**CHAPTER 1**

**1. INTRODUCTION**

**O**ver the past few decades, Embodied Conversational Agents(ECAs) have been used as virtual assistants that make easier the access to information or help in performing complex tasks. Due to their high computational requirements ECAs are usually run on desktop computers, but with the recent development ofhand-held devices both in hardware and software, it becomesnecessary to move ECAs to that new mobile scenario. Thus, we propose an open-source based platform for developing ECA based interfaces on Android-equipped devices. We also present a prototype for controlling a home automation system.

**1.1 GENERAL**

In this paper, we propose an open-source based platform for developing ECA based interfaces on Android-equipped devices. We also present a prototype for controlling a home automation system.

**1.2 OBJECTIVE**

Smart Grid has been characterized as an integrated system that can increase the efficiency, reliability and flexibility of the electricity network through a two-way flow of electricity and information.As the customers choose to tailor their energy consumptions in responding to price or environmental concerns, the peak load burden will bereduced, and hence Smart Grid can meet increased customer demand without addingexpensive infrastructure.At the same time, integrationof the renewable energy sources willincrease the power diversity, and reduce ourdependence on fossil fuel as well as the greenhousegases.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 Humor and Embodied Conversational Agents.**

In this paper, the role of humor in human-to-human interaction and the possible role of humor in human-computer interaction. The aim is to see whether it is useful for embodied conversational agents to integrate humor capabilities in the internal model of intelligence, emotions and interaction (verbal and nonverbal)capabilities. A current state of the art of research in embodied conversational agents, affective computing and verbal and nonverbal interaction is presented.

**2.2An Intelligent TV interface based on Statistical Dialogue Management.**

In this paper, an intelligent TV interface using a voice-enable dialogue system. This paper rests on the both directions: a new type of dialogue management model and its use for practical systems to commercialize. We devise a practical dialogue management model based on statistical learning methods. To analyze discourse context, we utilize statistical learning techniques for anaphora resolution and discourse history management. Contrary to the rule-based system, we develop an incremental learning method to construct dialogue strategies from the training corpus.

**2.3Grounded Language Modeling for Automatic Speech Recognition of Sports Video Michael Fleischman**

This paper describes show they are learned from large corpora of unlabeled video, and are applied to thetas of automatic speech recognition of sports video. Results show that grounded language models improve perplexity and word error rate over text based language models, and further, support video information retrieval better than human generated speech transcriptions.

**2.4 How was your day?’ An affective companion ECA prototype Marc Cavazza**

This paper presents a dialogue system in the form of an ECA that acts as a sociable and emotionally intelligent companion for the user. The system dialogue is not task-driven but is social conversation in which the user talks about his/her day at the office. During conversations the system monitors the emotional state of the user and uses that information to inform its dialogue turns. The system is able to respond to spoken interruptions by the user.

**CHAPTER 3**

**SYSTEM ANALYSIS**

* 1. **EXISTING SYSTEM:**

Smart Grid has been characterized as an integrated system that can increase the efficiency, reliability and flexibility of the electricity network through a two-way flow of electricity and information.As the customers choose to tailor their energy consumptions in responding to price or environmental concerns, the peak load burden will bereduced, and hence Smart Grid can meet increased customer demand without addingexpensive infrastructure.At the same time, integrationof the renewable energy sources willincrease the power diversity, and reduce ourdependence on fossil fuel as well as the greenhousegases.

**3.1.1 EXISTING SYSTEM DISADVANTAGES**

Long distance information exchange throughwide area networks (WANs) has largely beenlimited to phone-to-phone or phone-to-computercommunications for pure informationtransmission or acquisition. Development of human-to-device interfaces,such as home automation systems and integratedcar-driver interfaces, has largely beenlimited to local area networks (LANs) or personalarea networks (PANs), ranging from 10to 100 meters. On the other hand, today’sWANs and LANs are on their way to maturedevelopment by supporting scalable multimediaservices with increasingly *flexible* designs.

**3.2 PROPOSED SYSTEM**

* Embodied Conversational Agents (ECAs) are animated virtual characters that emulate human behavior and communication.
* Due to the limited computational power of hand-held devices compared to desktop computers, the most common architectures for ECA-based mobile applications rely on an external server that performs the processor intensive tasks, such as speech recognition, language understanding and text-to-speech.
* This describes a platform for developing ECAbasedinterfaces on Android hand-held devices.
* The proposedplatform is based on free and open source libraries. Wedeveloped a prototype installed on a tablet for controlling ahome automation system.

**3.2.1 PROPOSED SYSTEM ADVANTAGES**

* Proposed system includes 6 factorsVoice Activity Detector, Automatic Speech Recognition, Conversational Engine, Control Interface, Text-To-Speech, Virtual Head Animation
* It designs mobile-based device monitoring and control, which can be applied inboth fixed or moving LAN scenarios, such as vehicle electronics, power and energy systems, etc.,

**CHAPTER 4**

**SYSTEM REQUIREMENTS**

**4.1 HARDWARE REQUIREMENTS**

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. It should what the system do and not how it should be implemented.

