INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

WishToys: A Web-based Electronic Shopping Assistant Featuring Voice Access

Miroslav Cosic

A Major Technical Report

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Computer Science at
Concordia University
Montreal, Quebec, Canada

April 2001

ABSTRACT

WishToys: A Web-based Electronic Shopping Assistant
Featuring Voice Access

Miroslav Cosic

This report describes an electronic shopping assistant specialized in toys. The shopping assistant features two distinct user interfaces. One interface is a web-based graphical user interface (GUI) that fully uses the relatively high bandwidth of the Internet connection, allowing for high resolution and high colour content graphics. The GUI interface offers the full functionality of the shopping assistant. The other interface is a low-bandwidth voice user interface (VUI), offering a reduced set of features and allowing an easy and quick access to pre-selected list of toys for people on the move. The VUI is geared specifically towards mobile use, i.e. for users calling from cell phones.

ACKNOWLEDGEMENTS

I hereby wish to acknowledge the endless wisdom, patience, and capacity to offer guidance of my supervisor, Dr. Thriuvengadam Radhakrishnan, to whom I will forever be grateful not only for making this project a reality, but also for introducing me to the wonderful world of user interface design in his excellent courses.

I wish to thank Dr. Sabine Bergler for her extraordinary effort in reviewing this report, and Ms. Halina Monkiewicz for guiding me through the not-user-friendly thesis submission process.

This project is dedicated to my fiancée, Hatidza Kaleta, without whose love and support this and many other things couldn't have been possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: Introduction

With the rush to capitalize on the increasing popularity of the World Wide Web (henceforth referred to as the Web), retail companies are searching for ways to attract customers to their Web sites and retain them by offering them an interesting and pleasant on-line shopping experience. Several companies have announced electronic commerce solutions for retailers, integrating Web browser-based access through a graphical user interface (GUI) with telephone-based access through a voice user interface (VUI). With the requisite back office software and the consulting and set-up costs, the final cost for these systems may reach $50 000 to $150 000.

The WishToys demo is an individual effort in the same problem domain, focusing on the user interface, and attempting to improve on certain aspects of the GUI, as well as enhancing the functionality with the addition of a VUI, specifically for the toy shopping application domain.

1.1 History

The WishToys demo originated as an individual course project for the Human-Computer Communication (COMP 675) course. At the time, it was called the Electronic Shopping Assistant, a GUI application for browsing a toy catalogue and placing orders using a shopping cart metaphor. The finished application was then used as the subject matter for the Advances in Human-Computer Interaction (COMP 773) course, where several groups of students had the task to constructively criticize the GUI design and come up with a list of defects and suggested changes. Some of the suggestions resulting from that group work were implemented in the actual application during the course. As a result, the

starting point for this work was a GUI application that had already passed several iterations of different usability evaluations.

1.2 Objectives

The goal of this major report is to develop a VUI extension of the existing GUI interface, available over the phone, especially in a mobile setting. As this is not a commercial application, looking into different ways of doing the same thing is more important than adopting one approach and developing it into a fully functional application. At the same time, the intention is to put into practice the speech application design approach laid out in a 1998 book "Designing Interactive Speech Systems" by Danish researchers Bernsen, Dybkjær, and Dybkjær. [1]

CHAPTER 2: Design Methodology

The project that spawned the "Designing Interactive Speech Systems" book, [1] the Danish Dialogue Project (DDP), is a flight-information system in which the research team invested 30 person-years of work. Obviously, the resources available for the present work represent but a slim percentage of that amount. Nevertheless, the methodology that the researchers propose can still be partially applied to the specifics of this work.

2.1 DDP Methodology

As the authors of this book [1] did not adopt a specific name for their methodology, it will henceforth be referred to as the DDP methodology. As shown in Figure 2-1, the DDP methodology is an iterative development methodology.

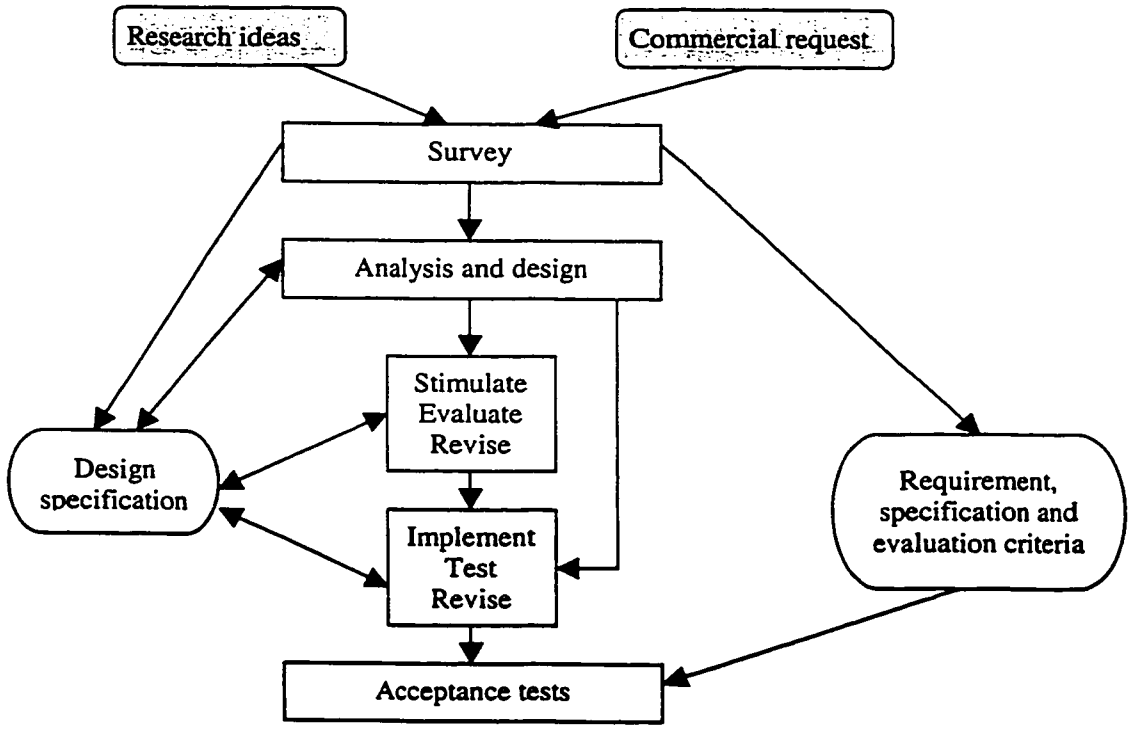

Figure 2-1: DDP methodology flowchart

The DDP methodology has the following major inputs:

- **Requirement Specification**: The requirements specification summarizes the goals and constraints identified through a survey.
- **Evaluation Criteria**: The evaluation criteria are based on the established requirement specification.
- **Design Specification**: Many issues concerning the operational aspects of system design do not belong to the requirement specification, but are needed in the design specification, which describes how to make a system that will satisfy the requirement specification and meet the evaluation criteria.

The above-mentioned inputs are used throughout the development process, consisting of the following phases:

- **Survey**: The survey should provide a reasonable and reliable estimate of project feasibility. It should cover strategic goals, system goals and constraints, and resource constraints. The requirement specification and evaluation criteria should be completed at the end of this phase.
- **Analysis and Design**: This phase should produce a design specification for the system with such a level of detail that it can serve as a basis for implementation. The design specification should be completed at the end of this phase.
- **Simulating, Evaluating, and Revising**: This phase involves running the simulated system and evaluating, transcribing and analyzing the results. The three project inputs may be impacted to a greater degree by the findings during this phase.
- **Implementing, Testing, and Revising**: This is the phase in which black box and glass box tests are performed. It also involves evaluation, transcription and analysis. The three project inputs may be impacted to a lesser degree by the findings during this phase.

- **Acceptance Tests**: The final phase is performed before and/or after installation, and may include controlled user tests, field tests, and a final acceptance test. The three project inputs can no longer be modified in this phase.

Each of the project inputs and development phases is described in a separate chapter in this major report, starting with the survey. An exception is made for the last two steps, i.e. the implementation step and the acceptance tests, which could not be performed due to a lack of resources.

2.2 GUI Development Methodology

Although the DDP development methodology was developed with speech applications in mind, there is nothing to prevent it from being used in a GUI setting with certain modifications. Such an exercise is outside the scope of this major report, both due to the amount of effort involved and due to the fact that the greater part of the development for the GUI component was already done as coursework for other courses. A description of those phases of the GUI component's development can be found in documents produced previously. [2, 3, 4, 5] Hence, this major report will cover only those phases of GUI development performed specifically for this work, which will be described in sections corresponding to the equivalent VUI development phases.

CHAPTER 3: Survey

3.1 Strategic Goals

The application to be developed is a web-based retail shopping application (also known as a business-to-customer or B2C application) specializing in toys. The GUI application is extended to include a phone-based VUI. As such, the application's strategic goals are service improvement relative to existing web-based retail shopping applications, and an exploration of the newest technologies merging speech recognition and networking. A list of existing similar applications is given in Appendix A.

At the outset, the VUI component's strategic goals were set to be developing an improved methodology for handling confusability issues, a special class of issues within the all-important field of speech recognition error prevention and correction. More specifically, the VUI would need to feature advanced "fallback position" elements for error correction.

Other objectives, which couldn't be pursued in this major report past a rudimentary stage for lack of resources, were service improvement and technology exploration by the addition of an intelligent agent Shopping Assistant feature to the GUI. The agent is an animated character that has text-to-speech capability, and would use networking (multi-agent system), self-learning, and collaborative filtering. Ideally, the agent would also provide the help functionality, either when requested by the user or when the agent detects that the user could be in need of help.

