

A  
Project Report  
On  
**Speech To Text Converter Application**

**BE SEM-VII**

**Subject : System Design Practices**

Prepared By  
**Mitul A. Shah (CE - 104)**  
**Ruchik P. Shah (CE - 107)**

Guided By  
**Mrudang Mehta**  
**Associate Professor**  
**Department of Computer Engineering**  
**Faculty Of Technology**  
**Dharmsinh Desai University**



**Dharmsinh Desai University, Nadiad**  
**Faculty Of Technology**  
**Department of Computer Engineering**

## 1. CERTIFICATE

DHARMSINH DESAI UNIVERSITY,  
NADIAD-387001, GUJARAT



This is to certify that the project carried in the subject of System Design Practices and recorded in this report is the bonafied work of

Mitul A. Shah (ID No.072140)

Ruchik P. Shah (ID No.072085)

Of B.E. semester – VII in the branch of Computer Engineering during the academic year 2010-2011.

**Prof. Mrudang Mehta**

Associate Professor,  
Department Of Computer  
Engineering  
Faculty of Technology,  
Dharmsinh Desai University, Nadiad

Date:

**Prof. C. K. Bhensadadia**

Head of the Department,  
Department Of Computer  
Engineering,  
Faculty of Technology,  
Dharmsinh Desai University, Nadiad

Date:

## CONTENTS

1. Certificate	2
2. Abstract	5
3. Introduction	6
3.1 Project Details	6
3.2 Purpose	6
3.3 Technologies	6
3.4 Tools and Platforms	9
3.5 Speech Recognition	9
4. System Requirement Specification	11
4.1 Software Requirement Identification	11
4.2 Requirements	11
4.3 Problems and Weakness of current system	12
4.4 User Characteristics	12
4.5 Hardware and Software Requirements	13
4.6 Constraints	13
4.7 Assumptions	14
5. Design	15
5.1 Use Case Diagram	15
5.2 Class Diagram	16
5.3 Sequence Diagrams	17
5.4 Activity Diagrams	19

5.5	State Chart	20
5.6	Flow chart	21
6.	Implementation	22
7.	Testing	30
7.1	Testing Plan	30
7.2	Testing Methods	30
7.3	Table	31
8.	Screen Shots	33
9.	Future Extension to Project	35
10.	Conclusions	36
11.	References	37

## 2. ABSTRACT

This speech recognition system recognizes the input given by the user as a speech using some recognizer and executes particular application. We have used Microsoft SAPI as a recognizer to recognize the word spoken by the user.

This software allows user to open a system application just by speaking the name of application or any other alias given to that application. It is working on the Speech Recognition technology. First user has to speak the name or alias of the application to open the application. After that, this software will search the path of application from the database. If it contains the path and name of that application in it then it will open the application otherwise it will show error message accordingly.

The main objective of this project is to execute an .exe file by speaking its name. To open any application, user does not require finding the path to open the application. If once we have added any application in database then we can easily open it by using this Speech to Text converter application. This will reduce the human efforts and saves time also.

### 3. INTRODUCTION

#### 3.1. Project Details



This is software which allows user to open an application just by speaking the name of that file. It is working on the Speech Recognition technology. In it first of all we have to speak the name of the application which we want to open. After speaking the name, software will search for path of that application in the table. If it is available, it will open that application otherwise it will give error message accordingly.

#### 3.2. Purpose

The purpose of this project is to aid user in starting any process. It starts executing any application which you said. It works on Speech Recognition technology.

#### 3.3. Technologies

The technology used in the development of this project is as mentioned below:

-  J2SE
-  JAVA Speech API 1.0
-  Microsoft Access-2007 Database

- J2SE

J2SE is a mostly used platform for application programming in Java. Java platform is used to deploy portable applications for general use. Java SE

consists of a java virtual machine (JVM) inbuilt, which must be used to run Java programs, together with a set of libraries (or "packages") needed to allow the use of them from the programs of the application. The JRE and JDK are the actual files that are downloaded and installed on a computer in order to run or develop java programs, respectively.

List of packages used in our project :

- java.io.\*
- java.awt.\*
- java.awt.event.\*
- java.util.\*
- java.sql.\*

- **JAVA Speech API 1.0**

- The Java Speech API specifies a platform-independent Java API for accessing various speech technologies.
- The Java Speech API is not an integral part of the J2SE API, but an optional extension to J2SE. The API itself provides the interface to the Java language technologies which implement only Speech API. Implementations of the Java Speech API, for example, "The Cloud Garden", "FreeTTS, IBM Speech for Java".
- Two types of speech technologies are: Speech recognition and Speech synthesis.
- Speech Recognition is used to convert speech to text with specific grammar. On the other hand Speech Synthesis is used to convert text to speech.

- **Microsoft Access-2007 Database**

Here we have used Microsoft Access as a database to store the application name i.e. alias and full path in it. All the application in

database may also have alias to open it. The aliases of applications are stored in the grammar file. When user speaks alias of any application, program gets path to that application from database and creates a process of the application.

- **Recognition Grammars (JSGF Java Standard Grammar Format)**

Speech recognition systems provide computers with the ability to listen to user speech and determine what is said. Current technology does not support unconstrained speech recognition: the ability to listen to any speech in any context and transcribe it accurately. To achieve reasonable recognition accuracy and response time, current speech recognizers constrain what they listen for by using grammars. JSGF (Java Speech Grammar Format) is used to recognize the speech from user.

The Java Speech API supports two types of grammars :

1. Rule grammars
2. Dictation grammars

These grammars differ in how patterns of words are defined.

They also differ in their programmatic use: a rule grammar is defined by an application, whereas a dictation grammar is defined by a recognizer and it is built into the recognizer.

A rule grammar is provided by an application to a recognizer to define a set of rules that indicates what a user may say. Rules are defined by tokens, by references to other rules and by logical combinations of tokens and rule references. Rule grammars can be defined to capture a wide range of spoken input from users by the progressive combination of simple grammars and rules.

A dictation grammar defines a set of words (possibly tens or thousands of words) which may be spoken in a relatively unrestricted way.



Dictation grammars are closest to the goal of unrestricted natural speech input to computers. Although dictation grammars are more flexible than rule grammars, recognition of rule grammars is typically faster and more accurate.

### 3.4. Tools and Platforms

We have used Eclipse as an IDE and Windows as a platform for this application.

### 3.5. SPEECH RECOGNITION

Speech recognition can be divided into several phases. In grammar design a recognition grammar is created. It defines the input that the speech recognizer should recognize. In signal processing the incoming audio frequencies are analyzed. In phoneme recognition the incoming audio patterns are compared to language phonemes. Then the sequences of phonemes are compared to words and patterns of words defined by the recognition grammar. The final phase is to provide the interested applications information that has been gathered about the incoming audio.

Recognition results are supplied periodically, usually after one separate command or sentence has been processed. The results are not necessarily unambiguous. Several guesses can be supplied as the result.

The most important task for speech application developers using the Java Speech API is grammar design. Grammars constrain what the speech recognizer has to expect from the user. A grammar or grammars define the current context and sentences that have no relevance in that context can be disregarded.

There are two basic grammar types in the Java Speech API: rule grammars and dictation grammars. As the name indicates, rule grammars are based on rules that constrain the input that is processed.

