RSC - Remote System Controller

Donfack Kana A.F
Department of Mathematics,
Ahmadu Bello University, Zaria, Nigeria.
e-mail: donfackkana {at} gmail.com

Madadjim Roland
Department of Mathematics,
Ahmadu Bello University, Zaria, Nigeria.

Abstract- **The advancement in mobile technology is fast changing the traditional way of computing. Several tasks which were previously performed only on personal computers are now possible with mobile devices. This paper presents RSC, a remote system controller, which is an application to control a remote computer through java enabled mobile devices such as mobile phone. Basic computer operations such as rebooting, shutting down a remote computer and file transfer from a computer to a mobile device are implemented.**

Keywords-Mobile application, Remote Computer control, java networking.

I. INTRODUCTION

Since the advent of mobile phone, (the first experimental hand-held in 1973 by Dr Martin Cooper and first hand-held to be commercially available in 1983 [12]) mobile technology has not stopped growing, exploring new dimension. According to [6] mobile phones are no longer mean for only voice data communication, but now-a-days the scenario has changed and voice communication is just one aspect of a mobile phone. Day to day, new technology are incorporated in mobile phones but are less useful in the hands of the users due to lack of applications that will make use of these resources. Handheld devices such as a mobile phone require certain applications and services for their efficient utilization. Since mobile phones, as the name indicates are mobile in nature [10], it will therefore be one of the most efficient way for real time monitoring of non mobile application that require close monitoring. For example, a video surveillance system record can be accessed on the mobile device in real time. Hence there is need for developing applications to handle communication and data access between mobile and non-mobile devices. Having a mobile device that can be used to access data from a local system from far distance can be of great additional value to individuals and corporations.

Technological developments have enabled the creation of mobile devices with the technical features which were previously conceived only in personal computer (PC) architecture [2]. Those features can be exploited in order to achieve optimal interaction between the PC and the mobile device. The advancements in 3G technology and wireless communication bring the convenience usage of mobile devices on internet [2]. These advancements are being used to add more functionality on mobile devices. As a result, more applications are developed to feed the ever expanding mobile features.

This paper develops a Remote System Controller (RSC), which remotely controls a computer system on the internet or local network.

II. RELATED WORKS

Several applications have been developed for controlling a PC but limited in the type of service they rendered and in the kind of technology used. They include RDM+, TeamViewer, jrDesktop, PocketDroid, VNC among many others.

TeamViewer [11] is used for remote control, desktop sharing and file sharing between computers. The software operates with the Microsoft Windows, Mac OS X, Linux, iOS, and Android operating systems. Machine running TeamViewer can be accessed with a web browser. In the default configuration, TeamViewer uses one of the servers of TeamViewer.com to start the connection and the routing traffic between the local client and the remote host machine. However in 70% of the cases after the handshake a direct connection via UDP or TCP is established.

Virtual Network Computing (VNC) [7] is a graphical desktop sharing system that uses the RFB protocol (remote framebuffer) to remotely control another computer. It transmits the keyboard and mouse events from one computer to another, relaying the graphical screen updates back in the other direction, over a network. VNC is platform-independent.

PocketDroid [2] is an android application designed to control remote desktops. It uses JAVA for the server side application and Android for the client side application. PocketDroid is used for file sharing between PC and android devices, start and stop the applications installed the target PC.

jrDesktop [4] is a cross-platform software for remote desktop control, remote assistance and desktop sharing. It is useful for home networking, helpdesk, system administration and collaboration. It supports java 1.5 and works only on LAN. It uses UDP instead of RMI/TCP, share only a specific user-defined region.

Remote Desktop for Mobile RDM+ [8], is a communication tool that gives the ability to connect to a remote desktop computer through the mobile device and interact with it remotely. RDM+ enables picturing of the remote desktop on the screen of the mobile device and to perform different usual keyboard and mouse commands.

In view of the above applications among many others, the technology and languages used in designing these applications differ from one programmer to another; From android platform to java platform, C# or Visual Basic. Most of the applications developed using Java platform differed from the RSC in the protocols used and a third party, that is, the web for server hosting was used in those applications.