3.2 System Goals and Constraints

3.2.1 Domain

The application should satisfy the needs of a growing group of people who have an active lifestyle, who have children, who have little time to waste on shopping, and who do not have objections to shopping over the web or over the phone. More precisely, the application should cater to the following segments of users or customers:

- Customers without a clear idea on what to buy, who need recommendations and/or an easy way to narrow down their options;
- Customers in a hurry to buy a present, which must be delivered before a certain deadline;
- Customers with children (relatives, friends' children, etc.) who participate in the toy selection process by browsing the toy catalogue independently or with their parents and composing "wish" lists. The concept of wish lists can be extended to other domains such as bridal registries and baby showers, in similar web shopping applications;
- Customers who might want to use the phone interface when they are in a hurry, because they remembered an important date at the last minute, or they procrastinated until the last moment. Another use for the phone interface is for the user to finalize a purchase of a child's wish list without using the home computer, so as to surprise the child. A third scenario is when a customer is away from home and a Web-enabled personal computer is not easily accessible.

The above-stated goals can be summarized as providing a shopping experience characterized by urgency, mobility, and speed. Once it is developed for the toy domain, the application could be adapted to other domains, following sets of generalized guidelines and findings.

3.2.2 Interactive Tasks

For the GUI, the set of interactive tasks includes all the functionality of the application: logging in to the application, selecting a PIN for new users, browsing and searching for toys, adding toys to a shopping cart and/or wish list, user profile management, shipping information management, billing information management, and transaction processing. The VUI needs to support a limited set of features: logging in, confirming the purchase of a wish list, purchasing a toy from a limited set of choices, choosing from a list of shipping addresses, leaving a message with additional instructions, or requesting to speak to a customer representative. The VUI dialogue will be system-driven, a common trait among all available Web-based speech recognition systems. Other VUI features dictated by the available technology are: (i) use of word spotting by default, unless continuous speech recognition is available, and (ii) use of default meta-commands to correct recognition failures and user mistakes.

3.2.3 Usability Objectives

In conformance to the users' expectations for all modern GUI applications, the average user should be able to use the GUI component without consulting any written documentation or taking part in any special training. The VUI being an extension of the GUI, a usability objective for the former would be for the average user to be able to use it after a single session with the latter.

3.3 Resource Constraints

3.3.1 Manpower and Time

1 graduate student part-time (8 hours per week) for 10 months

3.3.2 *Hardware*

- Portable computer (Pentium II @ 400 MHz, 192 MB RAM), for development and simulation;
- Hand-held microphone and speakers, or headset with microphone.
- VoxML application server with telephony equipment (for the finalized application)
- Windows Web server for the GUI part (for the finalized application)
- Windows database server (running SQL Server 7) for the common database part (for the finalized application)

3.3.3 *Software*

- **Microsoft Windows NT 4.0 SP 5** – The operating system was chosen because of the wealth of development tools, such as Visual Basic and Motorola's MADK, and because of its popularity (knowing that it is compatible with Windows 98 and 95). The production version of the application would need to run on a Windows Web server, and the client could be a Microsoft Internet Explorer browser on any of the previously mentioned versions of Windows.
- **Microsoft Visual Basic 6 SP 3** – Visual Basic allows for quick prototyping of GUI applications, and the code itself can then be reworked into an Active Server Pages (ASP) Web application that can run on a Windows Web server. A major issue to be resolved in transforming the Visual Basic application into an ASP application is the impact of the user's Web browsing operations on the internal state of the application.
- **Microsoft Access 2000** – Access was used for the database shared between the GUI and the VUI. The database can be converted into an SQL Server database for large-scale use.
- **Microsoft Agent SDK 2.0** – The Agent SDK is used both in development, for the rudimentary prototype of the GUI intelligent agent, and is also used natively for simulation in Motorola's MADK.

- **Microsoft Speech Recognizer 4.0** – The speech recognition engine is distributed as part of the Agent SDK and Motorola's MADK. Using a different engine would have introduced configuration problems on the development system. The finalized application's VUI component would run on a VoxML application server where it would use the recognizer provided by Nuance Communications.
- **Lernout & Hauspie TruVoice 6.0 for Windows** – The L&H text-to-speech engine is distributed as part of the Agent SDK and Motorola's MADK. Using a different engine would have introduced configuration problems on the development system. Ideally, professional spoken audio files would need to be used for the finalized application.
- **Motorola Mobile ADK (MADK) 2.0 [6]** – The MADK is a free integrated development environment (IDE) for creating and simulating VoxML applications. VoxML was chosen because of the availability of the free MADK, and for its ability to integrate with ASP code that would be shared with the GUI side of the application.
- **Sheridan Software ActiveListBar 1.0 [7]** – The ListBar is the GUI component commonly known as the "Outlook bar" because it became popular thanks to Microsoft Outlook.

Alternative web-based speech recognition technologies are briefly described in Appendix B.

CHAPTER 4: Requirements Specification

The starting assumptions about the application are as follows:

- The GUI component is intended for use by all levels of users, and therefore has several user interface levels;
- The VUI component is intended for use by adult users and therefore has only one user level;
- For each account, there is one Master User, to whose credit card all the purchases are charged. There may be any number of Family Member users associated to the account. Each user (Master User or Family Member) has a profile, which determines certain aspects of the GUI and contains information concerning purchasing privileges. A customer number and a personal identification number (PIN) uniquely identify each user;
- A Family Member profile needs to be created through the Master User profile. Once the profile is created, the Family Member can access the account independently of the Master User;
- Before the VUI can be used, users need to use the GUI to set up a Master User account (and optional Family Member accounts). A customer number and PIN are needed to help identify the caller when using the VUI. If the available technology will allow it, the PIN could be replaced by a voiceprint for speaker verification;

Starting from these assumptions, the requirements for both interfaces are given in Table 4-1 below. The columns to the right indicate to which interface the specific requirement is applicable.

Table 4-1: System requirements

Requirement Group	Interface	
Requirement	GUI	VUI
Identification		
1. A first-time user may log onto the system as a "Visitor/Guest."	✓	
2. An auxiliary user (see below under Profile Management) may log in by using their customer number and PIN.	✓	✓
3. A Master User (see below under Profile Management) may log in by using their customer number and PIN.	✓	✓
User Interface Presentation		
4. The interface will have three general levels: child, adult, and senior.	✓	
5. The user will have a choice of user interface level at first log-in.	✓	
6. The VUI will have one level, corresponding to the adult level in the GUI.		✓
Profile Management		
7. The user is assigned an automatically generated customer number.	✓	
8. A profile is associated with a customer number and PIN	✓	✓
9. A user who provides a name, billing address, and credit card information becomes a Master User.	✓	
10. An auxiliary user profile is associated with the Master User's billing address and credit card information.	✓	
11. Only a Master User has the ability to create auxiliary profiles.	✓	
12. The Master User determines purchasing privileges for auxiliary profiles.	✓	
Commerce-related Requirements		
13. The system will support items on sale.	✓	✓
14. The system will support best-selling items.	✓	✓
15. The Master User may provide a separate shipping address as part of the profile.	✓	
16. The Master User may add other shipping addresses in an Address Book.	✓	
17. Auxiliary users cannot see the associated Master User's credit card information.	✓	
18. Auxiliary users cannot modify the associated Master User's credit card information.	✓	
Help System		
19. A context-sensitive Help System will be available.	✓	✓

Requirement Group	**Interface**	
Requirement	GUI	VUI
20. A Shopping Assistant Agent – an intelligent agent whose presentation is adapted to the current interface level – will be available.	✓	
Error Handling		
21. The system will offer corrective answers in order to provide denial of false assumptions whenever possible.	✓	✓

CHAPTER 5: Evaluation Criteria

Based on the example of the DDP itself, the evaluation criteria can be stated in a somewhat imprecise manner, as many of the measurements are of a subjective nature. A set of criteria for this system follows:

- Close-to-real-time performance is required
- The first criteria can only be satisfied with acceptable recognition rates if the average user utterance length doesn't exceed 3-4 words and the maximum utterance doesn't exceed 10 words.
- The systems vocabulary may not exceed 500 words although this size may be insufficient.
- The system must sufficiently cover the task domain.
- The system must be robust; i.e. able to cope in a reasonable way with input that is either not understood or non co-operative.
- Restrictions on language and dialogue must be such that users can use restricted but intuitive and natural forms of language and dialogue.
- Flexibility must be optimized within the given constraints and criteria.

On this last point, other research has identified a set of evaluation criteria specifically targeting flexibility, as follows: [8]

- Will the design allow for expansion from an initially small configuration?
- Will the system adapt to users who vary in experience and in cognitive style?
- Is the design easily implemented on different hardware?
- Can the design be readily modified to suit an individual user?
- Is the design a description or a computational formalism?
- Is the design method explicit, within a set of formal concepts?

CHAPTER 6: Analysis and Design

6.1 Iterative Development Cycle

The DDP methodology calls for an iterative development cycle. From a first version of the interaction model, the project can go straight into an implement-test-revise cycle, or proceed to simulation before implementation. In both iterative cycle scenarios, the starting point is a good first interaction model, which should be as detailed as possible. The methodology allows for some form of re-design right up to the acceptance test phase. Unfortunately, the amount of work invested in this type of development cannot be adequately captured in a linear document such as this major report.

6.2 Cooperativity Guidelines

The DDP methodology introduces the notion of *cooperativity* as a key to successful design of a first interaction model. Cooperativity guidelines are given in Table 6-1, covering seven interaction aspects. The two types of guidelines are generic (GG) and specific (SG), where specific guidelines extend the generic ones. Some guidelines are conflicting, so designers need to make trade-offs.

Table 6-1: Cooperativity guidelines

Interaction Aspect	G/S G no.	Generic or Specific Guideline
Aspect 1: **Informativeness**	GG1	Make your contribution as informative as is required (for the current purposes of the exchange).
	SG1	Be fully explicit in communicating to users the commitments they have made.
	SG2	Provide feedback on each piece of information provided by the user.

