Testing Dialog Design in a Speech Application

by Stephen Keller for UX Magazine
http://uxmag.com/articles/testing-dialog-design-in-a-speech-application

Designing applications that use speech recognition as their primary user experience is a challenge that is compounded somewhat by the difficulty in testing an application that provides a speech interface.

In some ways, a voice user interface (VUI) has all the challenges of other interface design and testing, such as subjective feedback from users ("I don't like this"), along with the additional challenge of having to account for the error rate associated with speech recognition.

Here we'll discuss some of the methods associated with proving out the UX associated with a speech application dialog.

Dialog Design and Testing

A dialog is a common term for the UX of a speech application, because most speech applications take the form of a dialog between human and machine. Even most multi-modal applications such as Apple's Siri still largely model the interactions as a series of questions and answers. Because of this, test plans for speech applications often resemble literal scripts, like you might see for a play or a movie. A simple test script might look something like this:

COMPUTER: Thank you for calling the Acme Bank. What would you like to do today?

HUMAN: Transfer funds.

COMPUTER: Do you want to transfer funds from your savings account or your checking account?

HUMAN: Checking account.

The purpose of the test script, at a basic level, is to verify that simple functionality is working as intended. These scripts are similar to the unit tests used by developers to test specific pieces of code in an application. They are an important part of testing speech applications (if "transfer funds" is not a valid verb in the main menu prompt, that is an important bug to uncover), but they do not really test the design except to prove that the simplest use cases work as intended.

The Importance of a Blank Slate

When we refer to testing dialog design, we're really referring to testing the interface. To do this, the application must be exposed to users who have no scripts or preconceived notions about the interface. Using the play analogy again, every prompt may be understood differently than the author intended, and there are a vast number of responses that can be elicited for even commonly understood prompts.

All of these different lines in the play need to be tested in the real world, and they will all vary in terms of what is prompted, what is understood, and what is wanted based on word choice. All good user experiences should be somewhat intuitive and self-explanatory, with the interface providing the information necessary for novice users to quickly understand how to accomplish their goals. In dialog design, this means providing to the user a clear mental model of the application by having prompts that spell out what the user can do at any menu and how to navigate to other menus.

If a speech application is only exposed to users who follow test scripts, there is no way to verify that the interface meets these requirements. As any interface designer knows, humans are not robots and are often unpredictable. In order to know whether a dialog design provides a good experience, it must be exposed to users who are allowed to react naturally.

These "blank slate" testers should be brought in to test the application as early as possible—often while it is still in an alpha or beta state. Since a speech application's UX design is so tightly woven with its programming, getting feedback about the dialog design early on helps developers make and test any changes to an application's codebase.

Running Usability Tests

Once you've answered these basic questions about design and approach, it is appropriate to start building rough skeletons mapping how the user will interact with the system. One advantage of designing voice user interfaces over graphical interfaces is that VUIs can be modeled very well in flow charts, so this is the preferred approach for initial design.

There are a number of ways to perform usability tests of a speech application's dialog design. The most common ways leverage the following techniques:

- Informal or formal user feedback
- Speech tuning response files
- Listening to recordings of entire sessions or phone calls
- "Wizard of Oz" interactions

User Feedback

The simplest sort of testing is to allow users to experience a dialog and provide feedback. This may be informal (asking users to just write up their experiences) or it may be formal (asking users to fill out a detailed evaluation form). Allowing users to provide direct feedback is always useful with any kind of UX design, as it allows the designer to know what really matters to users and what doesn't.

User feedback is most helpful when it is combined with other types of testing, such as those discussed below. For example, by combining user feedback with full-call recording, it's possible to use free-form feedback ("It didn't seem to understand me much") to pinpoint the exact spot where users had trouble ("I had to repeat myself three times at this particular prompt").

Capturing Response Files

Every commercial speech recognizer on the market allows for the saving of what are called response or utterance files that act as a recording of what a user said and what the recognition result for that utterance was. In conjunction with speech tuning utilities (such as the LumenVox Speech Tuner), designers can listen to all of the interactions with application's users, transcribe the speech, and generate accuracy metrics that indicate how well the application is performing.

During this process, the designer can identify a number of issues. One of the main things that can be ascertained, especially if a lot of data is collected, is which problems are related to the design of the dialog compared to issues with the speech recognizer. For instance, users who are giving appropriate responses to a prompt but are not recognized probably indicate that the recognizer needs to be tuned or the grammars changed. On the other hand, if users are speaking out-of-grammar phrases, then the prompts and the dialog options probably should be reworked so that users have a better idea what is expected of them.

Full Session Recording

Recording and listening to entire phone calls or sessions can be costly in terms of time required, but can reveal issues that may not otherwise present themselves. For instance, if users are getting frustrated and quitting the application, this can be obvious when the entire call is heard in context but may not be clear if only individual interactions are heard.

"Wizard of Oz" Interactions

The most time-intensive form of usability testing involves using "Wizard of Oz" interactions. These involve a person "behind the curtain" listening to live users interacting with the system in real time. The Wizard, who can be thought of as the automated system in the background, listens to the human talk to the system and then inputs the user's choice through another mechanism (usually by putting an entry into a form using a mouse and keyboard on the back-end). In this sense, the human is replacing the speech recognizer and trying to determine the clarity of prompts and the range and wording of responses (this was also the subplot of an episode of Seinfeld where Kramer erroneously provided movie start times by reading a newspaper).

Using the Wizard of Oz model is a good mechanism for testing a dialog design before much programming or grammar design has been done. It allows the intended UX to be evaluated against actual users instead of just testers, since the system will still appear to work correctly and not upset paying customers who may not otherwise appreciate beta testing a new speech recognition system.

Conclusion

The best testing plan for speech applications will combine the methods above or will be a variation of one or more of them. When collecting user feedback on a speech application, it's usually a good idea to capture response files at the same time in order to perform more in-depth speech tuning. Full recordings should be enabled when doing Wizard of Oz testing, and so on. These methods will allow the designer to understand how real-world users interact with a speech system, and provide instructive input for improving and enhancing the quality of the dialog design.

More generally, the same testing methodologies can also be adapted to other types of user interfaces outside of speech recognition. This includes the UX for web transactions, web chat, call center scripting, kiosk interfaces, and other systems where user input may be open ended or require semantic interpretation. The more real world testing that can be performed prior to building a system, the closer the launched product will serve its intended purpose right out of the gate, and the less rework will be required.