Input that does not comply with the rules is disregarded. This way the recognition process becomes faster and more accurate. Java Speech API requires that all speech recognizers use rule grammars.

Dictation grammars allow more freedom for the user than rule grammars. Therefore dictation grammars are significantly more complex than rule grammars and usually require the use of statistical data. Dictation grammars have to be built into the speech recognizer product, they cannot be developed using the Java Speech API.

## 4. SYSTEM REQUIREMENT SPECIFICATION

### 4.1. Software Requirement Identification

Software requirement identification is an important part of System Development. Requirement identification makes the system developer know about basic functionalities of system that it must meet. The requirement identification provides the developer and the customer the means to access the quality once the software is built.

### 4.2. Requirements:

- R1 : System should provide facility to add application to database.

Description : User can add application and its alias to the database as an entry.  
Pre-condition : Select “Add” button.  
Input : Alias and absolute path to .exe file.  
Output : System stores Software name, path and alias in database. And also to grammar.

- R2 : System provides facility to remove application from database.

Description : User can remove application’s entry from database.  
Pre-Condition : Select “Remove” button.  
Input : Select Software name from list to remove.  
Output : Data entry removed from database.

- R3 : System provides facility to user to speak the words.

Description : User can speak word to open application.  
Pre-condition : Select “Speak to Open” button.  
Input : “Speak to Open” button.  
Output : System waits for user to speak.

- R4 : System provides facility to open executable application.

Description : User can speak words. When it is recognized the corresponding application is opened.

Pre-condition : User has spoken the word.

Output : Open application if its path is found in database.

#### 4.3. PROBLEMS AND WEAKNESSES OF CURRENT SYSTEM

- The current implementation doesn't provide that much efficient recognition of spoken text. And this reduces accuracy of software.
- Sometimes it recognizes same speech multiple times this causes to open application multiple times.

#### 4.4. USER CHARACTERISTICS

User of this application must have following characteristics.

- User should not be dumb.
- User must have some basic knowledge of computer as well as English.
- While adding the application name to the grammar file, user must provide the filename correctly and in proper case, i.e. case sensitive.

#### 4.5. HARDWARE AND SOFTWARE REQUIREMENTS

- Tools and Technology

Development Technologies : J2SE, JSAPI, Microsoft SAPI 5.1

Development Tools : Eclipse, Microsoft Access-2007 ,  
ODBC Driver

- Hardware Requirements

- Pentium IV or higher
- 1 GB RAM
- 40 GB HDD
- Microphone

- Software Requirements

- Windows XP/Vista/7
- Eclipse IDE
- Microsoft Access-2007

#### 4.6. CONSTRAINTS

- Hardware Limitations

Microphone should be of good quality so that filename is recognized properly.

- Interface to Other Application

The software which we have developed is not dependent on other application but is dependent on the rule grammar defined in the project.

- Higher Order Language Requirement

We require Structured Query Language (SQL) to access database.

- Reliability Requirements

Reliability of the product we develop is a very major requirement. So we ensure that the product which we have developed is reliable.

- Safety And Security Considerations

There is no consideration like safety and security as our project is not using any kind of private data. It is developed keeping in mind the normal user.

- Criticality of Applications

The main criticality of this system is that if the headphone or microphone is not of good quality or if there is lots of noise in the surroundings, then it may not recognize the filename correctly and hence will fail opening that file.

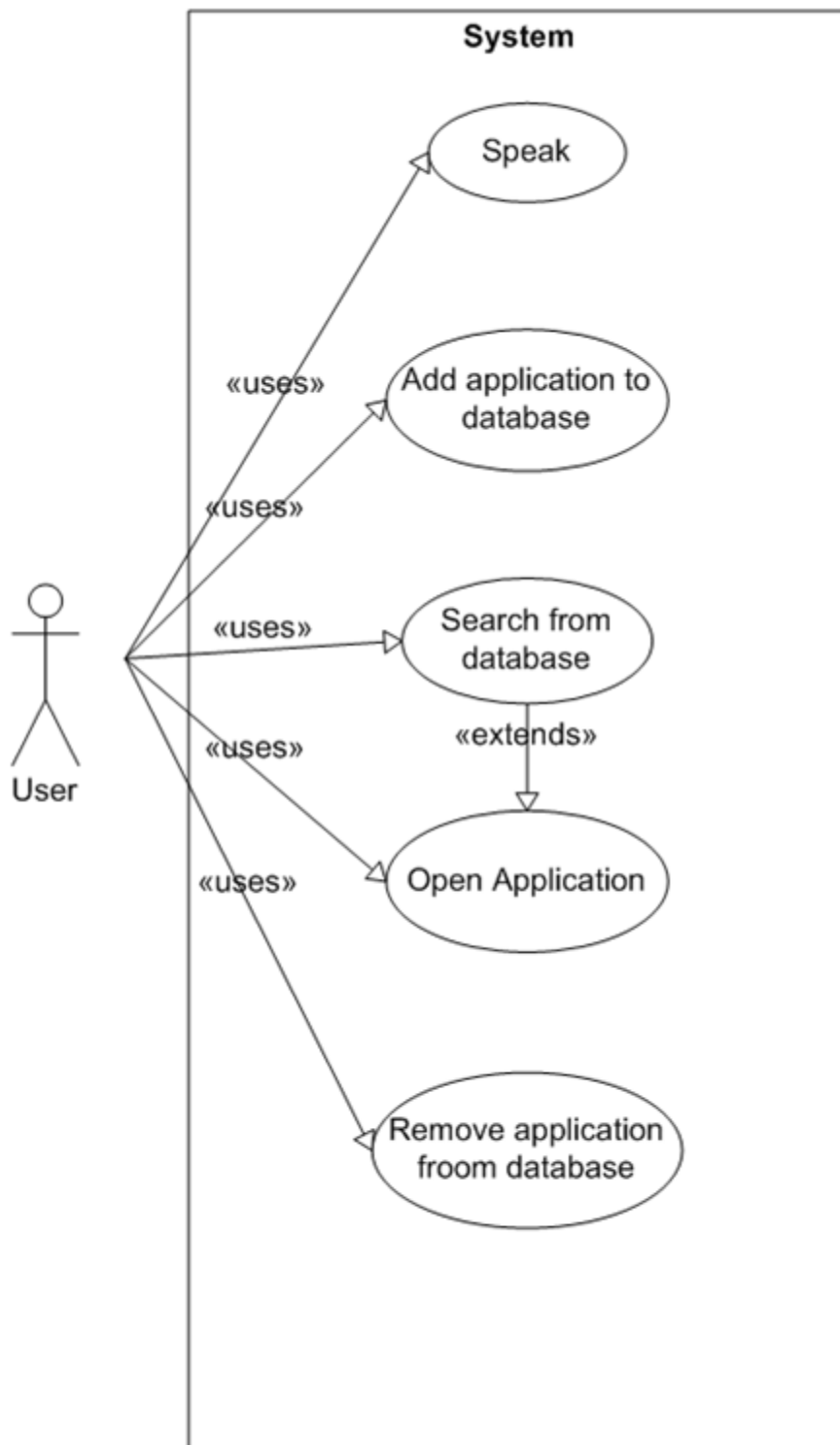
Also if the user doesn't have rights to open that file, System won't open that file.

