Vestec Automatic Speech Recognition Engine
Standard Edition
Version 1.1.1

Administration Guide

Vestec Automatic Speech Recognition Engine
Standard Edition
Version 1.1.1
Administration Guide

Table of Contents

About This Document

Vestec's Automatic Speech Recognition Engine (VASRE) is a speaker-independent speech recognition engine that supports a distributed architecture of servers and clients. VASRE works for Windows, GNU/Linux, and Open Solaris platforms to process an audio file or stream from external sources, such as telephone systems. The grammar can be built simply, with a list of keywords or pronunciations to be recognized, or with a more sophisticated industry standard.

This guide explains how to administer VASRE by describing the use of executable and script files developed to compile text grammars and control the Resource Manager (RM) and servers.

Audience

This guide is intended for speech application developers who are administering VASRE.

Organization

This guide is organized as follows:

- Section 1 outlines the basics on VASRE administration.
- Section 2 explains the administration tools developed for the GNU/Linux platform.
- Section 3 explains the administration tools developed for the Windows platform.

Conventions

Guides for VASRE use the following conventions:

- **Bold Arial** represents user utterances, recognized strings, and semantic results.
- `Courier New` represents file names, directory names, command line strings, and file contents.
- *Italic text* represents types, tokens, keywords, variables, and functions.
- Underlined text represents menu strings or texts in graphical user interface.
- *`Italic Courier New`* represents values replaced by you. For example, *`YYYY-MM-DD`* may represent a date in the year-month-day format.
- A paragraph starting with **N.B.** represents critical information or warning.
- For abbreviated terms, both singular and plural are spelled the same. For example, RM represents both resource manager and resource managers.

1 Basics on VASRE Administration

After VASRE package is successfully installed (see Installation Guide for detail installation procedures), you can administer VASRE using the executable files contained in the package. They provide you with control of the main software components such as Resource Manager (RM), server controller, servers, grammar compiler, and pronunciation generator.

The RM is the control tower of the VASRE system, which verifies the purchased license, oversees servers, and coordinates communication sessions between servers and clients. Notice the difference between the server controller and servers. A server controller is a program unit that reads the server parameters from the server configuration file and start servers accordingly. A server is a program unit dedicated to speech recognition. Under the VASRE distributed architecture, multiple server processes are allowed to run on a single machine. The grammar compiler processes a text grammar file to create the corresponding binary grammar file. The pronunciation generator outputs the phoneme sequences of given words.

The RM, servers, and clients can be distributed over multiple machines. For example, the RM and one server can operate on Machine A, while two servers on Machine B and three clients on Machine C. If you would like to distribute the VASRE components over multiple machines, just install the VASRE package on all the machines and start the components selectively. However, you must have only one RM in your system and the license must be issued for the machine on which the RM will run. It is recommended that the machines composing a distributed VASRE system run identical operating systems. Proper operation is not guaranteed if different operating systems are installed on the machines that comprise the VASRE system.

2 Administering VASRE for GNU/Linux

This section describes the executable files and script files of VASRE developed for GNU/Linux platforms.

2.1 License Manager

The license manager is implemented inside the RM. When the RM starts, it first opens the license files under `/opt/VestecASRE/License/` and reads the encrypted information representing the machine's identification and the number and type of server ports purchased. The information in any multiple license files are merged; for example, if a license file A allows one port and B allows two ports, the total available ports will be three. Note however that only one Starter Kit license will work per machine. Even if multiple Starter Kit license files exist in the license directory, only one license will be accepted while the others are rejected. The results of license processing are logged in the RM log file.

2.2 Resource Manager

The RM is a key component of VASRE that coordinates the communication sessions between the servers and clients. To start an RM instance, run the script `vasreRM`. Run `vasreRM` with the `status` option to check the status of the RM:

```
/etc/init.d/vasreRM status
```

To start, restart, or stop the RM, run `vasreRM` with the `start`, `restart`, and `stop` options, respectively, as illustrated in the following shell terminal example:

```
~ $ /etc/init.d/vasreRM status
No resource manager is running
~ $ /etc/init.d/vasreRM start
Starting resource manager: Done
~ $ /etc/init.d/vasreRM status
1 resource manager is running
~ $ /etc/init.d/vasreRM restart
Stopping resource manager: Done
Starting resource manager: Done
~ $ /etc/init.d/vasreRM status
1 resource manager is running
~ $ /etc/init.d/vasreRM stop
Stopping resource manager: Done
~ $ /etc/init.d/vasreRM status
No resource manager is running
~ $
```

