# 1. Introduction

LifDoff is a turn-based strategy game in which both players select moves simultaneously. The game puts two players against each other as they maneuver underwater ships to destroy oncoming enemies and collect resources, winning when the other player's ships have been destroyed.

## 1.1 Purpose

This Software Requirements Specification (SRS) is a description of all functions and specifications of LifDoff. It gives a comprehensive look at all aspects of the game. This document is based on the IEEE 830 Standard.

## 1.2 Scope

This project has two main components: *Clients* and the *Server*. The *Server* exists on one machine only. Its task is to do all game computations. *Clients* are the individual instances of the game - one for each player. The *Client* relays information to the *Server*, which performs calculations and sends information back to the *Client.* The *Player* makes decisions and actions, inputs these into the *Client*, which then sends this information to the *Server*. This means there are two types of interactions: *Player-Client*  where the user is inputting commands to the game, and *Client-Server*, where the local game is sending or receiving information from the hosted *Server*.

## 1.3 Definitions, acronyms, abbreviations

### 1.3.1 Definitions

* **Client:** The local instance of the game GUI on a player's machine.
* **Server:** The unique hosted game to which other clients connect.
* **Player:** The user playing the game using the client GUI.
* **Host:** The player who is running the server in addition to running their client. This is the player "hosting" the multiplayer game.
* **Action:**  A command that players can choose to issue during their turn.

### 1.3.2 Acronyms

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
| AI | Artificial Intelligence |
| (G)UI | (Graphical) User Interface |
| IEEE | Institute of Electrical and Electronics Engineers |
| LAN | Local Area Network |
| SRS | Software Requirements Specification |
| TCP | Transfer Control Protocol |
| UML | Unified Modeling Language |

## 1.4 References

* IEEE Recommended Practice for Software Requirements Specifications. New York: Institute of Electrical and Electronics Engineers, 1998.

## 1.5 Overview

Chapter 2 provides a description of the game itself, with textual explanations of the possible interactions a player can have with the game, such as actions, victory conditions, and basic gameplay. Chapter 3 contains diagrams that follow UML 2.0 specifications to detail all processes explained in Chapter 2.

# 2. Overall Description

This section describes the game scope and functionality.

## 2.1 Product Overview

This product will be a complete single- and multi-player game. It will have functionality for the saving and loading of games, for networked multiplayer from different machines, and for single player versus a computer controlled player (or players) of different difficulties. The game features customizable units with multiple upgrade paths, allowing for many unique combinations which gives a large amount of replayability to even the single player game.

It will use a package of external libraries for game design called pyGame. Other resources like graphics will be created by the LifDoff team. Multiple GUIs will be available throughout the game for players to interact with:

* The **Main Menu** screen will be displayed as soon as the game application is launched. Several options will be available: New Game, Load Game, Options, Credits, and Exit.
* The **Multiplayer Lobby** screen will be displayed after the Join Multiplayer or Host Multiplayer options are chosen from the Main Menu. In the Multiplayer Lobby, current players will be displayed, as will a chat box for players to talk to one another before the game begins.
* The **Game Screen** is the main view for each player once they have entered a game. This screen shows both player's units, along with the game board and current statistics. It also has buttons to open the Shop and Upgrade screens, and a button for the Main Menu.
* **In-Game Views** are the multiple screens accessed from within the Game Screen:
  + The **Shop Screen** is accessed from the Game Screen. This allows players to purchase new offensive units to send at the opposing players using the money they have collected.
  + The **Upgrades Screen** is also accessed from the Game Screen. This is the screen where players spend the experience points they have accrued on upgrades for their 3 defensive ships.
  + The **Pause Menu** is accessed from the Game Screen. It has options for adjusting sound options, opening help, and saving or quitting the current game.

Because LifDoff will be written in Python 2.7, this version or a compatible version of Python must be installed on all machines running the game. LifDoff will be an entirely stand-alone game that will not require any resources besides Python on the machine it runs on.

## 2.2 Product Functions

This section outlines the different functions of the LifDoff software. Main functions are the overall objectives of the software, and user functions are the actions users may take while inside one of the main functions.

