1.SYNOPSIS

SYNOPSIS

The term Remote Desktop refers to a software or operating system feature that allows a personal computer's desktop environment to be run remotely on one system (usually a PC, but the concept applies equally to a server), while being displayed on a separate client device.

Virtual Network Computing (VNC) is a graphical desktop sharing system that uses the Remote Frame Buffer protocol (RFB) 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.

The concept of VMONKEY is make from the difficulty to use the power point presentations remotely. For moving to the next slide, we must return back to the computer. The simple solution to the problem is that WIFI Mouse or Bluetooth Mouse. But there must need a hardware for the purpose. VMONKEY solve that problem.

With VMONKEY we have extended the features of the current system. VMONKEY is an Android app which enables a user to access a computer android device from a remote location.

The main highlight of this application is that we can use the
system resources such as keyboard and mouse remotely.

The existing system does have any extending feature for using system resources (Here keyboard and mouse) from a long distance. With VMONKEY we have extended the scope of the above mentioned system. And at the same time additional features are provided for ensuring the performance to be stable. The application is designed to provide an easy to use interface. This application will take Android OS platform to a higher level with the help of Internet facility. With Internet we have extended the range of connectivity. You can literally access a remote computer/android device from around anywhere in the world

2. SYSTEM STUDY

2. SYSTEM STUDY

2.1. EXISTING SYSTEM

In recent years, there has been a phenomenal growth in mobile or handheld computing and communication devices such as mobile phones, personal digital assistants, personal media players and so on. The evolution of mobile devices, especially in these modern days, has drastically changed the face of business. It is now currently attainable with our knowledge infrastructure, and powerful mobile devices, for some individuals in doing most of their work outside the workplace. With the emergence of the mobile phones of nowadays, particularly Apple iPhone, Android, HTC and Blackberry products, individuals can work nearly anywhere. A lot of mobile applications have taken control over the mobile market trend. Every day, new mobile applications are developed with its own compatibility, making sure it serves purposefully to a particular mobile phone model and its specifications.

Nowadays, besides using computers for working purpose, most of the users use their computers for entertainment purposes such as watching a movie, sharing photos, browsing music or playing games. Sitting on a particular spot is never a fun especially when viewing an entertaining media. It can be very impractical to be confined to the keyboard and mouse while sitting 5 or 10 feet from the computer. Therefore, remote controls or wireless mouse and keyboards are also available for computers to solve the limitation mentioned above. However, these remote controls or wireless keyboard and mouse have a fixed set of buttons attached to the device it has been attached and designed to control. A fixed distance from the computer would be another concern of using the existing remote controls or wireless keyboard and mouse. When sitting far from the computer, the user is unable to view clearly the items on the computer screen, which would limit them in controlling the computer.

2.1.1 RELATED ISSUES

Android platform has made everything so easy and achievable in its environment. In line with a research from Gartner, Android is poised to become the second worldwide mobile operating system by the end of year 2011. There have been lots of competitions in **22** A Mobile-Based Computer Controller via Android.

The Android Market. The development and improvement of an existing Android innovation has become a usual thing now in the Android Environment. Every Android developer wants to design or create a unique application that has a lot of improvement and advancement towards the existing applications. A developer who has just built an application might end up seeing more of the application just in a couple of months. Due to the fact that Android is an open source, it has inevitably given Android developers the chance to improve existing work and as well create what is currently invoke in the market trend. As regards to this project's aim, there have been existing works related to the proposed application which has been in the market since last and earlier this year. Below are some of the related existing Android Remote applications, with a brief outlines of their specifications.

a. Gmote : This is an Android remote application that has been used to control a VideoLan Client (also known as VLC) media player, with basic options such as play, pause, stop, forward track and backward track. Its advanced feature is the file browser that allows it user to pick and choose what to play and also a recently added feature of being able to play some of the media files directly on the phone as opposed to watching it on the computer. However, this remote application has its limitation in controlling one program.

b. gPad: This application allows users to define custom remotes via key strokes. Its limitations are only by using key strokes and key combinations, so application functionality has been limited. The implementation of gPad does not support a two way communication, i.e. the sending of data back to a user.

2.2. PROPOSED SYSTEM

With the help of application which we has developed providing the facility of controlling an android device via desktop and the control of desktop through an android device. We has provided additional features such as screen emulation facility, media control etc. Thus with the help of developed app provides the facility of projection of the screen of a desktop over the android device and the screen of an android device over the desktop, so that the user can be able to sit anywhere he would like to where the Wi-Fi range is available and can control the desktop via android device and an android device via desktop

Fig. 1 shows the overall program flow for this application. When the remote control is run, it follows the path shown on the figure below.

After application starts, the embedded java application server runs in parallel. Sound notification is implemented in the proposed application so as to let users being aware that their IP address has been validated and the user can proceed with the application.

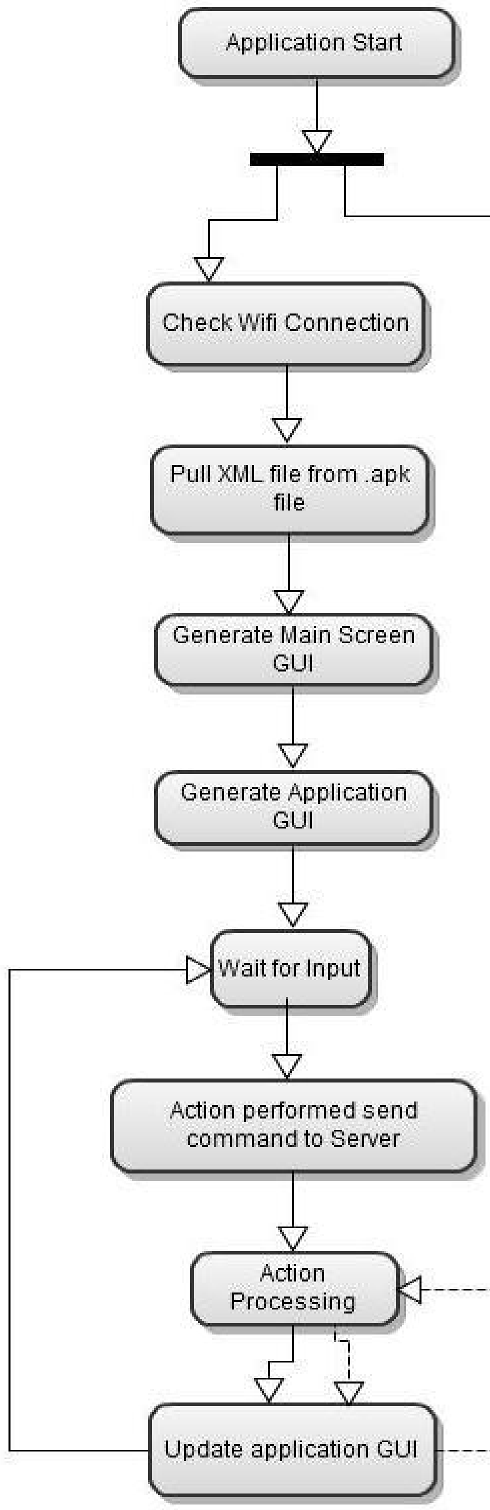


Fig 1

First, the application program starts by checking the Wi-Fi connection. The embedded web server runs in parallel once the application starts. The application pulls the xml file from the .apk file to generate the main screen and moves ahead to generate also the application Graphical User Interface (also known as GUI). The application GUI awaits an input from the user and when the action has been performed, it sends a command to the Web Server which then processes the action sent from the GUI. Then the Web Server resends back the action that has been processed to the Application GUI which then updates the input back to the user as expected.

2.2.1 ADVANTAGES

The main advantage of the application which we have developed is that it is providing facility of accessing system input resources remotely.

Another main advantage is that instead of using a WIFI for establishing connection among two devices we use the facility of internet connectivity. Thus it helps in providing a stable connectivity and provides a wide range communication.

Another advantage is that by the development of this application, it helps providing a maximum control over the system i.e. we can able to do almost all functions which can be performed over a desktop on an android device vice versa.

2.3. FEASIBILITY STUDY

Feasibility analysis is the procedure for identifying the candidate system, evaluating and electing the most feasible system. This is done by investigating the existing system in the area under investigation or generally ideas about a new system. It is a test of a system proposal according to its workability, impact on the organization, ability to meet user needs, and effective use of resources. The objective of feasibility study is not to solve the problem but to acquire a sense of its scope. Feasibility analysis involves 8 steps:

1.Form a project team and appoint a project leader.

2. Prepare system flow charts.

3.Enumerate potential candidate system.

4. Describe and identify characteristics of candidate systems.

5.Determine and evaluate performance and cost effectiveness of each candidate system.
6.Weight system performance and cost data.

7.Select the best candidate system.

8.Repair and report final project directive to management.

