Speaking and Listening to Computers

James A. Larson

Recent advances in speech processing technologies and the emergence of the VoiceXML computer language are increasing the adoption of speech applications that assist telephone and mobile phone users to perform useful tasks, such as installing equipment and diagnosing and repairing problems. This article describes the principles for phrasing voice prompts that encourage the user to speak or perform specific actions, for identifying the words and phrase that callers may speak in response to a prompt, and for assisting callers who fail to respond appropriately to a prompt. A programmer will then convert this information into a VoiceXML application by coding each prompt and response using VoiceXML 2.0/2.1.

Customers often need help. By using a telephone or mobile phone supported by speech synthesis and speech recognition technology, customers access product documentation to install, assemble, diagnose, or repair a product—all without being placed "on hold" or talking to a human support agent. This paper presents a methodology for formulating verbal instructions and questions for presentation to the user via speech synthesis and capturing the user's responses and input via speech recognition. This is NOT a tutorial on VoiceXML, an XML-based language for implementing speech application; instead it is a tutorial instructing technical writers how to formulate just-in-time training for customers using verbal dialogs that convey verbal instructions for product installation, assembly, diagnosis, and repair.

Customers may access this information from any touchtone telephone, from any location, anytime of day or night without being placed "on hold." Customers are iteratively instructed to perform tasks, one at a time, and report the success of each task. If the customer encounters trouble, the computer will rephrase its instructions and provide additional help to assist the customer to perform the specific task. If a customer experiences excessive trouble, the customer can be transferred to a human support agent. Most customers complete all tasks without intervention from a human customer support agent.

Writing a sequence of verbal instructions is similar to writing a theater script that specifies a dialog between two characters: the customer and the computerized customer support agent. The technical writer formulates a sequence of questions and/or instructions called *prompts*. The customer responds to each prompt by performing a simple task and then speaking a word or phrase, called a *response*. The collection of possible prompts and responses is called a *dialog flow*. Figure 1 illustrates a partial dialog flow for a collection of medical devices. Each box represents a prompt presented to the user via speech synthesis technology. Each arrow connecting a pair of prompts represents the response spoken by the user and interpreted by speech recognition technology. A path through the dialog flow is called a *dialog* and represents the conversation between a customer and the computer. Each customer response leads to another prompt until the customer achieves the goal of the dialog flow.

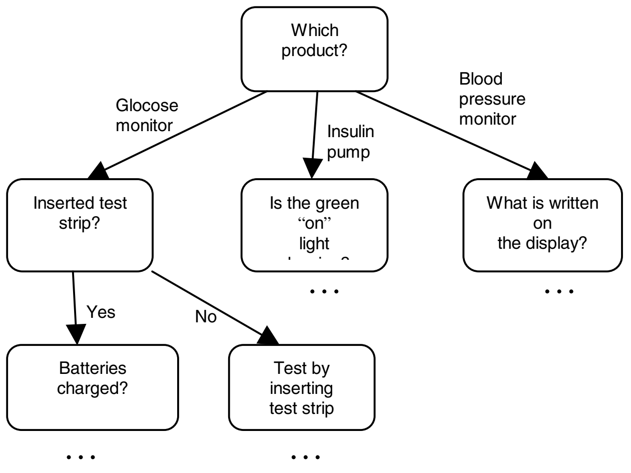

Figure 1. Example dialog flow

When creating a dialog flow, the technical writer resolves two related problems: (1) how to formulate a prompt that encourages the user to perform a specific action and respond to the computerized agent appropriately, and (2) how to identify response words that the customer will likely speak after hearing the prompt. Guidelines for creating dialog flows are found in [1,2]. A programmer will then convert the dialog flow into a VoiceXML application by coding each prompt and response using VoiceXML tags. This process is similar to how an HTML programmer writes HTML tags to create a Web page.

FORMULATING PROMPTS

Formulating prompts is similar to formulating a list of instructions. However, technical writers should formulate each prompt in such a way that the customer responds with only a word or phrase. Guidelines for formulating prompts include:

- Keep prompts short and direct. Customers appreciate short, direct questions. Avoid: “To whom do you wish to speak” Use: “Who do you want to speak to?”

