



# **L&H Automation Tools for ASR Context Management**

## **Reference Guide**

**Lernout & Hauspie Speech Products NV**  
**Flanders Language Valley 50**  
**B-8900 Ieper, Belgium**  
**Phone: +32 57 22 8888 Fax: +32 57 20 8489**

**Copyright**

(C) Lernout & Hauspie Speech Products N.V.  
Document No. D92751-21-00 L&H Automation Tools for ASR Context Management  
Reference Guide  
V1.00 © - November 1999

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording or any information retrieval system, without the written permission of L&H.

---

---

# Contents

<b>PURPOSE .....</b>	<b>1</b>
<b>FEATURES.....</b>	<b>3</b>
Introduction.....	4
asrbatchtool.ini .....	5
Sections .....	5
Sample .....	6
<b>COMMANDS AND THEIR USAGE .....</b>	<b>7</b>
Command Reference .....	7
Import .....	8
Export .....	10
Install .....	12
MakeInstall .....	13
MergeDict .....	14
ExtractDict .....	15
SaveDictList .....	16

---

## PURPOSE

The *L&H automation tools for ASR context management* allows users to perform some basic management operations on the ASRAPI database by means of command line executables.

These command line executables allow:

- The development of contexts for one application to be done at several different locations. The data in an ASRAPI database can be easily transported to a different machine.
- The extraction of the data from the ASRAPI database with the purpose of archiving it, storing it together with source code of the application or put it under a version control system.
- The automatic creation of context binaries suitable for engines on embedded systems. This allows you to integrate the creation of binary .ctx files in makefiles.

---

---

# FEATURES

The following functionalities are provided:

- Import and export of contexts in L&H Binary Export format.  
This format allows you to save contexts created or edited with the LexTool in a binary format (unreadable for humans).
- Import a BNF to an ASR context. Exception dictionaries can be specified to be used when compiling.
- Installation and extraction of dictionaries (in binary format) from ASRAPI database.
- Export a context to ASR1602 format.
- Merging and splitting of dictionaries.
- List information about objects currently in the ASRAPI database.

**Note:**

The tool allows you to specify the ASR engine and ASR database allowing you to use it on systems with multiple ASR engines installed.

## Introduction

There is one executable which can perform all actions. Several batch files are provided to make some actions easier to perform.

The executable is named `asrbatchtool.exe`. The basic operation mode is

*`asrbatchtool <command> [VariableSpecification...]`*.

The different commands are:

<b>Import</b>	Imports a context into the ASRAPI database
<b>Export</b>	Exports a context from the ASRAPI database to a binary format
<b>Install</b>	Installs a context, user or dictionary in the database.
<b>MakeInstall</b>	Makes an install file (directory) of an ASRAPI context, user or dictionary.
<b>MergeDict</b>	Merges two dictionaries to one
<b>ExtractDict</b>	Extracts a part of a dictionary and creates a smaller dictionary
<b>SaveDictList</b>	Saves the words present in a dictionary in a text file

The variables for each of these commands can be specified in the `asrbatchtool.ini` file or can be specified (overridden) at the command line. We call it variables instead of arguments because the order in which they are put on the command line is not important.

### **Note:**

The `asrbatchtool.ini` file is an MS Windows style `.ini` file. It must be ANSI or multibyte code; unicode `.ini` files are not supported.

You can specify variables on the command line by adding `-D<Var>=<Value>`. Each command needs several variables; to see what variables are used for a command, refer to the 'Commands and their usage' section below.

Through this method of working you have full control over every variable but you can also have a rather short number of variables on the command line by specifying the more or less static variables in the `.ini` file.

A sample of an `asrbatchtool` command line is:

```
Asrbatchtool makeinstall -DType=context -DUser=fv -DName=isdjeida  
-DDestDir=f:\test
```

It is also very easy to create batch files (`.bat`) based on `asrbatchtool` to make common use easier;

Commands like:

```
AsrMakeInstallCtx isdjeida f:\test fv
```

have the more classic batch file look & feel.



# asrbatchtool.ini

## Sections

### [Engine] Section

This section can contain variables and their values to be used by the asrbatchtool commands. The lines in this section have the following syntax: <Var>=<Value>. The variables themselves are case-insensitive but the values are case-sensitive.

### [Aliases] Section

The .ini file has an *[Aliases]* section, which contains abbreviations for the string variables to be used on the command line. Every string variable (NOT a number variable) specified in the *[Engine]* section can be replaced by an alias providing this alias is present in the *[Aliases]* section.

```
e.g. v3.2=asr3232.dll
      bufexp1602="ASR1602 Buffer Export"
      bnf="BNF Grammar Compiler"
```

This allows you to put things like -dformat=bnf on the command line. The alias itself is case insensitive but the value of an alias is case sensitive.