Interaction Aspect	G/S G no.	Generic or Specific Guideline
	GG2	Do not make your contribution more informative than is required.
Aspect 2: **Truth and Evidence**	GG3	Do not say what you believe to be false.
	GG4	Do not say that for which you lack adequate evidence.
Aspect 3: **Relevance**	GG5	Be relevant, i.e. be appropriate to the immediate needs at each stage of the transaction.
Aspect 4: **Manner**	GG6	Avoid obscurity of expression.
	GG7	Avoid ambiguity.
	SG3	Provide same formulation of the same question (or address) to users everywhere in the system's interaction turns.
	GG8	Be brief (avoid unnecessary prolixity).
	GG9	Be orderly.
Aspect 5: **Partner Asymmetry**	GG10	Inform the users of the important non-normal characteristics that they should take into account in order to behave cooperatively in spoken interaction. Ensure the feasibility of what is required of them.
	SG4	Provide clear and comprehensible communication of what the system can and cannot do.
	SG5	Provide clear and sufficient instructions to users on how to interact with the system.
Aspect 6: **Background knowledge**	GG11	Take partners' relevant background knowledge into account.
	SG6	Take into account possible (and possibly erroneous) user inferences by analogy from related task domains.
	SG7	Separate whenever possible between the needs of novice and expert users (user-adaptive interaction).
	GG12	Take into account legitimate partner expectations as to your own background knowledge.

Interaction Aspect	G/S G no.	Generic or Specific Guideline
	SG8	Provide sufficient task domain knowledge and inference.
Aspect 7: **Repair and Clarification**	GG13	Enable repair or clarification meta-communication in case of communication failure.
	SG9	Initiate repair meta-communication if system understanding has failed.
	SG10	Initiate clarification meta-communication in case of inconsistent user input.
	SG11	Initiate clarification meta-communication in case of ambiguous user input.

6.3 Additional Design Considerations

There is a wealth of literature covering speech application design issues, from which certain additional design considerations can be gathered. The following considerations are not specifically covered in Table 6-1:

6.3.1 Interruptible Prompts

Speech applications must minimize the amount of time required to retrieve information: they must be brief, selective about content, and interruptible. An experienced user need not endure a long-winded introduction or menu repetition. [9] This allows for a clean compromise between abbreviated instructions for expert users and lengthier tutorials for novices. Unfortunately, users tend to be more hesitant to interrupt than system designers might expect or desire. It may be that users lack confidence in their ability to recollect the proper commands or simply cannot be bothered to learn them, so it is helpful to keep explanations short. [10]

6.3.2 Prompt Speech Quality

Prompt speech quality interacts with the perceived utility of an application. If both can be used equally effectively, it is rarely the case that synthetic speech is more appropriate than recorded speech. If only a few spoken messages are needed, pre-recorded speech generally is adequate, less expensive, and likely to be better understood. For some data, however, synthesis is essential. Unfortunately, brand names and product names are particularly hard to pronounce correctly as they are not found in dictionaries. [9]

6.3.3 Menu Identification

If a system employs multiple menus, feedback can be a form of navigational cue by which each menu identifies itself before presenting its choices. Such a cue would also be appropriate if the user's utterance couldn't be understood; the system can remind the user what it thinks he or she is supposed to be doing. Similar vocal prods can also be made during the invocation of time-outs such as when the user doesn't make a selection after hearing an entire menu of choices. [10]

6.3.4 Repetition Strategy

If output consists of recorded speech, the application can only replay the recording, possibly doing so at a slower speed. In the case of synthesized text, however, the problem could be a spelling or typographical error, failure of the synthesizer's text-to-phoneme rules, or lack of user attention. One repetition strategy is to speak the same text again at a slower rate (75% of normal speed) in the hope that this enhances intelligibility. An alternative strategy (or a strategy to be used in case of a second repetition request) involves passing the text through a spell checker and combining a slower speech rate with letter-by-letter spelling of any words not found in the dictionary. Most typographical

errors would also fail the spelling check and thus would be spelled out. [10] In order to minimize the need for repetition in case of synthesized speech, the synthesizer should provide a feature to add new pronunciations to the morpheme dictionary.

6.3.5 Interaction Termination

If the user aborts the interaction, partially completed transactions might leave an inconsistent database or an open communication channel. In the worst case, when a user hangs up, the line is assigned to the next person calling in, who gets access to the previous caller's account. Dialogue designs should therefore account for the possibility that communication will cease at any unpredictable moment. [10]

6.3.6 Tone of Voice

It is critical to establish the proper tone of voice in messages and prompts. It is important not to assign blame for errors or problems, as poor message terminology and tone encourages users to blame themselves for problems that occur. [11]

6.4 Particularities of the Toy Shopping Domain

The specific application domain of the WishToys application adds the following considerations:

6.4.1 Speech Synthesis Difficulties

As mentioned above, speech synthesis is not appropriate for out-of-dictionary terms such as brand names. In the toy application domain, such terms are pervasive, including toy names, characters from movies, books, games, etc. Therefore, it is all the more important to obtain audio files with toy names and descriptions rather than rely on text-to-speech synthesis.

6.4.2 Speech Recognition Difficulties

Analogous to the speech synthesis difficulties, there are additional challenges with the recognition of all the special terms from the toy domain. If the recognizer offers the possibility to add new pronunciations, WishToys back-office database software would bring to the administrator's attention new words for which the pronunciation needs to be manually defined on each update of the toy catalogue.

6.4.3 Well-known Peak Traffic Periods

As an additional motivation to shorten the total user interaction time, the system is most likely to be heavily used in the run-up to holidays such as Christmas, Halloween, and Easter (if deployed in North America). Shortening the interaction time will enable more people to use the system, thereby increasing profits around those crucial dates.

6.5 Error Handling and Prevention

Although the cooperativity guidelines in Table 6-1 play an important role in error prevention and repair, the DDP methodology doesn't specifically cover these issues, except in the evaluation phase, where the emphasis is on user errors.

6.5.1 Error Prevention

The following factors are known to influence the error rate:

- **Word Similarity** - Without a thorough understanding of the acoustic representation used by the recognizer, it is difficult to know which words sound similar, but a good first approximation is that words containing similar phoneme sequences are likely to be a source of recognition errors. Ideally, the recognizer should offer a feature that reports similar words in a given grammar. [10]
- **Acoustic Environment** - Noise interferes with recognition, although some recognizers are more sensitive to noise than others. In a noisy environment, special noise-cancelling microphones may assist recognition. This feature may not

be available in a mobile setting, where it would all the more be desirable because of the added probability that the environment will be noisy. Given that a telephone circuit transmits only the portion of the speech spectrum below 3500 Hz, the recognizer must not rely on acoustic cues based on frequencies above this. [10]

- **Talkers' Characteristics** - Recognition simply works better for some people than others; this is what is sometimes referred to as the "sheep-and-goat" problem. Some users, the "sheep," can achieve high recognition accuracy. Others, the "goats," have considerable difficulty with recognition. Many people compensate for errors during recognition by speaking louder, which may cause distortion by overloading the audio circuits, or by trying to speak more distinctly, which may make their speech less similar to what the recognizer expects. [10, 12]

The main focus in minimizing errors should be on the above-mentioned issues. Additional techniques that can be used are:

- **Vocabulary Sub-setting** – Limiting the vocabulary to only those words accepted at a given moment. [10]
- **Language Model Sub-setting** – This is similar in concept to vocabulary sub-setting. The recognizer activates and deactivates portions of the language model (sometimes called "contexts") during the course of the interaction. [10]
- **Explicit Indication of Attention** – This technique is more appropriate for user-initiated recognition applications. One possibility is to have a mechanical switch to interrupt recognition when it is not desired. This approach removes some of the benefits of having speech recognition in the first place. Another approach is to use special voice commands to stop and restart recognition. The disadvantage is that the user is likely to forget that recognition is disabled and often wonder why the recognition is not working. A hybrid mechanical-acoustical approach is to have an array of microphones capable of detecting the directionality of the user's speech. Recognition would be enabled only when the user speaks directly towards the computer. [10]

6.5.2 Confirmation Strategies

The interface designer must consider the consequences of the application performing an incorrect action on the basis of misunderstood input and how to improve chances that the user will notice the error without doing further damage. By setting a high rejection threshold, there is a trade-off between a decrease in the number of incorrectly recognized words and an increase in rejection errors. Whether it is faster to reject many possibly correct utterances or accept errors and allow the user to correct them depends on the nature of the task, the complexity of the input, and the time required for recognition. [10]

There are several different confirmation strategies:

- **Explicit Confirmation** – When an incorrect recognition would cause an irrevocable performance error, explicit confirmation may be required from the user, elicited by echoing the request as recognized. If the user's response is then limited to "yes" or "no," vocabulary sub-setting can be employed to decrease the chance of incorrect recognition. Explicit confirmation is slow and awkward, and it offers the user little recourse except the opportunity to try again without causing damage. [10, 13]
- **Implicit Confirmation** – In implicit confirmation, the application tells the user what it is about to do, pauses, and then proceeds to perform the requested action. The application listens during the pause, thus giving the user the opportunity to cancel the request before any action occurs. Implicit confirmation allows the system to perform more effectively as recognition accuracy improves. The user is required to intervene only in the case of an error; otherwise the request and resulting action proceed smoothly. [10, 13]

Researchers are working on intelligent confirmation techniques, which rely on system knowledge and heuristics. One example is an information-centred heuristics approach,

[14] justified by the fact that it is cheaper on average to repair an incorrect reading a few times than to ask a question every time.

