JSPHINX

Enrollment No. :	**11103499**	**11104718**	**11103586**
Name	**: Mahima Goel**	**Shivam Tiwari**	**Siddharth Taneja**

Name of supervisor : Mr. Kishore Kumar Y.

December – 2014

Submitted in partial fulfilment in Degree of

Bachelor of Technology

In

Computer Science Engineering/Information Technology

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING & INFORMATION TECHNOLOGY

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

TABLE OF CONTENTS

DECLARATION

We hereby declare that this submission is our own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Place: Noida

Date: Jan 02, 2015

Signature:

Name: Mahima Goel

Enrollment No: 11103499

Signature:

Name: Shivam Tiwari

Enrollment No: 11104718

Signature:

Name: Siddharth Taneja

Enrollment No: 11103586

CERTIFICATE

This is to certify that the work titled "**JSPHINX**" submitted by "**MAHIIMA GOEL, SHIVAM TIWARI AND SIDDHARTH TANEJA**" in partial fulfilment for the award of degree of **B.Tech** of Jaypee Institute of Information Technology University, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor ……..

Name of Supervisor ……..

Designation ……..

Date ……..

ACKNOWLEDGEMENT

In performing our project, we had to take the help and guideline of some respected persons, who deserve our greatest gratitude. We would like to show our gratitude **Mr. Kishore Kumar Y., Assistant Professor, Jaypee Institute Of Information Technology** for giving us a good guideline for the project throughout numerous consultations. We would also like to expand our deepest gratitude to all those who have directly and indirectly guided us in writing this assignment.

Many people, especially our classmates and team members itself, have made valuable comment suggestions on this proposal which gave us an inspiration to improve our project. We thank all the people for their help directly and indirectly in the same.

Signature of the Student	……..
Name of Student	Mahima Goel
Enrollment Number	11103499
Signature of the Student	
Name of Student	Shivam Tiwari
Enrollment Number	11104718
Signature of the Student	……..
Name of Student	Siddharth Taneja
Enrollment Number	11103586

Date : Jan 02, 2015

SUMMARY

JSphinx, in its final stage, will be an API wrapped over CMU Sphinx 4 API. It will provide the required functionality to tackle the problems mentioned in the problem statement while providing abstraction from CMU SPhinx API to the developers. All functions and features of CMU Sphinx 4 will be relayed across and additional features will be added.

Machine Learning and training algorithms for phonetics identification are already present in CMU Sphinx 4. Machine Learning and training algorithms for accent recognition, sharing training set across devices, etc will be in the JSphinx module.

We will, primarily, use Java for code development and Maven architecture. In its final stage we would convert it into a Sonatype OSS repository API.

LIST OF FIGURES

LIST OF TABLES

Chapter 1. INTRODUCTION

1.1 GENERAL INTRODUCTION

JSphinx is a development over CMU Sphinx which integrates phonetic analysis & speaker identity with already existing speech to text API. This is intended to solve two problems

1. **Provide a common speaker identity**
   This is intended to share and sync the learning/training data on one device with all other devices of the same user.

2. **Improve speech to text performance across cultures and geographies**
   People from various cultures and geographies have different accents. Any modern speech to text program can be gradually trained to cater to accent of the user. But it takes time to train and personalize the speech to text program. Keeping a track of distinctive phonetic signatures mapped to similar learning sets can help in faster training of such programs.

1.2 CURRENT RELEVANT PROBLEMS

A few current open problems that we were able to pin point are

1. Standardization of speech to text API and establishing a common universal speaker identity wouldn't be easy as applications are vendor specific.
2. Creating a database to train and test JSphinx is a tedious task considering the variations in accents that are essential to this project.

1.3 PROBLEM STATEMENT

Speech to text applications are changing the way we perceive, control and interact with IT products around us. Using techniques like Machine Learning and AI to train the application and personalize it to user's tone we have increased the usability of such applications. However, there are two main problems with such systems.

1. If a user uses speech to text applications on N devices, then he trains each one of them independently to suit his accent. There is no universal speaker identity which allows to share learning data across devices.

2. Each user trains his own device. Even though many users from same culture of geolocations have similar accents there is no way to use this fact to increase the performance of speech to text applications.

Through JSphinx we will try to solve these two problems.

1.4 OVERVIEW OF SOLUTION

CMU Sphinx already applies Machine Learning techniques in the speech to text application and trains itself according to a user's accent. To solve the two problems mentioned above we will encapsulate CMU Sphinx in JSphinx. CMU Sphinx Engine (and database) will be accessed through the API and will run on the CMU architecture. JSphinx will interface with it as an inside module. It will have its own engine to extract signature phonetics and choose best fit text for a particular speech. It will use Machine learning and AI to achieve this. This data will be stored in JSphinx database and will run on our architecture.

In basic terms JSphinx uses CMU SPhinx as a module for speech to text processing and solves the two problems mentioned in the problem statement itself.

CHAPTER 2. BACKGROUND STUDY

2.1 LITERATURE SURVEY

2.1.1 SUMMARY OF PAPERS

Paper 1

Title	:	Environmental Robustness in Automatic Speech Recognition
Authors	:	Alejandro Acero, Richard M. Stern
Year Of Publication	:	1990
Publishing Details	:	Department of Electrical and Computer Engineering and School of Computer Science, Carnegie Mellon University, Pittsburgh , Pennsylvania

Summary : Efforts made to increase the robustness of CMU to the environment, two algorithms were proposed. First, SDCN, which adds a correction vector depending on the instantaneous SNR response of the input.Second, CDCN,uses a maximum likelihood technique to estimate noise and spectral tilt in the context of an iterative algorithm.An alphanumeric database was tested on two microphones ,a close talking and a desk-top microphone.

Weblink : Paper 1

Paper 2

Title	:	The 1999 CMU 10X Real Time Broadcast News Transcription System
Authors	:	Mosur Ravishankar, Rita Shankar, Bhiksha Raj, Richard M.S.
Year Of Publication	:	2000
Publishing Details	:	Department of Electrical and Computer Engineering and School of Computer Science, Carnegie Mellon University, Pittsburgh , Pennsylvania

Summary	:	Describes the architecture of the CMU-SPHINX-III fast decoder and the various components of the recognition system. The two models,full bandwidth and narrow bandwidth,were and the primary was tested. Various components of the CMU system,namely,Signal processing,Segmentation,Acoustic models,Language Models,were described and instantiated.
Web Link	:	Paper 2

Paper 3

Title	:	Speech Recognizer-Based Microphone Array Processing For Robust Hands-Free Speech Recognition
Authors	:	Michael L.S. , Bhiksha Raj, Richard M.S.
Year Of Publication	:	2002
Publishing Details	:	1)Department of Electrical and Computer Engineering and School of Computer Science, Carnegie Mellon University, Pittsburgh , Pennsylvania 2) Mitsubishi Electric Research Labs, Cambridge, USA
Summary	:	A new array processing algorithm for microphone array speech recog.. The new algorithm uses an objective function which utilizes information from the recognition system itself to optimize the parameters of a filter-and-sum array processor We see a 36% improvement of the new algorithm.
Web Link	:	Paper 3

Paper 4

Title	:	HMM-Based Audio Keyword Generation
Authors	:	Min Xu, Ling-Yu Duan, Jianfei Cai, Liang-Tien Chia, Changsheng Xu and Qi Tian
Year Of Publication	:	2004
Publishing Details	:	1) School of Computer Engineering, Nanyang Technological University, Singapore 2) Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore
Summary	:	Proposed classification method based on Hidden Markov Model (HMM) for audio keyword identification as an improved work instead of using SVM classifier. Unlike the frame-based SVM classification, HMM-based classifiers treat specific sound as a continuous time series data and employ hidden states transition to capture hidden information.
Web Link	:	Paper 4

Paper 5

Title	:	Minimum Variance modulation filter for robust speech Recognition
Authors	:	Yu-Hsiang Bosco Chiu and Richard M Stern
Year Of Publication	:	2009
Publishing Details	:	Dept of Electrical and Computer Engineering and Language Technologies Institute, Carnegie Mellon University, Pittsburgh PA, USA

Summary	:	Designing a modulation filter by data driven analysis which improves the performance of automatic speech recognition systems that operate in real environments. Recognition accuracy is measured using CMU-SPHINX III and DARPA Resource Management and Wall Street Journal speech corpus for testing and training.
Weblink	:	Paper 5

