Automated Speech Recognition is currently available via mod_pocketsphinx and mod_unimrcp.

Contents

- 1 Overview
- 2 Nuance
  - 2.1 Download
  - 2.2 System
  - 2.3 Installation
  - 2.4 License Manager
  - 2.5 Start and Test Speech Server
  - 2.6 Logs
  - 2.7 mod_unimrcp configuration
  - 2.8 Sample Lua IVR
  - 2.9 Sample English Grammar
  - 2.10 Sample Hindi Grammar
- 3 LumenVox
- 4 Vestec
- 5 Simmortel
- 6 Loquendo
- 7 Sphinx

Overview

Speech Recognition Engine is used to convert speech to text. It is an alternative User Interface to DTMF in IVR applications.

Engines from the following vendors have been evaluated.

Nuance

Nuance is the monopolistic supplier of speech recognition engines. Accuracy and performance are good but cost is prohibitively high for most applications. The SDK is free trial for 2 months. Cost of per port perpetual license varies from $800 - $2000 depending on vocabulary size and natural language. Indian English and Hindi are good. Other Indian languages are available but need to be evaluated for accuracy.

Download

```
NRec-9.0.19-i386-rhel3.tar.gz
NRec-9.0.1-en-IN.i386-rhel3.tar.gz
NRec-9.0.1-hi-IN.i386-rhel3.tar.gz
NLICMGR-11.7.0-x86_64-linux.tar.gz
eval-rec-9.lic
```

System

```
CentOS 5.5 x86_64
```

Installation

```
tar xvzf NRec-9.0.19-i386-rhel3.tar.gz
./install.sh
tar xvzf NRec-9.0.1-en-IN.i386-rhel3.tar.gz
tar xvzf NRec-9.0.1-hi-IN.i386-rhel3.tar.gz
rpm -ivh NRec-en-IN-9.0-1.i386-rhel3.rpm
rpm -ivh NRec-hi-IN-9.0-1.i386-rhel3.rpm
```

License Manager

```
yum -y install redhat-lsb
tar xvzf NLICMGR-11.7.0-x86_64-linux.tar.gz
cd Nuance_License_Manager
./install.sh
cd /usr/local/Nuance/license_manager/license
cp /root/eval-rec-9.lic .
cd ../components
./set-new-lic-file.sh /usr/local/Nuance/license_manager/license/eval-rec-9.lic
```

Check that the license log file /usr/local/Nuance/license_manager/license/nuance-lic.log has the following contents, which means that the evaluation license file has been correctly configured by the License Manager.

```
19:55:22 (lmgrd) License file(s): /usr/local/Nuance/license_manager/license/eval-rec-9.lic
19:55:22 (lmgrd) lmgrd tcp-port 27000
19:55:22 (lmgrd) Starting vendor daemons ...
```

Check that the Nuance License Manager is running.

```
# ps aux | grep -i nuance
root      4887  0.0  0.0  15912  1244 pts/0    S    19:55   0:00 /usr/local/Nuance/license_manag
root      4888  0.0  0.0  32236  2084 ?        Ssl  19:55   0:00 swilmgrd -T localhost.localdoma
```

Start and Test Speech Server

```
service NSSservice start
```

Check that Nuance client is able to talk to Nuance Speech Server from a different machine.

Logs

Nuance Recognizer logs are in /usr/local/Nuance/Recognizer/data

mod_unimrcp configuration

1. cat conf/mrcp_profiles/nuance-5.0-mrcp-v2.xml

```
<include>
<profile name="nuance5-mrcp2" version="2">
  <param name="client-ip" value="$${local_ip_v4}"/>
  <param name="client-port" value="5090"/>
  <param name="server-ip" value="10.60.20.47"/>
  <param name="server-port" value="5060"/>
  <param name="sip-transport" value="udp"/>
  <param name="ua-name" value="FreeSWITCH"/>
  <param name="rtp-ip" value="$${local_ip_v4}"/>
  <param name="rtp-port-min" value="4000"/>
  <param name="rtp-port-max" value="5000"/>
  <param name="rtcp" value="1"/>
  <param name="rtcp-bye" value="2"/>
  <param name="rtcp-tx-interval" value="5000"/>
  <param name="rtcp-rx-resolution" value="1000"/>
  <param name="codecs" value="PCMU PCMA L16/96/8000"/>
  <synthparams>
  </synthparams>
  <recogparams>
  </recogparams>
</profile>
</include>
```

1. cat conf/autoload_configs/unimrcp.conf.xml

```
<configuration name="unimrcp.conf" description="UniMRCP Client">
<settings>
  <param name="default-tts-profile" value="voxeo-prophecy8.0-mrcp1"/>
  <param name="default-asr-profile" value="nuance5-mrcp2"/>
  <param name="log-level" value="DEBUG"/>
  <param name="enable-profile-events" value="false"/>
  <param name="max-connection-count" value="100"/>
  <param name="offer-new-connection" value="1"/>
</settings>
<profiles>
  <X-PRE-PROCESS cmd="include" data="../mrcp_profiles/*.xml"/>
</profiles>
</configuration>
```