The RM is running in the background and will leave operation history in the log file `RM-`*`YYYY-MM-DD`*`.log` under `/var/log/VestecASRE/`, where *`YYYY-MM-DD`* represents the current date in the year-month-day format. For example, the log file of RM generated on January 1, 2010 will be named `RM-2010-01-01.log`. If the size of the log file reaches 5 MB, it will be renamed `RM-2010-01-01_0.log` and the upcoming log messages will be written to the empty log file `RM-2010-01-01.log`. If this file becomes larger than 5 MB again, it will be renamed `RM-2010-01-01_1.log`, and so on. You may delete old log files to save hard disk space.

The log file contains the history of all the events related to the RM operation. The format of each line is as follows:

```
HH-MM-SS: MESSAGE
```

where *HH-MM-SS* is a time stamp in the hour-minute-second format and *MESSAGE* is a string describing the event. For example, after the RM is started for the first time, the contents of the log file would be:

```
15-27-35:    Loading license file(s) ...
15-27-35:           VTE-XXXXXXXXXXXX: VESTEC_ENGINE
15-27-35:           # ports: 2
15-27-35:           VTE-YYYYYYYYYYYY: VESTEC_EN_US
15-27-35:    done (Single RM license: 2 ports, 500 voc)
15-27-35:    Resource manager is running at 9124 and waiting for server or client
connection ...
```

This sample log file is generated by the RM licensed by Digium license files. If you have Vestec license files, the message will be different. For the log message format of Vestec license files, see the example in Section 3.2.

In this sample log file, the first five lines indicate that two license files were loaded at 3:27 pm. These messages may differ according to the type of license files you have. The fifth line reports the license files were successfully loaded. The license allows two server ports with up to 500 vocabulary size for US English. The last line says the RM is waiting for server or client connection at the port of 9124.

Note that the RM port number is fixed as 9124. You may not change this port number from your side. You should run only one RM instance on each computer. You may start multiple instances, but only one will actually work.

If the RM fails to start, it will leave the associated messages in the log file. The possible errors are as follows:

- License files are missing or corrupt. If this is the case, get valid license files and copy them to the right directory `/opt/VestecASRE/License/`.
- Port 9124 is busy. If this is the case, terminate the process on port 9124.
- Failed to create, bind, or listen to listener socket. This message indicates some internal error happened related to TCP sockets. Restart your system and retry.

2.3 Server Controller

You can control the operation of recognition servers in two ways:

- Using script file `vasreSRVs` in `/etc/init.d/`
- Using executable files `StartSRVs` and `StopSRVs` in `/usr/bin/`.

If the RM is running remotely, modify the server configuration file `/etc/VestecASRE/vasre.conf` first. The configuration file must specify the IP address of the machine on which the RM is running. For example, if the RM is running at 1.2.3.4, you should add the following line to `/etc/VestecASRE/vasre.conf`:

```
-rm 1.2.3.4
```

Besides -rm, you may specify some options explained in the later part of this section. The changes you made in `/etc/VestecASRE/vasre.conf` will be applied when you restart the servers.

Enter the following command to check the status of the servers:

```
/etc/init.d/vasreSRVs status
```

To start, stop, or restart the servers, run `vasreSRVs` with the `start`, `stop`, and `restart` options, respectively. Starting the servers will take a while since it must load acoustic models and the dictionary.