6.5.3 Classes of Speech Recognition Errors

Before looking at error correction strategies, it is useful to list different classes of speech recognition errors: [10]

- **Rejection** – The user speaks a word from the recognizer's vocabulary, but it is not recognized.
- **Substitution** – A word spoken from the vocabulary is recognized as some other word.
- **Insertion** – Extraneous words are recognized because of noise.
- **Timeout** – The recognizer didn't detect speech input within the time allotted for the turn.

If the user speaks a word that is not in the vocabulary, the ensuing error can be seen as a rejection, substitution, or insertion.

6.5.4 User Response to Errors

Once the dialog branches into a perceived error condition, the user may respond in a number of ways, as follows: [15]

- **Repetition** – While this may work in the case of a recognition error, it never works in the case where the user's request is outside of the system's competence.
- **Semantic/Syntactic Substitution** – The technique may work if the rephrasing is substantial.
- **Elaboration** – This may work only if the original request was really brief.
- **Simplification** – A simplified request, necessarily briefer and often involving breaking a compound request into parts, may work in most cases.

- **Ignore Sub-dialog Prompt** – Users ignore a prompt that doesn't offer them acceptable alternatives. This strategy fails, because the application initiates a repair dialogue for the sub-dialog in question.
- **Goal Shift** – The change of goal when faced with an error can happen under at least three different sets of circumstances, with varying degrees of success: (i) The system responded with relevant information, although the response was not what the user requested; (ii) The original request was beyond the system's competence; (iii) The user gave up on the goal and tried something else.

6.5.5 Error Correction

Depending on the class of recognition error (rejection/timeout, insertion, or substitution), the application may be faced with an incomplete utterance, a meaningless utterance, or a valid request that is unfortunately not what the user asked. Research on error correction has shown that users overwhelmingly prefer to speak a complex command and then simply repeat individual incorrectly recognized words. [16] Apart from this ideal scenario, there are several error correction strategies to be considered: [10]

- **Additional Recognition Information** - If the recognizer provides additional information such as the next best match along with confidence scores, the application can evaluate whether the alternative is appropriate when the first choice is not meaningful in a particular context. In a similar fashion, if the recognizer provides confusability information, the application can look for highly similar words when the recognized word is inappropriate in the current context.
- **Alternate Input Modalities** - This strategy consists in having an alternate means of user input, to be used in parallel with speech (i.e. multi-modal input). Research has shown that many different alternative input techniques can be used to enhance the recognition rates and overall usefulness of a speech-based application. [12, 14]
- **Dialogue With the User** - The simplest error correction technique is to ask the user to repeat the request. If the user repeats the request word for word, it is quite

possible that the same recognition error will occur again. If the user instead rephrases the request, different recognition errors may occur, and it is challenging to merge the two requests to decide what the user really wanted. It is better for the application to ask the user one or more specific questions to resolve any input ambiguity. The phrasing of the questions is critical because the form in which the question is asked strongly influences the manner in which it is answered. [17] The goal of efficiency suggests that the questions be brief and very direct. The goal of minimizing user frustration would also suggest asking questions that can easily be answered by a single word because words spoken in isolation are more easily answered than connected speech.

- **Echoing** - Asking a terse question is fast but conveys little information about the words the application assumes to be correct. Echoing consists in offering extensive feedback to the user. Even when the application receives an apparently correct command sequence, echoing it to the user provides an opportunity for the user to take further action. The major drawback of echoing is that it takes time and makes the application much more verbose.

- **Backtracking** - When recognition results in a well-formed and plausible command, an application cannot detect the error and will perform the wrong action in the absence of a confirmation strategy. The user may recover from such an error by backtracking, in which the user simply cancels a request or asks the application to "undo" the action. This is appropriate only when the command can, in fact, be cancelled with little or no penalty. To implement backtracking, the application must remember its state before each round of user input.

CHAPTER 7: Design Specification

7.1 System Diagram

The main feature of the system is the integration of a high-bandwidth GUI with a low-bandwidth VUI in one system, using the same database. This system is readily adaptable to any advances in wireless device bandwidth and graphical capability, and the objective is to have the GUI part of the interface on a mobile device in the future. The system diagram is shown in Figure 7-1.

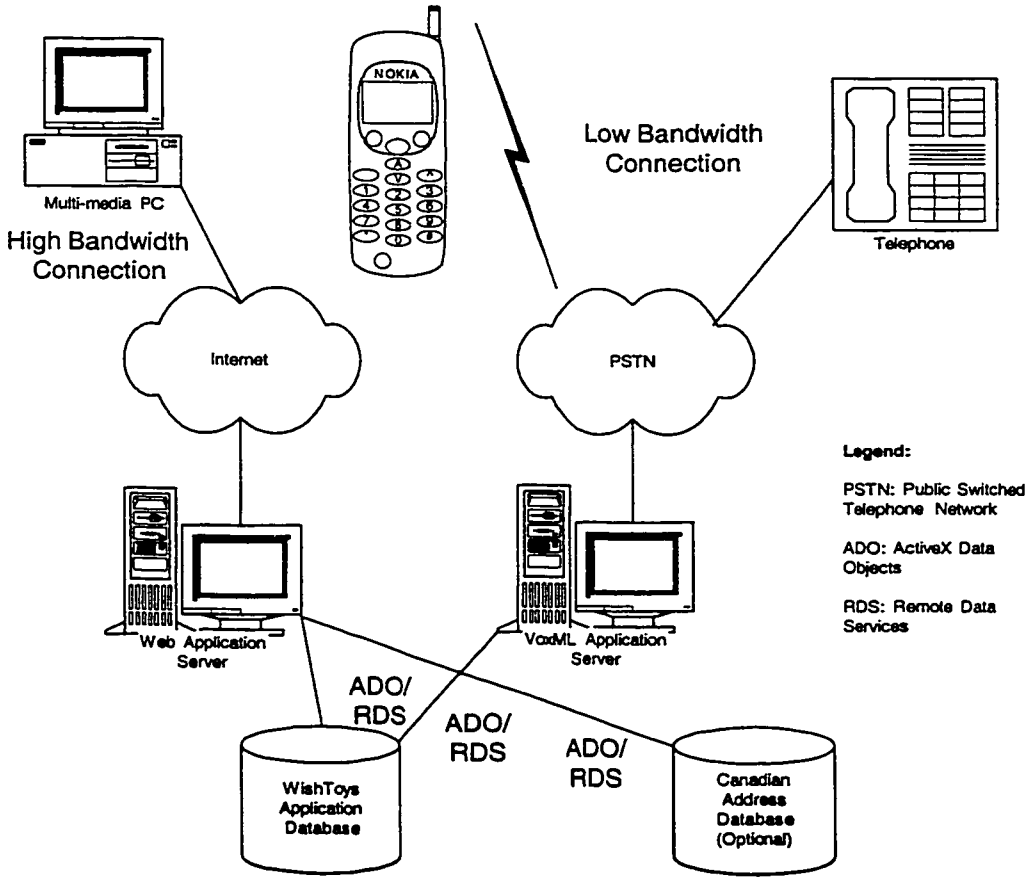

Figure 7-1: System diagram of the WishToys system

7.2 Limitations

This major report focuses on the user interface aspect of the system. The database used in the system contains the minimal set of features and a reduced set of data, sufficient for

usability evaluations. The system's back office functionality (order tracking, credit card verification and transactions, inventory management, etc.), an essential part of a real-world electronic commerce system, is deliberately omitted.

7.3 System State Machine

From a single user's standpoint, the system may be in one of two fundamental states: the system may be waiting for the user to log in, or the user is logged in. This behaviour precludes the possibility of having multiple sessions for the same user. Either the user opens one GUI session to log in (in the Web browser), or the user dials the telephone number of the VUI server and logs in through the voice interface.

In the perspective of family member accounts, it would be allowable to have the family member logged in simultaneously with the main user, as their shopping cart/shopping list/order history operations are kept separate. There is still an enforced limit of one login per customer number.

7.4 VUI Component

A description of the overall dialog flow is followed by a detailed description of each major dialog component. The context-sensitive help functionality is not shown in the diagrams, but it can be seen in the VoxML code provided in Appendix D.

7.4.1 Global Dialogue Flow

The global dialogue flow is shown in Figure 7-2. The flow starts with the Login Dialogue, past which the user is led to the Main Menu. The Main Menu has a variable number of options, depending on whether there are items in the user's Shopping Cart. If

there are such items, the user can review them or choose to proceed to the Checkout Menu.