#### 4.7. ASSUMPTIONS

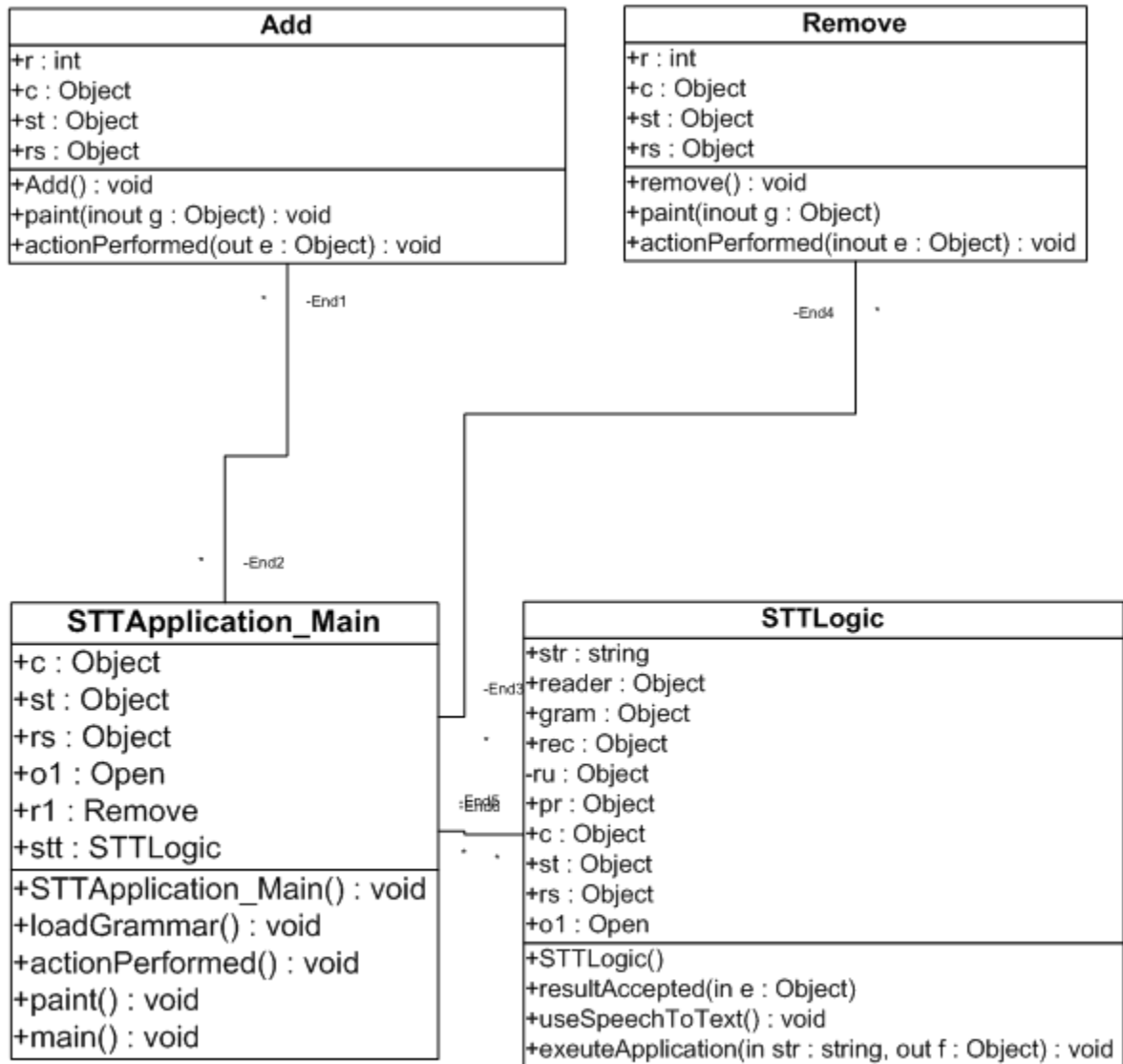
- We assume that user has good quality of Microphone.
- We also assume that user have the sufficient privileges to open the file.

