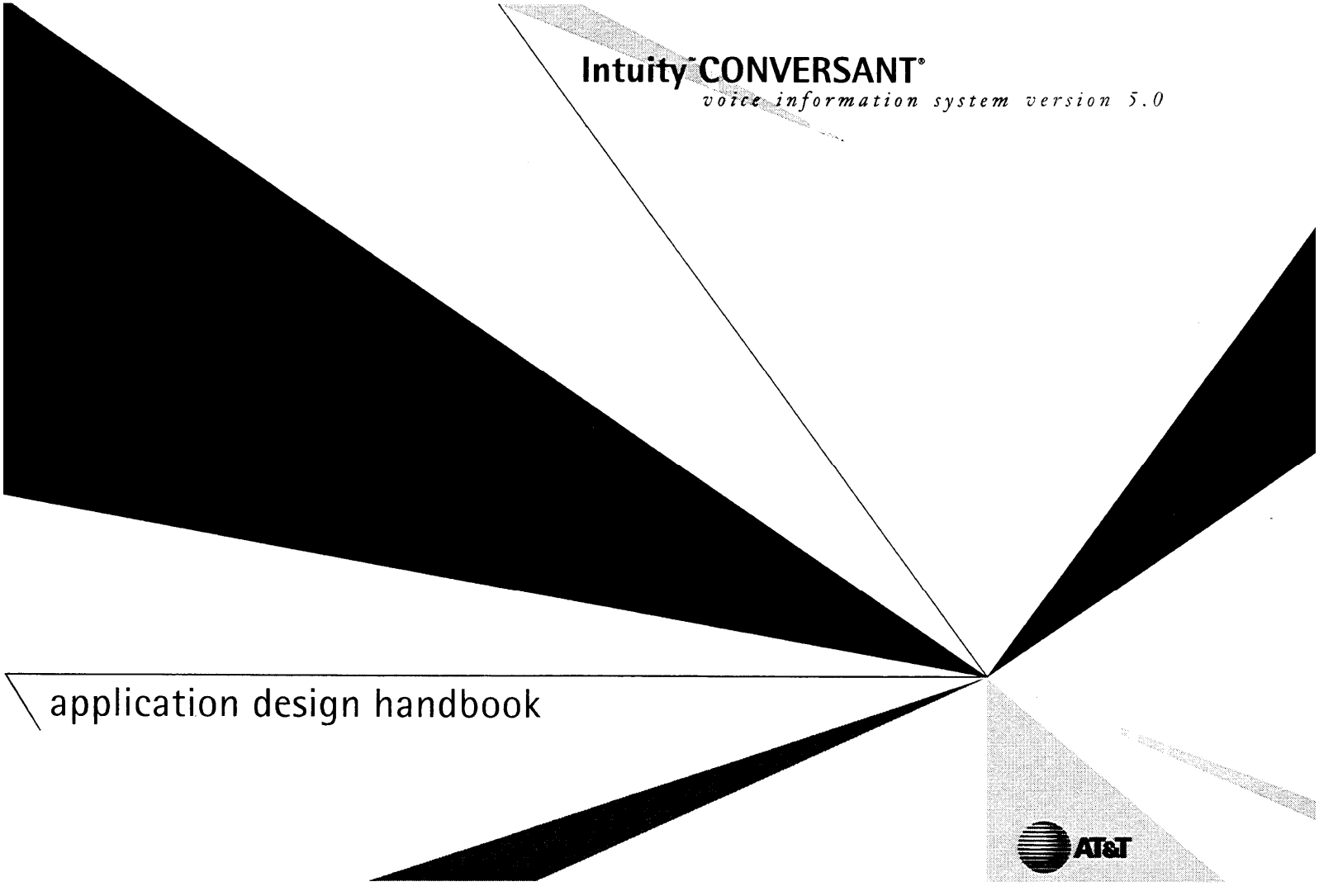

Intuity™ CONVERSANT®
voice information system version 5.0
application design handbook
AT&T

Notice

While reasonable efforts have been made to ensure that the information in this document was complete and accurate at the time of printing, AT&T can assume no responsibility for any errors. Changes and corrections to the information contained in this document may be incorporated into future reissues.

Ordering Information

The ordering number for this document is 585-310-551. To order this document, call the GBCS Publications Fulfillment Center at 1-800-457-1235 (international callers use 1-317-361-5353). For more information about AT&T documents, refer to the *Global Business Cornmunications Systems Publications Catalog,* 555-000-010.

You can be placed on a Standing Order list for this and other GBCS documents you may need. Standing Orders will enable you to automatically receive update versions of individual documents or document sets, billed to account information that you provide. For more information on Standing Orders, or to be put on a list to receive future issues of this document, please contact the AT&T GBCS Publications Fulfillment Center.

Trademarks

AT&T Fax Attendant System is a registered trademark of AT&T.
CONVERSANT is a registered trademark of AT&T.
FlexWord is a trademark of AT&T.
Intuity is a trademark of AT&T.
ORACLE is a registered trademark of Oracle Corporation.
UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/OPEN Company, Ltd.

We'd like your opinion.

Your feedback will help us refine and improve the Intuity™ CONVERSANT® VIS Version 5.0 Application Design Handbook. Please complete this form at your earliest convenience.

Please rate the effectiveness of this document in the following areas:

please **3** *one of the following*	Poor	Fair	Good	Excellent
Completeness	▼	▼	▼	▼
Clarity	▼	▼	▼	▼
Ease of finding information	▼	▼	▼	▼
Relevance to your tasks	▼	▼	▼	▼
Organization	▼	▼	▼	▼
Accuracy	▼	▼	▼	▼
Appearance	▼	▼	▼	▼

Overall satisfaction What do you like most about the handbook? Why?

please in shaded areas

most

What do you like least about the handbook? Why?

least

Please write any other comments here

Name

Position/Title Company

Address Phone

City State Zip

When you have completed this form, please return to address on back. Thank you!

preface

The Intuity™ CONVERSANT® Voice Information System (VIS) is a powerful tool. It gives you the capability to design and implement voice response applications that enhance the success of you business. With this power, though, comes the responsibility to design applications that effectively serve your customers.

This handbook contains guidelines and suggestions for good application design. It also outlines a process that will help you progress from design to implementation to deployment. The recommendations in the handbook may or may not be applicable to your situation and applications. Voice responce has aspects of both a **"science"** (the technology itself) and an **"art"** (the design of the caller/application interface). Although the handbook will help you better understand and employ the scientific aspects of voice response, its main purpose is to help you better understand and practice the artistic aspects. Special attention is focused upon advanced speech recognition technologies, since a number of our customers have asked for more guidance in dealing with issues such as increasing recognition accuracy.

This handbook concentrates on helping you plan what you want to do with your voice response applications, rather than describing how to do it. Refer to the Intuity CONVERSANT VIS customer documentation for implementation details.

The information is divided into six sections:

Section 1 : **introduction and overview**
provides background information to prepare you for the rest of the handbook, including a list of important terms.

Section 2 : **advanced technologies-what do they offer?**
gives an outline of the Intuity CONVERSANT VIS technology, what it is designed to do, and applications it can support.

Section 3 : **preparing to design a voice response transaction**
explains basic usability principles, the building blocks of good script design.

Section 4 : **understanding speech recognition**
compares and contrasts touch-tone input and spoken input, helps you understand the concept of speech recognition accuracy, and how it is related to the success of a script.

Section 5 : **designing the transaction**
makes up the bulk of this handbook. This section gives guidelines and suggestions for designing usable scripts, building on the information in the preceding sections; it includes many examples.

Section 6 : **deploying your script**
describes how to prototype and test scripts, set usability objectives, and make modifications based on test data.

The index at the end of the handbook should help you quickly locate particular topics.

The following conventions are used throughout the handbook:

- VIS spoken output is shown *"italicized and in quotation marks."*
- Caller input is shown < **in brackets and bold** >.
- Items defined in the Important Terms list are shown in italics the first time they are mentioned in the text.
- The footnotes for each section are located at the end of that section.

Before we start: it's helpful for us to know how useful you've found this handbook to be. We'd like your input on how to make it better in the future. After using the handbook, please detach the postage-paid form, complete the questions, and mail it in. Thanks in advance for your suggestions.

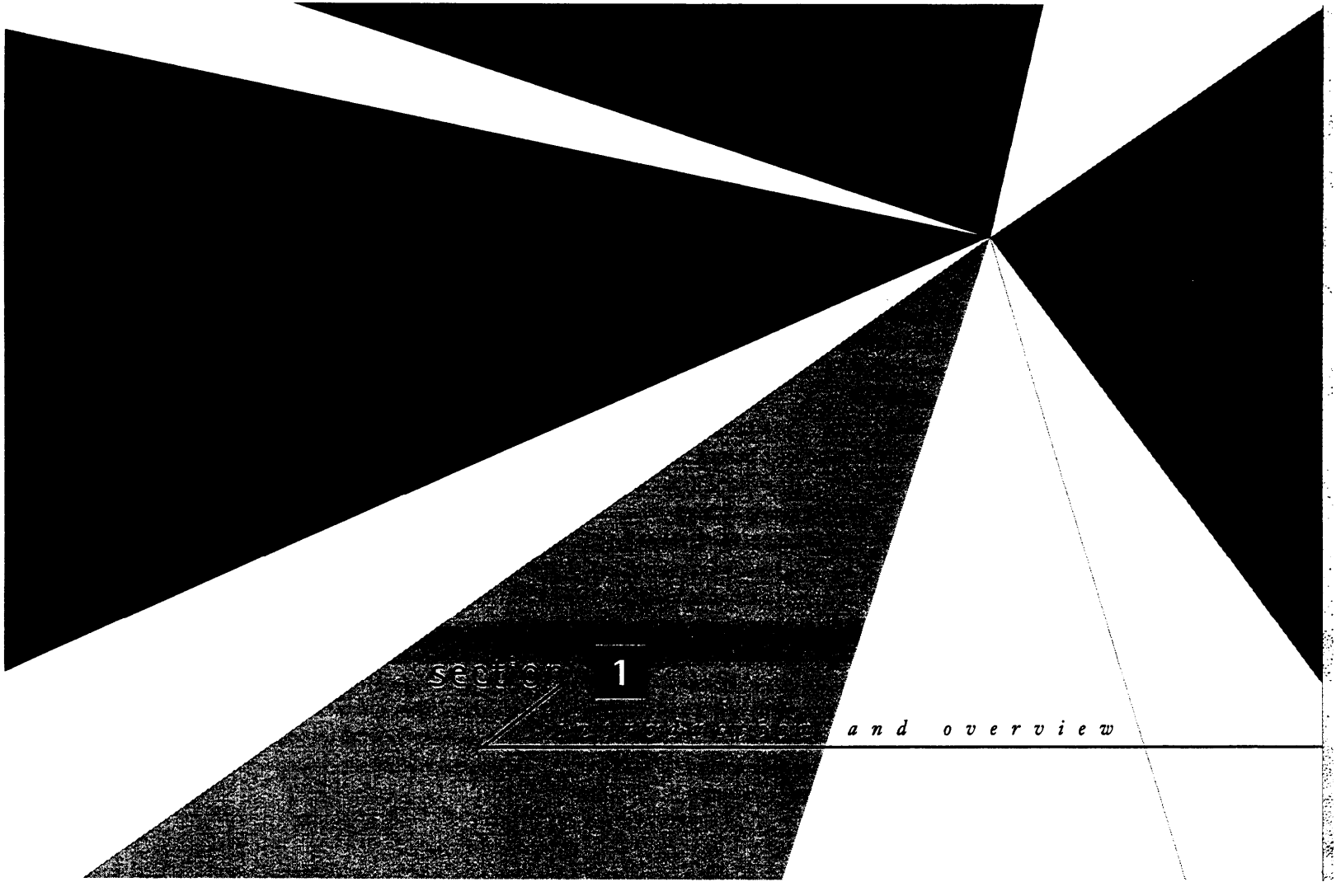
What's New?

The first edition of this handbook gave guidance in designing scripts to run on Version 4.0 of the product. This updated handbook reflects the changes introduced in Version 5.0 that can have an impact on the way you design application scripts. These changes include:

- FlexWord™ Speech Recognition now supports word spotting and phrase screening [rejection], so it works more like WholeWord Speech Recognition.
- You can now build your own vocabularies with the optional FlexWord Vocabulary Management Package.
- FlexWord Speech Recognition now supports up to 200 wordlists [increased from 100]. Wordlists may now contain up to 500 items [increased from 200].

For information on changes to areas such as script development, system architecture, host connectivity, maintenance, and platform, please refer to the Version 5.0 customer documentation.

contents


section 1
introduction and overview

Why Voice Response?

Voice response is an exciting way to combine the power of the telephone and the flexibility of the computer to enhance the success of your business. It can save you money and time, while giving your customers quick and easy, 24-hour access to the services and information they want and need. Voice response can help increase employee productivity, freeing them from answering repetitive customer inquiries. It can also decrease your need for costly customer mailings, while keeping your customers better informed than ever before.

A good application script is key to your success.

Although voice response is high-tech, its success depends heavily on the quality of the application using it. Merely purchasing a voice response system does not ensure that you and your customers will get the most out of it. You have a responsibility to ensure that callers are not confused or frustrated while interacting with the voice response system. You owe it to your customers and yourself to offer highly usable, even delightful, voice response interactions.

Designing a voice response application seems simple: just design the voice response transaction to mimic a transaction between two people. Unfortunately, this does not work for all transactions. The voice response system has capabilities and limitations that are different from those of people. When designing an application script, you must be aware of what the voice response system can and can't do, as well as what people can and can't do.

That is where this handbook can help. It contains the information you'll need to design easy-to-use voice response applications. It tells you what the voice response system can do well and not so well, and also tells you a bit about human capabilities. When designing an interaction between a person and a voice response system [or any computer], it is essential to know as much as you can about both the person and the computer. This handbook will give you information that you can apply directly to designing usable applications.

You may want some of your voice response applications to take advantage of advanced speech technology features, such as speech recognition and text-to-speech. *Speech recognition* allows callers who do not have *touch-tone* telephones [or cannot or will not use them] to interact with the VIS. It also can provide a more natural caller interface. Intuity CONVERSANT VIS offers two types of speech recognition: *WholeWord* speech recognition recognizes callers speaking digits and the words "yes" and "no."' *FlexWord* speech recognition allows you to define sets of *words* for your application to recognize. You can allow callers to speak the name of the function or item they want, which can be easier and more intuitive than remembering a numbered option. Another advanced speech technology, called *Text-to-Speech,* gives you the ability to speak long or variable text to your callers without the expense and effort of having a professional speaker pre-record the information. This handbook aims to give you the information you'll need to get the highest possible speech recognition accuracy, as well as offering techniques for optimal use of text-to-speech.

Learn to use advanced speech technologies to their fullest by using this handbook.

You also might want some of your voice response applications to send information to callers over a fax machine. Integrated *Fax* capabilities allow you to build applications that can send and / or receive fax messages. This handbook will also give you the information you need to use the fax capabilities to their fullest, allowing your callers to easily and quickly request the information they need.

This handbook will help you through the essential steps of *script* design Designing a voice response application requires going through several drafts, just like a written document. You cannot expect to come up with the best and most usable design the first time through. Even design professionals must go though an “iterative” design process: they test their application design first with test callers, then with actual callers, and make changes to the design based on the results of the tests.

Eliminating Caller Frustrations

Some callers may be confused when trying to use voice response services. Callers do not want to have to use applications with poorly-designed, confusing *user interfaces.* Callers who have been frustrated in the past would gladly follow the spoken instructions if they knew they could easily and quickly complete their intended task each time they encountered a voice response system. It should not be surprising that some callers become upset at having to endure a confusing interaction over the telephone.

Because of experiences with poorly designed user interfaces, many callers are wary of all voice response systems. Callers are not quick to forget being confused or frustrated by a voice response system, and they can direct those feelings at any voice response system with which they come in contact. In fact, some callers with touch-tone telephones do not use them when requested; instead, they opt to wait for an attendant to come on the line. In some voice response systems, this is the most efficient way to be served! Don’t let your application alienate customers.

Your goal is to make the automated transaction at least as, if not more, attractive and efficient as interacting with an attendant.

Important Terms

Here are some terms that will be used in this handbook. You might want to become familiar with them before reading on.

General Terms

Announcement	Speech played by the VIS to the caller which informs, but does not instruct the caller to act. Compare to “Prompt.”
Application Script	The computer program that defines and controls the voice response transaction between the VIS and the caller. Also referred to as ’’application,” “script,” or “transaction.”
Caller	The person who calls for a service, gets connected to the VIS, and interacts with an application script.
Fax	The optional VIS capability that allows a script to send pre-stored fax messages at the caller’s request [fax-on-demand], or receive incoming faxes from the caller [fax messaging].
Graphical Speech Editor Package	An optional package that provides a point and click, graphical way for you to record and edit prompts and announcements for your application scripts.

Menu	A prompt that gives callers a choice of two or more options. For example: *"For sales, press 1. For service, press 2. For an attendant, press 0."*
Prompt & Collect	The Script Builder action that is used to play a prompt to the caller, accept the caller's touch-tone or spoken response, and go to the appropriate place in the script to handle the caller's request.
Prompt	Speech played by the VIS that instructs the caller to enter information. Part of a Prompt & Collect action. Compare to "Announcement."
Script	See *"Application Script."*
Script Builder	An optional software package that allows you to define and generate voice response scripts to run on the Intuity CONVERSANT VIS.
Speech, Custom	The part of the VIS speech database that includes script-dependent, pre-recorded speech phrases.
Speech, Pre-Recorded	A prompt or announcement that has been recorded by a person. Can be either custom or standard speech.

Speech, Standard The part of the VIS speech database that includes pre-recorded speech phrases corresponding to numbers, ordinal numbers, days of the week, and months of the year. These phrases are used by scripts to speak numbers, monetary amounts, dates, and times in a natural-sounding way.

Text-to-Speech A VIS software package that converts printed text into spoken, computer-generated prompts and announcements.

Transaction The act of a caller interacting with an application script. In a typical transaction, the caller dials in to or gets transferred to the VIS, then the VIS answers and plays a greeting. The caller enters information in response to spoken prompts and the VIS speaks information back until the interaction incomplete. *Also see "Application Script."*

Usability The system quality that reflects whether or not callers can learn and use the features successfully. A system with high usability is called "easy to use," or "usable."

User Interface The aspect of the script with which callers interact; includes prompts and announcements from the VIS to which callers respond using touch tone or speech.

Voice Response A service that allows callers to conduct transactions with a computer over the telephone, with no human intervention.

Touch-Tone Terms

Dial Ahead — The touch-tone capability that allows the VIS to collect touch tones as they are entered by callers, even before they are asked for. The touch tones are then used in the order in which they were received. This allows callers to respond to more than one prompt at a time, without having to listen to the intermediate prompts.

Dial Through — The touch-tone capability that allows callers to respond while a prompt is being played. The play back of speech ceases and the script responds to the key that was pressed. This is similar to barge in for WholeWord SR. Also known as talk off.

Talk Off — *See "Dial Through."*

Touch Tone — The signal sent when a caller presses any of the twelve keys on a push-button telephone which sends dual tones [rather than rotary "pulses"].

Speech Recognition Terms

Barge In — The WholeWord speech recognition capability that allows callers to respond while a prompt is being played. This is similar to dial through for touch tone. It is not currently used in FlexWord SR.

Connected Digits A sequence of digits spoken by a caller without intentional or regular pauses in between digits. The WholeWord speech recognition capability can recognize digits spoken in this manner.

FlexWord Speech Recognition The VIS capability that recognizes callers saying one of many custom vocabulary words. Recognition is based on matching caller speech to word models fashioned from representations of sub-words [phonemes], the smallest unit of speech.

