~~Oxygen Alliance 2003 Workshop – February 24-28, 2003~~

SpeechBuilder Hands-on Activity: Controlling Jaim Through Speech Commands

1 Introduction

In this assignment, we will use SpeechBuilder and the Galaxy speech processing system to create a speech interface to our instant messenger client, Jaim. The objective is to allow for the user to use voice commands to perform various functions in Jaim. Your first and primary task will be to add support for connection commands, such as "Connect me to Bob." The audio for the voice commands will come ~~either~~ from your handheld ~~our your tablet PC~~. ~~The Galaxy system and SpeechBuilder will both run on your tablet.~~ Your Jaim program will connect to Galaxy using the "frame relay" – a special Galaxy server that allows external applications to communicate with the speech processing system.

2 Learn the SpeechBuilder Interface

• First, you need to make a SpeechBuilder domain for the commands that you wish to support. Access SpeechBuilder ~~on your tablet~~ by going to:

`http://~~localhost~~/SpeechBuilder/SpeechBuilder.cgi`

2.1 Domains

Your login is `guest`, password also `guest`. You will have one domain in your account, named "house." This is a small toy domain you can use to get familiar with the SpeechBuilder interface.

• Once you get to the main splash screen, select the `house` domain and hit "Edit." When you select a domain, you start at the action/attribute editing screen. In SpeechBuilder, you navigate with the drop-box at the top right of the screen (the one that says "Select another domain"). In this screen, you can view the details of a particular action or attribute.

2.2 Editing actions and attributes

• Take a look at the "status" action by selecting "status" in the action drop box and hitting the "Edit" button. You can add new sentences to the action by typing them in the editing window and hitting "Apply." You can select sentences to edit in the selection window, and hit "Edit" to pull them in the editing window.

• Think of another way of asking about things in the house (e.g. "i am wondering if the lights are turned on in the living room") and add it to the action. The example sentences in the "house" domain have various grammar syntax elements (e.g. "|", "[ ]"). You can familiarize yourself with the syntax of these, but you don't have to use them. Just type a sentence as you would say it in English. You don't need any capital letters or punctuation.

• From this screen, you can also edit attributes. Select the "room" attribute and hit "Edit." Add a new type of room to the list.

Once you add a room to the attribute "room," it becomes interchangeable with all the other rooms in the action sentences. For example, if you add "bathroom," you gain the ability to say "are the lights in the bathroom on" without explicitly adding this sentence to the "status" action. This generalization is a powerful feature of SpeechBuilder.

2.3 Back-end program location

As was discussed in the presentation, SpeechBuilder is able to configure a complete speech domain based on only a database table. However, in this lab we will use a different configuration, in which the domain

connects to an external application. The external application, in our case, will eventually be Jaim. However, for the house domain we will use an "echo script" – a CGI script that just repeats what the system thinks you said.

• From the main navigation pull-down, select "Edit back-end script URL." Make sure that the URL is `http://localhost/cgi-bin/echo.cgi` – the echo script. You shouldn't need to edit it for now.

2.4 Compiling the domain

Next, we will "compile" the domain. Compiling creates all the grammars and configuration files necessary for the Galaxy components to run properly.

• Click on the "Compile" button in the upper right of the SpeechBuilder screen. You may have to wait a few seconds as the domain is compiled. Skim over the output and see if you can figure out what is happening. **Note: You will need to compile the domain each time you make a change. No changes will take effect until you compile.** Don't worry if you get some chmod errors, that is normal.

2.5 Reduced sentences

SpeechBuilder can "reduce" your action sentences for you to a CGI-like meaning representation. This is done using the TINA natural language processor, which is also used to process the speech recognized during the runtime of your domain.

• Select "See reduced sentences" from the pulldown list and hit "Go." Take a look at the sentences. Note that only the words specified in the attributes make it into the frame. TINA extracts the meaningful words in a sentence by processing it according to a hierarchical parsing grammar.