## 5. DESIGN

### 5.1. USE CASE DIAGRAM

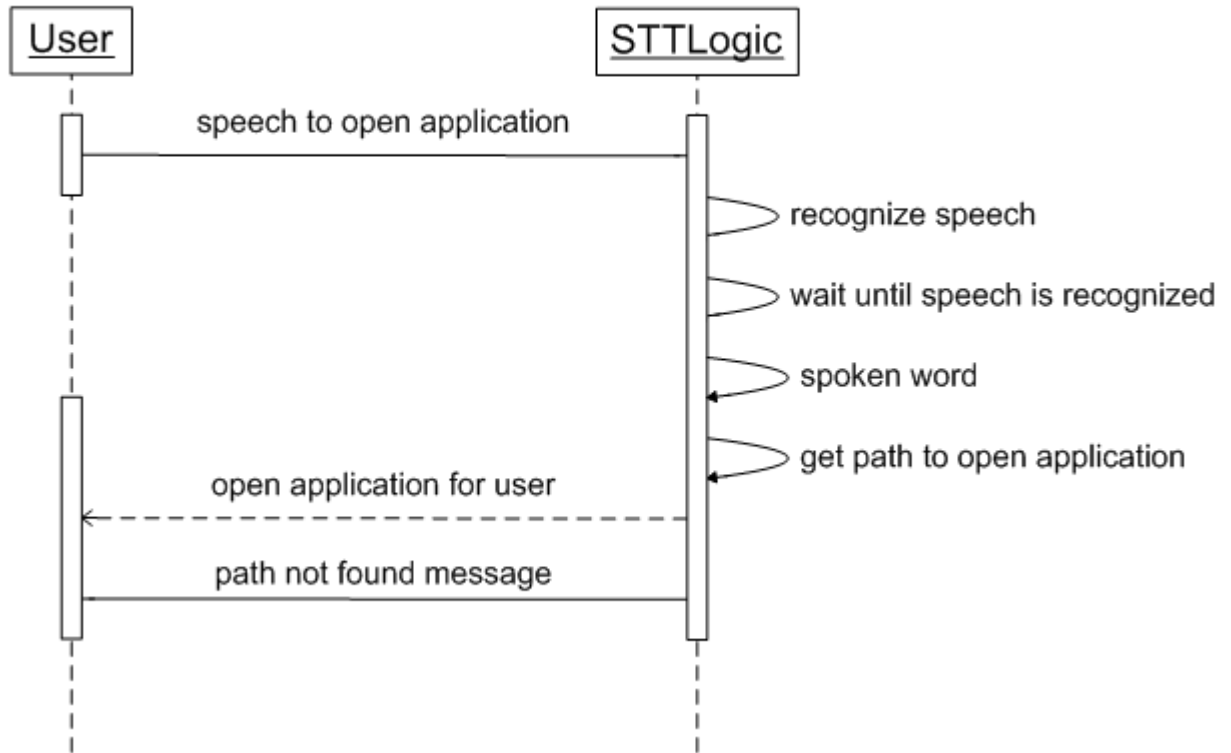


## 5.2. CLASS DIAGRAM

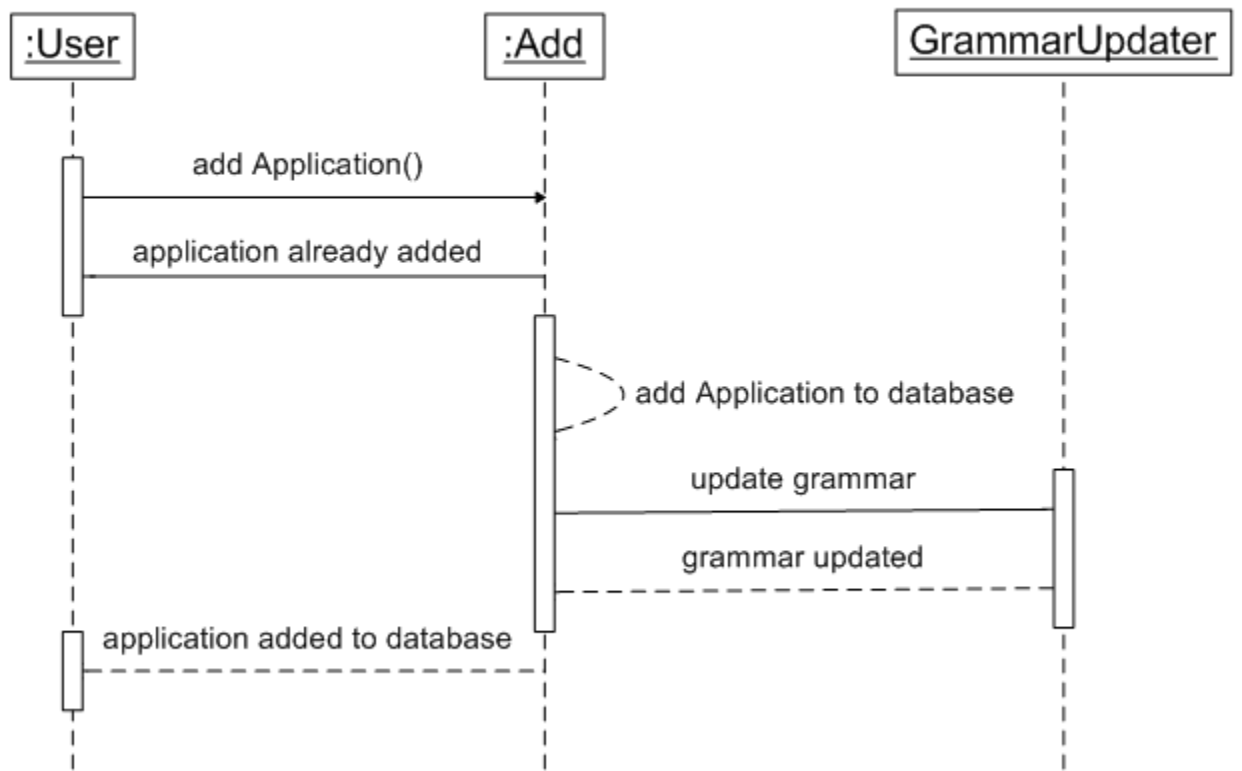




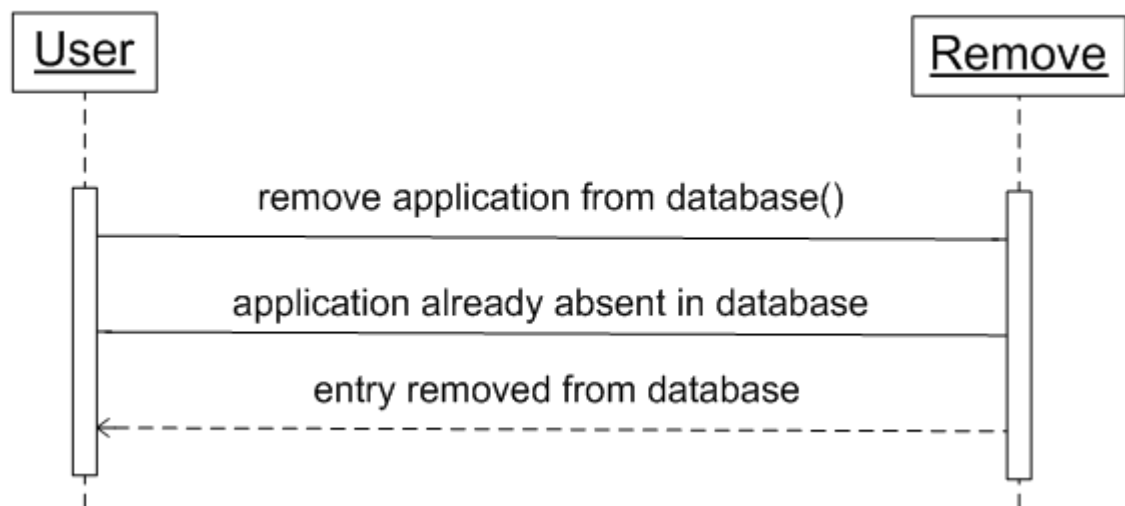
### 5.3. SEQUENCE DIAGRAM



For general behavior

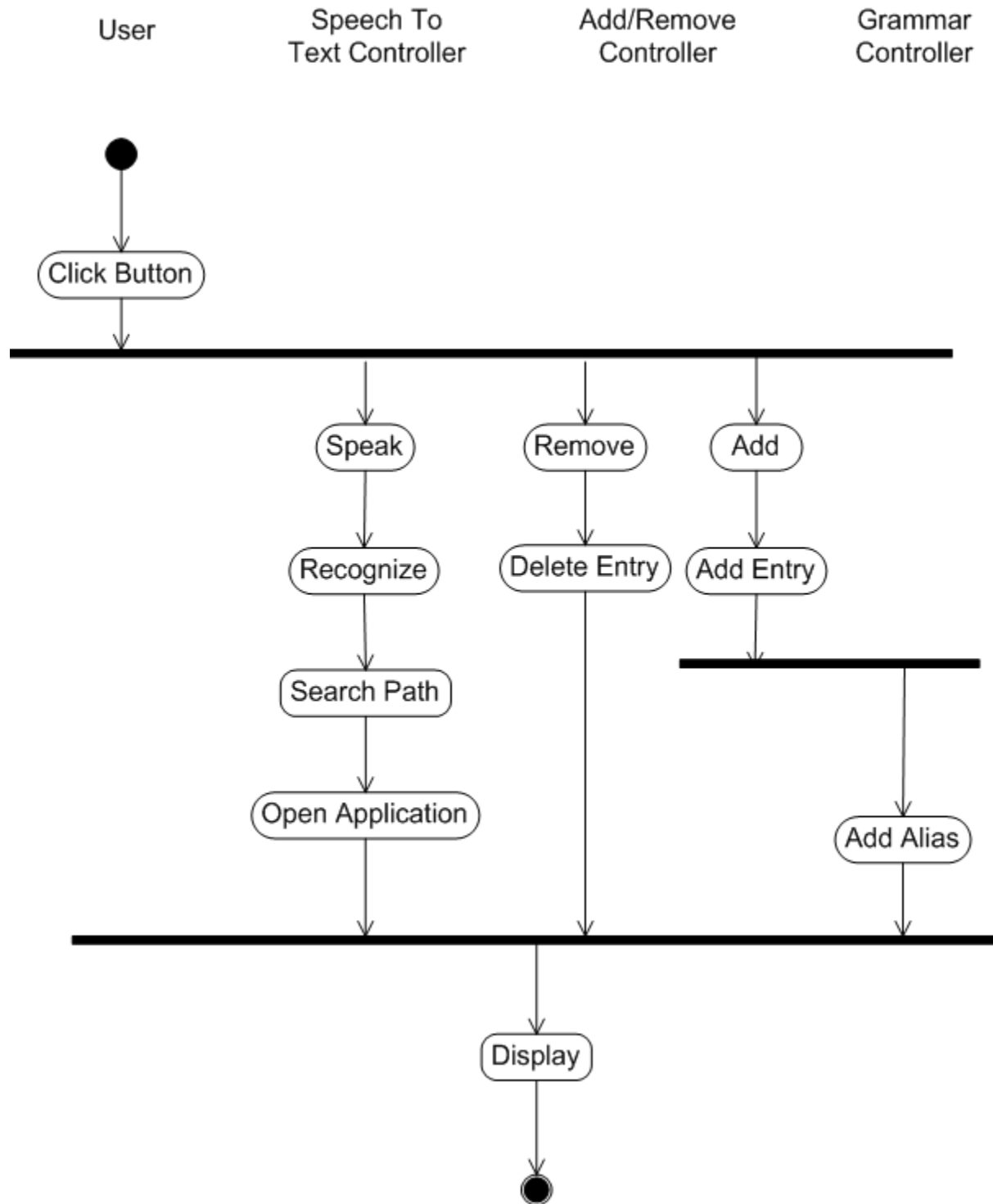


Adding a new application

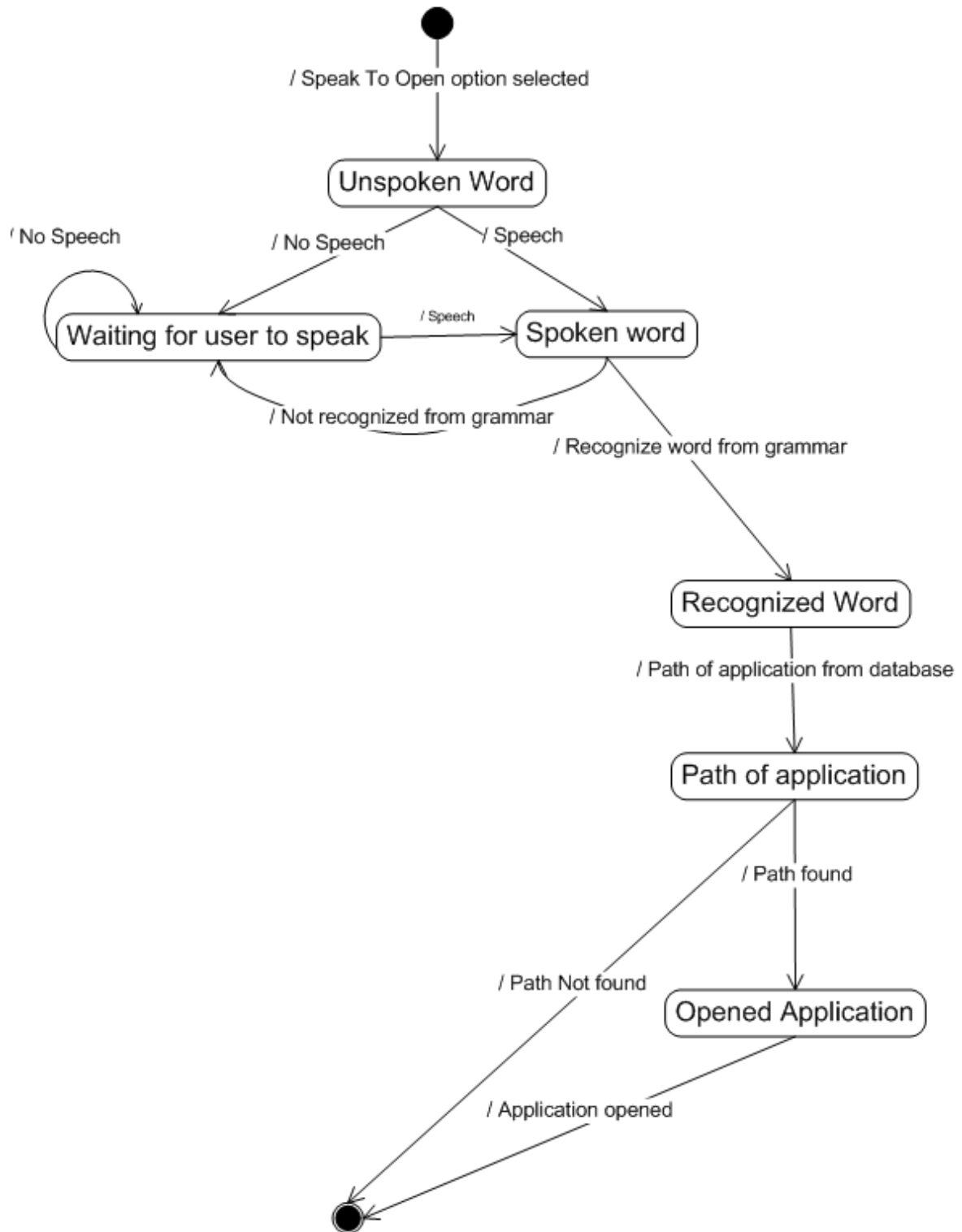


Removing an application

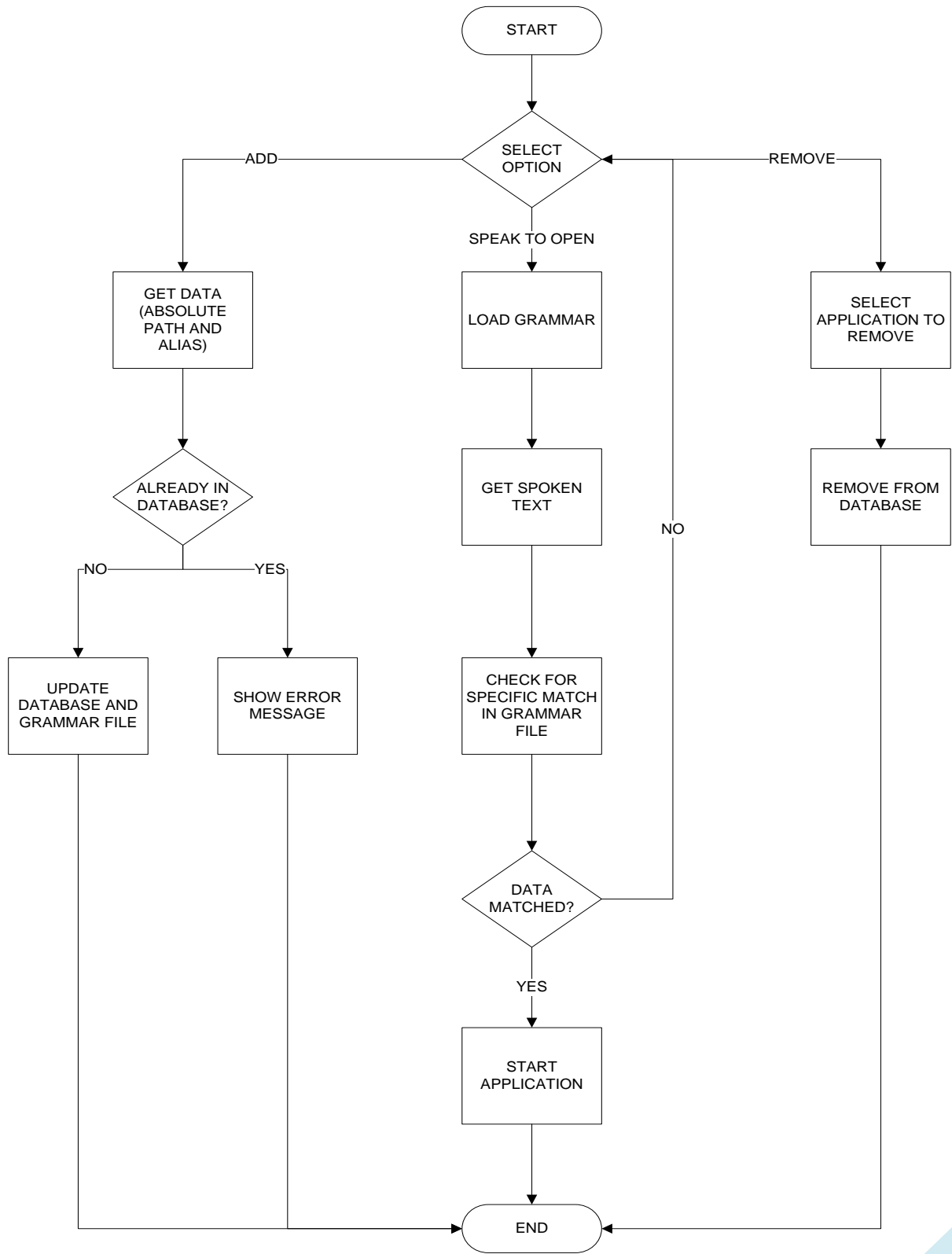
## 5.4. ACTIVITY DIAGRAM



## 5.5. STATE CHART



## 5.6. FLOW CHART



## 6. IMPLEMENTATION

This system is a single user application. It contains GUI that is way to work with this application. This system is GUI based and also allows user to use mouse. Use mouse increases the efficiency of the application.

There are eight different modules in this application to make it work effectively :

1. STTApplication\_Main
2. STTLogic
3. Add
4. Remove
5. Open
6. DialogBox
7. FileDialogBox
8. STTWindowListener

### ➤ STTApplication\_Main

In this module application is initialized. It starts GUI of the application. It provides us information how to use it and also lists the applications added before in database. It also lists the alias for particular application. We can also add or remove application to and from the database. Mainly defined methods in this module are as under :

```
public STTApplication_Main()
```

This is a constructor that places all the GUI components in the Frame and initializes the local variables.

```
public static void loadGrammar(String str)
```

This is a method called when new application is added to the database. The grammar file is updated and all the entries of database are displayed on main GUI.

We create an object of RandomaccessFile and write the alias of new application added to database:

```
RandomAccessFile raf;
```

```
.
```

```
.
```

```
raf=new RandomAccessFile("grammar.txt", "rw");
```

```
raf.seek((raf.length()-1));
```

```
raf.writeBytes(str1);
```

```
raf.close();
```

```
public void paint(Graphics g)
