necessary to make that scenario successful.

We have ensured that each requirement possesses the characteristics laid out in the guidelines laid out in the additional guidance for requirement-related activities in ISO/IEC/IEEE 12207 and ISO/IEC/IEEE 15288 [3]. These characteristics are that each requirement should be: **Necessary, Appropriate, Unambiguous, Complete, Singular, Feasible, Verifiable, Correct, Conforming**. [3]

To make sure the requirements themselves are well worded, we will use the language criteria in the guidance. This involves not using unbounded or ambiguous terms such as superlatives, vague pronouns, and subjective language.

To present the requirements we have used a table consisting of:

- ID
- Requirement description,
- Priority for the requirement,

And added additional information afterwards for those with:

- Environmental assumptions made
- Risks
- Alternatives

In the priority column, we decided:

- High = pertaining to those in brief
- Medium = pertaining to Those in interview
- Low = Design Decisions/Implied

This is so that when implementing the requirements, we can decide which features are integral to meeting the brief and focus on the ones with higher priority.

SSON

The brief requires us to create a game that allows users to roam around a body of water and engage in combat with enemy colleges and ships in order to attempt to take them over all while avoiding a number of obstacles (e.g. Lake Monsters, Bad weather etc.).

Many of our technical and user requirements have associated risks in their implementation. For example some of the graphical requirements have both technical and political risks since there is an inherent risk of graphical glitches with implementation but there also may be disagreements about how certain graphics should look. Furthermore, the implementation as a whole has a number of risks related to feasibility, inner team politics, schedule and software.[2]

2.(b)

User Requirements

ID	Requirement	Priority
UR.START_SCRN	User can start the game via a start screen	Low
UR.SCRN_NAME	User can choose a screen name	Low
UR.SEE_POS	The user must be able to see their sprite on the lake	High
UR.TUTORIAL	The user must be able to see a tutorial	Medium
UR.CLG_POS	User must be able to see where the colleges are relative to them	Medium
UR.UPDATE_POS	The user must be able to move the ship on the lake	High
UR.COLLECT_PNTS	The user must be able to collect points	High
UR.ATK_CLG	The user must be able to attack colleges	High
UR.COLLECT_LOOT	The user must be able to collect loot	High
UR.VIEW_PNTS	User must be able to view their points/score	High
UR.VIEW_LOOT	User must be able to view their loot	Low
UR.CPTR_CLG	The user must be able to capture colleges	High
UR.SEE_TASKS	The user must be able to see tasks to complete	Medium
UR.RESTART_GAME	The user must be able to restart the game at any time	Medium
UR.FINISH_GAME	The user must be able to finish the game by returning ‘home’	Low
UR.LOSE_GAME	The user must be able to lose/die	Low
UR.ENMY_SHIP	The user must be able to see enemy ships on the lake or in docks.	High
UR.OBSTACLES	The user will run into obstacles on the lake like monsters	High
UR.WEATHER	The game may enter a state of bad weather	High
UR.SPEND_LOOT	The user must have a way to spend gold	High
UR.POWER_UPS	There will be five different special power ups available to obtain	High
UR.DIFFICULTY	The user will have the option to choose from different levels of difficulty	High
UR.SAVE_LOAD	The user may save the game state at any point and resume it later	High
UR.ATK_SHIP	The user should be able to engage in combat with enemy ships	High

Software requirements

Functional requirements

ID	Requirement	User Requirements
FR.START.SCRN	Software must display a start screen	UR.START_SCRN
FR.START.START	Start screen shall have a ‘start’ button	UR.START_SCRN
FR.START.EXIT	Start screen shall have an ‘exit’ button	UR.START_SCRN
FR.START.NAME	Start screen shall have a text box for entering a screen name	UR.START_SCRN, UR.SCRN_NAME
FR.DISPLAY.GUI	The software must render environment with a Graphical user interface	UR.VIEW_PNTS, UR.VIEW_LOOT, UR.SEE_TASKS

