Team Battleship

Timothy Ransome

Gabriel Smith

Data Communications

and Networks I

Fall 2009

Battleship: Rules

Battleship is a classic guessing game played by two players, invented by Clifford Von Wickler and patented by Milton Bradley in 1943. The object of the game is to destroy every enemy ship before your ships are destroyed.

Players place their ships on a 10x10 square grid. According to the traditional Milton Bradley version of the game, five ships are used. The lengths of the ships are five, four, three, three and two units. Ships cannot overlap, and they must be entirely within the playing area.

After positioning the ships, players take turns "attacking" a square on the opponent's board. Opponents announce whether the attack hit or missed. Once every square of a ship has been hit, that ship is considered "sunk." A player loses when all of their ships are sunk.

Our version of Battleship uses a slightly modified variation of the traditional rules. Upon starting the client, ships are placed randomly on each player's board. The turns progress following the traditional rules, and the game ends once a player win the games. If a player disconnects while a game is in progress, his or her opponent is declared the winner.

Battleship: Protocol

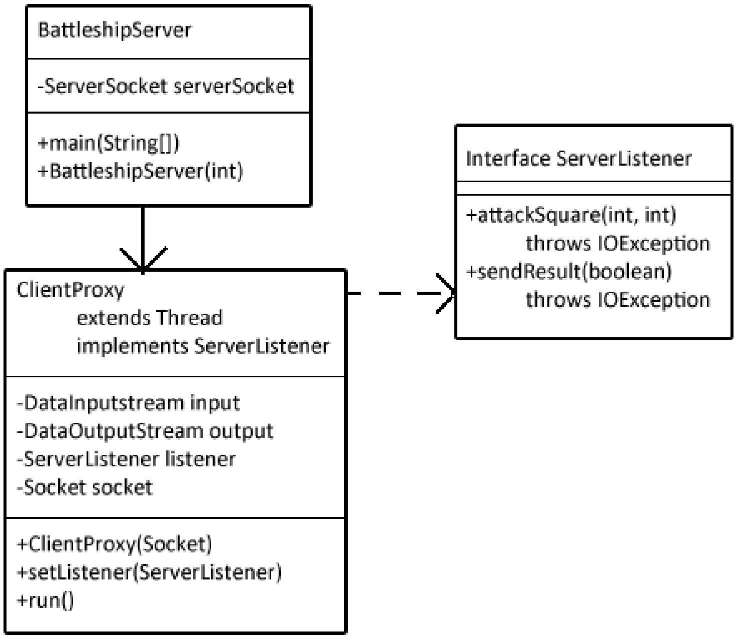
Two types of messages are sent over the application protocol. Attack messages are sent on a player's turn when they choose a position on the board to attack.

Attack messages are formatted as a lowercase 'a' character, followed by two integers. The integers correspond to the x and y components of the coordinate to attack. Once an attack is processed, the result must be sent back to the opponent.