### [Tool] Section

The .ini file has a *[Tool]* section in which the tool variables reside. These variables are used in the same manner as the engine variables, but they apply to all commands.

Below is a list of the tool variables and their behavior:

TOOLSHOWUSAGEINWINDOW	If this boolean is set to 1 and if no arguments are given then a dialog box is shown with the usage. Default value: 0.
TOOLSHOWSTATUSONSUCCESS	Set this boolean to 1 if you want the tool to display a status message on the console when the operation was performed successfully. Default value: 0.
TOOLLANG	Language for the tool. Error messages and usage information are influenced by this parameter. Currently this can be enu (American English) or jpj (Japanese).

## Sample

This is how an asrbatchtool.ini and command line instruction could look like.

### Asrbatchtool.ini

```
[Engine]
AsrDll=asr3232.dll
DbName="ASR1602 - Office"
User=Generic
AutoCreateUser=1

[Tool]
ToolLang=enu
ToolShowUsageInWindow=1
ToolShowStatusOnSuccess=1

[Aliases]
v3.2=asr3232.dll
v3.2b=asr3232b.dll

bnf="BNF Grammar Compiler"
binimp="L&H Binary Import"
binexp="L&H Binary Export"
expl602="ASR1602 Export"
bufexpl602="ASR1602 Buffer Export"
jpj="Nihongo"
enu="American English"
```

### Command

```
Asrbatchtool makeinstall -Duser="Jim Smith" -DName=isd -DDestDir=a:\isd
```

### **Note:**

It is good practice to apply the following rules:

- Asrbatchtool.ini has the more fixed variables and does not change often.
- The command line arguments are the ones that change more often and need flexibility.

# COMMANDS AND THEIR USAGE

## Command Reference

Following is a full description of all commands and their variables.

***Note:***

If a variable is optional, it is put between square [ ] brackets.

## Import

### Description

This command imports a context into the ASRAPI database.

Import can be from one of these sources:

- Binary format: You can import the files created by the export function. an be used to import a binary context previously exported with the Export function.
- BNF file: This is your source file for a context (humanly readable). You can use this import function to automate the development process.

### Variables

ASRDLL	File name of ASR DLL
DBNAME	Database name (an engine can contain more than one database)
FORMAT	Format of what you want to import
USER	Name of user who will own the context
LANG	Language for which the Import grammar/file is written
DICT	Dictionary you want to use for importing (compilation)
SOURCE	Source file of what you want to import
CONTEXT	Name of the new context
OPTIONS	Extra options
[AUTOCREATEUSER]	Boolean variable to create user and register language if not done yet. Default is enabled
[DELETECONTEXTFIRST]	Boolean variable to delete the context before importing. Needed to overwrite a context that exists in other language. Default is disabled

### Comments

If you want to import or export with the AsrBatchTool application, you need to specify a format. This format string indicates from what type you are importing (or to what type you are exporting). These import and export types are installed as a plugin on the ASRAPI engine. So depending on what was installed on a certain machine, different formats will be available. The AsrInfo tool can be used to display what types are available. AsrInfo can display 4 different types of formats. The 4 formats each apply to one ASRAPI SDK function.

'Grammar formats' = grammar compilation from file  
(asrCtxCompileGrammar)

'Export formats'	= export context to file (asrCtxExport)
'Import buffer formats'	= import context from buffer (asrCtxImportBuf)
'Export buffer formats'	= export context to buffer (asrCtxExportBuf)

Depending on the parameters you specify on the command line, AsrBatchTool will use one of these functions.

Three of these four functions are supported by AsrbatchTool.

'Grammar formats'	= import	
'Export buffer formats'		= export
'Export formats'	= export subformat=UI	

In each of these three cases the format parameter should be a string displayed in the AsrInfo list.

For example, your command line could look like this

```
AsrBatchTool import -Dformat="BNF Grammar Compiler" ...
```

because "BNF Grammar Compiler" is one of the possible 'Grammar formats'

**Note:**

A format string can even be different for the same format type depending on what ASR version is installed.

## Export

### Description

This command exports a context from the ASRAPI database to one of the following formats.

ASR1602 format:	Contexts can be exported in this format. This binary format is useable on embedded systems. This makes it possible to create your target binaries automatically.
Binary format:	Save a context in the L&H binary format. This exports a context from the ASRAPI database to one binary file. This binary file could be used for version control purposes.