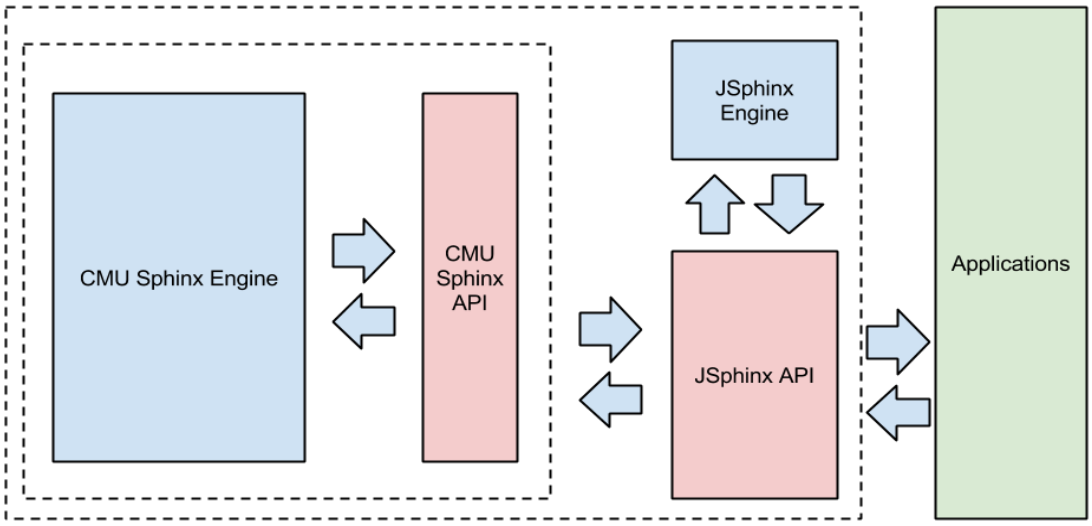
Paper 6

Title	:	Automatic Generation Of Subword Units For Speech Recognition Systems
Authors	:	Rita Singh, Bhiksha Raj, and Richard M.S.
Year Of Publication	:	2002
Publishing Details	:	IEEE Transactions On Speech And Audio Processing,Vol No. 2. February 2002
Summary	:	A complete probabilistic formulation for the automatic design of subword unit and dictionary,given only the acoustic data and their transcriptions. The problem of automatically designing the subword units and the dictionary given only a set of acoustic signals and their transcriptions,is met by forming a modeling perspective viz. identifying the sound class that best fit the training set.
Weblink	:	Paper 6

2.1.2 INTEGRATED SUMMARY OF LITERATURE

After scanning 25 years of literature of CMU Sphinx and looking into a couple of more projects we feel that speech to text applications have reached an appreciable development stage. It is time that we create a universal unique speaker identity for cross-domain applications for supporting portability and sharing the learning data across devices. So instead of having many stand-alone speech to text training instances we could just share all this data to make the process faster and more efficient.

To achieve this we plan to create a wrapper class around CMU Sphinx API that interfaces with a database that stores user's phonetic signatures to mark his accent against group learning data for similar phonetics. It will also be mapped to his/her universal speaker identity.



2.1.3 COMPARISON OF EXISTING APPROACHES WITH PROBLEM STATEMENT.

Parameters	Dragon Home	Voice Finger	ViaTalk	Tazti
Accuracy Score	88%	86%	64%	72%
Voice Profile	Yes	Yes	Yes	No
Voice Training	Yes	Yes	No	No
Additional Training	Yes	Yes	No	No
Accent Support	Yes	Yes	No	No
Microphone Status Icon	Yes	Yes	Yes	No
Compatible with Bluetooth speakers	No	No	No	No
Voice Transcription	No	No	Yes	No
FAQ	Yes	No	Yes	Yes
Tutorial and Demos	Yes	No	No	No
User Manual or Guide	Yes	No	No	Yes
Open and close programs	Yes	Yes	Yes	Yes
Web Search	Yes	Yes	Yes	Yes

Table 2.1

2.2 DETAILS OF EMPIRICAL STUDY

2.2.1 FIELD SURVEY

We asked around people for feedback on Siri, Okay Google and Windows Speech Recognition. We tried to get an idea of things they liked and disliked about their speech recognition apps.

2.2.2 EXPERIMENTAL STUDIES

No experimental studies have been conducted in this research yet.

2.2.3 NEW TOOLS AND TECHNOLOGIES

We have used the following new tools and technologies in this project so far

1. Maven
   http://maven.apache.org/

2. CMU-Sphinx
   http://cmusphinx.sourceforge.net/

3. Sonatype-OSS
   https://oss.sonatype.org/

CHAPTER 3. ANALYSIS, DESIGN AND MODELLING

3.1 REQUIREMENT SPECIFICATIONS

JSphinx, in its final stage, will be an API wrapped over CMU Sphinx 4 API. It will provide the required functionality to tackle the problems mentioned in the problem statement while providing abstraction from CMU SPhinx API to the developers. All functions and features of CMU Sphinx 4 will be relayed across and additional features will be added.

Machine Learning and training algorithms for phonetics identification are already present in CMU Sphinx 4. Machine Learning and training algorithms for accent recognition, sharing training set across devices, etc will be in the JSphinx module.

We will, primarily, use Java for code development and Maven architecture. In its final stage we would convert it into a Sonatype OSS repository API.

3.2 FUNCTIONAL AND NON FUNCTIONAL REQUIREMENTS

3.2.1 NON FUNCTIONAL REQUIREMENTS

Requirement 1

Requirement ID : NFR001

Requirement Name : Apache Server-enabled backend server space

Requirement Description : To run the API we will require linux web hosting running APache Server on it because Maven architecture guidelines work best on Apache servers.

Requirement 2

Requirement ID : NFR002

Requirement Name : Web client hosted on web

Requirement Description : To run the application using a web browser we will need to create a website that acts as a web client and interfaces with the API running in NFR001.

Requirement 3

Requirement ID : NFR003

Requirement Name : Mobile applications running locally on devices

Requirement Description : To demonstrate the portability of this project across platforms we need to create mobile applications for major manufacturers viz. iOS, Android and Windows Phone.

Requirement 4

Requirement ID : NFR004

Requirement Name : Training data

Requirement Description : Training data from other similar projects need to be extracted and used to teach JSphinx before it is tested on real data set.

Requirement 5

Requirement ID : NFR005

Requirement Name : Audio Samples

Requirement Description : To test this project we would require Audio samples of different accents. We will need to take these samples across cultures and geo locations to test the program on as many distinctive accents as possible.

Safety requirements

System should keep the safety of user data in mind and should not cause any harm to it.

Security requirements

The user data (voice samples) should be kept secure from outside intervention. The user database should be properly scrutinized to prevent any sort of hacking issues that are common in such cases.

Error handling

The user data is properly expertised in handling user data errors. Both expected and non-expected errors can be prevented before the data is fed to the exception handler module.

Reliability

The data is already reliable as it has been taken from speaker.

Correctness

The results obtained from the training set are then matched with the documented results available. This ensures the relative correctness of the output.

3.2.2 FUNCTIONAL REQUIREMENTS

The functional requirements are:

1. Accent switching within the text
2. Switching amongst multiple voices
3. Recognising words in various accents
4. Text grammar Check

3.3 ARCHITECTURE

JSphinx uses a modified MVC architecture. The components of this architecture are shown in the diagram below.

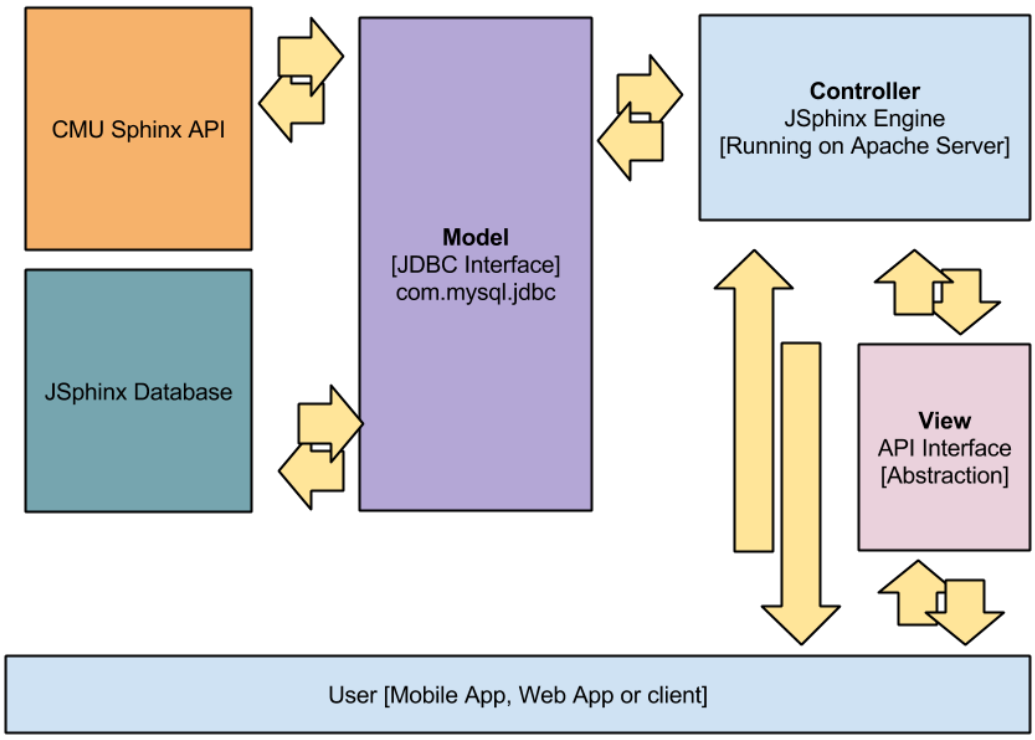


Figure 3.1

3.5 RISK ANALYSIS AND MITIGATION

Risk ID	Risk Description	Risk Area	Probability	Impact	Mitigation [Y/N]	Mitigation plan	Contingency plan
R001	Lack of training data	Resources	0.6	1	Y	Build our own training set	NA
R002	Confusing accent signatures	Algorithm	0.4	0.5	N	NA	Change algorithm
R003	Data Management Fails	Database	0.6	1	Y	USe big data methods instead of usual database	NA

Table 3.1

CHAPTER 4. IMPLEMENTATION AND TESTING

4.1 IMPLEMENTATION DETAILS AND ISSUES

We have been able to implement JSphinx desktop variant in Java without GUI. The following approach was used.

4.1.1 ARCHITECTURE OF CMU SPHINX

The CMU Sphinx architecture running in the backend of JSphinx is as follows when using an audio file as an input.