```

This method is used to update the GUI components after any events. Whenever the database is updated this method is called to update the contents which are displayed on the GUI.

```
ResultSet rs;
```

```
.
```

```
.
```

```
rs=st.executeQuery("select      Software.SoftwareName, Software.Alias      from  
Software");
```

```
while(rs.next())
```

```
{
```

```
    sw=rs.getString("SoftwareName");
```

```
    alias=rs.getString("Alias");
```

```
    str=str + alias + "\t\t (for " + sw + " )\n";
```

```
}
```

```
.
```

```
.  
.   
t2.setEditable(true);  
t2.setText(str);  
t2.setEditable(false);
```

```
public void actionPerformed(ActionEvent e)
```

This method is used to perform corresponding action when particular button is pressed.

```
public static void main(String args[])
```

This method initializes the application.

```
Frame f=new STTApplication_Main();  
f.setSize(700,500);  
f.setVisible(true);
```

### ➤ STTLogic

This module is the most important part of the application because it recognizes the speech and converts it to text. According to that text the system opens the corresponding application. This module also contains the method to open the application spoken by user. The methods of this class are :

```
public void resultAccepted(ResultEvent e)
```

This method is invoked when the engine recognizes the speech of user. It identifies the word spoken by the user.

```
Result r=(Result) e.getSource();  
ResultToken tokens[]=r.getBestTokens();  
for(int i=0;i<tokens.length;i++)  
{  
    System.out.println("In result accepted :: "+tokens[i].getSpokenText());
```



```
        str="" + tokens[i].getSpokenText();  
    }
```

```
public void useSpeechToText()
```

This method gets recognizer and loads it to recognize speech. It also loads grammar from grammar reader object file.

```
    Recognizer rec;  
    .  
    .  
    str="";  
    .  
    .  
    EngineList e1=Central.availableRecognizers(null);  
