Conquer the Seas

# Design Document

Benson Perry Matt Dannenberg Brian Shaginaw

Section 1: Introduction

# 1.1: Purpose

This section of the Design Document will outline the purpose of the entire design document. It should be read regardless of the further intentions of the reader, as it will describe which sections will be most relevant for different types of readers using this document. This is a living document, in that it will continue to be edited and modified throughout the design process until its eventual completion and submission.

## 1.1.1: Foreword

Conquer the Seas is the first game being developed by the in-house software team LIFDOFF for the McGill University Software Engineering Project course. The purpose of this document is to describe the technical design of the software in detail. The purpose of this document is to describe the software design of the Conquer the Seas game and to explain how and why these designs were chosen. This document will detail the design, specifications, programming management, and test cases and testing procedures of the game.

## 1.1.2: Scope

**Section 2** will start with the architectural considerations, then move to design considerations, and finally go in-depth into describing the high-level design of the game. This includes the discussion of the design pattern, the final domain model, and the final deployment diagram. **Section 3** will detail the actual software organization, including a view of the subsystems in both text and drawing. This will contain a detailed breakdown of one specific subsystem, with UML class diagrams and a list and explanation of class variables for this subsystem. There will also be a critical section which contains algorithm selection, critical code snippets, a state chart, and calling sequence diagram. **Section 4** discusses programming management by outlining the directory structure and programming tools, software building method, coding agreement, mitigation procedures, installation procedures, and training guidelines. **Section 5** will describe the test cases and testing procedures used for this project. This includes a description of test-driven development, functional test cases, the error logger, and how we built classes for auto-testing. **Section 6** is the appendix, which contains definitions and other useful information.

# 1.2: Audience

This document can be read from several different perspectives. It assumes a rudimentary knowledge of computer games and ideally some experience in software development. Regardless, the types of readers and which sections apply most importantly to them are as follows:

## 1.2.1: Game players

For the reader whose main focus is playing the game, **Section 4** will be the most pertinent. This section will contain information about how the game is stored locally on the user's machine, and will contain information about the install and setup of the game. It also contains the system requirements of the game. **Section 3** may also be of interest if the player wishes to modify code or see the inner workings of how the game is designed.

## 1.2.2: Developers

For the reader whose main focus is understanding and possibly modifying the code, **Section 3** will be the most important. This section outlines the specifics of the code, going into great depth about all aspects of the structure of the code. It also details algorithms used and has many detailed diagrams to explain the module/object decomposition. Individual subsystems are described in detail, so a developer could read this to understand how the software was written from the ground up.

# 1.3: References, Terms, Definitions

Conquer the Seas is a large software project that uses many technologies that will be discussed throughout this document. This section contains a listing of terms and abbreviations that will be used and may be referred back to for maximum clarity in further sections. It also contains references to the other two documents written for this software project so far.

## 1.3.1: References

Two documents have been written previous to this document. The Preliminary Design Document contained information about the basic functionality and gameplay of the game. It also contained a timeline of milestones and deliverables for which the project still holds to. The second document is the Requirements Document, which describes all of the functions and specifications of the game and is based on the IEEE 830 standard. It also contains many terms and definitions (including acronyms) that will be useful throughout this document as well. Reading these two documents before this document would give the reader a much greater understanding of the project. An implementation document will be released in the weeks following the submission of this document.

## 1.3.2: Terms and Definitions

This section is a list of terms and definitions that are essential to understanding the rest of the document. Other acronyms have been previously defined in the Requirements Document in Section 1.3.2.

* **pyGame**
  + pyGame is a set of Python modules designed for writing games in Python. It is free and released under the LGPL License (http://www.pygame.org/LGPL) and will need to be installed on all user machines along with the corresponding version of Python. For more information about pyGame, visit the pyGame website: http://www.pygame.org
* **Unified Modeling Language (UML)**
  + Unified Modeling Language (referred to as UML throughout the rest of this document) is a standardized modeling language for graphical representations of software design and software systems. Diagrams in this document will follow UML specifications.
* **Sprite**
  + A sprite is a 2-dimensional image or animated image used in video game graphics. In the context of Conquer the Seas, sprites will be used to draw and animate the different units and items. These graphics will be created entirely by the development team.
* **HTTP**
  + Hypertext transfer protocol. A request-response protocol that is used to send and receive data over the internet. In this project, it is used for non-LAN clients to connect to the server.

# 1.4: Polices and Tactics

This section outlines the goals of this document and this version of the software build, as well as the guidelines and development methods used for creating this document.

## 1.4.1: Goals

The software project outlined in this document is intended to achieve and address the following goals:

* Create a video game that is enjoyable to a diverse user base
* Incorporate advanced features like networked multiplayer and loading/saving of networked games
* Use detailed design methods that allow for easy-to-understand code which in turn may be modified by players or other developers

## 1.4.2: Guidelines

This document will be developed and maintained according to specific guidelines to maintain consistency throughout the document. The guidelines this document adheres to are as follows:

* Abbreviations and terms that are used will be defined in **Section 1.3.2** unless otherwise specified

## 1.4.3: Development Methods

This section outlines the methods used in developing this document. This is so that the design of this document can be traced back to these steps. The methods used in developing this document are outlined:

* A general outline of the sections of the document was created first
* Using this general outline, the introduction was fleshed out, with information filled in as relevant sections were completed
* Descriptions and other important explanatory text was added to diagrams
* Terms and definitions section was updated with newly added terms
* Proofreading and extensive editing

Section 2: The Design

# 2.1: Architecture Considerations

# 2.2: Design Considerations

When designing Conquer the Seas

# 2.3: High Level Design

## 2.3.1: System Overview

The architecture of Conquer the Seas is at its most basic level a Client - Server architecture. There is one server (hosted on a player's machine) and multiple clients can connect to this server. By using a Client - Server architecture, this game will be on par with most modern multi-player computer games. Multiple players will be able to connect to one hosted server. The user running the server also runs the client, which connects to the server on the user's local machine.

Server Clients

This architecture allows for two distinct roles to be filled by users of the software. The server needs to be able to handle multiple incoming connections over LAN or HTTP. This is accomplished using Python with pyGame libraries. Python is one of the most popular high-level programming languages today, and the readability and clear syntax it provides coupled with the large amount of available libraries makes it ideal for this task.

## 2.3.2: Server Architecture

## 2.3.3: Client Architecture