Three key considerations are involved in the feasibility analysis:

2.3.1. ECONOMIC FEASIBILITY

Economic analysis is the most frequently used method for evaluating the effectiveness of a candidate system. It is more commonly known as cost benefit analysis, the procedure to determine the benefits and saving that are expected from a candidate system and compare them with costs. If the benefits outweigh costs then a decision is made to design and implement the system. Otherwise make alterations in the proposed system.

2.3.2. TECHNICAL FEASIBILITY

The assessments of technical feasibility centers on the existing system and to what extent it can support the proposed addition. T[his was based on an outline design of system requirements in turns of inputs, files, programs, procedures, and staff. It involves financial considerations to accommodate technical enhancement.

2.3.3. OPERATIONAL FEASIBILITY

People are inherently resistant to change, and computers have been known to facilitate change. An estimate should be made about the reaction of the user staff towards the development of a computerized system. Computer installations have something to do with turnover, transfers and changes in job status. The introduction of a candidate system requires special effort to educate, sell and train the staff for conducting the business.

The candidate system was found to be technically, economically, and behaviorally feasible. The system was developed user friendly, needless training and improves the working environment. Justification for any capital outlay is that it will increase profit, reduce expenditure or improve the quality of service or goods, which in turn may be expected to provide increased profits. Disregarding the initial expenses, the candidate system was assessed to be feasible in all ways.

3. SYSTEM ANALYSIS AND DESIGN

SYSTEM ANALYSIS

SYSTEM ANALYSIS OVERVIEW

System analysis involves study of the current system in detail and to find out how it works and where the improvements have to be made. It also involves the detailed study of the various operations performed by the system and their relationship within and outside the system .The analyst and the user work in close association during the complete system analysis phase. System analysis is a phase that determines what is to be done for software development. The steps can be summarized as

- Identify the drawbacks of the existing system.
- Identify the need for conversion.
- Perform feasibility analysis.
- Identify hardware and software requirements.

3.1. BLOCK DIAGRAM

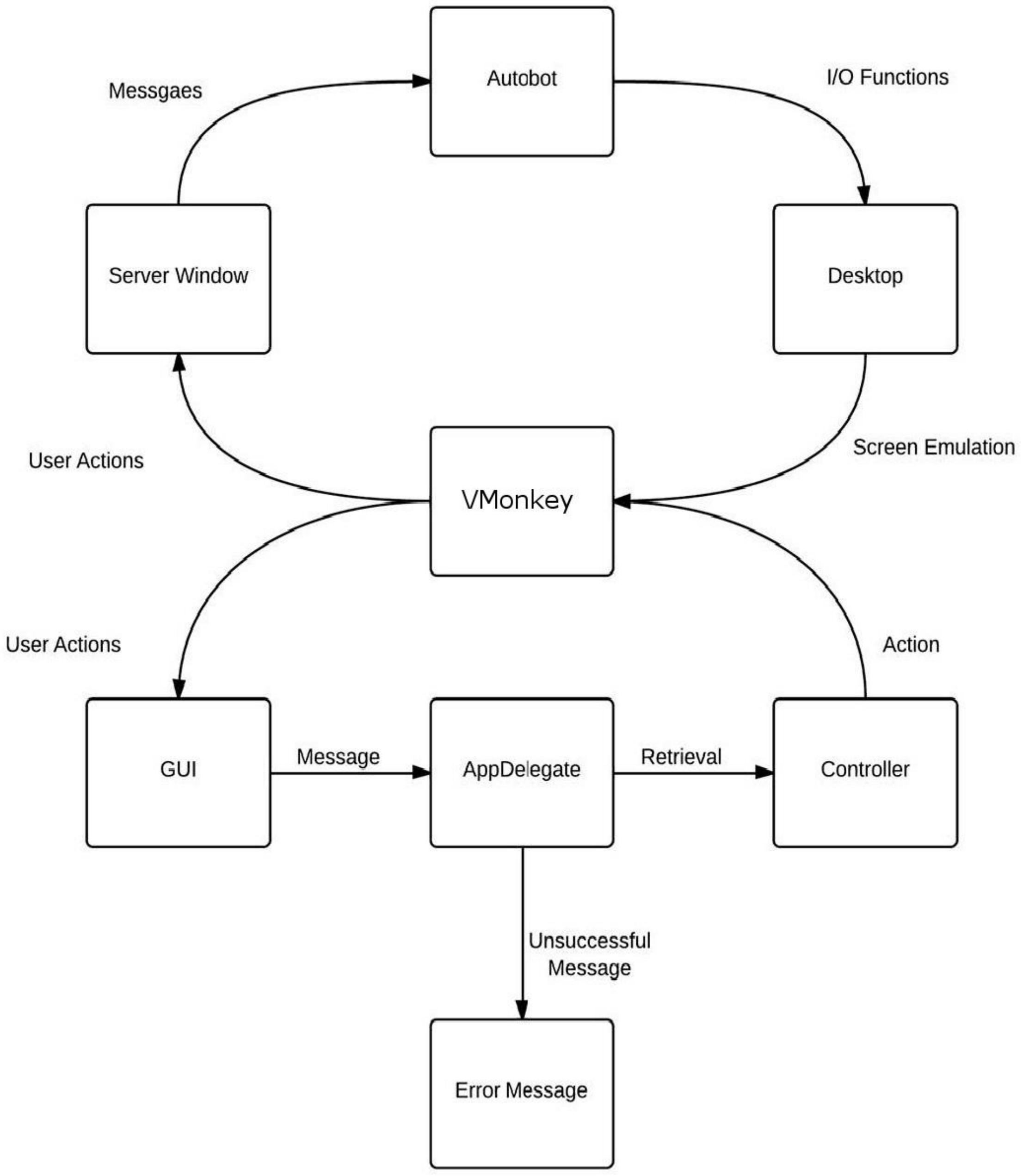


Figure 4.1 Remote Management System

3.1. DATA FLOW DIAGRAM

Data flow diagram is a graphical tool for structured analysis. It was Demacro (1979) who introduced DFD. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates output-data-flows which go to other processes or external entities or files. Data in files may also flow to processes as inputs.

DFD's can be hierarchically organized, which help in partitioning and analysing large systems. As a first step, one DFD can depict an entire system, which gives the system over view. It is called context diagram of level 0 DFD. The context diagram can be further expanded. The successive expansion of a DFD from the context diagram to those giving more details is known as levelling of DFD. Thus a top down approach is used, starting with an overview and then working out the details.

Basic DFD symbols

For constructing a DFD there are some symbols and notations to be considered. Primitive Symbols Used for Constructing DFD's is,

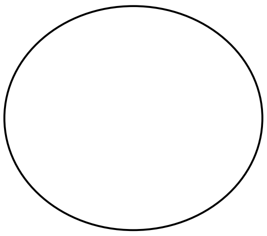
A data flow is a route, which enables packets of data to travel from one point to another.

Data to travel $\longrightarrow$ from one point to another.

Data may flow from a source to a processor and from data store or process. An arrow line depicts the flow, with arrowhead pointing in the direction of flow.

A process represents transformation where incoming data flows are changed into outgoing data flow. A function is represented using a circle. This symbol is called a process or a bubble.

Bubbles are denoted with the names of the corresponding functions.



A data store is a repository of data that is to be stored for use by one or more process may be as simple as buffer or queue or sophisticated simple as buffer or queue or sophisticated relational database. They should have clear names. If a process merely uses the content of store ant doesn't alter it, the arrow head goes only from the store to the process. If

a process alters the details in the store the double headed arrow is used.

Their names should be clear and understandable. simple as buffer or queue or sophisticated as relational database. They should have clear names. If a process merely uses the content of store ant doesn't alter it, the arrow head goes only from the store to the process. If a process alters the details in the store the double headed arrow is used. Their names should be clear and understandable.

A rectangle represents an external entity such as librarian,

library member etc. The external entities are essentially those physical entities which are external to the software system and interact with the system by inputting data to the system data to the system or by consuming the data produced by the system. A source or sink is a person or part of an organization which enters or receives information from the system, but is considered to be outside the contest of the dataflow model.

DFD-LEVEL 0

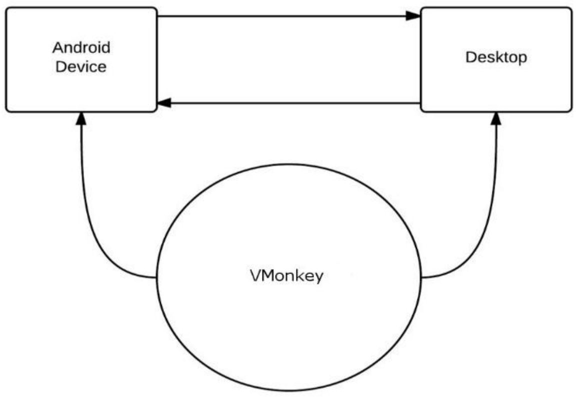


Figure 4.2 Level 0 DFD

Here we have our VMONKEY application which acts a interface between the android device and desktop. Using the application communication is established between the two.