### Variables

ASRDLL	File name of ASR DLL
DBNAME	Database name (an engine can contain more than one database)
USER	User owning the context
CONTEXT	ASRAPI database name of context
FORMAT	Format of what you want to export
[SUBFORMAT]	Specifies what export function should be used. Internally this variable decides if <i>asrCtxExport</i> or <i>asrCtxExportBuf</i> is used. The valid formats for these functions can be seen with the asrinfo program. This variable is optional and can be set to 'UI' to use the <i>asrCtxExport</i> function ( <i>asrCtxExportBuf</i> is the default). When UI (stands for user interface) is specified the batch tool will display some dialog boxes to specify output files. When 'UI' is not specified, the parameters specified below are used
DESTFILE	Destination file name. If <i>SPLITCTXBIN</i> is not set to 1 then this is the ASR1600V2 export type binary which contains .ctx and .wcl. Use .bin as extension
[SPLITCTXBIN]	Boolean which is default 0. If this variable is set to 1, the output of the export operation will be a .ctx and .wcl file. If you want to export immediately to a .ctx and .wcl, you should specify the .ctx name (with extension) in the DESTFILE parameter. The program will write a .wcl with the same base name in the same directory. This parameter is only valid for the "ASR1602 Buffer Export" format
OPTIONS	Extra options

## Comments

If you want to import or export with the AsrBatchTool application, you need to specify a format. This format string indicates from what type you are importing (or to what type you are exporting). These import and export types are installed as a plugin on the ASRAPI engine. So depending on what was installed on a certain machine, different formats will be available. The AsrInfo tool can be used to display what types are available. AsrInfo can display 4 different types of formats. The 4 formats each apply to one ASRAPI SDK function.

'Grammar formats'	= grammar compilation from file (asrCtxCompileGrammar)
'Export formats'	= export context to file (asrCtxExport)
'Import buffer formats'	= import context from buffer (asrCtxImportBuf)
'Export buffer formats'	= export context to buffer (asrCtxExportBuf)

Depending on the parameters you specify on the command line, AsrBatchTool will use one of these functions.

Three of these four functions are supported by AsrbatchTool.

'Grammar formats'	= import
'Export buffer formats'	= export
'Export formats'	= export subformat=UI

In each of these three cases the format parameter should be a string displayed in the AsrInfo list.

For example, your command line could look like this

```
AsrBatchTool export -Dformat=" ASR1602 Buffer Export" ...
```

because " ASR1602 Buffer Export" is one of the possible "Export buffer formats"

### **Note:**

A format string can even be different for the same format type depending on what ASR version is installed.

## Install

### Description

This command installs a context, user or dictionary in the ASRAPI database.  
This is suitable if you want to recreate the database from one machine onto another.  
You have to specify the .stp file, which was created with **MakeInstall**.

### Variables

ASRDLL	File name of ASR DLL
DBNAME	Database name (an engine can contain more than one database)
SETUPFILE	Name of .stp you want to install For this operation to succeed, you need to specify a directory (absolute or relative) for the .stp file. For example: -dsetupfile=test.stp will fail while -dsetupfile=.\test.stp wil succeed



## MakeInstall

### Description

This command makes an install directory from an existing ASRAPI database object (user, dictionary or context).

This binary data can then be archived for backup or transported to another PC for import.

### Variables

ASRDLL	File name of ASR DLL
DBNAME	Database name (an engine can contain more than one database)
TYPE	What to extract from database, can be context, user or dictionary
LANG	ASR language the user or dictionary belongs to (In case TYPE was user or dictionary)
USER	User who owns the context (In case TYPE was context)
NAME	Name of what you want to make an install of: context, user or dictionary
DESTDIR	Destination directory for install files (one .stp and one directory)

## MergeDict

### Description

This command merges two dictionaries into one. The two source dictionaries are ASRAPI database dictionaries.

### Variables

ASRDLL	File name of ASR DLL
DBNAME	Database name (an engine can contain more than one database)
LANG	Language of the dictionaries. Merging is only possible between dictionaries of the same language
DICT1	First source dictionary
DICT2	Second source dictionary
DESTDICT	Destination dictionary, (merged dictionary)

## ExtractDict

### Description

This command extracts a part of a dictionary and save it in a new ASRAPI database dictionary based on a text file containing the words that need to be extracted from the source dictionary.

### Variables

ASRDLL	File name of ASR DLL
DBNAME	Database name (an engine can contain more than one database)
LANG	Language of the dictionaries. Merging is only possible between dictionaries of the same language.
DICT	First source dictionary
WORDLIST	Name of the text file that contains a list of words which are a part of the DICT dictionary and that should be extracted and put into a separate dictionary. You can create such a text file with the <b>SaveDictList</b> command
DESTDICT	Destination dictionary, (partial dictionary)

## SaveDictList

### Description

This command saves a text file with the words present in a dictionary. The output can be used for the **ExtractDict** command.

### Variables

ASRDLL	File name of ASR DLL
DBNAME	Database name (an engine can contain more than one database)
LANG	Language of the dictionary. A dictionary always belongs to a language. The ASRAPI dictionaries per language can be viewed with the <i>asrinfo</i> program
DICT	ASRAPI dictionary name
WORDLIST	File name of the outputted text file which will contain the words of the dictionary. This text file can be used in the <b>ExtractDict</b> command.