* 1 GB RAM
* 80 GB Hard Disk
* Intel Processor
* GPRS activated Android Mobile
* Datacard

**4.2 SOFTWARE REQUIREMENT**

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team’s progress throughout the development activity.

* JDK 1.7
* Eclipse with Andoid plugin
* Android SDK
* Windows 7 32bit

**CHAPTER 5**

**PROJECT DESCRIPTION**

**5.3.1 MODULES NAME**

* Voice Activity Detector:
* Automatic Speech Recognition
* Conversational Engine
* Control Interface
* Text-To-Speech
* Virtual Head Animation

**5.3.2.MODULE DESCRIPTIONS**

**Voice Activity Detector:**

The Voice Activity Detector’s (VAD) role is to discriminate the user’s voice frames from those containing noise. This module reads the digitized audio samples acquired from a microphone and sends the filtered raw audio to the ASR. The actual implementation of the VAD module is based on the Sphinx Base library, which was modified so it can work with the OpenSL ES native audio libraries present on Android.

Filtered Raw

**ASR**

**Microphone**

**VAD**

Audio Audio

**Automatic Speech Recognition**

The Automatic Speech Recognition (ASR) module performs speech to text conversion. It takes as input the utterance with the user’s speech that come from the VAD and sends the resultant text to the CE. In the proposed platform, the ASR module is based on the Pocket Sphinx speech recognition library.

**VAD**

**ASR**

**CE**

**Audio Text**

**Conversational Engine**

The Conversational Engine (CE) extracts the meaning of the utterance, manages the dialog flow and produces the actions appropriate for the target domain. It generates a response based on the input, the current state of the conversation and the dialog history. It was also added support for an object-oriented database that can decrease the dynamic memory usage at the expense of an increment of the response time

**ASR**

**CI**

**CE**

**Speech**

**Control Interface**

The Control Interface translates the commands said by then user to a format that can be understood by the target applications or services running on the same device or accessible remotely. This module is domain-specific and has to be re-implemented or adapted for every new target application.

**User commands Target Application**

**CI**

**Text-To-Speech**

The TTS module implementation is based on the eSpeak library. The Text-To-Speech (TTS) subsystem carries out the generation of the synthetic output voice from the text that comes as a response from the CE. It sends to the VHA module a list of the phonemes with their duration so animation and artificial speech match up. The TTS module implementation is based on the eSpeak library.

**synthetic**

**VHA**

**TTS**

**CE**

**Voice**

**Virtual Head Animation**

This module receives as inputs both the mood information from the CE and the list of the phonemes’ durations from the TTS module. By processing the inputs, it generates the visemes (the visual representation of the phonemes) and the facial expression that will be rendered along with the synthetic voice.

**TTS**

**CE**

**Mood information**

**CHAPTER 6**

**SYSTEM DESIGN**

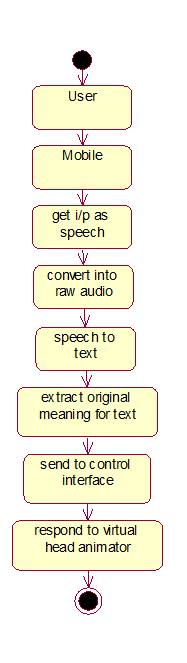
**6.1 GENERAL**

Design Engineering deals with the various UML [Unified Modeling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product.

**6.2 UML DIAGRAMS**

**6.2.1 ACTIVITY DIAGRAM**

Activity diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. UML, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. UML activity diagrams could potentially model the internal logic of a complex operation. In many ways UML activity diagrams are the object-oriented equivalent of flow charts and data flow diagrams (DFDs) from structural development.

****



**6.2.2 USE CASE DIAGRAM**

A use case diagram is a type of behavioral diagram created from a Use-case analysis. The purpose of use case is to present overview of the functionality provided by the system in terms of actors, their goals and any dependencies between those use cases.

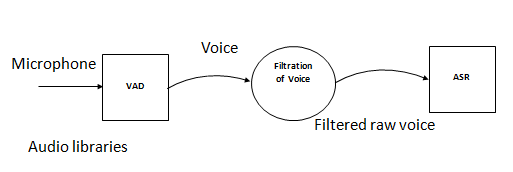
****



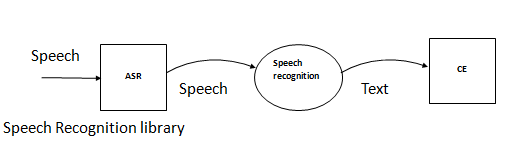
**6.2.3 DATA FLOW DIAGRAM**

A data flow diagram (DFD) is a graphical representation of the “flow” of data through an information system. It differs from the flowchart as it shows the data flow instead of the control flow of the program. A data flow diagram can also be used for the visualization of data processing. The DFD is designed to show how a system is divided into smaller portions and to highlight the flow of data between those parts.