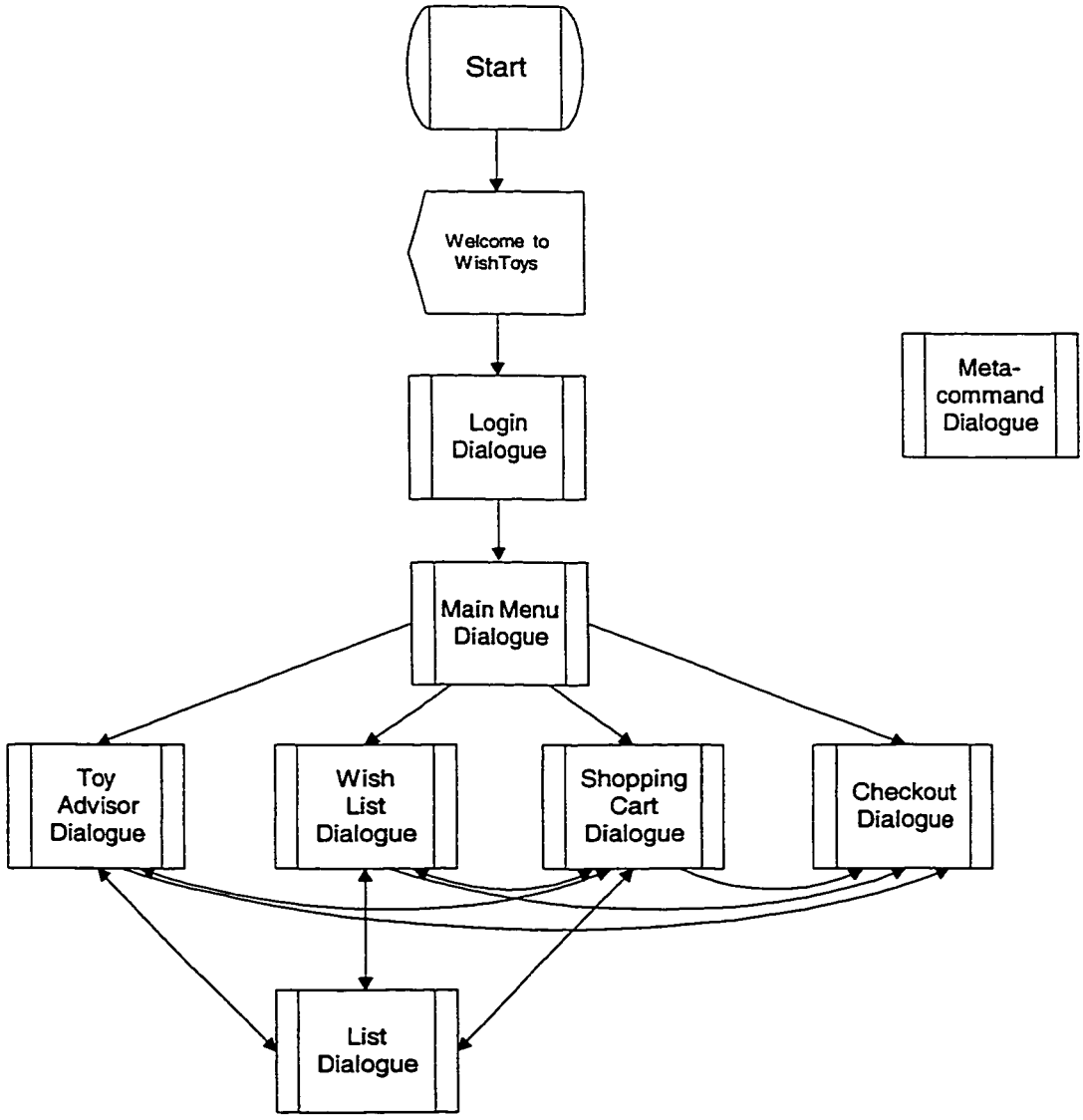

Figure 7-2: Global dialogue flow

Other choices in the Main Menu are Wish Lists and the Toy Advisor. A separate set of choices is available in the Meta Command Dialogue. The system has the following built-in commands: Help, Cancel, Repeat, Exit and Goodbye. Hanging up the receiver is also a meta-command. The difference between Exit and Goodbye is that Goodbye is similar to

hanging up, while Exit applies only to the current branch of the dialogue and returns to the Main Menu.

7.4.2 Login Dialogue

The Login Dialogue, shown in Figure 7-3, consists of two steps: the customer number acquisition, and the PIN acquisition. If the recognizer obtains a valid customer number from the user, the user will be prompted for the PIN. The prompt refers explicitly to the acquired customer number, following guideline SG2 and the requirement for denying false assumptions. If the customer number is not in fact what the user intended to say, the user can say, "Cancel" in order to return to the customer number acquisition step.

For the second step, a personal identification number (PIN) is requested. This step might be replaced by a speaker verification voice-recognition step, where the customer speaks a password that was previously acquired and stored in the customer profile. The voice sample could be acquired through the web GUI, during the creation of the user profile. It could also be acquired using a hybrid login sequence, as follows: first-time users can log in with their customer number and PIN, which leads them to a "Voice Sample Recording" dialogue. Once the voice sample is obtained, the user should log in by repeating the password used for the sample. This would both shorten the login sequence, by reducing it from two steps to one, and simplify the process by requiring the user to remember a password rather than numbers.

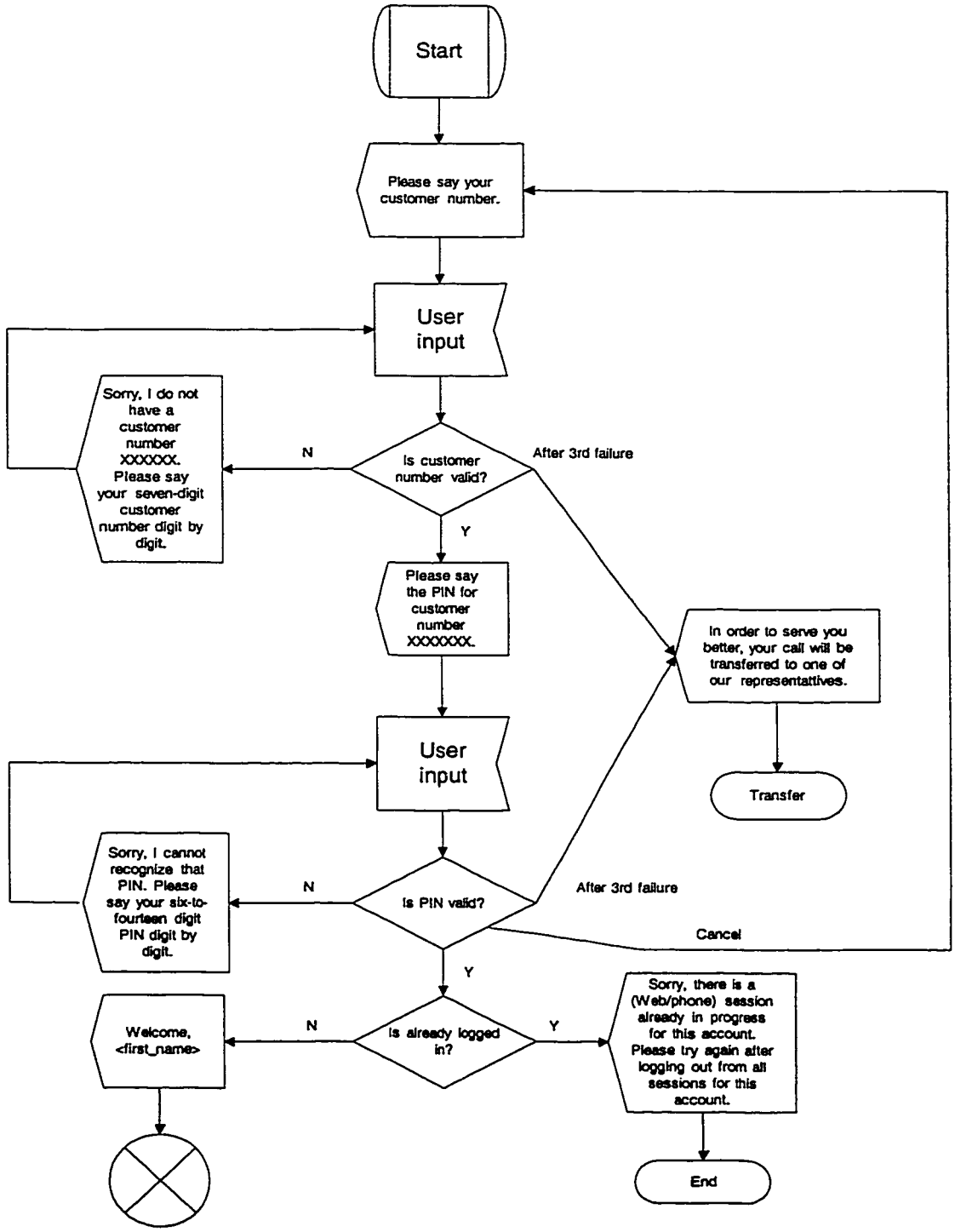

Figure 7-3: Login dialogue flow

After the login sequence correctly identifies the user, the system checks whether there is an active session involving the user's account. In order to preserve data consistency throughout the transaction, the system will refuse to let the user proceed in such a case. In

itself, this refusal of service will probably lower user satisfaction levels, and a strategy to minimize user disappointment is needed. For instance, if the other active session is a web session in the name of a Family Member, that means that vital information such as billing and shipping information will not be affected for the Master User and for other Family Members. In fact, the Master User could use the VUI for everything except that Family Member's wish list, which would be locked. That way, the VUI would only refuse simultaneous sessions by the Master User, with some additional mutual exclusion logic for other cases.

7.4.3 Main Menu Dialogue

The Main Menu Dialogue is shown in more detail in Figure 7-4. There are two possible flows in this dialogue, depending on whether the Shopping Cart is empty or not. In order to satisfy guidelines GG3 and SG10, the system will not reject outright the utterances "Shopping Cart" and "Checkout" even though the cart is empty.

7.4.4 Wish List Dialogue

The Wish List dialogue, shown in Figure 7-5, allows the user to choose a wish list by profile name or customer number. Normally, a Master User would refer to a Family Member's Wish List by the Family Member's profile name, rather than the customer number. Profile names being unique only within one Master User's account, they cannot be used to identify a Wish List within another Master User's account. In such a scenario, the user has to know the customer number of the Family Member from the other account. This is a trade-off between forcing GUI users to choose globally unique (and awkward) profile names and forcing VUI users to keep track of customer numbers rather than profile names.

When the desired profile name/customer number is acquired in the Wish List menu, the user can review the list in the List dialogue or proceed directly to Checkout.