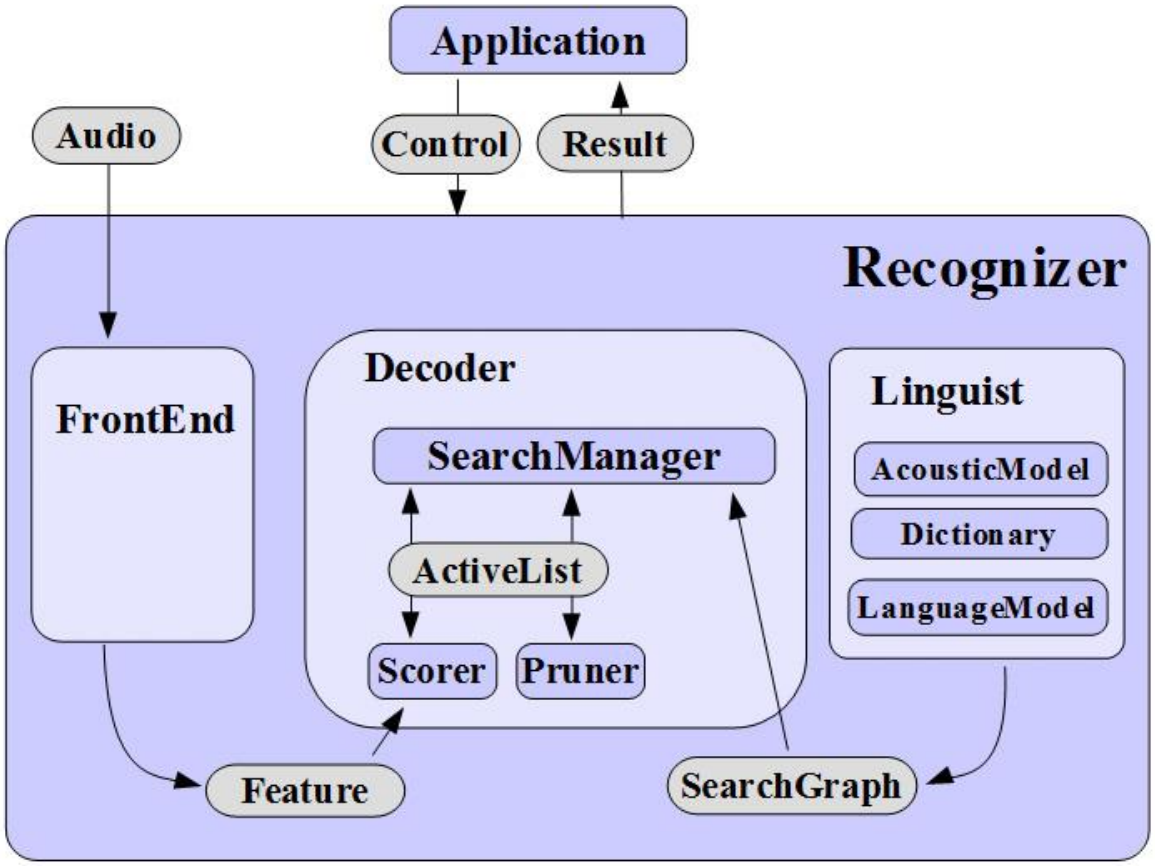


Figure 4.1

The application only interfaces to the Recognizer and the audio file is an external input. However, for turning this into a live speech to text application a feedback in form of instrumentation module needs to be added as follows.

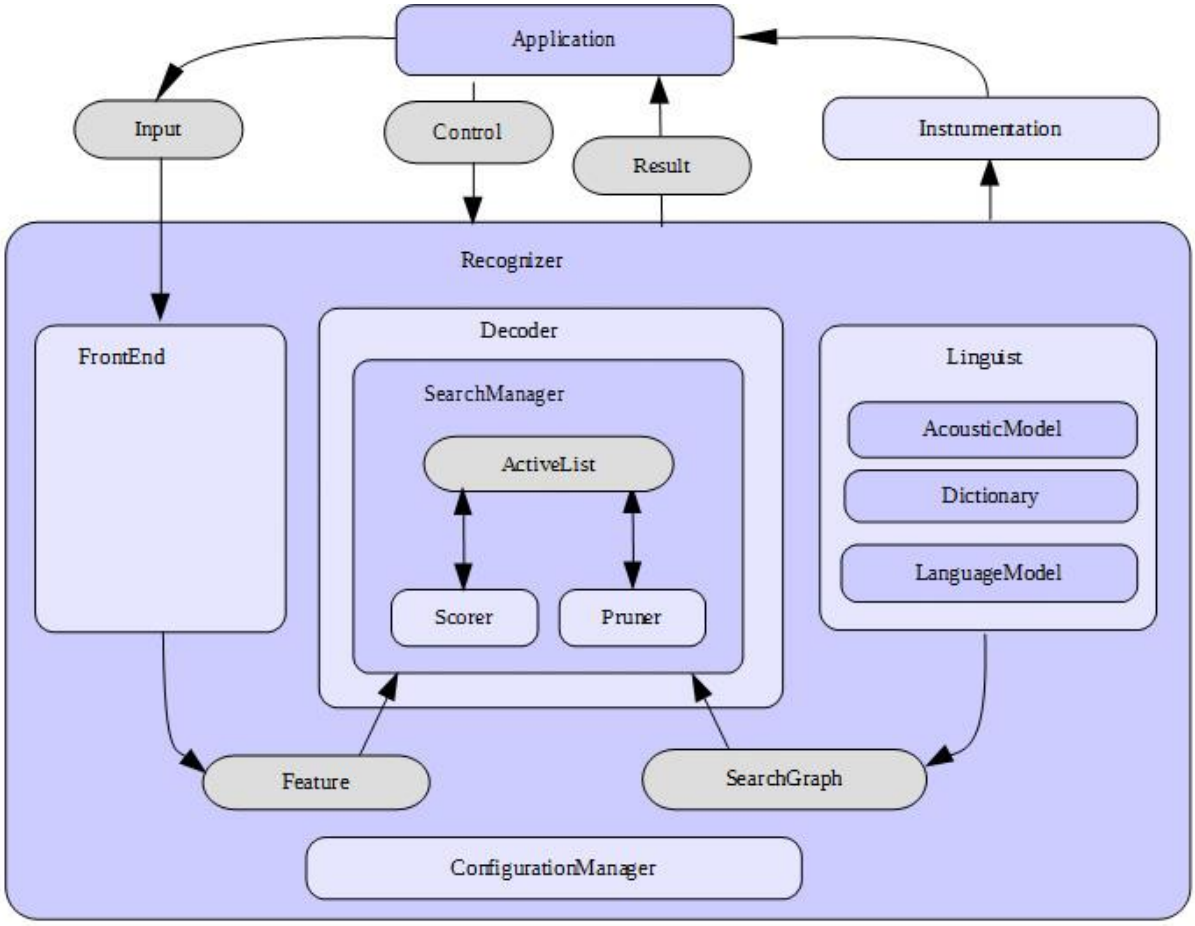


Figure 4.2

The instrumentation provides a feedback to the frontend application to decide upon providing timeslots for the user to speak to provide input and read text to receive output.

4.1.2 ARCHITECTURE OF JSPHINX

JSphinx is a project written to interface to CMU Sphinx. Sphinx4 written in Java was used and compiled locally by us to eliminate the use of CMU Sphinx as a web service. The architecture of JSphinx Desktop is as follows

