Voximal Asterisk integration

The Asterisk Module

The Voximal installs the app_voximal Asterisk application module that uses the process voximald to execute the VoiceXML pages.

For more information on applications, just type “core show applications” at the Asterisk CLI prompt. To show details of how you use that particular application in this file (the Asterisk Dial plan).

Type:

```
*CLI> core show applications
```

For example, the voximal application:

```
*CLI> core show application voximal
```

The Asterisk Application

Application that launch a VoiceXML session in the asterisk channel and when complete, return control.

Syntax

```
Voximal([URL|Name|Number])
```

The URL can by set with different ways :

```
No parameter, the application will use the VoiceXML accounts (match called
number with accounts numbers) : Voximal()
Pass the URL as the appplication parameter. Example :
Voximal(file://tmp/test.vxml)
Pass the account name or the account number. Example: Voximal(test)
Pass the “@” to allocate a VoiceXML channel and pass the execution to this
dialplan extension. Example : Voximal(@600)
```

Set the VOXIMAL_URL variable before executing the Voximal application. Don't pass any parameter, and use the configuration accounts section. If account number match with the called number, its URL(s) and parameters will be use.

The following describes how to execute a VoiceXML session.

Configuration files

Two configuration files are used :

- VoiceXML interpreter configuration file (to overwrite default settings) : /etc/voximald.conf
- Asterisk Application/module configuration file : /etc/asterisk/voximal.conf

Configuration example

Example:

```
; VoiceXML Configuration
;
[general]
wav_codec=gsm
videosilence=;silence
audiosilence=;silence
debug=4
video=yes

[license]
max=100
key=…

[account1]
name=Test1
url=http://localhost/vxml/index.php
max=1

[account2]
name=Test2
url=http://localhost/vxml/demo/index.vxml
max=3
dialformat=SIP/%s@ovh-out
```

To assign an extension to a VXML account just follow this example, where we are assigning the previous account to three extensions number in your /etc/asterisk/extension.conf asterisk:

```
[default]

exten => 981001001,1,Voximal(Test1)
exten => 981001002,1,Voximal(Test2)
exten => 981001003,1,Voximal(Test3)
```

NOTE: When you update your voximal.conf file, remember to refresh configuration making a command “voximal reload” in your CLI*> prompt. If you have added SIP, PRI or new extensions you must launch “sip reload”, “extensions reload”, “dialplan reload” or reload asterisk/voximal processes. Use the command “voximal show accounts” to dump your accounts.

```
CLI*> diaplan reload
CLI*> voximal reload
CLI*> voximal show accounts
```

Example:

Add extensions to the Asterisk dial plan /etc/asterisk/extensions.conf, if you want to set the URL in the dialplan:

```
exten => 888,1,Answer
exten => 888,n,Wait(3)
exten => 888,n,Voximal(file:///root/example.vxml)
exten => 888,n,Hangup
```

You can create and edit the file /root/example.vxml with the GNU text editor, VI, for example.

This example will work if you have text-to-speech configured. If not, use a pre-recorded wav or gsm file to replace the “Hello world!” text by an <audio> tag. For more information, see the format extensions supported by Asterisk.

```
<?xml version="1.0"?>
<vxml version = "2.0" xmlns="http://www.w3.org/2001/vxml">
 <form>
  <block><audio src="hello.wav"/></block>
 </form>
</vxml>
```

Save the file in the same directory as the VoiceXML script (relative reference in this example). Reload the extensions configuration with:

```
*CLI> extensions reload
```

Call the service by calling:

```
SIP:888@<your server address>
```

Voximal variables

Variables to set before or filled after the Vxml() execution :

VOXIMAL_URL

If the variable VOXIMAL_URL has been set when vxml runs, the value of that variable will be used for the URL unless the parameter is not set to the application.

VOXIMAL_LOCAL

Force the called number (variable in VoiceXML context session.connection.local.uri).

VOXIMAL_REMOTE

Force the caller number (variable in VoiceXML context, session.connection.remote.uri).

VOXIMAL_MARK

Allow to add his value/mark in the VoiceXML browser logs associated to this call/VoiceXML session.

VOXIMAL_ID

If the variable VXML_ID has been set when vxml runs, the VoiceXML session ID variable called “telephone.id” is set with this value (in the VoiceXML execution session context).

VOXIMAL_PARAM

If the variable VOXIMAL_PARAM (VOXIMAL_AAI is an alias) has been set when vxml runs, the value of that variable will be used as “telephone.param” (in the VoiceXML execution session context) and session.connection.aai.