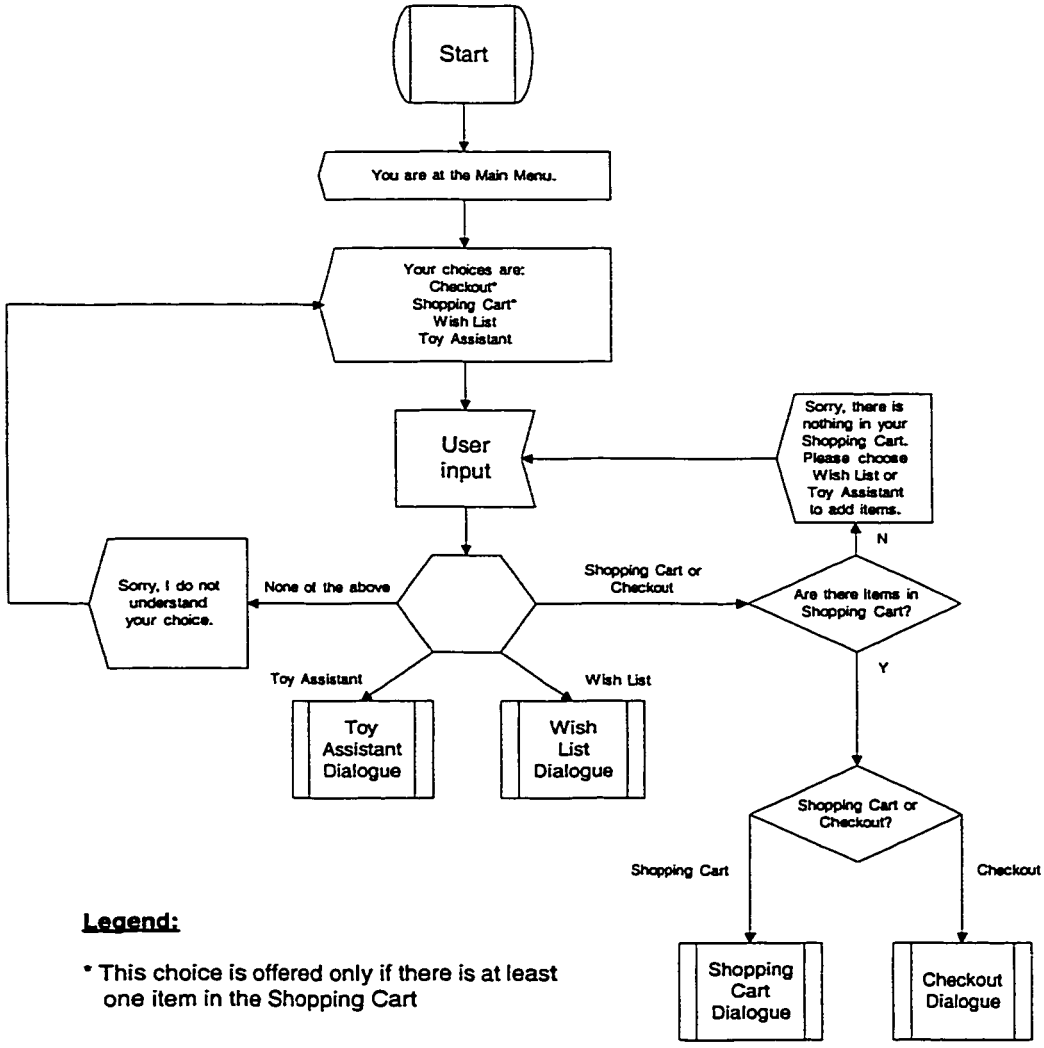

Figure 7-4: Main Menu dialogue flow

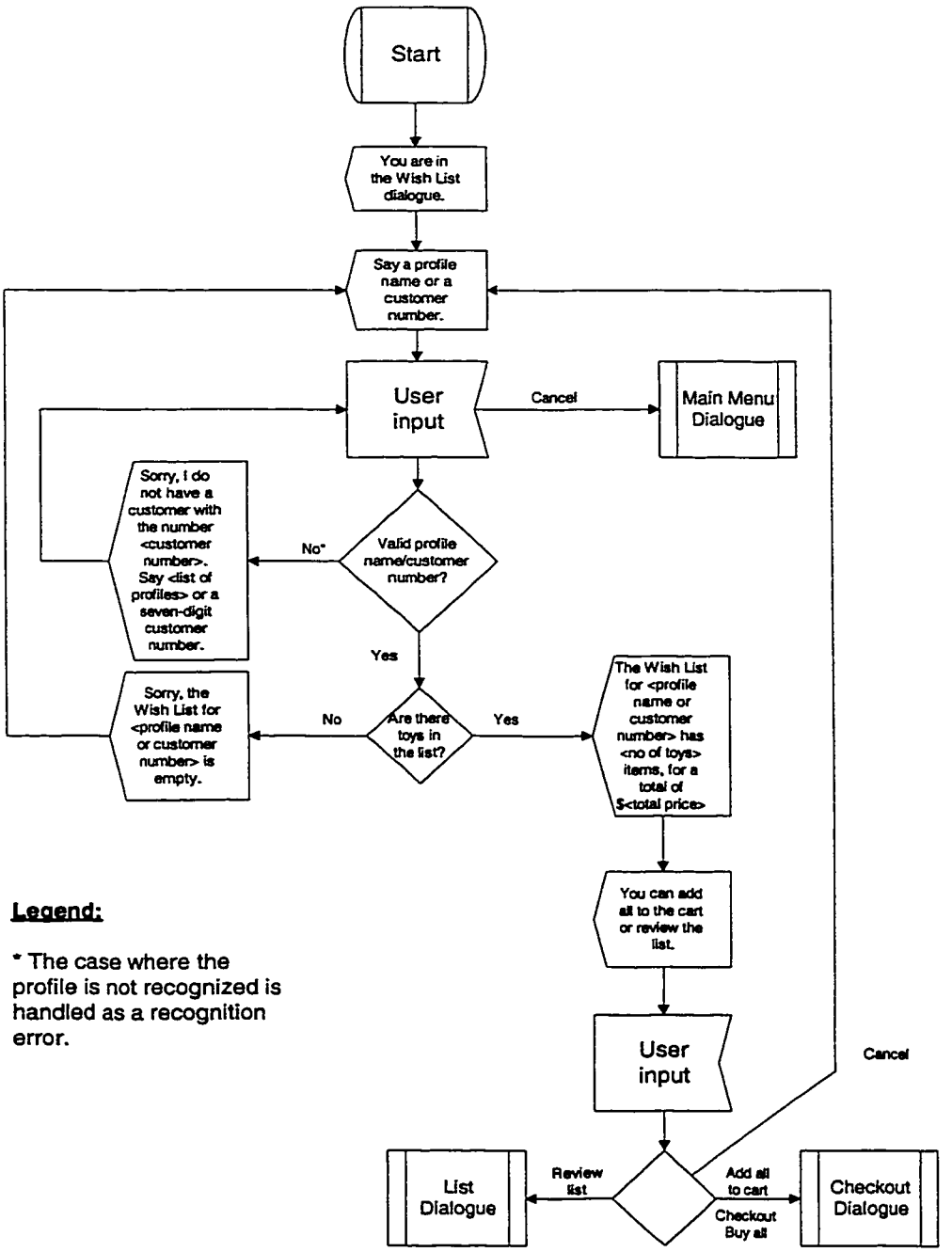

Figure 7-5: Wish List dialogue flow

7.4.5 List Dialogue

The List dialogue, shown in Figure 7-6, is reused for the Wish List, and for the search results in the Toy Advisor. The dialogue offers a generic list browsing functionality, with the use of the tapering technique to shorten the prompts. Another strategy used in shortening the prompts is not to provide details on a particular toy until explicitly

requested by the user. As the List dialogue is always accessed through another dialogue, the Cancel command acts as a "return to previous dialogue" in this context.

