

**Middleware**

Prepared by

Mazen Alkoa’ & Ahmad Nabulsi

Submitted to

**Dr. Sufyan Samara**

A documentation report of graduation project (66581/3)

Computer Engineering Department

Faculty of Engineering

An-Najah National University

Content List

List of figures……………………………………………………………………………1

Acknowledgements…………………………………………..……………….…….1

Abstract……………………………………………………………..…………….………1

1 Introduction …….……………………………………………………………………2

1.1 Overview.…………………………………… ……………………..……2

1.2 Existing Problems …………………………………………………...2

1.3 Related work ………………………..…………………………………2

1.4 Related topics ………………………………………………………….2

1.5 Motivation for Carrying out project …………………………3

2 Methodology………………………………………………………………….…….4

3 Result and Discussion…………………………………………………….……12

4 Conclusions and Recommendations …………………………….…..…12

4.1 Conclusions…………….………………………………….……….……12

4.2 Recommendation.……………………………………………………12

5 References …………………………………………………………….……………13

6 Appendix ……………………………..…………………………………………….13

**List of figures**

Figure 1.Flow of activities in android………………………………………………6

Figure 2.Flow of stages of the work………………………………………………..7

Figure 3.Communication between two peers…………………………………9

Figure 4.Dialog for selecting device to connect with…………………….10

Figure 5.Synchronization between two players……………………….……12

Figure 6.Middleware layer in context……………………………….………….16

Figure 7.transient synchronous communication…………………………..17

Figure 8.Client-server interaction………………………………………………….17

Figure 9.The sequence of activities in client-server interaction…….18

Figure 10.Timing diagram of interaction between two players…....19

**Acknowledgements**

First of all we want to thank our advertiser Dr. Sufyan Samara and Dr.Alaa’ Al-Masri for providing us with some information about android and for their advisements in our project.

We would like to extend our thanks to our parents who gave us the motivation to accomplish this project.

**Abstraction**

Many developers develop applications that can run in single machine and they hope to make these applications available on different machines and interact between these two machines to make their final applications, but they don’t do that due to the difficulty of making the connection and the problems that result during attempting to establish this connection.

In our project we have developed a middleware that makes the interaction between two applications on different machines easy, and it takes care of making the connection secure and fast. In addition this middleware is responsible to make the synchronization between the applications that runs on different machines.

In our project we insist to make it more efficient by achieving the most important factor of being good middleware , we tried to reach the best levels of transparency by hiding the communication and the interactions between threads ,and we tried to make it more developer friendly ,they don’t have to do many things ,and also they don’t have to learn a lot of things to be able to use our middleware ,all things that they will do is to import our package and use our flexible functions.

The project is divided into two main parts, the first part use the Bluetooth to achieve the communication and the other part use the TCP protocol for communication. The first part is divided into three modules .The first module is used for scanning the available devices, the second module has the responsibility of establishing the connection and taking care of interactions and the last module is to interface with developers’ applications and they see only its functions.

1. **Introduction** 
   1. **Overview**

In our application we have developed a middleware project, this project is divided into two main parts; the first part uses middleware in centralized system and the other part uses middleware in distributed system. In the centralized system we used one machine (computer) as a coordinator that controls the distributed machines (android mobile phones) and this coordination is hidden in our middleware, and so our different mobiles interact with others using our middleware as they are in the same machine.

In the second part of the project (the distributed system) we used different machines with no coordinator and so each machine interacts with others as peer to peer interaction. In the two parts we tried to make the transparency one of the main objectives that we tried to achieve them.

In the project we used mobiles with android operating system and we used the eclipse program to implement this application, and it uses the Java language that is modified with new functions to be suitable for the android environment, we used this operating system instead of java to me due to many reasons that are explained in Related topics .

* 1. **Existing problems**

Many developers develop their applications in one machine and hope to make them run into different machines, but usually they don’t, and that is related to the difficulty of making communications between the machines and they want something that can make it easy for them to do so. The best way is to provide them with a middleware that can take care of the communication and hide it.

* 1. **Related works**

Different middleware projects have done before, some of them are very efficient and they simulate the perfect theoretical middleware concepts, and they have good results in huge systems, others have less satisfying results and these results due to many problems such as week synchronizations, less optimized code and less transparency (hiding more work and difficulties). In our project we tried to simulate the perfect middleware, but we have small bugs.

