# Team iPatch

## Assessment 1: Requirements

Christian Pardillo Laursen, Filip Makosza, Joseph Leigh, Josh Wakefield, Tina Weng & Oliver Relph

# Elicitation, Justification and Negotiation of Requirements

In order to produce a valid table of requirements the requirement engineering process laid out in the SEPR requirements engineering lecture [1] was followed. In it the following steps were taken:

**Elicitation -** To collect the information necessary to start formulating requirements the assessment brief was read thoroughly by all team members, after which a discussion of what we wanted our game to look like took place. From here, a set of questions aimed at clearing up ambiguities and eliciting further requirements from the client was created and used in an interview with them. This provided enough context about what the client wanted to allow us to move on to analysis

**Analysis -** The first round of requirements analysis was a group meeting where our aim was to build a complete understanding of how the game would work. This was done by drawing aspects of the game on a whiteboard and discussing how they would be implemented in the game, changing the whiteboard drawing as needed. The final drawings were archived for future reference and all decisions regarding the game were recorded in a text document. Care was taken to cover all important aspects of the game. In addition, a UML diagram was created to fully cover the structure of the game and ensure no sections were left out of the requirements.

**Specification -** Requirements were specified from the documents produced in the analysis step. The IEEE Recommended Practice [2] was followed to an extent, although some parts were ignored to better match our requirements to the project at hand. The requirements were specified by covering all documents produced until then and making sure all decisions were recorded in a requirement. After producing a first draft, we held another meeting where we read the requirements together and discussed anything that was either missing or did not match someone's expectations, after which requirements were reformulated to match the outcome of this meeting

**Validation -** Once we had specified the requirements to a standard everyone was happy with we held another interview with the client where he looked through all requirements and pointed out any parts he did not agree with, after which changes were made to accommodate the feedback we were given.

The requirements are presented in a table, as this allows the user to view what they find relevant easily and introduces a uniform structure that improves consistency and ease of use. The table has 4 columns, which were chosen to provide a useful and complete requirements specification:
- User requirement: This describes what is expected of the system
- System requirements: These represent how the system will implement the user requirement at a lower level. They do not, however, describe implementation.
- Fit criteria: They provide an aid in testing whether the requirement has been reach and thus make it much easier to know what the aim is in implementing them.
- Risks / assumptions / alternatives: Additional comments which are useful when implementing the requirements, as they detail what to be careful in the implementation and what alternatives are available.

In addition, a constraints table is added to list any constraints imposed on our project. These are different from requirements as they are not easily testable, so they are kept separate.
Requirements and constraints are numbered to ease traceability.

# Requirements

| User requirement | System requirements | Fit criteria | Risks / assumptions / alternatives |
|---|---|---|---|
| 1. The player must control a ship with their keyboard | 1.a The player shall be able to change the spatial coordinates of the ship through input from the keyboard<br>1.b The ship has a numerical health parameter<br>1.c When the ship takes damage from any source the player's health is reduced | The player is able to command the ship to move forward.<br>When the ship is hit by an enemy, the ship's health is reduced | Risk - the player is unable to figure out the controls<br>Mitigation - show the controls on screen at the start of the game<br>Assumption - the player has access to a keyboard |
| 2. The game must allow for navigation and combat to happen | 2.a When the player enters combat the game state shall change accordingly<br>2.b In combat mode, the player can fire their cannons and cause damage to enemy ships or colleges | When the player gets close to an enemy, the mode changes | Alternative - only have one mode and combat is always enabled. |
| 3. The UI must display health, gold and points | 3.a There are elements on the screen that display health, gold and points values in real time<br>3.b These do not obscure gameplay by being placed close to an edge or a corner | Health, gold and points are shown on the screen in an unobtrusive spot | Alternative - hide the gold and points from the UI and only show them when paused |
| 4. The game perspective must be top-down 2D | 4.a The camera is positioned to view the game from the top down.<br>4.b The camera is far enough away to allow for awareness of the surroundings while being close enough to not hinder gameplay | The player sees the ship from the top and can confidently maneuver it while being aware of incoming enemies | Risk - the player is unfamiliar with the perspective<br>Mitigation - m |
| 5. The game must have a pause function | 5.a When the player presses the escape key the game mode shall change to paused<br>5.b When paused, there shall be no updates to the game state.<br>5.c When paused, a pause menu shall appear and allow the user to tweak options or quit the game | The player has a way of making the pause menu appear and thus pause the game until play is resumed | Assumption - the pause function is obvious to the player |
| 6. There must be a points system that indicates general performance | 6.a Points added via combat and overcoming hazards.<br>6.b The amount of points received scales with how difficult the enemies are<br>6.c Points are added at the end of the game depending on how quickly the game is finished | When the player defeats an enemy, their points increase.<br>As time passes, fewer points are granted at the end of the game | Assumption - player understands that points are good |