• Click on one of the sentences to see a graphical representation of the TINA parse tree. The grammar for the house domain is relatively flat, so you don't see much hierarchy in the parse tree. We won't be needing any concept hierarchy for our Jaim domain, either. However, a developer can enforce a concept hierarchy in the parsing grammar by using hierarchical concepts (as outlined in the presentation).

2.6 Talking to your domain

Now that you have compiled your domain, you can actually talk to it!

• You need to start several things on your machine to make this possible:

- We will use an open source speech synthesizer called Festival. Festival is not part of Galaxy, but we will use it for now (in the future, SpeechBuilder will give you the ability configure your own concatenative speech synthesizer). Open a new window and start Festival by typing

  ```
  festival --server
  ```

- Open a new window and start the Galaxy servers. You will need to go to the domain directory; do so by typing

  ```
  cd /home/sls/Galaxy/SpeechBuilder/users/guest/DOMAIN.house
  ```

  Then, open a new window, and start the domain by typing

  ```
  ./oxclass.cmd yes yes
  ```

  The first "yes" tells the script to pop up windows to show you the output of the various servers that it will run (e.g. speech recognition, natural language processing, etc.) The second "yes" tells the domain to use Festival. In the future, you can say "no" for the first switch if you don't care about the output of the servers (but it usually useful for debugging). Take a look at what servers get started. Try to figure out what each one of them is trying to do.

- Adjust the mixer. On the tablet, go to the KDE menu → Multimedia → Sound Mixer. Then adjust your main volume up to about 80% of maximum. On the iPAQ, go to Penguin → Audio → mixer. Adjust the volume up to max.
- Start the audio client. You can run it on either the tablet or the handheld. First, open a new window. Then, on the tablet, type

  ```
  galaudio /dev/dsp (TABLET_HOSTNAME) guest 12126 push
  ```

  On the iPAQ, type

  ```
  galaudio /dev/sound/dsp (TABLET_HOSTNAME) guest 12126 push
  ```

  What do all these parameters mean? `/dev/dsp` and `/dev/sound/dsp` are Linux audio input/output devices that the `galaudio` client uses to talk to the microphone and the speaker. They are different on the tablet and the handheld because the audio hardware and drivers are different. The `TABLET_HOSTNAME` is the address of the speech technology server; substitute your tablet's name here. `guest` is the identifier for your domain. In SpeechBuilder domains, this is always the login of the developer. `12126` is the port number of the audio server to send the data to. `push` tells the audio application to operate in push-to-talk mode, meaning that you have to hold down a button while you are talking.
- Okay, you should now be able to talk to your domain! On the tablet, hold down the left shift key while you are talking. On the handheld, hold down the record button (on the left side of the iPAQ – same button as for face recognition).

3 Creating a SpeechBuilder domain for Jaim

Your next task is to design a SpeechBuilder domain for the Jaim instant messenger client. Remember that our initial goal is to be able to connect to your friends using speech. So instead of selecting our friend's name in the roster and clicking "Chat," we will merely say a command such as "Connect me to Jason."

• Go back to the SpeechBuilder interface. From the main navigation pull-down box, select "Select another domain." Create a domain by typing "jaim" in the domain name box and clicking "Create." Next, click "Edit" to start editing the domain.

Next, you must create actions and attributes for your domain. You should make an attribute to represent the names of the people that you will be contacting, and an action to specify example sentences for saying the command.

• Create an attribute called "person" by typing "person" in the Attribute type-in box (second from the top) and clicking "Create." Then, type in the names of your friends in the class – the people you would like to contact using Jaim. Don't forget to click "Apply."

• Create an action called "connect" by typing "connect" in the Action type-in box (the top one) and clicking "Create." Then, type in some example ways that you would tell the system to connect to your friends. For example, "i would like to talk to kevin" or "please let me talk to jason". You don't need to use any capital letters, but make sure that the spelling of the names of your friends is consistent with the spelling you used inside the Attribute. Remember that the system automatically generalizes on your example sentences, so you don't have to repeat the sentence for each person in your domain (for example, you don't need to type both "talk to jason" and "talk to kevin"). The more different examples you give, the better coverage your domain will have – so type in a bunch!