**LEVEL 0:**

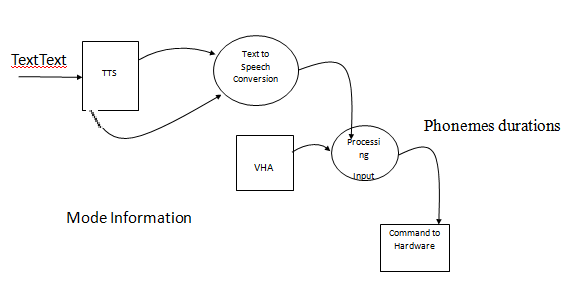


**LEVEL 1:**

****

****

**LEVEL 2:**



**6.2.4 SEQUENCE DIAGRAM**

A sequence diagram in UML is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a message sequence chart. Sequence diagrams are sometimes called Event-trace diagrams, event scenarios, and timing diagrams.



**6.2.5 COLLABORATION DIAGRAM**

A collaboration diagram show the objects and relationships involved in an interaction, and the sequence of messages exchanged among the objects during the interaction.

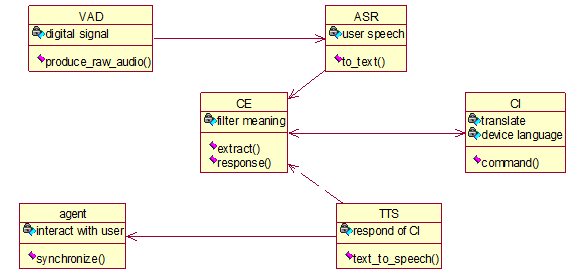
The collaboration diagram can be a decomposition of a class, class diagram, or part of a class diagram. It can be the decomposition of a use case, use case diagram, or part of a use case diagram.

The collaboration diagram shows messages being sent between classes and object (instances). A diagram is created for each system operation that relates to the current development cycle (iteration).

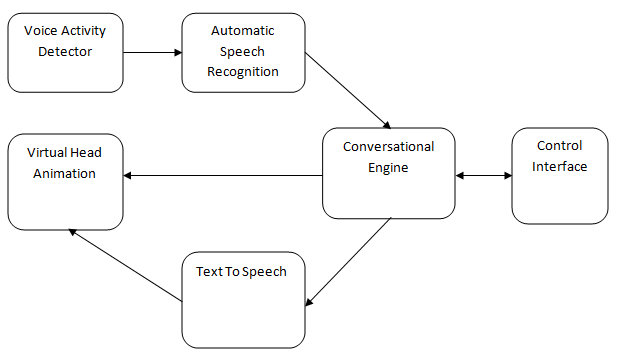


**4.1.6. CLASS DIAGRAM**

A class diagram in the UML is a type of static structure diagram that describes the structure of a system by showing the system’s classes, their attributes, and the relationships between the classes. Private visibility hides information from anything outside the class partition. Public visibility allows all other classes to view the marked information. Protected visibility allows child classes to access information they inherited from a parent class.

****

**6.2.7SYSTEM ARCHITECTURE**

****

****

**6.2.8. E-R DIAGRAM**

In software engineering, an **entity-relationship model** (**ERM**) is an abstract and conceptual representation of data. Entity-relationship modelling is a database modelling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database, and its requirements in a top-down fashion. Diagrams created by this process are called **entity-relationship diagrams**, **ER diagrams**, or **ERDs**.



Fig 6.2.8 E-R DIAGRAM

**CHAPTER 7**

**SOFTWARE SPECIFICATION**

**7.1 GENERAL**

This chapter is about the software language and the tools used in the development of the project. The platform used here is Android. The Primary language is Java on Android. In this project Android is chosen for implementation.

**7.2 ANDROID**

A free, open source mobile platform, Linux-based, multiprocessing, Multithreaded OS. Android is not a device or a product It’s not even limited to phones You could build a DVR, a handheld GPS, an MP3 player, etc.

**7.3 MySQL 5.6**

MySQL Server is a Structured Query Language (SQL) based, client/server relational database. Each of these terms describes a fundamental part of the architecture of SQL Server.

**7.3.1.DATABASE**

A database is similar to a data file in that it is a storage place for data. Like a data file, a database does not present information directly to a user; the user runs an application that accesses data from the database and presents it to the user in an understandable format.

A database typically has two components: the files holding the physical database and the database management system (DBMS) software that applications use to access data. The DBMS is responsible for enforcing the database structure, including:

* Maintaining the relationships between data in the database.
* Ensuring that data is stored correctly and that the rules defining data relationships are not violated.
* Recovering all data to a point of known consistency in case of system failures.