| | | | |
|---|---|---|---|
| 7. The player must be able to accumulate gold, and spend it to regain health and obtain upgrades | 7.a Gold count is increased when an enemy is defeated or captured<br>7.b The cost of items is subtracted from current gold when purchasing and the item is added to the list of owned items<br>7.c The player cannot have a negative gold count and is therefore impeded from purchasing items that would lead to this state | When the player defeats an enemy, they obtain gold as a reward. If the player has enough gold, they can purchase a repair to regain health or an upgrade to e.g. improve their weapons | Assumption - the function of gold is obvious to the player |
| 8. The player has health and can be damaged in combat. When the player's health reaches zero, they die | 8.a The player has a health value that can be depleted by hazards or in combat<br>8.b When the health value reaches zero the gameplay ends and a game over screen is shown | When the player takes a hit their health is reduced. If this puts them below 0 health, they die and the game over screen shows | Assumption - the player understands the repercussions of death.<br>Risk - the player is demoralised by dying and stops playing |
| 9. The map must contain islands. The player can interact with some | 9.a Islands cannot be passed through<br>9.b The player can interact with certain kinds of islands<br>9.c Colleges and departments shall be represented as islands. | When the player tries to pass through an island they instead collide with it. The player has a way to interact with islands | Assumption - there will be no structures that cannot be represented as islands. |
| 10. The player can purchase upgrades and repairs from departments | 10.a The player can purchase repairs which restore their health value<br>10.b The player can purchase upgrades which increase their maximum health and alter their attacks | When interacting with a department, the player is presented with a screen that allows them to purchase items | |
| 11. There must be at least five colleges and three departments in the game | 11.a Colleges can be either allied or hostile, this state can be altered by capturing them. Capturing is done through combat.<br>11.b Departments are split into branches, and each department sells one specific kind of upgrade<br>11.c Each level contains one college and at least one department branch | There are 5 levels the player can progress through, each containing their own college.<br>In each level a branch from each department is present | |
| 12. The player must be able to capture enemy colleges | 12.a The player can engage an enemy college, and will enter combat mode upon doing so<br>12.b When the college is defeated it adopts a friendly behaviour and the next stage is unlocked | When the player brings a college's health to zero, a gate to the next level unlocks and they gain a reward | |
| 13. Levels are finite and have defined edges. Going past them results in damage to the player | 13.a If the player travels beyond the playable area, their ship begins to take sustained damage over time<br>13.b The player is warned that they are approaching the edge of the playable area so they have time to turn back | When the player sails outside of the playable area, their ship is destroyed before they reach the edge of the map | Alternative - add walls around the edges of levels to prevent the player from leaving |

| | | | |
|---|---|---|---|
| 14. While roaming the sea, the player may encounter bad weather | 14.a There is a weather parameter that randomly changes between calm and stormy<br>14.b The storm generates wind which pushes the player in a set direction<br>14.c Stormy weather can create waves which travel in a straight line, which damage the player ship if they are touched | The game has two distinct weather modes, one of which makes navigation a greater challenge | Risk - bad weather makes navigation frustrating. Mitigation - clearly indicate what direction the storm is pushing the player in |
| 15. The game must have an objective. It should not be achievable at the start | 15.a The objective shall be communicated at the beginning and corresponds to defeating the last college<br>15.b The final objective shall be defeating an enemy with the highest health count in the game | The player finds out about the objective soon after they start gameplay. The player wins the game when they complete the objective | |
| 16. The player can encounter enemy ships and can engage in combat with them | 16.a There are several enemy classes which are distinguishable and have different behaviours<br>16.b When the player is close to an enemy, the enemy will engage and the mode will switch to combat<br>16.c The gold count shall increase by a variable amount upon defeating an enemy | When sailing around, the player can randomly encounter different kinds of enemies that will attempt to shoot at them | Risk - the amount of enemies is overwhelming Mitigation - a maximum amount of enemies on the screen at one time is established and enforced |
| 17. There must be a minigame separate from the main game | 17.a The UI shall change to indicate the player is in the minigame<br>17.b The player can earn loot from the minigame<br>17.c The minigame is accessible from its own island<br>17.d The minigame can be exited to resume normal play | When the player interacts with the minigame island, the UI changes and they can play the minigame until they decide to exit | Risk - player does not understand the minigame Mitigation - add an explanation and demonstration before letting the player have their first go |

## Constraints

| Constraint | Reason |
|---|---|
| C.1 The game must be playable on a typical home windows pc | The game will be used to show at open days, therefore we want it to run on the PCs the university will use |
| C.2 The game must on average last between 20-25 minutes from start to finish | As the game will be shown in open days, we'd like the players to complete it in one play session. In addition, since development time is limited, it allows us to focus on core mechanics rather than on extending game time |
| C.3 A 5 minute demo must show each gameplay mechanic in the game | For the game to be shown in open days, we want the player to feel like they understand the mechanics without having to play through the whole game |

# References

[1]     Kolovos, Dimitris. 2018. "Introduction To Requirements Engineering". Lecture, University of York, 2018.

[2]     IEEE Std. 830-1998, "IEEE Recommended Practice for Software Requirements Specifications, IEEE Computer Society," Software Engineering Standards Committee, 20 October 1998, No. SH94654.