Multimodal Application Design Issues

IBM White Paper

December 2003

Contents

© Copyright International Business Machines Corporation 2003. All Rights Reserved.
Note to U.S. Government Users Restricted Rights — Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Introduction

A multimodal application provides end-users with a flexible interface that allows interaction with the device using a combination of keyboard, touch screen, stylus, telephone keys, and voice. The most important characteristics of a multimodal application are the use of voice and its integration with the visual representation. Due to these special characteristics, you need to make special considerations when designing a multimodal application. This paper will discuss some important design issues.

Interface Design Issues

An important phase in designing a multimodal application is to design a speech user interface. This section discusses key speech-interface design issues, including voice, visual + voice, dialog, and synchronization considerations.

For more information, please refer to *VoiceXML Programmer's Guide* included with IBM® WebSphere® Voice Server SDK V3.1.1 and Voice Toolkit V4.1.5 for WebSphere Studio.

Voice Considerations

Synthesized speech and recorded prompts are two methods for playing prompts in your application. Synthesized speech (text-to-speech or TTS) is easier to maintain and modify. For this reason, TTS is typically used during application development. It is also useful as a placeholder when the data to be spoken is "unbounded" (not known in advance), which makes it impossible to prerecord. When deploying your applications, however, you should use professionally recorded prompts whenever possible because users expect commercial systems to use high quality recorded speech with natural pronunciation.

In designing the prompts in your speech interface, you also need to decide whether you will use terse or personal prompts. Each has its advantages and disadvantages. Terse prompting style uses the fewest words possible so that it makes efficient use of time, and it often leads users to respond succinctly, producing spoken responses that are easy to recognize. Its main disadvantage is that if the prompt is too terse, users might misinterpret its meaning. On the other hand, personal prompting style is perceived as more human, but it can lead users to produce responses that are beyond the actual recognition abilities of the system.

Part of the challenge of designing a good speech interface is that you don't want to force users to hear more than they need to hear, and you don't want to require them to say more than they need to say. Adhering to the following guidelines can help you achieve this:

- When designing menus, limit the number of menu items based on how much information users must remember.
- Group information by separating introductory/instructive text from prompt text, separating text for each menu item, ordering menu items and cycling through menu items.
- Choose the right words for your application's prompts and menus.
- For dialogs that users might revisit in a single session, you may want to create multiple, progressively shorter prompts. After the initial pass through the dialog, you can then play the tapered prompts on subsequent passes.
- Use user input confirmations judiciously, and before providing lengthy confirmation feedback, you may want to ask users if they want to hear this confirmation.

Visual + Voice Considerations

After talking about some of the issues involved in designing the voice part of the application, it is time to discuss issues related to both visual and voice that are specific to the multimodal environment.

Multimodal applications consist of both voice and visual elements. There is not necessarily a one-to-one mapping between them. Some information is better presented using voice, other information is better presented using visual, and you may combine both formats in most cases.

In general, welcome and introductory information can be well presented using voice to catch the user's attention as soon as the application starts. The elements that contain brief information, such as short instructions and e-mail subjects, are also good for voice. In an e-mail example, if the application can read the e-mail subject lines out loud, users can individually choose to read or listen to a particular e-mail rather than going into each one.

Most visual elements that require user input or action can be enabled using voice. The following are some common examples:

- Text field – When users put focus in a text field, a prompt can be played, and the user's voice response can be displayed in the text field. This is especially useful and convenient when the user does not have access to the keyboard.
- List box – Users can select items in lists using voice. It is useful when the mouse is not available to the user, and it is especially convenient when the user is familiar with the content of the list. For example, in a list box of US states, most users know the items in the list and know exactly which one they need to select, so the user can just say the state's name, for example, "Florida," to select the item without the need to open the drop down list.
- Checkbox – If a group of checkboxes is enabled, users can select one or more selections using voice input instead of making multiple clicks.
- Link – Users can link to another page or section in the application using voice. In this case, you should use a short name for the link and provide a brief description next to it.
- Button – Users can invoke a button click action using voice. For example, they can say "Submit" instead of clicking or pushing a Submit button.

Some information is not easy to present using voice, such as graphics, diagrams and tables. These are better presented in visual format. If you want to add voice to these visual elements, you need to give special consideration to the wording of the speech by emphasizing the key information it depicts. For example, if you want to present a pie chart using both visual and voice, the application may say, for example, "The biggest segment is … and the smallest segment is …" when the chart appears.

Since a multimodal application has both visual and voice interfaces, it is very important to keep a consistent 'Sound, Look, and Feel' for the application. To promote the consistency, you may want to adopt the following guidelines:

- Adhere to accepted standards for visual interface design, such as using the same visual presentation style for all the pages in your application.
- Use a consistent strategy for determining which information to present verbally, which to present visually, and which to present using both interfaces.
- Try to use the same terminology in both interfaces whenever possible. For example, don't have the speech interface prompt the user to "Say Yes or No," while the visual interface displays buttons labeled "Okay" and "Cancel."
- Choose one prompt style (refer to the previous discussion in "Voice Considerations") and use it consistently throughout your application.
- Try to use the present tense and active voice for all prompts. For example, use "Transfer how much?" rather than "Amount to be transferred?"