**7.3.2.Structured Query Language (SQL)**

To work with data in a database, you must use a set of commands and statements (language) defined by the DBMS software. There are several different languages that can be used with relational databases; the most common is SQL. Both the American National Standards Institute (ANSI) and the International Standards Organization (ISO) have defined standards for SQL. Most modern DBMS products support the Entry Level of SQL-92, the latest SQL standard (published in 1992).

### 7.3.3.MYSQL Features

MySQL Server supports a set of features that result in the following benefits:

### Ease of installation, deployment, and use

MySQL includes a set of administrative and development tools that improve your ability to install, deploy, manage, and use MySQL across several sites.

### Scalability

The same database engine can be used across platforms ranging from laptop computers running Microsoft Windows® 95/98 to large, multiprocessor servers running Microsoft Windows NT®, Enterprise Edition.

### Data warehousing

SQL Server includes tools for extracting and analysing summary data for online analytical processing (OLAP). SQL Server also includes tools for visually designing databases and analysing data using English-based questions.

### System integration with other server software

SQL Server integrates with e-mail, the Internet, and Windows.

### Databases

A database in MySQL Server consists of a collection of tables that contain data, and other objects, such as views, indexes, stored procedures, and triggers, defined to support activities performed with the data. The data stored in a database is usually related to a particular subject or process, such as inventory information for a manufacturing warehouse.

SQL Server can support many databases, and each database can store either interrelated data or data unrelated to that in the other databases. For example, a server can have one database that stores personnel data and another that stores product-related data.

Alternatively, one database can store current customer order data, and another; related database can store historical customer orders that are used for yearly reporting. Before you create a database, it is important to understand the parts of a database and how to design these parts to ensure that the database performs well after it is implemented.

**7.4.1.Android Platform**

A free, open source mobile platform, Linux-based, multiprocessing, Multithreaded OS. Android is not a device or a product It’s not even limited to phones You could build a DVR, a handheld GPS, an MP3 player, etc.

Android is a software stack for mobile devices that includes an operating system, middleware and key applications.

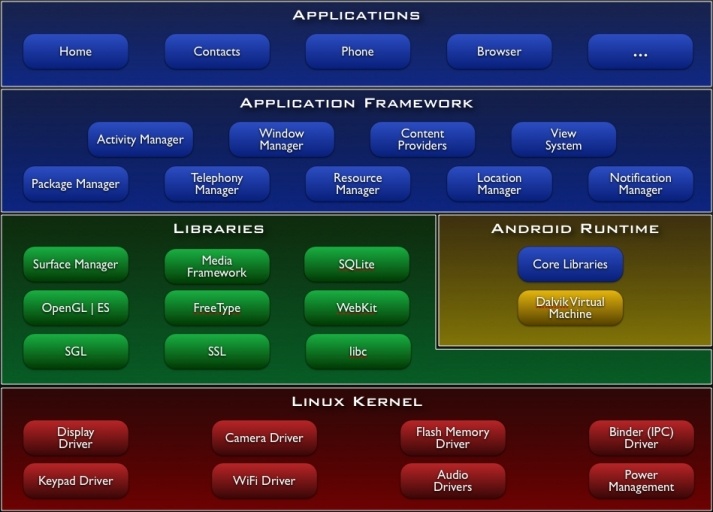
The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language.

Features

* Application framework enabling reuse and replacement of components
* Dalvik virtual machine optimized for mobile devices
* Integrated browser based on the open source Web Kit engine
* Optimized graphics powered by a custom 2D graphics library; 3Dgraphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
* SQLite for structured data storage
* Media support for common audio, video, and still image formats(MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)

**7.4.2.Android Runtime**

* Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language.
* Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently.
* The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint.
* The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management.
* Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework.



**7.4.3.AndroidDevelopment Tools**

These are the most important parts of the Android APIs:

The Android SDK includes a variety of custom tools that help you develop mobile applications on the Android platform. Three of the most significant tools are:

* **Android Emulator** -A virtual mobile device that runs on our computer -use to design, debug, and test our applications in an actual Android run-time environment
* **Android Development Tools Plug-in** -for the Eclipse IDE - adds powerful extensions to the Eclipse integrated environment
* **Dalvik Debug Monitor Service (DDMS)**-Integrated with Dalvik -this tool let us manage processes on an emulator and assists in debugging
* **Android Asset Packaging Tool (AAPT)** – Constructs the distributable Android package files (.apk)
* **Android Debug Bridge (ADB)**–provides link to a running emulator. Can copy files to emulator, install .apk files and run commands.

Lifecycle of activity

****

**7.4.4.Overview Of XML:**

XML (Extensible Markup Language) is a set of rules for encoding documents electronically. It is defined in the XML 1.0 Specification produced by the W3C, and several other related specifications, all gratis open standards

XML’s design goals emphasize simplicity, generality, and usability over the Internet is a textual data format, with strong support via Unicode for the languages of the world. Although XML’s design focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services

**Advantages:**

The main usage of xml Is we can store and retrieve data easily with the help of xml.