* 1. **Related topics**
     1. **Middleware in general**

Middleware is a class of software technologies designed to help manage the complexity and heterogeneity inherited in distributed systems. It is defined as a layer of software above the operating system but below the application programs that provide a common programming abstraction across a distributed system. In doing so, it provides a higher-level building block for programmers than Application Programming Interfaces (APIs) such as sockets that are provided by the operating system.

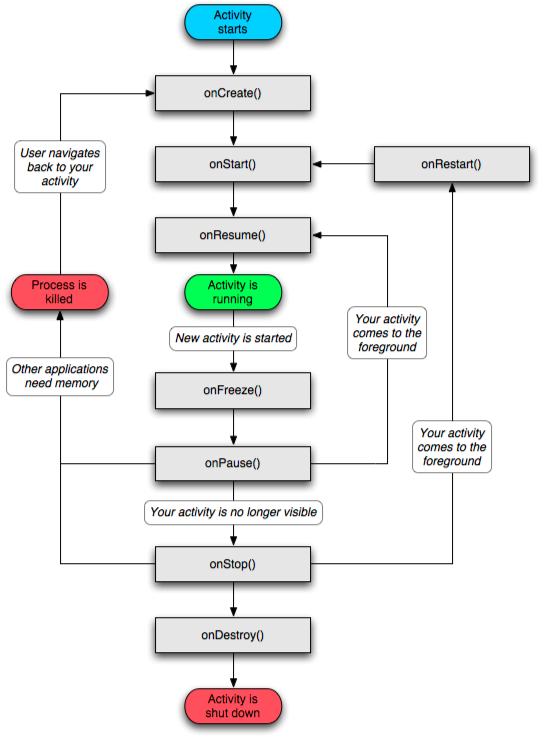
This significantly reduces the burden on application programmers by relieving them of this kind of tedious and error-prone programming. Middleware is sometimes informally called “plumbing” because it connects parts of a distributed application with data pipes and then passes data between them.

* + 1. **Android in general**

Android is operating system that is used in mobiles and it has more advantages than others, and it can be programed using special program which is eclipse which uses Java language and android has some features such as:

* Android language supports touch screen, Java2Me doesn’t.
* More flexible.
* The emulator is better than aJava2Me emulator.
* Android completeness.
* What you can do in Jav2Me you can do in Android.
* Android is growing.

The sequence of the methods that are called during the lifetime of an activity is very important point to understand the project because most of these methods have been used in our project, and some of them are critical for the project as the overriding method @onDestroy () and @onStop () and in the following flow char we will introduce on them.



**Figure 1.Flow of activities in android**

* 1. **Motivation for Carrying out project**

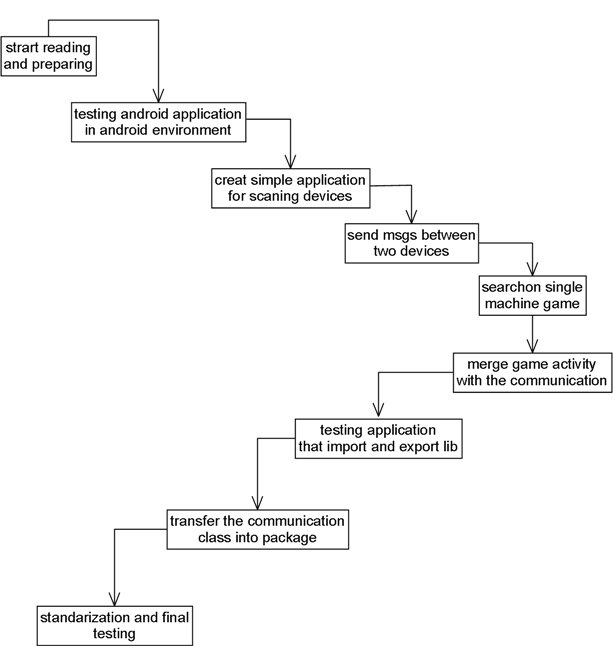
There are many aims of this project:

* Developing a middleware that can be used from developers in easy way
* To make it easy for applications to be run on different machines such as playing games on Bluetooth

And in our final test there will be a game (tic-tac-toe) that runs in different machines using Bluetooth.