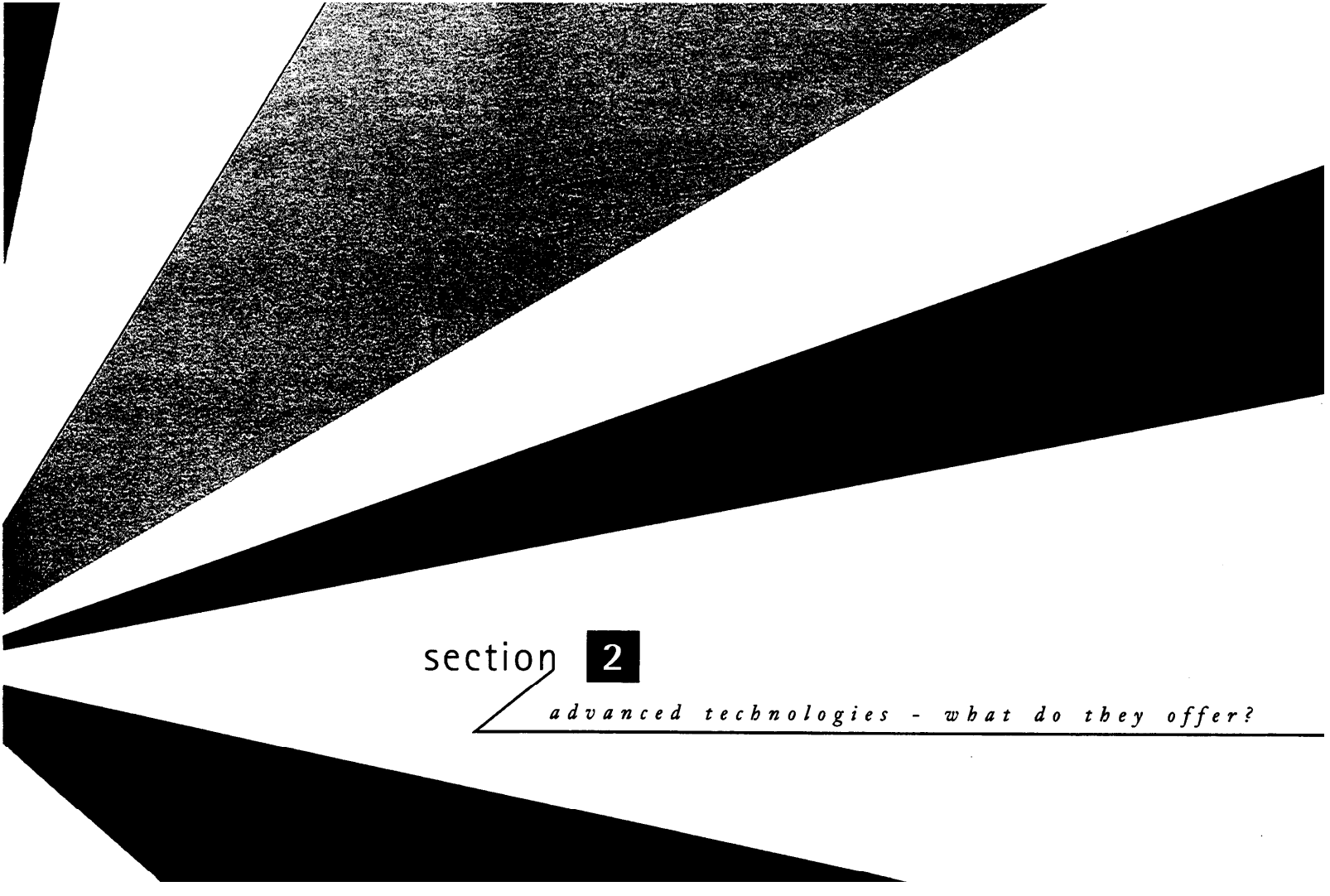
FlexWord Toolkit An optional package that provides a point and click, graphical way for you to define custom wordlists for FlexWord speech recognition.

Grammar The set of rules by which speech is recognized by WholeWord speech recognition. For example, a Prompt and Collect statement using the US_1_5 grammar will recognize the words “one,” “two,” “three,” “four,” and “five” in U.S. English. Somewhat comparable to a FlexWord wordlist.

Key Word One of the list of words that the VIS is instructed to recognize at a particular point in the transaction. For FlexWord SR, this is an item on the current wordlist [such as “small,” “medium,” or “large”]. For WholeWord SR, this an item in the current grammar [such as “yes” or “no”].

Phrase Screening The speech recognition capability that decides whether or not a candidate key word is a close enough match to be declared a valid key word. Works in conjunction with word spotting.

Recognition Type	The choices that are associated with the Mode field on the Script Builder Prompt & Collect form. The recognition type, minimum, and maximum number of digits values taken together allow the system to select a speech recognition grammar to be used for that recognition event. *Also see "Grammar."*
Recognize	The process within the VIS that compares caller speech to internal models and returns a match to the script.
Vocabulary	The set of wordlists associated with a particular FlexWord application script.
WholeWord Speech Recognition	The VIS capability that recognizes callers saying the digits zero [and oh] through nine, yes, and no. Recognition is based on matching caller speech to word models fashioned from many samples of people saying each entire, "whole"' word.
Word	In FlexWord SR, any word or phrase that can be recognized. Must be associated with a wordlist.
Wordlist	In FlexWord SR, a set of words that can be recognized at any one Prompt & Collect statement. Somewhat comparable to a WholeWord grammar.
Word Spotting	The speech recognition capability that allows the recognize to pick out key words from a stream of caller speech, which may include extraneous speech, background noise, or caller noises. Works in conjunction with phase screening.



section 2

advanced technologies - what do they offer?

Intuity CONVERSANT VIS offers a number of advanced technologies. This section describes what each advanced technology does well, the types of applications for which each is best suited, and how the technologies work together to allow you to build robust and useful applications.

Touch Tone

Touch-tone input continues to be a mainstay of voice response applications. More than half of the telephones in the United States are equipped with touch tone, although touch tone penetration varies considerably across geographic areas. A greater proportion of business locations have touch-tone capabilities than do residences. If you find that your *callers* will tend to have touch-tone telephones, it is economical and efficient to allow touch-tone input for most transactions.

If you are planning to provide voice response capabilities outside of the United States, you can use the VIS to write touch-tone applications in any language. There is a good chance, though, that many callers outside the U.S. will not have touch-tone capability. If touch-tone service is not widely available in your

country of interest, the VIS provides some speech recognition capabilities in several languages other than U.S. English. See the next section for more information.

Touch tone works well for allowing callers to make *menu* choices. It is also good for collecting numerical data such as credit card numbers or personal identification numbers. The VIS correctly recognizes touch-tone input virtually 100% of the time. Callers may make mistakes when entering touch tones, though, and a well-designed script will handle caller errors gracefully.

You can allow callers to *dial through* [also known as “talk off”] a *prompt.* That is, they can respond with a touch tone while the prompt is playing. The prompt will stop as soon as the touch tone is detected. You can also allow experienced callers to *dial ahead.* That is, they can respond to more than one prompt without waiting. If they are familiar with the upcoming choices, they do not have to listen to the next prompt before pressing the required touch-tone keys.

Whole Word Speech Recognition

Since some of your callers will probably not have touch-tone service, it makes good business sense to find a cost-effective way of providing service to callers who cannot or will not use touch tones. Many non-touch-tone calls can be handled by allowing callers to speak their responses.

The Intuity CONVERSANT VIS Speech Recognition package [called “WholeWord SR[1]” throughout this handbook] can be used to recognize a limited set of words, which includes:

- The words “yes” and “no”
- Single digits [zero through nine, and “oh” as a synonym for zero]
- A series of digits [also known as *connected digits*].

Callers can speak a sequence of digits without intentional pauses in between.[2] Shorter digit sequences are recognized more accurately than are longer digit sequences. See Section 4 for a discussion of recognition accuracy.

WholeWord SR supports *barge in.* Experienced callers do not have to wait until the end of a prompt to begin speaking their responses. As soon as the *recognizer* recognizes something the caller says, the prompt stops playing. This allows a single script to support both inexperienced callers [who can listen to the entire prompt] and experienced callers [who can barge in when they know what they want to say].

Barge in and wordspotting capabilities allow the caller to speak naturally and still be recognized.

WholeWord SR also supports *word spotting.* Callers do not have to say the *key word* in isolation. The recognizer can pick the key word out of a sequence of other words. For example, if the recognizer is listening for callers to say “yes” or “no”, it can also recognize “yes” if callers say “Yes, please, I do.” The recognizer finds a key word most accurately when the word is said all by itself, without any other words before or after it.

Version 5.0 offers three separate WholeWord SR packages. Each package supports recognition of some. spoken input in a single language. The languages currently supported are:

- U.S. English
- Canadian French
- Mexican Spanish[3]

Each package can recognize:

- Equivalents of the words “yes” and “no”
- Equivalents of single digits [zero / oh through nine].

Please note that Version 5.0 does not support the recognition of sequences of connected digits for languages other than U.S. English. For Canadian French and Mexican Spanish, digit strings can be entered in a tone-paced manner. See *Bilingual and Non-U.S. English Scripts* in Section 5 for more information.

In Version 5.0, any two of the above language packages can be used together on a single VIS to support bilingual applications. You can design a script that asks callers to indicate their preferred language, then play prompts and *announcements,* and accept single spoken digits and the words “yes” and “no,” in one of the two languages installed on your VIS. See *Bilingual and Non-U.S. English Scripts* in Section 5 for more information.

WholeWord and FlexWord [a different type of speech recognition, described below] capabilities can be used in the same application script. This gives you a high degree of flexibility in what you can ask callers to say. When using *Script Builder,* you can specify that a *Prompt & Collect* action use either WholeWord SR or FlexWord SR, depending on what you want callers to say.

WholeWord SR Cautions:

WholeWord speech recognition is most successful when it is used to augment a touch-tone application by handling callers who do not have touch-tone telephones. For scripts requiring callers to enter long sequences of digits, touch tone will tend to be more accurate than speech recognition.

Instead of an attendant, use WholeWord SR to handle callers who do not have touch-tone service.

The most successful applications first offer callers a chance to indicate that they have a touch-tone telephone [usually by pressing 1 on the keypad]. If no touch tone is detected, the application prompts callers to respond with spoken input [instead of transferring the call to an attendant].[4] In this manner, callers with rotary telephones and callers reluctant to enter touch tones can both be accommodated efficiently. Since non-touch-tone calls can now be automated, the number of calls handled by attendants is reduced.

Do not make the mistake of eliminating all attendants! Even a well-designed automated transaction cannot serve every caller. Some callers will have problems which cannot be handled within an automated transaction; some callers will become confused and make too many errors. Since callers may want and need to speak to a person, you can provide excellent customer service by making sure that there are some attendants available.

If your application takes input that can easily map to touch tones, do not ignore touch tone in favor of speech recognition. Prompting for speech input only in places in the script where touch tone would perform well might cause problems. For more information on this, see Sections 4 and 5.

Script requirements and caller characteristics will help you decide if SR is a good alternative.

Take care in deciding to allow spoken input. Some applications might not lend themselves to spoken input. If you think that callers will often be speaking from a noisy environment like a car or a train yard, touch-tone input will likely be quicker and more accurate. Some callers may have security concerns about speaking private information [like an account number] aloud if they will sometimes use the service from outside their homes. If callers must enter a long series of digits or more than a few different data items, touch tone can work faster and achieve better accuracy than WholeWord SR.

When using more than one language, a Version 5.0 script cannot be bilingual in the same way that a person is bilingual. A bilingual person can recognize two languages simultaneously. A bilingual script, how-

ever, needs to know ahead of time what language the caller will speak each time a response is spoken. The recognize can only recognize speech in a single language at any Prompt & Collect action. Scripts can be set up to determine the language a caller speaks at the beginning of the script, then conduct the rest of the interaction entirely in a single language.

Version 5.0 of the VIS may not be able to speak out dates, times, and monetary amounts in a satisfactory way for non-English languages, since the routines which speak this data are currently based on U.S. English conventions. See *Bilingual and Non-U.S. English Scripts* in Section 5 for more information.

FlexWord Speech Recognition

Another cost-effective way of handling callers who cannot or will not use touch tones is the Intuity CONVERSANT VIS FlexWord Speech Recognition package [abbreviated as “FlexWord SR” throughout this handbook], a flexible way for you to offer innovative services to your callers. FlexWord SR is also appropriate for scripts where it would be awkward for callers to enter touch tones [like when entering a name].

The FlexWord SR package can be used to recognize callers speaking words from a specific *vocabulary* defined by you, the script designer.[5] Since you are in complete control of specifying the words or phrases you want the VIS to recognize within your application, the vocabulary is specifically tailored to your applications. Allowing callers to say the option they want instead of saying a number assigned to the option can make the interaction much more natural and easy-to-use.

There are two ways to make your FlexWord vocabularies. The *FlexWord Toolkit* allows you to build your own vocabularies quickly and easily. The optional Toolkit gives you a point and click, graphical environment in which to add, delete, or change words on new or existing wordlists. You can use these wordlists in your applications immediately after you define them.

The AT&T custom vocabulary service is also available to build your vocabularies for a fee. Consider the size of your vocabularies and how many FlexWord application scripts you plan to have when deciding to purchase the Toolkit or use the custom vocabulary service.

Sample FlexWord Vocabulary [8 wordlists, 53 words total]

Wordlist 1 [11 words]
skirt
blouse
dress
pants
shorts
sandals
shoes
socks
belt
help
attendant

Wordlist 2 [6 words]
small
medium
large
extra large
help
attendant

Wordlist 3 [6 words]
petite
small
medium
large
help
attendant

Wordlist 4 [8 words]
sky blue
cranberry
brown
natural
teal green
rust
help
attendant

Worldlist 5 [6 words]
black
brown
light tan
olive
help
attendant

Wordlist 6 [5 words]
women's
men's
child's
help
attendant

Wordlist 7 [5 words]
narrow
medium
wide
help
attendant

Wordlist 8 [6 words]
visa
master card
american express
discover
help
attendant

FlexWord vocabulary items can be single words or multiple-word phrases, which are all referred to as *words.* The words must be divided into groups called *wordlists.* A wordlist includes all the words that can be spoken at a particular prompt. Each wordlist can contain up to 500 words, and you can have up to 200 wordlists.[6] At each prompt, you can ask callers to speak a word from one of the wordlists. FlexWord SR can support choices from small menus [such as a choice of clothing sizes] up through large menus [such as the names of all 495 people in your office].

Version 5.0 supports twice as many wordlists, and two and a half times as many words on each, as Version 4.0.

Each application has its own *vocubulury,* or set of wordlists. Althought only one Version 5.0 FlexWord package can be installed on a VIS at a time, a single FlexWord package can support many applications, each with its own vocabulary. Application scripts can even share wordlists.

As mentioned before, FlexWord SR and WholeWord SR can be used together in a single application script. This allows you many possibilities in what you can ask callers to say. When using Script Builder, you can specify that a Prompt & Collect action use either Flexword SR or WholeWord SR, depending on what you want to prompt callers to say. If you want callers to say yes or no, a series of digits, or single digits, you would choose a WholeWord *recognition type.* If you want callers to say a word or phrase from your custom vocabulary, you would choose a FlexWord wordlist.

FlexWord SR supports word spotting. Callers do not have to say the key word in isolation. The recognize can pick the key word out of a sequence of other words. For example, if the caller says “I'd like New Zealand, please,” the recognize can find the key word “New Zealand.” Remember, though, that the recognize finds a key word most accurately when the word is said all by itself.

Unlike WholeWord SR, Version 5.0 FlexWord SR does not support barge in. Callers must wait until the end of a prompt to begin speaking their response This has script design implications, since a single script must support both experienced and inexperienced callers. Inexperienced callers will tend to need more information from prompts, while experienced callers will need less, and may try to respond before the prompt is finished [especially if they're used to doing so at prompts which use WholeWord SR]. See Section 5 for what can be done to accommodate both types of callers, and how to best combine FlexWord SR and WholeWord SR in a single application.

Version 5.0 FlexWord SR can have a U.S. English vocabulary only. Due to underlying processes optimized for U.S. English, words and phrases in other languages will not be recognized well, especially if pronounced by native speakers of the other languages.

FlexWord recognition accuracy depends on a number of attributes, including the degree of similarity among the words on the wordlist, the size of the wordlist, and caller behavior [such as whether or not extra words or phrases are spoken along with the key word]. Although recognition accuracy can be assessed only on an application basis, performance is best for smaller vocabularies whose words and phrases are very distinct from each other. See *Using FlexWord Speech Recognition* in Section 5 for a discussion of this issue.

Flex Word SR Cautions:

Like WholeWord SR, FlexWord speech recognition is most successful when it is used to augment a touch-tone interaction. The most successful applications first offer callers a chance to indicate that they have a touch-tone telephone [usually by pressing 1 on the keypad]. If no touch tone is detected, the application prompts callers to respond with spoken input [instead of transferring the call to an attendant]. In this manner, both callers with rotary telephones and callers reluctant to enter touch tones can be accommo-

dated efficiently. Since non-touch-tone calls can now be automated, fewer calls will be handled by attendants. As in the WholeWord SR discussion above, using FlexWord SR does not mean that you can forgo attendants altogether.

If your application takes input that can easily map to touch tones, do not ignore touch tone in favor of speech recognition. Allowing only speech input where touch tone would perform well might cause problems. For more information on this, see Sections 4 and 5.

As with WholeWord SR, take care in deciding to allow spoken input. Some applications might not lend themselves to spoken input. If you think that callers will often be speaking from a noisy environment like a car or an airport, speech recognition might not be the best solution.

Text-to-Speech

This advanced speech technology allows your application to convert text to spoken words. This ability is especially useful when it is not practical to pre-record the information to be read [if it is lengthy, changes often, and / or comes from a database]. Information you could consider reading out with *Text-to-Speech* includes:

- News wire stories
- Magazine articles
- Rules and regulations
- Names and telephone numbers of doctors associated with a medical plan
- Telephone directory entries.

In most of these cases, the cost of having a professional speaker pre-record all the mention the time involved, is prohibitive. Text-to-Speech is an economical way to access to extensive or quick-changing information.

The Intuity CONVERSANT VIS Text-to-Speech feature expands abbreviations. If the text you want spoken out contains abbreviations such as “Ave.” or “Mr.”, Text-to-Speech can read them out as “Avenue” or “Mister.” Further, Text-to-Speech also allows you to specify the type of information that is being spoken. If you prepare Text-to-Speech to read an address, it will read “Dr.” as “Drive,” but if you prepare it to read a name, it will read Dr. as “Doctor.”

Text-to-Speech prompts can also be used in place of *pre-recorded speech* when prompting callers. This can save you some time when you prototype your application [see Section 6].

The Text-to-Speech capability is fully compatible with touch-tone input. If the script allows, callers can press a touch tone to dial through a Text-to-Speech prompt or announcement the same as if it were a pre-recorded prompt or announcement. Similarly, you can also allow callers to dial ahead, responding to several prompts in a row.

Text-to-Speech is also compatible with WholeWord and / or FlexWord input. For WholeWord SR, if the script allows, callers can respond with speech during a Text-to-Speech prompt or announcement the same as if it were a recorded prompt or announcement. However, when using Text-to-Speech prompts with FlexWord input, barge in is *not* available in Version 5.0.

Text-to-Speech Cautions:

The Text-to-Speech feature works primarily as a “reading machine.” It assumes that the information it is reading is structured in standard English sentences [complete with punctuation, capitalization, subject, object, and verb]. If the information you want spoken is not written in complete sentences [for example, a series of database fields like “name,” “address,” and “telephone number”], the output will not be as understandable as full sentences. See *Using Text-to-Speech* in Section 5 for more information.

Text-to-Speech pronounces words in a male voice with a U.S. English accent. Many callers find the speech easy to understand, but callers may have to get used to it. Because of this, it might not be suitable for some applications. See the *Using* Text-to-Speech *section* of Section 5 for a discussion of this issue.

Fax

The Fax capability lets any VIS application script send fax messages to callers [fax-on-demand], and receive incoming faxes from callers [fax messaging]. Your applications can give callers the opportunity to request information such as:

- Blank forms [such as tax forms, applications, or entry blanks]
- Customer service information
- Financial or medical records
- Reservation confirmations
- Advertising brochures.

To get these fax capabilities, you must purchase and install the AT&T Fax Attendant System® on the same processor as your VIS software.[7] You then have the power to write voice response applications which can send graphic and textual information to callers' faxmachines upon their request.

The AT&T Fax Attendant System is a feature-rich package in its own right. It provides fax mail [fax messaging], fax machine coverage and overflow control, remote fax retrieval, a customizable fax-on-demand service, and many other features. If you choose, you can use this package separately from your voice response applications.

Fax Actions let your script use UNIX shell commands, even if the script doesn't send fax messages.

The VIS Script Builder Fax Actions are the key to fax-on-demand. The fax Actions allow voice response applications to send either pre-stored faxes or faxes created directly by the application. You can easily customize your own cover pages, You can even tag faxes for delivery when telephone rates are low. And, if the caller's fax machine is busy, the VIS automatically tries again later. The Fax Actions also let an application execute UNIX® shell commands to create text files for transmission to the caller.[8] Applications can send anything that can be put into a text file [including host screens and ORACLE® database reports]. You can even put text files into the VIS using a scanner or a fax machine.

Fax capabilities can be used in the same application script as WholeWord SR, FlexWord SR, Text-to-Speech, and touch tone. This gives you many options for designing innovative applications. For example, you could design an application which gives physicians access to hospital patients' medical records. Once the physician phones into the VIS and gives a login and personal identification number [using either WholeWord SR or touch tone], he or she enters a patient identification number and listens to computerized medical records spoken out using Text-to-Speech. The physician can also opt to have a copy of the hospital record faxed to his or her office.

Voice Response Applications

Voice response can be used in a wide variety of situations. Let the following examples give you ideas on how to use the power of voice response for your business.

Repetitive Information Requests

In many situations, pre-recorded prompts and announcements will work well. If pre-recorded speech is not possible [for example, when the information is lengthy or quick changing], you can speak it out with Text-to-Speech. You can also give callers the option to request that the information be sent by fax. Application areas include the following:

- **Directions**
- **Classified advertisements**
- **Movie times**
- **Inventory availability**
- **Concert information**
- **Extracurricular activities, game times**
- **Rules or regulations**
- **Company savings plan statements**
- **Product brochures**
- **Park or recreational locator**
- **Interest rates**
- **Sports scores or racing results**
- **Medical records**
- **Transportation rates / schedules**
- **Loan rates**
- **Foreign exchange rates**
- **School homework**
- **Branch / store locator**
- **Price Lists**

- **Repair status**
  Customer can enter a job number to find out the status and estimated cost of repairs being performed.

- **Flight information**
  Callers can listen to or receive a fax listing all flights consistent with their specified travel needs.

- **Conference information**
  Registered callers can hear agendas and descriptions of sessions or request copies via fax.

- **Restaurant locator**
  Callers can select restaurants by location or cuisine, then have menus, directions, or other information spoken or sent by fax.

- **Real estate information**
  Callers could enter a house identification number found on a “for sale” sign, and request a fact sheet to be either read with Text-to-Speech, or faxed to them.

