Jaypee Institute of Information Technology, Sector-62 Noida Department of CSE/IT

Major Project

JSphinx

Under the guidance of:
Mr. Kishore Kumar Y.
Assistant Professor
Dept. of CSE
JIIT, Noida

Submitted by:
Mahima Goel (11103499)
Shivam Tiwari (11104718)
Siddharth Taneja (11103586)

Contents

- Introduction
- Need for JSphinx
- Problem Statement
- Objective
- Methodology
- Implementation
- Applications
- Conclusions
- References

Introduction

- ➔ JSphinx is a development over CMU Sphinx which integrates phonetic analysis & speaker identity with already existing speech to text API.
- ➔ Provides a common speaker identity
- ➔ Improves speech to text performance across cultures and geographies

Problem Statement

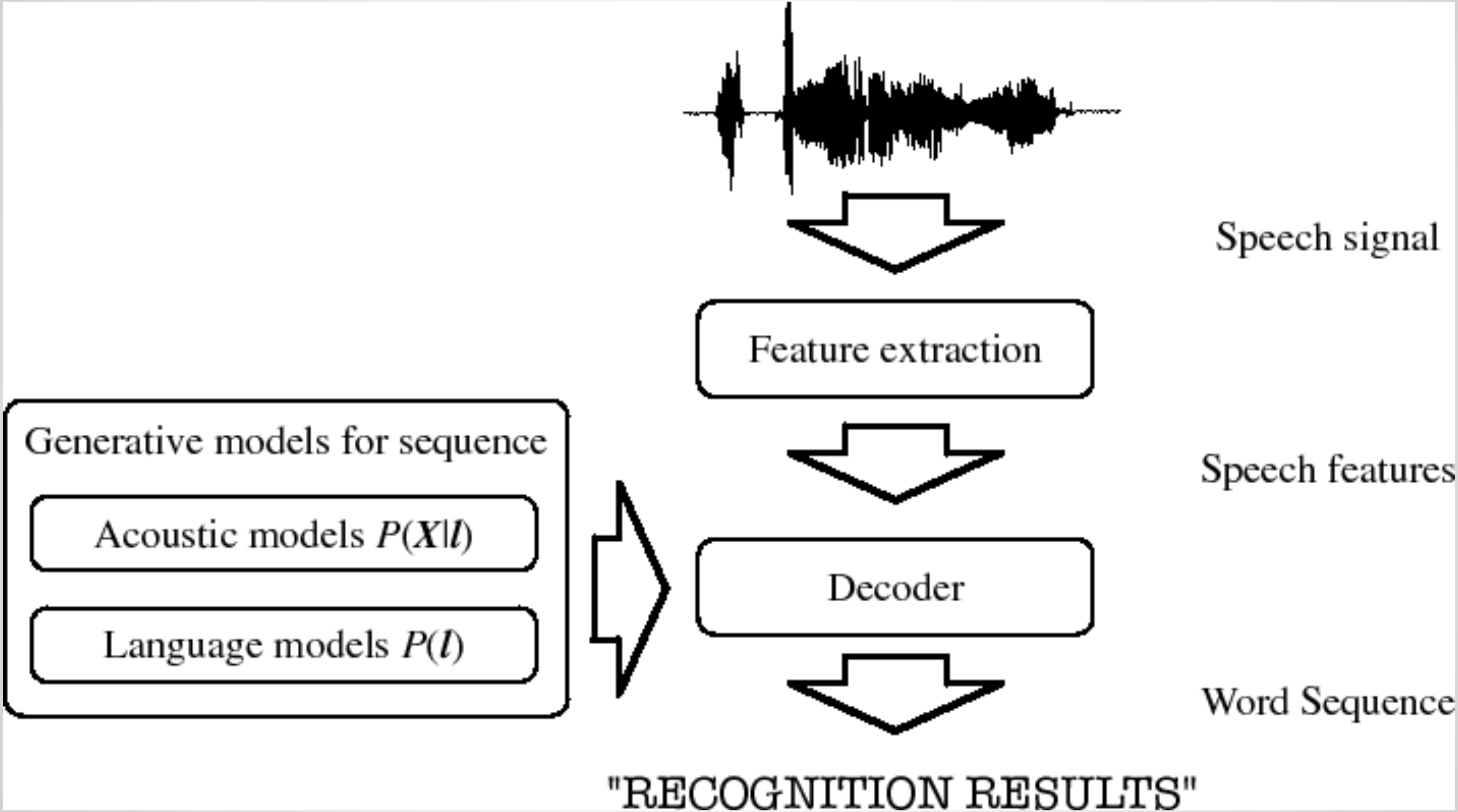
Many applications use speech recognition, and use learning to adapt to the accent of the user. However, a big problem is that if a user has N devices, he has to train all the devices individually to adapt to his accent.