DFD-LEVEL 1

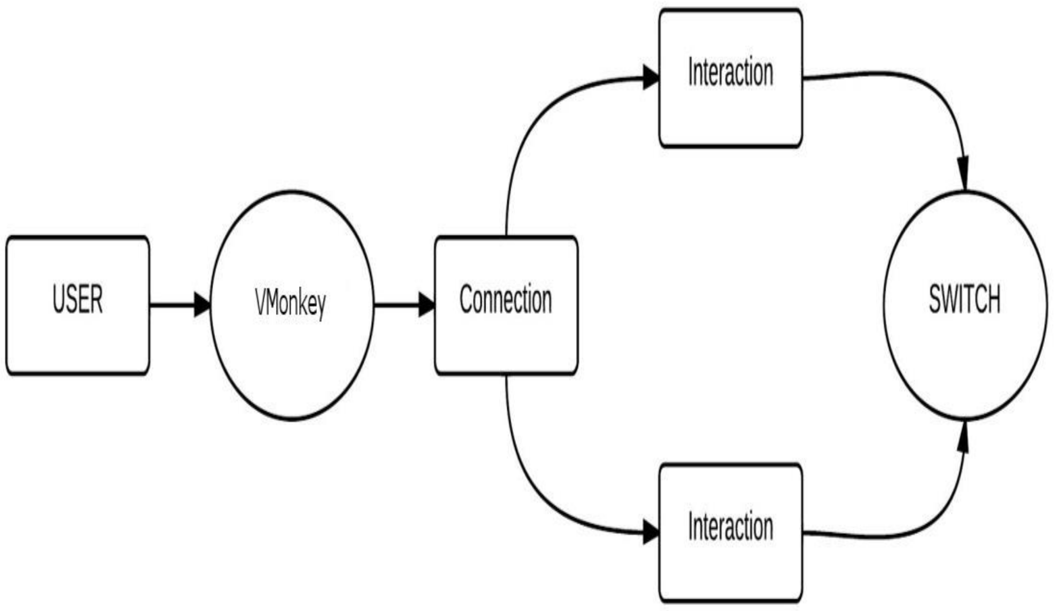


Figure 4.3 Level 1 DFD

Here we have a detailed look at the application. First VMONKEY is opened. And then the connection is established for the corresponding mode. ie, Remote Client or Remote Server. Once the connection is established the user interactions are carried out. These actions carried out will bring out the corresponding changes on the remote device/desktop.

DFD-LEVEL 2

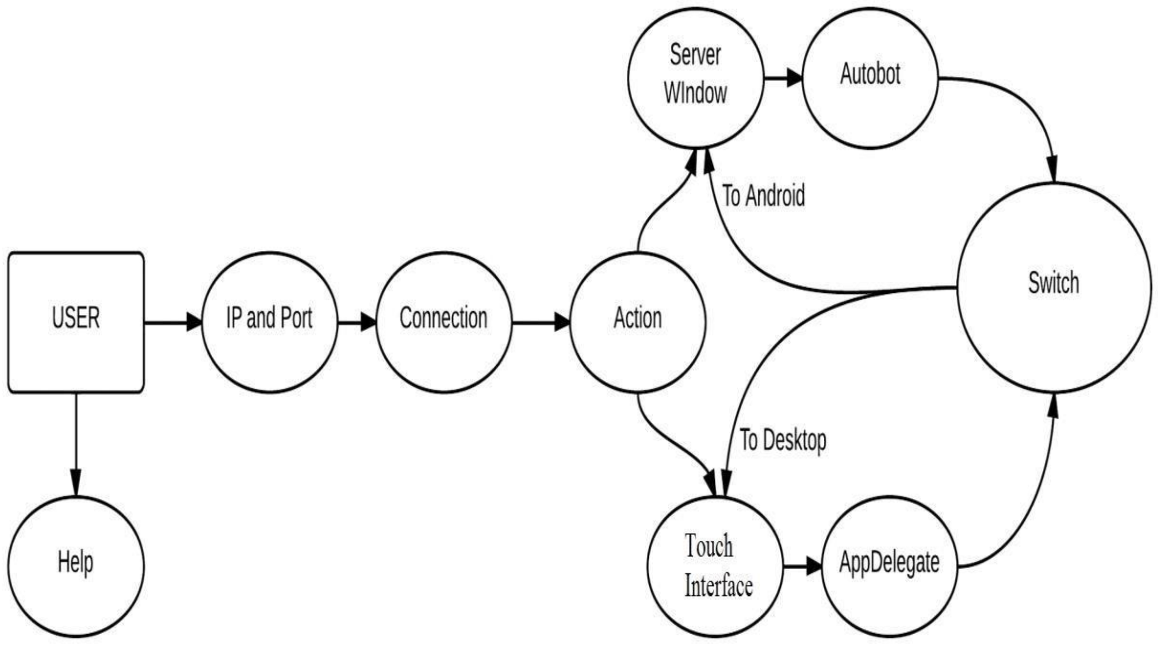


Figure 4.4 Level 2 DFD

Now let's take a deeper look at how the whole system works. Again we have a user at one end. The user interacts with a remotedevice/desktop using Remote Droid. First the user is asked for ip address and port number. Using those two fields the connections established. Once the connection is established user actions are carried out. When an action is carried out onto a android device through a desktop, the action is collected on the Server Window. The Autobot carries it out on the device. If an action is not carried out correctly the corresponding error message is displayed.

On the other hand when an action is carried out onto a desktop through an android device, the actions are carried out on a touch interface.And the actions carried out on the touch interface are collected on the app delegate which executes the input/output operations.Finally the switch section allows the user to switch connectivity mode anytime.

DESIGN OVERVIEW

The design phase is the next first for systems development life-cycle phases. It is the phase in which the detailed design of the system selected in the study phase is accomplished, and the user oriented performance specification is converted into a technical design specification. The principal activities performed during the design phase include the general system design, the design of all outputs, input design and the design of databases.

3.2. INPUT DESIGN

Inaccurate input data are most common cause of errors in data processing. Errors entered by data entry operators can be controlled by input design. Input design is the process of converting user oriented inputs to a computer-based format. Input data are collected and organized into groups of similar data.

The goal of designing input data is to make data entry easy, logical and free from errors as possible. In the design of input the following steps must be considered:

- The allocated space for each field
- Field sequence, which must match that in the source document
- The format in which data fields are entered

We have to keep in mind the following things to design the system:

- What data to input
- What medium to use

- The dialogue to guide users in providing input
- Methods for performing input validation and steps to follow when errors occur.
- Input design is a part of overall system design which requires very careful attention. Often the collection of input data is the most expensive part of the system, in terms of the equipment used; it is the point of most contacts for the users with the computer systems; and it is prone to error.

3.3. OUTPUT DESIGN

Outputs from computer system are required primarily to communicate the result of processing to users or sometimes to other systems, including machine based systems. They are also used to provide a permanent copy of these results for later considerations. These are various types of output required by most systems, the main ones are:

- *External outputs*

Whose destination is outside the organization and which require special attention because they project the image of the organization.

- *Internal outputs*

Whose destination is within the organization and which require careful design because they are the users among interface with the computers.

- *Operational outputs*

Whose use is purely with the computer department, example program listings, usage statistics etc.

- *Interactive outputs*

  This involves the user in communicating directly with the computer.

3.4. PROCESS DESIGN

Objectives

The purpose of the design projects is to provide us with an opportunity to place our course work knowledge into a process context.

Report format

Each team should prepare a single report documenting the design and presenting the results. A typical organization includes: introduction-describe the process, product and market, competitive processes and products. Pertinent literature or patent references should be included.

Design

The section should take one through each major unit operation, provide the information on which the unit was designed and sized. Important calculations can be included here.

Highlights

Provides scalable, cost effective, engineering-level architectural and design solutions designed to help optimize our IT system availability resilience. Delivers a customized technique that focuses on recommended practices, design analysis to help make sure we meet availability targets. Helps show us how can we improve our service availability so that we can reduce IT costs, enhance income as losses are produced, and optimize our environment by further integrating IT into business practices, includes prioritized plan of action to help us make informed cost versus benefit decisions about our availability needs.

4.SYSTEM SPECIFICATION

4.1 HARDWARE REQUIREMENTS

COMPUTER

Processor	: Intel I3, 2.9 GHz
Primary memory	: 2GB DRAM
Storage	: 80 GB Hard Disk
Mother Board	: ACER
Keyboard	: Standard 102 keys
Display	: VGA 15” colour monitor

ANDROID DEVICE

Processor	: 1GHz
Primary Memory	: 512MB
Android Version	: Ice Crème Sandwich and above
Connectivity	: WIFI

4.2. SOFTWARE REQUIREMENTS

Platform	: Windows 7 operating system or Above Operating System
IDE	: Eclipse, NetBeans
Front end	: Android API
Designing tool	: Eclipse
Server	: Java

- Client : Android

5.SYSTEM DESCRIPTION

5.SYSTEM DESCRIPTION

5.1. HARDWARE DESCRIPTION

VMONKEY is an android application which helps the user to control android device from desktop and desktop from android device. The software required to develop VMONKEY are java, Eclipse IDE, NetBeans IDE, SDK, NDK etc. In order to run this software in your desktop computer it should have minimum configuration of 2GHz dual core processors and 1GB ram. For the proper working of eclipse it need large amount of memory to load required data from Software Development (SDK), Native Development Kit (NDK) to the IDE. So for the smooth running of the software we advise you to have a higher configuration than the minimum. After developing the application it must be given to public. VMONKEY works on Android phone which have Ice Cream Sandwich or higher versions of Android. The minimum hardware Requirements for Android are minimum of 1GHz processor and 512 MB ram. For better performance we advise you to use a high speed internet like WIFI or 3G for establishing connection. Performance of the application is based on the hardware and the internet connection. If you have an android device with high configuration and a poor internet connection then the performance will be comparatively low.

