Dialog based user interfaces featuring a home cooking assistant

Author: R. M. Wasinger  
School of Electrical and Information Engineering  
The University of Sydney  
email: rainer.wasinger@wasinet.com  
Supervisor: Prof. Hansen Yee

Abstract: Natural language dialog components have improved significantly since they were first discussed more than 20 years ago. This paper leverages the current position of the maturing speech industry and explores effective user interface design for speech applications. In particular, discussion focuses on the creation of a home cooking assistant based on a multi-modal design. Issues relating to the implementation of dynamically changing multiple-context rule grammars are also discussed. So too are methods of interacting with an underlying database.

Keywords  
Dialog based user interfaces; multi-modality; multiple-context and dynamically changing rule grammars; home cooking assistant.

1. Project Goals

The goal is to design and build effective dialog based systems. This goal can only be accomplished through a careful study of user interface design, rule grammar design and design of components that the system interacts with including a database and a properties file. These issues will be discussed in the context of a home cooking assistant.

The home cooking assistant is a system that incorporates audio, visual and touch into a friendly cooking companion. Coded in Java, its primary function is to supply the user with recipes and to guide them in the cooking process. It also allows for new recipes to be incorporated into an accompanying database and is expected to appeal to markets including magazine and cookbook publishers, restaurants and computer owners.

2. Background

With regards to natural languages, multi-modality refers to the use of multiple communication channels in a user interface. The current most popular communication channels include sight (eg. monitor) and touch (eg. keyboard). Sound (eg. microphone) is a new addition to the “box of crayons” and is used in the home cooking assistant. Sound, and in particular speech, has three important characteristics that must be observed when designing a Graphical User Interface (GUI). Speech is transient, invisible and asymmetric. It is transient because once it has been spoken it is gone. Graphics displayed to a screen on the other hand are persistent. It is invisible because you cannot see speech. Finally, it is asymmetric because people can speak a lot more efficiently than they can listen and understand what is being said to them.

Speech is not always appropriate. This is illustrated by the following properties:
- The task can be more easily accomplished using a mouse and keyboard.
- The users work environment is particularly noisy.
- Large quantities of information must be presented to the user.
- The task requires the user to compare data items.
- The information is personal or confidential.
In these instances, the influence of design factors can still lead to a useable interface. For example, if the work environment is particularly noisy, the grammars could be minimised to achieve higher levels of accuracy. This is the case in the home cooking assistant where there is a degree of noise present from background devices such as blenders, ovens, exhaust fans and radios.

In order to fully speech-enable an application, an input and an output channel are required. Speech input is needed from the user for commands such as “find a recipe”. Speech output is also required from the system for feedback and verification purposes.

Speech recognition is concerned with converting user speech - from a microphone - into parseable tokens that can be interpreted by the system. The particular speech engine capable of performing this task is a Speech-To-Text (STT) engine.

Speech synthesis on the other hand is concerned with converting text from the system into speech and presenting this to the user. A Text-To-Speech (TTS) engine performs this task.

Speech engines use phoneme dictionaries to interpret words into sounds and sounds back into words. Phonemes are the basic unit for speech and are commonly found in dictionaries in the form of pronunciation guides.

A typical speech application such as the home cooking assistant can be divided into 3 layers. The speech engines are on the bottom layer. The speech interfacing libraries make up the middle layer and the actual application is on the top layer.

3. Proposed system

The goal of the home cooking assistant is to find recipes, detail recipes and allow for the addition, removal and modification of recipes.

Another goal of the program is to function using three different sets of modalities. This was to allow maximum flexibility to the user in terms of required software (such as a speech engine) and hardware (including keyboard, monitor and microphone). The modality groups are:

- Speech only.
- Touch and visual only.
- Speech, touch and visual.

The GUI for the home cooking assistant can be seen in figure 1.

The application is programmed in Java and uses the Java Speech Application Programming Interface (JSAPI) to interact with the speech engines. The JSAPI compliant engines being used are the IBM ViaVoice STT and TTS speech engines. These components illustrate the 3 layered approach described above.

The speech engines as used by the home cooking assistant use rule grammars to interpret what the user has said. These grammars – in contrast to dictation grammars – do not require any initial training. Furthermore, the language defined by these grammars is often fairly small when compared to dictation grammars. This places a limit on what the user is allowed to say but also has the effect of significantly increasing the accuracy of the overall system.