Also, each user trains his own device, even though many users from the same geolocation have similar accents, therefore are applying similar training. There is no way to use this fact to increase the performance of the application in existence thus far.

JSphinx will try to overcome these problems.

Objective

To create a speech-to-text API that is a development over CMU Sphinx that addresses the problems of re-training over new devices and maps the users according to their geolocation.



Speech signal

Feature extraction

Speech features

Generative models for sequence

Acoustic models $P(\boldsymbol{X}|\boldsymbol{l})$

Language models $P(\boldsymbol{l})$

Decoder

Word Sequence

"RECOGNITION RESULTS"

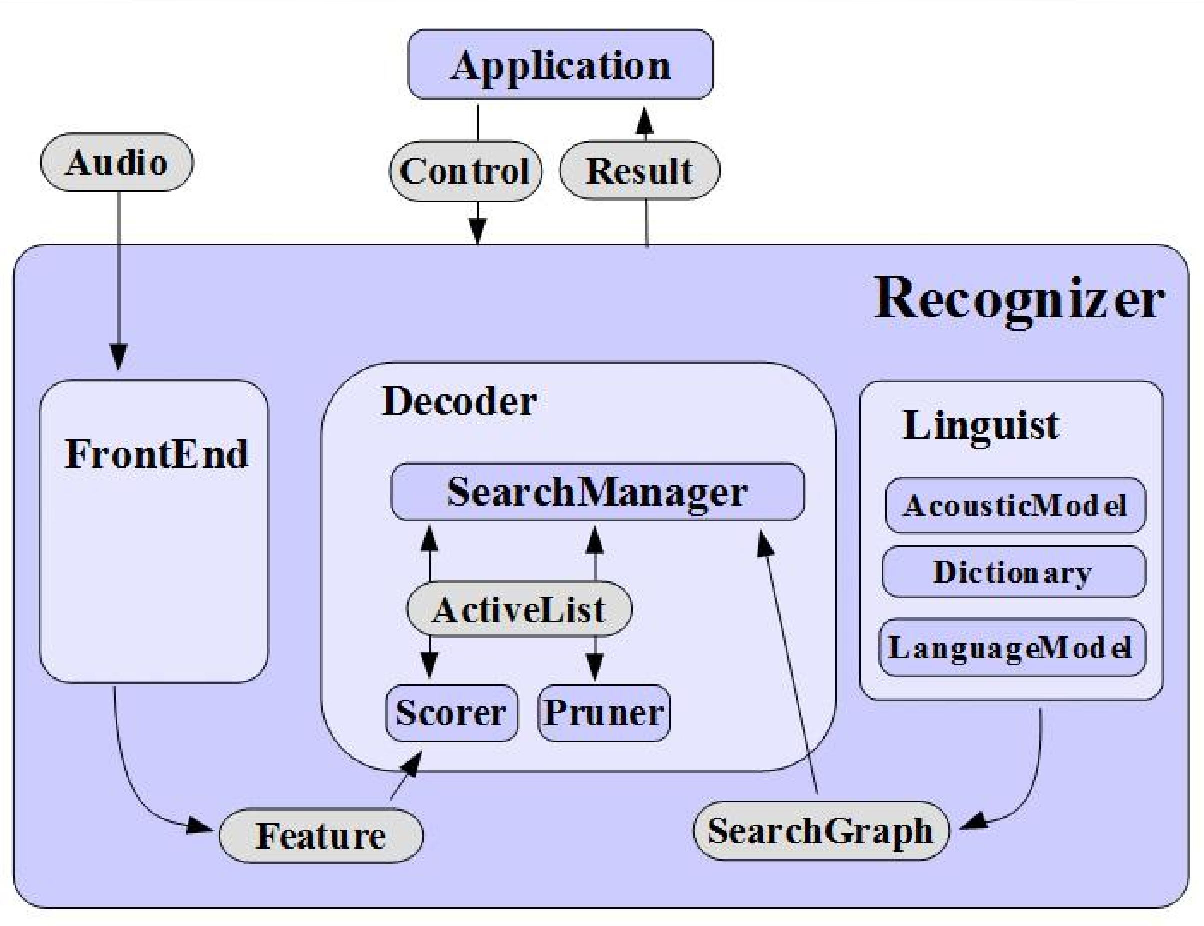
Need of JSphinx

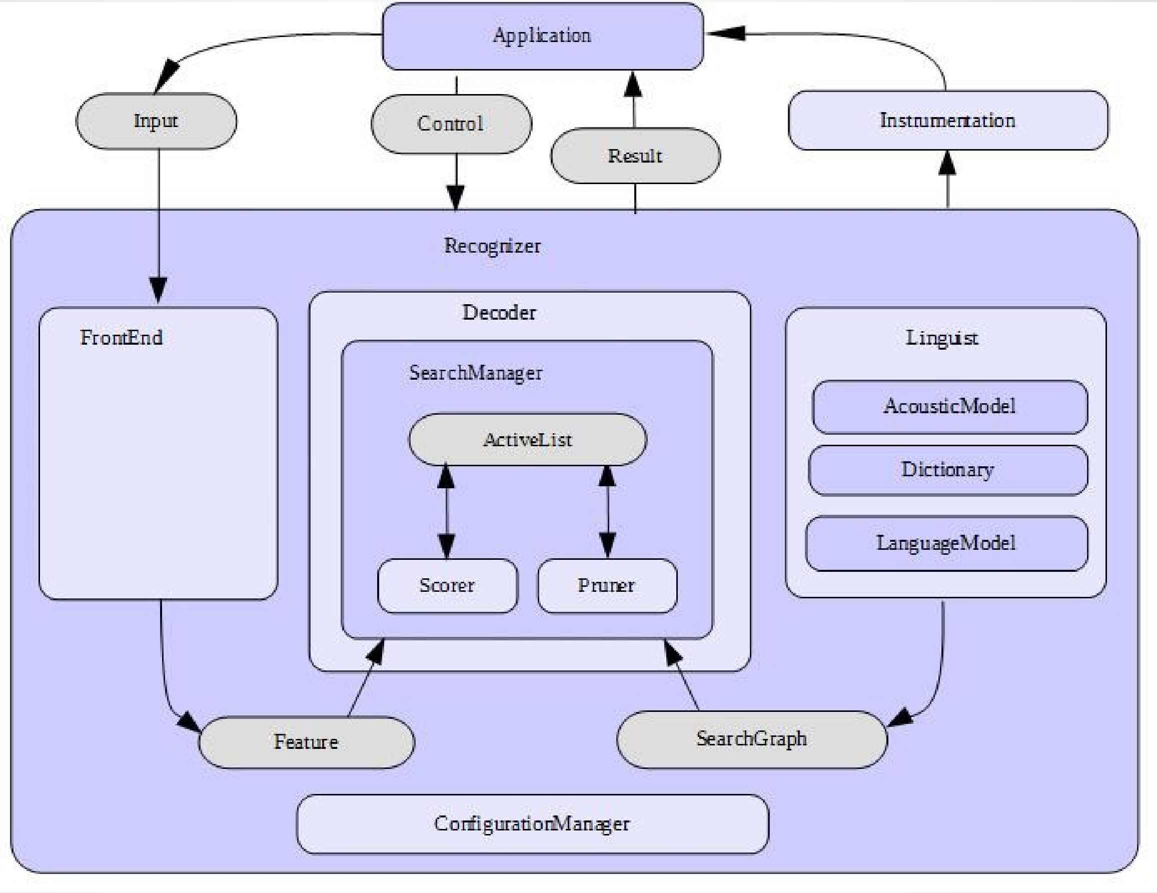
- Provide a common speaker identity: This is intended to share and sync the learning/training data on one device with all other devices of the same user.
- Improve speech to text performance across cultures and geographies