- Use conversational English rather the proper English. The prompts are part of a conversation, not a formal paper to be published. Avoid: “To where do you want to go?” Use: “Where do you want to go to?”

- Use active voice. Active voice encourages the customer to perform the action and then respond. Avoid: “Your account number is requested.” Use: “Please enter your account number.”

- Use second person. Using second person makes it clear that the user should perform some task and then respond. Avoid: “We need the account number which is located on the upper left-hand corner of the statement.” Use: “Say your account number; it is located on the upper left-hand corner of your statement.”

- Avoid subjunctive mood. Avoid: “Should the city name be incorrect, you may backup by saying ‘cancel.’” Use: “If the city name is wrong, say ‘cancel.’”

- Avoid compound questions. Customers often have trouble responding to compound questions with a single a word or phrase. Avoid: “Do you want to cancel or change your order?” Use: “To process your order, say ‘confirm,’ ‘change,’ or ‘cancel.’”

- Reserve “enter” and “press” for encouraging the customer to press the buttons on the phone, and use “speak” and “say” for verbal responses. Use: “Please enter you account number.” Use: “Please say your name.”

- Use simple, closed questions. If the user doesn’t already know the possible answers, enumerate the answers. Use: “What is your destination city? Abion, Burnville, Cocoa Beach, or Davidson Cove?” when the user does not know the options. Use “Which month?” because most users know the names of the months.

FORMULATING RESPONSES

The computerized help agent will use speech recognition to determine if the customer has performed a specific task and responded appropriately. In order to avoid processing delays and misrecognitions, pay careful attention to the choice of possible responses for each prompt. Guidelines for formulating responses include:

- Use words that customers are likely to speak. This avoids forcing the customer to memorize the set of possible responses for each prompt.

- Keep the number of possible responses as small as possible. The speech recognition system is faster and more accurate if the number of possible responses is small.

- Avoid similar sounding responses, such as “delete” and “repeat” or “backup” and hangup.” The speech recognition system may confuse words that sound similar.

- Avoid noisy words. Speech recognition systems often have trouble with words containing “f,” “s,” and “z” sounds.

- Avoid low energy words. Speech recognition systems often have trouble deciphering words containing “m” and “n.”

- Align the response with its prompts.

  Avoid:
  Prompt: “What is your destination station? Abion, Burnville, Cocoa Beach, or Davidson Cove?”
  Responses: Abion, Burnville, Cocoa, Davidson

  Use:
  Prompt: “What is your destination station? Abion, Burnville, Cocoa Beach, or Davidson Cove?”
  Responses: Abion, Burnville, Cocoa Beach, Davidson Cove

FORMULATING FALLBACK PROMPTS

Customers sometimes fail to respond by speaking one of the specified response words or phrases. The speech recognition engine classifies these failures in one of three ways:

A *help event* occurs when the customer may ask for help.

A *nomatch event* occurs when the customer speaks a word or phrase that is not a specified response word.
A *noinput event* occurs when the customer fails to speak anything at all.

Technical writers often write fallback prompts to be presented to the user when these events occur. *Fallback prompts* encourage the user to speak again. Here are some example fallback prompts:

Example 1:

> Computer: What is your departure station? Abion, Burnville, Cocoa Beach, or Davidson Cove?
> Customer: Help.
> Computer (fallback prompt): In order determine what price you pay, we need to know where you are traveling from. What station are you departing from? It is Abion, Burnville, Cocoa Beach, or Davidson Cove?

Example 2:

> Computer: What is your departure station? Abion, Burnville, Cocoa Beach, or Davidson Cove?
> Customer: (no response)
> Computer (fallback prompt): I'm sorry, I did not hear you. Please say the name of your departure station. It is Abion, Burnville, Cocoa Beach, or Davidson Cove?

Example 3:

> Computer: What is your departure station? Abion, Burnville, Cocoa Beach, or Davidson Cove?
> Customer: Cogville
> Computer (fallback prompt): I'm sorry, I did not understand you. Please say the name of your departure station again. It is Abion, Burnville, Cocoa Beach, or Davidson Cove?