**XML Introduction:**

MIDP devices have memory constraints when it comes to code, both in terms of the amount of code you can store on the device, and memory available to applications at runtime. So, keeping the size of applications and features in check is of paramount importance to the J2ME developer. That’s wheresmall-sized XML parsers come into play.

**XML parsers**

This section describes the XML parsing process and introduces some small XML parsers for MIDP.

**XML parsing process**

The XML parsing process operates in three phases:

* XML input processing. In this stage, the application parses and validates the source document recognizes and searches for relevant information based on its location or its tagging in the source document; extracts the relevant information when it is located; and, optionally, maps and binds the retrieved information to business objects.
* Business logic handling. This is the stage in which the actual processing of the input information takes place. It might result in the generation of output information.
* XML output processing. In this stage, the application constructs a model of the document to be generated with the Document Object Model (DOM). It then either applies XSLT style sheets or directly serializes to XML.

An application that implements such a processing model is called an XML parser. You can integrate animal parser into your Java applications with the Java API for XML Processing(JAXP). JAXP allows applications to parse and transform XML documents using an API that is independent of any particular XML processor implementation. Through a plug-in scheme, developers can change XML processor implementations without altering their applications.

Create Android Application

The first step is to create a simple Android Application using Eclipse IDE. Follow the option File -> New -> Project and finally select Android New Application wizard from the wizard list. Now name your application as Hello World using the wizard window as follows:

**7.4.5.JDBC**

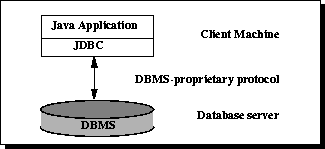
Java Database Connectivity (JDBC) is a programming framework for Java developers writing programs that access information stored in databases, spreadsheets, and flat files. JDBC is commonly used to connect a user program to a "behind the scenes" database, regardless of what database management software is used to control the database. In this way, JDBC is cross-platform . This article will provide an introduction and sample code that demonstrates database access from Java programs that use the classes of the JDBC API, which is available for free download from Sun's site .

A database that another program links to is called a data source. Many data sources, including products produced by Microsoft and Oracle, already use a standard called Open Database Connectivity (ODBC). Many legacy C and Perl programs use ODBC to connect to data sources. ODBC consolidated much of the commonality between database management systems.JDBC builds on this feature, and increases the level of abstraction. JDBC-ODBC bridges have been created to allow Java programs to connect to ODBC-enabled database software .

**JDBC Architecture**

Two-tier and Three-tier Processing Models

The JDBC API supports both two-tier and three-tier processing models for database access.



In the three-tier model, commands are sent to a "middle tier" of services, which then sends the commands to the data source. The data source processes the commands and sends the results back to the middle tier, which then sends them to the user. MIS directors find the three-tier model very attractive because the middle tier makes it possible to maintain control over access and the kinds of updates that can be made to corporate data. Another advantage is that it simplifies the deployment of applications. Finally, in many cases, the three-tier architecture can provide performance advantages.

**CHAPTER 8**

**IMPLEMENTATION**

**8.1 GENERAL**

This chapter describes the application implementation to achieve Home Automation System using Android based on ECA.

**8.2. CODING FOR IMPLEMENTATION**

**CONTROLLER INTERFACE**

package com.eca;

importjava.io.IOException;

importjava.io.InputStream;

importjava.io.OutputStream;

importjava.sql.ResultSet;

importjava.util.Enumeration;

importjavax.swing.\*;

importjavax.comm.\*;

importcom.commondb.Common\_DB;

public class Main implements Runnable {

CommDriver driver;

CommPortIdentifierpid;

SerialPort port;

Enumeration ports = null;

InputStreamips = null;

OutputStream ops = null;

String commid="";

public Main() {

try {

commid = JOptionPane.showInputDialog("Enter your Comm Port ID","COM");

Common\_DB cd = new Common\_DB();

driver = (CommDriver)Class.forName("com.sun.comm.Win32Driver").newInstance();

} catch (InstantiationException | IllegalAccessException

| ClassNotFoundException e) {

e.printStackTrace();

}

driver.initialize();

}

public void run() {

try {

ports = CommPortIdentifier.getPortIdentifiers();

while(ports.hasMoreElements()) {

pid = (CommPortIdentifier)ports.nextElement();

if(pid.getPortType()==CommPortIdentifier.PORT\_SERIAL) {

if(pid.getName().equalsIgnoreCase(commid)) {

System.out.println("\nRunning..\n");

port = (SerialPort)pid.open("ECA Thread", 2000); ips = port.getInputStream();

ops = port.getOutputStream();

while(true) {

ResultSetrs = Common\_DB.ViewParticularData("GPRSECA", "instructions", “passkey", "MySecret");

if(rs.next()) {

String instrction = rs.getString("ins"); ops.write(instrction.getBytes());