- **Music Line**
  Callers can hear current songs, or find out when and where particular artists will be performing.

- **Shipping**
  Callers can dial in for the latest status information about their shipping job, which can be spoken out or faxed to them.

- **Lottery numbers**
  Callers can hear the most recent winning numbers, or enter the date of a particular drawing.

- **Customer service information**
  Updates on product maintenance issues and their resolutions could be faxed or read.

- **Shareholder services**
  Callers can request a financial statement and prospectus, either spoken or by fax.

- **Blank forms**
  Applications, voter registration forms, entry blanks, or tax forms could be sent by fax.

Information from Large Databases

Callers can browse titles of articles, then select one and either listen to it or request a fax. Titles and entire articles can be read out using Text-to-Speech, or pre-recorded. Application areas include the following:

- Magazine articles
- News wire stories
- Electronic mail
- Public records such as deeds or marriages
- Directory listings
- Congressional Record
- Catalog item descriptions
- Telephone directory listings

Order Processing

You can design the application to accept either touch-tone input, spoken input, or both. Callers can be given the opportunity to receive a confirmation or status information by fax. Application areas include the following:

- Purchases, catalog ordering
- Employee time reporting
- Class registration
- Renewing subscriptions or memberships
- Pay-per-view television

- **Hotel reservations**
  Callers can register and pay by telephone, then receive a confirmation fax including a map and directions.

- **Signing up for time slots**
  Callers sign up for tee times or restaurant reservations.

- **Benefits registration**
  Callers sign up for a combination of benefits, and can request confirmation via fax.

- **Talent agency**
  Producers can call to request information on actors with particular talents or physical attributes. Information can be spoken out, and photographs and resumes can be faxed to the caller.

- **Brokerage services**
  Callers can buy and sell stocks and bonds, receiving a fax confirmation of the transaction.

- **Travel reservations**
  Callers can reserve a seat and pay over the telephone. Confirmation can be sent via fax.

- **Wholesale merchandise orders, restocking**
  Manufacturer calls a supplier to request spare parts. A confirmation of the order can be faxed to the manufacturer.

Information Gathering

Use touch-tone or spoken input to gather information from those who call a designated number. Application areas include the following:

- Consumer or political surveys
- Customer feedback / complaints
- Shareholder balloting
  Callers can vote for a company's slate of officers.

footnotes

1. *Tbis speecb recognition technology is called "Whole Word" because the internal representations of the words are based on samples of thousands of people speaking "whole" words.*

2. *Note that numbers above nine must always be spoken as single digits. For example, callers would enter 35 by saying "three, five."*

3. *The VIS now supports additional languages & countries on the Version 4.0 generic. Contact your AT&T representative for details.*

4. *See* Combination Touch Tone and Whole Word S R Scripts *in Section 5 for an example.*

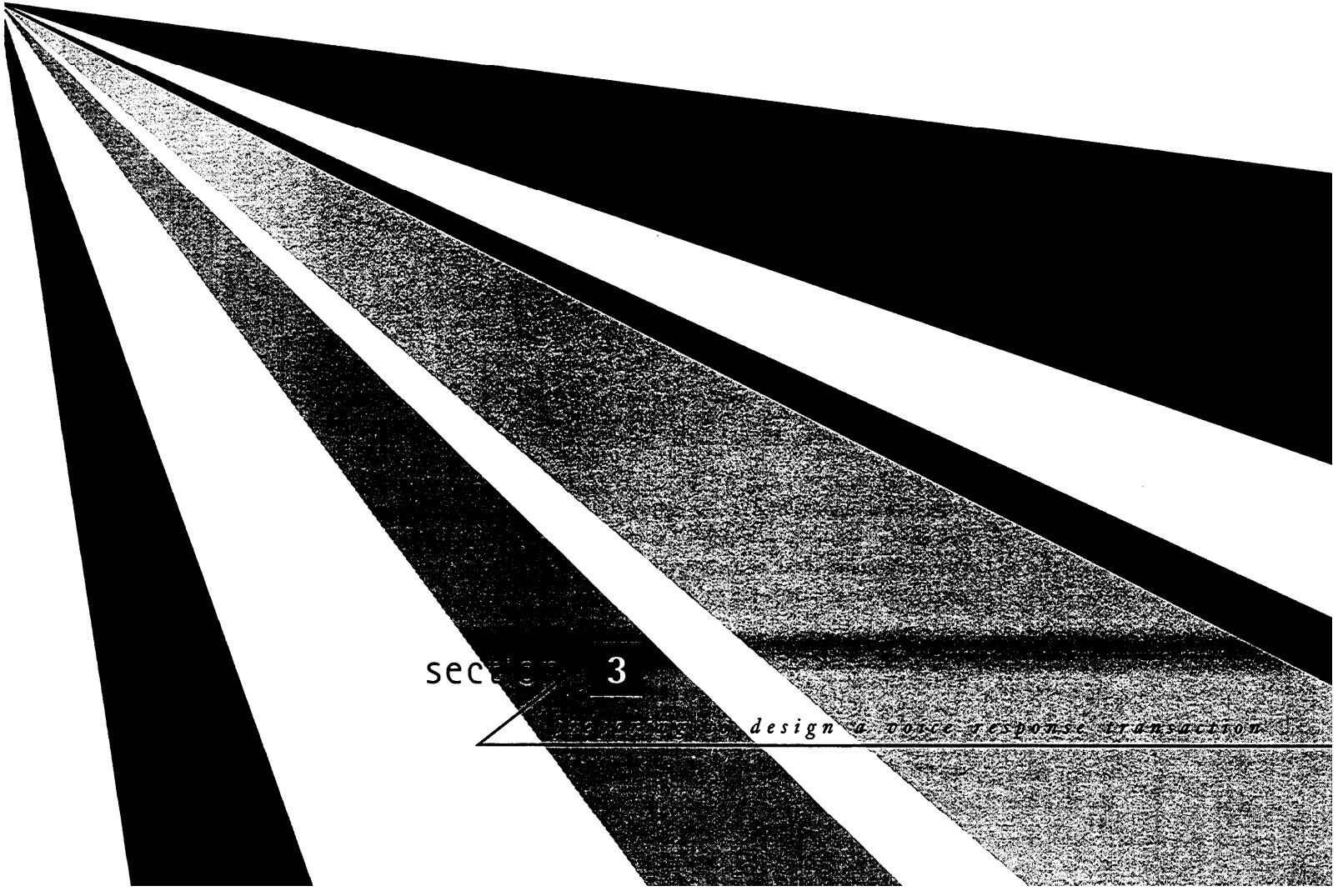
continued

5. *Flex Word speech recognition differs from WholeWord in that the internal representations of the words are based on pieces of words [called phonemes or subwords], instead of "whole" words. These representations of phonemes can be combined in an infinite number of ways in order to piece together words for the system to recognize. This means that we can bypass the lengthy and costly* sample *gathering process that must be done to support Whole Word recognition.*

6. *Although you may have up to 500 words on a wordlist and 200 different wordlists, a single Flex Word package currently supports only 2000 total words. AT&T* will *soon begin performance testing on larger vocabluaries. If you are interested in a larger number of total words, contact your AT&T representative.*

7. *The Script Builder Fax Actions [which support fax-on-demand] will soon be available without the Fax Attendant System. Fax messaging will still be available through the Fax Attendant System only. See your AT&T representative for details.*

8. *The Fax Action that supports execution of UNIX shell commands, called "Exec_UNIX," can also be used in application scripts that do not send faxes. The Exec_UNIX action gives you the ability to issue any type of shell command, and to use the output in your application. For example, a script could use Text-to-Speech to read out the contents of electronic mail files accessed through a shell command issued by the Exec_UNIX action. See* Using Fax-on-Demand *in Section 5 for more information.*


section 3

There are a number of basic principles that are useful when designing voice response applications. Experts call these "usability principles," since they are the cornerstone of designing a usable application. Once you understand these general recommendations, you will be prepared to move on to the more specific recommendations in the next section.

Know Your Callers

This principle guides all good user interface designers. Choices you make in the transaction design must be informed by who the application will serve. Find out as much as you can about the people who will phone in, and use the information when designing scripts.

You maybe able to gather much of this information by talking to your colleagues. Support and services personnel can be fine information sources because of their extensive experience with customers. You might also want to interview people in the marketing and sales departments.

Relevant information to gather includes the following:

Who are your callers?

Your callers have personal attributes that could affect how they will interact with the application you are designing.

What language do they speak? If you expect that many callers will have an accent that's different than what the speech recognition package you are using expects, be careful. You might consider offering a bilingual service. If you are offering service in only one language but anticipate many accented callers,[1] you may want to send rotary callers to an attendant immediately, since accented speech will not usually be recognized as accurately as non-accented speech.

How old are they? If you expect that a high percentage of callers may be children, you should use simple language and very short menus. Also, since speech recognition works best with deeper voices, you may wish to use touch-tone input only, and send rotary callers to an attendant immediately. If you expect that many of your callers will be senior citizens, you might want to allow them more time to respond. Make sure to reduce confusion by making the transactional simple as possible.

Are they employees of your company? You may want a script designed for in-house use to appear and perform differently than one designed for your customers. For example, if you provide training classes and / or job aids, you probably can use shorter and more concise prompts. Jargon and technical terms may be appropriate for employees, if you can assume that they all will know what you are talking about.

Note any other personal attributes that could affect how callers will interact with your application, and work to make sure that your application serves ail callers well.

How often do they call?

Some applications are meant to be used only one time, while others can be used more often. Callers may even access some applications several times a day.

If you expect callers to use the application rarely, your prompts and announcements may need to contain explicit information and instructions. Prompts for applications used very often can be made extremely short, so as to not burden callers. If you expect your callers to be mixed in their experience levels, allow the more experienced callers to dial through or barge in during prompts.

Don't fall into the trap of designing only for yourself; make sure the script design suits your callers.

How well do they know the subject matter?

If you expect that most of your callers will be unfamiliar with the subject matter [for example, if you are describing a new service or product], you should take special care in structuring and presenting the information so that your audience can understand it.

Will they be given a user guide? Will they consistently use it?

If you know your exact calling population [for example, if all callers will be subscribers to a service], you can take advantage of this and provide a user guide or job aid. User guides can be very simple, providing just enough information so that callers will not be surprised by the automated transaction. More detailed user guides can include all the prompts and options, so callers can prepare answers ahead of time and become oriented to the application.

If you distribute a user guide and expect that most callers will use it, you may be able to provide shorter and more concise prompts than if callers did not have a guide. Beware, though, since this could adversely affect callers who forget to use the guide.

Use Simple and Natural Dialogue

Learning as much as you can about callers will allow you to determine what is "simple and natural" for them. Design uncomplicated, straightforward applications that use terms familiar to the callers.

Minimize Demands on the Caller's Memory

Psychological research has shown that a caller's short-term memory can hold no more than five to nine separate pieces of information at one time. For each menu item you present to callers, two different pieces of information must be remembered the option [*"For rate information..."*] and the action required to choose the option [*"...press 2."*] Menus should offer no more than four or five items at once, since callers cannot remember more than that.

See Section 5 for specific recommendations.

The fewer items callers have to remember, the more likely it is that they will remember them. Try to make the interaction easy for the callers. Keep menus short, and avoid wordy prompts and announcements.

Be Consistent

Consistency in the structure of menu choices and in the presentation of information helps prevent callers from becoming confused. By using consistent language and requesting consistent action of callers, you simplify the interaction and reduce demands on memory.

Provide Feedback

Whenever callers press touch tones or speak in response to a prompt, let them know how the system interpreted the response. This clarifies the transaction for callers. For instance, if callers press zero for an attendant, the script could say *"Please hold for an attendant."* Although the script does not explicitly state that it heard the caller press zero, the interpretation of the key press is implicit in the announcement.

Provide Easy Exits

To prevent callers from getting trapped in an application script, tell them how to reach an attendant, end a transaction, or return to the main menu. You can give this information at the beginning of the transaction. Provide these types of escapes within your script so that callers can always return to a place they know or obtain help as necessary. This gives your callers more control over the interaction.

Offer Shortcuts

If you expect some callers to call often enough to become experts, provide ways to shorten their interaction with the application. For example, allow them to respond before the end of a prompt so that they can move through the interaction at a pace that is comfortable for them. You may want to consider providing separate scripts [or separate branches of a single script] for novice and expert callers.

Provide Good Error Messages

When a caller makes an error:

Although good script design can minimize mistakes, caller errors are inevitable. A well-designed script deals gracefully with caller errors, allowing callers to recover quickly and easily after taking an action that the application considers inappropriate. The application should indicate that the caller made an illegal or incorrect entry, then give more information about how to make an appropriate entry.

For example, if the first prompt states: *"Please say your ID number now"* and the caller does not respond, the script could re-prompt with: *"Please say the 5-digit ID number, found at the bottom of your statement."*

When the recognizer makes an error:

Good script design can help callers understand what should be spoken, minimizing speech recognition mistakes due to caller error. However, recognizer errors will occur. A well-designed script deals gracefully with them, letting callers control the transaction so that recognition errors are not catastrophic. The application should speak back what it recognized, and callers should be given the opportunity to confirm or deny the recognition results. If a recognition error occurs, then callers are in control and can repeat the information. For example:

"What department would you like?"
< **lawn care** >
"Calling hardware. Yes or No?"
< **no** >
"What department would you like?"
< **lawn care** >
"Calling lawn care. Yes or No?"
< **yes** >
"Transferring"

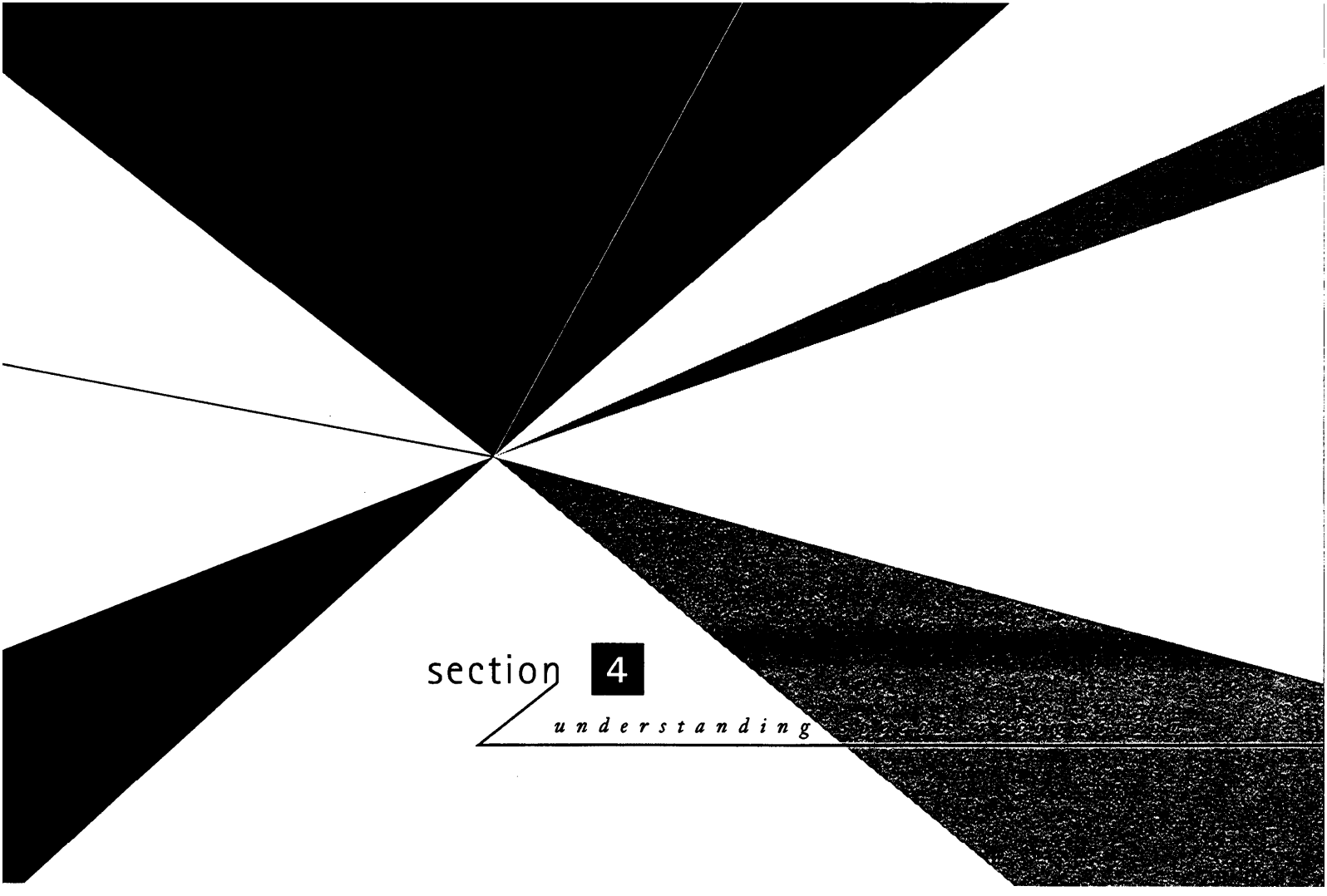
Prevent Errors

By following the above principles and the specific recommendations in Section 5, you can prevent errors from happening. Well-designed applications take into account the strengths and weaknesses of both the callers and the technologies, and will delight your customers!

---

footnotes

1. *That is, "accented" with respect to the WholeWord package in use. A native of Ohio calling in to a French Canadian application is considered accented.*


section 4
understanding

Touch Tone vs. Speech Recognition-What's So Different?

You may have already offered touch-tone-based voice response services to your customers. Now you may be interested in making use of speech recognition technologies. There are some special things you need to know that will help you approach speech recognition wisely. Although you might already know how to design good touch-tone applications, this section will help you appreciate the differences between touch tone and speech recognition, and how the technologies affect script design.

The way the VIS handles touch-tone and voice input are fundamentally different, so the script designer must take care to use each in a way that is tailored to how callers will interact with the script.

Let's compare and contrast touch-tone-based interactions with speech recognition-based interactions. Here's a summary, with more explanation to follow:

	Touch-Tone Input	Spoken Input
Possible Input	One or more touch tones.	Any noise that a person can make, or any background noise that can be heard over the telephone.
What System "Knows"	Touch tones.	Set of word models restricted by grammar.
How System Analyzes Input	Takes touch-tone signals and maps to number on the key.	Takes speech sample and tries to match it to the active word models
System Accuracy	Interpretation extremely reliable [virtually 100%].	Interpretation very reliable, but not perfect [for example, approximately 97% per digit, lower for a series of digits].
Conclusion	Excellent mapping between input and what recognizer "knows." **"Perfect" accuracy.**	Imperfect mapping between input and what recognizer "knows." **"Imperfect" accuracy.**

Possible input: In a touch-tone based interaction, the only possible input is the pressing of one or more touch-tone keys. Callers press keys on the telephone keypad which correspond to what they want to tell the script. It's possible, though unlikely, that speech or noise in the background will mimic a touch tone.

For a speech recognition interaction, the possible input is limitless! Anything that can be transmitted over the telephone counts as input, including any noise that callers can make [words, throat clearing, sneezes, breathing] and any background noise that can be transmitted over the telephone [office noise, traffic sounds, doors slamming, dogs barking].

What the system "knows:" In a touch-tone based interaction, the VIS is setup to detect touch-tone key presses only. It cannot "hear" anything else that comes over the telephone.

For a speech recognition interaction, at each point in the script where the recognizer "listens" for input, the script designer has indicated what the script should listen for. For WholeWord SR, the recognizer is guided by a recognition type / *grammar* [for example, US_1_5 means it will listen for one U.S. English digit between one and five, inclusive]. For FlexWord SR, the recognizer is guided by a wordlist, meaning it will listen for one of the words or phrases on the specified wordlist [for example, the wordlist "main_menu" might contain the words *order, status, information, help, and attendant].*

How the system analyzes input: In a touch-tone based interaction, the system "hears" the dual tone, multi-frequency signal that makes up the touch tone and maps it to the corresponding number on the key.

For a speech recognition interaction, the recognizer "listens" to the stream of speech, and tries to find one of the words that the script knows is currently allowed.

System accuracy: In a touch-tone based interaction, the system almost always interprets the touch tone correctly. Touch tone recognition mistakes are rarely made.

For a speech recognition interaction, the interpretation of speech is not as reliable as for touch tones. For example, single digits are correctly recognized at about 97% [see the next part for a discussion of accuracy]. Individual callers will get better or worse results depending on the level of background noise, the amount of extraneous speech or sounds, the pronunciation of the recognized words, and voice qualities such as pitch and volume.

See *Digit Entry* [Validating *a Digit* Sequence] in Section 5 for ways of improving recognition accuracy beyond 97% per digit.

Conclusion: In a touch-tone based interaction, there is excellent mapping between what callers can input and what the system "knows." Therefore, system accuracy will be virtually flawless.

For a speech recognition interaction, there is imperfect mapping between what callers can input and what the recognizer "knows." Therefore, recognize accuracy will not be flawless.

Why this concept is so important? *A successful application takes advantage of the best aspects of speech recognition, while minimizing the flaws.* Here's an example. Let's say that you are working on a script that callers will use to report problems with telephone service. Part of your transaction involves asking callers for a telephone number. For a touch-tone interaction, you could prompt *"Enter your 10-digit telephone number."* The caller can do one of three things: press 10 keys, press fewer than 10 keys, or do nothing.