Guidelines for fallback prompts include:

- Never place blame on the user. Users do not like to be told that they are at fault. Avoid: "That is incorrect. Try again. What is your destination station?" Use: "I'm sorry, I didn't understand you. What is your destination station?"

- Provide additional information and encouragement in successive fallback prompts.

- Prompt: "What is your departure station? Abion, Burnville, Cocoa Beach, or Davidson Cove?"

- First fallback prompt: "I'm sorry, I did not hear you. Please say the name of your departure station. Is it Abion, Burnville, Cocoa Beach, or Davidson Cove?"

- Second fallback prompt: "I'm sorry. I still didn't hear you. Please say the name of your departure station. The station name is shown at the top of the station kiosk."

- Do frustrate users. After 4 or 5 failed attempts, transfer the customer to a live customer support agent.

- Avoid customer hyperarticulation. *Hyperarticulation* occurs when the customer begins to speak loudly, slowly, and over emphases each syllable. Rephrase the question so the caller responds using different words. Replace the question with a series of questions, or include acceptable responses as part of the fallback prompt.

USING TOUCHTONES AS RESPONSES

Dual **T**one **M**odulated **F**requency (DTMF), often referred to a touchtones, are available on most telephones and all mobile phones. (The old rotary telephones cannot produce touchtones, but are rarely used today.) Touchtones are seldom misrecognized. However, customers love to hate user interfaces consisting only of DTMF for three main reasons:

1. Customers get lost in long sequences and hierarchies of prompts.
2. It is impossible to "undo" or backup; customers must hangup and redial.
3. Many customers find dialogs using only DTMF to be time-consuming and tedious.

However, customers will use DTMF for special situations:

- *Security*—account numbers, passcodes, PINs, and other key information which should not be spoken for others to hear
- *Privacy*—customer weight, birthdate, and other information which the customer may not want others to hear.
- *Backup*—when the speech recognition fails, especially in noisy environments and for users with accents
- *Accessibility* for customers who are hearing impaired. In this case, a reasonable fallback prompt may encourage callers to use DTMF rather than speak.

USABILITY METRICS

What to test for?

Each application should have a single specific goal. We perform tests to determine how well the application helps customers to achieve that goal.

Types of tests include:

- *Stress tests*—determine how the application works under extreme conditions such as an assembly instruction application on Christmas Eve.

- *Function tests*—determine if functions are implemented correctly such as is the product repaired, and does it work satisfactorily after repair?

- *Usability tests*—determine if callers enjoy using the application and/or are able to use the application effectively to achieve specific tasks.

While all of these tests are important, this paper concentrates on usability testing because the technical writer should specify much of the usability testing criteria.

Why is usability testing important?

Usability testing is important to:

- *Determine when the application is ready for customers*—deploying a verbal customer support system, before it is ready, is as bad as publishing documentation that is not completely correct.

- *Measure improvement*—developers use usability tests as yardsticks to measure how well the application performs. Testing not only indicates positive or negative improvement, but also

  quantifies the amount of improvement. The developer optimizes the application to meet or surpass the usability criteria. These tests determine when iterative testing and fine-tuning is complete.

- *Enable comparison*—similar applications can be compared using the same usability test. For example, if Application A callers consistently perform more efficiently than Application B callers, then Application A can be said to be "better" than Application B.

- If application designers have not defined metrics, then technical writers should define the metrics to measure the degree to which the application succeeds in helping callers achieve their goals. A group of Voice User Interface (VUI) designers attended a workshop at SpeechTEK 2005 and identified ten usability metrics for measuring effective voice user interfaces [3]. These metrics fall into two general categories—preference and performance.

Preference metrics measure callers' likes and dislikes.

Preference metrics are subjective and measure callers' reactions after using the application. Example preference metrics include:

1. *Caller satisfaction*—the degree to which the voice user interface meets callers' expectations

2. *Ease of use*—callers' perceptions of using the application

3. *Quality of output*—callers' subjective ratings of voice intelligibility or voice quality

4. *Perceived first-call resolution rate*—perceived successful completion rate on the first call, including both voice user interface and possible interaction with a human agent

5. The above preference tests are generic and apply to almost every speech application. Technical writers should refine the metrics to

be more specific and relevant to the goal of the application. For example, if the application consists of instructions to repair a product, preference metric 4 might be restated as "Measure the caller's success or failure to repair the product."