- Whenever possible, use a parallel structure (all nouns, all gerunds, etc.) in lists. For example, use "Say one of the following: List current address, Change address" rather than "Say one of the following: List current address, Address change".
- Maintain consistency in acceptable user input. For example, if you allow affirmative responses of 'Yes,' 'Okay,' and 'True' to one yes or no question, you should allow the same responses for all yes or no questions in your application. You can accomplish this by using the built-in types or reusing your own grammars.
- Use pauses consistently. For example, you may choose to interpret 5 seconds of user silence as an indication that the user needs help. If so, use this silence timeout interval consistently throughout the application.

Dialog Considerations

Dialog design is critical in the process of designing a multimodal application. When users don't know what they can say at a given point in a dialog, the interaction between the user and the application can quickly break down. To help users avoid this "What can I say now?" dilemma, try adopting consistent sound, look, and feel standards to create dialogs that behave consistently within and between your applications.

- Try to write voice prompts that clearly indicate to users what they can say. Examples of such prompts are: "Say either Yes or No," "Send mail to which person?" or "Please say one of the following: E-mail, Voice mail, Faxes."
- When an application behaves in a way that the user didn't anticipate, the user becomes confused and is more likely to respond inappropriately to subsequent prompts, causing the interaction between the user and the application to break down. To help users avoid this "But why did the application say or do that?" dilemma, try using consistent sound, look, and feel standards described earlier to create dialogs that maintain synchronization between the application's actual state and the user's thought of the application's behavior.
- You can improve application consistency and decrease user learning curves by reusing dialog components (such as sub dialogs for data collection, error recovery, and other common tasks, VoiceXML's built-in field types and application-specific grammars) whenever possible.
- In the visual interface, try to position equivalent objects in similar on-screen locations within your applications. For example, if multiple dialogs have buttons labeled "Okay" and "Cancel," be consistent as to which is displayed on the left and which is on the right.

Synchronization Considerations

Synchronization is an issue specific to the multimodal environment. Since the multimodal application is presenting information to the user using both visual and voice interfaces, the two interfaces should always be synchronized.

When you design your application, you must take into account that dialog transitions involve both interfaces. For example, while the voice browser is processing a dialog in page one, the visual browser should not be directed to page two unless the voice browser is also making a transition to that page. If the user forces the transition manually, the voice interface should adjust accordingly by either stopping the process for page one and starting the process for page two or by playing an error message. The visual interface should display the page to match the voice response as well.

Besides synchronizing dialog transitions, you should also pay attention to the transitions between elements. For example, if the voice control moves from a text field to a list box, you should make sure that the visual focus is also moved from the text field to the list box. Conversely, if the visual focus initiates the transition, the voice should respond accordingly as well.

You also need to avoid long paragraphs of information at one time because users may easily lose their attention while listening and reading. You should make paragraphs as brief as possible if you want to

display the information in both formats. If you have to present the information using a long paragraph, it is better to display it in visual form only so that users can choose to read it at their own speed.

Application Design Issues

Since a multimodal application is also a web application, the general web application design issues also apply to the multimodal application. Like any other web application, you have the choice of creating a static or dynamic multimodal application, and you need to consider how to manage the resources of the application efficiently. This section discusses these two design issues.

Static vs. Dynamic Considerations

The purpose of the multimodal application determines whether to make the application static or dynamic. If the purpose of the application is to display the fixed information and there is no user interaction needed, then a static multimodal application is sufficient. On the other hand, if the information to display changes frequently (such as flight information) or user interaction is involved (such as submitting a form), then you need to write a dynamic multimodal application. However, in most cases, you are more likely to create an application that has both static and dynamic pages depending on the functions each page provides. You can even have a multi-frame page that contains both static and dynamic content.

One issue that is specific to the multimodal application is whether to use the static grammar or to generate the grammar dynamically. Although the dynamically generated grammar adds flexibility to the multimodal application, there is a price to pay. Every time a grammar changes, it needs to be recompiled, which is a relatively slow process. The larger the grammar, the longer it will take to compile. If you use a web browser that has the functionality to cache the precompiled grammar, you need to think and plan very carefully before you use the dynamically generated grammar. Keep in mind that the grammar compilation is an expensive operation and try to use precompiled static grammars as much as you can. If you have to dynamically generate some of the grammars, try to minimize the negative impact on the application caused by the grammar recompilation. The easiest way to achieve this is to reduce the size of the grammar. If this is not an option, then follow some grammar design guidelines to optimize performance. For example, avoid having too many branches at each node of a grammar and avoid using recursive rules.

Resource Management Considerations

Since multimodal applications do not require the availability of a keyboard and mouse, they are more commonly used in small devices. When you design a multimodal application for small devices, you need to pay special attention to resource management. Due to storage limitation of small devices, there is also a limitation on the resources you use for the multimodal application. If you need to install the application on the device locally, avoid using large images and large grammars due to the limited disk space available. By placing the application on the server, it solves the storage limitation problem of the small device and gives you access to more resources, but it will make the application run slower due to the added time for resource fetching. The use of precompiled grammars is especially helpful in the case of remote access because it can reduce the application processing time to compensate for the fetching delay.

Conclusion

The design of a multimodal application is a broad topic. At the same time, it is an art. What we have discussed does not cover all the aspects involved in the topic and it does not give you clear-cut rules to follow, but it certainly provides you some general guidelines and tips that you should consider as you design your multimodal applications. Just like any other learning process, practice is still the key to success.

Notices and Trademarks

This publication was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1784
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
TL3B/B503
3039 Cornwallis Road Rd.
Research Triangle Park, NC 27709

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows: © (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks or registered trademarks of the International Business Machines Corporation in the United States, other countries, or both:

IBM

WebSphere

Intel and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.