* Main Functions:
  + Host a game
  + Play a game
* User Functions:
  + Host multiplayer
    - Set number of players
    - Add AI players
    - Kick players
    - Change number of players
  + Join multiplayer
  + Load game
  + Chat in multiplayer lobby
  + Select actions
  + Complete turn
  + Save game
  + Exit game

## 2.2.1 Use Cases

Diagram goes here

## 2.3 User Characteristics

|  |  |  |  |
| --- | --- | --- | --- |
| **User** | **Specific Goals** | **Features** | **Interface Expectations** |
| Casual Gamer | Have fun during first time playing game | Easy to learn, | Intuitive even for inexperienced gamer |
| Moderate Gamer | Have fun quickly, enough depth to entertain on multiple plays | Not too difficult to learn, enough of a learning curve to make experience valuable | Intuitive, could be complicated if it is similar to other similar games |
| Intense Gamer | Lots of strategy, many levels of depth to learn | Lots of features, experienced player can beat inexperienced player almost every time | Intuitive, offers hotkeys for easier and quicker pace, can be as complicated as necessary |
| Server Host | Can easily set up server, little to no options to configure | Simple, easy to set up, lack of unnecessary or overly complicated features | Simple interface |

## 2.4 Constraints

This section outlines the constraints placed upon this software. It details individual constraints and the solutions that must be taken to prevent them from negatively impacting the software.

Obviously, hardware limits certain features of the game. The game must run in a small enough amount of RAM that most computers can play it with no decrease in functionality. This means that the number of units on the screen may need to be limited to stay at an optimal level of performance.

Because players will be playing from their own unique machines, steps must be taken to ensure stable (or at least predictable) performance when outside factors interfere with gameplay. Turns will be time-limited, so that if a player is taking too long, other players will not have to suffer indefinitely. Also, if a player disconnects due to loss of power or network issues, the game can be restored from a temporary save that will be written periodically without player interaction.

Player skill and game familiarity must be taken into account. As developers, the game will be incredibly familiar, but a player who has never seen the game before needs to be able to quickly pick up and understand the game without too much help.

## 2.5 Assumptions and Dependencies

This section details the assumptions the LifDoff team will make in regards to the users and machines that are running the software in addition to the necessary dependencies for successful execution.

It is assumed that all machines have Python 2.7 or higher installed, as this is necessary for the game to run. It is also assumed that any users that wish to engage in a multi-player game are both connected to the internet or a local network. It is also necessary that each machine has the ability to connect to or host a server, and thus the network cannot be blocked by a firewall that restricts the ability of the LifDoff software to communicate with a remote server or client.

## 2.6 Apportioning of Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | TITLE | DESCRIPTION | DELAY REASON | WHEN |
| 1 | Multiple Upgrade Trees (per ship) | Each ship could have 3 possible upgrade trees resulting in 9 unique ships, this can be reduced to one per ship | Run out of time |  |
| 2 | Screen Transitions | Screen transitions between different menus (main menu background scaling down to be the actual background of the game, etc) | Inexperienced development team |  |
| 3 | >15 Items | Having a large number of items might be unrealistic | Could be unrealistic to expect so many items, inexperienced development team |  |

## 2.7 Risk Management

This section outlines the risks this project will face and details their impact, probability, and exposure. It also describes the risk control methods that will be used to account for these risks.

### 2.7.1 Risk Assessment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Title | Description | Probability | Impact |
| 1 | PyGame library lacks functionality | PyGame library could not have methods or modules that are necessary for our game | High | High - the cost of impact would be having to change or rewrite features incompatible with PyGame libraries |
| 2 | AI Issues | The AI player for single player could be too strong/too weak | Medium | Medium - imbalanced AI would not be game-breaking but would make it much less fun for single player. |
| 3 | Randomly generated terrain is unfair | Each player has randomly generated terrain - if it isn't fair, one player could be at a significant disadvantage | Medium | Medium - this would make the game much less fun |
| 4 | Complicated interface | The game is too difficult for new players to pick up and play without having lots of problems | Low | Medium - this could severely deter one portion of the userbase from enjoying this product |

### 2.7.2 Risk Control

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Type | Leverage | Solution Summary |
| 1 | Assume |  | Plan for the fact that adjustments will have to be made to compensate for incompatibilities or missing features from PyGame |
| 2 | Avoid |  | Make AI too weak and make it stronger as time permits |
| 3 | Assume |  | Try to minimize overall impact of terrain at first, turn it up as long as balance holds |
| 4 | Avoid |  | Try to avoid this all together by erring on the side of a simplistic design, and then adding more complex features as deemed possible |