- If the caller presses 10 keys, the application reads back the telephone number for confirmation: *"Did you enter 614-860-4001?"*

- If the caller presses fewer than 10 keys, the application detects this and re-prompts the caller for a valid entry.

- If the caller does nothing, the script receives no touch tones. When this happens, the script can re-prompt, providing more information.

For a similar speech recognition interaction, you could prompt *"Say your 10-digit telephone number."* The caller can say anything at all in response [words or other sounds] or say nothing. All the recognizer can do is listen, map the sounds to the appropriate grammar [US_DIG, with minimum and maximum digits = 10], and come up with a series of digits. No matter what the caller says, the recognizer does the same thing: it listens to the sounds coming through the telephone, and tries to find 10 digits.[1]

- If the caller says 10 digits, the recognize tries to map the utterance to 10 digits, then reads back the result for confirmation:
  "Did you say 674-860-4001?"

- If the caller says fewer than 10 digits, the recognize tries to map the utterance to 10 digits, then reads back the results for confirmation:
  "Did you say 618-604-0001?"

- If the caller remains silent and there is any background or line noise, the recognize will do the same thing-map what it hears to 10 digits, and present the result for confirmation.

- If the caller remains silent and is in very quiet surroundings, the recognize may find nothing to match. If this is the case, it returns a null string [meaning “silence”] to the script, and the script can re-prompt, giving the caller more information. This is considered a “correct rejection.”

- If the caller says “I don’t have a phone number yet,” the recognizer again tries to map the utterance to 10 digits, and reads the result back for confirmation: *“Did you say 201-888-9182?”*

In the second, third, and fifth cases, when callers say something that the script is not expecting, the script has no way of telling so, since current technology does not allow us to reject all undesired speech. As it stands, the interaction can confuse the caller and undermine the caller’s confidence in the voice response system and your company.

When designing speech recognition applications, then, we must keep a special goal in mind to increase the probability of a successful application: *do all you can to get callers to say only what you want them to say, when you want them to say it.* Figuring out how to do this takes effort and testing [see Section 6].

What can be done to improve the script design in the example? What could increase the chance that callers will say what the recognize is prepared to accept? One way to do this is to first ask if the caller has the required information, by prompting: *“Do you have your telephone number available?”* Callers who say “no” can be quickly transferred to the attendant, while callers who say “yes” can be prompted for the number.

Remember, the way the VIS handles touch-tone input and voice input are fundamentally different, so the script designer must take care to use each in a way that will maximize callers' success in interacting with the script. Section 5 includes many suggestions on how to do this.

Understanding Speech Recognition Accuracy

Caller Behavior and System Behavior

The previous part mentioned speech recognition accuracy. This part will give you some more information about recognition accuracy as it relates to application design.

The following diagram applies to both WholeWord and FlexWord speech recognition. When interacting with an application script, callers can do one of three things:

- Say what's expected [a key word or key word, perhaps with non-key words before and/or after]
- Say something that's not expected [non-key word]
- Say nothing.

The recognizer can do one of three things in response:

- Match the key word that the caller actually said
- Match a different key word
- Find no match at all.

The caller and recognizer behaviors result in caller-observable system behaviors, as follows:

→ recognizer ↓ caller	Correct Key Word Matched	Incorrect Key Word Matched	Nothing Matched
Caller Said Key Word(s)	correct recognition	incorrect recognition	no recognition
Caller Said Non-Key Word or Said Nothing	insertion error	insertion error	correct rejection

The white boxes in the table above show that there are two ways the recognizer can be “correct”:

- Correctly recognize that the caller said a legal key word [correct recognition]
- Not find a match for a non-key word or for silence. [correct recognition].

Also, there are three ways the recognize can make an error:

- Incorrect recognition [recognizing the uttered key word as a different key word]
- No recognition
- Insertion error [recognizing a non-key word as a key word].

When designing a speech recognition application, you want to do all you can to concentrate the system responses in the white boxes. To encourage correct recognition, your prompts and announcements must help callers to say key words with a minimum of extra speech. Your control over correct rejection also comes from your application design, and how well it encourages callers to say things that the recognizer can handle.

What is Recognition Accuracy?

The previous section mentioned that the recognition accuracy rate for a single digitis 97%. What exactly does this mean? Accuracy is measured over many speakers, different types of applications, various environmental conditions, and regional accents. 97% accuracy does not mean that every caller speaking the digit will be correctly recognized 97% of the time. Nor does it mean that 97% of people who call in will be recognized correctly.

A 97% accuracy rate means that for a large number of many different types of callers over an extended period of time and under a number of different conditions, 97% of valid inputs will be correctly recognized. Some callers will do better than 97%, while some will do worse.

Note that this definition of recognition accuracy assumes that callers are saying something that the recognizer can recognize. Should a caller give an invalid input, the recognizer should reject it.

For isolated digits, the VIS recognition accuracy rate is 97% per digit. Accuracy drops slightly as digit sequences get longer, due to the difficulty of detecting boundaries between digits. When two or more digits are spoken in a row ["connected digits"], the rate is $.97^{n}$, where n = the number of digits that the recognizer expects callers to say. So, for example, a three-digit sequence is expected to have an accuracy rate of $.97^{3}$, or 91%. In general, you can subtract about three percentage points for each additional digit spoken in a row.

You can improve recognition accuracy for connected digits by using custom grammars, checksums, or database lookups. For a discussion of these topics, see *Digit Entry* [Validating a Digit Sequence] in Section 5.

This recognition accuracy concept is similar whether you're talking about WholeWord or FlexWord speech recognition. Since the WholeWord vocabulary is fixed at 13 items, we can measure and advertise accuracy rates. However, since FlexWord vocabularies are completely customized for each application, there is no overall accuracy rate. For FlexWord SR, accuracy can only be assessed on a per-application basis. Each FlexWord application will have a unique set of words and phrases, which are grouped into wordlists. With some effort, the recognition accuracy of each separate wordlist can be measured.

Generally speaking, you can expect that the more different the words on the wordlist are from each other, the better the recognition accuracy for that wordlist will be. Many factors can affect FlexWord accuracy:

Good script design can minimize recognition accuracy issues.

- How different the words on the wordlist are from each other
- [more difference is better]
- Word length [words with a few syllables are better than one syllable words].
- Number of words in a wordlist [fewer is better]
- How well callers know how to pronounce the words [more accurate
- pronunciation is better]
- Background noise [quieter is better]
- Distracted or unprepared callers [prepared, concentrating callers are better]
- Application user interface design [a usable design is better].

The principles discussed in Section 5 will help you optimize the caller-script interface design, choose workable FlexWord wordlists, and educate and inform callers about what to say.

Actual measurement of speech recognition accuracy is a complex process. First, measure the success of your application using the technique outlined below. If these measures do not meet your needs, you may want to contact your AT&T representative for guidance on how to estimate and measure speech recognition accuracy.

What is a Successful Speech Recognition Application?

The success of a voice response application using speech recognition does not rely on recognition accuracy alone. Depending on what your success factors are, recognition accuracy could play a relatively minor role.

Speech recognition is a probabilistic technology. This means that there is always a chance that recognition will be either correct or incorrect. Therefore, the expectation of 100% accuracy is unrealistic. Not even people achieve 100% recognition accuracy in their conversations, especially over the telephone and with background noise. Although people may seem to be "perfect recognizers," we are not. We make use of very effective error correction mechanisms to clarify our communications. For instance, if you said to your friend "I'll meet you downtown on Sunday" your friend might ask "Did you say Monday?" Your friend wasn't sure what she heard, so she asked for clarification. Just as you can not always count on others to understand everything you say, your application must also be able to double-check spoken input.

Since we can't expect 100% recognition, how can we tell when a voice response application is successful? Consider multiple criteria when defining success. This involves measuring some aspects of how you currently do business, and deciding on a specific area [or a few different areas] to target for improvement. Think about why you are interested in automating some of your business interactions. To save time? To save money? To better serve your customers? To free personnel for more challenging tasks? Take your

most important reasons for automation, and determine measurable objectives for automation. For example, if you want to use voice response to better serve your customers, you might choose as an objective to reduce call holding times. A measureable objective might be to reduce call holding times by 50%. You could also figure out the yearly savings expected from reduced holding times, and factor this into your plan. Here are some examples of other types of measurable objectives:

Your measurable objectives will vary from script to script.

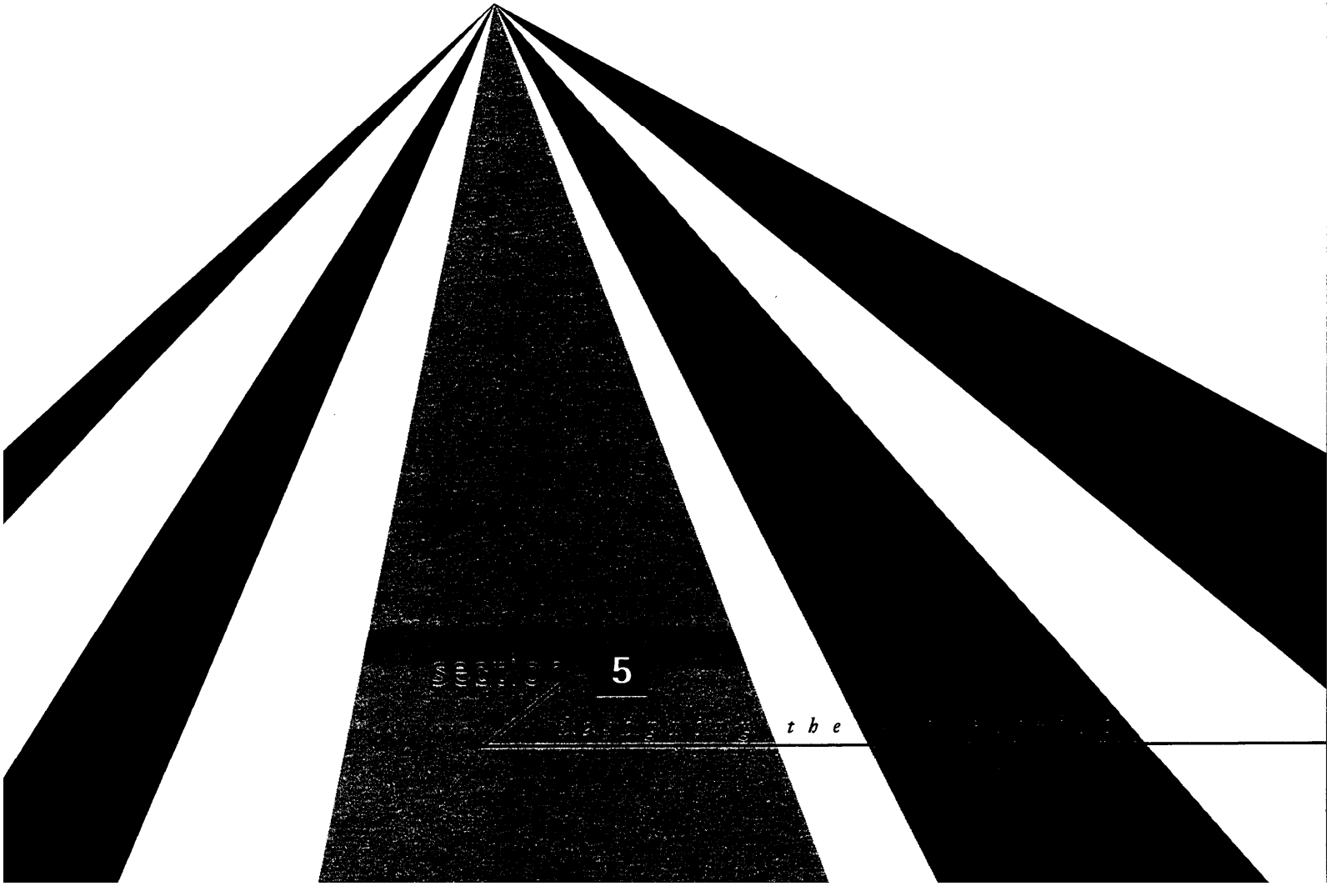
- Reduce your cost per call by 30%
- Increase call completion rates by 45%
- Reduce the percentage of callers who get a busy signal by 50%
- Increase customer satisfaction with the voice response transaction by 25%.

Remember to estimate the rate of savings for each objective.

Determining what your true objectives are, as well as how to measure them, will allow you to judge the effectiveness of your voice response applications.

footnotes

1. *If the recognizer detects no energy, the recognized result is "silence."*


5

This section applies the information contained in the previous sections. Each of these guidelines and suggestions indirectly related to one or more of the usability principles described in Section 3. Unless otherwise indicated, all suggestions are applicable to all Intuity CONVERSANT VIS Version 5.0 technologies.

Planning Considerations

This part offers some ideas for planning voice response applications. Thinking about these areas may give you inspiration about how you might approach your script design.

Make Use of Human Factors / Usability Engineering Resources

Human Factors and usability consultants are experts in the design and testing of human-computer interfaces. Some even specialize in telephone-based user interfaces. If your company does not have a pool of usability or Human Factors resources, you could turn to an independent consultant. These consultants are available on a contract basis to assist you in application design, testing, and deployment.

Contracting professional design and usability testing services can be cost effective. Off loading some of the effort can save you time and frustration. Customer satisfaction can also be increased, since you will offer your customers a well-designed, extensively tested application.

AT&T offers Human Factors design consultation services on a limited basis. For more information, contact your AT&T services specialist. For a directory of individuals and companies offering independent consulting in Human Factors and usability, call the Human Factors and Ergonomics Society at [310] 394-1811.

Use Professional Speakers

Pre-recorded prompts and announcements spoken by a professional speaker [such as a voice over artist or a disc jockey] can enhance the quality of your application script. It is usually appropriate to use the same recorded voice throughout the transaction. An exception is when you have a good reason to try to distinguish between different parts of the application. For instance, you may want to have “help” speech recorded by a male voice when the remainder of your prompts and announcements are in a female voice. This might help your callers to know their location within a script.

Throughout this section, if a topic pertains to a specific technology, it will be tagged with a technology icon, as on page 67. If the Intuity CONVERSANT VIS icon appears [as at the top of this page] the information is useful for *all* technologies offered.

Intuity CONVERSANT VIS offers a professional custom speech recording service in both male and female voices, in the language of your choice. If you are interested in purchasing custom or *standard speech* [1], contact your AT&T representative. The main advantage of using the custom speech recording service is that the speakers who will record the U.S. English custom speech currently are the speakers who recorded the standard speech, and so will provide continuity to scripts that use both custom and standard speech.

You many choose to contract a professional speaker on you own. An advantage of using your own professional speaker is that you may be able to obtain more control and faster response when adding new speech phrases. If you choose to contract a professional speaker on your own, consider the following guidelines:

- Record the speaker's voice to ensure you like the way it sounds. You may want to record several different speakers to compare the quality of their recorded voices.
- Make sure the speaker can maintain a constant speaking rhythm and intonation throughout the recording session.
- Make sure the speaker can maintain a constant, acceptable level of volume and distance from the telephone or microphone while recording.
- The speaker's pronunciation should be clear, but the words should not be over-enunciated.
- The recording environment should be as quiet and acoustically "dead" as possible. A carpeted room with soft walls is usually sufficient.

- Have all phrases prepared for the speaker to read in advance of the recording session.

If you contract a professional speaker, you can use the Intuity CONVERSANT VIS *Graphical Speech Editor Package* to record and modify the speech phrases you use for your application scripts. The Graphical Speech Editor allows you to rearrange parts of speech phrases by cutting, copying and pasting within a point and click, graphical user interface. See your AT&T representative for more information.

Consider Offering a User Guide

A printed user guide can also remind customers to use your service.

Consider providing your callers with an easy-to-understand, printed user guide. A good user guide can educate callers, set their expectations, and put them at ease. A user guide should provide enough information so that callers will be prepared to interact with the application. For example, if the application requires callers to enter account numbers, spouses' Social Security numbers, or other information they might not have memorized, the user guide can remind them to have this information on hand. More detailed user guides can list all the prompts and options, to allow callers to prepare ahead of time and follow along during the transaction.

The nature of your application and calling population will determine whether or not a user guide is appropriate. User guides are most useful when you know and can easily reach the members of your calling population. Examples of known caller populations include:

- Subscribers to your service
- Customers to whom you send bills
- Students, teachers, and parents of the school you serve
- Members of your company.

Consider Offering Caller Training

If you know that a particular group of callers will use the script extensively, consider offering training classes. This is most appropriate when you have a limited calling population, and / or a potentially complex or unusual application. The most useful training classes allow callers to phone into the script and do hands-on learning.

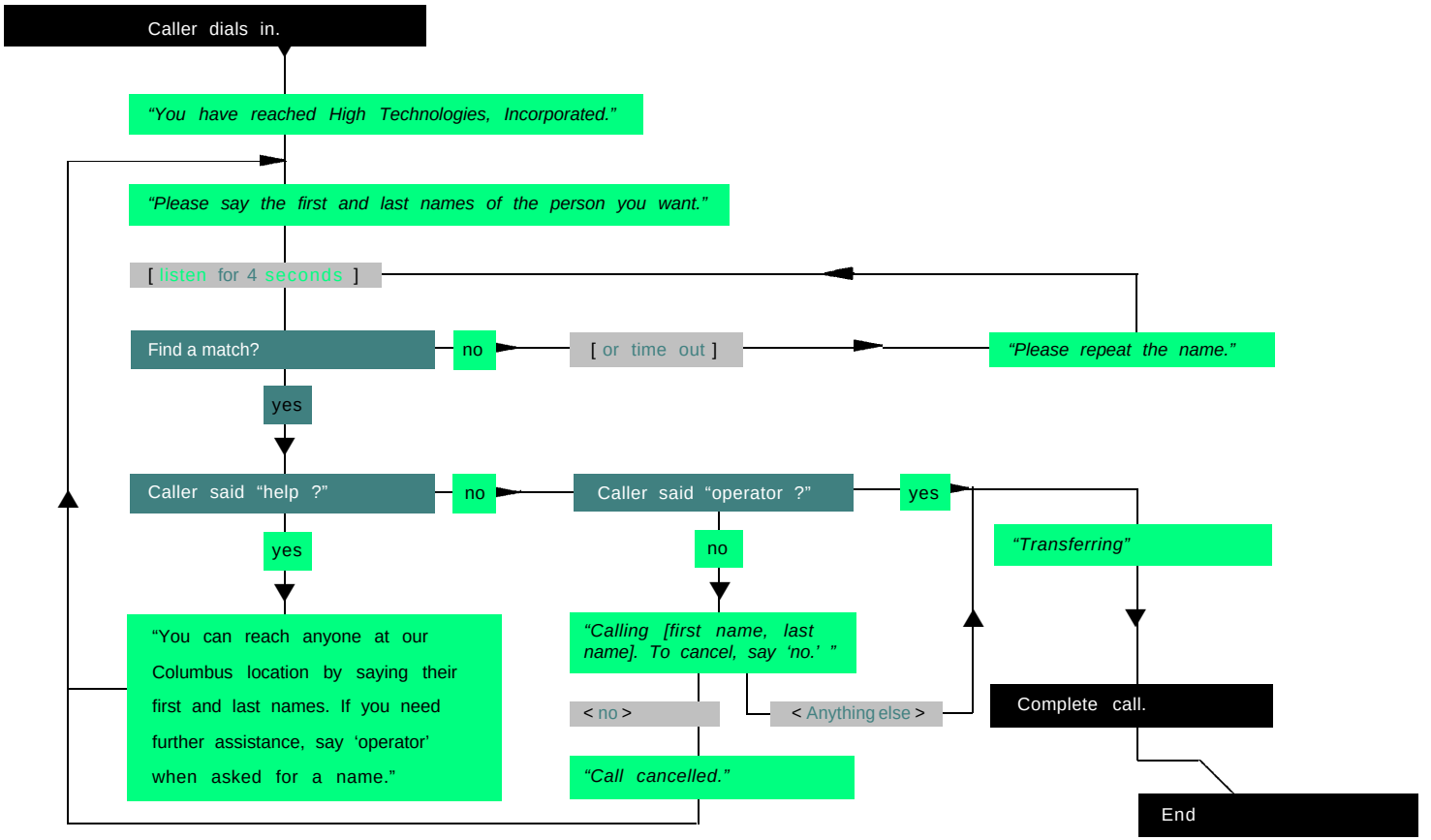
General Design Guidelines

The guidelines in this part apply to applications using any or all of the technologies discussed in this handbook: touch tone, WholeWord Speech Recognition, FlexWord Speech Recognition, Text-to-Speech, and Fax.

Flowcharting and Outlining

When planning your application, it's useful to have a paper-based representation of how you want the application to appear to callers. By systematically planning the flow of the transaction out on paper, you will find it easy to see the structure of the application. Using a paper representation of the application also allows you to plan what to do when errors occur, something that is easy to overlook. If the application design will be done by one person [or team] and the implementation will be done by another, a thorough paper representation is an invaluable communication tool. If not, planning out your application on paper allows you to separate the task of designing the application from the task of implementing it. Often, people who perform both the design and implementation get caught up in implementation, to the detriment of design.