    EngineModeDesc desc=(EngineModeDesc) e1.firstElement();  
    rec=Central.createRecognizer(desc);  
    rec.allocate();  
    .  
    .  
    reader=new FileReader("grammar.txt");  
    gram=rec.loadJSGF(reader);  
    gram.setEnabled(true);  
    rec.addResultListener(new STTLogic());  
    rec.commitChanges();  
    rec.requestFocus();  
    rec.resume();
```

```
public static executeApplication(String str)
```

When the word is recognized system creates a process by using system call to open the application.

```
Process pr;
Runtime ru;

.
.
.

rs=st.executeQuery("select Software.SoftwarePath from Software where
Software.Alias='" + str + "'");
if(rs.next())
{
    str=rs.getString("SoftwarePath");
    ru=Runtime.getRuntime();
    pr=null;
    if(str!=null && (!str.equals("")))
    {
        try
        {
            pr=ru.exec(str);
        }
        .
        .
    }
    .
    .
}
```

### ➤ Add

This module is used to add the new application name and its path to the database. User also has to provide alias of the software because software might not be able to recognize longer or complicated words easily. This system uses Java Speech API version 1.0. Speech technology is newly introduced by Sun in Java. So it is not containing solutions for all possibilities. Mainly defined methods implemented are :

```
public Add()
```

This is constructor to load the GUI components in the Frame.

```
public void paint(Graphics g)
```

This method updates the GUI after some time or event.

```
public void actionPerformed(ActionEvent e)
```

This method is used to perform the action according to the button pressed. If user wants to add an application and he presses button of "Browse" the file then :

```
FileDialogBox fdb;
```

```
.  
.
```

```
fdb=new FileDialogBox(this,"Select Application");
```

```
file=fdb.getFile();
```

```
path=fdb.getDirectory();
```

```
if((file.endsWith(".exe")))
```

```
{
```

```
    t1.setText(path + "" + file);
```

```
    t1.setEditable(false);
```

```
}
```