1. **Methodology**

Here is a flow chart that illustrate how the sequence of the work was in the first part of the project (distributed system)

****

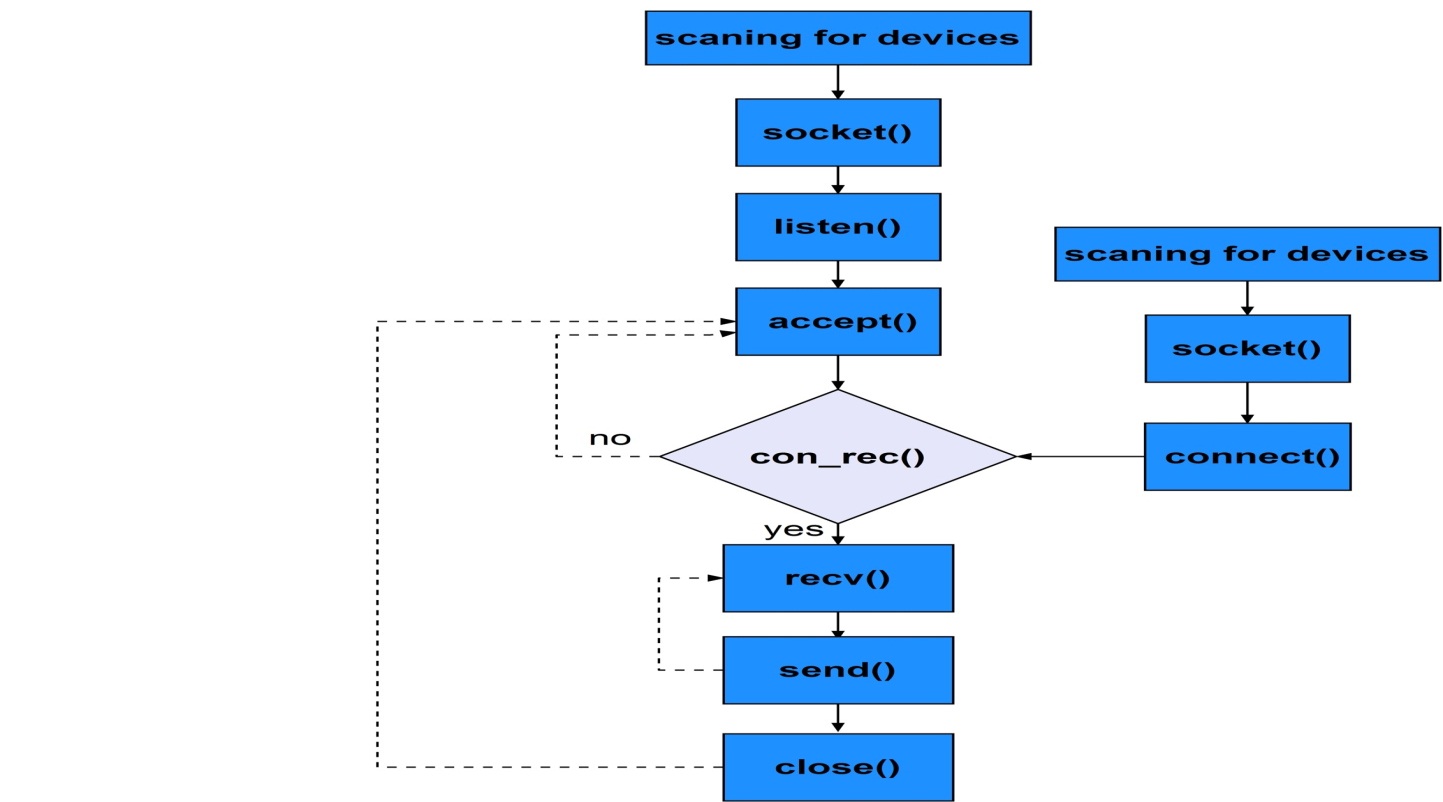
**Figure 2.Flow of stages of the work**

We divided the project into different stages, in the first stage we have learnt how to use the android operating system, how to install the android software, and how to use the emulator, as a result of this stage we were able to write programs and test them in emulator. In the second stage we have studied the requirements of the two main games’ categories; first shoot and strategy games and we found that message passing interface is the most efficient way for communications because sending pictures and sounds may increase the overhead of the application and slow it down.

