

# Manual for Real-Time AI Lab Work (SpeechMania)



Jelle de Haan, May 1999  
Translated and updated by  
Boi Sletterink, 2000-2003

# Real-Time AI Assignment: SpeechMania

## Introduction

While the university distributes a DUT-phone directory every year, not all people have one at hand. Therefore it is often cumbersome to find a teacher's or staff member's phone number for staff and students. The alternative is to look it up on the internet. This electronic version of the directory is not well known and only accessible through a computer with internet access. The goal of this assignment is to make available a single point phone-based information system that automatically retrieves phone numbers of our group's staff members.

## Assignment

Design and implement a dialog with SpeechMania that recognizes a staff member's name, retrieves the phone number and informs the speaker. The actual search of the phone number can be done by a prefab library that retrieves a phone number from the electronic version of the directory by name.

The program only needs to recognize the names of the staff members of the KBS group. Therefore the application must know the names of all KBS group staff members (that are available in the phone database). The directory is at <http://ldap.tudelft.nl:8888/> . You can find the names at <http://www.kbs.twi.tudelft.nl/People/Staff/> .

## Global planning

The design and implementation of the system are to be spread over five afternoons:

1. Design the dialog in SpeechMania. Make a start on the implementation in HDDL, SpeechMania's programming language.
2. Finish HDDL code and offline testing with DCOFFLIN.
3. Build an application lexicon and generate prompts.
4. Online testing and debugging, optional model evaluation and training.
5. Final tests, presentation and demonstration.

SpeechMania and HDDL manuals are available in the lab, and the necessary parameter files and an example HDDL file that demonstrates the use of the phone number retrieval library are available in each group's home directory (drive H:). This example does not provide a decent dialog, so you will have to come up with your own dialog.

## Course evaluation

Evaluation of each group's results will be based on a report, presentation and demonstration. On the last day, each group will present their work, with a short report describing the dialog structure and code, and solutions to specific problems encountered during the design and implementation phases. After each presentation the work should be demonstrated online.

## Writing the HDDL program

Before you start hacking your HDDL program, you should have an idea of how the conversation on the phone should be. The best way to get it clear is to draw a sort of flow chart, which shows the directions the conversation can get into and possible actions to take. This is also a good plan because you'll have to include the dialog design in your

After designing your dialog, you can start implementing it. Since most people are not familiar with HDDL, it is probably a good idea to modify the example after making a backup copy. The file name is important, since it is set in the configuration files, which are used both by the building tools and by the online environment. The configuration files are fairly complex and fragile, so changing those is not recommended and at your own risk.

## Building the Lexicon

Before an application can be run, one must make a lexicon. The lexicon lists all words that can be recognized and how they should be pronounced. Here is a description of all steps necessary to create a proper lexicon.


1. Generate a wordlist with HDDLPARS. Use the following commandline:

```
hddlpars prm\lexicon.prm
```

This will generate several messages. You can safely ignore INFO messages and most WARNING messages. If you see any ERROR messages, your sources need to be fixed.

The folder `lexicon` will now contain the file `directory.wdl`. This file contains all words that should be recognized by SpeechMania.

2. Start the lexicon manager (Start | SpeechMania | Lexicon Manager 2.0 or run **lexmgr** from the commandline). Add a new lexicon with File | New.
3. Add a custom language from the Tools menu. Fill in:  
Language: `directory`  
Background Lexicon: `C:\PHILIPS\SPLANG\DUTCH\LEXICA\NL_gener.lex`  
User lexicon: `user.lex` (in your homedir's lexicon folder)  
Lexicon Model File Name: `C:\PHILIPS\SPLANG\DUTCH\AUTOTRANS\Nl5.plm`  
Lexicon Mapping File Name: `C:\PHILIPS\SPLANG\DUTCH\AUTOTRANS\Nl.map`  
Phoneme Inventory File Name:  
`C:\PHILIPS\SPLANG\DUTCH\AUTOTRANS\Nl.phi`  
In case the file do not exist in these directories, you can find them in one of the directories below. This depends on the installation.
4. Add wordlist by selecting Tools | Add | Wordlist. Select the wordlist generated in step 1. This will generate an application lexicon, `directory.lex`.