Sample Lua IVR

In dialplan conf/dialplan/default.xml put the following extension

```
<extension name="unimrcp">
  <condition field="destination_number" expression="^(.*)4948611$">
    <action application="answer"/>
    <action application="lua" data="names.lua"/>
  </condition>
</extension>
```

cat scripts/names.lua

```
session:answer()
--freeswitch.consoleLog("INFO","Called extension is '" .. argv[1] .. "'\n")
welcome= "abhishek/welcome_to_knowlarity.wav"
menu = "abhishek/speak_name.wav"
nohear = "abhishek/sorry_no_hear.wav"
nounderstand = "abhishek/sorry_no_understand.wav"
forward = "abhishek/forwarding_to.wav"
thankyou = "ivr/8000/ivr-thank_you_for_calling.wav"
goodbye = "voicemail/8000/vm-goodbye.wav"
--
grammar = "names"
asrtag = "names"
no_input_timeout = 5000
recognition_timeout = 5000
confidence_threshold = 0.2
--
session:streamFile(welcome)
--freeswitch.consoleLog("INFO","Prompt file is '" .. prompt .. "'\n")
--
tryagain = 1
while (tryagain == 1) do
--
      session:execute("play_and_detect_speech",menu .. "detect:unimrcp {start-input-timers=false
      xml = session:getVariable('detect_speech_result')
      _,_,pre,result,suf = string.find(xml,"(.*)" .. asrtag .. ":(.*)}(.*)")
      _,_,pre,confidence,suf = string.find(xml,"(.*)confidence=\"(.*)\"(.*)")
--
      if (result == nil) then
              freeswitch.consoleLog("CRIT","Result is 'nil'\n")
              freeswitch.consoleLog("CRIT","Confidence is 'nil'\n")
              session:streamFile(nohear)
              tryagain = 1
      elseif (tonumber(confidence) < confidence_threshold) then
              freeswitch.consoleLog("CRIT","Result is '" .. result .. "'\n")
              freeswitch.consoleLog("CRIT","Confidence is LOW '" .. confidence .. "'\n")
              session:streamFile(nounderstand)
              tryagain = 1
      else
              freeswitch.consoleLog("CRIT","Result is '" .. result .. "'\n")
              freeswitch.consoleLog("CRIT","Confidence is HIGH '" .. confidence .. "'\n")
              prompt = "abhishek/" .. result .. ".wav"
              session:streamFile(prompt)
              tryagain = 0
      end
end
--
```

```
session:streamFile(forward)
-- put logic to forward call here
--
session:streamFile(thankyou)
session:sleep(250)
session:streamFile(goodbye)
session:hangup()
```

Sample English Grammar

1. cat grammar/sr.gram

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://www.w3.org/2001/06/grammar" version="1.0" xml:lang="en-IN" root="sr" tag-
<rule id="sr" scope="public">
  <one-of>
    <item>
      <ruleref uri="#sales"/>
      <tag>sr='sales'</tag>
    </item>
    <item>
      <ruleref uri="#support"/>
      <tag>sr='support'</tag>
    </item>
    <item>
      <ruleref uri="#voicemail"/>
      <tag>sr='voicemail'</tag>
    </item>
    <item>
      <ruleref uri="#fax"/>
      <tag>sr='fax'</tag>
    </item>
  </one-of>
</rule>
<rule id="sales">
  <one-of>
    <item>sales</item>
  </one-of>
</rule>
<rule id="support">
  <one-of>
    <item>support</item>
  </one-of>
</rule>
<rule id="voicemail">
  <one-of>
    <item>voicemail</item>
  </one-of>
</rule>
<rule id="fax">
  <one-of>
    <item>fax</item>
  </one-of>
</rule>
</grammar>
```

Sample Hindi Grammar

cat grammar/test.gram

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://www.w3.org/2001/06/grammar" version="1.0" xml:lang="hi-IN" root="names" t
<rule id="names" scope="public">
  <one-of>
    <item>
      <ruleref uri="#avni"/>
      <tag>names='avni'</tag>
    </item>
    <item>
      <ruleref uri="#bhagirath"/>
      <tag>names='bhagirath'</tag>
    </item>
    <item>
      <ruleref uri="#abhishek"/>
      <tag>names='abhishek singh'</tag>
    </item>
  </one-of>
</rule>
<rule id="avni">
  <one-of>
    <item>????</item>
  </one-of>
</rule>
<rule id="bhagirath">
  <one-of>
    <item>??????</item>
  </one-of>
</rule>
<rule id="abhishek">
  <one-of>
    <item>??????</item>
    <item>?????? ????</item>
  </one-of>
</rule>
</grammar>
```

LumenVox

Started off from Sphinx, the free and open source project at CMU but considerable proprietary development has been for MRCP and acoustic modelling. Indian English and Hindi are available but the cost of SDK is around $4000, so evaluation looks expensive.

Vestec

Vestec provides SDK for $25 or free evaluation. Per port license fee is around $200. Indian English and Hindi are available and to be evaluated.

Simmortel

Simmortel uses a mix of open source and proprietary product to deliver good accuracy for medium vocabulary applications in Indian English and Hindi. However, MRCP is not available and CPU usage for concurrent calls is very high.

Loquendo

Provides good European languages. Acquired by Nuance.

Sphinx

Sphinx is open source and free. It works well if you have trained acoustic model for your language and application. MRCP integration needs to be done for any real application.