In the third stage we began the programing part .First step in programing is to create a simple application on android to be familiar with this language and to test how the emulator work .Second step is to create a Bluetooth application that discovers the other active devices. The last step is to improve this application to create a connection and send messages between two mobiles. In this stage we faced many problems and difficulties due to the communications, one of them is that the emulator doesn’t support the Bluetooth communications, and so we had to make all the following tests on real devices .In addition, it was possible for any test to force close the device and make it not respond to interaction until we restart the device and this due to interactions between threads and sockets , for this problem we have searched a lot for the correct communications and the correct interactions between threads specially that android use Java language in which the threads is more difficult to use than other languages such as C# .

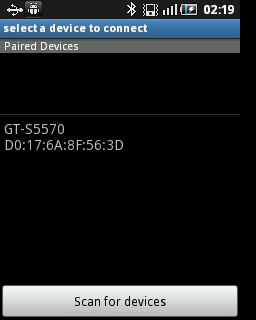
After long research we found that there is many different standardized procedures and techniques that we must use them in sequential manner, for example inside the implementation of the overriding method onDestroy (); we have to close the input stream first then the output stream then the sockets that are used from the threads by using the close(); method and finally we have to stop the thread using thread.stop(); method and we advise the programmers who try to use communications in their applications to use these standardized methods .

As a result of this stage we have a correct interaction between two peers, and that done by different procedures sequentially that flows as the following flowchart



**Figure 3.Communication between two peers**

From the flowchart we noticed that the application tell the two mobiles to scan for the devices then show the paired devices and make the ability to scan for other devices and the application knows about paired devices by using method getBondedDevices () that is implemented inside Bluetooth adapter class and we implement this to be shown as dialog as the following picture



**Figure 4.Dialog for selecting device to connect with**

We noticed from the picture that we have a paired device with name GS-S5570 with mac address 17:6A:8F:56:3D the user can choose the previous paired devices or scan for others. And we achieve this by using the broadcast receiver class that listens for the broadcast messages that is sent from the Bluetooth adapter when it finds some devices that are discoverable or when the discovery is finished.

The problem that encountered us in this part is that to show the view of dialog of the previous class we have to launch it as an activity not as class and the object of the activity is not useful here because this doesn’t provide us with the ability of changing the view .After search about this problem we have found that to launch a different activity from your main activity you have to use startActivity(Intent intent); and in order to get the result from this activity we have to use instead of that startActivityForResult(Intent intent,int request\_code); that starts the activity that is determined by the content of the intent and the result will be set in the called activity by using setResult(Intent intent); and then the overriding method onActivityResult will be called in the calling activity.

After this step of choosing the device the connection will be done as shown in the previous flowchart. First ,client establish a connection with the server by using socket.accept(); and it blocks here until the server perform socket.connect(); then it enters the connected state that repeatedly try to read from the input stream and put them on ***receive buffer*** and at the same time it can handle the requests for doing different things as interaction with the view or sending data to the server (not blocking) and all that is done by using threads and to handle that we make three main classes for connection that extends threads the first one is for handling the attempting to connect and the second that handles the acceptance and those two threads pass their sockets to the third thread (connected thread) then they stop to minimize the overhead on the application .

In the connected thread we can handle the incoming and outgoing messages by putting the thread in wait state to handle the read signals while the socket is connected and signals error if a problem with this socket happens, and to make it possible for sending messages we make a copy of this thread.

As a result of this stage which takes long time, we have a perfect implemented application that takes care of the scanning for devices and communications.

In the fourth stage we tried to find some game that runs on a single machine which can be run on different machines (two sides of playing) we chose the tic-tac-toe game which we found its implementation and we began to analysis this implementation and understand it .This implementation has two main classes one that extends the activity which acts as the main activity and the second class extends the View class.

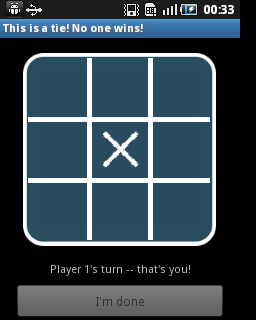
In this stage we don’t have real problems only we need some time to analysis this game and to find the control points that control the flowing of the game. And as a result of this stage we have a good implemented game on a single machine that have two sides of playing the first is the user and the second is the computer and we have made some changes of the implementation to make it applicable to the different stages.

The stages continue with a new stage that tries to make a game that can be played in two different machines. This stage is really the most difficult part; it takes most of the time .In this stage we had to make some synchronization between the two machines, as an example if one player plays on a machine, the other must be unable to play, and so we have added some new control states that control the flow of the game and the communication. In this stage we encountered many problems ,some that appears due to interactions between the communication and the game ,others appears as a result of wrong understanding of some critical points of the game implementation which forced us to return back to the previous stage for more understanding.