5. Change the lexicon version, by clicking on the Database View button , doubleclick on Database View, `directory` and Application Lexicon List. Select `directory.lex` and rightclick on it. Choose Set Version Of Application Lexicon | Version 1
6. Generate a standard transcription with Process | Transcribe All. This transcription is based on rough Dutch pronunciation rules; odd or foreign words such as names typically go wrong here, so you'll have to check and correct the transcription.
7. Check phonetic spelling of all words. Select `directory.lex` and rightclick on it. Choose Manual Transcription. You can edit the transcription in the Temporary Phonetic List. For an overview of SpeechMania's phonetic spelling language, see the Phonetic Transcription Guidelines (page 13 contains a list of phonemes and symbols) and appendix A, SpeechMania SAMPA for American English with a Dutch recognizer. Make sure to click on Apply to Lexicon(s) after making changes.
8. Save the Lexicon Manager database file as **lexicon\directory.lxd**. Close the Lexicon Manager.

## Recording prompts

SpeechMania cannot generate speech by itself, it can only play samples. Therefore, all prompts have to be either pre-recorded, generated in advance with a speech synthesizer program such as Fluency or generated on-the-fly by using a special module (fluency.dll). While recording your own prompts can certainly give your application a very personal touch and give a better result when done professionally, it is usually hard to make recordings of decent quality. Typical problems include background noise (from the classroom) and different sound levels within the same sample. Therefore, we recommend using the text-to-speech synthesizers for this exercise.

The Fluency Speech Editor is installed on our systems. Unfortunately, it is a Dutch text-to-speech tool, but you can change the pronunciation and intonation to get what you want. There are also a few web-based (English) text-to-speech resources that allow you to generate samples from text, such as from AT&T (<http://www.research.att.com/~ttsweb/cgi-bin/ttsdemo>) and Lernaut & Hauspie (now owned by ScanSoft, website at <http://www.lhsl.com/realspeak/demo/>). Make sure your samples are 16 bit, mono, 8kHz uncompressed WAV files. You can use the standard windows tool Sound Recorder in Start | Programs | Accessories | Multimedia | Sound Recorder to convert them if necessary. On Win2000 systems you can find it in the Entertainment folder instead of the Multimedia folder.

On-the-fly generation is also possible, but requires major changes in the sourcecode and will not be covered here. One major disadvantage of it is that it only pronounces Dutch sentences correctly. Here is how to record or generate prompts:

1. Generate a promptlist with HDDLPARS: `hddlpars prm\phrases.prm`. This will generate a file `phrases.dat` (in the directory where you start it) which contains an educated guess of all prompts the system can say.
2. If you're going to record prompts with a microphone, make sure recording works properly. You can use the Sound Recorder to check this. Recording prompts with a microphone is not advisable unless you know how to do it right. A better solution is to generate the prompts with Fluency or use one of the web-based tools. Type the sentence, play it and save the waveform.
3. Start the Recording Station program (from the start menu or run `rcstat` from the commandline. Create a new Phrase Database. Choose `H:\phrases\directory.ddf` as name. Choose defaults for other items.
4. Import the phrase list generated in step 1 with `File | Import Phrase List`.
5. Check the list of prompts. HDDLPARS retrieves most prompts, and adds all strings it encounters, but dynamically generated prompts will not be found. To add a prompt, use `Phrase | Insert Phrase`. After prompts are added choose `Cancel`. To delete a prompt, rightclick on it and choose the appropriate menu item.
6. To record a prompt with a microphone, use `Recorder | Start Recording Session`. To use a pre-recorded or generated sample, you can just assign the right file with the `Phrase | Assign File With Original Recording`. You can use Fluency to generate prompt samples.
7. Normalize the prompts. This needs to be done in order to make sure all prompts have about the same soundlevel. Select all prompts with the Edit menu and then normalize them with `Phrase | Normalize`. You can check the results by playing the normalized prompts. If the results are not satisfactory, fiddle with the settings in `Options | Settings | Post Processing` and normalize them again.
8. Exit the recording station. All results will be saved automatically.

## Testing the application

There are two ways to test your application: offline (simulating speech input by typing on the keyboard) and online (the real thing). Offline testing can be done on any NT machine with the SpeechMania software installed; online testing only on the machines that have a phone interface card installed.

### Offline testing

1. From the command prompt, start **dcofflin prm\dcofflin.prm**.
2. Type **c start** to "pick up the phone". Any input starting with a space will be seen as input coming from the speech recognizer. Other commands include **c too\_long** and **c nothing\_recorded** to simulate error conditions. For a more detailed overview, see page 52 of the Application Creation Environment manual .
3. Type **c stop** to quit.

### Online testing

Run the Dialog Application Manager (WinDAM) on the machine with the phone card. Since there is only one machine available with a phone card, please use it as little and as short as possible. The default settings will make the dialog manager run on number 89573. It should pick up before the second ring.