5.2. SOFTWARE DESCRIPTION

Android is one of today's most popular terms in the world of mobile operation systems. Android phones and Android applications are so popular among people that they have established a secured position among users in the world of mobile phones. Android is the software stack for a mobile device that includes an operating system, middleware and key applications .It is a Linux based operating system. Android Inc. was founded in 2003 in Palo Alto, California, the United States in October, 2003 by Andy Rubin, Rich Miner and some other workers. In 2005 August Android Inc. was acquired by Google with the key employees of Android Inc. moving to Google. At Google Andy Rubin and his team developed a mobile device platform based on the Linux kernel known as Android and was introduced in 2007 when a group of industries came together to form Open Handset Alliance. Open Handset Alliance is an association of firms to develop open standards for mobile devices. Since then thousands of Android-based phones and applications had been launched to the market and have achieved tremendous success.

In the market many programming languages are available such as C, C++, PHP, Flash and Java. Among these available languages Android uses Java as a native program- Ming language. All the applications for Android are created using Java and then com- piled using the javac compiler. Those compiled files are then converted into Dalvik Executable (.dex) file and ran in Dalvik Virtual Machine. Dalvik Virtual Machine is a virtual machine just like in Java which is used to test and run applications. The Android applications developed using Java use Android Software Development Kit (SDK) whereas the applications developed using C/C++ programming languages use Android Native Development Kit (NDK).

The reason for Android to choose Java over other programming languages is because Java is an object oriented programming language, simple, secure, dynamic, portable, has high performance, supports multithreading and has automatic garbage collection. Besides Java C/C++, Flash can also be used to create Android applications.

The Android operating system can be divided into five major layers: Application, Application Framework, Libraries, Android Runtime and Linux kernel. These are the basic components that an Android application consists of. Applications is the top layer of the architecture. This is the layer where the core applications of a device such as phone calls, an email client, SMS program, calendar, maps, browser, contacts, and others can be found. These applications are written in Java and other languages. The Application Framework is the second layer of the architecture. This is the framework or the outline that a developer has to follow during application development. Developers are given full access to the same framework Application Programming Interface (API) as used by the Applications layer. This layer is just like a basic tool that can be used by a developer to develop much more complex applications.

The third layer of the architecture is libraries. It consists of sets of C/C++ libraries used by various components of the Android system. These libraries are available to a developer for use through Application Framework. Some of these libraries are Surface Man- ager, Media Framework and Web Kit. These libraries provide the facility to use in-built functions to carry out common tasks, or to reuse the functions for other purposes. The fourth layer in the architecture is the Android Runtime. It consists of sets of core libraries that are available in the Java language and Dalvik Virtual Machine. Dalvik Virtual Machine is software that behaves like a real device and is used for testing and running an application as its own process.

The last layer of the architecture is Linux Kernel. Android depends on Linux version 2.6 for the core system services such as memory management, security settings, power management software and several drivers for hardware, file system access, networking and inter-process communication. The Kernel also acts as an abstraction layer be- tween hardware and the rest of the software stack

As shown in figure 1, the five layers are the basic components required to form the Android operating system. Each layer has its own function and groups various pro- grams together to run specific functions of the operating system. The activity in Android is responsible for creating a window of an application. An application can have any number of activities. An activity in an Android application has its own life cycle. An activity lifecycle has the starting and the ending state with many other states in between. When an activity lifecycle method is called, there is a change in the state of an application which is handled by the application components.

An Android activity can be in four states. The states are the active/running state, pause and resume state, stop and restart state and finally dead/destroy state. The active/running state is the state where an activity has been started. An activity is currently active and is always at the foreground or at the top of the activity stack. The highest priority in the stack is always the currently running stack. Such an activity is only killed if it tries to cause an abnormal error to a device by occupying the excess memory. During the pause and resume state an activity is not active but is partially visible. This state is usually caused when a newly launched activity does not occupy the whole screen or is semi-transparent. This activity is still alive and has the second highest priority in the activity stack. An activity gets killed by Android if sufficient memory is not available for the application to be in other states. An activity resumes with the previous state when the semi-transparent activity finishes.

The stop and restart state is the state where an activity is completely invisible in the view. Another activity completely obstructs its view. The activity is still alive and does not lose its progress. In this state an activity has the lowest priority in the activity stack and can be killed by the operating system just to provide memory to the higher priority activity. If a user returns to an application by any means, then the activity restarts.

The dead/destroy is the final state of an activity. An activity in Android can be manually destroyed by a user by pressing the back button or by a system if required. An activity will try to reinstate its previous state but is usually destroyed by the Android system in order to free memory for other activities. All these all states form a lifecycle for an activity in Android.

Figure 2 shows how the lifecycle of an Android activity occurs.

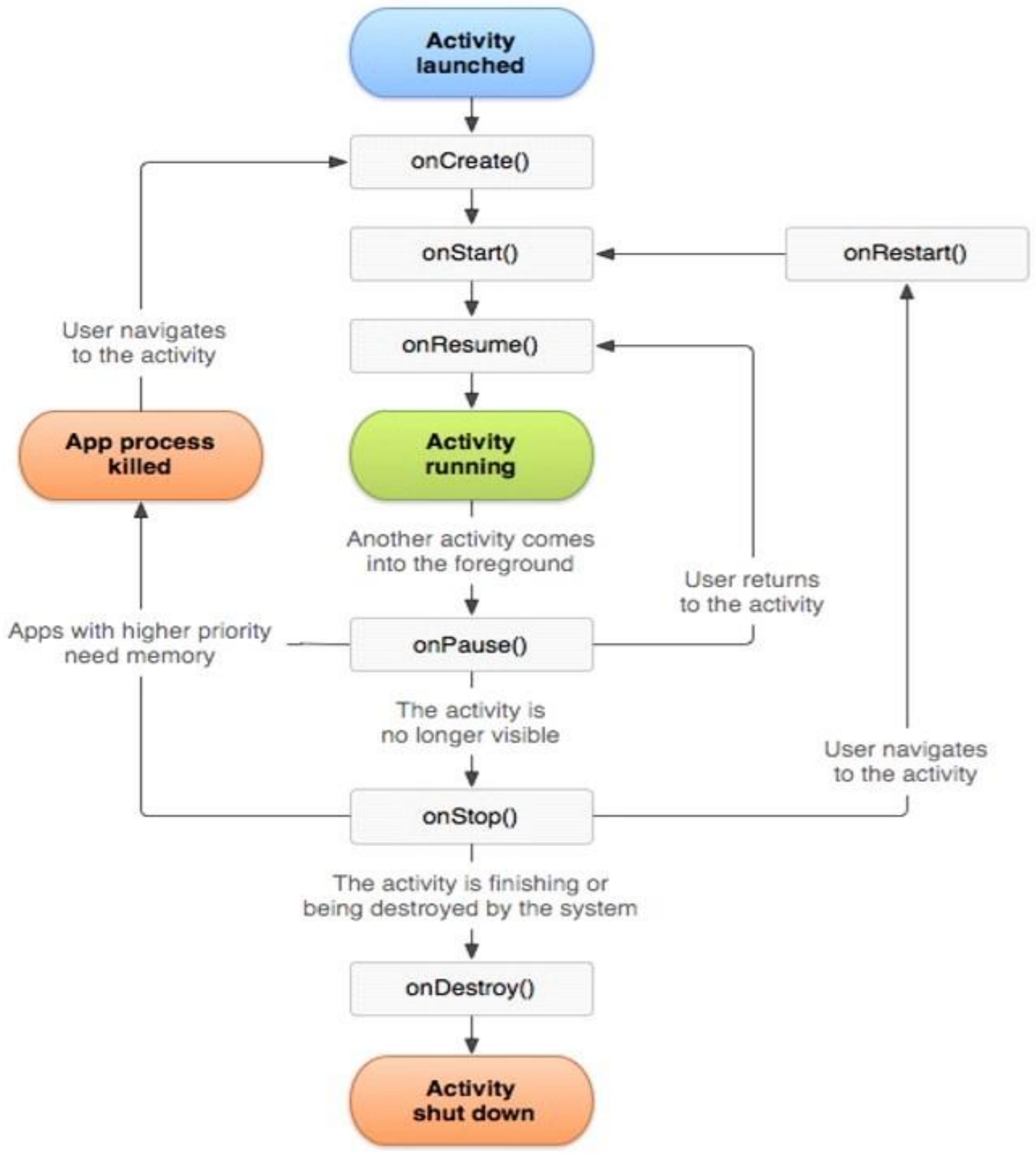

Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Figure 3.1 Android Life cycle

Figure shows how an activity is started, paused, resumed and destroyed in its lifecycle. Android provides methods that can be called to handle the changes when a state in an activity changes. As the activity is started, the OnCeate() method is called. This method initiates an activity and performs the tasks such as creating a view, reading layouts, initializing variables or binding data. Then the OnStart() method is called, which makes an activity visible .An activity is still inactive during the OnStart() call. An activity becomes visible in the view when the OnResume() method is called. During this call an activity becomes active and users can interact with an activity.