Two popular ways of showing the application on paper are flowcharts and outlines. Flowcharts are more "computer-like," but they are a good, spatially-oriented way of showing the transaction. As shown in the figure to the right, flowcharts use a few simple symbols to represent prompts, announcements, and deci-


Caller dials in.
"You have reached High Technologies, Incorporated."
"Please say the first and last names of the person you want."
[listen for 4 seconds]
Find a match?
no
[or time out]
"Please repeat the name."
yes
Caller said "help ?"
no
Caller said "operator ?"
yes
"Transferring"
yes
no
"You can reach anyone at our Columbus location by saying their first and last names. If you need further assistance, say 'operator' when asked for a name."
"Calling [first name, last name]. To cancel, say 'no.' "
< no >
< Anything else >
Complete call.
"Call cancelled."
End

sion points within the script, and arrows to show the transition from one to another. You need not follow any standard method of using symbols to represent segments of the script, but make sure that those with whom you are communicating can understand the symbols you use.

Another way of showing the application flow on paper is an outline, as shown in the figure below. Outlines are easier for some people because they use only words, not symbols. If you already have a way of communicating the structure of the application that works for you, use it.

Sample outline:

Caller dials in.

"You have reached High Technologies, Incorporated."

1 *"Please say the first and last names of the person you want."*

2 [listen for four seconds; listen for anything on Wordlist 1, which contains all names.]

If match nothing ["silence"];
"Please repeat the name."
Go to **2.**

If match the word "help":
"You can reach anyone at our Columbus location by saying their first and last names. If you need further assistance, say 'operator' when asked for a name."
Go to **1.**

If match the word "operator":
"Transferring."
Transfer the call to the operator's extension.
Finished

If match a name on the wordlist:
"Calling {first name, last name.} To cancel, say 'no.' "
[listen for 2 seconds; listen for < **yes** > or < **no** >].

If match the word "no."
"Call canceled."
Go to 1.

If match the word "yes" **or nothing** ["silence"]:
"Transferring."
Transfer the call to the named person's extension.
Finished.

Whatever method you choose, plan your application carefully before implementing it. Be sure to list all prompts and announcements to be played to callers. Make sure you show what should happen to each point at which you expect caller input. Include what should happen when callers give correct or legal input. Error cases are also important. What happens if callers give incorrect or illegal input? How many times will you re-prompt after hearing illegal input? Will the call be transferred to an attendant? What if callers do not respond? How many times will you re-prompt after hearing no response? A good flowchart or outline will cover all possible cases.

Provide an Attendant

There will always be callers who need extra assistance, especially those calling for the first time. Provide a person who can handle callers having trouble with the voice response application. As a general rule, allow callers to try to enter information only two or three times, then, if they are not successful, transfer the call to an attendant. Allow callers to speak with an attendant by using a key press [usually 0], or by speaking a particular word [like attendant, operator, or help].

Some service providers believe that having a voice response system means that attendants are no longer necessary. Voice response systems can decrease the number of attendants required, but a minimal number of attendants is still necessary. Attendants are costly. However, consider the cost of not having an attendant to deal with confused callers. The risk of alienating callers and driving them to your competition is a risk you may not be able to afford. Allowing your callers to reach a person when necessary will let you better serve your callers.

There are a few exceptions to this suggestion, however. If your service is designed for a very limited audience, if all callers are highly trained, or if customer service is not a priority, attendants may not be necessary.

Menu Prompts and Announcements

Prompt and Announcement Length

Make prompts and announcements as short as possible while remaining polite and informative. Keep the prompts brief, but not abrupt. The more quickly callers can do what they called to do, the more satisfied they will be. For example:

Instead of prompting:
"To listen to information on travel to California, say 1."
You could say:
"For California travel information, say 1."

Instead of prompting:
"If you are calling to register for classes, press 3."
You could say:
"To register for classes, press 3."

There are some trade-offs to keep in mind when considering prompt length. If you expect to have a high proportion of one-time or first-time callers, you might want to provide more information in prompts. This usually makes the prompts longer. This is acceptable if you think your callers will need the extra information. Try to include only helpful information, though, and express it in a short, direct manner.

You also might want to use longer, more informative prompts if you are asking callers to enter information in a way that might be unfamiliar or awkward. For instance, consider a caller entering a date. Most people think of months by name. For a touch-tone transaction, callers would likely be asked to type in the one or two-digit number corresponding to the month. December would be 1,2. With WholeWord SR, December would be represented by the words "one, two," which most callers would usually say as "twelve." Both of these examples require that callers think of the information in a way that might be a bit awkward, and so prompts should be explicit enough to instruct callers what to do. These longer prompts could also include examples, such as:

"Next, enter the month listed on your bill. Say two numbers, such as 'one, two' for December, or 'zero, five' for May."

If you need to include more information, do so, but make it as brief as possible.

FlexWord SR can help make your script easier for callers to use. In the above example, instead of using the number for the month, you could have a FlexWord wordlist containing the names of the twelve months. The caller would be able to speak the name of the month instead of the number representing it.

Menu Prompt Wording

Present the option before the action.

In menu prompts, always list the option first and the action second, rather than the other way around. For example:

Instead of prompting:
"Say 2 to receive the information by fax."
You should say:
"To receive the information by fax, say 2."

This reduces callers' memory load. Prompts listing the action before the option require callers to remember each action until they decide whether or not the option is the one they want. When the option is listed first, callers need only remember the action that pertains to the desired option.

Also, presenting the option before the action means that callers who are familiar with the transaction don't have to remember each option. When they hear the first option and know they don't want it, they don't have to remember it. They can go on to listening to the second option and action. Inexperienced or confused callers, though, may go on listening to the menu, trying to remember all the options and actions, since they're not sure what choice they want.

Consistently word the menu choices.

Use the same basic structure for each option-action pair in a menu prompt. This helps to set caller expectations about what's ahead. For example:

Instead of prompting:
"To hear sales information, press 1.
For services, press 2.
If you would like to speak to an attendant, press 0."

You could say:
"For sales, press 1.
For services, press 2.
For an attendant, press 0."

Make the menu choices dissimilar.

Be sure that menu choices are distinct from each other, so callers will know exactly which option is the one they want. If you find that callers become confused as to which option they should choose, consider rewording the choices.

Avoid jargon and acronyms.

Unless you have reason to assume otherwise, assume that callers do not use the same acronyms and technical terms you do when dealing with information. Avoid acronyms and abbreviations which may be unfamiliar; speak out the complete words. Use terms that are familiar to those you consider your typical callers.

Number of Menu Options

As mentioned in Section 3, too many menu options presented at once may overwhelm callers. In general, offer no more than four or five options on any single menu.

If you have more than four or five options to present, segment them into meaningful groups and present a multi-level menu. For example, if you are offering information on seven state parks, don't list them all out in one menu. Your first menu could classify them into groups, and say:

" *For parks with camping facilities, press **1.***
*For other parks, press **2.*** "

If the caller presses 1:
" *For Cantwell Cliffs, press 1.*
For Old Man's Cave, press 2.
For Stoney Ridge, press 3. "

If the caller presses 2:
" *For Lugano Falls, press 1.*
For Munroe Bluff, press 2. "

For Rialto Beach, press 3.
For Seneca Rocks, press 4."

In certain cases, it might be tolerable to present more than five menu items at once. If all callers are highly trained and practiced, having longer menus may save time [as long as you allow them to dial through or barge in during the menu prompt; see *Using Dial Through and Barge In,* later in this section].

Menu Choice Ordering

To help speed the transaction, present the most likely menu choice first, the second most likely second, and soon. If you do not know the preferred order, make a guess and adjust if necessary during your iterative testing interval [see Section 6].

Numbered Menu Options

When presenting menu options which can be chosen by number, present them in numerical order. Avoid skipping numbers.

If you choose to change the order of the options in your menu to present the more likely choices sooner, be sure to edit or re-record the menu prompt so that the first option presented can be chosen by selecting one, the second by selecting two, and soon.

Keep in mind that frequent or practiced callers, and those with user guides, probably don't listen to all menu prompts. When you change one or more menus in your script, consider adding a special announcement before or immediately after the greeting, such as *"The menu has changed; please listen carefully."* After a period of time [depending on how frequently your callers use the script], remove the special message.

Navigating the Menus

If your menu tree is complicated, or you think that callers will want to get information from many different menus, provide a choice on each menu to allow callers to move around between menus. For example you could add a choice to allow callers to back up to the previous menu, or a choice that would allow callers to start again at the top of the menu tree. For example:

"For parks with camping facilities, press 1.
For other parks, press 2."

< 1 >
"For Cantwell Cliffs, press 1.
For Old Man's Cave, press 2.
For Stony Ridge, press 3.
For the previous menu, press 4."

< 3 >
"Stony Ridge offers 15 tent camping sites, each with a water spigot. Cooking fires are permitted. Toilet and shower facilities are available. Reservations are accepted for April first through November first."

"To make a reservation, press 1.
For directions, press 2.
For the previous menu, press 3.
To re-start at the begninning, press 4."

Although the menu should automatically repeat if the caller does not respond, you might also want to allow callers to have the menu replayed as needed. You could include an option such as *"To repeat the menu, press 9."*

Digit Entry

Validating a Digit Sequence

When you know that a certain set of digit sequences are valid, use whatever techniques you can to validate callers' inputs. This is especially important with spoken input, but can also be done for touch tone. These validation methods help you increase the recognition accuracy for connected digit sequences. Three ways to validate an entry are discussed below:

- Custom grammars
- Checksums
- Database lookups

Using a custom grammar.

You can validate spoken digits by using a custom grammar obtained from AT&T. A grammar is the set of rules by which the recognizer recognizes data. A number of grammars are already provided with Script Builder [a list can be seen in the Choices menu for the Mode field on the Prompt & Collect form]. Custom grammars are special additions to this standard set, designed for your specific needs. For example, if you know that all valid callers will have Ohio telephone numbers, you could request a custom grammar that only validates the area codes 614, 513, 419, and 216. This helps the recognizer make accurate judgments about caller speech.

Using a checksum.

Another way to validate an entry is to take advantage of a checksum. Some numbers [like credit card numbers] have a checksum built in to the structure of the number, so that the number can be checked against a formula to determine if it is valid. If the digit series you are accepting has a built-in checksum, you can use a data interface process [DIP], Script Builder Evaluate statement, or external function to evaluate the checksum. Once the system determines that the entry is legal, it can speak the entry back for confirmation, if necessary [see *Feedback and Confirmation* later in this section].

Using a database lookup.

You can also validate an entry by checking it against a database of valid numbers. For example, when accepting a telephone number, you could try to locate the number in the database of all customer telephone numbers. If the number is found, it can be spoken back to the caller for confirmation. If it is not found, the system can search the database for a telephone number that matches the second best recognition result. If the second choice is found, then the system can speak it back for confirmation.

Variable-Length vs. Constant-Length Digit Sequences

A constant-length digit sequence is a series that is always made up of the same number of digits. A Social Security number is a constant-length digit sequence, since each one consists of nine digits. A variable-length digit sequence can contain different numbers of digits in different situations. A house number in a street address is a variable-length digit sequence.

The VIS recognizes constant-length sequences much more accurately than variable-length sequences. Whenever possible, specify the length of the connected digit sequence to be recognized, to increase recognition accuracy.

If your application must accept a variable-length digit sequence, you may be able increase recognition accuracy by using a two-step entry process. First ask for the number of digits, then ask for the digits. The recognize then knows how many digits to look for in this particular instance. For example:

"How many digits are in the test code?"
< **3** >
"Please say the three-digit code now."

If the digit sequence can have more than nine digits, remember to remind the caller to say the number in digit format ["one" "two"] rather than as a natural number ["twelve"]. See *Using Examples Within a Prompt* later in this section.

Accepting Credit Card Numbers

Different types of credit cards may have different length numbers. If you allow callers to use one of several types of credit cards, you might want to use a menu to prompt them for the type of card, then for the numbers. This lets you know how many numbers to expect from callers, thereby turning a variable-length number into an constant-length number. For example:

"What type of credit card will you be using today? American Express Discover, MasterCard, or Visa?"
< **VISA** >
"What are the first four digits on your Visa card?"

Entering Long Digit Sequences

For entry of sequences longer than four or five digits, consider “chunking” the input. That is, prompt callers to read the series of digits in segments [“chunks”], then confirm each chunk before going onto the next. This is especially useful if the digit sequence cannot be validated with a checksum, custom grammar, or against a database. If the number is already “chunked” on whatever callers read it from, use those chunks. For example:

“Please say the first four digits of your Visa number.”
< **4384** >
“4384. Is that correct?” {pause} “Please say yes or no.”
< **yes** >
“What are the next three digits?”
< **955** >
“955. Is that correct?”
< **yes** >
“And the next three digits.”
< **688** >
“788. Is that correct?”
< **no** >
“Please repeat the three digits.”
< **688** >
“688. Is that correct?”
< **yes** >

“And the final three digits?”
< **833** >
“833. Is that correct?”
< **yes** >

This transaction goes faster if callers are allowed to barge in during the prompt to answer the yes / no questions quickly.

Tone-Paced Digit Entry

WholeWord speech recognition supports the entry of connected digit sequences. That is, callers can say a series of digits in one long breath, without intentional pauses in between. Many callers find this the most natural way to enter a series of digits.

You may have had experience with voice response systems which do not support connected digit entry. In order to simulate connected digit input, other systems use “pacer tones.” When a script uses pacer tones, callers are prompted to speak the first digit after a tone, then wait for another tone before speaking the next digit. This continues until all digits have been read in.

If you prefer the pacer tone-type digit entry method, you can write VIS scripts which use pacer tones [see *Bilingual and Non-U.S. English Scripts* later in this section]. This type of data entry has not been proven to increase the recognition accuracy rate for a digit sequence, though. Pacer tones may even result in lower accuracy, since callers may become confused by the entry method. Connected digit entry remains the recommended VIS digit entry method.

Using Examples Within a Prompt

You may need to prompt callers to enter information in digit format that is not usually expressed in digit format. For example, dates are usually spoken “June eighteenth” or “six eighteen.” However, WholeWord SR only recognizes single digits. June eighteenth must bespoken “six one eight” or “oh six one eight.” This format is awkward for many callers. In order to make this easier, you can include an example in the prompt. For example:

“ Next, enter the month and day of the month you were born. Use two digits for the month, and two digits for the day. For example, for June eighteenth you would say ‘oh six, one eight.’ Please enter the month and day now.”

Use an example whenever you think that inexperienced callers may need some extra help.

Yes / No Questions

How to Ask Yes / No Questions

In situations where you want callers to respond with a touch tone “yes” or “no,” ask them to press 1 for yes or 2 for no. For example:

“ If this is correct, press 1. If not, press 2.”

1 and 2 are recommended, although you may want to use 1 and 0, or Y and N. Even if you prompt callers to press only 1 or 2, you can set up your script to accept additional numbers. For example, 1 or Y for yes and 2, 0, and N for no. Whatever symbols you use to represent yes and no, use the same symbols throughout the script. Don’t confuse callers by using 2 for no in one place and 0 for no in another.

In situations where you want callers to respond with a spoken “yes” or “no,” ask a question that is usually answered with the word “yes” or “no.” Callers will find this natural, and will tend to make fewer mistakes than if you ask them to say a number representing yes or no. For example:

Instead of prompting:
“If this is correct, say 1. If not, say 2.”

You should say:
“Is this correct? Please say yes or no, now.”

Structure your yes / no questions similarly throughout the script, so that callers will know what to expect.

Recognizing “yes” and “no” is best done with WholeWord SR. If you do not plan to purchase the WholeWord package, you can try a FlexWord wordlist consisting of “yes” and “no,” but be aware that the accuracy for the FlexWord version may be lower than that of the WholeWord version.

With FlexWord SR, however, you are not limited to using the words “yes” and “no” as responses to this type of question. You may want to try prompting for “affirmative” and “negative,” or “right” and “wrong” within your application script.[2] Ask the question in such as way as to get callers to say one of the required responses. Be sure to test this with callers, though, to be sure that the interaction is natural enough for them [see Section 6].

See *Using FlexWord Speech Recognition* later in this section for more details.

Pacing Yes / No Questions

With barge in: After asking a yes / no question, pause in order to give callers time to respond, then present the possible answers. The prompt will stop playing as soon as the recognizer detects a “yes” or “no.” For example:

“You said 64587. Is this correct?”
{~1.5 second pause}
“Please say yes or no.”

Using the Prompt & Collect action, ask the question, play a series of silence phrases, then present the options. Here is an example of how part of your Script Builder code will look if you ask the caller for five digits, then confirm the entry within the Prompt & Collect action:

```
Prompt & Collection
      Prompt
            Speak with Interrupt
                  Phrase: “Please enter your 5-digit cust. no.”
      Input
            Mode: US_DIG
            Min Number of Digits: 05
            Max Number of Digits: 05
      Checklist
            Case: “Input OK”
                  Speak with Interrupt
                        Phrase: “You said”
```

```
                        Field: $CI_VALUE As C
                        Phrase: “Is this correct?”
                        Phrase: “sil.500”
                        Phrase: “sil.500”
                        Phrase: “sil.500”
                        Phrase: “Pls say yes or no.”
                    Confirm
                Case: “Initial Timeout”
                    Reprompt
                Case: “Too Few Digits”
                    Reprompt
                Case: “No More Tries”
                    Quit
        End Prompt & Collect
```

Without barge in: If barge in is not allowed, you have two alternatives:

1 Ask the question, but do not list the options. For example:
“You *said 645987. Is this correct?”*

This works especially well for experienced callers. If you find that callers become confused, try the second alternative.

2 List the options, but do not leave an extended pause between the question and the options. For example:
“Would *you like to place another order? Please say yes or no.”*

Pacing the Script

When Callers Must Wait

If you know callers will face a brief wait, always let them know. For example, you could say

"Please *wait while your request is processed."*

If you know that the caller may have to wait for longer than a few seconds for the script to continue [because of a database call or host connection, for example], it's a good idea to fill the gap during the wait. You could record a long phrase,[3] perhaps music, and play it to the caller while the information is being located. Or, play a different phrase every seven to ten seconds to let callers know that work is being done. In order to do this, you must use an external function. See the example of this in the TFLUSH section of *Intuity CONVERSANT VIS Version 5.0 Application Development, 585-310-227,* paying special attention to the description of the wait indicator.

Allowing Time for Caller Responses

Pace the interaction so that callers have time to enter the required information and listen to the prompts and announcements. You can determine how comfortable callers are with the pace of the interaction by testing the script as described in Section 6.

Don't let the script respond to caller actions too quickly. After callers are asked to make an entry, play a speech phrase consisting of about a half second of silence before the next prompt or error message. Since many telephones have the keypad built in to the receiver, this pause leaves time for callers to lift the receiver back to their ears after pressing the keys. If your calling population includes senior citizens, this pause can give them a little more time to react.

Adjustable Pacing

You can set up your script to allow pauses to be adjusted quickly and easily. Let's say you want to have a pause as described in the previous paragraph, but you want to be able to tune the length of the pause during your script testing phase. In the beginning of your script, define a field called "entry_pause." Assign to "entry_pause" the speech phrase number associated with one of the silence phrases.[4] For example:

```
Set Field Value
     Field: entry_pause = 1074
```

Then, throughout the script, speak the field "entry_pause" before the prompts that follow caller entries. for example:

```
Announce
     Speak without Interrupt
          Field: entry_pause As NX
          Phrase: "Next, please enter your 5-digit PIN."
```

If you find during testing that the script responds too quickly, you need only go to where you've set the field value and use a longer silence phrase. Thus, you can make many changes throughout the script by changing only one line. You can do the same thing for other pauses throughout your script, such as pauses between menu items.

Feedback and Confirmation

General Feedback Guidance

Feedback announcements should be concise and unambiguous. If you want to use “please” and “thank you,” use them conservatively so that callers can complete the transaction as quickly and efficiently as possible.

Avoid referring to the system as “I” or “we.” If callers get the impression that the system is a person, they are likely to use more words to make their answers more friendly, which makes speech recognition more difficult. Also, callers might think that you are trying to fool them into thinking that the computer is a person, and they may be resentful. For example:

Use:
“Sorry. Please repeat.”

Rather than:
“Sorry, I did not understand you. Please repeat.”

If a request will take more than two or three seconds to fill, assure callers that work is being done. For example:

“Looking for your records. Please wait.”
or
“Please wait while your information is located.”

Confirmation Feedback

When callers enter information, you must choose the type of feed back to give. There are times where you should read the entry back and allow the caller to confirm it. Other times, feedback is necessary, but confirmation is not.

1 *When you want confirmation.*

In cases where a recognition error or caller touch-tone error would have a big impact [like credit card numbers or call destinations], you should give callers an opportunity to confirm the entry. To do so, repeat the entry and prompt callers to say “yes” or “no” to confirm it.

“What department would you like?”
< **Hardware** >
“Calling hardware. Yes or no?”

“What is your customer number?”
< **432886** >
“Customer Number 432886. Is this correct?”
{pause}
“Please say yes or no, now.”

“Please enter your customer number.”
< **432886** >
“432886. If this is correct, press 1. If not, press 2.”

In Script Builder, you can include this confirm step in two different ways. You could use the "built-in" Confirmation when defining the original Prompt & Collect action, or you could use a separate Prompt & Collect statement to do the confirmation. The Confirm within the Prompt & Collect action works well for U.S. English WholeWord SR and touch tone.[5] However, in Version 5.0, it works for FlexWord SR *only if you have both Text-to-Speech and U.S. English WholeWord SR installed on the same VIS.* If you are using FlexWord SR and do not have both of these packages, you must confirm with a separate Prompt & Collect statement. If using a WholeWord language other than U.S. English, you must confirm with a separate Prompt & Collect statement.