}

Thread.sleep(1000);

}

}

}

}

} catch (InstantiationException | IllegalAccessException

| ClassNotFoundException | PortInUseException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

} catch (Exception e) {

e.printStackTrace();

}

}

public static void main(String[] args) {

System.out.println("Initializing Home Automation System\n");

System.out.println("Scanning for COM ports... Please Wait..\n\n");

new Thread(new Main()).start();

}

}

**ANDROIDAPPLICATION**

packagecom.voicebasedeca;

importjava.sql.Connection;

importjava.sql.DriverManager;

importjava.sql.Statement;

importjava.util.ArrayList;

importjava.util.Locale;

importandroid.app.Activity;

importandroid.content.ActivityNotFoundException;

importandroid.content.Intent;

importandroid.os.AsyncTask;

importandroid.os.Bundle;

importandroid.speech.RecognizerIntent;

importandroid.speech.tts.TextToSpeech;

importandroid.util.Log;

importandroid.view.Menu;

importandroid.view.View;

importandroid.widget.EditText;

importandroid.widget.ImageButton;

importandroid.widget.TextView;

importandroid.widget.Toast;

importandroid.widget.ToggleButton;

public class MainActivity extends Activity implements

TextToSpeech.OnInitListener {

protected static final int RESULT\_SPEECH = 1;

privateImageButtonbtnSpeak;

ToggleButtonlighttoggle, fantoggle;

privateTextViewtxtText;

EditTextipaddress;

public String voicedata = "", dispdata = "";

privateTextToSpeechtts;

Connection con = null;

Statement ps = null;

privateintdev = 0;

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

tts = new TextToSpeech(MainActivity.this, MainActivity.this);

txtText = (TextView) findViewById(R.id.txtText);

ipaddress = (EditText) findViewById(R.id.ipaddr123);

lighttoggle = (ToggleButton) findViewById(R.id.lighttoggle);

fantoggle = (ToggleButton) findViewById(R.id.fantoggle);

btnSpeak = (ImageButton) findViewById(R.id.btnSpeak);

btnSpeak.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(

RecognizerIntent.ACTION\_RECOGNIZE\_SPEECH);

intent.putExtra(RecognizerIntent.EXTRA\_LANGUAGE\_MODEL, "en-US");

try {

startActivityForResult(intent, RESULT\_SPEECH);

txtText.setText("");

} catch (ActivityNotFoundException a) {

Toast t = Toast.makeText(getApplicationContext(),

"Ops! Your device doesn't support Speech to Text",

Toast.LENGTH\_SHORT);

t.show();

}

}

});

lighttoggle.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (lighttoggle.getText().toString().equals("light on")) {

voicedata = "3";

dispdata = "light on";

}

if (lighttoggle.getText().toString().equals("light off")) {

voicedata = "4";

dispdata = "light off";

}

SendDatasdata = new SendData();

sdata.execute();

}

});

fantoggle.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (fantoggle.getText().toString().equals("fan on")) {

voicedata = "1";

dispdata = "fan on";

}

if (fantoggle.getText().toString().equals("fan off")) {

voicedata = "2";

dispdata = "fan off";

}

SendDatasdata = new SendData();

sdata.execute();

}

});

}

@Override

publicbooleanonCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.main, menu);

return true;

}

@Override

protected void onActivityResult(intrequestCode, intresultCode, Intent data) {

super.onActivityResult(requestCode, resultCode, data);

switch (requestCode) {

case RESULT\_SPEECH: {

if (resultCode == RESULT\_OK && null != data) {

ArrayList<String> text = data

.getStringArrayListExtra(RecognizerIntent.EXTRA\_RESULTS);

voicedata = text.get(0);

if (voicedata.equals("fan start")) {

dispdata = "Fan Start";

voicedata = "1";

}

if (voicedata.equals("fan stop")) {

dispdata = "Fan Stop";

voicedata = "2";

}

if (voicedata.equals("lamp on")) {

dispdata = "Lamp on";

voicedata = "3";

}

if (voicedata.equals("lamp off")) {

dispdata = "Lamp off";

voicedata = "4";

}

if (voicedata.matches("(.\*)switch(.\*)on(.\*)lamp(.\*)")

|| voicedata.matches("(.\*)turn(.\*)on(.\*)lamp(.\*)"))

{

dispdata = "Lamp on";

voicedata = "3";

}

if (voicedata.matches("(.\*)switch(.\*)off(.\*)lamp(.\*)")

|| voicedata.matches("(.\*)turn(.\*)off(.\*)lamp(.\*)"))

{

dispdata = "Lamp off";

voicedata = "4";

}

if (voicedata.matches("(.\*)turn(.\*)on(.\*)bulb(.\*)"))

{

dispdata = "Lamp on";

voicedata = "3";

}

if (voicedata.matches("(.\*)turn(.\*)off(.\*)bulb(.\*)"))

{

dispdata = "Lamp off";

voicedata = "4";