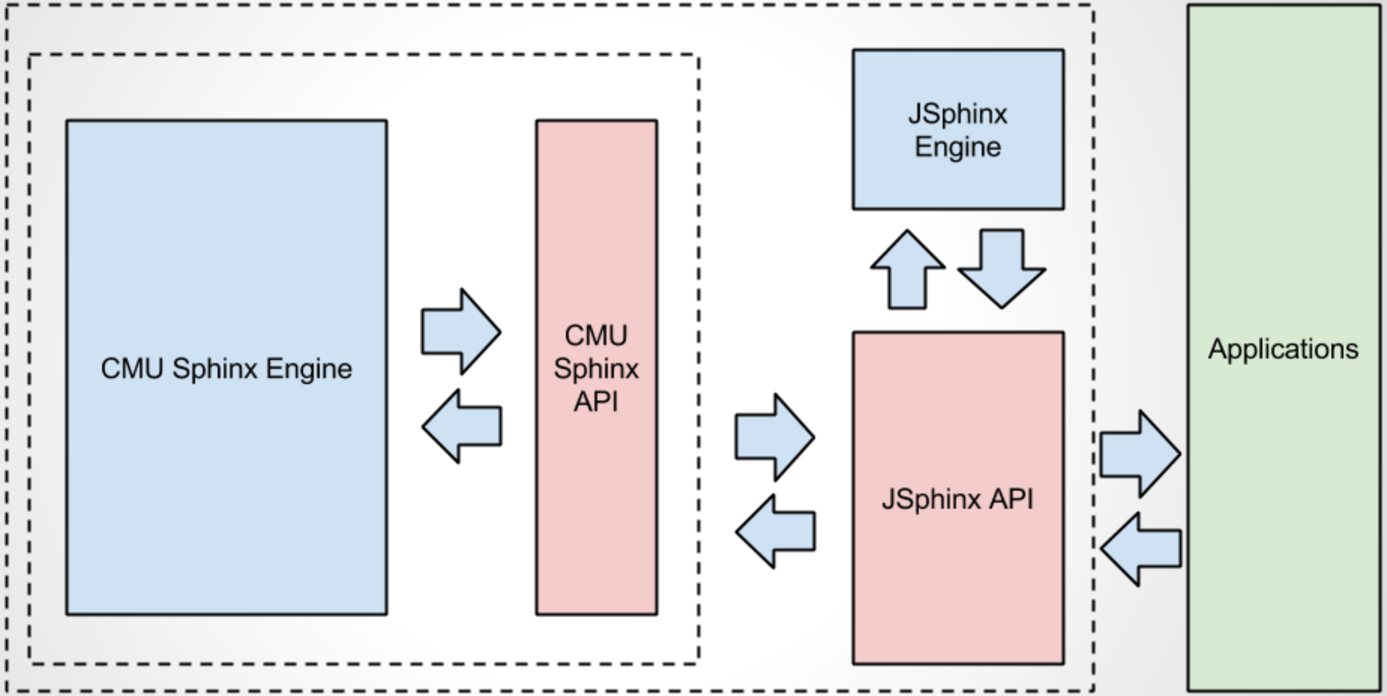
Implementation

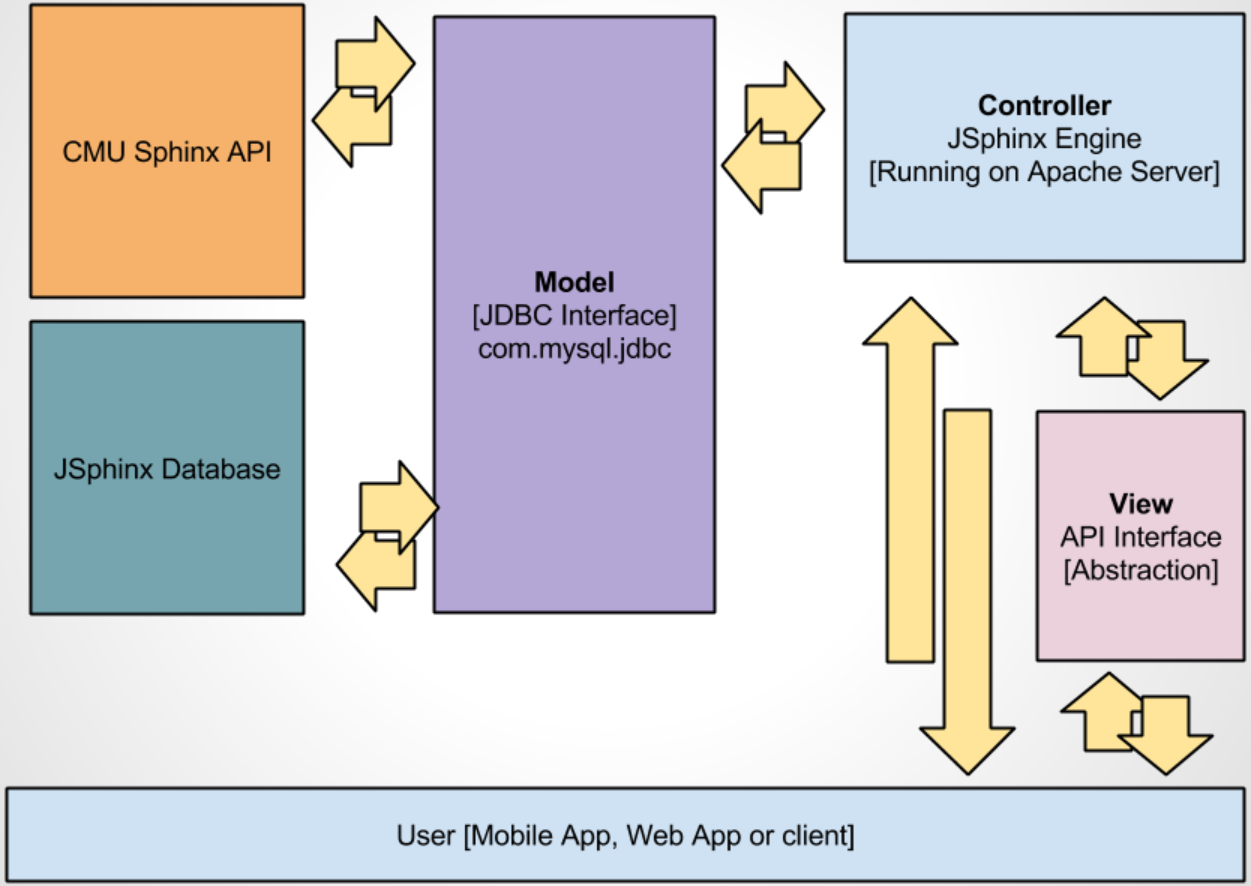
- Platform : Netbeans IDE
- Language : Java
- API : CMU Sphinx