If user has selected the application and want to add it to database then he should press "Add" button.

```
ResultSet r;
```

```
r=st.executeQuery("select * from Software where Software.SoftwareName='" +  
file + "'");
```

```
if(r.next())
```

```
{
```

```
    // Already present in database
```

```
}
```

```
else
```

```
{
```

```
    rs=st.executeUpdate("insert into Software values('" + file + "', '" +
```

```
t1.getText() + "', '" + t2.getText() + "')");
```

```
    str=t2.getText();
```

```
STTApplication_Main.loadGrammar(str);
System.out.println("str = " + str);
if(rs>0)
{
    data="Data has been updated suceesfully";
    //update GUI by calling repaint()
}
}
```

### ➤ Remove

If we have uninstalled any application from the Computer then we have to remove its entry from table to avoid conflict. So we select the name of the application to remove and remove the whole entry of that application from table. Its defined methods are :

```
public Remove()
```

This is constructor to load the GUI components in the Frame.

```
public void paint(Graphics g)
```

This method updates the GUI after some time or event.

```
public void actionPerformed(ActionEvent e)
```

This method is used to perform the action according to the button pressed. If user selects any application name and presses then that entry for application is removed from database :

```
int r;
if(str.equals("") || str.equals("None"))
{
    data="You have selected " + str + ".";
    try
    {
        Thread.sleep(2000);
    }
}
```

```
        catch(Exception ee)
        {
            ee.printStackTrace();
        }
        repaint();
    }
    else
    {
        .
        .
        r=st.executeUpdate("delete from Software where
        Software.SoftwareName='" + str + "'");
        if(r>0)
        {
            Thread.sleep(2000);
            data="Successfully removed from database.";
            repaint();
        }
        .
        .
    }
}
```

## 7. TESTING

Testing is the process carried out on software to detect the differences between its behavior and the desired behavior as stipulated by the requirements specifications. Testing is advantageous in several ways. Firstly, the defects found help in the process of making the software reliable. Secondly, even if the defects found are not corrected, testing gives an idea as to how reliable the software is. Thirdly, over time, the record of defects found reveals the most common kinds of defects, which can be used for developing appropriate preventive measures such as training, proper design and reviewing.

### 7.1. Testing plan

The testing sub-process includes the following activities in a phase dependent manner :

- Create Test Plans.
- Create Test Specifications.
- Review Test Plans and Test Specifications.
- Conduct tests according to the Test Specifications, and log the defects.
- Fix defects, if any.
- When defects are fixed continue from activity.

### 7.2. Testing methods

In testing of this application we have used white box testing. By including some debugging statements and some extra println to exactly know the execution path.

### 7.3. Table

No	Test Case	Expected Output	Avg. Accuracy (%)
1	Speaking a filename that exists in the grammar file.	Open particular application	60
2	Speaking a filename that is not in the grammar file.	Filename should not be recognized	90
3	User doesn't speak for a considerable time.	Don't do anything	90

The main problem by testing we caught is that is sometimes recognizes same spoken text for more than one time. And it causes the same application to start multiple times. And also the accuracy of current implementation of Java speech API is somewhat less so it affects our recognition part of system.

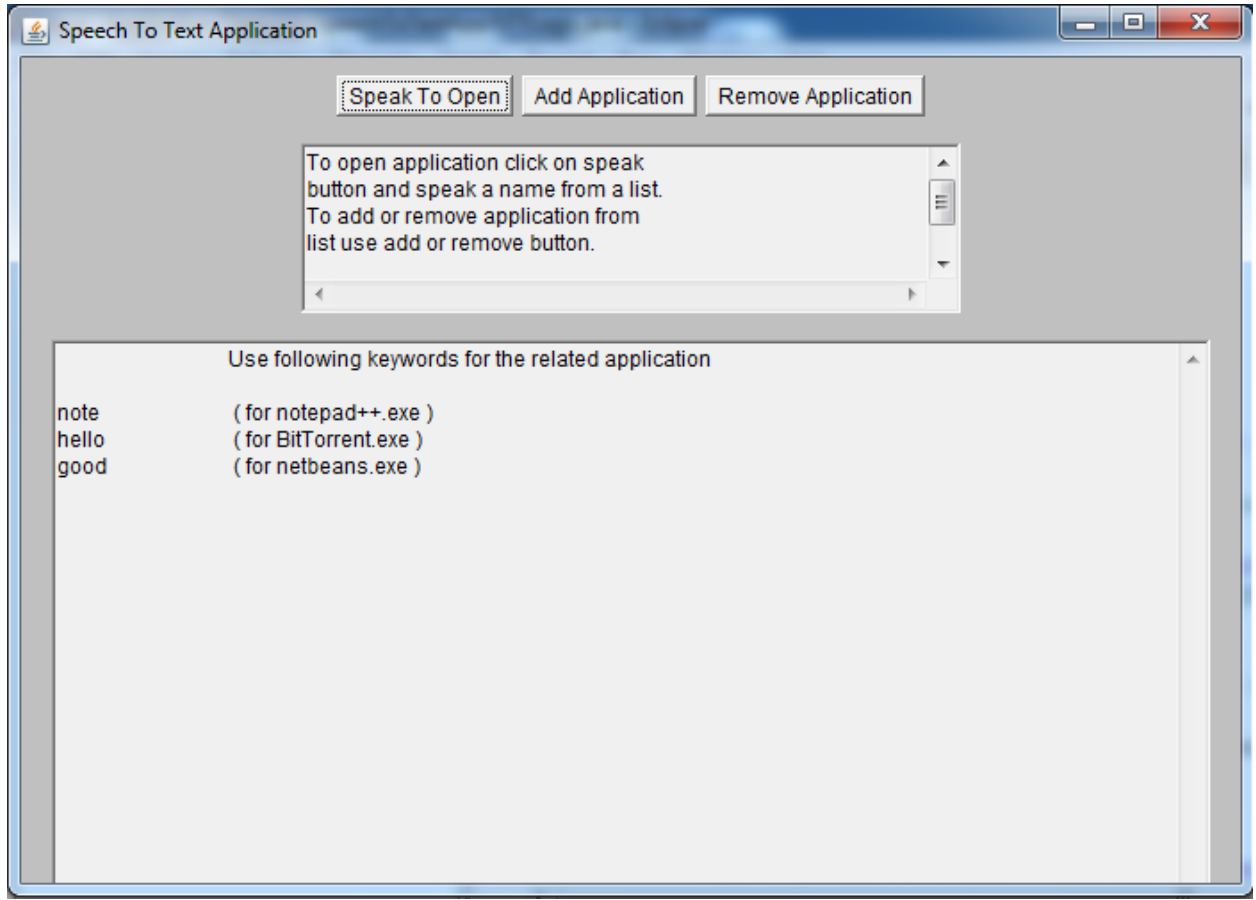
For testing the following words are used:

Word Used	Test case 1	Test case 2	Test case 3
Hello	Executed	Not recognized	Automatically opens
How	Executed	Not recognized	Idle
Good	Not executed	Not recognized	Idle