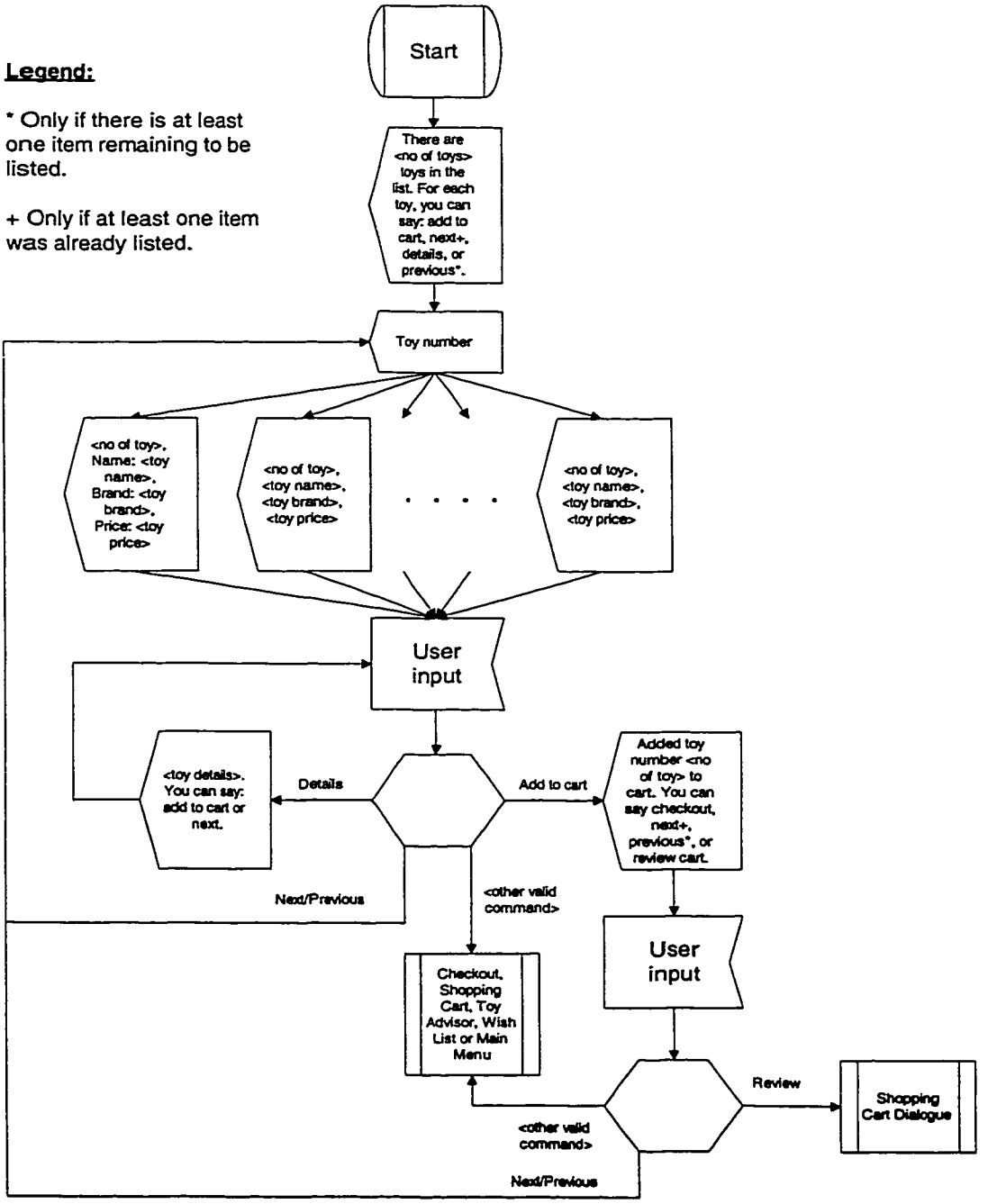

Figure 7-6: List dialogue flow

Unlike in the preceding dialogue flows, the repair dialogues are not shown due to the complexity of the diagram.

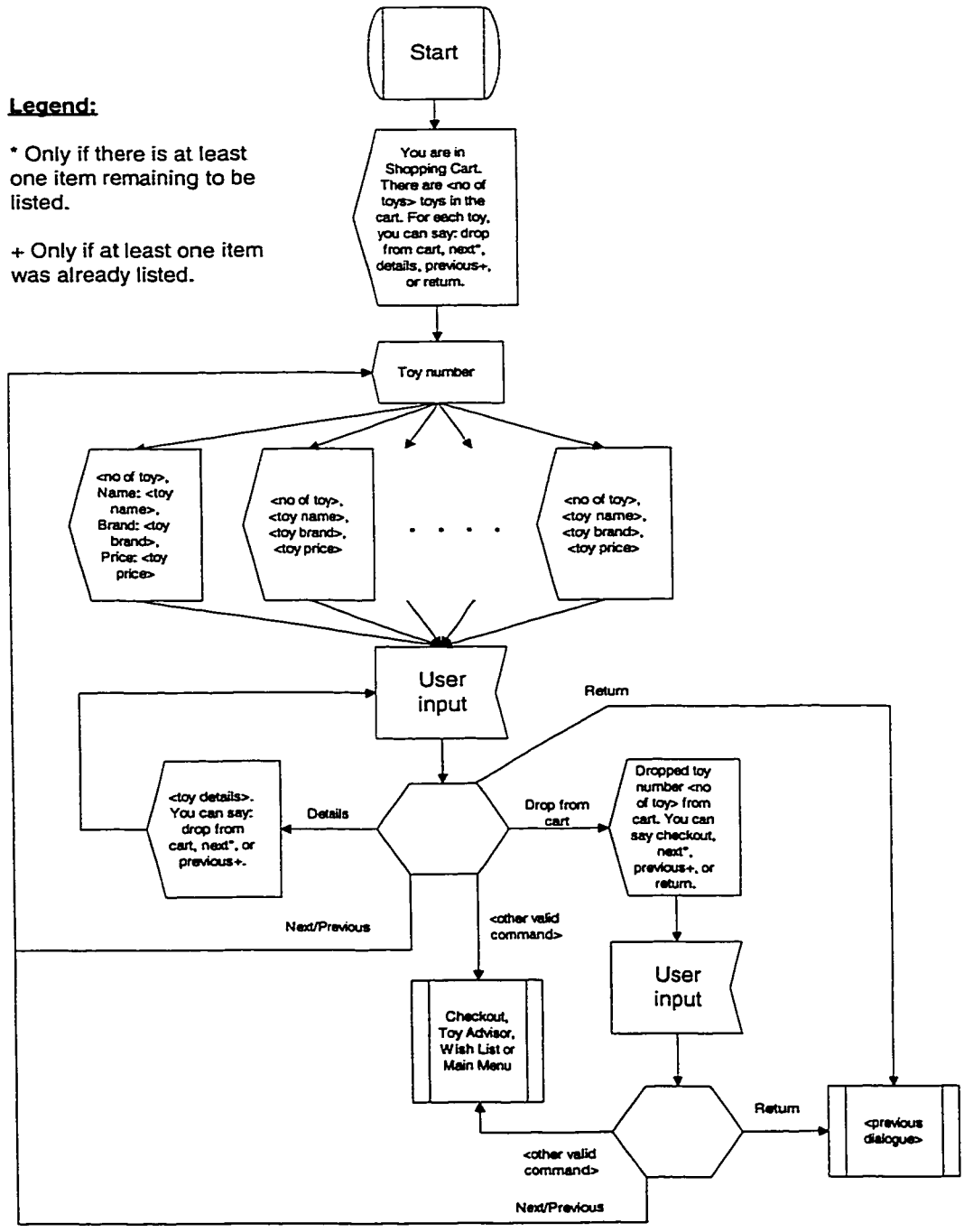

Figure 7-7: Shopping Cart dialogue flow

7.4.6 Shopping Cart Dialogue

The Shopping Cart dialogue, shown in Figure 7-7, is very similar to the list dialogue, with the significant distinction that the user can remove items from the list, rather than add items to the list. Here again, the "cancel" command can be used to return to the

dialogue from which the user entered the cart, and the "return" command can be explicitly invoked for this purpose.

7.4.7 Checkout Dialogue

The Checkout dialogue is shown in Figure 7-8. The design of this dialogue is not complete, reflecting the fact that the checkout functionality in the GUI part was not completed either. The repair dialogues are not included in the diagram due to complexity. Main features of the dialogue are the option to backtrack in an intelligent manner at each turn using the Cancel meta-command, and reuse of shipping information from the GUI application.

Specifically, if the user has chosen a Wish List, the address associated with the profile name or customer number will be the only choice offered. The user can of course say "New address" to speak to a customer representative. The representative can offer to add that address to the user's profile using appropriate back office tools. If there are toys in the Shopping Cart that are not from a Wish List, then the user has more choices, depending on the number of entries in the address book. The shipping address information is used to dynamically determine the shipping cost and whether the order can be delivered same-day or overnight.

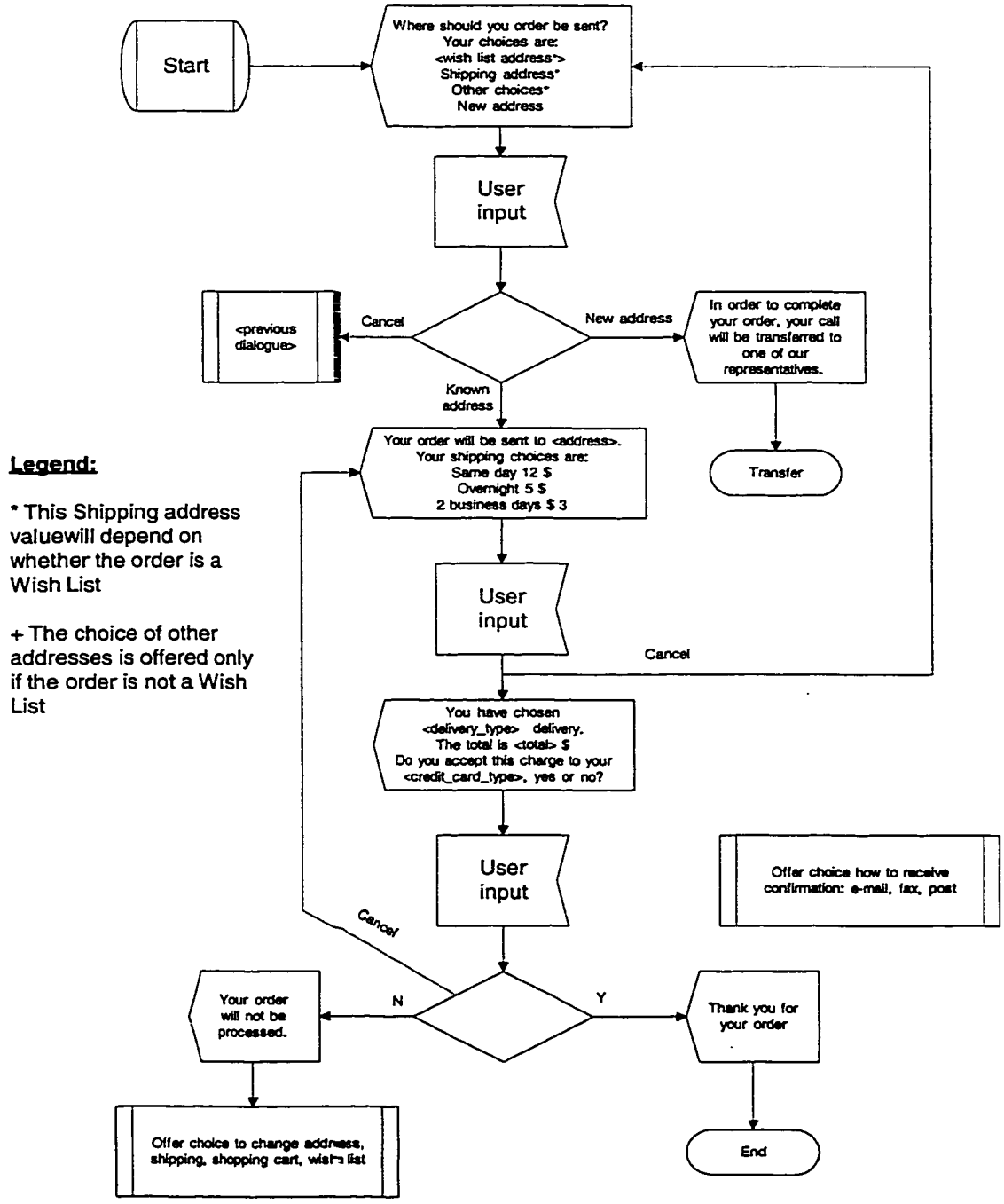

Figure 7-8: Checkout dialogue flow

7.4.8 Toy Advisor Dialogue

The Toy Advisor dialogue, shown in Figure 7-9, mirrors the multi-criterion search feature in the GUI application. The design is partially developed, as far more effort needs to be

invested in fleshing out all the details of the required sub-dialogues for the age, gender, and price.