The API has been tested for a set of commands and words.


Application
Audio
Control
Result
Recognizer
Decoder
FrontEnd
SearchManager
Linguist
AcousticModel
Dictionary
LanguageModel
ActiveList
Scorer
Pruner
Feature
SearchGraph


Application
Input
Control
Result
Instrumentation
Recognizer
FrontEnd
Decoder
SearchManager
ActiveList
Scorer
Pruner
Linguist
AcousticModel
Dictionary
LanguageModel
Feature
SearchGraph
ConfigurationManager


CMU Sphinx Engine
CMU Sphinx API
JSphinx Engine
JSphinx API
Applications



CMU Sphinx API

JSphinx Database

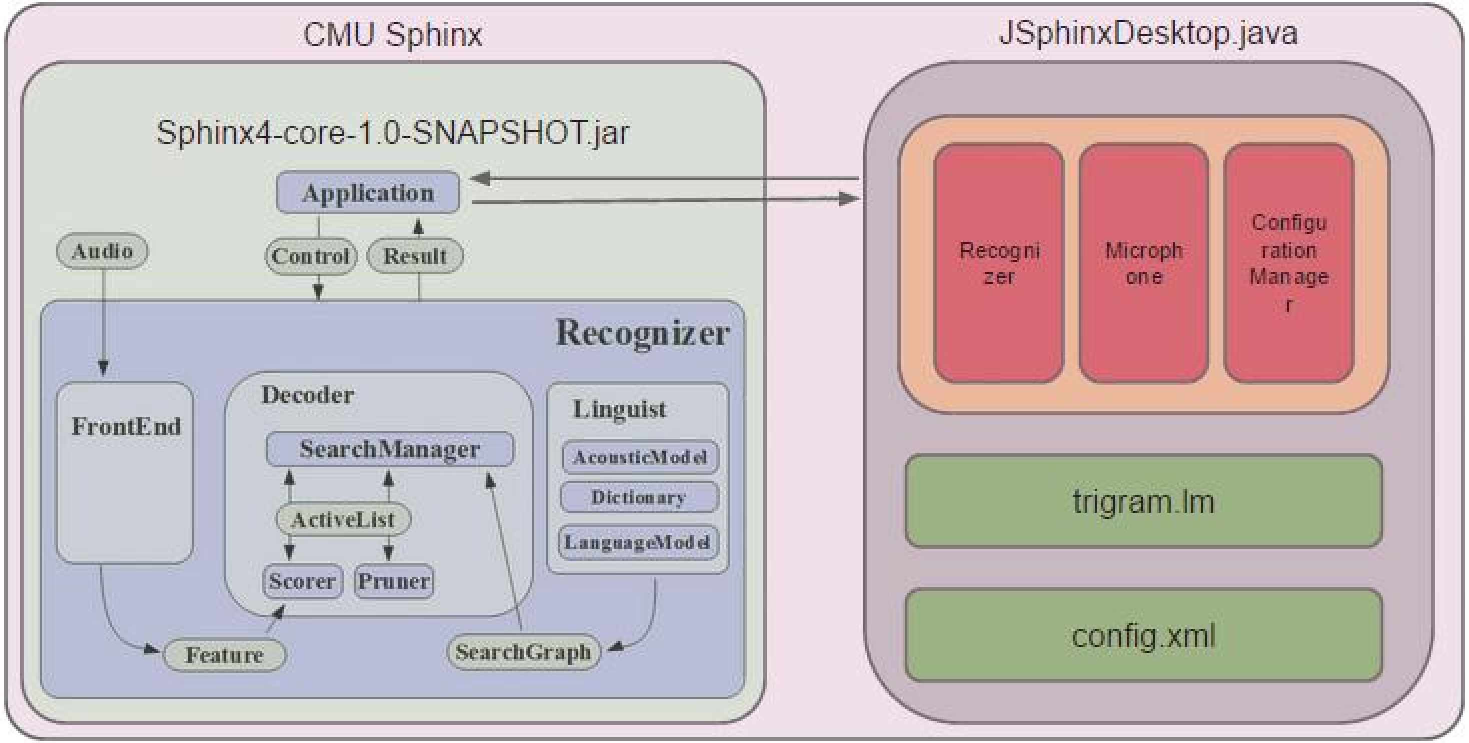
Model
[JDBC Interface]