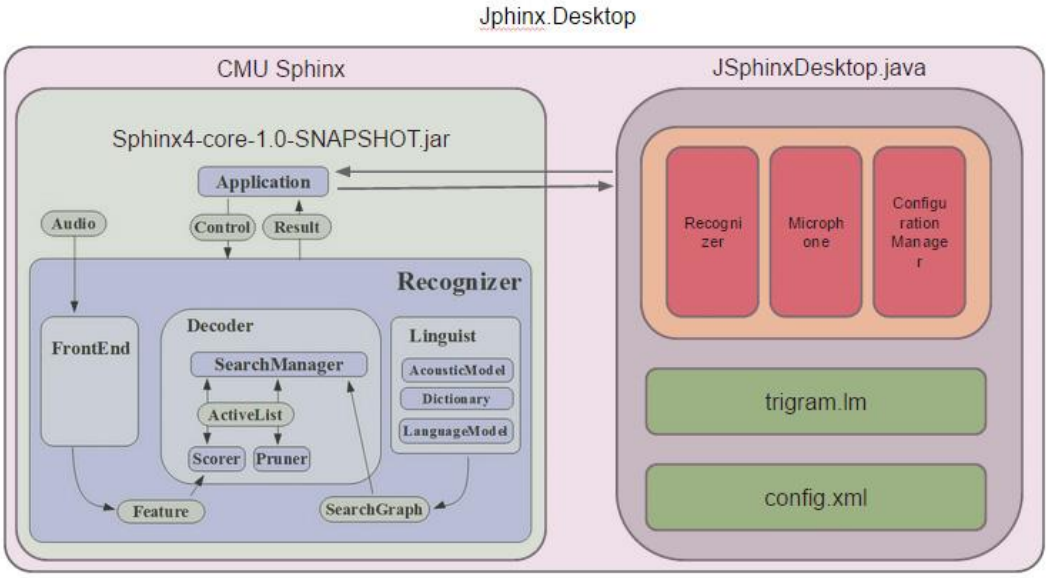


Figure 4.3

4.1.3 SOURCE FILES OF CMU SPHINX [Sphinx4 Java]

The packages used in CMU Sphinx are listed below

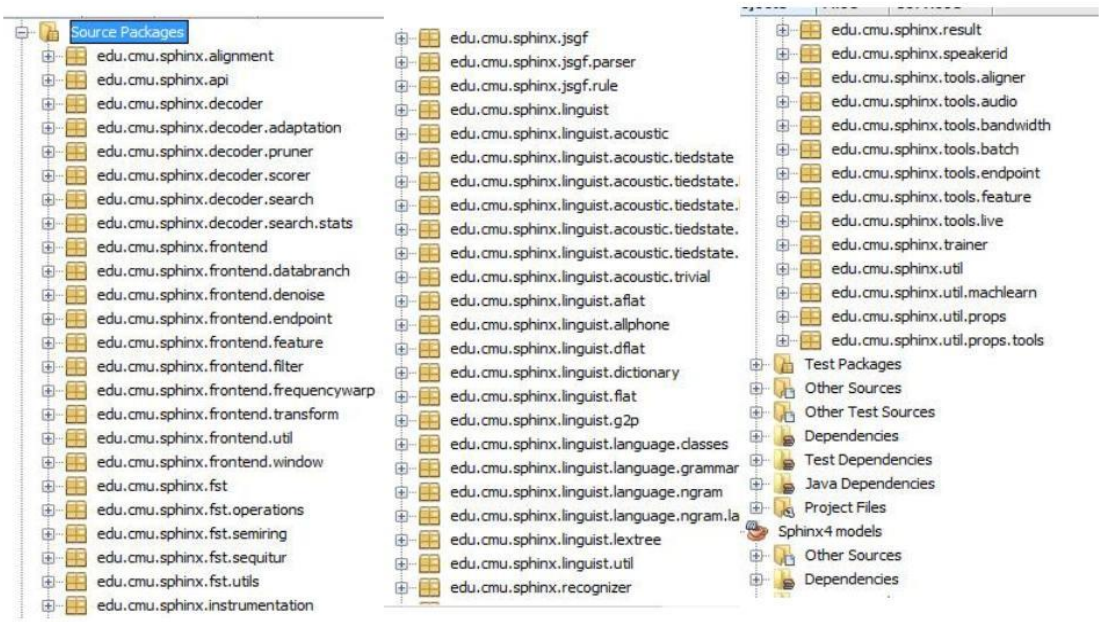


Figure 4.4

4.1.4 SOURCE FILES IN JSPHINX DESKTOP

JSphinxDesktop.java

```
/*
* To change this license header, choose License Headers in Project Properties.
* To change this template file, choose Tools | Templates
* and open the template in the editor.
*/
package jsphinx.desktop;


import edu.cmu.sphinx.frontend.util.Microphone;
import edu.cmu.sphinx.recognizer.Recognizer;
import edu.cmu.sphinx.result.Result;
import edu.cmu.sphinx.util.props.ConfigurationManager;

public class JSphinxDesktop {

  /**
   * @param args the command line arguments
   */
  public static void main(String[] args) {
    // TODO code application logic here

    ConfigurationManager cm;

    if (args.length > 0) {
      cm = new ConfigurationManager(args[0]);
    } else {
      cm = new
ConfigurationManager(JSphinxDesktop.class.getResource("JSphinxDesktop.config.xml"));
    }

    // allocate the recognizer
      System.out.println("Loading...");
    Recognizer recognizer = (Recognizer) cm.lookup("recognizer");
    recognizer.allocate();

    // start the microphone or exit if the programm if this is not possible
    Microphone microphone = (Microphone) cm.lookup("microphone");
    if (!microphone.startRecording()) {
      System.out.println("Cannot start microphone.");
      recognizer.deallocate();
      System.exit(1);
    }

      printInstructions();

    // loop the recognition until the programm exits.
```

```
    while (true) {
      System.out.println("Start speaking. Press Ctrl-C to quit.\n");

      Result result = recognizer.recognize();

      if (result != null) {
        String resultText = result.getBestFinalResultNoFiller();

        System.out.println("You said: " + resultText + '\n');
            if("alone".equals(resultText)){
         System.out.println("Do not be afraid, baby!");}

      } else {
        System.out.println("I can't hear what you said.\n");
      }
    }
  }

  private static void printInstructions() {
        System.out.println("Sample sentences:\n\n" +
          "the green one right in the middle\n" +
                "the purple one on the lower right side\n" +
                "the closest purple one on the far left side\n" +
                "the only one left on the left\n\n\n");
  }
}
```

