Contents

[Introduction 1](#_Toc304554654)

# Introduction

The objective of this project is to create an interactive program that can be enjoyed by students from the sixth, seventh and eighth grade for an upcoming faculty of computer science career day event. The resulting software must be high quality, meaning it should be sleek and free of any major bugs or design flaws, on top of being well implemented. It should also demonstrate the fun things that can be accomplished by computer science students, which is why it would perhaps be best to design something that is fun to use; such as a game. The computers that will run our software will have access to the Internet, so we could even create an online game. The easiest way to do this would be to use a client-server architecture, where both students use a client that is connected to a server. This would be sure to impress prospective students.

We have a very short time to design and implement our software; as such, it cannot be overly complex. We also expect not to be allocated more than two computers at the event, which means that it should be a two player game. Finally, since we will be presenting our software at an even where software from other teams will also be present, we should make sure that our program can be understood and enjoyed in a relatively short time frame. This also means that the GUI must be simple and easy to understand.

Given the above constraints, we will be designing an implementation of Battleship. Battleship is a two player game that is simple enough to be implemented at the desired level of quality in a short time. From the player’s point of view, it is easy to play, it does not take a long time to complete a game and (more importantly) it’s tons of fun!

# Work Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| **Activity number** | **Activity** | **Duration (all in hours)** | **Dependencies** |
| 1 | Creating a work breakdown | 8 | - |
| 2 | Evaluate risks | 6 | 1 |
|  | Study the Battleship protocol |  |  |
| 3 | Create a high level server design plan | 6 | 2 |
|  | Create UML diagrams for the server |  |  |
|  | Write class documentation for the server |  |  |
|  | Review server documentations and UML diagrams |  |  |
|  | Design server test cases to test networking |  |  |
|  | Design server test cases to test Battleship protocol |  |  |
|  | Design server test cases to test that the Battleship rules are correctly implemented |  |  |
|  | Implement networking-related code for the server |  |  |
|  | Implement the game logic-related code for the server |  |  |
|  | Run the server networking test cases and debug |  |  |
|  | Run the server game logic test cases and debug |  |  |
|  | Run the Battleship protocol test cases and debug |  |  |
|  | Write test report for the server |  |  |
|  | Design the user interface |  |  |
|  | Create mock-ups of the user interface |  |  |
|  | Create a non-technical report to present the GUI to the CS department |  |  |
|  | Create a technical design document for a hypothetical programmer that could take over the project |  |  |
|  | Create a test plan to test that the client follows the Battleship protocol |  |  |
|  | Create a test plan to test that the client’s GUI works as designed |  |  |
|  | Implement the client |  |  |
|  | Run the Battleship protocol test plan on the client and debug |  |  |
|  | Run the GUI test plan on the client and debug |  |  |
|  | Create test plan to make sure the server and two instances of the client work together |  |  |
|  | Run the above test case to ensure that the server and the clients work together and debug |  |  |
|  | Compile all the source code together and submit |  |  |
|  | Each team member writes his final commentary |  |  |