University of New Brunswick
Faculty of Computer Science

Course: **CS2043 – Software Engineering I**

Deliverable #: 02

Instructor: **Natalie Webber**

Date: February 2, 2017

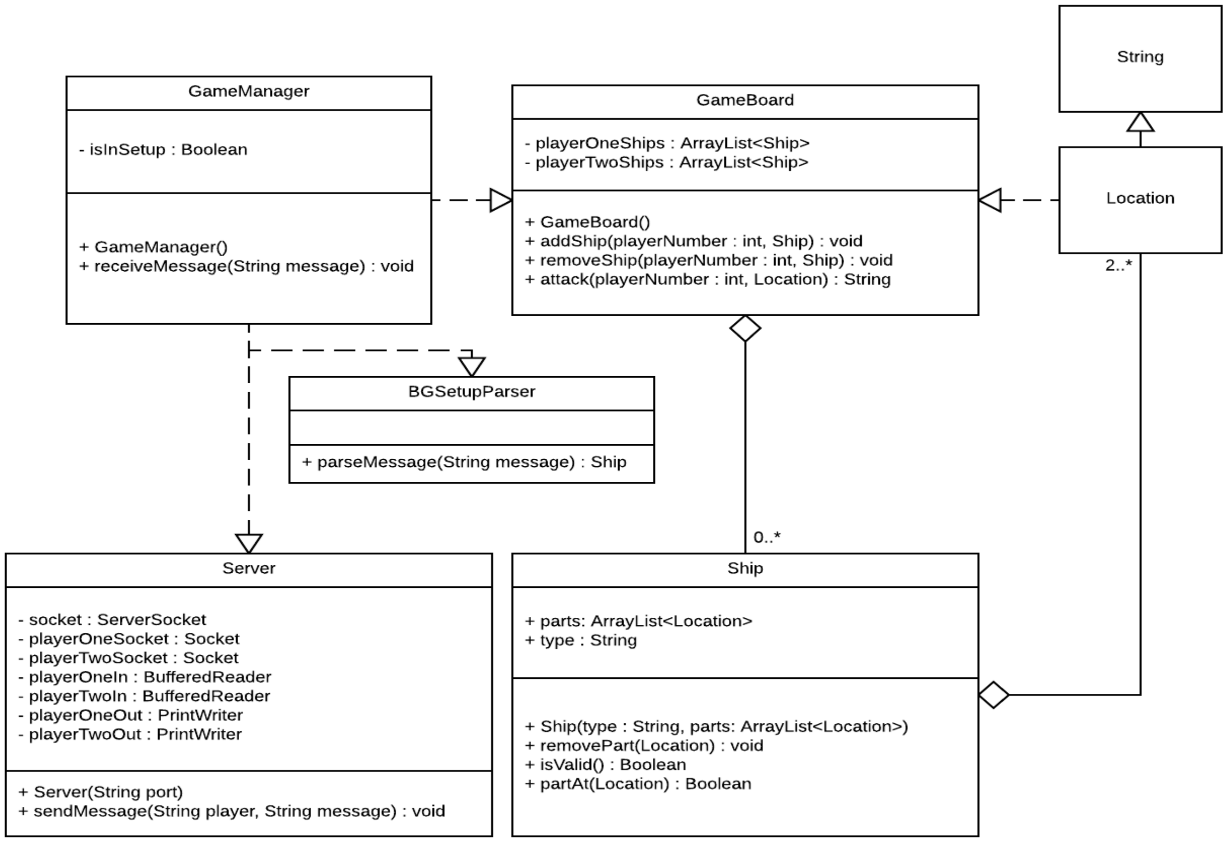
Project group members:

Student #1: 3516474 / *Shane Pelletier*
Student Number / Student Name

Student #2: 3413735 / Andrew Hampton
Student Number / Student Name

Table of Contents

UML Diagram



String
Location
2..*
GameManager
- isInSetup : Boolean
+ GameManager()
+ receiveMessage(String message) : void
GameBoard
- playerOneShips : ArrayList<Ship>
- playerTwoShips : ArrayList<Ship>
+ GameBoard()
+ addShip(playerNumber : int, Ship) : void
+ removeShip(playerNumber : int, Ship) : void
+ attack(playerNumber : int, Location) : String
BGSetupParser
+ parseMessage(String message) : Ship
0..*
Server
- socket : ServerSocket
- playerOneSocket : Socket
- playerTwoSocket : Socket
- playerOneIn : BufferedReader
- playerTwoIn : BufferedReader
- playerOneOut : PrintWriter
- playerTwoOut : PrintWriter
+ Server(String port)
+ sendMessage(String player, String message) : void
Ship
+ parts: ArrayList<Location>
+ type : String
+ Ship(type : String, parts: ArrayList<Location>)
+ removePart(Location) : void
+ isValid() : Boolean
+ partAt(Location) : Boolean

UML Description

The **GameManager** class is responsible for starting the server and for handling the game logic. It contains a Boolean flag to track if the game is being setup or if the game has begun. Its constructor starts the server, and its receiveMessage() method is called by the server whenever the server receives a message from the user.

The **Server** class is responsible for handling connections with the user and passing data between the user and the **GameManager**. It contains the socket that the server uses to accept connections, the sockets for player one and player two, a BufferedReader for players one and two, and a PrintWriter for players one and two. The BufferedReader is used to read from the players' sockets, and the PrintWriter is used to write to the players' sockets. The server's constructor takes in the port number to open a connection on as a String, and the sendMessage() method sends a message over the appropriate player's socket.

The **BGSetupParser** is responsible for parsing the setup messages from the **GameManager** and returning correctly constructed ships. It contains one method, parseMessage(), that the **GameManager** calls if it is in the game setup phase, and which returns a correctly constructed ship or null if there is an error in constructing the ship.

The **GameBoard** is responsible for keeping track of the current state of the game and for manipulating the state whenever the **GameManager** requests it to. It contains a list of both player's ships. Its constructor initializes the lists of ships. The addShip() method adds a ship to the specified player's list; the removeShip() method removes the ship from the specified player's list; the attack() method attempts to attack a ship in the specified player's list, returning the result of the attack (e.g. hit, miss, sunk).

The **Ship** is responsible for holding a single ship's parts and allowing those parts to be modified. It contains a list of the locations of a ship's parts and the type of the ship. Its constructor constructs a new ship from a type and a list of locations; its removePart() method removes the part at the specified location; its isValid() method returns true if all of the ship's parts are in valid locations; its partAt() method returns true if there is a part at the specified location.

The **Location** class is simply a renaming of Java's String class in order to make the design clearer when the design wishes to deal specifically with the location of a ship's part rather than simply a string indicating e.g. the name of the ship or a message passed to the server.

Time Sheet

Activity	**Start**	**Completion Date**	**Num Hours – Shane Pelletier**	**Num Hours – Andrew Hampton**
Prepare Project Plan	2017-01-13	2017-01-24	1.5	1.5
Prepare UML diagram and documentation	2017-01-24	2017-02-02	1.5	1.3s