JSphinxDesktop.config.XML

```
<?xml version="1.0" encoding="UTF-8"?>

<!--
 Sphinx-4 Configuration file
-->

<!-- ******************************************************** -->
<!-- biship configuration file                                -->
<!-- ******************************************************** -->

<config>
  <!-- ******************************************************** -->
  <!-- frequently tuned properties                              -->
  <!-- ******************************************************** -->
  <property name="absoluteBeamWidth" value="5000"/>
  <property name="relativeBeamWidth" value="1E-120"/>
  <property name="absoluteWordBeamWidth" value="200"/>
  <property name="relativeWordBeamWidth" value="1E-80"/>
  <property name="wordInsertionProbability" value="0.7"/>
  <property name="languageWeight" value="10.5"/>
```

```
<property name="silenceInsertionProbability" value=".1"/>
<property name="frontend" value="epFrontEnd"/>
<property name="recognizer" value="recognizer"/>
<property name="showCreations" value="false"/>


<!-- ********************************************************* -->
<!-- word recognizer configuration                             -->
<!-- ********************************************************* -->

<component name="recognizer"
                type="edu.cmu.sphinx.recognizer.Recognizer">
  <property name="decoder" value="decoder"/>
  <propertylist name="monitors">
    <item>accuracyTracker </item>
    <item>speedTracker </item>
    <item>memoryTracker </item>
    <item>recognizerMonitor </item>
  </propertylist>
</component>

<!-- ********************************************************* -->
<!-- The Decoder configuration                                 -->
<!-- ********************************************************* -->

<component name="decoder" type="edu.cmu.sphinx.decoder.Decoder">
  <property name="searchManager" value="wordPruningSearchManager"/>
  <property name="featureBlockSize" value="50"/>
</component>

<!-- ********************************************************* -->
<!-- The Search Manager                                        -->
<!-- ********************************************************* -->

<component name="wordPruningSearchManager"
type="edu.cmu.sphinx.decoder.search.WordPruningBreadthFirstSearchManager">
  <property name="logMath" value="logMath"/>
  <property name="linguist" value="lexTreeLinguist"/>
  <property name="pruner" value="trivialPruner"/>
  <property name="scorer" value="threadedScorer"/>
  <property name="activeListManager" value="activeListManager"/>
  <property name="growSkipInterval" value="0"/>
  <property name="checkStateOrder" value="false"/>
  <property name="buildWordLattice" value="false"/>
  <property name="acousticLookaheadFrames" value="1.7"/>
  <property name="relativeBeamWidth" value="${relativeBeamWidth}"/>
</component>


<!-- ********************************************************* -->
```

```
<!-- The Active Lists                                   -->
<!-- ********************************************************** -->

<component name="activeListManager"
        type="edu.cmu.sphinx.decoder.search.SimpleActiveListManager">
    <propertylist name="activeListFactories">
            <item>standardActiveListFactory</item>
            <item>wordActiveListFactory</item>
            <item>wordActiveListFactory</item>
            <item>standardActiveListFactory</item>
            <item>standardActiveListFactory</item>
            <item>standardActiveListFactory</item>
        </propertylist>
</component>

<component name="standardActiveListFactory"
        type="edu.cmu.sphinx.decoder.search.PartitionActiveListFactory">
    <property name="logMath" value="logMath"/>
    <property name="absoluteBeamWidth" value="${absoluteBeamWidth}"/>
    <property name="relativeBeamWidth" value="${relativeBeamWidth}"/>
</component>

<component name="wordActiveListFactory"
        type="edu.cmu.sphinx.decoder.search.PartitionActiveListFactory">
    <property name="logMath" value="logMath"/>
    <property name="absoluteBeamWidth" value="${absoluteWordBeamWidth}"/>
    <property name="relativeBeamWidth" value="${relativeWordBeamWidth}"/>
</component>

<!-- ********************************************************** -->
<!-- The Pruner                                         -->
<!-- ********************************************************** -->
<component name="trivialPruner"
        type="edu.cmu.sphinx.decoder.pruner.SimplePruner"/>

<!-- ********************************************************** -->
<!-- TheScorer                                          -->
<!-- ********************************************************** -->
<component name="threadedScorer"
        type="edu.cmu.sphinx.decoder.scorer.ThreadedAcousticScorer">
    <property name="frontend" value="${frontend}"/>
</component>

<!-- ********************************************************** -->
<!-- The linguist configuration                         -->
<!-- ********************************************************** -->

<component name="lexTreeLinguist"
        type="edu.cmu.sphinx.linguist.lextree.LexTreeLinguist">
    <property name="logMath" value="logMath"/>
```

```
    <property name="acousticModel" value="wsj"/>
    <property name="languageModel" value="trigramModel"/>
    <property name="dictionary" value="dictionary"/>
    <property name="addFillerWords" value="false"/>
    <property name="fillerInsertionProbability" value="1E-10"/>
    <property name="generateUnitStates" value="false"/>
    <property name="wantUnigramSmear" value="true"/>
    <property name="unigramSmearWeight" value="1"/>
    <property name="wordInsertionProbability"
        value="${wordInsertionProbability}"/>
    <property name="silenceInsertionProbability"
        value="${silenceInsertionProbability}"/>
    <property name="languageWeight" value="${languageWeight}"/>
    <property name="unitManager" value="unitManager"/>
</component>


<!-- ********************************************************** -->
<!-- The Dictionary configuration                               -->
<!-- ********************************************************** -->
<component name="dictionary"
    type="edu.cmu.sphinx.linguist.dictionary.FastDictionary">
    <property name="dictionaryPath"
        value="resource:/edu/cmu/sphinx/models/acoustic/wsj/dict/cmudict.0.6d"/>
    <property name="fillerPath"
        value="resource:/edu/cmu/sphinx/models/acoustic/wsj/noisedict"/>
    <property name="addSilEndingPronunciation" value="false"/>
    <property name="wordReplacement" value="&lt;sil&gt;"/>
    <property name="unitManager" value="unitManager"/>
</component>


<!-- ********************************************************** -->
<!-- The Language Model configuration                           -->
<!-- ********************************************************** -->
<component name="trigramModel"
    type="edu.cmu.sphinx.linguist.language.ngram.SimpleNGramModel">
    <property name="location"
        value="resource:/jsphinx/desktop/JSphinxDesktop.trigram.lm"/>
    <property name="logMath" value="logMath"/>
    <property name="dictionary" value="dictionary"/>
    <property name="maxDepth" value="3"/>
    <property name="unigramWeight" value=".7"/>
</component>


<!-- ********************************************************** -->
<!-- The acoustic model configuration                           -->
<!-- ********************************************************** -->
<component name="wsj"
```

```
        type="edu.cmu.sphinx.linguist.acoustic.tiedstate.TiedStateAcousticModel">
    <property name="loader" value="wsjLoader"/>
    <property name="unitManager" value="unitManager"/>
  </component>

  <component name="wsjLoader"
type="edu.cmu.sphinx.linguist.acoustic.tiedstate.Sphinx3Loader">
    <property name="logMath" value="logMath"/>
    <property name="unitManager" value="unitManager"/>
    <property name="location" value="resource:/edu/cmu/sphinx/models/acoustic/wsj"/>
  </component>

  <!-- ********************************************************** -->
  <!-- The unit manager configuration                             -->
  <!-- ********************************************************** -->

  <component name="unitManager"
    type="edu.cmu.sphinx.linguist.acoustic.UnitManager"/>


  <!-- ********************************************************** -->
  <!-- The frontend configuration                                 -->
  <!-- ********************************************************** -->

  <component name="mfcFrontEnd" type="edu.cmu.sphinx.frontend.FrontEnd">
    <propertylist name="pipeline">
      <item>microphone </item>
      <item>preemphasizer </item>
      <item>windower </item>
      <item>fft </item>
      <item>melFilterBank </item>
      <item>dct </item>
      <item>liveCMN </item>
      <item>featureExtraction </item>
    </propertylist>
  </component>

  <!-- ********************************************************** -->
  <!-- The live frontend configuration                            -->
  <!-- ********************************************************** -->
  <component name="epFrontEnd" type="edu.cmu.sphinx.frontend.FrontEnd">
    <propertylist name="pipeline">
      <item>microphone </item>
      <item>dataBlocker </item>
      <item>speechClassifier </item>
      <item>speechMarker </item>
      <item>nonSpeechDataFilter </item>
      <item>preemphasizer </item>
      <item>windower </item>
      <item>fft </item>
```

```
        <item>melFilterBank </item>
        <item>dct </item>
        <item>liveCMN </item>
        <item>featureExtraction </item>
    </propertylist>
</component>

<component name="microphone"
          type="edu.cmu.sphinx.frontend.util.Microphone">
    <property name="closeBetweenUtterances" value="false"/>
</component>

<component name="dataBlocker" type="edu.cmu.sphinx.frontend.DataBlocker"/>

<component name="speechClassifier"
          type="edu.cmu.sphinx.frontend.endpoint.SpeechClassifier">
    <property name="threshold" value="13"/>
</component>

<component name="nonSpeechDataFilter"
          type="edu.cmu.sphinx.frontend.endpoint.NonSpeechDataFilter"/>

<component name="speechMarker"
          type="edu.cmu.sphinx.frontend.endpoint.SpeechMarker">
    <property name="speechTrailer" value="50"/>
</component>

<component name="preemphasizer"
                  type="edu.cmu.sphinx.frontend.filter.Preemphasizer"/>

<component name="windower"
         type="edu.cmu.sphinx.frontend.window.RaisedCosineWindower"/>

<component name="fft"
    type="edu.cmu.sphinx.frontend.transform.DiscreteFourierTransform"/>

<component name="melFilterBank"
    type="edu.cmu.sphinx.frontend.frequencywarp.MelFrequencyFilterBank"/>

<component name="dct"
        type="edu.cmu.sphinx.frontend.transform.DiscreteCosineTransform"/>

<component name="liveCMN"
          type="edu.cmu.sphinx.frontend.feature.LiveCMN"/>

<component name="featureExtraction"
                  type="edu.cmu.sphinx.frontend.feature.DeltasFeatureExtractor"/>

<!-- ********************************************************** -->
<!-- monitors                                                   -->
```

```
<!-- ********************************************************** -->

<component name="accuracyTracker"
           type="edu.cmu.sphinx.instrumentation.BestPathAccuracyTracker">
    <property name="recognizer" value="${recognizer}"/>
    <property name="showRawResults" value="false"/>
    <property name="showAlignedResults" value="false"/>
</component>

<component name="memoryTracker"
           type="edu.cmu.sphinx.instrumentation.MemoryTracker">
    <property name="recognizer" value="${recognizer}"/>
        <property name="showDetails" value="false"/>
        <property name="showSummary" value="false"/>
</component>

<component name="speedTracker"
           type="edu.cmu.sphinx.instrumentation.SpeedTracker">
    <property name="recognizer" value="${recognizer}"/>
    <property name="frontend" value="${frontend}"/>
        <property name="showDetails" value="false"/>
</component>

<component name="recognizerMonitor"
           type="edu.cmu.sphinx.instrumentation.RecognizerMonitor">
    <property name="recognizer" value="${recognizer}"/>
    <propertylist name="allocatedMonitors">
        <item>configMonitor </item>
    </propertylist>
</component>

<component name="configMonitor"
           type="edu.cmu.sphinx.instrumentation.ConfigMonitor">
    <property name="showConfig" value="false"/>
</component>


<!-- ********************************************************** -->
<!-- Miscellaneous components                                   -->
<!-- ********************************************************** -->

<component name="logMath" type="edu.cmu.sphinx.util.LogMath">
    <property name="logBase" value="1.0001"/>
    <property name="useAddTable" value="true"/>
</component>
</config>
```

4.1.5 SCREENSHOTS

Building Sphinx4 data locally drawing from sonatype OSS repository.