The server controller and servers themselves are also running in the background and will create log files. The log file name of the server controller is `SRVs-`YYYY-MM-DD`.log`, where YYYY-MM-DD represents the current date. The log file name of the servers is `Port-`PORT`_`YYYY-MM-DD`.log`, where PORT represents the port number of the server. Like the log file of the RM, these log files are rolled if the size reaches 5MB. The following shows the example log file contents of the server controller:

```
16-37-06:
16-37-06: ===============================================================
16-37-06: Location of resource manager:          127.0.0.1
16-37-06: Number of recognition servers:         1
16-37-06: Base port number of servers:           10500
16-37-06: Sampling frequency of audio:           8000
16-37-06: Number of best recognition results:    1
16-37-06: Search accuracy:                       1000
16-37-06: Degree of rejecting short words:       10
16-37-06: Logging level:                         2
16-37-06: ===============================================================
16-37-06:
16-37-06: Initializing ...
16-37-07: --- Done.
16-37-07: Starting Server1 at 10500 ...
16-37-09: --- Done.
```

Each line of the log file starts with three hyphenated two-digit numbers representing the time of day in the *`HH-MM-SS`* format. The log file shows the setting of the started servers and the results of the start-up operation. The details on the server settings will be explained later in this subsection.

After the server has started, the contents of the RM log file will be updated as shown below. The servers connected to the RM and their status are added.

```
15-27-35:    Loading license file(s) ...
15-27-35:           VTE-XXXXXXXXXXXX: VESTEC_ENGINE
15-27-35:           # ports: 2
15-27-35:           VTE-YYYYYYYYYYYY: VESTEC_EN_US
15-27-35:    done (Single RM license: 2 ports, 500 voc)
15-27-35:    Resource manager is running at 9124 and waiting for server or client
connection ...
16-37-09:    1 rec. server is connected.
16-37-09:    [00] 127.0.0.1:10500     8kHz connected idle
```

New lines will be appended to the RM log file whenever changes in server status occur.

You may also use `StartSRVs` and `StopSRVs` in `/usr/bin/` to start or stop servers. Open a shell terminal and enter `StartSRVs` with no argument. One server will start with the default setting. Or, you may use the following options to start the servers with specific settings:

1. `-rm` *`ip_rm`*: Specify the IP address of the RM using the `-rm` option. *`ip_rm`* represents the IP address of the RM. If you don't use the `-rm` option, the default value 127.0.0.1 representing localhost will be used.
2. `-ns` *`num_srv`*: You may specify the number of servers to be started with the `-ns` option. If you don't use this option, one server will start.
3. `-port` *`port_num`*: You may specify the starting port number of servers with the `-port` option. The allowed range of *`port_num`* is 10000–15000. If you don't use this option, the default port number 10500 will be used.
4. `-nb` *`num_best`*: You may specify the maximum number of recognition results to be output (n-best) with the `-nb` option. The allowed range of *`num_best`* is 1–16. If you don't use this option, one n-best alternative will be output.
5. `-ac` *`acc_level`*: You may specify the level of recognition accuracy with the `-ac` option. The higher *`acc_level`* is, the higher the accuracy you may expect at the cost of slower processing. The allowed range of *`acc_level`* is 100–10000. If you don't use this option, the default value of 1000 will be used.
6. `-rj` *`sw_rej_deg`*: VASRE internally places a penalty on short words such as **a** or **i** during the recognition process since short words are often recognized falsely. To control the degree of penalty, use the `-rj` option with *`sw_rej_deg`*, which represents the degree of short word rejection. The allowed range of *`sw_rej_deg`* is 1–40 and the default value 10 should work well for general recognition purposes. If your grammar contains many short words and too many speech inputs are rejected by VASRE, try increasing this parameter.
7. `-log`, `-nolog`, `-log0`, `-log1`, or `-log2`: You may specify the level of logging using these options. `-nolog` or `-log0` will log nothing, while `-log` and `-log2` will log everything. `-log1` logs basic things. If you don't use this option, everything will be logged.
8. `-c` *`cfg_file`* or `-conf` *`cfg_file`*: Use these options to pass the configuration file to `StartSRVs`. The configuration file is a text file that enumerates one or more options as described above. You may enumerate the options in a single line or over multiple lines. If you use the `-c` or `-conf` option inside the configuration file, it will be ignored.

These options can be used for `/etc/VestecASRE/vasre.conf` as well.

For example, if you are going to start two servers at 10300 and 10301 to output 4 best results with the accuracy of 500, the command will be:

```
StartSRVs -ns 2 -nb 4 -ac 500 -port 10300
```

Note that when you start recognition servers using `/etc/init.d/vasreSRVs` script, options described in the configuration file `/etc/VestecASRE/vasre.conf` will be internally loaded. If you wish to start the servers with specific options, modify `/etc/VestecASRE/vasre.conf` before starting servers using `/etc/init.d/vasreSRVs` script.