## Evaluating the application (optional)

Logs from actual trials of the application can be used to evaluate and improve its performance. This evaluation can be used to spot common problems in the application and to train the recognizer. For this course, evaluation and training is not necessary, but if you have some time left, you can use it to possibly improve recognition rates. For real applications, one should use a different corpus for training and evaluation, and quite a number of dialogs are necessary for a useful training set.

### Dialog Transcription

In order to have a correct reference, all dialogs must be transcribed.

1. Start the Transcription Station. Create a new dialog database called **dialogs/directory.ddb**. Choose defaults for the other options.
2. Transcribe the dialogs. Doubleclick on the first dialog to start. The first soundfile is played. Correct the transcription generated automatically by the recognizer and click Accept. When all sentences in the dialog are corrected, the program will ask you to whether to switch to the Attribute View. Click Yes. Select `Item | Insert Attribute`. Click on Insert and Ok. Click Accept, Yes, Yes, Yes to continue with the next dialog.

### Evaluation

3. Create a corpus by running `corpmgr prm/corpmgr.prm`. Check the generated `corpus.wdl` file for words that are not in the lexicon. If there are any words that are not in the lexicon, transcription went wrong. Repeat the transcription step.
4. Run the evaluation program with `sreval prm/sreval.prm`. Results are written to `sreval.res`.

### Training

1. Run `LMTRAIN prm/lmtrain.prm`.
2. Check if recognition improves: `sreval prm/sreval_lm.prm`. Results are written to `sreval_lm.res`.

# Appendix A

## SpeechMania SAMPA for American English with a Dutch recognizer

Unfortunately, we only have a Dutch speech recognizer with our SpeechMania installation. Fortunately, Dutch and English are similar enough that this is not really a problem. The following two tables show the equivalent or similar symbols for phonemes. Since some symbols are only approximations, you may find that using another similar symbol works better for you; this also depends somewhat on your own accent. Typically, the list given here will work best for people with a Dutch accent.

This list is based on the [SAMPA for American English](#) page. For more information on phonetic transcription with SpeechMania, see the Phonetic Transcription Guidelines manual.

Consonants				
Symbol	(SAMPA)	Example word	Transcription	(in SAMPA)
p	(p)	pin	pIn	(pIn)
b	(b)	bin	bIn	(bIn)
t	(t)	tin	tIn	(tIn)
d	(d)	din	dIn	(dIn)
k	(k)	kin	kIn	(kIn)
G	(g)	give	GIv	(gIv)
tS	(tS)	chin	tSIn	(tSIn)
dZ	(dZ)	gin	dZIn	(dZIn)
f	(f)	fin	fIn	(fIn)
v	(v)	vim	vIm	(vIm)
t	(T)	thin	tIn	(TIn)
d	(D)	this	dIs	(DIIs)
s	(s)	sin	sIn	(sIn)
z	(z)	zing	zIN	(zIN)
S	(S)	shin	SIn	(SIn)
S	(Z)	measure	mES@	("mEZ@`)
h	(h)	hit	hIt	(hIt)
m	(m)	mock	mAk	(mAk)
n	(n)	knock	nAk	(nAk)
N	(N)	thing	tIN	(TIN)
r	(r)	wrong	rON	(rON)
l	(l)	long	lON	(lON)
w	(w)	wasp	wAsp	(wAsp)
j	(j)	yacht	jAt	(jAt)



## Vowels

Symbol	(SAMPA)	Example word	Transcription	(in SAMPA)
I	(I)	pit	pIt	(pIt)
E	(E)	pet	pEt	(pEt)
A	(I)	pat	pAt	(p{t)
A	(A)	pot	pAt	(pAt)
A/Y	(V)	cut	kAt/kYt	(kVt)
u	(U)	put	put	(pUt)
i	(i)	ease	iz	(iz)
e:	(e)	raise	re:z	(rez)
u	(u)	lose	luz	(luz)
o:	(o)	nose	no:z	(noz)
O	(O)	cause	kOz	(kOz)
a:I	(aI)	rise	ra:Iz	(raIz)
OI	(OI)	noise	nOIz	(nOIz)
Au	(aU)	rouse	rAuz	(raUz)
Yr	(3`)	furs	fYrz	(f3`z)
@	(@)	allow	@laU	(@"laU)
@	(@`)	corner	kOrn@	("kOrn@`)

NOTE: The first column in the table is the Dutch SpeechMania "dialect" and it is what you can use. The second column, between parenthesis, is the official Sampa for American English symbol for the phonemes. You should not use this notation in SpeechMania as it uses a different notation and the recognizer only knows Dutch phonemes anyway. The third column is an example in plain American English; the fourth and fifth columns are the phonetic translations in SpeechMania dialect and SAMPA, respectively.