In this stage we have to transfer the playing to messages that can be sent, in our example (tic-tac-toe) is the cell’s choosing and it is put in a byte that is put in array of bytes of size 1 that transferred to the different part using the communications classes that we talked about previously .In the second side the message will be understood and it will be interpreted as it is in the same machine and the user pressed the selected cell.

In the sent messages we don’t have to add the resource or the destination of the message when passing it; because we have a well-defined connection that uses sockets as TCP communications .And before sending the message the other player isn’t able to play, and to make this we added states and when the player finishes his work he presses the done button and inside OnClickListener we call the selectTurn(State player); method that changes the player(state) and checks if the player is player2 then it disables the view and so the first player can’t do anything of the view and this is achieved in android by using gameView.setEnabled(false); and if the player is player1 we enable the view.

Here is the picture of the view:

**Figure 5.Synchronization between two players**

We notice from the picture that the view is disabled in one player’s machine and is enabled in the other’s. In addition we notice that there is a button called done that we have discussed its work previously.

In this stage we have some problems during testing that is usually caused from the longtime of waiting for result from the sender and this is caused from wrong implementation and we have followed the code to fix this problem because forcing close the application leads to make the device not respond until it is restarted due to the sockets and the stream that are opened, and application has no time to close them even they are implemented in the overriding methods onDestroy (); and onStop ();.

As a result of this stage we have an application on which there is a real game that can be played on different machines using the connection classes that we have developed but that classes are in the same project (not imported).

In the fifth stage we have searched about how to make packages and we found different ways, the best way is using export and import techniques to export the project to package.jar file which is a compressed file, and in this way the developers of the android put their packages that can be imported then from any project.

This stage is depending on search and tries more than any other stage .We spent more time on search to make packages.jar. First of all we tried to compress the folder that contains the classes using one of the compressing programs as WinRAR to .jar format and using this technique provides us the ability of access the functions on the compilation stage and no syntax or compilation errors, but during executing the application a runtime exception will be thrown and force close.

After long search we found that to inform the android that we will use this library we shall go through different steps:

1. Export the project to .jar file.
2. Import it in the project that will use them as library (using wizard).
3. Import the library in the code.

More illustrations in the appendix 3.

As a result we can use the functions of the imported libraries and the android can find the activities even in runtime during executing the application.

To update the content of the library first we had exported the library and imported it again and this made the update difficult and slow, we tried to decompress the .jar file and then recompress it, and this was destroying the .jar file, finally we found that to update the content of the .jar file without destroying it we have to add the files to the .jar file without decompressing it. And the steps are illustrated in the appendix 3.

As a result of this stage we have an application that use the libraries that are in a package that can be imported , and we have the ability of updating the content of this package easy.

In the final stage regarding the first part of the project (middle ware using Bluetooth communications) we have done the integration of the final output from this part from the project, then we have tested the project in this playing game.

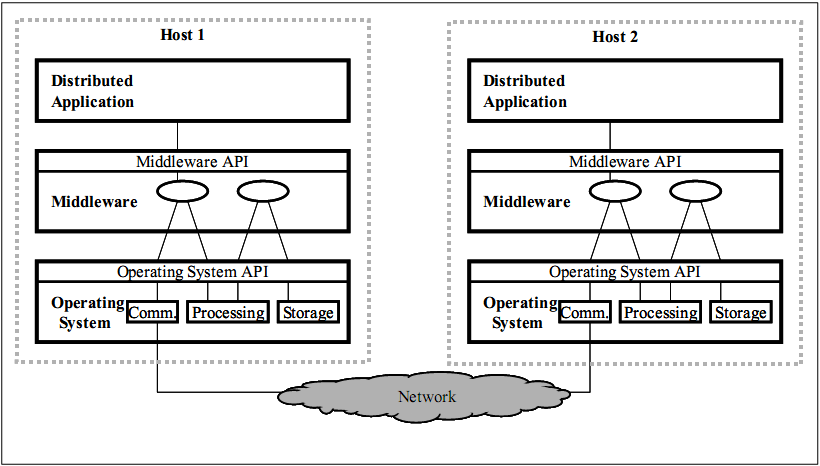
The first step of this stage is to make the names of classes and the functions standardized and to support the functions that the programmer can use in one class that interfaces with the developers’ applications and we name this class as MiddleWareStart and this name indicates that this class will start the communications and launch other activities. And we provide the programmers two main functions that are used to send and receive messages, these two functions are int MiddleSend (byte [] send, byte [] receive); and int MiddleReceive (byte [] receive); and how using these two functions with examples is illustrated in the help in the appendix.