com.mysql.jdbc

Controller
JSphinx Engine
[Running on Apache Server]

View
API Interface
[Abstraction]

User [Mobile App, Web App or client]

Jphinx.Desktop



Applications

API is intended to be ported across various mobile and non-mobile devices such as cellular phones, tablets, laptops, desktop computers, etc.

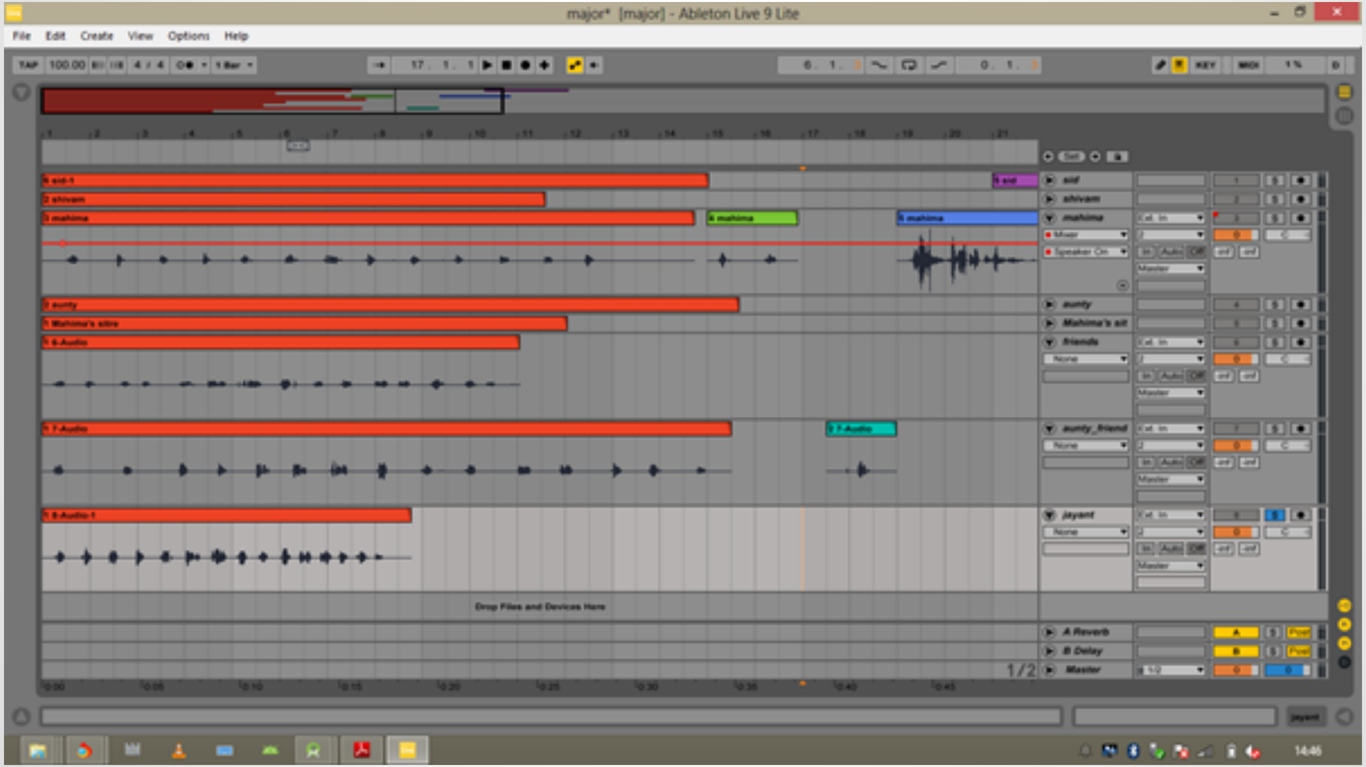
Phase 1 - Conclusion

- Successfully implemented CMU SPHINX
- Successfully created .lm files for the samples
- Successfully implemented Speech-to-Text recognition.
- Successfully designed the interface for the first deliverable of the JSphinx api - Mobile application