III. ARCHITECTURE OF RSC

RSC uses java connection-oriented sockets and system commands to achieve it functionality. Java socket is used in RSC on both the client and server application to provide compatible and efficient streams for the communication between the client and server. Transmission Control Protocol (TCP), type of socket was chosen to provide reliable, bidirectional, point-to-point, and stream-based connection between hosts. For the implementation of the system commands, both the internal and external commands were used in the RSC system. The internal commands are used to manipulate the files while the external commands are used to perform the computer operations such as shutdown and reboot.

The RSC system is made up of a client-side RSC, which runs on the mobile device with java enabled capability, and a server-side RSC that runs on the PC. The communication link between these two parts of the RSC uses TCP. Fig.1 describes the relationship and operation of the two parts of the RSC.

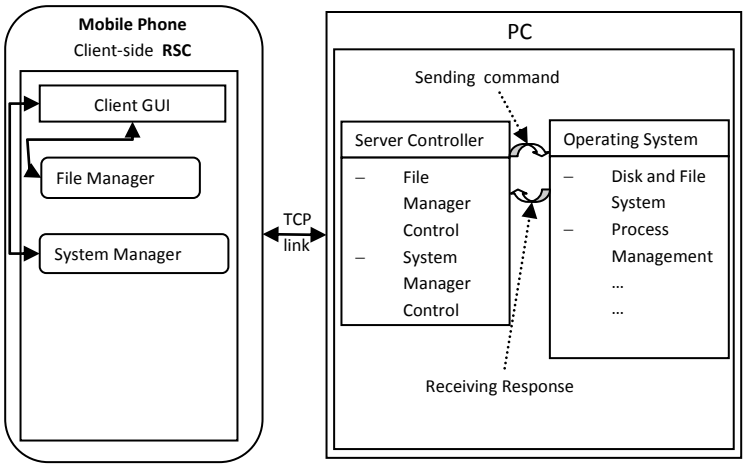

Figure 1: Basic Architecture of RSC

A. Client-side of RSC

The client-side of the RSC is divided into two modules as shown in figure2: the file manager module, responsible for files transfer and manipulation, and the system manager module, responsible for performing some basic system operations of the remote computer.

- File Manager Module

This module is responsible for sending various requests to the server for files transfer. The operations in this module are dependent of each other. Firstly, the list of all disk drives found on the remote computer is requested, followed by subsequent listing of the contents of selected disk drives. Operations such as copying or deleting a file can be performed on the files.

- System Manager Module

This module controls the remote computer itself by accessing its resources. Commands are sent directly to be executed by the PC, they comprise shutting down instructions, rebooting instructions etc...

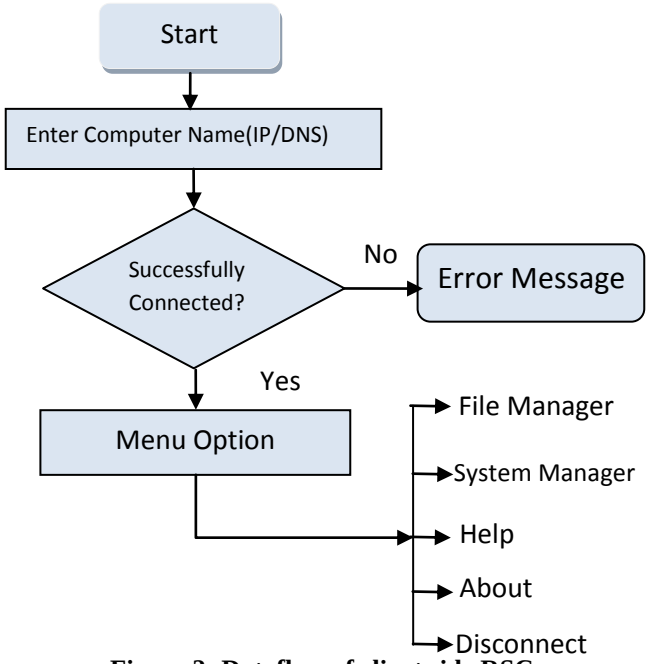

Figure 2: Dataflow of client side RSC

B. Server-side of RSC

This module, which resides on the computer to be controlled. It is responsible for receiving commands and requests from the mobile side RSC and executes them. The results are sent back to the client application. All processing are performed at this module, making RSC a thin-client model.

The communication in the RSC uses the TCP/IP. According to [3] TCP is relatively application-oriented in that using its socket facilities provides applications with a bi-directional byte stream between two hosts located at application endpoints. A connection-oriented service is best for applications that require characters to be received in the same order in which they were sent, such as keystrokes typed from a terminal or bytes in an ASCII file transfer [3].