• You need to edit the back-end script setup. From the main navigation pull-down, select "Edit back-end script URL." Set your domain up to use the frame relay server instead of a CGI script (remember the presentation?) by typing "`relay:guest_jaim`". Don't add any spaces or punctuation. **Make sure you hit "Apply" after you do this. Otherwise your changes will have no effect.**

• Next, click "compile" to compile your domain.

• Take a look at the reduced sentences (as described above) to make sure that all of your sentences got parsed properly. Make sure the frame for the connect action looks something like "`action=connect&frame=(person=jason)`". If it doesn't you may need to add something to your domain – ask for help.

• Now, we can run the domain and talk to it. First, we will use a python "echo script" as the back-end for this domain. You can get an example echo script by typing:

```
wget http://server/test-echo.py
```

• Bring up an xemacs window with this file: `xemacs test-echo.py`. Please study it carefully; read all the comments and understand them. You will definitely not be able to get very far in the assignment without an understanding of this script. Any questions?

Okay, now we are ready to run the new domain.

- Start the Galaxy servers. You will need to go to the domain directory; do so by typing

  ```
  cd /home/sls/Galaxy/SpeechBuilder/users/guest/DOMAIN.jaim
  ```

  Then, open a new window, and start the domain by typing

  ```
  ./oxframe.cmd yes yes
  ```

  Note that we are using a different startup script, `oxframe.cmd`, since we are using the frame relay server with this domain.

- Open a new window and start the echo back-end script:

  ```
  python test-echo.py
  ```

- Open a new window and start the audio client. Probably better run it on the tablet for now for easier debugging:

  ```
  galaudio /dev/dsp (TABLET_HOSTNAME) guest 12126 push
  ```

A few notes about the concept of turn management in Galaxy. The python script, through the frame relay, acts as the turn manager. A "conversation" begins when an audio client `galaudio` connects to Galaxy. The turn manager (your script) then takes over. Once a conversation is over (or you kill the audio client), your turn manager script can keep running. But, you will need to restart the audio client to start another conversation.

4 Incorporate speech support into Jaim

First, a note: if you still have your face ID code in Jaim and the face ID server is not running, your Jaim code will crash. You may need to either take this code out or start up your face ID server again.

All right, it's finally time to integrate things! Remember the GTK event loop in Jaim? For face ID support, we modified the login GTK event loop. For speech support, we will modify the main GTK event loop inside the `run` subroutine. Open up `jaim.py` and find this loop. It looks like this:

```
while 1:
    while gtk.events_pending():
        gtk.mainiteration()
    self.con.process(JABBER_TIME_SLICE)
```

Basically, this task involves integrating code to process the speech commands into this event loop.

• You should start by fitting the code from `test-echo.py` into your Jaim application. Instead of the loop started by `while (active)`, you can just paste the contents of the loop into the Jaim code. Then you should effectively have a working back-end echo script inside Jaim. Now, all you need to know is how to use the information coming from the Galaxy server to open up a chat window in Jaim. Figure 1 (on next page) has some example code that does just that:

```
elif (action == 'connect'):
   # Get the name
   name = c.frame.getAttribute('person')

   # If we actually got a name
   if (name):
      # Build up a ``Jid'' -- Jabber ID
      # The ``Jid basically uniquely identifies a particular user on a
      # particular server

      domain = self.ourJid.getDomain()
      userJid = jabber.JID(node=name,domain=domain)

      # If we don't already have a window open for this person
      if name not in ChatWindow.windowList.keys():
         # Open a window
         ChatWindow(self.rosterWindow.ourJid, userJid, self.rosterWindow.con)
         # Speak a reply
         c.sb_reply('Connecting to ' + name)
      else:
         # Speak a reply
         c.sb_reply('Already connected to ' + name)
```

Figure 1: Sample code to control Jaim connections

Okay, this should be enough for you to complete your assignment. This is a lot to digest, so please take your time, and... questions are welcome!

A quick note as to what else you can do with this. There is a number of things that can be controlled with speech inside Jaim. For example: sending messages, killing Jaim, registration, etc. If you are done with this lab early, feel free to try any of these!