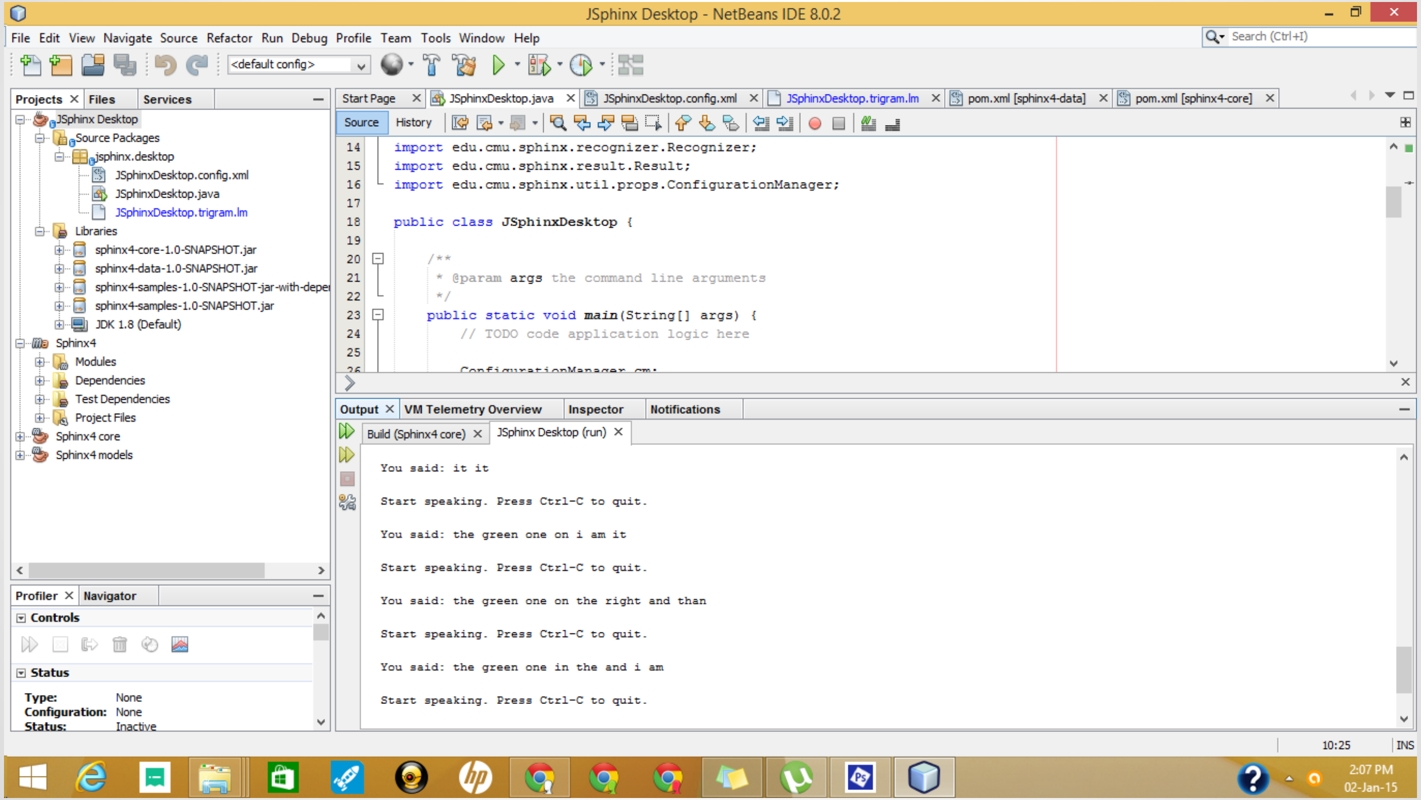
Phase 1 - Future Work

- shall customize the API according to demographic divisions across the country
- training modules shall be developed for the different regions of India - West, North , East , South
- client shall be able to save the training state,and can use it on multiple mobile devices
- make deliverables for the mobile devices - Android,iOS