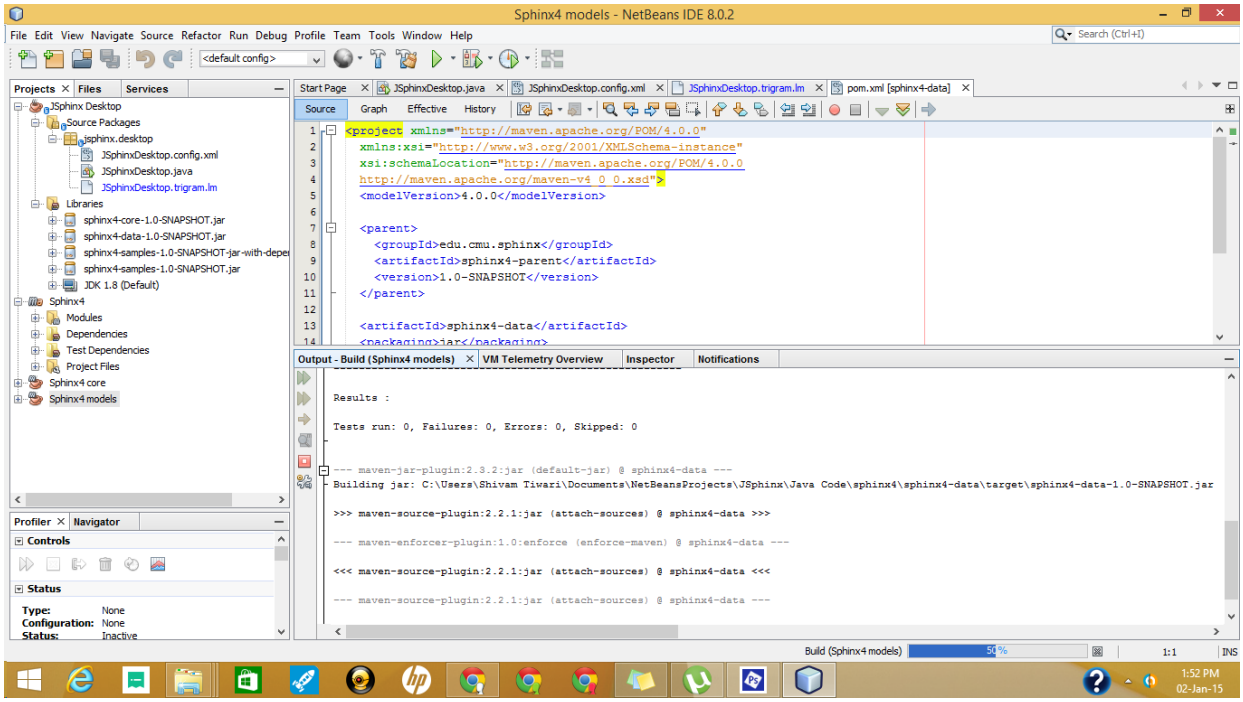


Figure 4.5

Successful build creates Sphinx4-data-1.0-SNAPSHOT.jar

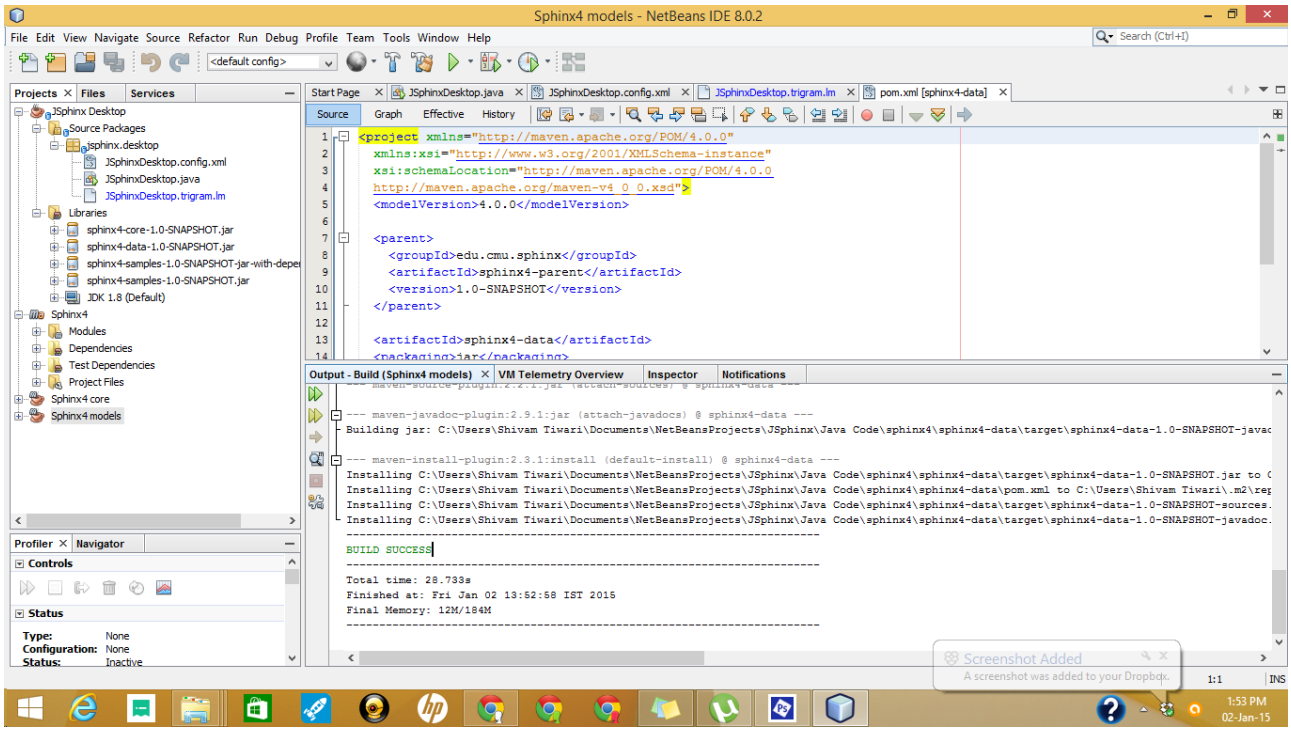


Figure 4.6

Building Sphinx4-core on Apache ANT drawing from Sonatype OSS

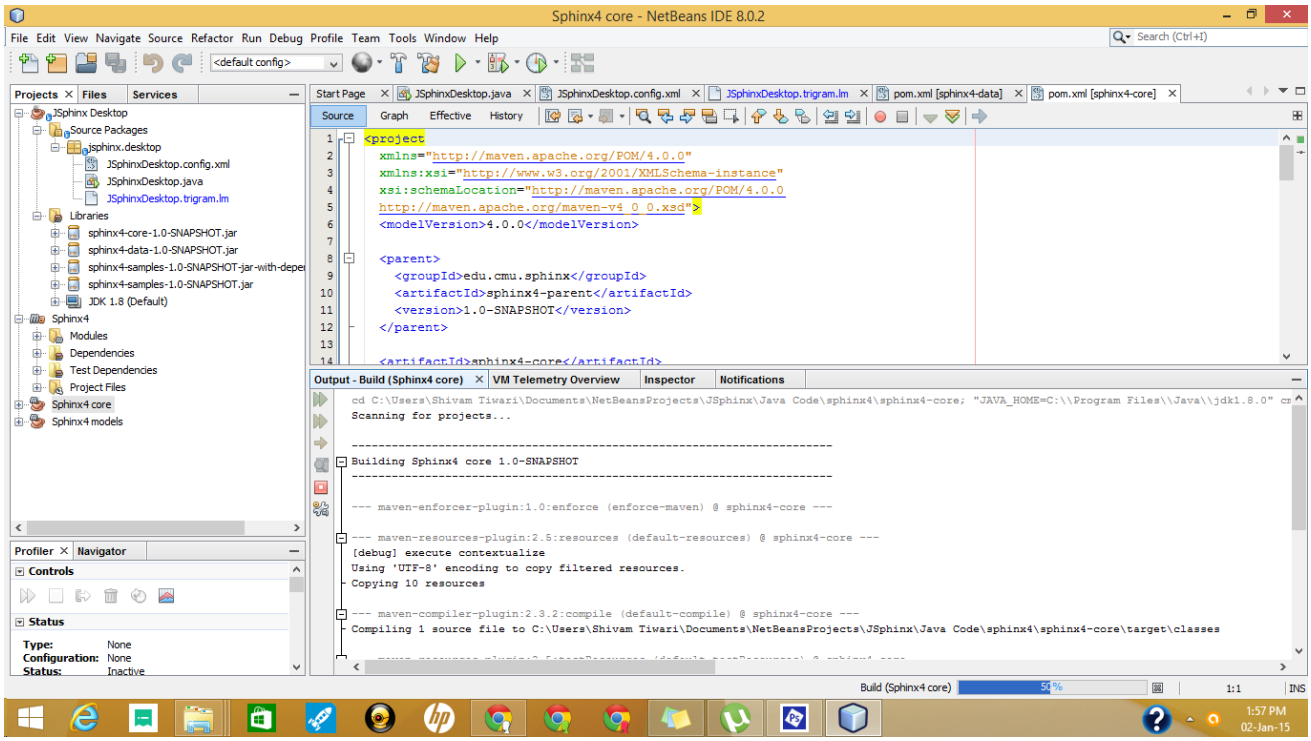


Figure 4.7

Running onboard tests for 94 modules

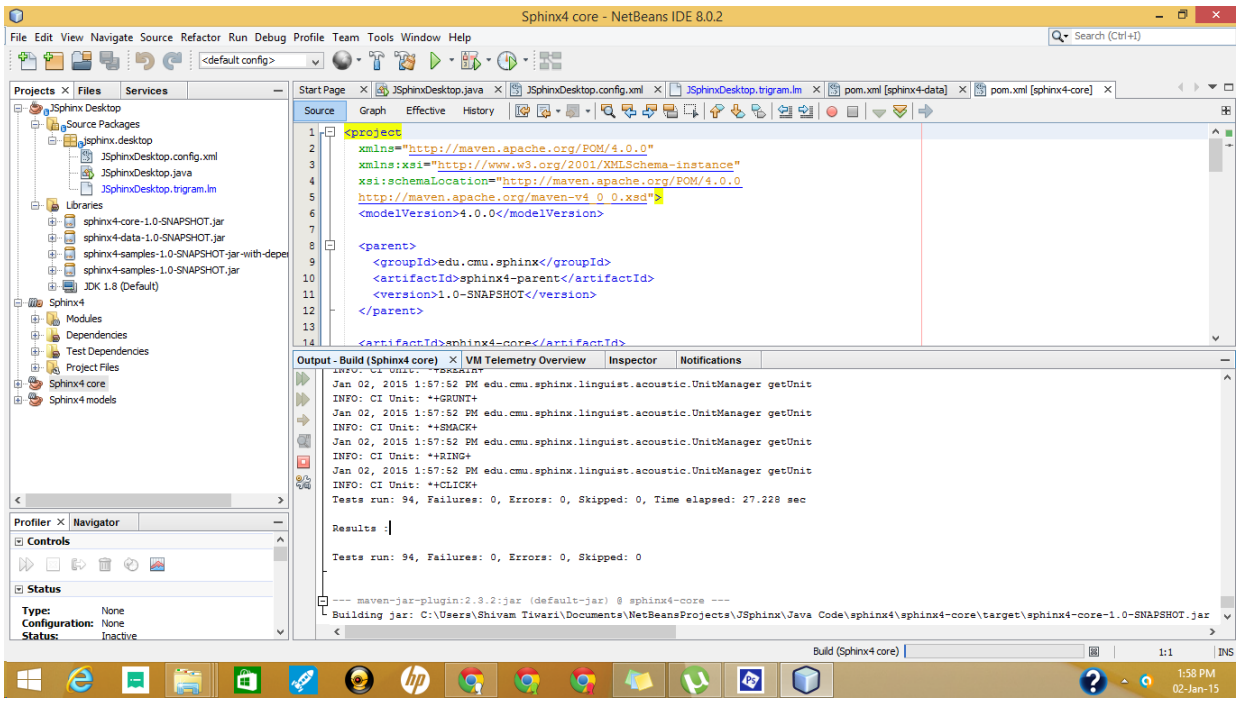


Figure 4.8

Successful build generates Sphinx4-core-1.0-SNAPSHOT.jar

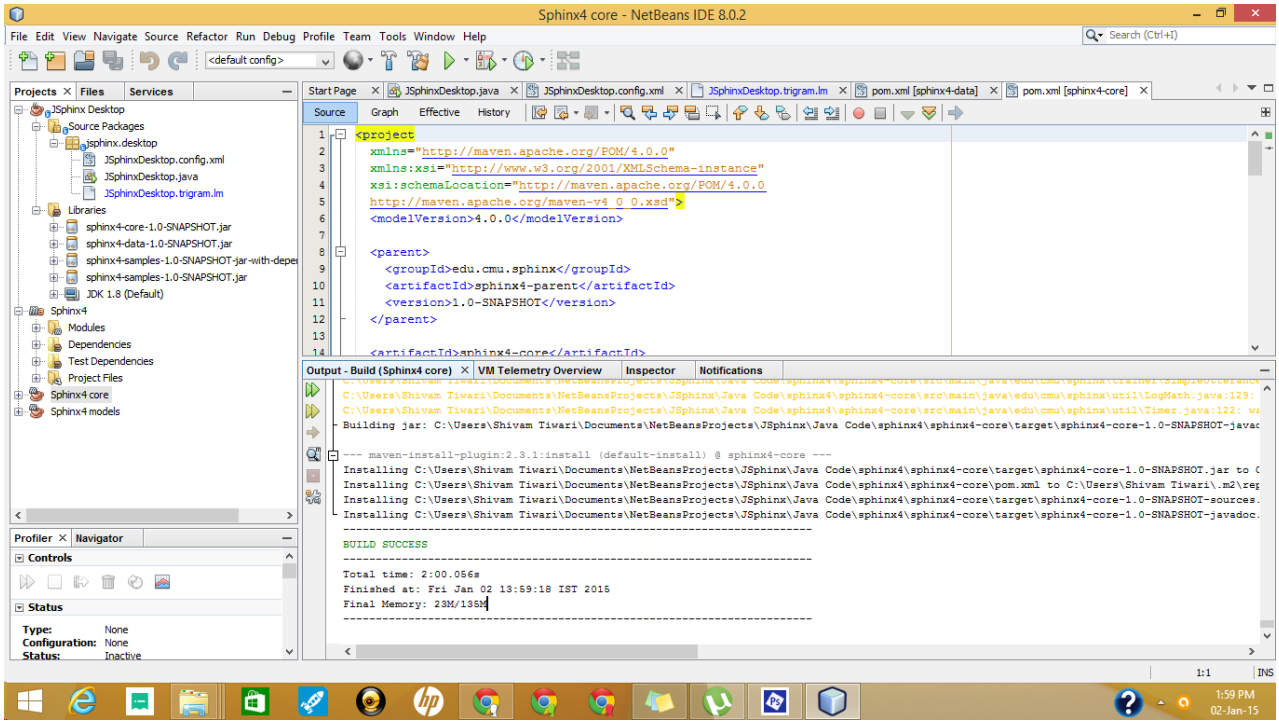


Figure 4.9

Building JSphinx Desktop project

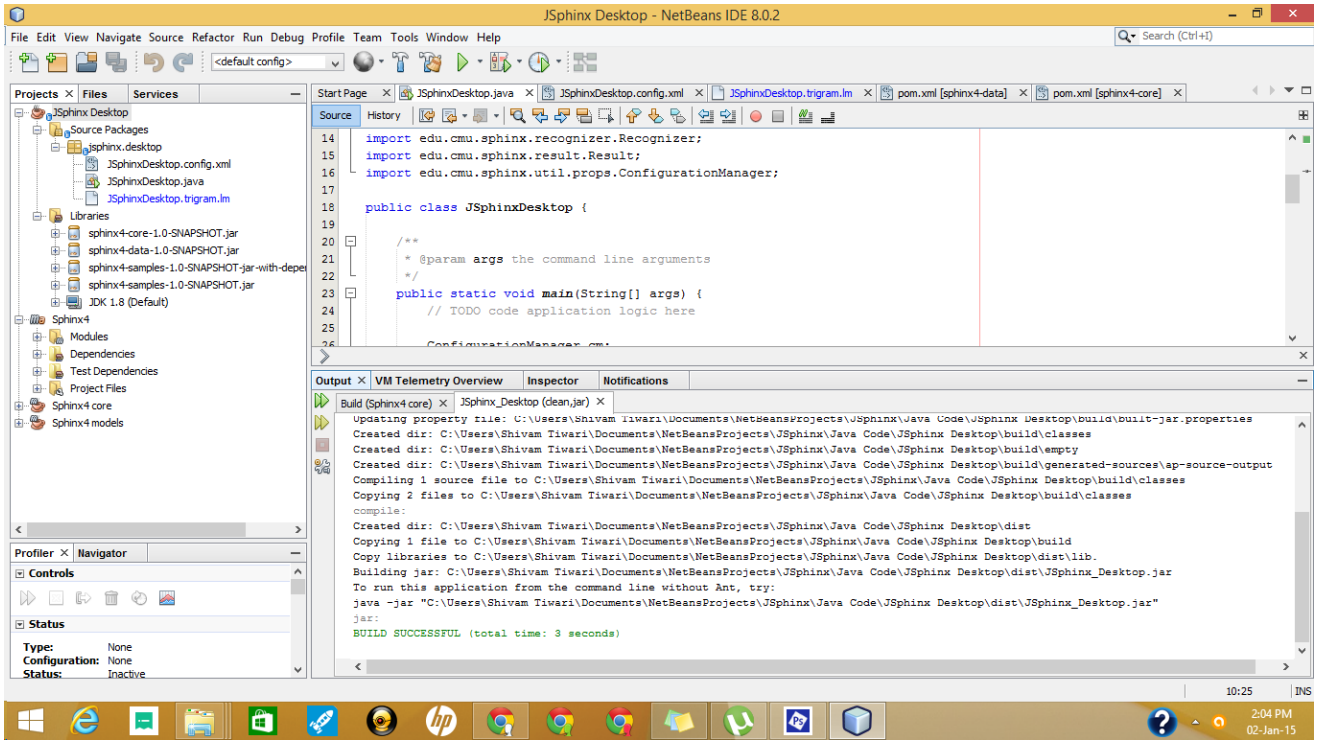


Figure 4.10

Running JSphinx Desktop for single words

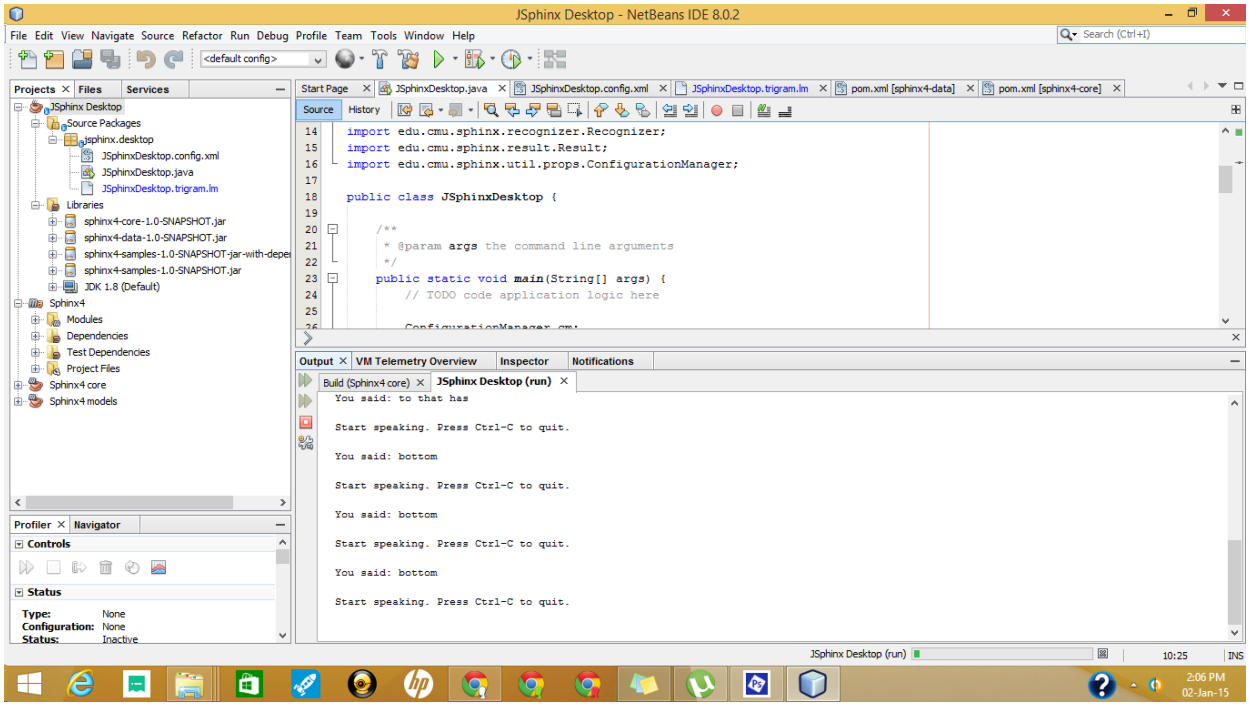


Figure 4.11

Running JSphinx desktop for sentences

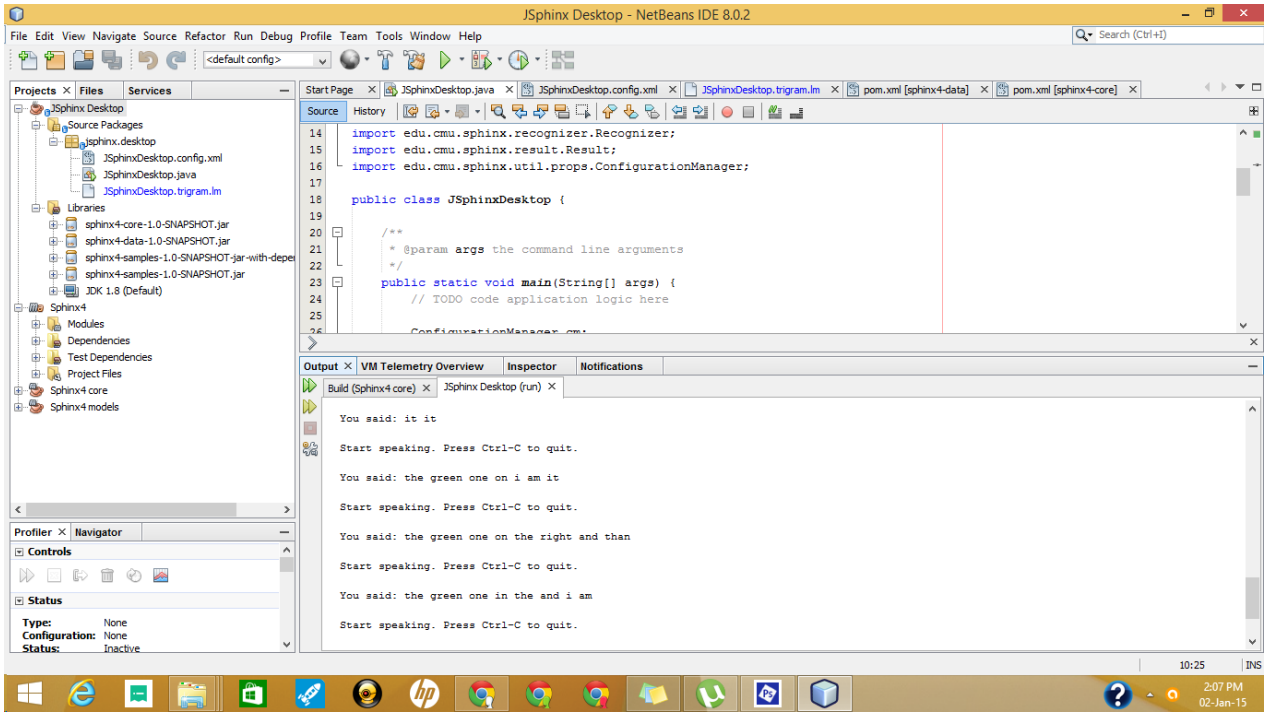


Figure 4.12

4.2 TESTING

4.2.1 TESTING PLAN

S.No.	Type of test	Will Test be performed?	Explanation	Software Component
1.	Requirements Testing	Yes	To test the successful implementation of all the requirements in the problem statement	Java NetBeans
2.	Unit Testing	Yes	To check if each part of the code is working individually	Java NetBeans
3.	Integration Testing	Yes	To check if the individual modules work together when integrated	Java NetBeans
4.	Performance Testing	Yes	To test how the system performs in terms of responsiveness and stability under a particular workload	Java NetBeans
5.	Stress Test	Yes	To test the stability under intense workload.	Java NetBeans
6.	Compliance Testing	No	Testing done to check if the software complies with the company standard. Not required in our project	N/A
7.	Security Testing	No	To reveal security flaws. Doesn't apply to our system	N/A
8.	Load Testing	No	To test the system under expected load. Already covered in stress testing	N/A
9.	Volume Testing	No	To test the software under a certain amount of data. Already covered in stress testing	N/A

Table 4.1

4.2.4 LIMITATIONS

We have tried solving almost all the errors that we have faced in the project, but to some extent some of the areas which still need refinement are:

1. Check for a more efficient solution

2. Generalize the problem

3. The code written for the problem is not optimized. There are many loops and comparisons which can be optimized.

4. Its efficiency can be increased by changing the way a table is made, messages are sent etc.

CHAPTER 5. FINDINGS AND CONCLUSION

5.1 FINDINGS