VOXIMAL_RESULT

After execution, the VoiceXML result of <exit> tag and the property ‘expr’ are accessible by the variable VOXIMAL_RESULT.

VOXIMAL_ERROR

After execution, the VoiceXML application notify the error cause, if the VoiceXML session cannot be launched.

- VOXIMAL_ERROR=(empty) ; No error occurs.
- VOXIMAL_ERROR=INITALISATION ; Session refused, Asterisk module not connected to the VoiceXML browser.
- VOXIMAL_ERROR=LICENSE ; Session refused, license locking
- VOXIMAL_ERROR=ACCOUNT_LIMIT ; Session refused, max session for the account reached.
- VOXIMAL_ERROR=SPEECH ; Session refused, cannot allocate the Speech (ASR) ressource.
- VOXIMAL_ERROR=TTY/TDD ; Session refused, no TTD availabe.
- VOXIMAL_ERROR=BILLING ; Session refused, billing interface refuse the session.
- VOXIMAL_ERROR=INTERPRETER ; Session refused, critical error with the VoiceXML browser connection.

Asterisk global variables

You can set and get variable at any moment from the VoiceXML syntax using the <transfer> tag :

GET

```
<transfer name="getvar" bridge="true" dest="execute:get(ID)" />
<block>
  <prompt>
    End of Transfer test <value expr="getvar$.value" />.
  </prompt>
</block>
```

SET

```
    <transfer name="toto" bridge="true" dest="execute:set(MOMO)=10">
    </transfer>
```

Asterisk Online Help

Online help can be accessed by typing the following command at the CLI prompt:

```
*CLI> help voximal
```

CLI Management Commands

Now your Voximal and Asterisk PBX are running, you can manage the Voximal using the Asterisk prompt *CLI>:

```
*CLI> voximal debug
```

Enable Voximal debugging for the Asterisk application.

```
*CLI> voximal debug interpreter
```

Enable Voximal debugging for the interpreter application.

```
*CLI> voximal no debug
```

This command disables Voximal debugging for the Asterisk application.

```
*CLI> voximal no interpreter debug
```

This command disables Voximal debugging for the interpreter application.

```
*CLI> voximal show cache / voximal cache show
```

Print the files in cache (if debug mode is enabled) and/or the number of file in the cache.

```
*CLI> voximal cache clear
```

Delete all the files in the cache.

```
*CLI> voximal cache purge
```

Delete all the files in the cache older than the maxage parameter set in the configuration.

```
*CLI> voximal show license
```

Use this command to show the license information.

```
*CLI> voximal reload
```

Reload the configuration.

```
*CLI> voximal show configuration
```

Use this command to show the configuration summary of VoiceXML interpreter.

```
*CLI> voximal show accounts
```

Show the accounts configured.

```
*CLI> voximal show account <number>
```

Show the accounts details of the account ID specified.

```
*CLI> voximal show statitiscs
```

Provides a dump statistics on VoiceXML interpreter

```
*CLI> voximal show dates
```

Provides a dump dates on VoiceXML interpreter

*CLI> voximal show sessions </code> Provides a dump sessions on VoiceXML interpreter.

```
*CLI> voximal show session
```

Provides a full dump session on VoiceXML interpreter.

```
*CLI> voximal originate chantype/number=application(parameters)
```

Originate an outgoing call, you can request to use a VoiceXML session.

The following entries are the Asterisk CLI commands for the VoiceXML browser.

Examples

Add extensions to the Asterisk dial plan /etc/asterisk/extensions.conf:

Example use with an URL parameter:

```
[incoming]
exten => 888,1,Answer
exten => 888,n,Wait(3)
exten => 888,n,Voximal(http://localhost/vxml/index.vxml)
exten => 888,n,Hangup
```

Example to catch the Vxml() errors :

```
exten => _X.,1,Voximal()
exten => _X.,n,NoOp(${VOXIMAL_ERROR})
exten => _X.,n,GotoIf($["${VOXIMAL_ERROR}" == ""]?hangup)
exten => _X.,n,Busy()
exten => _X.,n(hangup),Hangup()
```

You can create and edit the file /root/example.vxml with the GNU text editor, VI, for example.

```
# vi /root/example.vxml
```

NOTE:

This example will work if you have text-to-speech configured. If not, use a pre-recorded wav or gsm file to replace the “Hello world!” text by an <audio> tag. For more information, see the format extensions supported by Asterisk.

```
<?xml version="1.0"?>
<vxml version = "2.0" xmlns="http://www.w3.org/2001/vxml">
 <form>
  <block><audio src=”hello.wav”/></block>
 </form>
</vxml>
```

Save the file in the same directory as the VoiceXML script (relative reference in this example).