Other components that the system interacts with include a database and a properties file. The database is a Microsoft Access database. A database was implemented because it shifts the majority of error and recipe format checking to the database, which already has features to deal with these problems. The properties file is currently being used to make certain system properties accessible to a user. These properties include the “username” with whom the assistant will talk to, the “speaklength” to indicate how much text the synthesizer should attempt to speak back to the user and a variable called “detailed” allowing the user to choose from two different grammars, one containing a small set of rule grammars and another containing a larger set of rule grammars.

The main reason why it is believed a system such as the home cooking assistant could exist is because the eyes and hands are both busy while cooking. This would normally prevent the chef...
from reading the somewhat small writing found inside cookbooks.

4. Advantages of such a system

To better understand why such a system may be required, it is first important to look at typical tasks performed in a kitchen. These include:

- Writing shopping lists and unpacking shopping.
- Deciding what to cook and cooking.
- Washing dishes and packing them away.

The majority of tasks performed in the kitchen, aside from cooking are fairly monotonous. However, they all require both the use of eyes and hands. This restricts a user’s ability to perform anything else while carrying out one of these tasks.

When compared to a paper cookbook, the following advantages become apparent:

- Easier to use: Can be used from the other side of the kitchen and when the eyes and hands are busy.
- Time saving: Other tasks can be performed while using the assistant.
- Companionship: Future versions will provide a degree of chitchat to prevent the user from getting bored while in the kitchen.
- Database customisability: The user can store the recipes they want to, unlike a book, which from a user’s perspective contains both good and bad recipes.

5. System Architecture

There are at least three different architectures for speech systems [2]: system initiative, user initiative and collaborative. System initiative systems are the simplest type and typically involve the computer telling the user what to do. This makes for a particularly rigid system. User initiative systems rotate these roles by putting the user in charge of the computer. The home cooking assistant is based on this type of architecture. A third type of system is based on a collaborative effort between the computer and the user. It can be used in complex situations including crisis management and solution planning.

There are also different types of rule grammars, which can be used. The home cooking assistant uses multiple context grammars. Each state of the system (as shown in figure 2) is based on a different context. This increases accuracy by creating a larger number of small and independent grammars. In the assistant for example, choosing a recipe is a different context to detailing a recipe. Fixed context grammars in contrast to multiple context grammars are simpler and are based on a single grammar.

A scenario illustrating how the home cooking assistant is initialised can be seen in figure 3:

6. Testing and results

The system was tested on two different levels: code and usability. The platform for testing was Windows 2000. Based on the results returned from the testers, changes were made and ideas for future work generated.
The results from figure 4 show that over 80% of people thought that speech technology was beneficial to the user. Results also showed that just on 50% thought the program had some difficulty in understanding them. In defence of this result, based on the observations and walkthroughs performed, certain aspects were common to most people who achieved below average recognition results. The microphone placement was often too far away and the user was often not speaking audibly. This includes speaking too quietly, too fast or with broken speech.

It generally took users 15 – 30 minutes of microphone training before the recognition results increased. Note that rule grammar systems do not require any system training by the user. The only training required is in terms of becoming familiar with the microphone being used.

### 7. Future work

From the results and opinions obtained from the testers, there are several aspects, which will be changed in future versions of the home cooking assistant. These include:

- Menu bar: Containing help, access to recipe details via the keyboard and a GUI for adding recipes to the system. The addition of recipes is currently performed through a GUI in the database and help is currently an attached text file.
- Smaller dynamic grammars: To overcome future scaling problems when the database of recipes becomes very large.
- Other options: Including a cooking timer and clock.

Future versions will also allow for a wide range of incremental updates that will provide the product with a longer market life. These features could include:

- Substitution advice: To suggest relevant ingredient substitutions.
- Weekly programs: Such as diet planners and weekly menus / shopping lists.
- Random recipe chooser: To choose a recipe based on your preferences and the time of day.
- Companionship: To include a more sophisticated personality, capable of interacting in simple chitchat.

### 8. Conclusions

The results show that the implementation of speech is definitely beneficial to certain areas. The home cooking assistant is one such area, which has been accepted by end users. Furthermore, a particularly interesting path has been created for future work on this topic. This path is one that many would previously have associated with the realms of imagination, however this paper has hopefully demonstrated the reality with dialog based user interfaces and the potential for a home cooking assistant.

### 9. Acknowledgements

I would like to acknowledge Dr. Raymond Lee, Prof. Hansen Yee, family, friends and all of the testers. These people have always had time for me. They have always been patient and have always kept me on the straight and narrow path. Many thanks for all of the support and effort!

### 10. References