IV IMPLEMENTATION AND RESULT

The RSC system involves communication between a mobile device and a remote computer, which is based on socket programming. RSC is implemented using Java. The choice of java is due it features. Java is platform independent. Because of Java flexibility and robustness, it provides the best deal for the development of such applications. Java platform is widely used (Java platform covers more than 1/3 of mobile devices market). Java programs can be deployed on both mobile devices and non mobile devices, independent of host hardware and operation systems. Client-side and server-side Java applications can integrate seamlessly under consistent Application Programming Interface (API) designs. The server application of the RSC was designed using the Java 2 Standard Edition (J2SE) platform while the client application is designed using Java 2 Micro Edition (J2ME).

The mobile device running an RSC-client can connect to the remote computer through internet. The RSC-client opens a socket connection to pass required information and commands to the server which in turn executes them. It also receives the response from the server, thus acting like a mobile remote control.

A. Establishing Connection

The RSC-client needs to connect to the remote PC through its address which can be an IP address or a DNS name. If RSC is used behind a router, port forwarding should be enabled.

A graphical interface is provided for user to enter the address of the targeted PC. In the RSC, when provided with a valid target PC address, a connection is established between the mobile device and the remote computer on the provided address. A port opened on the remote PC will be listening for incoming data, which will be bound to the mobile device port and so enabling communication between them. Listing 1 and Listing 2 show the pseudo-code for the connection phase.

```
open a connection through IP on a given port
create inputstream and outputstream
send request on outputstream
read response on inputstream
```

Listing 1: Client's connection pseudo-code

```
create a socket
bind to a well-known port
place in passive mode
while(true){
    wait for client connection request
    establish a connection with client
    handle client request
create an inputstream and outputstream
  while(client write){
    read a client request on inputstream
    process request
    send a reply on outputstream
  }
close the client socket
}
close passive socket
```

Listing 2: Server's connection pseudo-code.

Upon a successful connection to the server, the mobile user is presented a GUI menu of the RSC where the user can fully interact with the system.

B. File Manager

The implementation of the file manager is to achieve effective file browser which will enable the user to view the remote computer data content and to perform basic manipulation on these data. Fig.3 shows a view of the file browser interfaces captured from a mobile phone simulator.

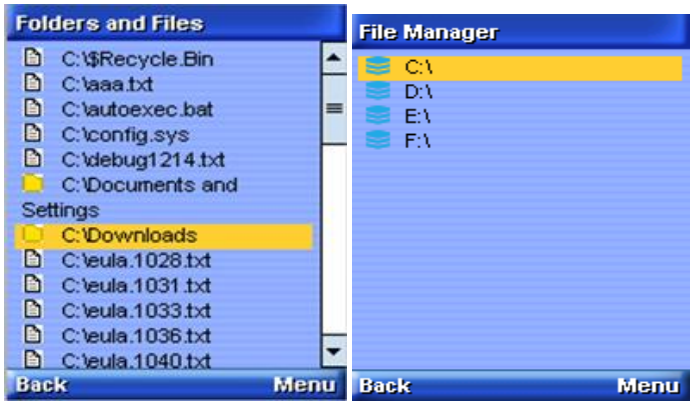

Fig. 3: File Browser Interfaces

This interface is implemented using the high-level Mobile Information Device Profile (MIDP)'s user interface components.

RSC provides the user with a total access to the disk drives of the remote computer with the possibilities of copying or deleting the stored files.

C. System Management

This function provides basic control of the remote computer such as shutting down and rebooting. The commands for the execution of this functionality are sent from the mobile device in a string format and are executed on the server-side program. Both the internal and external commands are sent from the mobile device to the PC.

The processing of the instructions sent from the client-side modules of the RSC are executed on the server-side. This makes the RSC a thin Client/Server application.

D. Server Controller

The server-side executes the instructions as they receive them form the mobile device. If the instruction received is of internal commands, it is routed by the server controller to the disk and file system management of the running operating system while instruction of external commands are routed to the process management of the running operating system. Responses obtained from the execution of these instructions are sent back to the controller which transmits them to the client-side program of the RSC.