Speech is the primary means of communication between people. Whether due to technological curiosity to build machines that mimic humans or desire to automate work with machines, research in speech and speaker recognition, as a first step toward natural human machine communication, has attracted much enthusiasm over the past five decades, we have also encountered a number of practical limitations which hinder a widespread deployment of application and services.

Speech recognition is a challenging and interesting problem itself. We started our work on CMU SPHINX, an open source project and learnt about its functionalities, enough to further develop and modify it in accordance with our requirements.

5.2 CONCLUSION

In this project we have implemented a wrapper class on existing CMU Sphinx.

We have successfully been able to implement live voice recognition on continuous speech.

An interface (mobile device) for implementation of the API has been developed for the same.

Sample training sets for the system have also been created.

5.3 FUTURE WORK

We plan to implement the API in the mobile device interface that we have developed for the same.

Furthermore, we intend to train the system and develop the code for geographical mapping of speech. The training set for the same has been developed.

Accuracy of the system also will be developed in the future.

REFERENCES

1. http://archive.today/20120722204028/http://www.af.mil/news/story.asp?id=123071861
2. http://cmusphinx.sourceforge.net/wiki/
3. http://cmusphinx.sourceforge.net/wiki/tutorialam
4. http://mobyle.pasteur.fr/cgi-bin/portal.py#forms::hmmbuild
5. http://waset.org/publications/1192/continuous-feature-adaptation-for-non-native-speech-recognition
6. http://www.comp.leeds.ac.uk/roger/HiddenMarkovModels/html_dev/main.html
7. Michael Brandstein and Darren Ward, Eds., MicrophoneArrays - Signal Processing Techniques and Applications,Springer-Verlag, New York, 2001.
8. Singh, R., Seltzer, M., Raj, B., and Stern, R.M, Speech in noisy environments: robust automatic segmentation, feature extraction, and hypothesis combination , Proc. ICASSP 2001, Salt Lake City.

RESUMES OF GROUP MEMBERS

SIDDHARTH TANEJA

Email: siddharth11103586@gmail.com

Contact no: + 918586967575

Education

Year - Examination	***Board/University***	***%Marks***
B-Tech (CSE) 2011 – 2015	Jaypee Institute of Information Technology , Noida ,UP	CGPA of 5.5/10 (63%)
2011	CBSE 12th Grade (Sr. Secondary) Delhi Public School, R.K. Puram	81.6%
2009	CBSE 10th Grade (Secondary) Delhi Public School, R.K. Puram	85.33%

Skills

a. Expertise Area : Work Presentation, Public Relations, Team Management