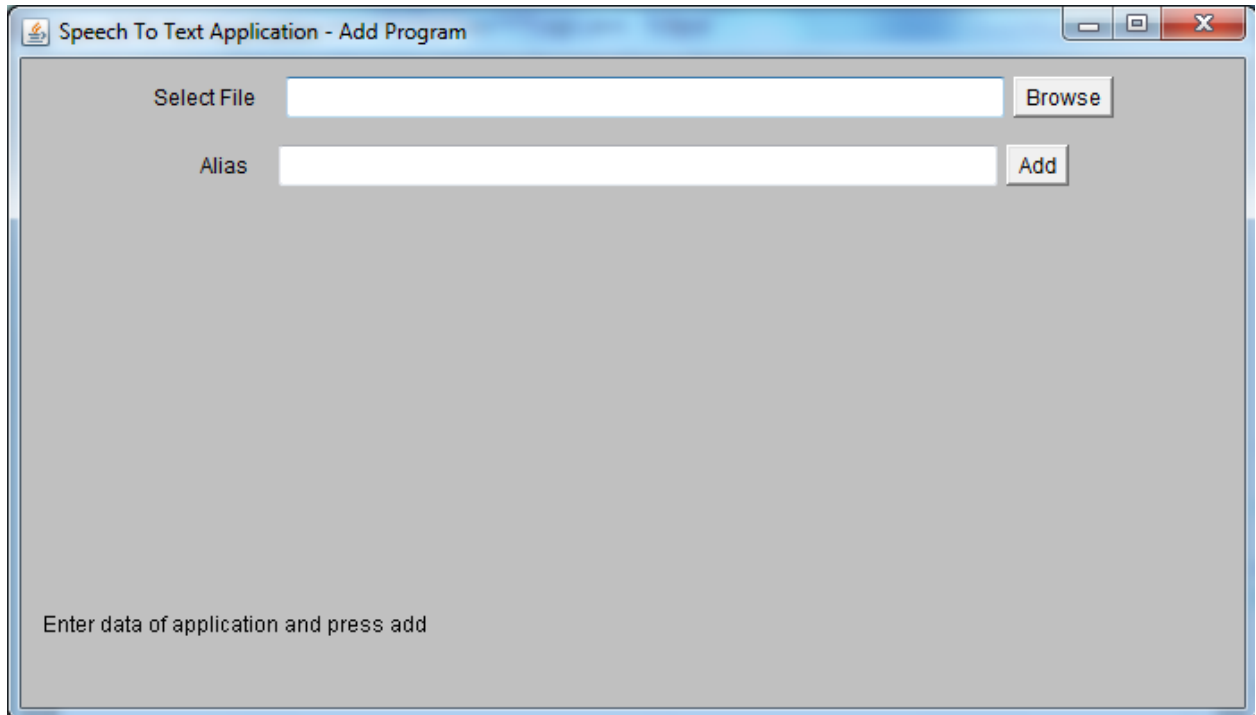
Fine	Not executed	Not recognized	Idle
You	Executed	Not recognized	Idle
Me	Executed	Not recognized	Idle
Bittorrent	Not executed	Not recognized	Idle
Note	Executed	Recognized	Idle
Notepad	Executed	Not recognized	Idle
Avast	Not executed	Not recognized	Idle



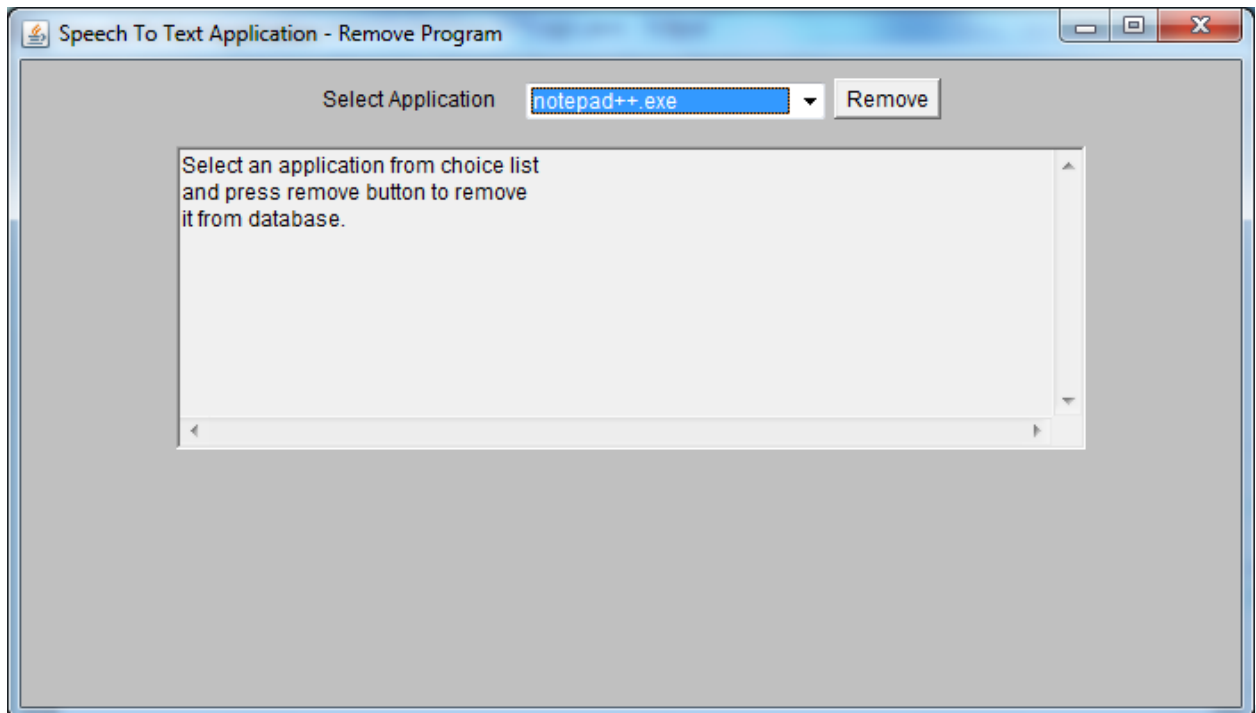
## 8. SCREEN SHOTS



This is main page of our application where you can speak the alias to open any of the listed application



GUI for adding an application to list. Here you have to specify the absolute path to `.exe` and alias for that file.



Screen for removing an application here in drag list you have to select application which you want to remove and press button "Remove".

## 9. FUTURE EXTENSION TO PROJECT

- Software can be error free but cannot be failure free. It can have semantic errors in it. Our system also has some limitation.
  - Filename must contain characters only.
  - It can recognize only English language.
  - User must have rights to open the executable file.
  
- Future extensions of this application are as follows:
  - Whenever a file is removed from database its alias should be removed from grammar file.
  - It should open all types of files except .exe files.

## 10. CONCLUSION

User can use this software to open the application by speaking the name of the alias of the software. User can open application if its path is stored in database and its alias is stored in grammar. User need not to remember whole path of the application.

➤ Advantages of application :

- Less time to open application.
- We need not to use keyboard or mouse to open application.
- Useful for disabled person.

➤ Disadvantage of Application :

- Speech technology is new technology. So, it may have bugs and they have to be identified.
- And as already said accuracy is also less for this implementation.

## 11. References

- Java Speech API guide
- Java Standard Grammar Format (JSGF) guide
- Java Complete Reference
- [java.sun.com](http://java.sun.com)
- [forums.sun.com](http://forums.sun.com)
- Software Engineering by Rajib Mall
- [www.wikipedia.com](http://www.wikipedia.com)
- [www.roseindia.net](http://www.roseindia.net)
- Cloudgarden documentation