An activity is at the top of the activity stack during the OnResume() call. If the current activity is sent to the background by any means, then the OnPause() method is called. There are two possible scenarios during this call. An activity is either destroyed or resumed by a user. An activity can be destroyed by the operating system to free the memory. In this case an activity restarts from the OnCreate() method with the BundlesavedInstanceState parameter which store the previous state of an activity. The next scenario would be to close the new activity and resume an activity from the OnResume() method.

If none of the above mentioned cases occur, an activity is passed to the OnStop() method. This happens when a user has pressed the back button or a new activity has started that completely covers the previous one. There are three possible scenarios during this call. The system can kill an activity to free the resources and the cycle is started again from OnCreate(). Another case is that an activity is resumed from OnStart() when a user brings an activity from the background to the foreground. The final case is the OnDestroy() method is called and an activity is destroyed. This happens when an application is terminated by a user or the system. The entire life cycle of an activity is in between the OnCreate() and OnDestroy() methods. An activity is visible during the OnStart() and OnStop() methods. Finally an activity is in the foreground be- tween the OnPause() and OnResume() methods.

The Android SDK is a collection of software development kits used to develop applications in the Android platform. The Android SDK is available for free to download and use from the official Android developers’ website. The SDK is available for all operating systems. It can be used alone to develop an application with a command prompt or can be used with IDEs such as Eclipse and NetBeans. The IDE provides a graphical inter- face for easier and faster development of an application .

The Android SDK consists of the following features:

- Required libraries
- Debugger
- An emulator
- Documentation for the Android application program interfaces (APIs)
- Sample source code
- Tutorials for the Android operating System.

For each release of a new version of Android, a corresponding SDK is released. A separate SDK is available for a separate version of Android. In order to use the latest feature in

Android, developers need to install the latest SDK depending on the phone’s Android version. To develop an application Android provides a custom plug-in for

Eclipse called ADT. ADT provides an integrated environment for development of an application by extending the abilities of Eclipse. This helps developers to create an Android project, design an application user interface, debug an application and export and import an application package. The native language used for the Android application development is Java for which Java Development Kit (JDK) is needed. JDK can be downloaded for free from official Sun Microsystems website. All the above mentioned plug-ins and IDEs are available for free downloads, and complete sets of instructions on installation are available on the Android official website.

With more and more phone manufacturing companies evolving today, there has always been competition between vendors regarding the technology provided to customers.

The factors affecting the popularity of phones are their looks, performance, user experience, price and applications. One of the important factors affecting user experience is a user interface. The user interface is a component of an application that a user uses to interact with a device. Designing a better user interface and providing better technology to interact with the interface has been an essential task in designing phones. One of the ways of interacting with a user interface is using the touch screen. Touch screens can be used to feed inputs and to produce outputs. A simple and more intuitive interaction is provided to a user allowing direct and natural interaction with the devices.

The touch screen is an electronic visual display with which a user interacts by touching the screen. A human finger or a touch pen can be used to interact with the screen. It is fast, simple, easy and natural compared to other input devices, so many companies manufacturing electronic devices have acquired the technology, including Android. Several of issues must be kept in mind before designing User Interface (UI) for touch screens. Some of the basic principles to be followed during designing a UI for the touch screen are as follows:

- The UI should allow immediate access to the components.
- Gestures such as a tap, click or flicker should kept simple and smart.
- The UI should be designed for real world use (considering the size of finger)
- The UI should be kept simple and clear

Designing a UI in Android has its own techniques and components. Android uses Views, View Groups and Activities for designing a UI. Views are the class containing all the basic building blocks or component for a UI design. They contain all the UI controls, layouts and widgets. It is not always necessary to use the components provided by the Views class for the UI design. A new view or components can be created by sub classing the existing class. Views can be comprised of multiple views known as View Groups. A View Group itself is a subclass of Views and can contain multiple child Views. An activity is the screen displayed to the user. To display the UI, View is as- signed to an activity. Generally XML files are used to design a UI in Android applications. An XML file must contain a root layout and this root layout can contain many nested layouts and view

Most Android phones use the touch screen as the primary medium to interact with a user. The screen of a phone is used to interact with a user interface of an application. Android supports both a single touch and multi touch. Touch events in an Android phone are supported by view, custom views and an activity. Touches are interpreted in an Android phone in the form of gestures, such as swipe, flicker, scroll, fling or drag. Android has provided proper classes and methods to handle the touch events. Some of the classes and the methods that handle touch are MotionEvent, TouchEvent, GestureDetector and VelocityTracker.

To detect a touch in Android the MotionEvent class is passed to a view using the On- TouchEvent method. The MotionEvent class contains information such as coordinates of the touched point, the number of pointers, size and pressure of each pointer. The OnTocuhEvent method then can be used to use the points. The MotionEvent class provides some constants that can be used to determine the action caused by a touch. ACTION_DOWN detects a new touch, ACTION_MOVE detects finger moving on the screen and ACTION_UP detects the removal of the finger from the screen. The two constants ACTION_POINTER_DOWN for pointer down and ACTION_POINTER_UP for pointer up are used for detecting a multi touch.

The two other classes used to detect a touch are GestureDetector and VelocityTracker. GestureDetector must use GestureOverlayView in the view to detect gestures. The class detects gestures such as swipes and flings. This class can be used to draw figures and write by a moving finger on the screen. Likewise the VelocityTracker class can be used to track the velocity of the touch events. [19,604-619] this project also uses the touch feature of the phone. The touch screen and keyboard of the phone acts as a mouse and keyboard for the server simultaneously.

Eclipse

Eclipse is an open-source IDE donated to the community by International Business Machine (IBM). Java and the Android programs are written using Eclipse. Eclipse can also be used to program in other programming languages by means of external plug-in. Also the plugins can extend the usability of Eclipse. Eclipse is developed by an open source community which has about 2000 open source projects related to different aspects of software development. Syntax-highlighting editor, incremental code compilation, a thread-aware source-level debugger, a class navigator, a file/project manager, and interfaces to standard source control systems are the features provided by Eclipse, once combined with JDT.

Eclipse was started as project for IBM which later in 2001 was released as an open- source project. An association was then formed for further development of Eclipse as open source, and the original board members were Borland, IBM, Merant, QNX Soft- ware Systems, Rational Software, Red Hat, SusE, TogetherSoft and WebGain. Later in 2004 the Eclipse foundation was formed. It is a nonprofit organization responsible for governing the Eclipse projects. Figure 6 shows the basic window of the Eclipse.

The whole figure is known as the work bench window. The window consists of a package explorer view, a type hierarchy view, a Java outline view, wizards for creating the Java elements, a Java editor, a short cut bar, a menu bar and a tool bar, as shown in figure 6. The Package explorer shows the entire project that is currently in the workspace and the project highlighted in the package explorer is the project that is currently in use. The Type hierarchy view shows the sub and super hierarchies whereas the Java outline view shows the classes and the Java units. Wizards for creating the Java elements consist of the Java classes, packages and interfaces. The Java editor is where all the codes for application development are written. The codes can be highlighted, edited or fixed in this area. All the commands such as edit, run, build and others are included in the shortcut bar, menu bar and tool bar. To develop the client-side application of the project in the Android platform using the Java programming language, Eclipse was used.

NetBeans IDE

NetBeans is also an IDE used mainly for Java programming. Other languages such as JavaScript, PHP, Python, Ruby, Groovy, C, C++, Scala and Clojure can also be used to program in NetBeans. NetBeans is written in Java and is compatible with all the op- erating systems capable of running the Java Virtual Machine. Initially developed in 1996 by a student of Java IDE, NetBeans was later acquired by Sun Microsystems and was released as an open source project. NetBeans is available for free download from the NetBeans official website. The download includes documentation, sample projects, tools and a module to integrate with Java. Like Eclipse NetBeans functionality can also be extended by external plug-ins. To program in Java using Netbeans JDK (Java De-velopment kit) plug-ins must be installed. The latest version available of NetBeans is Version 7.0. Figure shows the window of the NetBeans IDE.