Screenshot : Building Training Set



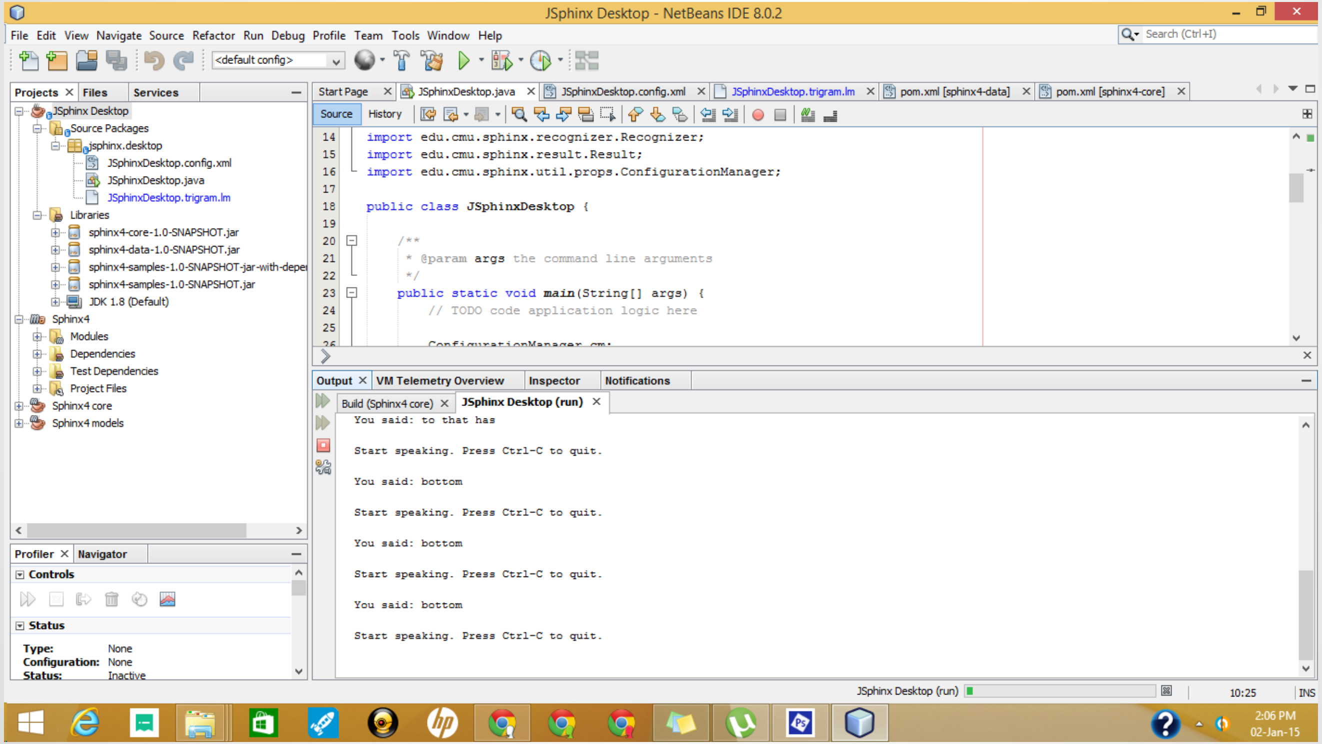
Screenshot: Running JSphinx desktop for sentences



```java
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;
```

```
You said: it it

Start speaking. Press Ctrl-C to quit.

You said: the green one on i am it

Start speaking. Press Ctrl-C to quit.

You said: the green one on the right and than

Start speaking. Press Ctrl-C to quit.

You said: the green one in the and i am

Start speaking. Press Ctrl-C to quit.
```

Screenshot : Running JSphinx Desktop for single words



```
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;
```

```
You said: to that has

Start speaking. Press Ctrl-C to quit.

You said: bottom

Start speaking. Press Ctrl-C to quit.

You said: bottom

Start speaking. Press Ctrl-C to quit.

You said: bottom

Start speaking. Press Ctrl-C to quit.
```

Type: None
Configuration: None
Status: Inactive

References

- http://cmusphinx.sourceforge.net/wiki/
- http://mobyle.pasteur.fr/cgi-bin/portal.py#forms::hmmbuild
- 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.
- http://www.comp.leeds.ac.uk/roger/HiddenMarkovModels/html_dev/main.html
- http://en.wikipedia.org/wiki/List_of_speech_recognition_software
- http://waset.org/publications/1192/continuous-feature-adaptation-for-non-native-speech-recognition