6. Preference testing collects preference scores from callers after they test the system. After testing the application, testing specialists conduct interviews with callers, who score the various preference criteria. Or callers may be asked to enter preference scores onto a paper questionnaire, a Web page, or a verbal VoiceXML form.

Performance metrics measure how well the application performs for callers.

Performance metrics are objective. They measure how well the application performs. Several callers test the application with aggregate scores calculated for each metric. The aggregate scores reflect the overall measure of the application performance. Example preference metrics include:

1. *Time-to-task*—the time from answering the call to the time the caller begins performing the desired task

2. *Task rate*—the percentage of calls that trigger a specific task start-point or end-point

3. *Task completion time*—the time to complete a specific task

4. *Correct transfers*—the number of calls successfully transferred to the correct party

5. *Abandonment rate*—the percentage of callers who hang up before carrying out a specific task

6. *Containment rate*—the percentage of calls not transferred to human agents

The above preference tests are generic and apply to almost every speech application. Technical writers and developers should refine the questions to be more specific and relevant to the goal of the application. For example, if the application is a sequence of instructions for repairing a product, time-to-task (metric 1) might be restated as "Measure the amount of time from the initiation of the repair activity to the repair completion." Performance metrics are measured while a customer uses the application.

Both preference and performance tests are important. Preference tests indicate how well callers enjoyed using the application, while performance tests indicate how effective the application helps the caller perform tasks.

OTHER ACTIVITIES

Personas. Many voice application experts believe that technical writers should specify a *persona*—the voice equivalent of a Web site's "look and feel." A persona reflects the overall personality and style of the voice agent. Many company applications will have the same persona, which identified as part of the company "brand." Personas usually have an age, gender, educational background, and personality. Technical writers need to keep the persona in mind when formulating the wording for the prompts.

"Wizard of Oz" testing. This technique is frequently used to debug a dialog flow before it is coded. The technical writer and a perspective customer talk with each other using the telephone. The technical writer pretends to be the computer, and the customer responds to prompts uttered by the technical writer. This is an excellent way to detect flaws in the dialog flow, as well as improve prompt wording and acceptable responses.

Implementation. A VoiceXML programmer converts the dialog flow into VoiceXML code, debugs the code, and prepares it for testing.

Audio prompt recording. Some customers have difficulty in understanding synthesized voices. Many customers tire of listening to synthesized voices after a few minutes. Many developers replace voice synthesizers with recordings of voice actors speaking the prompts. Recording sessions may involve a voice actor and a voice coach who directs the actor to speak each prompt with the appropriate inflections and pronunciations as they might be spoken by the persona.

BEYOND SPEECH APPLICATIONS

Speech applications have no visual mode. Illustrations and pictures cannot be presented to the user. A new kind of application, called *multimodal*, enables customers to see text, illustrations, and pictures on the screen of a telephone or mobile phone while the customers interact by speaking and listening with an artificial customer support agent. Presenting customers with schematics, illustrations, pictures, and even short video clips will certainly enhance customer support applications. Unfortunately, the necessary advanced communication networks and software and hardware standards are not yet in place.

REFERENCES

[1] Balentine, Bruce and David P. Morgan. How to Build a Speech Recognition Application: A Style Guide for Telephony Dialogues, 2nd ed. San Ramon, CA: EIG Press, 2001.

[2] Cohen, Michael H., James P. Giangola, and Jennifer Balogh. Voice User Interface Design. Boston: Addison-Wesley, 2004.

[3] Larson, James A., et al. Ten criteria for measuring effective voice user interfaces. Speech Technology Magazine, November–December 2005. See also http://www.speechtechmag.com/issues/10_6/cover/12612-1.html.

James A. Larson
Research Scientists
Intel Corporation
16055 W. W. Walker rd., #402
Beaverton, OR 97006 USA
503-645-3598

James A. Larson is manager of advanced human input/output at Intel Corporation and is author of the home study guide and reference The VXMLGuide http://www.vxmlguide.com/. His Web site is http://www.larson-tech.com/..