}

if (voicedata.matches("(.\*)switch(.\*)on(.\*)fan(.\*)")

|| voicedata.matches("(.\*)turn(.\*)on(.\*)fan(.\*)"))

{

dispdata = "Fan on";

voicedata = "1";

}

if (voicedata.matches("(.\*)switch(.\*)off(.\*)fan(.\*)")

|| voicedata.matches("(.\*)turn(.\*)off(.\*)fan(.\*)"))

{

dispdata = "Fan off";

voicedata = "2";

}

if (voicedata.length() != 1) {

dispdata = "pardon";

voicedata = "4";

}

// execute thread

SendDatasdata = new SendData();

sdata.execute();

}

}

break;

}

}

@Override

public void onDestroy() {

if (tts != null) {

tts.stop();

tts.shutdown();

}

super.onDestroy();

}

@Override

public void onInit(int status) {

if (status == TextToSpeech.SUCCESS) {

tts.setLanguage(Locale.US);

tts.setLanguage(Locale.getDefault());

} else {

Log.e("TTS", "Initilization Failed!");

}

}

classSendData extends AsyncTask<String, Void, Void> {

int res = 0;

@Override

protected Void doInBackground(String... params) {

try {

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection("jdbc:mysql://"

+ ipaddress.getText().toString() + ":3306/gprseca",

"root", "password");

ps = con.createStatement();

res = ps.executeUpdate("UPDATE instructions SET ins='"

+ voicedata + "' WHERE passkey='MySecret'");

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

@Override

protected void onPostExecute(Void result) {

if (res > 0)

txtText.setText(dispdata);

else

txtText.setText("Server down");

tts.speak(txtText.getText().toString(), TextToSpeech.QUEUE\_FLUSH, null);

}

@Override

protected void onPreExecute() {

}

@Override

protected void onProgressUpdate(Void... values) {

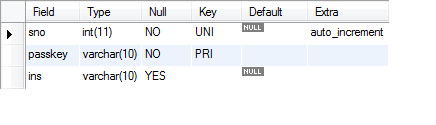
}

}

}

**8.3 DATABASE DESIGN STRUCTURE**

**Instructions Table:**



**CHAPTER 9**

**SOFTWARE TESTING**

**9.1 GENERAL**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**9.2 DEVELOPING METHODOLOGIES**

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used.

The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

**9.3 TYPES OF TESTS**

**9.3.1 Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produces valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**9.3.2 Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/ Procedures: interfacing systems or procedures must be invoked.

**9.3.3 System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**9.3.4 Performance Test**

The Performance test ensures that the output be produced within the time limits,and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

**9.3.5 Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**9.3.6 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Acceptance testing for Data Synchronization:**

* The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
* The Route add operation is done only when there is a Route request in need
* The Status of Nodes information is done automatically in the Cache Updating process