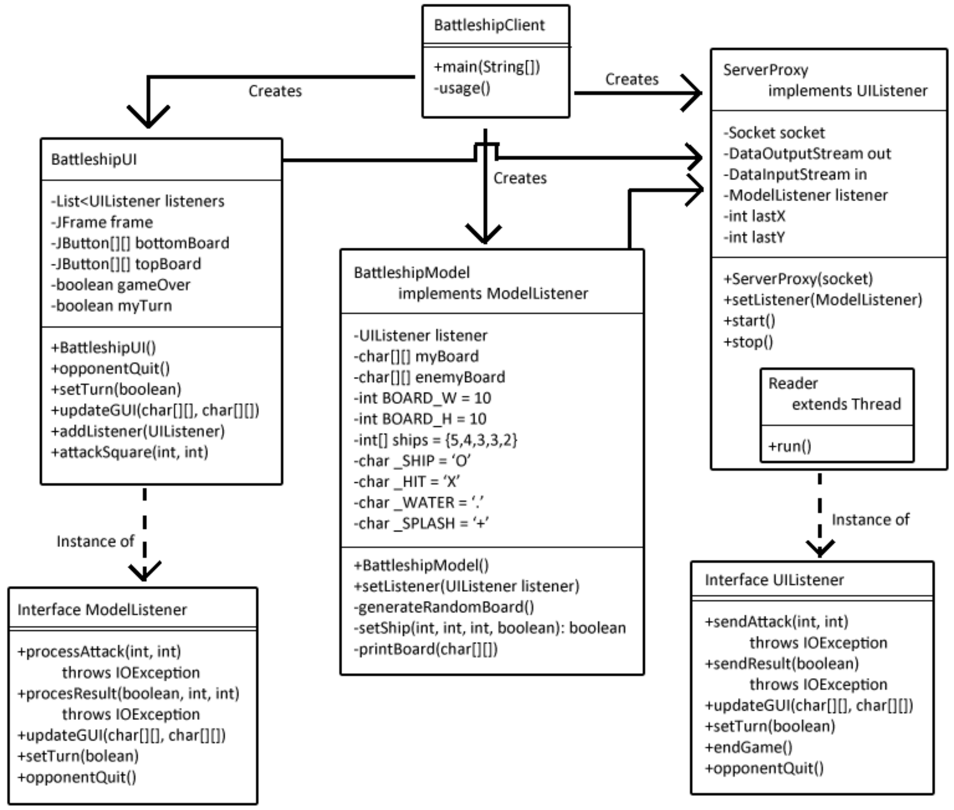
Result messages are formatted as a lowercase 'r' character, followed by a boolean. If the boolean value is **true**, it means that the previous attack hit a ship. If the boolean is **false**, then the player missed the ships and hit empty water.

Battleship: UML

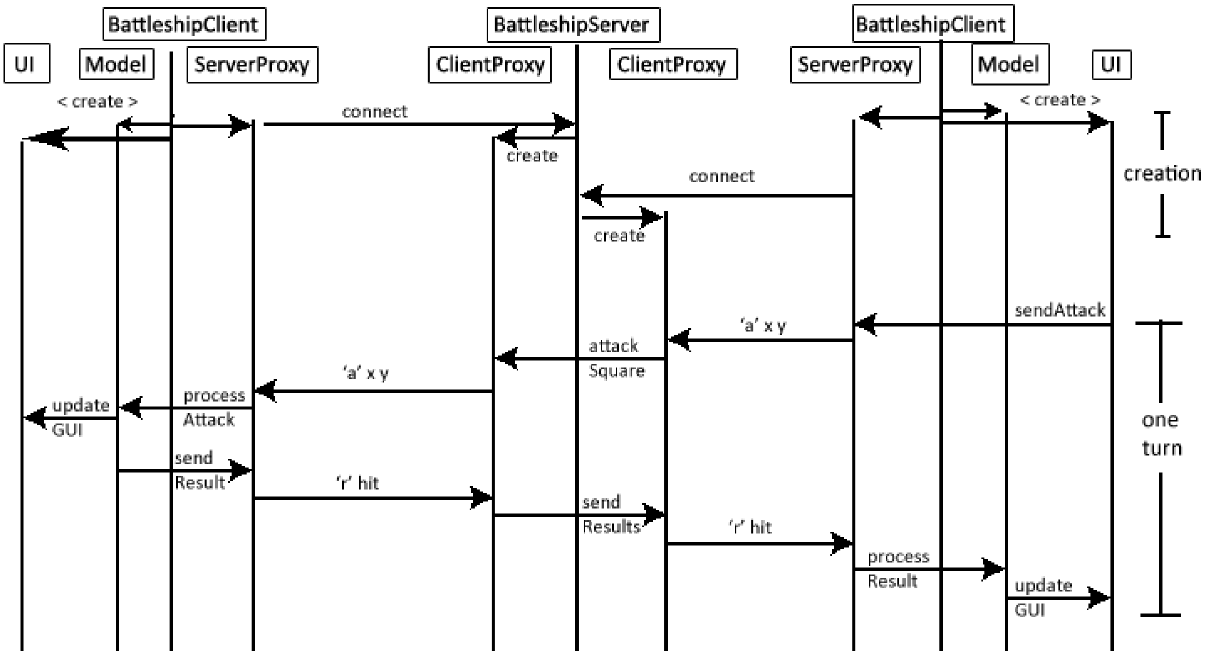
Server UML:



Client UML:



Sequence Diagram:



Compiling and Running Battleship

Compiling:

Before compiling the software, extract all files from the .jar archive. For Unix machines, use the following command to extract all files from an archive:

```
jar xvf <filename.jar>
```

Once the files have been extracted, they must be compiled. To compile all the .java files, use the following command:

```
javac *.java
```

Running:

Before running any Battleship clients, a server must first be set up to pair players together. To start the Battleship server, run the following command in the same directory as the Battleship class files:

```
java BattleshipServer <port>
```

Where **<port>** is an open, unused port on which the server can listen on. The server will run in the background and should not present you with any output.

Once the server has been started, clients can begin to connect to it. To start a Battleship client, run the following command in the same directory as the Battleship class files:

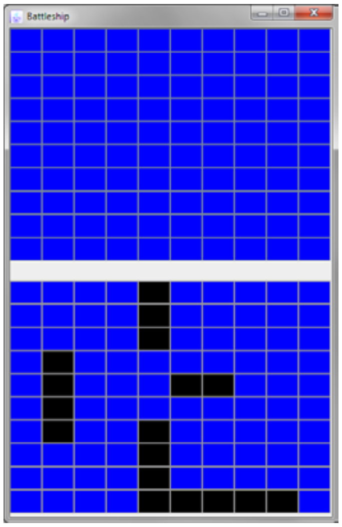
```
java BattleshipClient <host> <port>
```

Where **<host>** is the address where the Battleship server is running and **<port>** is the port on which the server is listening.

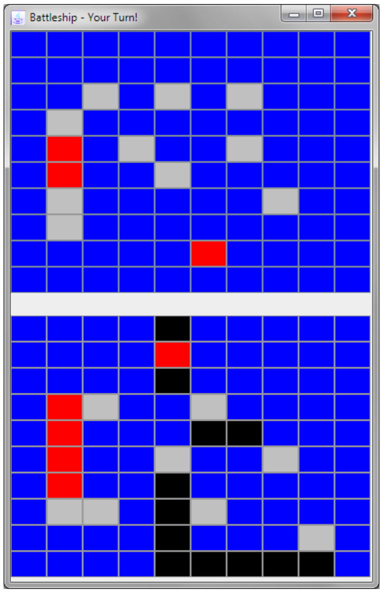
Playing Battleship:

If the client is successfully started, the server will attempt to match them with a waiting player. If there are no players waiting to play, the user will be placed into a queue until an opponent connects. Once two players are paired in a match, users will be presented with a screen containing two 10x10 grids. The bottom grid corresponds to the user's ship configuration.

The top grid displays the known configuration of the opponent's board. As positions are attacked, the opponent's board will be populated with the results of the attacks.



Initial Board Setup



Example Game Board

To begin the game, a player chooses and clicks any square of their opponent's board to attack that position. A player can only attack a previously un-attacked position. The attack will be sent to the opponent, and the results of the attack will be displayed. Once an attack is sent, it is then the opponent's turn. That player takes their turn, and the position they attack is displayed on your board at the bottom of the window.

Players continue taking turns attacking positions until one player runs out of ships. Once a player loses all their ships, the game ends. A window will pop up notifying the player of whether they win or lose, and the clients will disconnect.

What we learned:

During the development of Battleship, we encountered and overcame several obstacles. Each of these obstacles became a learning experience for us. We learned the most about developing network applications, since each of us had previous experience working with GUIs as well as games. Specifically, we learned how to develop a game with two clients taking turns sending messages and returning responses. Any future network applications we develop will be done more smoothly now that we have the knowledge needed to complete it from start to finish.