To stop all the running servers, run `StopSRVs`. Use the `-f` option if you would like to skip the confirmation step.

If the server controller fails to start the servers, it will leave corresponding error messages in the log file. The server controller may fail if:

- An argument or the configuration file is wrong.
- The server controller fails to create a temporary directory or files.
- The recognition servers fail to start due to license limit, communication failure with RM, or other servers taking the port.

Consult the error message and fix the problem accordingly.

2.4 Grammar Compiler

The recognition server accepts speech recognition grammars in two formats: a text grammar and a binary grammar. A text grammar is a string written in the Speech Recognition Grammar Specification (SRGS) format, a World Wide Web Consortium (W3C) standard. It represents speech grammar rules in an Augmented Backus–Naur Form (ABNF) or an Extensible Markup Language (XML) format. See the Grammar Developer's Guide for more details on the SRGS format.

The VASRE assumes that a text file with an extension `.grm` will contain the SRGS grammar text in the ABNF format. Some sample grammar files are available in `/opt/VestecASRE/Samples/Grammars/`. The functionality of each grammar is described in detail in Section 15 of the Grammar Developer's Guide.

Using the grammar compiler, you can convert a text grammar file into a binary grammar file whose extension is `.gout`. Since the binary grammar file is directly readable by the server without a computationally expensive compilation process, it is a better strategy to make binary grammar files ready before starting the VASRE system. If you need to generate the grammar dynamically while the server is running, you should pass the text grammar to the server. The server will process the text grammar and return the error code and the associated message in case of syntax errors.

The grammar compiler `GramGen` is in `/usr/bin/`. Create a text grammar file and run `GramGen` with the `-g` option from a shell terminal to compile the grammar. For example, if you would like to compile `a.grm`, the command would be:

```
GramGen -g a.grm
```

If the text grammar file is free of syntax errors, `GramGen` will terminate with an exit code zero and generate `a.gout`; otherwise, a non-zero exit code will be issued and the associated error message will appear in the shell terminal. See the appendix of the Grammar Developer's Guide for the full list of error codes.

With the `-gout` option, you may specify the output file name of `GramGen`. If the output file name ends with something other than `.gout`, `.gout` will be automatically appended to the output file name.

Use the `-kwd` option with `GramGen` to consider every phrase or sentence as a single token. A token is the minimum recognizable unit described in the text grammar. The text grammar representing an alternative of long tokens will exhibit better recognition performance than its counterpart representing a combination of short tokens. Note however that if you use `-kwd`, the weights and tags in the text grammar will be ignored. See Grammar Developer's Guide for further details.

For example, consider a speech recognition grammar representing two-digit numbers: **zero zero**, **zero one**, ..., **nine nine**. As shown below, you may write the text grammar in two different ways: an alternative of 100 possible phrases composed of two words (left-hand side) or the repetition of a single word saying one-digit number (right-hand side). See the Grammar Developer's Guide for details.

```
#ABNF 1.0;
root $num;
$num = "zero zero"
     | "zero one"
     | "zero two"
     // ...
     | "nine nine"
     ;
```

```
#ABNF 1.0;
root $num;
$num = $digit $digit;
$digit = zero | one | two
       | three | four | five
       | six | seven | eight
       | nine
       ;
```

For this example, the former approach will exhibit better recognition performance than the latter. But, enumerating all the possible phrases to be spoken can be a very tedious and painful task in some cases. To avoid this, write the grammar in a compact way using ABNF rules supported by SRGS, and compile the text grammar with the `-kwd` option to have recognition performance as good as the alternative of long tokens.

For example, if you compile the right-hand side grammar in the above example with `-kwd`, the binary grammar output will have the same effect as the left-hand side grammar.

Once the grammar is compiled with `-kwd`, you will never have a word-based confidence score from the recognition results. For example, in the above two-digit recognition example, if you compile the right-hand side grammar with `-kwd`, the recognition result will have a single confidence score for the whole phrase instead of two scores for each digit.