The second step is to make all work of communication done though one function in the main class MiddleWareStart and we named it start (); and this step is more difficult than the first step because it needs to perform difficult operations; first it will launch the DeviceListActivity and this is difficult because trying to use startActivityForResult (Intent intent, int request\_code); from inside the middleware can’t be used because the application throws a null pointer exception and force close the application , and we spent a lot of time trying to solve this problem and searching for many days in the internet and asking the developers and no good results.

We tried many techniques one of them is to pass the activity from the main application to the MiddleWareStart class (which is we have to do for other functions) and then use activityFromMain.startActivityForResult this solves the runtime error but causes another problem is that the overriding method onActivityResult is never called. We spent a lot of time trying different ways but no results. Finally, we reached the result by using BroadcastSender and BroadcastReceiver and this solved the problem.

As a result of this stage, the first part of the project is done, many problems have encountered us during work and they have been resolved.

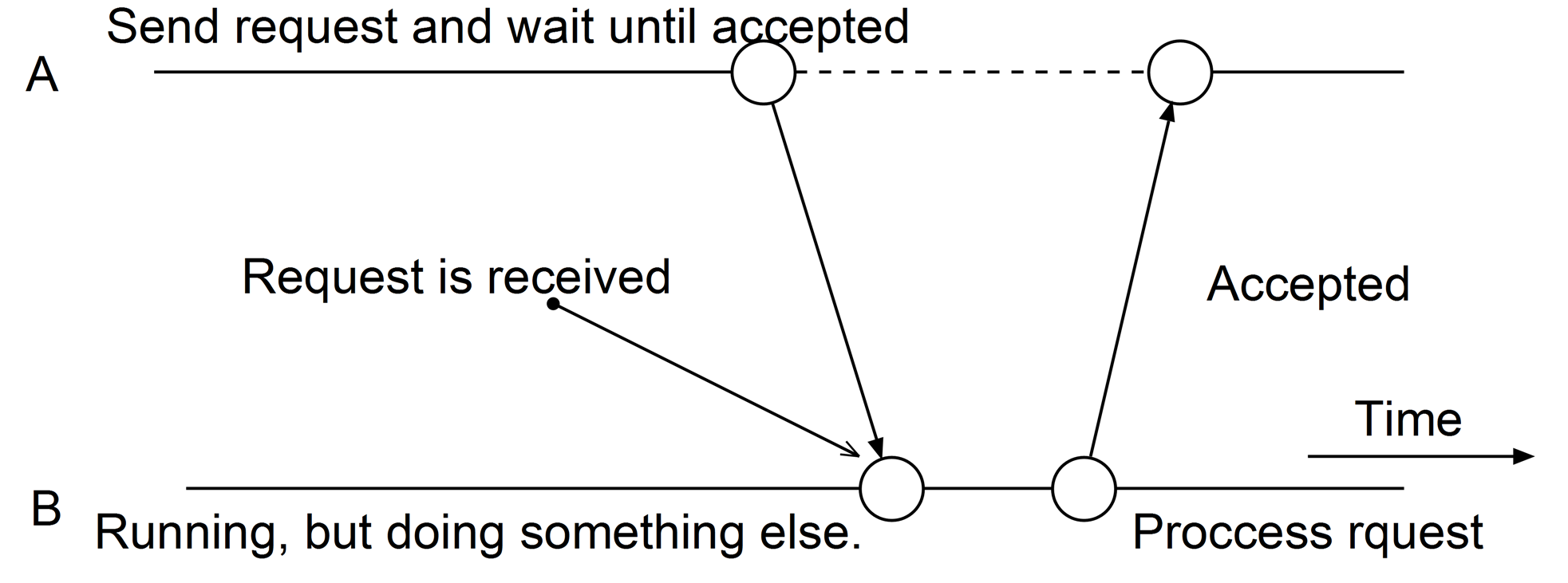
Here is the a figure that shows how the final application will be as in our system



**Figure 6.Middleware layer in context**

The final middleware that we have use message passing interface with two types of messages, the first one is point to point messages and the second is broadcast messages, and the type of communication is transient and synchronous, which means that message is only stored by system while sender and receiver are executing and that’s the object of communication during playing.