Android Virtual Device

The Android Virtual Device (AVD) is an emulator included in the Android SDK. The AVD lets a user to run, build and test an application without the need for any physical devices. AVD can be considered an exact replica of a real Android device, except that lacks a few features. The AVD cannot be used to make calls. The Android market is not accessible from the AVD. The applications that use sensor and Bluetooth cannot be tested in AVD. Except for this, testing an application in the AVD appears similar as in a physical Android device. Android mobile phones can also be used as emulators instead of using emulator provided by SDK. All the instruction sets required for using an Android device as an emulator can be found in the

Android developer's official web- site. Also using an Android device as an emulator is faster compared to using AVD. Figure 9 shows the Android Virtual Device provided by an Android SDK.

5.3. SYSTEM DESCRIPTION

5.3.1 MODULES

VMONKEY project can be divided into two projects they are

- REMOTE CLIENT
- REMOTE SERVER

Remote client is used to control desktop from android device and Remote server is used to control android device from desktop. Here we have five main modules. Those modules are listed as below and the functionality and design of each module is described further.

Connection Establishment

The client–server model of computing is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system. A server host runs one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers which await incoming requests.

Device To Desktop

This is the first section in the connection establishment. Here a Android device is used to control a desktop. For establishing connection you need a android device and a desktop with a working internet connectivity. For establishing connection from device to desktop you will need an ip address to communicate with the desktop. The ip address lets you identify the system you want to communicate with. Once the Ip address matches connection is established. Here connection is established using ANDROID DEBUG BRIDGE (ADB) 5037 port. Android Debug Bridge (adb) is a versatile command line tool that lets you communicate with an emulator instance or connected Android-powered device. When the server starts, it binds to local TCP port 5037 and listens for commands sent from adb clients—all adb clients use port 5037 to communicate with the adb server.

Interaction

Clients and servers exchange messages in a request-response messaging pattern: The client sends a request, and the server returns a response. This exchange of messages is an example of inter-process communication. To communicate, the computers must have a common language, and they must follow rules so that both the client and the server know what to expect. The language and rules of communication are defined in a communications protocol. All client-server protocols operate in the application layer. The application-layer protocol defines the basic patterns of the dialogue.

Device To Desktop

This section describes about how interaction takes place when we control a desktop using a android device. The desktop is controlled by two sections Server Window and Autobot

The server has the duty to collect actions that are carried out on the device. For eg : a mouse tap. Such actions are collected down in the server window.

Once the actions are collected in the server window the autobot collects them and executes them on the desktop. That each and every action carried out on the device is executed on the desktop via the autobot.

6.SYSTEM TESTING AND IMPLEMENTATION

6.SYSTEM TESTING AND IMPLEMENTATION

Software testing is a critical element of software quality assurance and represents the ultimate reviews of specification, design and coding. Testing present an interesting anomaly for the software. Testing is vital to the success of the system. Errors can be injected at any stage during development. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved.

During testing, the program to be tested is executed with set of test data and the output of the program for the test data is evaluated to determine if the program is performing as expected. A series of testing are performed for the proposed system before the system is ready for user acceptance testing.

TYPES OF TESTING:

- Unit Testing
- Integration Testing
- Validation Testing
- Output Testing
- User Acceptance Testing

6.1 SYSTEM TESTING

6.1.1. UNIT TESTING

Unit testing focuses verification effort on the smallest unit of the software design, the module this is known as module testing. Since the proposed system has modules the testing is individually performed on each module.

Using the details description as a guide, important control paths are tested to uncover errors within the boundary of the modules. This testing was carried out during programming stage itself. In this testing step each module is found to be working satisfactorily as regards to the expected output from the module .In our system we want to check the informations like whether the inputs are saved to back end correctly. So Every form includes this testing because we want to maintain our database because informations like Employee details, calculation tables including Test and Clr should be maintained correctly. These are checked in the programming step itself.

6.1.2. INTEGRATION TESTING

Data can be test across an interface, one module can have adverse effect on another, sub function when combined may not produced the desired function. Integration testing is a systematic technique for constructing the program structure while at the same time conducting test to uncover errors associated within the interface.

The objective is to take unit tested modules and built a program structure that has been dictated by design. All modules are combined in this testing step. The entire program is tested as a whole. Correction is difficult at this stage because the isolation of causes is complicated by the vast expense of the program. Thus in the integration testing step all the errors uncover are corrected for the next testing step. Primarly we have met with several errors like data save and table linking. These are corrected well.

6.1.3 VALIDATION TESTING

At the culmination of integration testing, software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of software test-validation testing begins. Validation testing can be defined in many ways, but a simple definition is that validation succeeds when the software functions in manner that is reasonably expected by the user. Software validation is achieved through a series of tests that demonstrate conformity with requirement. After validation test has been conducted, one of two conditions exists.

- The function or performance characteristics confirm to specifications and are accepted.
- A validation from specification is uncovered and a deficiency created.

There was some errors with our project in this stage too. Because there are some validation problems like saving the deatls without filling all the fields , Data type mismatch errors and so on. These are corrected in this validation stage.

Deviation or error discovered at this step in this project is corrected prior to completion of the project with the help of the user. Thus the proposed system under consideration has been tested by using validation testing and found to be working satisfactorily.

6.1.4 OUTPUT TESTING

After performing the validation testing, the next step is output testing of the proposed system since no system could be useful if it does not produce the required output in the specific format. The output generated or displayed by the system under consideration is tested asking the users about the format required by them. Here, the output is considered in two ways: one is on the screen and the other is printed format.

In the first test we saw that our reports are disorderd and not Interactive. We found that Customer bills, Salary payment statements like outputs should be interactive. We made it in this step.

The output format on the screen is found to be correct as the format designed according to the user needs. For the hard copy also, the output comes out as specified by the user. Hence output testing doesn’t result in any connection in the system.

6.1.5 USER ACCEPTANCE TESTING

User acceptance of a system is the key factor for the success of any system. The under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at a time of developing and making for ‘**VMONKEY**’.

The testing of the software began along with coding. Since the design was fully object oriented, first the interfaces were developed and tested. Then unit testing was done for every module in the software for various inputs, such that each line of code is once executed.

After all modules were coded the integration test were carried out. Some minor errors were found in the output at the earlier stage and each of them was corrected. In the implementation of user interface part no major errors were found. After the software was completely developed, the testing was done. The output of the software were correct and accurate during the time of demonstration, after that no errors were reported.

6.2. IMPLEMENTATION

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the new system for the users, that it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the change over, an evaluation, of change over methods.

Implementation is the final and important phase. The most critical stage in achieving a successful new system and in giving the users confidence that the new system will work and be effective. The system can be implemented only after thorough testing is done and if it found to working according to the specification. This method also offers the greatest security since the old system can take over if the errors are found or inability to handle certain type of transactions while using the new system.

At the beginning of the development phase a preliminary implementation plan is created to schedule and manage the many different activities must be integrated into plan. The implementation plan is updated throughout the development phase, culminating in a changeover plan for the operation phase. The major elements of implementation plan are test plan, training plan, equipment installation plan and a conversion plan.

There are three types of implementation:

- Implementation of a computer system to replace a manual system.

- Implementation of a new computer system to replace an existing one.
- Implementation of a modified application to replace an existing one, using the same computer.

In our case it was about to implement a new system to replace manual system. All the operations in the ”VMONKEY“ was conducted automatically. It give us to an extend experience to the user to use input device remotively.

VMONKEY enables users around the globe to view, diagnose, and take full control of a remote computer/android device to resolve any problem! No matter how far apart the users are, VMONKEY works as if the two machines are on the same table. All you need is an Internet connectivity and an Android device with VMONKEY installed on it. No matter where you or your remote computer/android device is located, you will always be able to run a session and connect to the remote computer/android device. Our technology guarantees reliability, security, and privacy for you.

With VMONKEY we have extended the features of the current system. VMONKEY is an Android app which enables a user to access a computer android device from a remote location. With this application the user can switch connections using SWITCHING. In short the application provides a flexible interface for the user to control their computer/android device.

7. CONCLUSION

7. CONCLUSION

The project entitled “VMONKEY” was completed on time and was tested with proper data. This project explores the possibility of controlling the computer remotely using an Android phone device. The proposed prototype is able to control a lot of operations a normal computer keyboard and mouse would perform. It practically turns a mobile phone into a wireless keyboard and mouse using a wireless network via a portable mobile device running under an Android Platform Operating System. It helps mobile phone users on facilitating their work in study life, home life or working life, where the use of the prototype helps in easing the device control.

It is proven that this prototype would relieve a pain in the neck and also the normal back ache due to constantly sitting at a particular place. With the help of this prototype, these stressful moments will be minimized as users will be having a very relaxed position as intended. This is a convenient application for simple operations and for manipulating such computer without the keyboard and mouse been connected.