**9.2.7 Build the test plan**

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

**CHAPTER 10**

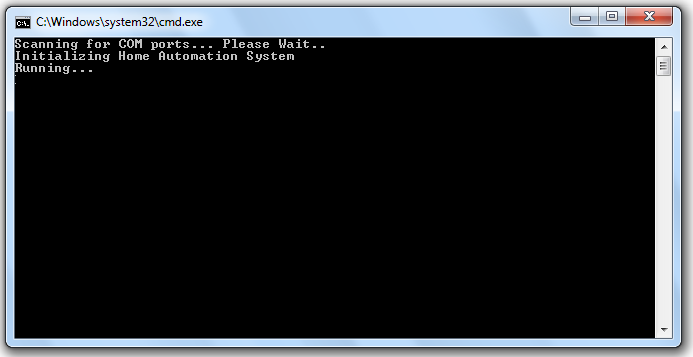
**SNAPSHOTS**

**10.1 GENERAL**

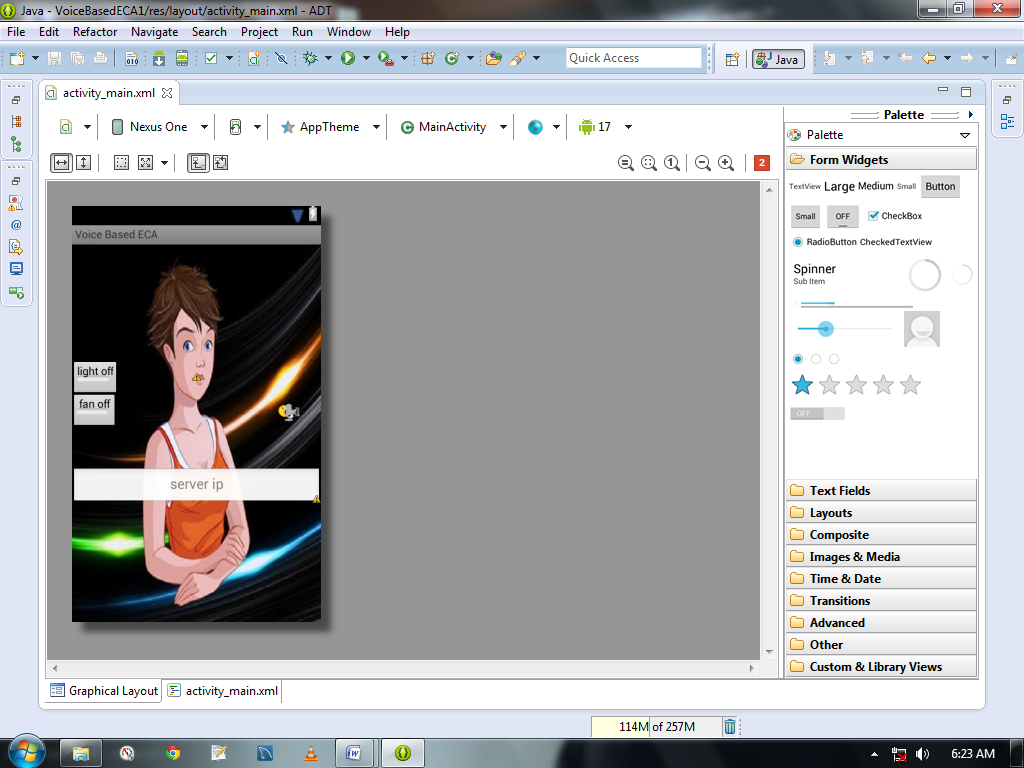
Snapshot is nothing but every moment of the application while running. It gives the clear elaborated of application. It will be useful for the new user to understand for the future steps.

**10.2. VARIOUS SNAPSHOTS**

**Running Controller**

****

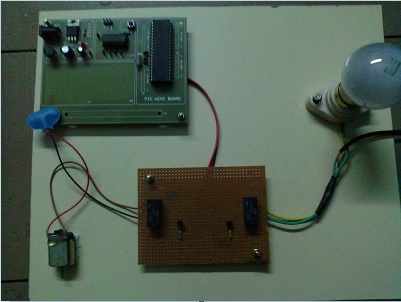
**Running on Android Emulator**

****

**Running on Android Device**

****

**PHYSICAL COMPONENTS**

****

**CHAPTER 11**

**APPLICATION**

**11.1. GENERAL**

This section gives the details of application of our ECA based Home Automation System.

**11.2. APPLICATION:**

* mobile-based device monitoring and control
* vehicle electronics
* power and energy systems

**CHAPTER 12**

**CONCLUSIONS**

The main goal of this work was to describe a platform aimed at developing ECA-based interfaces on hand-held devices equipped with Android. Thus, we proposed a possible architecture and gave implementation details for such platform. The whole platform is based on free and open source libraries and a first prototype was developed for controlling a home automation system. we propose an open-source based platform for developing ECA based interfaces on Android-equipped devices.We developed a prototype installed on a tablet for controlling a home automation system. We also present a prototype for controlling a home automation system. Due to the limited computational power of hand-held devices compared to desktop computers, the most common architectures for ECA-based mobile applications rely on an external server that performs the processor intensive tasks, such as speech recognition, language understanding and text-to-speech.

**12.2. REFERENCES**

[1] M. M. Louwerse, A. C. Graesser, D. S. McNamara, and S. Lu, “Embodied conversational agents as conversational partners,” Applied Cognitive Psychology, vol. 23, no. 9, pp. 1244–1255, 2009.

[2] B. De Carolis, I. Mazzotta, N. Novielli, and S. Pizzutilo, “Social robots and ECAs for accessing smart environments services,” in Proceedings of the International Conference on Advanced Visual Interfaces, ser. AVI ’10.New York, NY, USA: ACM, 2010, pp. 275–278.

[3] M. Cavazza, R. Santos de la C´amara, M. Turunen, J. Rela˜no Gil, J. Hakulinen, N. Crook, and D. Field, “How was your day?’ An Affective Companion ECA Prototype,” in Proceedings of the SIGDIAL 2010 Conference, Association for Computational Linguistics. Tokyo, Japan: Association for Computational Linguistics, September 2010, p. 277–280.

[4] H.-J. Oh, C.-H. Lee, M.-G. Jang, and K. Y. Lee, “An Intelligent TV interface based on Statistical Dialogue Management,” Consumer Electronics, IEEE Transactions on, vol. 53, no. 4, pp. 1602 –1607, nov. 2007.

[5] M. Santos-P´erez, E. Gonz´alez-Parada, and J. M. Cano-Garc´ıa, “EfficientUse of Voice Activity Detector and Automatic Speech Recognition in Embedded Platforms for Natural Language Interaction,” in Highlights in Practical Applications of Agents and Multiagent Systems, ser. Advances in Intelligent and Soft Computing. Springer Berlin Heidelberg, 2011, vol. 89, pp. 233–242.