Reload the extensions configuration with:

```
CLI> extensions reload
```

Call the service by calling:

```
SIP:888@<your server address>
```

Load Balancer

Example using the Dialplan and 4 different accounts (with differents ASR and TTS ressources).

```
exten => s,1,NoOp(Balancing)
exten => s,n,Set(ACCOUNTMAX=4)
exten => s,n,Set(TRIES=0)
exten => s,n,Set(ACCOUNT=${RAND(1,${ACCOUNTMAX})})
exten => s,n(retry),Set(TRIES=$[${TRIES} + 1])
exten => s,n,Voximal(account${ACCOUNT})
exten => s,n,ExecIf($["${VOXIMAL_ERROR}" = ""]?Goto(app-blackhole,hangup,1))
exten => s,n,ExecIf($["${TRIES}" = "${ACCOUNTMAX}"]?Goto(app-
blackhole,hangup,1))
exten => s,n,Set(ACCOUNT=$[${ACCOUNT} + 1])
```

```
exten => s,n,ExecIf($[${ACCOUNT} = ${ACCOUNTMAX}]?Set(ACCOUNT=1))
exten => s,n,goto(s,retry)
```

Troubleshooting (for Support)

This chapter covers troubleshooting procedures for the VoiceXML Browser, describing some basic techniques that can be used when working with Vxi.

Collecting Information for Technical Support As part of the process of reporting problems, download log files and core files from the VoiceXML Browser and send them to I6NET, together with the current configuration files.

```
/var/log/voximal/log.txt (default configuration)
```

Log Files

The log files contain information about the operation of the VoiceXML Browser. The file is /var/log/voximal/log.txt, which details the VoiceXML processing on the VoiceXML Browser. If a failure occurs and you need to contact Voximal support (at support@voximal.com), they may ask you to activate traces to allow analysis of the system functions and make a complete appraisal of the problem. To do so you must follow these

procedures:

Edit the configuration file voximald.conf in /etc/. The levels are defined by these lines which are the# API/general log traces for each component:

```
client.log.diagTag.2000                         VXIInteger  0
client.log.diagTag.2001                         VXIInteger  0
client.log.diagTag.3000                         VXIInteger  0
client.log.diagTag.3001                         VXIInteger  0
client.log.diagTag.3002                         VXIInteger  0
```

The ranges are associated to different interfaces:

```
200x:  Cache interface
300x:  Internet HTTP interface
400x:  ECMAscript interface
500x:  Prompt interface
600x:  Recognize interface
700x:  Telephony/Session interface
800x:  XML Interpreter
900x:  Object interface
1000x: Main/Client application
```

To enable a level, set the 1 value and 0 to disable. To validate the modifications, the Voxiaml interpreter must be restarted. The interpreter is restarted if you restart the Asterisk.

The log file is generated in /var/log/voxiaml, and is named log.txt. The location and filename are

configurable. To purge the file type:

```
# > /var/log/voximal/log.txt
```

NOTE:

1. Never delete the log.txt directly, otherwise you should restart the Voximal to generate a new one.
2. An Apache/PHP script exists generate the traces from a standard Internet browser (Internet Explorer, Mozilla/firefox…). Get it from our web site
3. At the end of the trace record, don’t forget to stop it to recover optimal real time function. With the V4.x release, you can dynamically enable/disable the interpreter traces with an Asterisk *CLI> command.

Full traces:

```
*CLI> voximal debug interpreter
```

Disable the interpreter traces:

```
*CLI> voximal no debug interpreter
```

Current Sessions To determine how many VoiceXML sessions are currently active on the system, use the statistics dump on the CLI. This command displays the current number of sessions on the VoiceXML Browser:

```
*CLI> voximal show statistics
```

File descriptors To find out how many file descriptors are being used follow this:

Find out program PID:

```
# ps -ef | grep asterisk
```

Find out program PID (other command):

```
# pidof asterisk
```

List of files opened by PID (details):

```
# ls -l /proc/[PID]/fd
```

List of files opened by PID (counter):

```
# ls -l /proc/[PID]/fd | wc -l
```

More information about file descriptors:

```
# lsof | grep "[PID]"
```