There are two cases, where `-kwd` is ignored. One is the case where the text grammar represents too many phrases or sentences. The allowable number of phrases varies according to the words used. The other case is that the text grammar contains *$GARBAGE*, which is a special rule representing any speech segment. For these two cases, `GramGen` outputs a warning message saying that the `-kwd` option has been ignored. Note also that the `-kwd` option ignores tag statements in the text grammar and the compiled grammar won't generate semantic results. See Section 13 of the Grammar Developer's Guide for more details.

`GramGen` works with a pronunciation dictionary containing most common English words. Nevertheless, `GramGen` may fail to find the pronunciation of a certain word such as an international name. If this is the case, `GramGen` will stop compilation issuing an error message. For example, if you compile `Name.grm` containing an international name that is missing in the dictionary, `GramGen` will complain.

To resolve this situation, you may provide the pronunciations of such words manually as described in Section 7 of the Grammar Developer's Guide. Alternatively, you may allow `GramGen` to guess the pronunciation of such words using the `-ap` option, which represents auto pronunciation. In very rare cases, even `-ap` may fail to guess the pronunciation. If this is the case, provide the pronunciation manually or remove the word from the text grammar.

Note that, for the grammar compiler on the recognition server side, the `-kwd` option is off while `-ap` option on. You may not change this setting.

`GramGen` may output a set of sentences and the associated logical parsing represented by the given text grammar. This feature is helpful when you want to double-check whether the text grammar was written correctly. The set of sentences can be generated either exhaustively or randomly. To generate it exhaustively, add the `-genexh` option to the command. If the `.grm` file is syntactically correct, `GramGen` will create the `.gout` file first and then fill a text file with all possible sentences represented by the grammar. The name of the text file is automatically generated as *gout_file_name*_genexh.txt. For example, if you would like to enumerate the sentences represented by a.grm, use the following command:

```
GramGen -g a.grm -genexh
```

Then, you will see the following messages:

```
~ $ GramGen -g a.grm -genexh
Loading pronunciation dictionary ...
--- Done.
Preprocessing .grm file ...
--- Done.
Generating .gout ...
--- Done.
Generating sentences exhaustively ...
--- Done.
~ $
```

After creating a.gout, GramGen will add the sentences to a.gout_genexh.txt. Each line in the text file represents one sentence.

There is no internal limitation on the number of sentences defined for -genexh. That is, GramGen will keep adding sentences to the file, regardless of how many sentences there are. You will see the size of the text file grows while GramGen adds the sentences. Type Ctrl+C will stop GramGen, while retaining any sentences already generated.

To generate the sentences randomly, use the -genran option with the number of sentences to be generated. For example, if you would like to generate 50 random sentences from a.grm, use the following command:

```
GramGen -g a.grm -genran 50
```

This command makes GramGen create a text file whose name is *gout*_genran.txt where *gout* represents .gout file name, and fills the text file with 50 random sentences.

Use the -lp option to generate the logical parsing of the sentences. Logical parsing is a formal syntax for describing the sequence and relation of tags and rule references to the tokens that are input to the grammar processor. With the -lp option, each line of the text file has a sentence and the corresponding logical parsing separated by a tab character.

Note that GramGen never overwrites the output file of -genexh and -genran. For example, if *gout*_genexh.txt already exists, GramGen will output the sentences to *gout*_genexh1.txt. If *gout*_genexh1.txt exists, *gout*_genexh2.txt will be created and so on.

2.5 Pronunciation Generator

With the pronunciation generator, Pron, you may check whether a certain word exists in the VASRE pronunciation dictionary and how it is pronounced.

Open a shell terminal and enter Pron with one or more words separated by white spaces. If you would like to see the pronunciations (phoneme sequences) of **boy** and **girl**, for example, the command would be:

```
Pron boy girl
```

The pronunciations of the given words will be output on the shell terminal window. If one word has multiple pronunciations, all of them will be output over multiple lines.

Arguments of Pron are case-insensitive. If a given word is not found in the pronunciation dictionary, Pron will guess the pronunciation and output it followed by `(recommended)` as shown in the following screen shot:

```
~ $ Pron Daejeon
Loading dictionary ...
--- Done.
daejeon: "d eh jh iy ah n" (recommended)


~ $
```

You may use hyphens, underscores, and periods as a part of the arguments, but they will be regarded as white spaces. For example, if you use **high-speed** as an argument, you will have the pronunciations of **high** and **speed** separately.