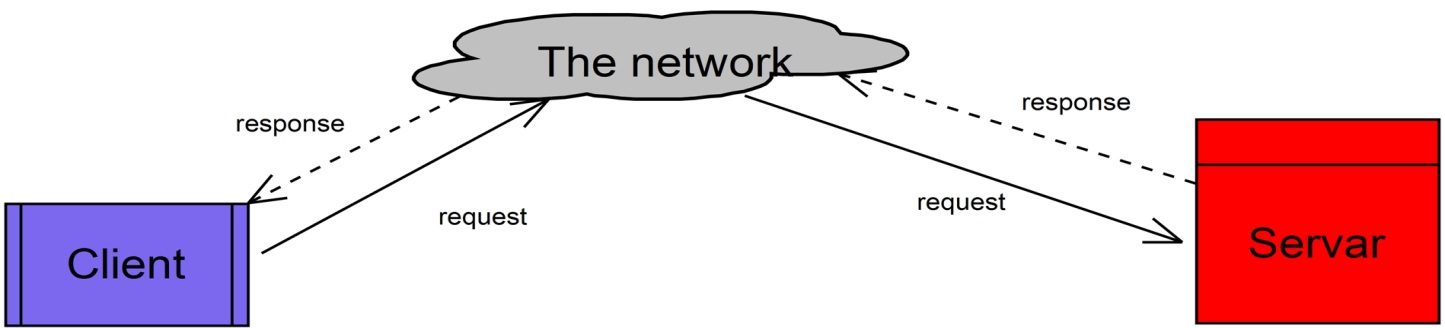
Our communication is Delivery-based transient synchronous communication at message delivery, and so we don’t have to block until response, and instead the broadcast message inform the activity that the response have reached and this will enable the view. As a result, the communication will be more flexible. And here the graph that illustrate the communication:



**Figure 7.transient synchronous communication**

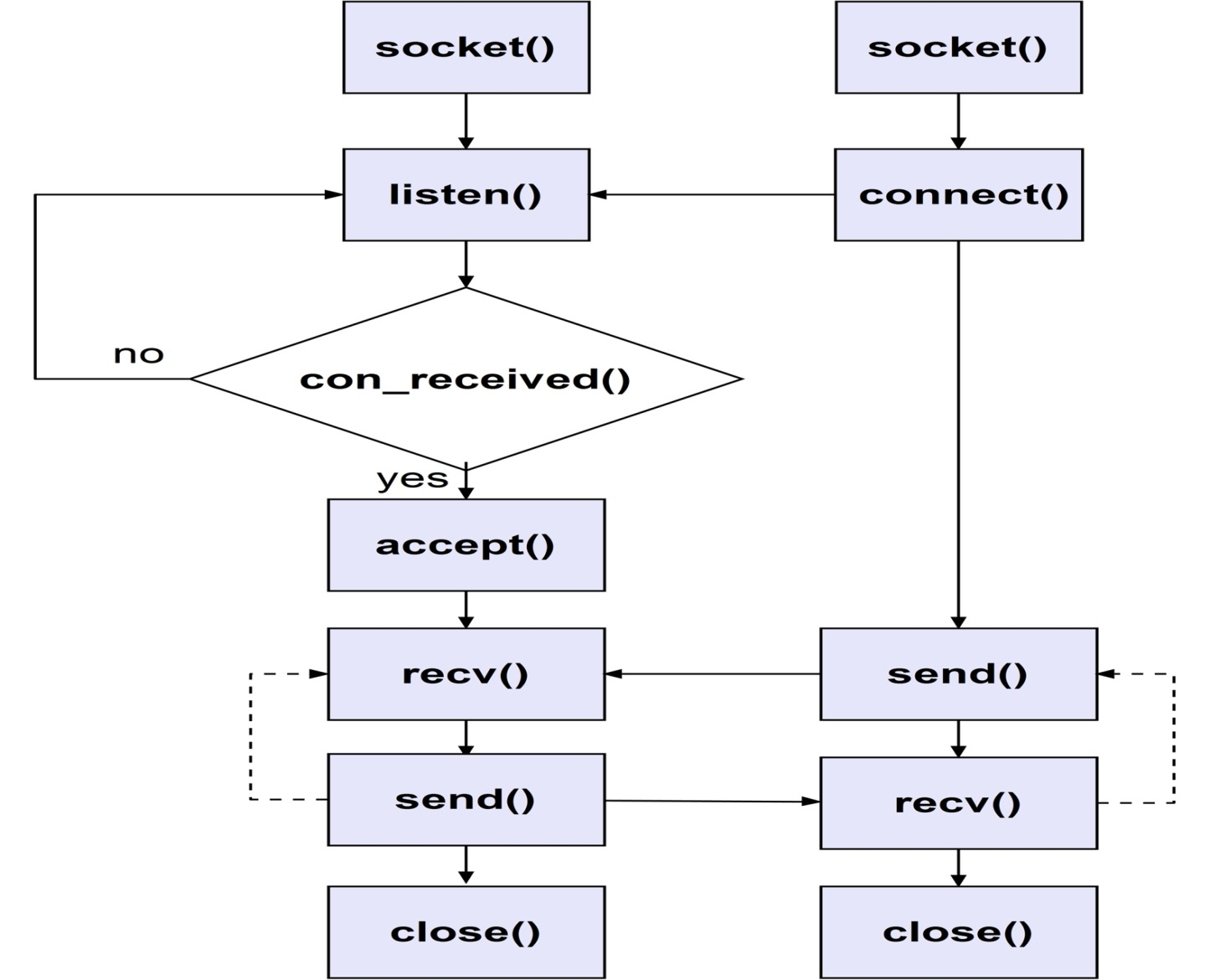
Now we will talk about the second part of the project.