FR.DISPLAY.EDGE	The software must display a lake with boundaries	UR.SEE_POS, UR.OBSTACLES
FR.DISPLAY.SHIP	The software must display the user in the form of a privateer sprite	UR.SEE_POS
FR.DISPLAY.CLG	The software must display colleges	UR.CLG_POS
FR.DISPLAY.DOCK	The software must display other ships docked at colleges	UR.ENMY_SHIP
FR.DISPLAY.HUD	The software must include loot, points, health and a mini map in HUD	UR.VIEW_PNTS, UR.VIEW_LOOT, UR.SEE_TASKS
FR.DISPLAY.CAM	The software's camera must follow the users sprite	UR.SEE_POS
FR.TUTORIAL	The software must display a tutorial embedded within the gameplay	UR.TUTORIAL
FR.FREEMOVE	The software must allow the users sprite to freely move around the lake via input	UR.UPDATE_POS
FR.COLLISION	The software must implement a object collision system	UR.OBSTACLES, UR.LOSE_GAME
FR.BOUNDARY	The software must implement a boundary to the gameplay area	UR.UPDATE_POS
FR.AWRD.POINTS	The software must award points passively and for completing tasks/defeating colleges.	UR.COLLECT_PNTS
FR.AWRD.GOLD	The software must award gold for completing tasks/defeating colleges.	UR.COLLECT_LOOT
FR.ATTACKCURSOR	The users sprite must attack the colleges and ships when they are clicked on	UR.ATK_CLG, UR.ATK_SHIP
FR.CLG_ATTACK	The college must attack the player	UR.ATK_CLG
FR.CLG_HEALTH	The college must lose health when attacked	UR.LOSE_GAME
FR.CLG_CONVERT	When college loses all health, college becomes friendly	UR.CLG_POS, UR.ATK_CLG
FR.CLG_INFO	College must be implemented with [.1] colour, [.2] name and [.3] friendly docked boats [4].Health [5]. Plunder	UR.CLG_POS, UR.ATK_CLG
FR.OPTNL_TASKS	The game must generate a series of optional tasks that the user may complete	UR.SEE_TASKS
FR.KILL_SCRN	The game must display a kill screen on [.1] victory, [.2] loss, [.3] on restart button	UR.LOSE_GAME, UR.FINISH_GAME, UR.RESTART_GAME
FR.GAME_SOUND	The game shall have a sound: [1] Music [2] Sound effects [3] Be mute-able	UR.START_SCRN

Non Functional requirements

ID	Requirement
NFR.NETWORK	The program shall not connect to the network
NFR.STABLE	The Game shall be stable and not crash
NFR.GAME_TIME	The game shall only last 5-10 minutes - Fit criteria: 9/10 run throughs will last less than 10 minutes
NFR.UPDATE	Updated game files should be easily available

NFR.CVD	The game must be accessible to those with colour vision deficiency
NFR.LOAD_TIME	The software must load quickly - Fit criteria: The game must load in < under 30 seconds
NFR.SIMPLICITY	The game must be simple enough/have a good enough tutorial to be able to be played by those with no prior experience - Fit criteria: 9/10 players will be able to understand the game by the end of the tutorial

Constraint requirements

ID	Requirement
CR.RESOLUTION	The game should be able to be displayed on range of resolutions and make good use of space - Fit criteria: Display on 13”-27” screens, test with 13”, 47”
CR.LOW_SPEC	The game must run on a system with minimum specs 4gb RAM, a standard UK keyboard and mouse - Fit criteria: Game must be playable on system with these specs
CR.DEADLINE	Project must be completed by 4/5/2022

Environmental Factors:

- **NFR.GAME_TIME -** The game will be used at an open day where time is limited and fast throughput of people is preferable
- **NFR.UPDATE -** The user may have little to no technical experience, eg. with github and creating JAR files.
- **NFR.SIMPLICITY, UR.TUTORIAL, FR.TUTORIAL -** The user may have little to no game playing experience.
- **UR.RESTART_GAME** The user must be able to restart the game at any time

Associated Risks/Further Comments:

- **UR.START_SCRN:** Any offensive screen names or ones that contain profanity may have to be filtered out. This could be mitigated by a potential filter in the screen box.
- **UR.CLG_POS:** Arrows could be confusing for users, especially colourblind ones and will need identifiers for friendly/enemy colleges.An **alternative** to this could be via the MiniMap
- **UR.VIEW_LOOT:** May take up too much time while being low on our priority list.
- **UR.SEE_TASKS:** Tasks could come across as vague and confusing for some users.
- For any requirements that are about display or graphics there is a potential for internal disagreements about how the UI should look, resulting in unnecessary lost time.
- **FR.COLLISION:** Collision systems can often result in bugs and glitches with clipping resulting in a lowered user experience.
- **FR.FREEMOVE:** Controls must be intuitive and movement must be seamless.
- **FR.AWRD.POINTS/FR.AWRD.GOLD:** Risk of differences between points and gold being unclear to users.
- **NFR.STABLE:** Although the game has low machine requirements, it is hard for us to control the number of crashes that may occur.
- **NFR.GAME_TIME:** There may be a situation in which we compromise gameplay elements in order to reduce game time.
- **CR.RESOLUTION:** Risk of graphical glitches in scaling if not implemented properly.
- **CR.LOW_SPEC:** Performance cap may limit our sprite usage as too many may cause performance issues on lower spec machines

Bibliography

[1]
M. Yousuf and M. Asger, “Comparison of Various Requirements Elicitation Techniques,” *International Journal of Computer Applications (0975 – 8887)*, Apr. 2015. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.695.5985&rep=rep1&type=pdf.

[WB.1] https://engteam14.github.io/usecase

[2]
“ISO/IEC/IEEE International Standard - Systems and software engineering -- Life cycle processes -- Requirements engineering,” *ISO/IEC/IEEE 29148:2018(E)*, vol. 1–104, 2019, doi: 10.1109/ieeestd.2018.8559686.https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8559686