Time line for the RC server:

26-07-14: I out line the main\_server class, it will reference a connections\_server class that will contain all of the subs for creating the socket and accepting. I am making it as of today as a bare bones scrip that connects then disconnects without doing anything, this is because until I finalise that hardware I cant be sure what I will need the server to do, I imagine I will need one class for using the GPIO pins and one for other operations but I don’t know at this point.  
  
I have gone thru the code and added informative error messages in the try and catch statements.   
  
Upon testing it appreres that sometimes depending on the network adapter being used the server dose not blind to the correct IP/Adapter. To fix this I have asked the user to enter the correct ip (along with instructions for finding this) and then after some research I have found I can use InetAddress.getByName(string) and used that when creating the ServerSocket.

30-7-14: I have changed from using a PrintWriter and buffred reader to Objectinput and output stream. Because as I put in the protocol document it allows me to send any data type. I used [this](http://www.java2s.com/Code/Java/Network-Protocol/ObjectInputStreamandObjectOutputStreamfromSocket.htm) to help me set up the streams, it turns out they are very similer to the printwriter and buffred reader, so little modification was needed to the code. But when i used the ObectInputStream.readobject to try to get an array it threw an error because it was of type integer not object so I used the advice given [here](http://stackoverflow.com/questions/12860780/receive-and-print-string-array-object-over-socket) to solve this issue.

11-8-14: I have made it so that the server loops constantly accepting input from the client, I have made it loop until the server has a disconnect command or there is an error, I have also gone and modified the existing try and catch loops so do this.

Mile Stone 1 11-8-14: This is the point that the server can fully communicate and receive the input looping. The error catching works. But it dose not move the car.

16-8-14: when the Client Dissconects it causes an EOFExeption(End of File) to be thrown, to stop this causing any problems I have made it so that the the main Subroutine throws EOFExeption.

I am also preparing for the components to arrive by adding the code to control the GPIO pins on the pi, the libery I am using ([pi4j](http://pi4j.com/)) only provides dabian packages to install it or JAR’s. So I need to work out how to install JAR’s so that eclipse my IDE will “see” them, so I can compile my project. I find [this](http://www.leepoint.net/notes-java/background/13files_and_directories/86packages-installing.html) web page, and I opt to put them in the directory of my JRE. Eclips however gave me an error Saying it is not accessible due to restriction on required library. I resurched a fix and found one [here](http://stackoverflow.com/a/9586411), this will not be a problem when it runs on the pi because I will install the package on the pi. Upon looking thru the [example’s](http://pi4j.com/example/control.html) and I see that it references GPIO pin 0-20 that not all of exist on a raspberry pi and higher numbers also do ([normal for reference](http://www.element14.com/community/servlet/JiveServlet/previewBody/68203-102-6-294412/GPIO.png)). After some looking thru the documentation, it uses a different way of representing the usable pins, stated [here](http://pi4j.com/usage.html#Pin_Numbering). This brings up another problem, I have a raspberry pi model b+ and pi4j only has support for the model b. But this should not be a problem as the only diffrences between the GPIO pins is that the b+ has 40vs26 pins, but the first 26 pins are the same so, so long as I only use those I should be fine. I will use pins **15, 16 and 1** for the driving motor and **pins 6, 10 and 11** for the steering motor.

17-8-14: I have use a Thread that will loop and update the directions of the motors. This is in the Move class and will be started one there is a connection, it then loops, and when the mainsever recives the array it tell the move class that there is an update and turns the array in to 3 local variables( in the doit Sub). The main loop in the thread then changes what pins are on in acodence to how those veriables have changed.   
I have don’t it this way because I did not want to take time of the main thread(they are done on a different thread) this is in case the operation takes more than 1/16 of a second and it misses some input and causes an error.

25-8-14: Im trying to locate the source of the lag. So I first check that it is not lag in the sockets conection, to do this I make it so that when the server recives a folward order it prints “front” to console, I notice that this takes 1 or 2 seconds, so I believe this is cause of the lag. After some internet searching I found [this](http://stackoverflow.com/questions/15729112/lag-over-sockets-java), this suggests that the lag is related the the TCP/IP connection and is inherent to TCP, and that instead I should use UDP. So using a [linked article in the answer](http://systembash.com/content/a-simple-java-udp-server-and-udp-client/) and the diagram sockets [documentation](http://docs.oracle.com/javase/7/docs/api/java/net/DatagramSocket.html), I rewrite the server to use UDP. I find in doing this that I cant directly take an integer or an array directly from bytes, so I split the array and take one string(containing just a number) at a time, then convert it to an int. because of the covertion from TCP to UDP I will no longer be using the Connections\_server class

26-8-14: upon testing the server with the temp UDP class, I find I get an error “at java.lang.NumberFormatException.forInputString(NumberFormatException.java:65)” this is related to the converting an int to a string, after seeing [this](http://stackoverflow.com/questions/9455070/getting-a-numberformatexception) I believe it is because when I convert it to a string in the client it pads it with empty space so I need to use the String.trim to get rid of that.  
after I did this and they where comunicationg the program would quit when ever I pressed the front or back button. So I commented out the part that sends the command and sideways so only the front part weill be sent, and I also made a 1s delay rather than 1/16s and this worked, I think it did not work because one of the problems with UDP is that packets are not garanted to be recived in the order they were delivered. So to fix that problem I have made changes to the protocol

29-8-14: I have changed the pins for front and back as after testing they are reversed, I think this is due to bad labelling on the original circuit board. I also changed the text that is displayed when you first start the program. This is to make it clearer and to acknowledge that I used the pi4j libery and to say that it is licensed under the Apache License, Version 2.0 as I am required to do. I also added a copyright to Tom Hazell it now says:

Copyright (C) 2014 Tom Hazell

This uses the Pi4j library to control the GPIO pins. It is licensed under the Apache License, Version 2.0. For a copy go to: http://www.apache.org/licenses/LICENSE-2.0

Make sure the raspberry pi is connected to a network (WIFI or Ethernet) and use ifconfig in terminal to find the IP address of the raspberry pi.

Enter the IP for the raspberry pi to listen on.

IP:

6-9-14: I will be changing the server so that when it recives a 6 it will replay with a one. I am also removing one of the places to recive a packet so that it recives once then loops,this is because the second one it not necessary as I can just recive it in the same place in code, this means that the motors will be updated more frequently. To do this I have added M.doit(Arrayin); in to the if loop and added an extra claase to the top of it saying “if (onei == 6){” then it send a “1” back to the client