8. BIBLIOGRAPHY

8. BIBLIOGRAPHY

❖ Learn Android App Development
Wallace Jackson

❖ Android Application Development for Dummies
Donn Felker

- http://www.remotedroid.net/

- http://www.gmote.org/

- http://sites.google.com/site/myremoteandroid/

9. APPENDIX

9.1 SAMPLE SOURCE CODE

Main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="fill_parent"
android:layout_height="fill_parent">
      <TextView android:text="VMoNKey" android:id="@+id/tvTitle"
            android:layout_width="wrap_content"
android:layout_height="wrap_content"
            android:textSize="20dp" android:textStyle="bold"
            android:layout_centerHorizontal="true"></TextView>

      <TextView
          android:id="@+id/tvInstructions"
          android:layout_width="fill_parent"
          android:layout_height="50dp"
          android:layout_below="@id/tvTitle"
          android:layout_gravity="left"
          android:layout_marginLeft="5dp"
          android:layout_marginRight="5dp"
          android:layout_marginTop="5dp"
          android:text="Copyright@2015 by VSoft Technologies

Please enter your IP Address" />

      <Button android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:text="@string/txt_connect"
            android:layout_marginLeft="0dp" android:layout_marginTop="5dp"
            android:layout_marginRight="5dp"
android:layout_marginBottom="5dp"
            android:id="@+id/btnConnect" android:layout_gravity="left"
            android:layout_alignParentRight="true"
android:layout_below="@id/tvInstructions" />

      <EditText
          android:id="@+id/etIp"
          android:layout_width="fill_parent"
          android:layout_height="wrap_content"
          android:layout_below="@id/tvInstructions"
          android:layout_marginBottom="5dp"
          android:layout_marginLeft="5dp"
          android:layout_marginRight="0dp"
          android:layout_marginTop="5dp"
          android:layout_toLeftOf="@id/btnConnect"
          android:enabled="true"
          android:focusable="true"
          android:text="192.168.43.118" />

      <TextView android:id="@+id/tvPrefs"
android:layout_width="wrap_content"
            android:layout_height="wrap_content"
android:text="@string/txt_prefs_copy"
            android:layout_marginLeft="5px"
android:layout_marginRight="5px"
            android:layout_marginTop="0px"
android:layout_marginBottom="5px"
            android:layout_below="@id/etIp" />

      <TextView android:id="@+id/tvRecentHosts"
            android:layout_width="wrap_content"
android:layout_height="wrap_content"
            android:layout_marginLeft="5px"
android:text="@string/txt_recentlyUsedHosts"
            android:layout_marginTop="5px"
android:layout_marginBottom="5px"
            android:textStyle="bold" android:layout_centerHorizontal="true"
            android:layout_below="@id/tvPrefs" />
      <ListView android:id="@+id/lvHosts"
android:layout_width="fill_parent"
            android:layout_height="fill_parent"
android:layout_alignParentBottom="true"
            android:layout_below="@id/tvRecentHosts"/>

</RelativeLayout>
```

VMonkey.java

```
package com.vsoft;

import android.app.Activity;
import android.content.Intent;
import android.os.*;
import android.util.Log;
import android.view.*;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import android.widget.TextView;
import android.widget.Toast;
import com.vsfot.R;
import java.util.*;
```

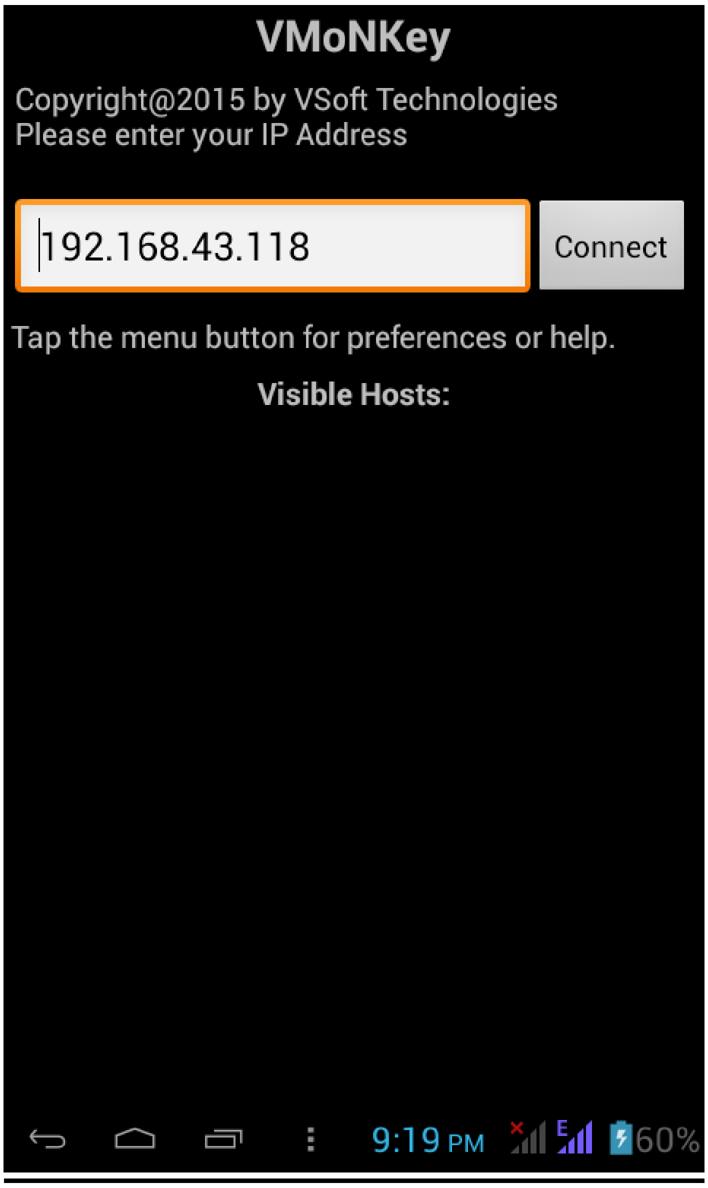
```
public class VMonkey extends Activity {
      private static final String TAG = "VMonkey";
      // menu item(s)
      public static final int MENU_PREFS = 0;
      public static final int MENU_HELP = 1;


      //
      private EditText tbIp;
      //
      private HelpDialog dlHelp;
      //
      private DiscoverThread discover;
      private Handler handler;
      private SimpleAdapter adapter;
      private Vector<String> hostlist;

      public VMonkey() {
            super();
      }

      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            requestWindowFeature(Window.FEATURE_NO_TITLE);
      setContentView(R.layout.main);
            //
            this.handler = new Handler();
            // set some listeners
            Button but = (Button)this.findViewById(R.id.btnConnect);
            but.setOnClickListener(new View.OnClickListener() {
                  public void onClick(View v) {
                        onConnectButton();
                  }
            });

            // check SharedPreferences for IP
            Settings.init(this.getApplicationContext());

            //
            this.tbIp = (EditText)this.findViewById(R.id.etIp);
            if (Settings.ip != null) {
                  this.tbIp.setText(Settings.ip);
            }
            //
            if (this.dlHelp == null) {
                  this.dlHelp = new HelpDialog(this);
            }
            // discover some servers
            this.hostlist = new Vector<String>();

      ((ListView)this.findViewById(R.id.lvHosts)).setOnItemClickListener(ne
w AdapterView.OnItemClickListener() {
                  public void onItemClick(AdapterView adapter, View v, int
position, long id) {
                        onHostClick(position);
                  }
            });
      }

      private void updateHostList() {
            FoundHostsAdapter adapter = new
FoundHostsAdapter(this.hostlist, this.getApplication());

      ((ListView)this.findViewById(R.id.lvHosts)).setAdapter(adapter);
      }


      /** OS kills process */
      public void onDestroy() {
            super.onDestroy();
      }

      /** App starts anything it needs to start */
      public void onStart() {
            super.onStart();
      }

      /** App kills anything it started */
      public void onStop() {
            super.onStop();
      }

      /** App starts displaying things */
      public void onResume() {
            super.onResume();
            this.discover = new DiscoverThread(new
DiscoverThread.DiscoverListener() {
                  public void onAddressReceived(String address) {
                        hostlist.add(address);
                        Log.d(TAG, "Got host back, "+address);
                        handler.post(new Runnable() {
                              public void run() {
                                    updateHostList();
                              }
                        });
                  }
            });
            this.discover.start();
      }


      /** App goes into background */
      public void onPause() {
            super.onPause();
            this.discover.closeSocket();
      }

      // menu

      public boolean onCreateOptionsMenu(Menu menu) {
            super.onCreateOptionsMenu(menu);
            //
            menu.add(0, MENU_PREFS, 0,
R.string.txt_preferences).setShortcut('0',
'p').setIcon(R.drawable.icon_prefs);
            menu.add(0, MENU_HELP, 0, R.string.txt_help).setShortcut('1',
'h').setIcon(R.drawable.icon_help);
            //
            return true;
      }
```

```
	public boolean onOptionsItemSelected(MenuItem item) {
		//
		switch (item.getItemId()) {
		case MENU_PREFS:
			//
			this.onPrefs();
			break;
		case MENU_HELP:
			//
			this.onHelp();
			break;
		}
		//
		return super.onOptionsItemSelected(item);
	}