b. Programming Languages : C , C++ , Python
c. Web Technologies : HTML , CSS , SQL
d. Adobe Photoshop, Adobe Illustrator
e. Audio Engineering

Projects

a. Project JSphinx - A VUI for portable devices (July ’14 – Present)
b. Architectural Security System – A unique security and remote monitoring system (February ’14 – May ’14)
c. Information Portal – Tourist information portal about Delhi, ‘incredibledilli.in’(August ’13 – December ’13)
d. Spy Bot – An autonomous bot, operating through Remote Desktop Connection(January ’13 – May ’13)

Work Experience

a. Summer Internship at Hitachi Systems Micro Clinic Pvt. Ltd. Duration: 6 weeks
b. Student partner, Vitarka Solutions. Duration: 5 months
c. Internship at India Infoline Pvt. Ltd. Duration :1 month

MAHIMA GOEL

Email: mahima.goel155@gmail.com

Contact no: + 919560412821

Education

Year - Examination	*Board/University*	*%Marks*
B-Tech (CSE) 2011 – 2015	Jaypee Institute of Information Technology , Noida ,UP	CGPA of 5.0/10
2011	CBSE 12th Grade (Sr. Secondary) Apeejay School, Sheikh Sarai	70%
2009	CBSE 10th Grade (Secondary) Apeejay School, Sheikh Sarai	75%

Skills

Expertise Area: Work Presentation, Public Relations, Team Management

Programming Languages: C, C++, Python

Web Technologies: HTML, SQL

Adobe Photoshop, Adobe Illustrator

Projects

Project JSphinx - A VUI for portable devices (July ’14 – Present)

Architectural Security System – A unique security and remote monitoring system (February ’14 – May ’14)

Information Portal –Information portal historical monuments in India (August’13 – December ’13)

Work Experience

Summer Internship at KPMG, Gurgaon. Duration: 6 weeks

SHIVAM TIWARI

Email: iam@shivamtiwari93.in

Contact no: + 919555428965

Education

Year - Examination	***Board/University***	***%Marks***
B-Tech (CSE) 2011 – 2015	Jaypee Institute of Information Technology , Noida ,UP	CGPA of 6.1/10
2011	CBSE 12th Grade (Sr. Secondary) KPS, Alaknanda	89%
2009	CBSE 10th Grade (Secondary) KV1-NSB	94.6%

Skills

Expertise Area : Design

Programming Languages : Java

Web Technologies : Java Enterprise

Projects

Project JSphinx

Performance comparison of Global and Preferential crawlers

Apriori analysis tool in java

Data Analysis Application for Project ASRU

Placement prediction system

Project ASRU

Creating a render farm on a shared network

Demonstration for heat seeking missile navigation

Channelizing Rain Water

Work Experience

Chief Knowledge Officer at ScrapLabs.