Confirming an entry within Script Builder is summarized in this illustration:

DATA ENTERED WITH	HOW TO CONFIRM
Touch tone	Use Confirm within Prompt & Collect action
WholeWord SR U.S. English	Use Confirm within Prompt & Collect action
WholeWord SR Canadian French or Mexican Spanish	Use separate confirm statement
FlexWord SR [with U.S. English WholeWord SR and Text-to-Speech]	Use Confirm within Prompt & Collect action
FlexWord SR [with U.S. English WholeWord SR only]	Use separate Confirm statement and recorded phrases for all words
FlexWord SR [with Text-to-Speech only]	Use separate Confirm statement and FlexWord yes / no wordlist
FlexWord SR [with no other packages]	Need separate Confirm statement, recorded phrases for all words, and FlexWord yes / no wordlist

Note: In any of the above conditions where you do not have U. S. English WholeWord SR, you could opt to have the caller confirm with a touch tone.

Confirmation in Non-U.S. English Languages

The Confirm within the Prompt & Collect action only works when you are using touch-tone input or WholeWord SR with U.S. English. WholeWord packages using other languages must use a separate Prompt & Collect statement to confirm an entry. A pre-recorded speech phrase for each digit and yes / no is necessary.

2 *When you don't need confirmation.*

In cases where a recognition error or a caller touch-tone error has little impact [that is, if it's fast and easy for callers to return to the point in the script where they really wanted to go], you may choose to not allow callers to confirm an entry. Skipping the confirmation of non-essential entries can also speed the transaction.

Even if you do not confirm an entry, it's still important to let callers know how the recognize interpreted the input. To do so, include context-relevant information in the next announcement you play.
For example:

"What month?"
< **August** >
"Artists performing at the Palace Theater for August include..."

Error Handling

Handling Caller Errors

Caller errors occur when callers enter information that the script considers incorrect, or when callers enter nothing at all. When callers make errors, provide them with informative error messages; that is, error messages that give callers a good idea as to what went wrong and how to correct it. To assure that callers hear error messages, do not allow barge in or dial through during at least the first part of error messages.

Make sure your script design accounts for all types of errors callers can make.

Allow callers no more than two or three tries to enter information. After the first error [and second, if allowing three], the script should speak out a prompt which contains more information, so that callers can get a better idea of what is required. After the last try, tell callers to hold for an attendant, then transfer the call.

To speak out an informative error message using Script Builder's Prompt & Collect action, use either the Try Again or Reprompt action. Be sure to specify the informative error message in the associated Voice Response field. Specify an appropriate error message for each type of error that can occur [Initial Timeout, Too Few Digits, Not on List]. For example:

If the first prompt is:
"What time would you like your wake-up call?"
and the caller does nothing [initial timeout]

A more informative prompt could be:
"Please enter the time you would like your wake-up call. Use the telephone keypad to enter two digits for the hour and two digits for the minutes."

If the first prompt is:
"Enter the date of the lottery."
and the caller has pressed some touch tones, but not enough [too few digits]

A more informative prompt could be:
"Please enter the date of the lottery for which you want to hear winning numbers. Use the telephone keypad to enter two digits for the month, and two digits for the day. If a month or day has only one digits, use a zero before the digit."

If the first prompt is:
"What catalog will you be ordering from?"
and the caller makes an invalid entry or the recognizer does not recognize a spoken entry as valid [not on list]

A more informative prompt could be:
"Sorry. You can order from the Hot Sauce Heaven catalog or the Caffeine Corner catalog. Please speak the name of the catalog."

Error Message Wording

Avoid using negative or accusatory wording when indicating an error. Phrases like "invalid," "bad," "wrong," or "incorrect" can make callers feel threatened. "Sorry" is a concise way to convey that the input was not recognized or was incorrect, and "Please repeat" or "Please re-enter" are brief ways to ask callers to act again.

Handling Speech Recognition Errors

Recognition errors occur when the recognizer cannot match the caller's speech to a phrase it is prepared to recognize. The caller is not necessarily at fault, so remember to avoid using negative or accusatory wording here.

Choose one concise phrase and use it consistently within a single application. For example:

Good:	*"Sorry, Please repeat."*
Avoid:	*"We're sorry, but your speech cannot be understood. Please repeat the category you want."*

When using a small FlexWord wordlist [five or fewer items], re-prompt with a short "sorry" phrase, then give a list of the valid options. For example:

"Sorry. Please say checking, savings, interest rates, loan rates or operator, now."

When using a large custom wordlist, include some helpful information in the re-prompt. For example:

"Department, please."
< **excuse me?** >
"Sorry. Please say the name of the department you want, or speak with an by saying 'attendant.'"

Combination Touch Tone and Whole Word SR Scripts

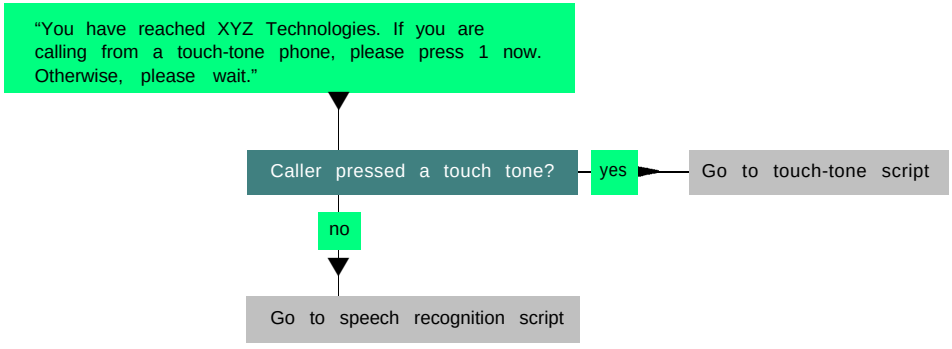
Choosing an Approach

In order to handle callers who do not have touch-tone telephones, you can use WholeWord speech recognition to allow rotary callers to speak digits and yes / no. There are two different approaches you could take to setting up the application:

Write two separate, nearly identical scripts.

This frees you to use prompts that are best for touch-tone entry in one script, and those that are best for speech recognition entry in the other.

In many cases, touch-tone entry is more accurate and less costly than speech recognition. Therefore, you probably want callers who have touch-tone telephones to use their keypads to interact with the script. To encourage this, start your script with a “dial 1“ prompt. A dial 1 prompt gives callers a chance to indicate that they have a touch-tone telephone, and helps speed the transaction. A dial 1 prompt might look like this:



The Script Builder code to support the above dial 1 prompt is shown below.

```
start:
Answer Phone
Prompt & Collect
      Prompt
            Speak with Interrupt
                  Phrase: “You have reached XYZ Technologies.”
                  Phrase: “If you are calling from a touch-tone phone...”
      Input
            Max Number of Digits: 01
      Checklist
            Case: “Input OK”
                  Continue
            Case: “Initial Timeout”
                  Goto  sr_script
            Case: “Too Few Digits”
                  Goto  sr_script
            Case: “No More Tries”
                  Quit
End Prompt & Collect

touchtone_script:
#put touch-tone script here

sr_script:
#put speech recognition script here
```

Use a single script flow.
This requires that every prompt be appropriate for both entry methods.

Caution: Only do this if you feel your system resources can handle the load, since many callers with touch-tone telephones may choose to use speech recognition instead. Remember also that recognition accuracy is lower for spoken input than for touch-tone input.

If you are using a single script flow to handle both touch-tone and non-touch-tone callers, a dial 1 prompt is not necessary. When using a single script, you should structure your prompts so that they give callers the choice of using the keypad or speaking their responses.

At the beginning of the script, tell callers that entries may be made with either the keypad or by speaking. or example, the script might say:

"Welcome to XYZ Technologies. During this call, you can use your telephone tough tone keypad to make entries, or you can simply speak your responses."

Then, make sure that the prompts you present are appropriate for either kind of input. There are two ways to do this:

1 Tell callers to "enter" the information. For example:
"Enter *the zip code of the restaurant you visited."*

2 Present the prompt as a question. For example:
"What is the zip code of the restaurant you visited?"

Either style is appropriate. In order to do this in Script Builder, you can set up the script just like you would when expecting WholeWord input. For digits, the script automatically accepts touch tones in place of the spoken input.

For yes and no, you must tell the script what touch-tone keys are associated with the words “yes” and “no,” and specify the action associated with each. This is done on the third page of the Prompt & Collect form. You also must tell callers what to press instead of saying “yes” or “no,” in case they prefer to key in their response, although you can wait until the re-prompt to do so. A typical set of prompts to do this might be:

“Would you like to leave us a message regarding the service agreement?”
{pause}
“Please enter yes or no.”
re-prompt: *“Please say yes or no. Or, for yes, press 1. For no, press 2.”*

The Script Builder code to support the above interaction looks like this:

```
Prompt & Collect
      Prompt
            Speak with Interrupt
                  Phrase: “Would you like to leave a message. . .?”
                  Phrase: “sil.500”
                  Phrase: “Please enter yes or no.”
      Input
            Mode: US_YN
            Max Number of Digits: 01
```

```
        Checklist
            Case: “Y”
                  Continue
            Case: “N”
                  Continue
            Case: “1”
                  Continue
            Case: “2”
                  Continue
            Case: “Not on list”
                  Speak with Interrupt
                       Phrase: “Say y or n. yes, 1. no, 2.”
                  Try Again
            Case: “Initial Timeout”
                  Speak with Interrupt
                       Phrase: “Say y or n. yes, 1. no, 2.”
                  Try Again
            Case: “Too few digits”
                  Speak with Interrupt
                       Phrase: “Say y or n. yes, 1. no, 2.”
                  Try Again
            Case: “No More Tries”
                  Quit
End Prompt & Collect
```

Note: It's important to use "Try Again" instead of "Reprompt" here. If you use "Reprompt," the script will repeat the *"Would you like to leave a message. . ."* prompt after the *"Please* say yes *or no. Or for yes, press 1." For no, press 2"* prompt, which would be awkward.

Using Touch Tone When Recognition Fails

If you are using WholeWord speech recognition, you may be able to increase the number of successful interactions by encouraging callers to use touch tones when spoken entries have not been recognized correctly. This is especially appropriate for long digit sequences.

If the recognizer must recognize long digit sequences, you can have the re-prompt instruct callers to key in the number if they have a touch-tone telephone. Since a script written to accept spoken input of digits and will also accept touch-tone input, the implementation is not complicated.

A typical prompt set might look like this:

"Please enter the 6-digit code for the fax information you want."
< **135683** >
"You entered 135688. Is this correct?"
< **No** >
"Please re-enter the code. If you have a touch-tone telephone, you may want to enter the code using your keypad, instead."

Modifying a Touch-Tone-Only Script to Include Speech Recognition

If you currently have a touch-tone script you'd like to update to support speech recognition in addition to or in place of touch-tone input, it's unwise to merely change the prompts from using the word "press" to using the word "say." A script that has been optimized for touch tone only may require substantial revisions in order to support the addition of speech recognition. For example, if your script prompts *"Press 1 for yes, or 2 for no,"* it does not make sense to change it to *"Say 1 for yes, or 2 for no."* It makes better sense to say *"Say yes or no."*

There are other things you must watch out for, too. For instance, "initial timeout" in the Prompt & Collect action means different things for touch tone and for speech recognition. For touch tone, initial timeout represents the number of seconds the script will wait for the first touch tone from the caller. For WholeWord SR and FlexWord SR, initial timeout represents the number of seconds from the end of the prompt until the script will demand a result from the recognize. This is the case whether barge in is on or off.

When adding speech recognition, examine the touch-tone script to see what the prompts say, and what action is expected from callers. Think about the capabilities of the speech recognition package[s] you will be using, and where they might fit in to the application script.

If you plan to use FlexWord SR, pay special attention to places where numbers are associated with choices. For example, a touch-tone prompt that says

> *"To listen to homework, press 1. To leave a message for the teacher, press 2. To get a different class, press 3."*

can be restructured to take advantage of custom wordlists. You could change the prompt to say something like:

"Would you like to listen to homework, leave a message for the teacher, or choose a differenct class?"

Your wordlist could contain these words:

WORD	MAPS TO THIS OPTION
listen to homework	"listen to homework" option
homework	"listen to homework" option
listen	"listen to homework" option
leave a message for the teacher	"leave a message for the teacher" option
leave a message	"leave a message for the teacher" option
teacher	"leave a message for the teacher" option
choose a different class	"choose a different class" option
different class	"choose a different class" option
different	"choose a different class" option
class	"choose a different class" option

Refer to *Using FlexWord Speech Recognition* later in this section for more information.

If you plan to use WholeWord SR, your scripts will be able to recognize yes / no and connected digits. In places where the touch-tone script prompted for digit entry [especially long sequences], you'll want to introduce custom grammars, database checking, checksums, or chunking to get the highest recognition accuracy possible [see the section on Digit Entry earlier in this section].

Using Dial Through and Barge In

When expecting touch-tone input, you can allow callers to respond while a prompt is playing. This is called *dial through* [also known as "talk off"]. Similarly, for WholeWord SR, you can allow callers to speak while a prompt is playing. This is called *barge in.* For each Prompt & Collect action, barge in and / or dial through can either be on [enabled] or off [disabled].

The barge in / dial through capability is only available with WholeWord speech recognition and touch tone; FlexWord SR does not support barge in Version 5.0. Callers must always listen to the complete FlexWord prompt before responding.

Experienced callers appreciate being able to shorten the transaction by not listening completely to each prompt. However, for WholeWord SR, barge in can take up too much of the speech recognition resource. If resources are a problem, you might choose to forbid barge in until you can expand your speech recognition resources. See *Intuity CONVERSANT VIS Version 5.0 System Description,* 585-310-225 for more information about allocating resources.

The wording, length, and structure of prompts can either encourage or discourage callers from responding during the prompt. You want to ensure that the caller's behavior matches how you have set up the prompt–either barge in / dial through is enabled or disabled.

Encouraging Caller Dial Through / Barge In

If you have the barge in / dial through capability turned on for a prompt, you should encourage callers to respond when they are ready, even during the prompt. This saves time for experienced callers who already know what to say or do. To encourage barge in or dial through for a menu, you could leave a pause after each option, like this:

"For sales, say 1. {1.5 second pause}
For service, say 2. {1.5 second pause}
For an attendant, say zero."

Using the Prompt & Collect action, play an option, then play a series of silence phrases equivalent to 1.5 seconds. Silence is not necessary after the final option.

To encourage barge in for a yes / no prompt, see *Timing of Yes / No Prompts* earlier in this section.

Discouraging Caller Dial Through / Barge In

If you have the barge in / dial through capability turned off for a prompt [or if you are using FlexWord SR at this prompt], you want callers to respond after the prompt is finished. This ensures that the recognizer will hear everything callers say or press. To discourage barge in / dial through for a menu, do not leave a pronounced pause between options. For example:

"For sales, say 1. For service, say 2. For an attendant, say zero."

You can also discourage barge in for a yes / no prompt. See *Timing of Yes / No Prompts* earlier in this section for more information.

When barge in is turned off or unavailable, make sure that your recorded prompts have no silence at the end. If there is silence at the end of a prompt, callers are led to believe that the prompt is over and will begin speaking. When this happens, the recognizer will miss part of the utterance even though callers think they waited until the end of the prompt to respond.

When using FlexWord SR in Version 5.0, always discourage barge in. You can tell callers at the begninning of the application to wait until the VIS stops talking before responding. Use short, direct prompts so callers won't get bored and attempt to answer too quickly.

Dial Through / Barge In for Error Messages

For error messages, it is wise to require callers to listen to at least part of each error message, so that the interaction stops and the mistake can be corrected. For WholeWord SR and touch-tone input, turn off barge in / dial through during the first part of each error message. For example:

[turn dial through or barge in off]
"Sorry."
[turn dial through or barge in on]
"Try again."

Consistency in Using Dial Through / Barge In

Since callers expect and appreciate consistency, it's a good idea to consistently apply barge in and / or dial through in your applications. Allowing callers to barge in during some prompts and not others could cause confusion. Strive to allow barge in always or never in any particular application.

This presents a challenge when you use FlexWord SR in the same script as WholeWord SR or touch tone or both. In the interest of consistency, applications that take FlexWord input, even at one prompt, should never allow barge in because FlexWord SR does not support it. You may find, however, that your callers appreciate the ability to barge in during some prompts, and may be able to tolerate some inconsistency. During your iterative design phase [see Section 6], determine what will work for your calling population, and design your scripts accordingly.

Using Flex Word Speech Recognition

As mentioned in Section 2, the FlexWord package recognizes callers speaking words from a vocabulary that you define, and therefore is specifically tailored to your applications. Allowing callers to say the option they want instead of saying a number assigned to the option can make the interaction much more natural and easy to use. Here are some guidelines on getting the most out of FlexWord SR.

Deciding on a FlexWord Vocabulary

Since FlexWord SR gives you the freedom to specify your own custom vocabularies, you have the power and responsibility of making sure your application is easy to use. How you set up your wordlists can contribute greatly to the success of your application.

FlexWord vocabulary items can be single words or multiple-word phrases, which are all referred to as “words.” The words must be divided into groups called wordlists. A wordlist includes all the words that can be spoken at a particular prompt. Each wordlist can now contain up to 500 words, and you can now have up to 200 wordlists [Version 4.0 supported up to 100 wordlists with up to 200 words on each].[6] FlexWord SR can support choices from small menus [with a wordlist consisting of different clothing sizes] up through large menus [with a wordlist consisting of the names of all 495 people in your office].

Each application has its own vocabulary, or set of wordlists. Although only one Version 5.0 FlexWord package can be installed on a VIS at a time, a single FlexWord package can support many applications, each with its own vocabulary. Application scripts may share wordlists with one another.

At any Prompt & Collect action, only one wordlist can be active. That means that only one wordlist [with up to 500 words] can be under consideration by the recognizer at any one time. The words in each wordlist must be distinct enough from each other to allow the recognize to work properly. It doesn't matter if two words are very similar, as long as they're on different wordlists.

For examples of wordlists, see pages 18 and 94.

What is a good word or phrase to use in a wordlist? Consider a number of attributes:

Length

Words with more than one syllable are recognized more reliably than shorter words. For example, the phrase "add entry" would be better to use than the word "add." A phrase can be fairly long. Even a phrase as long as "The quick brown fox jumped over the lazy dog" can be supported. This should accommodate all your needs. Note, though, that the longer the phrase is, the more chance there is of callers forgetting it or making mistakes while speaking.

Sound

Try to use words that don't sound very much alike. To the recognizer, vowels are slightly more important than consonants. Avoid using words with similar vowel sounds. Words that differ in only one or two consonants are difficult for the recognizer to distinguish between. For example, the words "connect" and "comment" sound similar to the recognizer, even though they don't sound very similar to people. The same goes for "misdirect" and "disconnect."

Meaning

Choose words and phrases that would occur naturally to your typical caller. One way to determine this is to survey some representative callers and ask what words or phrases they would use for actions in your script. Don't rely on just your knowledge and intuitions, since your impressions may be very different than those of the callers.

Synonyms

If you find that callers use more than one word for the same thing, you may want to add both words to your word list if there is room. See *Modifying a Touch-Tone Only Script to Include Speech Recognition* earlier in this section for an example of using more than one phrase to correspond to a single menu choice. Take care that the synonyms you use are not confused by the recognize with other words already in your wordlist, unless you want to use that confusion for better recognition. For example, in a name dialer, you might want to add different forms of some names, like using both "Joe Tanner" and "Joseph Tanner" to represent the same person.

Throw Aways

If you find that callers tend to say something that is not acceptable, you can add that word to the wordlist. In your application, if that word is recognized, you can handle it as a throw-away and reprompt. For example, if the wordlist contains sizes and the valid sizes are "small," "medium," and "large," you could add "petite" to the wordlist. If the recognizer recognizes that a caller says "petite," the application can handle that case by saying *"Sorry. We do not offer petite sizes. Please choose small, medium, or large."*

Caution: If you want to recognize digits in the same application as your FlexWord vocabulary, you must have the WholeWord package installed on your VIS. Do not attempt to build FlexWord vocabularies containing digits. WholeWord models have been painstakingly refined, and FlewWord models for the same words will result in lower accuracy. Also, recognizing more than one word or phrase in a row [which would be required to recognize connected digits], cannot be done with FlexWord SR in Version 5.0.

If you want to recognize “yes” and “no,” this is also best done with WholeWord SR. If you do not plan to purchase WholeWord SR, you can try a FlexWord wordlist consisting of “yes” and “no,” but be aware that the accuracy for the FlexWord version will be lower than that of the WholeWord version. You might want to try synonyms for yes and no, as described in *Yes/ No Questions* earlier in this section.

Building a FlexWord Vocabulary

There are two ways to build your FlexWord vocabularies. The Intuity CONVERSANT VIS FlexWord Toolkit allows you to make your own wordlists quickly and easily. The optional Toolkit gives you a point and click, graphical environment in which to define lists of words that callers can say. You can use the Toolkit to add, delete, or change words on new or existing wordlists. The Toolkit also allows you to hear the expected pronunciation of each word, and lets you to adjust it if necessary. You can use these wordlists in your applications immediately after you define them.

If you prefer to have someone else build your vocabularies, the AT&T custom vocabulary service can do this for you for a fee. Consider the size of your vocabularies and how many FlexWord application scripts

you plan to have when deciding to purchase the Toolkit or use the custom vocabulary service. Contact your AT&T representative for more information.

Prompting for FlexWord Input

Version 5.0 FlexWord SR does nonsupport barge in. You should structure your prompts to encourage callers to wait until the end of the prompt before speaking. See *Using Dial Through and Barge In* earlier in this section for suggested prompt structures.