	//

	private void onConnectButton() {
		String ip = this.tbIp.getText().toString();
		if (ip.matches("^[0-9]{1,4}\\.[0-9]{1,4}\\.[0-9]{1,4}\\.[0-
9]{1,4}$")) {
			try {
				Settings.setIp(ip);
				//
				Intent i = new Intent(this, PadActivity.class);
				this.startActivity(i);
				this.finish();
			} catch (Exception ex) {
				//this.tvError.setText("Invalid IP address");
				//this.tvError.setVisibility(View.VISIBLE);
				Toast.makeText(this,
this.getResources().getText(R.string.toast_invalidIP),
Toast.LENGTH_LONG).show();
				Log.d(TAG, ex.toString());
			}
		} else {
			//this.tvError.setText("Invalid IP address");
			//this.tvError.setVisibility(View.VISIBLE);
			Toast.makeText(this,
this.getResources().getText(R.string.toast_invalidIP),
Toast.LENGTH_LONG).show();
		}
	}

	private void onHostClick(int item) {
		this.tbIp.setText(this.hostlist.get(item));
		this.onConnectButton();
	}

	private void onHelp() {
		this.dlHelp.show();
	}

	private void onPrefs() {
		Intent i = new Intent(VMonkey.this, PrefsActivity.class);
		this.startActivity(i);
	}

}
```

Vmonkey.mainfest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.vsfot" android:versionCode="8"
android:versionName="1.4.5.8">

      <application android:icon="@drawable/icon"
android:label="@string/app_name">
            <activity android:name="com.vsoft.VMonkey"
android:label="@string/app_name"
                  android:screenOrientation="user">
                  <intent-filter>
                        <action android:name="android.intent.action.MAIN"
/>
                        <category
android:name="android.intent.category.LAUNCHER" />
                  </intent-filter>
            </activity>

            <activity android:name="com.vsoft.PadActivity"
                  android:screenOrientation="user"
android:windowSoftInputMode="stateVisible|adjustPan">
            </activity>
            <activity android:name="com.vsoft.PrefsActivity"
                  android:screenOrientation="portrait"></activity>

      </application>
      <uses-sdk android:minSdkVersion="3" android:targetSdkVersion="5"/>

      <support-screens android:largeScreens="true"
            android:normalScreens="true" android:smallScreens="true"
android:anyDensity="true"
            />

      <uses-permission android:name="android.permission.INTERNET"></uses-
permission>
      <uses-permission android:name="android.permission.WAKE_LOCK"></uses-
permission>
</manifest>
```

9.2 INPUT OUTPUT FORMS

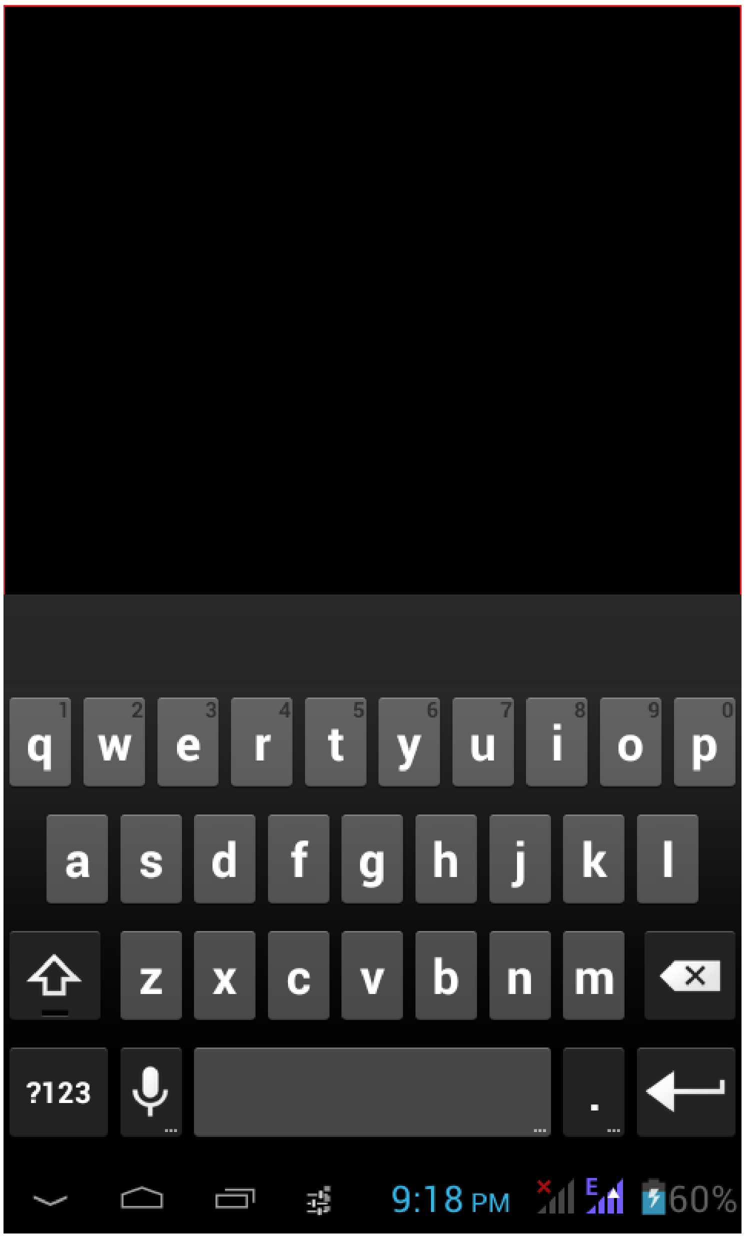

VMoNKey

Copyright@2015 by VSoft Technologies
Please enter your IP Address

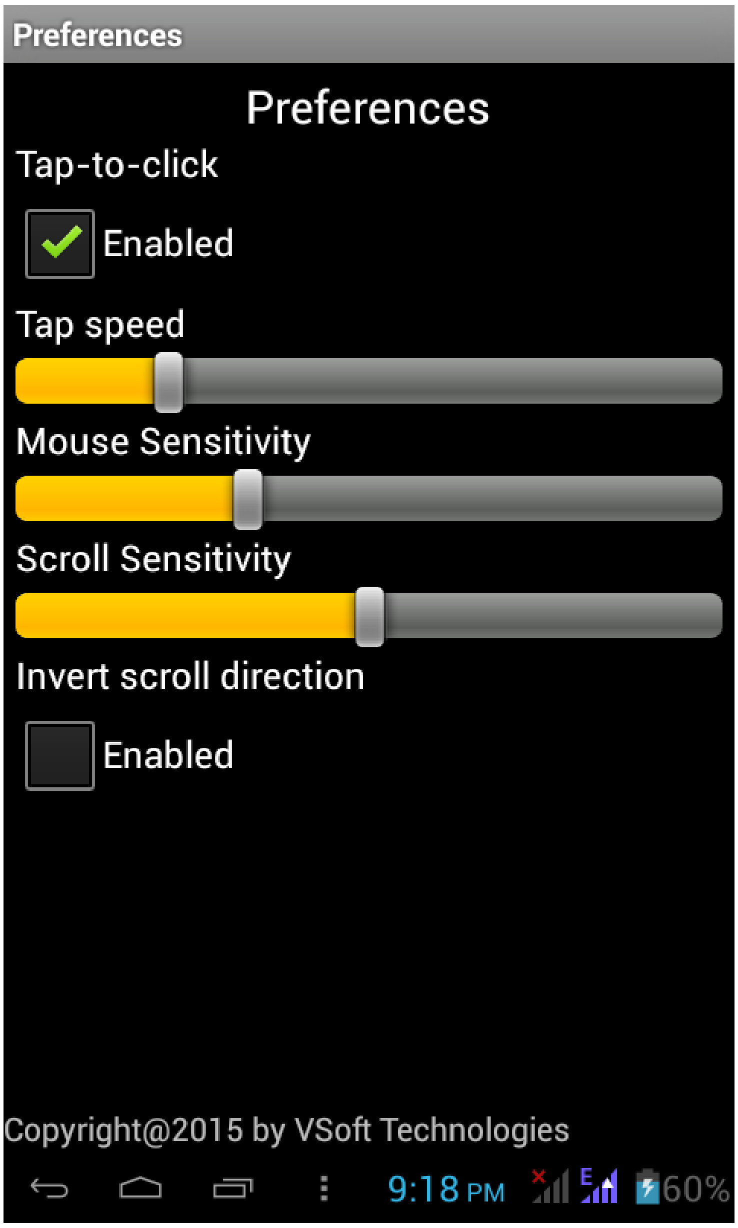
192.168.43.118

Tap the menu button for preferences or help.

Visible Hosts:


q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123
9:18 PM
60%

9:18 PM
60%



Preferences

Preferences

Tap-to-click

[✓] Enabled

Tap speed

Mouse Sensitivity

Scroll Sensitivity

Invert scroll direction

[ ] Enabled

Copyright@2015 by VSoft Technologies