The first stage in this part is to make a simple application that connect a client to server and do the interaction between them, in this step we have implemented the server side in c# and the client in java (android language) and the interaction is shown in the following figure



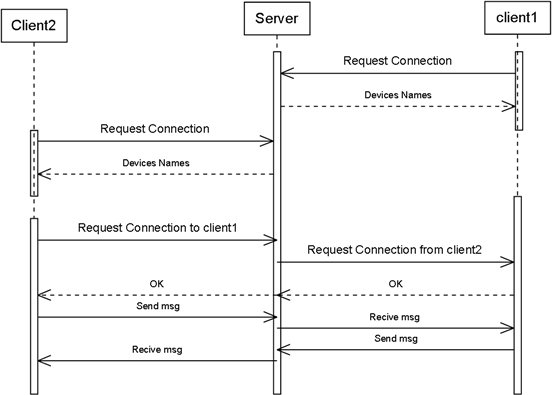
**Figure 8.Client-server interaction**

As shown from the figure, the client first requests a connection from the server and then the server will reply to that request and the flow of the actions will be as the following flowchart



**Figure 9.The sequence of activities in client-server interaction**

As a result of this stage we have a good client-server interaction. In the third stage we have developed this application and make client-server-client interaction application and the interaction is shown in the following timing diagram:



**Figure 10.Timing diagram of interaction between two players**

As a result of this stage we have good client-server-client communication. In the fourth stage we hide the sever side in our middleware to act as peer to peer communication. As a final stage of this part we have done the packaging to be available for use by developers and programmers.

1. **Result and Discussion**

In our project we have developed a middleware application that has two versions one acts in centralized systems and the other in the distributed systems, the first one use two android mobiles that are connected together using our middleware that depends on Bluetooth communication. In this part we have developed an application that uses our middleware for testing, in this application two players play using Bluetooth communication (tic-tac-toe).

In the second part of the project we have used a server (one computer) as a coordinator and many paired devices as machines in a centralized system, every two machines (mobile phones) play together as two peers and the our middleware hide the server and so the two machines acts as two peers.

1. **Conclusions and Recommendations**
   1. **Conclusions**

* There are two types of systems that can connect different machines together, one is centralized and the other is distributed system
* The best middle wares have good transparency that hide many attributes and in our project we hide the complexity and the communication between the machines and they acts as in the same machine.
  1. **Recommendations**

For the developers that want to improve our project, we advise them to create other parts that support the communications with Wi-Fi, and with computers (not only mobiles).Also we want to advice developers to use the java language that is modified to be suitable for android environment and finally to reduce the locking calls in the application as they can.

1. **References**

* [www.google.com](http://www.google.com)
* [www.codeproject.com](http://www.codeproject.com)
* [developer.android.com](http://www.codeproject.com)
* [www.stackoverflow.com](http://www.stackoverflow.com)

1. **Appendix**

* Our code project on CD**.**
* The middleware help on CD.
* Packaging help on CD.