Segmenting Long Wordlists

If you want to recognize more than 500 items at the same time, you may be able to categorize the items, breaking them down into smaller wordlists. For example, you may want an application that allows callers to speak the name of the person they want to talk to, and your company has 1000 employees. A wordlist can hold no more than 500 entries. If no department has more than 500 employees, you could have a separate wordlist for each department, and a wordlist containing the department names. Callers could be prompted in two stages: **[1]** say the department name, and **[2]** say the name of the desired person. Categorizing the items may not make sense for all applications, but consider doing so if you want to recognize more than 500 items at a time.

Using Text-to-Speech

As mentioned in Section 2, this advanced speech technology allows your application to convert text to spoken words. This ability is especially useful when it is not practical to pre-record the information to be read [if it is lengthy, changes often, or comes from a database]. Text-to-Speech is also useful when prototyping application scripts. Here are some guidelines on using Text-to-Speech.

Using Text-to-Speech Prompts and Announcements

Dial Through and Barge In

When Text-to-Speech prompts are used with touch-tone input, callers can and interrupt the prompt just as they can when the prompt has been pre-recorded. Callers can also interrupt Text-to-Speech prompts with WholeWord input. However, when using Text-to-Speech prompts with FlexWord input, barge in is not available. Callers cannot barge in during a Text-to-Speech prompt with FlexWord voice input. Therefore, when accepting FlexWord input, structure your Text-to-Speech prompts so that they encourage callers to wait until the end before responding [see *Using Dial Through and Barge In* earlier in this section].

The same goes for announcements. Callers cannot stop a Text-to-Speech announcement with FlexWord input, only with WholeWord or touch-tone input. Be aware of this, and structure the script so that callers won't get stuck listening to a long announcement with no way of stopping it. Consider prompting callers to press any touch-tone key when they've heard all they want to hear of an announcement.

Use Text-to-Speech in your scripts to provide your callers with up-to-date or lengthy information.

If you have problems dialing through a prompt or announcement with touch tone, make sure that type ahead is enabled. When using type ahead, remember to flush all touch tones from the buffer whenever the script plays an error message.

Using Both Text-to-Speech and Pre-Recorded Speech Prompts

A single script can speak prompts and announcements in pre-recorded speech only, Text-to-Speech only, or a combination of the two. You may experiment with scripts that use only Text-to-Speech output, although novice callers find this harder to understand than pre-recorded speech. Some script designers choose to speak out unchanging information [including most prompts] with pre-recorded speech, unless

the number of phrases makes pre-recording unmanageable, and only use Text-to-Speech to speak out information that is long, changes often, or comes from a database, and is therefore difficult or impossible to pre-record. The best way to decide between these two alternatives is to prototype and test both [see *Testing Text-to-speech Scripts* later in this section].

Here is an example of how you might use both Text-to-Speech prompts and pre-recorded prompts in a script. Let's say you might have a wordlist consisting of 350 names. Instead of pre-recording all 350 names, you could use Text-to-Speech to speak the recognized name back for verification, and then use pre-recorded speech for the rest of the prompts. For example:

"Name, please."	[pre-recorded]
< **Dana Frost** >	
"Dana Frost"	[Text-to-Speech]
"Yes or no?"	[pre-recorded]

The best way to decide between using all Text-to-Speech output or a combination of Text-to-Speech output and pre-recorded speech is to prototype and test both with representative callers [see *Testing Text-to-Speech Scripts* later in this section].

Using Text-to-Speech for Prototyping Scripts

When you are first putting your script together, it's good to be able to test it out for yourself. Without Text-to-Speech, you must record each speech phrase in order to test the script. To get around this, you can use Text-to-Speech to speak the phrases instead of recording them in your own voice. This saves you time, so can help shorten the prototyping cycle. When you are sure of the wording you want, you can then have the phrases recorded by a person in order to do testing with representative callers.

Getting the Most Out of Text-to-Speech

Acclimating Callers to Text-to-Speech Output

You may find that callers may need to get used to the machine-like accent of Text-to-Speech output in order to understand it well. To facilitate this, you may want to consider speaking out some less important information before speaking out the information callers need. For instance, your script might use Text-to-Speech to say *"Thank you. Please wait while the information is located. The XYZ Technologies distributor closest to you is. . ."* before giving the name and address of the store. Giving callers a chance to tune in to the rhythm and intonation of Text-to-Speech will help them better understand it.

Because Text-to-Speech output is sometimes difficult to understand, consider giving callers the option of having the information repeated, or even spelled out, if necessary. Spelling is especially useful with names, as Text-to-Speech may not always pronounce names as the caller would.

Structuring the Input

Keep in mind that the Text-to-Speech feature works primarily as a "reading machine." It operates under the assumption that the information it is reading is structured in standard English sentences [using punctuation, capitalization, subject, object, and verb]. In order to make Text-to-Speech output sound most natural, use good grammar, complete sentences, and punctuation in the text to be spoken. If you are reading lengthy information, chances are the text is already in complete sentences.

What if the information you want to speak is not written in complete sentences? Since data fields cannot be punctuated, you may be able to control the output by varying the speaking rate and pauses between the information [see later in this section]. Since Text-to-Speech cannot find standard English structure in unstructured input, the output will not be as intelligible as output for full sentences.

Data Types

When speaking out a proper name, address, telephone number, or fraction, use “class detectors” within your script to specify the type of data to be spoken. Class detectors help Text-to-Speech decide which abbreviations are appropriate for the data. The class detectors are documented in *Intuity CONVERSANT VIS Version 5.0 Speech Development,* 585-310-228.

Be extremely careful when using class detectors, however. Turn them off as soon as you have finished speaking that class of information. Always leave one space before and after each class detector, and type them exactly as they appear in the documentation. Remember to use lower-case letters only. Here is an example of a script speaking out an address from database fields:

```
Announce
      Speak with Interrupt
            Text:     “ \!nar The address of our nearest location is ”
            Phrase: “sil.500”
            Field:    store_address as A
            Phrase:   “sil.500”
            Field:    store_city as A
            Phrase: “sil.500”
            Field:    store_state as A
            Phrase: “sil.500”
            Text:     “ \!nac ”
```

Abbreviations

Text-to-Speech can recognize and speak out most standard English abbreviations as full words. For example, “Blvd.” is spoken as “Boulevard.” You may come across an abbreviation that Text-to-Speech does not know, though, and therefore cannot pronounce. Since the full list of abbreviations is not documented at this time, make sure that you test your application extensively [see Section 6] so you can find any problems that might occur with abbreviations. To handle an abbreviation that Text-to-Speech does not know, substitute the entire word, if possible, for the abbreviation so that it can be spoken correctly.

If you are reading text from a database over which you have no control, you may only be able to modify abbreviations by using a data interface process [DIP].

Speaking Rate and Pauses

To help callers understand Text-to-Speech output, you can vary the Text-to-Speech speaking rate where necessary by using escape sequences within the script. For example, you might want to slow down the speech when you read an address The escape sequences are documented in *Intuity CONVERSANT VIS. Version* 5.0 *Speech Development,* 585-310-228.

Take care when using escape sequences. Always leave one space before and after each escape sequence, and type the sequences exactly as they appear in the documentation. Be sure to turn off speaking rate escape sequences as soon as you are finished speaking at that rate. Here is an example of a script changing speaking rate:

```
Announce
        Speak with Interrupt
                Text: “Your confirmation number is \!r4 743211 \!r2.
                Text:  Please call us if you do not receive your
                Text:  tickets \!r4 in 7 days \!r2 .”
```

You can also control the pace of the Text-to-Speech output by inserting pauses. If you are speaking text which is under your control, the easiest way to do this is by using punctuation within the words to be read. Remember to punctuate exactly like you would in a sentence [for example, don’t leave a space before a period or a comma]. You can also use the documented escape sequences to insert pauses.

When speaking out a large block of text, you may hear a pause where you don’t want a pause to be. First, check to see if there is any stray punctuation causing the pause. If not, you can insert a short recorded silence phrase before the sentence during which you heard the pause. This should eliminate the misplaced pause. If the text block is from a remote database, however, this may not be possible. You may need to read in the text to different buffers in order to take care of this problem.

Typographical Errors

Text-to-Speech pronounces exactly what it is written, so typographical errors can cause misproununcia-tions. In order to make Text-to-Speech output as understandable as possible, take care to minimize typo-graphical errors.

Capital Letters

When you want Text-to-Speech to spell something out, make sure the input is in capital letters. For example, you probably want the abbreviation for the Environmental Protection Agency spelled out as “E, P, A“ instead of being pronounced as “eppah.”

If you are speaking input from a remote database and the data is in all capital letters, remember that it will be spelled out. If you want it to be pronounced as words, you must use a data interface process [DIP] to convert the information to upper and lower case.

Pronunciation

Text-to-Speech may mispronounce words, especially some names. It relies on built-in rules, but cannot account for all exceptions. If Text-to-Speech mispronounces a word, you can use phonetic spelling to correct the pronunciation. For example, Text-to-Speech pronounces the name “Bagge” as “baggy,” but the person prefers a silent “e.” You can change the spelling of the name to “Bag” to cause Text-to-Speech to pronounce it correctly.

Test early to discover script design issues to be addressed.

Another way to overcome mispronunciation or misunderstandings is to spell some of the information. This is especially useful for names. You can have your script speak the name, then spell it out. Or, you can give callers the option of having a name spelled out.

Testing Text-to-Speech Scripts

Setting up scripts with easy to understand Text-to-Speech output is a process of trial and error. It’s very important to plan the application, then test it out with real data to see that the speech can be understood. If you have more than one alternative, test them both. Once you are satisfied that you yourself

can understand the output well, test with representative callers [see Section 6]. This is necessary because once you have gotten used to listening to Text-to-Speech output, it is difficult to “untrain” yourself and to listen as an inexperienced caller. What is pleasing to you will not necessarily be pleasing to your callers. Test, refine, and retest as often as necessary.

Bilingual and Non- U.S English Scripts

As mentioned in Section 2, your Version 5,0 VIS can support up to two different WholeWord speech recognition packages, each in one of the three supported languages: U.S. English, Canadian French, and Mexican Spanish.[7] There are some things that you should know in order to get the most out of speech recognition in languages other than U.S. English.

Bilingual Scripts vs. Bilingual Service

Intuity CONVERSANT VIS allows you to provide callers with bilingual, and even multi-lingual, service. This means that the VIS can play prompts and announcements, and recognize some speech, in several languages. In some cases you will be able to provide this type of service with a single script. In other cases, you will need to implement a different script for each language. The complexity of implementation depends on what you want the service to do for your customers.

No matter how simple or complex your service, we suggest that all bilingual services allow callers a chance to indicate their preferred language at the very beginning of the transaction. Then the caller can be handled by the script [or portion of the script] that speaks and recognizes speech in the language that the caller has chosen. See below for more information.

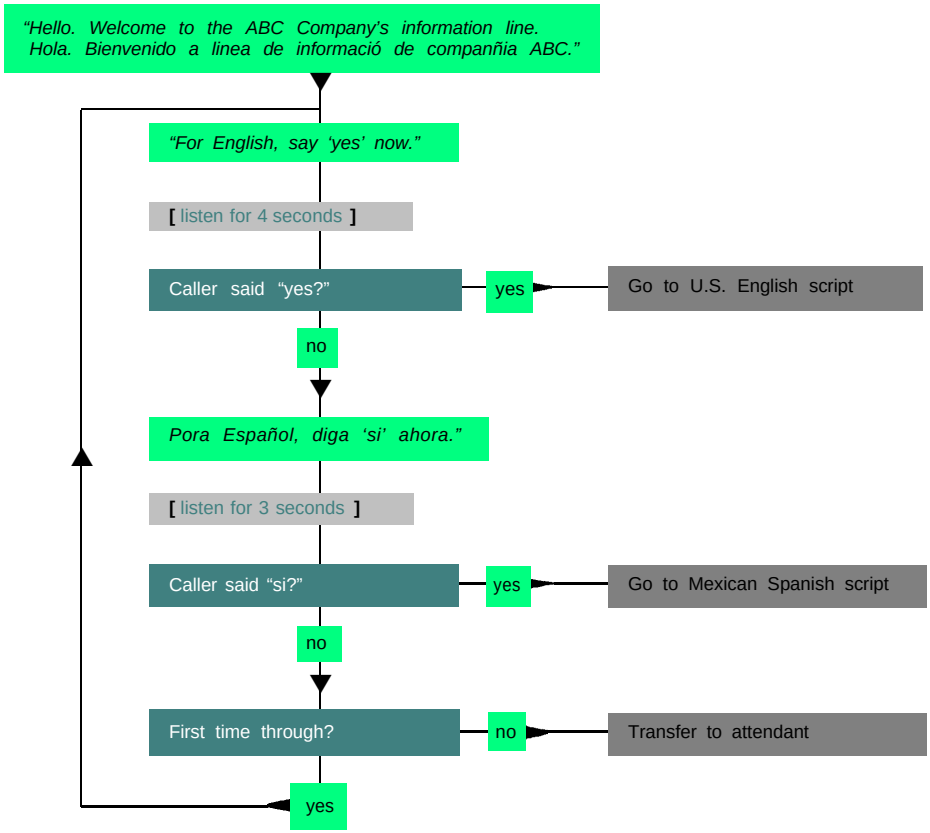
If your script does not need to speak dates, times, monetary amounts, or numbers to the caller, you can use a single script to handIe the interactions in both languages. Just set it up in two different parts, each prompting and recognizing speech in a single language. When the caller chooses a preferred language in he beginning of the script, just branch to the correct section.

In Version 5.0, a script cannot be bilingual in the same way that a person is bilingual. A bilingual person can recognize two languages simultaneously. A script cannot listen simultaneously for callers to say “yes” in U.S. English or “si” in Mexican Spanish. The recognizer can only recognize speech in a single language at any Prompt & Collect actions When you write the script, you must indicate the recognition type to be used for recognition, and recognition types each work for only one language. All the valid recognition types in your VIS are listed in the Choices menu for the Mode field on the Prompt & Collect form.

If your script must speak dates, times, monetary amounts, or numbers to the caller, each language must be handled in a separate script.[9] This is because Version 5.0 a script can have only one set of standard speech phrases, which are the phrases that are used to speak numbers. Therefore, each script can only speak numbers in a single language.

Selecting a Language

The part of the script that asks the caller to pick one of two languages is called a *language gate.* Since the VIS can only recognize a single language at any one Prompt & Collect statement, the language gate must be structured carefully. The speech recognition language gate illustrated below consists of two questions, one in each language. Each question asks the caller to say a supported vocabulary word in one of the languages. The script then branches to a different script depending on the caller’s response. A good language gate is shown on the following page:


"Hello. Welcome to the ABC Company's information line.
Hola. Bienvenido a linea de informació de companñia ABC."
"For English, say 'yes' now."
[listen for 4 seconds]
Caller said "yes?"
yes
Go to U.S. English script
no
Pora Español, diga 'si' ahora."
[listen for 3 seconds]
Caller said "si?"
yes
Go to Mexican Spanish script
no
First time through?
no
Transfer to attendant
yes

The Script Builder code to support the language gate on the previous page is shown below.

```
start:
Answer Phone
Announce
     Speak with Interrupt
          Phrase: “Welcome / Bienvenido”
Set Field Value
          Field: howmanytimes = 0
ENGLISH:
Evaluate
If homanytimes > 1
     Announce
          Speak with Interrupt
               Phrase: “Transferring.”
     Transfer to 1234 Type: Blind
End Evaluate
Set Field Value
     Field: howmanytimes = howmanytimes + 1
Prompt & Collect
     Prompt
          Speak with Interrupt
               Phrase: “For English, say yes now.”
     Input
          Mode: US_YN
          Max Number of Digits: 01
          No. of Tries To Get Input: 01
     Checklist
          Case: “Input Ok”
               Continue
          Case: “Initial Timeout”
               Goto SPANISH
```

```
          Case: “Too Few Digits”
                 Goto SPANISH
          Case: “No More Tries”
                 Goto SPANISH
End Prompt & Collet
Evaluate
If $CI_VALUE = “Y”
     External Action: Execute
          Application_Name:  “english_script”
          Write_Call_Data_Record:  “no”
          Return Field: returnfield
     End External Action
     Announce
          Speak with Interrupt
                 Phrase: “We are exeriencing tech. difs. . .”
     Quit
End Evaluate
SPANISH:
Prompt & Collect
     Prompt
          Speak with Interrupt
                 Phrase: “Para Español, diga si ahora.”
     Input
          Mode: MS_YN
          Max Number of Digits: 01
          No. of Tries to Get Input: 01
     Checklist
          Case: “Input Ok”
                 Continue
          Case: “Initial Timeout”
                 Goto ENGLISH
          Case: “Too Few Digits”
                 Goto ENGLISH
```

```
          Case: “No More Tries”
                Goto ENGLISH
End Prompt & Collect
Evaluate
If $CI_VALUE = “Y”
     External Action: Execute
          Application_Name:   “spanish_script”
          Write_Call_Data_Record: “no”
          Return Field: returnfield
     End External Action
     Announce
          Speak with Interrupt
                Phrase: “We are experiencing tech. difs. . .”
     Quit
End  Evaluate
Goto  ENGLISH
```

You can also have a language gate for a touch-tone only application script. First, greet the callers in each language. Then say, for example, *“For English, press one. Para Español, marque dos.”* Go to the appropriate script based on caller input.

Multi-Lingual Scripts

If you have at least two VIS computers, each with one or two WholeWord speech recognition packages, you can handle callers in up to three languages with what appears to callers to be a single script. You could start with the language gate illustrated above. Instead of transferring to the attendant if the caller doesn’t say “yes” or “si” at one of the expected times, you could transfer the call to a script on the other VIS. The other VIS could present a script in the final language [or transfer to the attendant] depending on the caller’s response.

Single-Digit and Yes / No Recognition

Currently, the Canadian French and Mexican Spanish packages recognize callers speaking single, isolated digits only. Connected digit recognition is available for U.S. English only. Equivalents for “yes” and “no” are supported by all WholeWord packages.

Repeating Single Digits Back to Callers

In order for the VIS to speak back spoken digits for confirmation, you must ensure that there is a recorded phrase associated with each digit. Script Builder will supply the phrase tags for these digits, and allow you to record them.[10]

Tone-Paced Entry of Number Sequences

If you are doing speech recognition with a language package which does not support connected digits, you can still allow callers to enter number sequences. Connected digit input can be simulated by using pacing tones. When a script is set up to use pacing tones, callers are prompted to speak the first digit after a tone, then wait for another tone before speaking the next digit. This continues until all digits have been read in. For instance, an interaction might go like this:

“You will now be asked to enter your customer number. Please wait for the tone, then say the first digit. When you hear the second tone, say the second digit. Continue until you have said all the digits. Begin at the tone.”
beep
< 4 >
beep
< 3 >

beep
< 2 >
beep
< 8 >
beep
< 8 >
“You said 43288. Is this correct? {pause} Please say yes or no.”

The Script Builder code to support this interaction looks like this:

```
start:
Answer Phone
Announce
      Speak with Interrupt
            Phrase: “You will now be asked to enter your number. . .”
LOOP1:
Set Field Value
      Field: count = 0
      Field: data = “”
LOOP:
Prompt & Collect
      Prompt
            Speak with Interrupt
                  Phrase: “beep”
      Input
            Mode: UK_DIG
            Max Number of Digits: 01
```

```
            Initial Timeout: 03
            Interdigit Timout: 03
      Checklist
            Case: “Input Ok”
                  Continue
            Case: “Initial Timeout”
                  Speak with Interrupt
                        Phrase: “Sorry. Pls start over. Say 1st . . .”
                  Goto LOOP1
            Case: “Too Few Digits”
                  Speak with Interrupt
                        Phrase: “Sorry. Pls start over. Say 1st . . .”
                  Goto LOOP1
            Case: “No More Tries”
                  Speak with Interrupt
                        Phrase: “Please hold for an attendant.”
                  Transfer to 1234
End Prompt & Collect
External Function
     Function Name: concat
     Use Arguments: data  data  $SCI_VALUE 0
Set Field Value
     Field: count = count + 1
Evaluate
If count >= 5
     Goto STOP
End Evaluate
Goto LOOP
STOP:
Prompt & Collect
     Prompt
            Speak with Interrupt
```

```
                    Phrase: “You said.”
                    Field: data As C
                    Phrase: “Is this correct?”
                    Phrase: “sil.500”
                    Phrase: “sil.500”
                    Phrase: “sil.500”
                    Phrase: “Say yes or no, now.”
        Input
              Mode:  UK_YN
              Max Number Of Digits: 01
              Initial Timeout: 03
              Interdigit Timeout: 03
        Checklist
              Case: “Y”
                    Continue
              Case: “N”
                    Speak with Interrupt
                          Phrase: “ Sorry, start over. Say 1st . . .”
                    Goto  LOOP1
              Case: “Not On List”
                    Reprompt
              Case: “Initial Timeout”
                    Reprompt
              Case: “Too Few Digits”
                    Reprompt
              Case: “No More Tries”
                    Speak with Interrupt
                          Phrase: “Please hold for an attendant.”
                    Transfer to 1234
End Prompt & Collect
Disconnect
Quit
```

Repeating Digit Sequences Back to Callers

With non-English languages, the only reliable way to speak callers' numeric entries back to them is as a "character" [C]. This means that callers speaking "one, two, three" will hear their input read back as *"one, two, three."* rather than *"one hundred twenty three."* This should be appropriate for most applications.