If you would like to check the pronunciations of a word containing apostrophes, surround it with double quotation marks.

3 Administering VASRE for Windows

This section describes the executable files and script files for VASRE developed for Windows platforms.

N.B. The grammar compiler and pronunciation generator are executable from the `\VestecASRE\Bin\` directory only. If you attempt to run them from different directories, they will fail, and an error message will be issued.

3.1 License Manager

The license manager is implemented inside the RM. When the RM starts, the license manager checks whether the files under `\VestecASRE\License\` are valid license files. Each valid license file is processed by the license manager to load encrypted information such as license ID, machine's host ID, the number and type of server ports, and purchased languages. If multiple license files exist, the number of ports and purchased languages encrypted in the files are aggregated. For example, suppose that `\VestecASRE\License\` contains two license files A and B, which allow one server port and two server ports, respectively. Then, the RM will accept three servers.

Starter Kit is a special type of license that allows only one port per machine for cheaper price. Even if you buy two Starter Kit licenses for your machine, only one of them will be loaded by the license manager. However, a single Starter Kit license can be combined with standard licenses. The progress and results of license processing are logged in the RM log file explained in Section 3.2.

3.2 Resource Manager

The RM is a key component of VASRE that coordinates the communication sessions between the servers and clients. The RM runs as an NT service, which is a background process loaded by the Service Control Manager (SCM) of the NT kernel. While the VASRE package is installed, the RM is registered and started as a service. Whenever the machine starts, the RM will starts automatically and keeps running until the machine shuts down. This will continue until you stop the NT services or uninstall the VASRE package.

Three batch files are provided under `\VestecASRE\` for the manual control of services:

- `StartRM.bat`: The batch file creating and starting the NT service for the RM
- `StartSRVs.bat`: The batch file creating and starting the NT service for the server controller (see Section 3.3 for the details.)
- `StopServices.bat`: The batch file stopping and deleting the NT services for the RM and server controller, if any

These three batch files use `sc.exe`, which is a service-control tool included in the Windows operation system by default. For better understanding of the batch files, consult their contents.

To start the RM service manually, run `StartRM.bat`, which creates and starts `\VestecASRE\Bin\RM.exe` as a service named VasreRM. To stop the RM service, run `StopServices.bat`, which stops and deletes the services for the RM.

N.B. Note that `StopServices.bat` stops not only the RM service but also the server controller, which is the other NT service starting recognition servers. For more information, refer to Section 3.3.

N.B. To run `StartRM.bat` or `StopServices.bat` under Windows Vista and 7, you need administrator's privilege. If you run the batch files from the **Vestec ASRE** menu, right-click on the menu item to select **Run as administrator**. If you run the batch files from the command-line window, start the command-line window with administrator's privilege: Right-click on the command-line window icon and select **Run as administrator**.

The executable file `\VestecASRE\Bin\RM.exe` can be run as a service only. If you run `RM.exe` directly, you will see the following message and the RM won't start:

```
C:\VestecASRE\Bin>RM.exe
This is an executable file for NT service.
See Administrator's Guide for further details.

C:\VestecASRE\Bin>
```

The RM leaves operation history in the log file `\VestecASRE\Log\RM-`*`YYYY-MM-DD`*`.log`, where *`YYYY-MM-DD`* represents the current date in the year-month-day format. For example, the log file of RM generated on January 1, 2010 will be named `RM-2010-01-01.log`. If the size of the log file reaches 5 MB, it will be renamed `RM-2010-01-01_0.log` and the upcoming log messages will be written to the empty log file `RM-2010-01-01.log`. If the size of this file reaches 5 MB again, it will be renamed `RM-2010-01-01_1.log`, and so on.

