University of New Brunswick
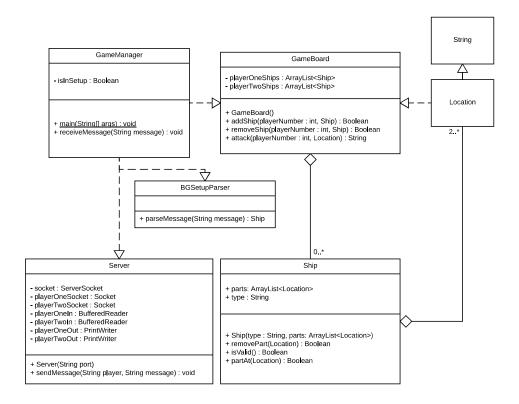
Faculty of Computer Science

Course: CS2043 – Software Engineering I Deliverable #: ـــــــ3ـــــــــــ

Instructor: Natalie Webber Date: ــــــــــــ2017-02-09ـــــــــــــــ

Project group members:

Student #1: ـــــــــ3413735ـــــــــــــــــــــ / ـــــــــــــAndrew Hamptonـــــــــــــــــــ

Student Number / Student Name

Student #2: ــــــــــــ3516474ــــــــــــــــــ / ــــــــــــــــShane Pelletierــــــــــــــــــ

Student Number / Student Name

Table of Contents

Updated UML diagram

GameManager

- isInSetup : Boolean

+ <u>main(String[] args) : void</u>
+ receiveMessage(String message) : void

GameBoard

- playerOneShips : ArrayList<Ship>
- playerTwoShips : ArrayList<Ship>

+ GameBoard()
+ addShip(playerNumber : int, Ship) : Boolean
+ removeShip(playerNumber : int, Ship) : Boolean
+ attack(playerNumber : int, Location) : String

String

Location

2..*

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) : Boolean
+ isValid() : Boolean
+ partAt(Location) : Boolean

This is an updated version of our previous UML diagram. The GameManager has the main method in it, and addShip(), removeShip(), and removePart() have the return type of boolean.

The Server Test Plan

Test Case Number	Method or Subsystem Tested	Input Data To Use	Brief Explanation	Expected (Correct) Result
1	Ship.isValid()	new Ship("FR", new ArrayList<Location>.add ("A1"));	Testing a ship that's too small	isValid() should return "false"
2	Ship.isValid()	new Ship("FR", new ArrayList<Location>.add ("A1").add("A2").add("A3"));	Testing a ship that's too big	isValid() should return "false"
3	Ship.isValid()	new Ship("FR", new ArrayList<Location>.add ("A1").add("B2"));	Testing a ship that's placed diagonally	isValid() should return "false"
4	Ship.isValid()	new Ship("FR", new ArrayList<Location>.add ("A11").add("A12"));	Testing a ship that's placed off the grid	isValid() should return "false"
5	Ship.isValid()	new Ship("FR", new ArrayList<Location>.add ("A1").add("A2"));	Testing a ship that's placed correctly	isValid() should return "true"
6	Ship.partAt()	new Ship("FR", new ArrayList<Location>.add ("A1").add("A2"));	Testing with a part that does not exist at the location	partAt("A3") should return "false"
7	Ship.partAt()	new Ship("FR", new ArrayList<Location>.add ("A1").add("A2"));	Testing with a part that does exist at the location	partAt("A2") should return "true"
8	Ship.removePart()	Ship temp = new Ship("FR", new ArrayList<Location>.add ("A1").add("A2")); temp.removePart("A2");	Testing removing a part	partAt("A2") should return "false"
9	GameBoard.attack()	Player 1 has FR at locations "A1" and "A2"	Testing attacking a ship that damages, but does not sink, the ship	attack(1, "A2") returns "hit"
10	GameBoard.attack()	Player 1 has FR at locations "A1" and "A2"	Testing attacking a ship and missing	attack(1, "A3") returns "miss"

11	GameBoard.attack()	Player 1 has FR at locations "A1" and "A2", with "A1" having already been hit, and player 1 has another ship on the board	Testing attacking a ship that sinks a ship	attack(1, "A2") returns "sunk"
12	GameBoard.attack()	Player 1 has FR at locations "A1" and "A2", with "A2" having already been hit	Testing attacking a ship that has already been hit	attack(1, "A2") returns "miss"
13	GameBoard.attack()	Player 1 has FR at locations "A1" and "A2", with "A1" having already been hit, and player 1 has no more ships on the board	Testing attacking a ship that sinks a ship	attack(1, "A2") returns "win"
14	GameBoard.attack()	Player 1 has FR at locations "A1" and "A2"	Testing attacking an invalid location	attack(1, "A11") returns "err"
15	GameBoard.addShip()	Player 1 has FR at locations "A1" and "A2", called with locations "A2" and "A3"	Testing adding an overlapping ship	Returns false
16	GameBoard.addShip() Player 1 has FR at locations "A1" and "A2", called with locations "A2" and "A3"	Player 1 has FR at locations "A1" and "A2", called with locations "A3" and "A4"	Testing adding a ship that does not overlap	Returns true
17	BGSetupParser.parseMessage()	Called with "AC A1 A2 A3 A4 A5"	Testing parsing a valid message	Returns a Ship where Ship.isValid() returns true
18	BGSetupParser.parseMessage()	Called with "AC A1"	Testing parsing a ship that is too short	Returns null
19	BGSetupParser.parseMessage()	Called with "AC A1 A2 A3 A4 A5 A6"	Testing parsing a ship that is too long	Returns null
20	BGSetupParser.parseMessage()	Called with "AC A1 A2 A3 A4 A6"	Testing parsing a ship with a gap	Returns null
21	BGSetupParser.parseM	Called with "AC A11 A2	Testing parsing a	Returns null

	essage()	A3 A4 A5"	ship with a part off the grid	
22	Server construction	Called with "3141", does not connect properly	Testing unsuccessful connection	Throws IOException
23	Server constructor	Called with "3141"	Testing successful connection	Does not throw an exception

Time Sheet

Activity	Start	Completion	Number Hours Shane Pelletier	Number 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	1.3
Prepare Test plan	2017-02-02	2017-02-09	1.5	1.5