Confirming Digit Sequences

The Version 5.0 speech recognition Confirm within the Prompt & Collect action works only when you are using the U.S. English WholeWord package. WholeWord packages using other languages must use a separate Prompt & Collect statement to confirm caller entries. See *Feedback and Confirmation* earlier in this section for more information.

Dates, Times, and Monetary Amounts

In Version 5.0, the capability of speaking out dates, times, and monetary amounts for languages other than U.S. English can be added to your system on a custom basis. Version 5.0 supports only U.S. English formats for dates, times, and monetary amounts. You may be able to make some of the English formats work for your language. See your AT&T representative for more information about obtaining these capabilities on a custom basis.

Using Fax

As mentioned in Section 2, the Intuity CONVERSANT VIS Script Builder Fax Actions allow voice response applications to send fax messages to callers. Currently, these fax-on-demand capabilities are available when you have the AT&T Fax Attendant System installed along with your VIS. The Script Builder Fax Actions will soon be available separately. Fax messaging will still be available through the Fax Attendant System only. See your AT&T representative for details.

This section contains some tips on designing good Fax applications. See *Intuity CONVERSANT VIS Version 5.0 Script Builder,* 585-310-727 for more information on the topics in this section.

Letting Callers Know What's Available

If you have a long list of different types of information available to callers, allow callers to request that the list be faxed to them. If each fax is tagged with a number, callers who have the list can simply call in and request a fax by number, without having to listen to a menu. For example:

> *" Welcome to State Hospital's Pharmaceutical Information Line. Please enter the four-digit number of the drug you would like to know about. To receive a fax listing all drugs, press the pound key. "*

Managing Delays

Be sure that the activities you allow callers to perform do not leave them waiting for an uncomfortable amount of time. See Section 6 for information on testing your script to determine how callers react to delays.

There are a number of things you can do to avoid delays.

Avoid delays by doing the processing after the call is finished.

You can do this by using the ON_ HANGUP_ EVENT label within your script. All steps that come after the label will be done when the caller hangs up. Although this saves the caller time, if the processing fails the caller is no longer on the line and so can not be informed. If this happens, you could set the script to notify the administrator [possibly by sending electronic mail] when this happens, so that the caller may be contacted.

Store information as fax files.

Information stored as a text file must first be converted to fax format before being sent out as a fax. Information stored as a fax file does not need to be converted if sent with the FAX_Send action. To speed up calls, store information in fax format [by using the “Fax Response Workspace Administration” window] whenever possible. Use text format mainly for information that changes often.

Choose Fax Actions carefully.

- If you are combining a number of text files, use the UNIX command “cat” instead of FAX_Combine. Only use FAX_Combine if you are combining fax files, or fax and text files.

- If you do not need to customize your cover page for every different fax, you can just use FAX_Send, rather than FAX_CovrPage and FAX_Send. In FAX_Send, send your cover page as the first file, and the desired fax information as the second file.

Use fewer fax channels.

If you find that callers experience long delays, response times may be sped up by running the script on fewer channels.

If you know callers will face a brief wait, always let them know. For example, you could say

There are things you can do to make an unavoidable wait more pleasant. See *Pacing the Script* earlier in this section for a discussion of this issue.

Assuring that Fax Messages are Sent Successfully

As in any script, the system might have a problem when attempting to complete an instruction. Remember to structure your script to detect problems like this. Always check the return value from an action which tries to send a fax. If the return value indicates problem, inform the caller that there is a problem and the fax cannot be sent.

Speaking the Job ID back to the caller is a good way to track the delivery of a fax. If the fax is not received and the caller phones in to find out why, the system administrator can use the Job ID provided by the caller to trace the problem [by using the “Fax Transmission Control” window].

Exec_UNIX: Beyond Fax Response

the Exec_UNIX Fax Action gives you the power to execute commands, programs, or shell scripts, and to access databases on host computers. The innovative aspect of this action is that it can also be used in application scripts that do not send faxes. Exec_UNIX allows you to do in Script Builder what you formerly could do only with C language, either using a data interface process [DIP] or the Intuity Response application Programming Interface [IRAPI]. If you know the UNIX shell, you can use Exec_UNIX!

For example, let's say you wanted a script to read a subscriber's electronic mail over the telephone. You could first ask the caller to enter a login ID and password for security. You might then use Exec_UNIX to issue a shell command to find the first mail message in the caller's directory, then locate the line containing the word "Subject." You could then use Text-to-Speech to read out the subject line, and then offer the caller the option of listening to the message. If the caller chooses to listen, you could have Text-to-Speech speak the contents of the file, or skip to the next message in the mailbox.

The Exec_UNIX action can return to the script up to 127 characters, or up to the first "new line" character.

footnotes

1. *Standard speech includes phrases which allow scripts to speak numbers, amounts, dates, and times. The phrases include numbers, ordinal numbers, days of the week, and months of the year.*

2. *You may even want to try using a variety of yes/no synonyms in this wordlist. Prototype and test!*

3. *Phrases can be up to four minutes long.*

4. *Standard speech contains seven different silence phrases: siI.050, sil.100, sil.200, sil.300, sil.400, sil.500, and sil.600. Each phrase name represents the number of milliseconds of silence that makes up the phrase. For example, for 500 milliseconds of silence, play sil.500. Each phrase has a phrase number, which can be different for each application. If you do not know the speech phrase number for a silence phrase, go to the UNIX prompt and type grep* {phrase name] /speech/talk/{application name).pl, *where {phrase name} is the name of one of the silence phrases.*

5. *When using the built-in Confirm, if the confirm prompt the script reads out [for example, "You said 'sales.' Is this correct?"] is made up of a combination of Text-to-Speech and recorded speech, the caller will not be able to barge in during the prompt. The caller must wait until the end of the prompt to respond. If the confirm prompt is read out entirely with recorded speech or entirely with Text-to-Speech, the caller is able to barge in during the prompt (if the script is set up to allow barge in). See Using Barge In and Talk Off later in this section.*

6. *Although you may have up to 500 words on a wordlist and 200 different wordlists, a single Flex Word package currently supports only 2000 total words. AT&T will soon begin performance testing on larger vocabularies. If you are interested in a larger number of total words, contact your AT&T representative.*

7. *The VIS now supports additional languages and countries on the Version 4.0 generic. Contact your AT&T representative for details.*

8. *If you would like to experiment with recognizing words in more than one language simultaneously, a special custom grammar may be purchased from AT&T to support your effort.*

9. *This is true if you are using Script Builder to implement your script. If you use script language or a C language script which uses the Intuity Response Application Programming Interface [IRAPI] instead, you can change speech pools and speak numbers in more than one language.*

10. *Since only one version of each recorded digit is supported, gender mismatches may happen for languages that use different versions of digits depending on the gender of the object. You may be able to structure your sentences to eliminate this issue.*

section 6

deploying your script

Now that you have a good idea of how to design a voice response application script, what's the next step? Actually, there are several next steps to getting your application "in service." No matter how carefully you've planned out your script, your first design won't be perfect. Even people who have designed scripts for years don't get it right the first time. Designers of the most successful scripts arrive at their designs through *iterative testing.* "Iterative" means "repetitive." This involves testing your script a number of times, using the data gathered during one test to refine the script design before testing again. Iterative design and testing helps you find out if your application script lives up to the usability principles described in Section 3.

Iterative design involves a repeating cycle of activities that you do until your application design is the best it can be. This involves the following steps:

1. Specify a preliminary script design.
2. Prototype the preliminary script.
3. Set usability objectives.
4. Test the preliminary script with a group of potential callers and gather data.
5. Check data against usability objectives; refine the script based on the data gathered in the test.
6. Re-test the script and gather more data.
7. Iterate [repeat steps 5 and 6] until usability objectives are met.
8. Put the tested and tuned script into service.

Each of these steps is described below.

Specify a Preliminary Application Script

First, plan out a first draft how the transaction will appear to callers [see *Flowcharting and Outlining* in Section 5]. Use your knowledge of the callers, the application requirements, and the transaction as it presently occurs between people to guide your design. Refer to the guidelines and suggestions given throughout this handbook to help you decide how to present and gather information from callers. Also take into account any lessons you've learned while designing applications in the past.

Sketch out a flowchart or outline of your application design. If you have more than one possible design, sketch them both out. Write out all the prompts and announcements that will be spoken to the caller, as well as what you expect the caller to do or say at each point in the script. Pay special attention to what should happen when errors occur. When should the caller be transferred to the attendant? For each prompt, what should the corresponding re-prompt be? Make sure the application fulfills all the requirements placed on it.

After you have a satisfactory flowchart or outline, share it with your colleagues and others who have a stake in the voice response application and ask for feedback. Listen to their concerns, make modifications to the application flow based on the most useful feedback.

2 Prototype the Preliminary Script

Once you have a satisfactory paper representation of the complete application flow, implement it in proto type form. A "prototype" is the "first or original model" of something. Prototypes are not permanent; we expect to refine and modify them. Implement the script, or have it developed by a colleague, using Script Builder, script language, or C language [using the Intuity Response Application Programming Interface, IRAPI]. All aspects of this prototype do not have to be perfect. For example, it's quicker to use Text-to-Speech output or pre-record prompts and announcements in your own voice instead of hiring a professional speaker. If you are reading large blocks of text to callers, you don't have to have all of the real files available. Instead of transferring callers to the attendant, you can play a place holder phrase like *"At this point, you would be connected with an attendant."* Implement the most important parts of the transaction, including error handling. Cut corners when you need to, since this prototype is only the first step towards your final script. Many changes will likely occur before your final design is reached. Call in and test the prototype yourself to see that it functions as intended.

3 Set Usability Objectives

Next, decide on some usability objectives. Usability objectives allow you to measure whether or not your script will serve your callers well. Different services will require different usability objectives. Remember to make your objectives measurable, so that your testing will yield data to help you decide if the script meets the objectives. Here are some examples of usability objectives:

- 95% of test callers will be able to complete the transaction in less than 45 seconds with no errors.
- 90% of test callers will be able to recover from all errors, thereby avoiding transfer to an attendant.
- 98% of test callers will rate the script's ease of use as "very good" or "excellent."

The usability objectives you set should reflect the aspects of the script that are important to you.

4 Test the Preliminary Script

When you have a working prototype of your application, it is ready to be tested. Have a small group of representative callers use the application script to accomplish a set of typical tasks. It's important to make sure that the people tested are truly representative of the calling population. If your script is designed for customers, don't test it with your colleagues [who probably have knowledge or experience that customers won't have]. If you know that a large proportion of the callers will be senior citizens, for example, make sure that most of the people you test are senior citizens.

As the tester, your task is to collect two types of data:

1 **Performance measurements.** Two important measures are time [how long it takes for callers to complete various tasks] and errors [the number and types of errors made, and where in the script they occurred].

The collection of performance measures sets usability testing apart from market research.

2 **Opinion feedback.** Important measures include caller judgments on how clear the prompts are, whether each task was easy to do, and overall level of satisfaction with the script.

It is very important to collect both types of data, since each gives you different information about the application. All of the data can later be used to guide you in modifying the application design so that it meets the usability objectives you have set.

Testing the user interface of the application with potential callers is called *usability testing.* This type of testing is done to see how easily the intended audience can use the application to perform specific tasks. You can perform usability testing in a number of ways, depending on available facilities, time, and resources.

You can adapt these usability testing methods to the time, resources, and facilities available.

The more structured usability testing methods involve bringing potential callers into a controlled, lab-type environment. In the lab, give them a list of tasks to perform by calling into the script. Observe callers interacting with the script; videotape or record them if possible, so you can refer to the tapes later and take performance measurements [such as counting the number of errors and measuring the time taken to complete the assigned tasks]. When the callers are finished using the prototype script, gather their preferences and reactions to it on a questionnaire [see *next page for a sample questionnaire].* You might want to gather opinion feedback by talking one-on-one with each caller, asking each a set of identical questions.

sample questionnaire

name [optional]:

sex: ● male ● female

Is U.S. english your native language? ● yes ● no

if no, what is your native language?

overall, i find the service	difficult to use				easy to use
	❶	❷	❸	❹	❺
i find the recognition accuracy	poor				excellent
	❶	❷	❸	❹	❺
the prompts present	too little information				too much information
	❶	❷	❸	❹	❺
the information in the printed document was	hard to understand				easy to understand
	❶	❷	❸	❹	❺

did you have any problems using the service? if so, please describe.

what do you like best about the service?

what do you like least about the service?

The less structured usability testing methods involve similar tasks and measurements, but do not require that potential callers be brought into a controlled environment. You can give the potential callers a list of tasks to perform by calling into the script, but allow them to call from their homes or offices. Measure performance by listening in to the transactions as they happen, or record them for later measurement. After the callers have used the prototype script, have them fill out a questionnair to indicate their preferences and reactions to the tasks. You might choose to phone them afterwards and interview them instead of having them complete a questionnaire.

When you observe callers interacting with the application during usability testing, pay attention to everything that happens during the call. You can observe by sitting with callers as they phone in to the application, or by listening in on a telephone line by using a channel monitor script available from your AT&T representative.

When observing, pay particular attention to:

Points in the script where callers become confused
You can assume a caller is confused if you hear particular extraneous speech [such as “Huh?” or “I don’t know what you mean”], response delays, or silence instead of a response. Often you can tell callers are confused just by looking at their facial expressions or listening to their tone of voice.

Information you get from observing callers interacting with your application can help you decide how to modify the application to make it easier with which to interact. For example, you might find that callers sound unsure or delay responding at a prompt that says *“What is your customer number?”* To remedy the problem, you could prompt differently, saying *“Your eight-digit customer number may be printed on the*

back of your catalog. Do you have a customer number? " Callers who say "no" can be transferred to an attendant, while callers who say "yes" can be prompted next with *"What is your customer number?"*

Points in the script where callers say words that are not key words

For example, you might find that a number of callers gave non-key word responses to the question "What *department do you want?"* In this case, you might want to list the departments after you have prompted once and no key word was recognized. The script could say *"What department do you want? Please say sales, service, credit, or office."*

Recognition errors

If your script uses speech recognition, listen to what the caller says, and what the recognizer determines that the caller says. You can learn a lot about your script this way. If you want the caller to exhibit a particular behavior and you don't see it, you can make changes in the prompts to try to make the caller behave in a certain way. For example, if you do not want callers to barge in during the prompt but callers tend to barge in anyway, you might want to state near the beginning of the transaction that callers should wait until the end of the prompt before responding, or use the word "now" when the system is ready to accept input.

If you are using FlexWord SR, substitution errors [that is, where the recognize mistakes one word for another] are extremely important. If, for many callers, one option is often mistaken for another, you may want to consider changing the wording of one of the options so the recognize does not confuse the two options. Remember, vocabulary iteration is expected; do not be surprised if it takes some tuning to get a highly successful application script [see *What is a Successful Speech Recognition Application?* in Section 4].

System performance

Pay attention to how the system behaves. Is the response time too slow? If the server goes down or other system problems occur, does the script handle the call in a graceful manner?

5 Refine the Preliminary Script Based on Usability Objectives

After gathering performance measurements and opinion feedback, analyze the information to see if the usability objectives you set are being met. If not, let the data guide you in refining your script. If you observe callers having trouble with the script, use both performance and opinion data to try to figure out where and why. Then make changes to the script based on your findings.

6 Re-test the Script

Next, test your modified script with another group of representative callers. Go through the same type of testing you used before, and gather the same types of data. Compare the results of the second round of testing to the first, to see if the script refinements had a positive effect on meeting your usability objectives.

7 Iterate Until Usability Objectives are Met

If your usability objectives are not yet met, refine the script again, based on the second round of testing. If you make refinements, be sure to retest the script with potential callers to see that the changes you made move you towards meeting your usability objectives.

When you are confident in your script, you can do end-to-end testing of your entire application. This may include database lookups, host interfaces, in-service and out-of-service hours, and other aspects of the

application. You might consider putting your application in service for a subset of real callers to use for real transactions. Only do this if you are confident that your script has been proven easy to use during internal tests. Continue gathering both types of data.

Put the Tested and Tuned Script into Service

Once you are satisfied that your usability objectives have been met, put your tested and tuned script into service for all of your intended callers. You might want to do this a single channel at a time, to see how the system performs under load. It's important to monitor the acceptance of your script, and how well the callers are completing their intended tasks.

When the script is in service, check to see that the usability goals are still being met.

Listening to real callers interacting with your script is time-consuming, but can reveal valuable information about your caller population as well as the application script. This can impact your decision to deploy an application using a particular technology. For example, you might have assumed that only a small percentage of callers would have non-U.S. English accents, when in fact the proportion was quite large. In this case, you probably do not want to offer a speech recognition-only application. Consider encouraging callers to use touch tone, and only if they have a rotary telephone would you offer callers the option to give spoken input.

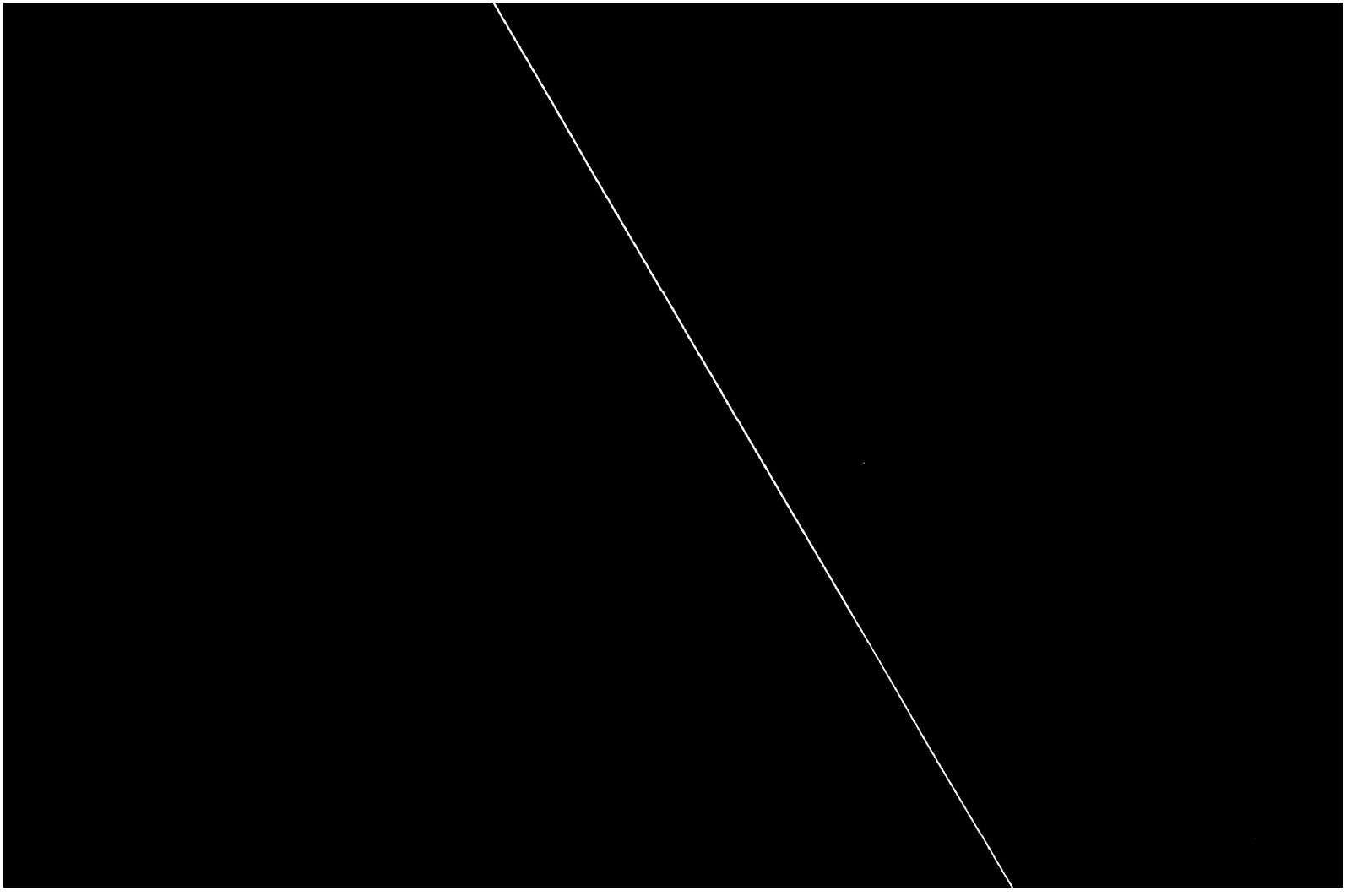
Even when your script is up and running and serving your customers, it's a good idea to keep an eye on performance. You may want to monitor random calls to listen to how callers are interacting with the script. You might also consider collecting opinion feedback from your customers, to help you refine the voice response services you offer.

epilogue

We hope you have found the guidelines and suggestions in this handbook useful and relevant, and that you now feel more comfortable with the design aspects of voice response applications. Your callers will benefit from the implementation of your new [or renewed] knowledge.

We encourage you to continue learning about Intuity CONVERSANT VIS and voice response in general. Make sure you subscribe to our electronic information service, ACCESS, to keep upon late-breaking news regarding your VIS. ACCESS uses AT&T Mail to disseminate the latest news about applications, products, service offerings, security tips, Change Notices, industry issues, and more. Contact your AT&T representative for more information.

If you have not already done so, please complete and mail the feedback form located in this handbook. We want your opinions about the handbook, and your ideas on how we can make our product and documentation better serve your needs.



585-310-551
comcode 107316606
december 1994