The log file contains the history of all the events related to the RM operation. The format of each line is as follows:

    HH-MM-SS: MESSAGE

where *`HH-MM-SS`* is a time stamp in the hour-minute-second format and *`MESSAGE`* is a string describing the event. Consider the following sample RM log file contents:

```
15:27:35>>   Loading Vestec license file(s) ...
15:27:35>>   Getting host information ...
15:27:35>>          Version: 1.1.1
15:27:35>>          Host ID: 9fc7cdb65082596e02104216fec45271
15:27:35>>          O/S (0), Arch (1), Engine Type (4)
15:27:35>>   Reading license file(s) ...
15:27:35>>          C:/VestecASRE/License/my_1port.lic ...
15:27:35>>          C:/VestecASRE/License/my_5port.lic ...
15:27:35>>   Verifying license file(s) ...
15:27:35>>          C:/VestecASRE/License/my_1port.lic ...
15:27:35>>                 (# ports: 1; voc size: 500; languages: EN_US)
15:27:35>>          C:/VestecASRE/License/my_5port.lic ...
15:27:35>>                 (# ports: 5; voc size: 500; languages: EN_US)
15:27:35>>   # total ports: 6; voc size: 500; languages: EN_US
15:27:35>>   Resource manager is running at 9124 and waiting for server or client
connection ...
15:27:45>>   1 rec. server is connected
15:27:45>>   [00]   192.168.1.80:10500  8kHz   connected    idle
```

The first five lines show the host information, such as the engine version of 1.1.1, 32-byte host ID, O/S index of 0, architecture index of 1, and engine type of 4. The next three lines report the reading process of two license files: `my_1port.lic` and `my_5port.lic`. The next five lines show the verification process. The number of ports, vocabulary size, and languages contained in each license file are reported. If errors are found in the license files, the corresponding error messages will be output.

The total number of ports licensed and the port number of the RM are shown over the next two lines. The RM port number is fixed as 9124 and is not modifiable. The last two lines show the number of servers and their details. In this example, one server started at the port 10500 of the machine whose IP address is 192.168.1.80. The server is ready to accept the audio of 8kHz sampling frequency.

If the RM fails to start, it will leave the associated error messages in the log file. To confirm whether the RM is running, run Windows Task Manager (by typing Ctrl+Alt+Del) and select **Processes** tab. For Windows Vista and 7, press the button named **Show process from all users**. If `RM.exe` is found in the list, that means RM is running. Otherwise, `RM.exe` has stopped for some reason such as:

- License files are missing or corrupt. If this is the case, get valid license files and copy them to the right directory `\VestecASRE\License\`.
- Port 9124 is busy. If this is the case, terminate the process taking port 9124.
- Failed to create, bind, or listen to listener socket. This message indicates that an internal error happened related to TCP sockets. Restart your system and retry.

For example, if license files are missing the log message will be as follows:

```
16:39:54>>   Loading Vestec license file(s) ...
16:39:54>>   Getting host information ...
16:39:55>>         Version: 1.1.1
16:39:55>>         Host ID: 9fc7cdb65082596e02104216fec45271
16:39:55>>         O/S (0), Arch (1), Engine Type (4)
16:39:55>>   Reading license file(s) ...
16:39:55>>   Verifying license file(s) ...
16:39:55>>   # total ports: 0; voc size: 0; languages:
16:39:55>>   Error: RM terminates due to no ports licensed.
```

The last two lines of the log file indicate that no license files are available and hence RM terminates.

3.3 Server Controller

The server controller is a program unit that reads server configuration and starts multiple servers based on the configuration. Like the RM, the server controller is registered and started as an NT service by the VASRE installation package.

To start the service manually, run `StartSRVs.bat`, which creates and starts `\VestecASRE\Bin\VasreSRVs.exe` as a service named VasreSRVs. Like `RM.exe`, `VasreSRVs.exe` is executable only as a service. To stop the service, run `StopServices.bat`. Note that `StopServices.bat` stops both the RM and server controller.

N.B. To run `StartSRVs.bat` or `StopServices.bat` under Windows Vista and 7, you need administrator's privilege. If you run the batch files from the **Vestec ASRE** menu, right-click on the menu item to select **Run as administrator**. If you run the batch files from the command-line window, start the command-line window with administrator's privilege: Right-click on the command-line window icon and select **Run as administrator**.

The server configuration is specified in `\VestecASRE\Bin\VasreSRVs.cfg`, which is a text file in which a line starting with a semicolon is regarded as a comment. Initially, no parameter is specified in the configuration file and the server controller starts the server with a default configuration: one server running at the local host port of 10500.

N.B. You must not rename or move the configuration file. If the server controller fails to find the configuration file, servers won't start.