The pseudo-code in Listing 3 shows the process used by the server-side program in executing the commands of the file manager module of the client-side. It lists and sends the list of all the drives to the mobile device through a port. Java File class is used in accessing the PC resources.

```
File[] roots ← File.listRoots();
        for( i ← 0; i<roots.length; i++){
            l ← roots[i].toString();
            writer.writeUTF(l);
        }
```

Listing 3: Server File listing pseudo-code

The pseudo-code in Listing 4 describes how a file is selected and transferred from the PC to the mobile device.

```
File listOfFiles ←new File(msg);
        File[] filesArray ← listOfFiles.listFiles();

        for( j←0;j< filesArray.length;j++){
            fileName ← filesArray [j].toString();
            writer.writeUTF(fileName);
        }

        File f ← new File(msg);
        if(f.exists()){
BufferedInputStream d←new BufferedInputStream(new
FileInputStream(f));
BufferedOutputStream outStream ← new
BufferedOutputStream(monClient.getOutputStream());
   byte buffer[]← new byte[size];
   int read;
   while((read ← d.read(buffer))!=-1){
      outStream.write(buffer, 0, read);
      outStream.flush();
    }
      d.close();
      monClient.close();
     }else{
      writer.writeUTF("File not found!!!");
      writer.flush();
     }
```

Listing 4:RSC transfer pseudo-code

V. CONCLUSION

In this paper, an application is developed to instruct a remote system from a mobile device. With the fast growing mobile technology and the capabilities of incorporating new applications into mobile devices, a remote system controller is developed to run on java enabled mobile devices. The application implements the basic system commands that instruct the computer to perform a specific task. Mobile device and remote computer here communicate through a direct connection under the control of the user without the need of a third party application or hosting site. This direct connection reduces the exposition of the system to high security risk. It is hoped that this will serve as a foundation for the

implementation of applications to enable a mobile user to use his mobile phone to control and monitor its non mobile applications through a personal computer from far distance.

ACKNOWLEDGMENT

The phone simulator used in this work was freely downloaded from Netbeans 7.2 integrated with jre 7 update 5. Available at http://www.oracle.com/technetwork/java/javase/downloads/jdk-7-netbeans-download-432126.html

REFERENCES

[1] Britton, Ch (2000) IT architectures and middleware: strategies for building large, integrated system. Addison-Wesley: Boston, 2000.

[2] Chaitali Navasare, Deepa Nagdev and Jai Shree (2012), "PocketDroid- A PC Remote Control", 2012 International Conference on Information and Network Technology (ICINT 2012) IPCSIT vol. 37 (2012) IACSIT Press, Singapore.

[3] Dick Steflick and Prashant Sridharan, (2000). Advanced Java Networking: Prentice Hall, pp. 49 – 53

[4] Java Remote Desktop – jrDesktop http://jrdesktop.sourceforge.net/

[5] Karl Kurbel, Andrez Dabkowski and Anna Maria Jankowska (2003), A multi tier architecture for mobile enterprise resource planning. *Wirtschaftsinformatik(1)* 75-94

[6] Kumar S., Qadeer M.A. and Gupta A. (2009), Location based service using android, In Proc. Of the 3rd IEEE Internet Multimedia Services Architecture and Applications 2009, Bangalore, India.

[7] Olivetti and Oracle Corporation (1997), Olivetti & Oracle Research Lab in Cambridge, United Kingdom

[8] RDM+ Remote Desktop for Mobiles http://www.rdmplus.com/

[9] Sameer Kul**k**arni, S. Diwan and N.K. Bansode (2004), Device Independent Mobile Application Controller For Remote Administration Of A Server Over A GPRS Link Using a J2ME Cellular Phone, In:IEEE India Annual Conference, INDICON 2004. IEEE, Los Alamitos .

[10] Shashi Kumar N.R., R Selvarani and PushpavathiT.P. (2008), "GPRS Based Intranet Remote Administration GIRA", *JRI- Journal of Research & Industry Volume 1 Issue 1 December 2008,*

[11] TeamViewer(2005) TeamViewer GmbH, http://www.teamviewer.com/en/index.aspx?pid=google.tv.s.int&gclid=CLvqpLjb77MCFSnJtAodbE8A_g Retrieved 2012-10-24.

[12] Wikipedia, the free encyclopedia, Mobile phone. [Online] Available: http://en.wikipedia.org/wiki/Mobile_phone (July, 2012)