N.B. Even if you purchased multiple-port license, you should update the server configuration file to specify the number of servers to be started on the local machine. If not, only one server will start. This is because VASRE provides the flexibility to distribute servers over remote machines.

The options you may specify in the configuration file are `-rm`, `-ns`, `-port`, `-nb`, `-ac`, `-rj`, and `-log`. See Section 2.3 for full details on these seven options. For example, if you are going to start two servers at 10300 and 10301 to output the four best results with the accuracy of 500, the configuration file will contain the following four lines:

```
-ns 2
-nb 4
-ac 500
-port 10300
```

To apply the changes made in the configuration file, restart the server controller either by restarting the machine or by running the three batch files in the order of `StopServices.bat`, `StartRM.bat`, and `StartSRVs.bat`.

N.B. You should work with the server configuration file when you would like to start servers on remote machines. For example, assume that the RM is running on Machine A and you would like to start servers on Machine B. Install the package on Machine B and update `\VestecASRE\Bin\VasreSRVs.cfg` to specify the IP address of Machine A using `-rm` option. You should also register the RM on Machine A and the servers on Machine B as exceptions of Windows Firewall by performing the following (Skip the following if you run the RM and servers on the same machine): Open **Control Panel** on Machine A and run **Windows Firewall**. Select **Exceptions** tab and click **Add Program...** button to open **Add a Program** dialogue box. Click **Browse...** button and select `\VestecASRE\Bin\RM.exe` and click **OK**. Restart the servers on Machine B by running `StopServices.bat` and `StartSRVs.bat`. Open **Windows Firewall** dialogue box of Machine B and register `C:\Windows\Temp\Vasre.exe` as an exception as you did for Machine A.

The operation history of the server controller will be reported in the log file `\VestecASRE\Log\SRVs-`*`YYYY-MM-DD`*`.log`, where *`YYYY-MM-DD`* represents the current date in the year-month-day format. The example contents of the log file is as follows:

```
16:13:25>>   ===============================================================
16:13:25>>   Location of resource manager:         127.0.0.1
16:13:25>>   Number of recognition servers:        2
16:13:25>>   Base port number of servers:          10300
16:13:25>>   Sampling frequency of audio:          8000
16:13:25>>   N-best:                               4
16:13:25>>   Search accuracy:                      500
16:13:25>>   Degree of rejecting short words:      10
16:13:25>>   Logging level:                        2
16:13:25>>   ===============================================================
16:13:25>>   Initializing ...
16:13:27>>   --- Done.
16:13:27>>   Starting Server1 at 10300 ...
16:13:33>>   --- Done.
16:13:33>>   Starting Server2 at 10301 ...
16:13:39>>   --- Done.
```

It will show the configuration used to start the servers, and any error messages which are issued.

The server is a program unit dedicated to speech recognition. One or more servers are launched by the server controller. The process name of the server controller and servers are `VasreSRVs.exe` and `Vasre.exe`, respectively. From Windows Task Manager, you will see one server controller process and as many server processes as you started.

Note the difference between the GNU/Linux version and Windows version of VASRE regarding server dependency on the RM. For the GNU/Linux platform, the server process will continue even if the RM process terminates. On the other hand, the Windows server process will stop if its port is closed from the client and the RM terminates for some reason.

Each server process also leaves log messages in `\VestecASRE\Log\Port-`*`PORT_YYYY-MM-DD`*`.log`, where *`PORT`* represents the port number of the server and *`YYYY-MM-DD`* represents the current date. The log file of the server shows all the history of client request processing.

3.4 Grammar Compiler

The grammar compiler `\VestecASRE\Bin\GramGen.exe` converts a text grammar file into a binary grammar file. The usage of `GramGen.exe` is identical to the GNU/Linux version of the grammar compiler except that `GramGen.exe` is executable only from `\VestecASRE\Bin\`. See Section 2.4 for the full description on the grammar compiler.

3.5 Pronunciation Generator

The pronunciation generator `\VestecASRE\Bin\Pron.exe` enables you to check whether a certain word exists in the VASRE pronunciation dictionary. If the word doesn't exist in the dictionary, `Pron.exe` recommends a pronunciation. Like `GramGen.exe`, the usage of `Pron.exe` is identical to the GNU/Linux version of pronunciation generator. Refer to Section 2.5 for the full usage of `Pron.exe`.