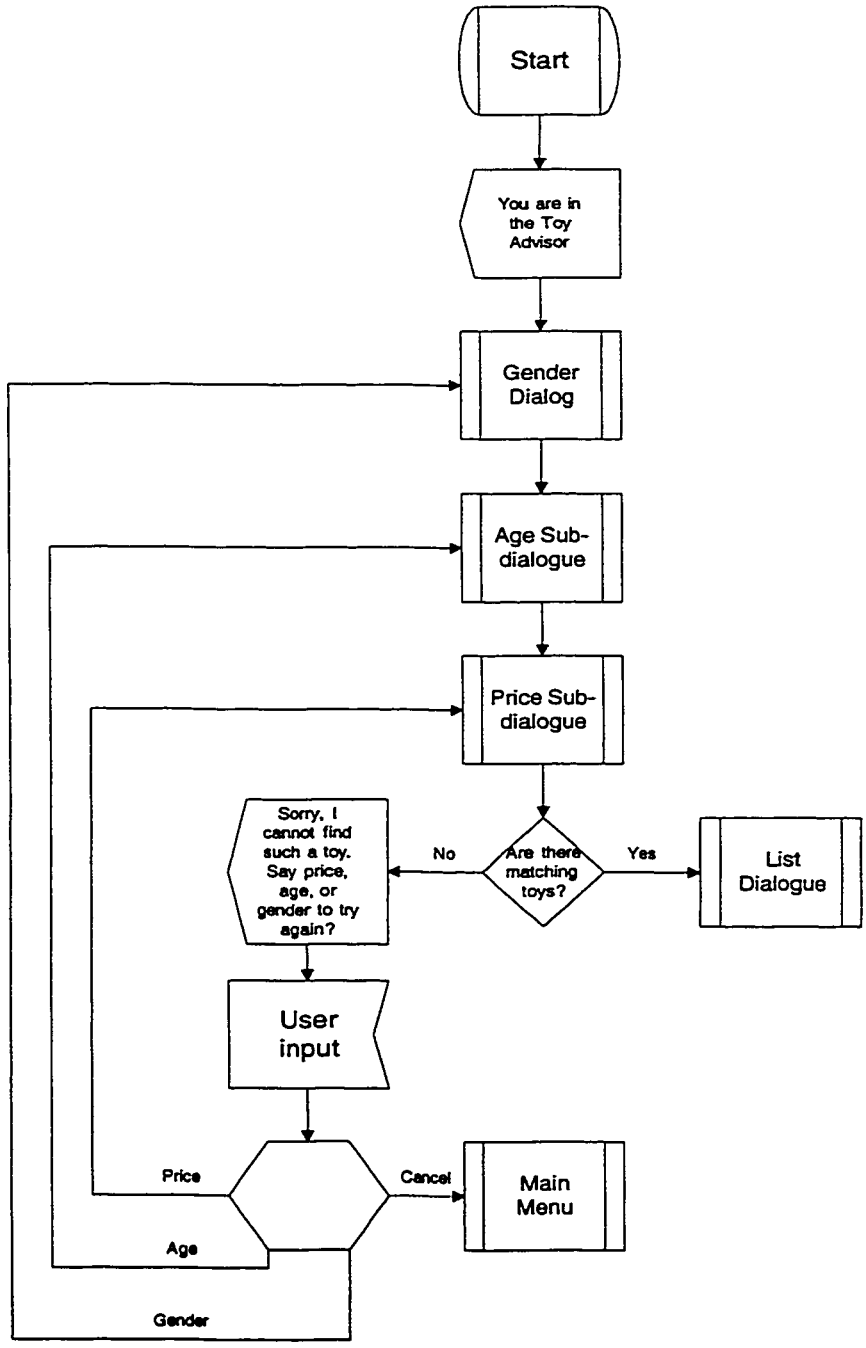

Figure 7-9: Toy Advisor dialogue

7.4.9 VUI Design Highlights

In addition to following cooperativity, error prevention, and other guidelines from the : Analysis and Design section, the WishToys VUI employs additional strategies, as follows:

- Reduce prompt length by applying the tapering technique. [13] Through tapering, explicit prompts become gradually more implicit as the user is trained, conditional on previous successful recognitions. This technique can also be applied to lists of information, where tapering is performed within the list itself.
- Prime the user for terminology used in the VUI. The terms used in the GUI are reused in the VUI in a consistent manner. As the user must first use the GUI to set up an account before accessing the VUI, there is exposure to the application's vocabulary before interacting with the VUI for the first time. Research has shown that user errors decrease rapidly with usage, so this priming effect should go a long way towards improving the recognition success rate and overall usability of the system.
- Offer hints to the user, [13] gradually reducing the frequency and length of hints as usage increases. If a user has not logged on to WishToys in a while, the hints will reappear, but they will gradually disappear at an even faster rate than the initial rate.
- Offer incremental and expanded prompts. [13] In case of a recognition failure, the user hears a longer prompt the second time around. An incremental prompt is used where the list of available options is small (e.g. in a command menu) enough that the choices can be communicated to the user. An expanded prompt is a rephrased prompt, used when the list of available options is large (e.g. customer numbers).
- Offer full recovery for operations. This strategy doesn't reduce errors directly, but it does reduce user frustration when errors occur. By knowing that operations are recoverable, the user will hopefully be less frustrated when faced with errors, which in turn reduces the incidence of errors provoked by user frustration.

7.5 GUI Component

The overall flow of the GUI application is far less rigid than the flow of the VUI application. The inherently "modal" speech dialogues in the VUI are replaced with "modeless" screen dialogues in the GUI. The two initial screens, the Login screen and the UI Level selection screen, are exceptions in the sense that they are "modal." Once past those two screens, the user is in control, and all transitions are possible, as shown in Figure 7-10. In addition to the ability to switch between screens, each screen in the group of Browsing Screens can display toys either in a list or individually in more detail. The different screens of the GUI are illustrated with a screen capture of their implementation in Visual Basic (Figure 7-12 to Figure 7-23).

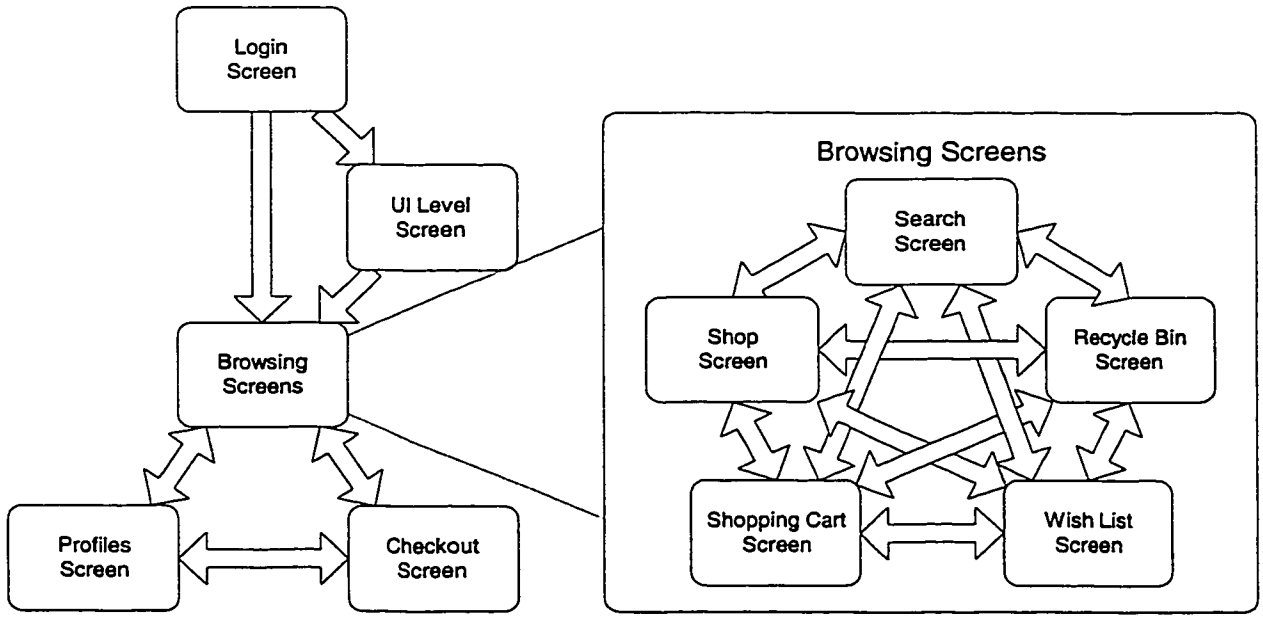

Figure 7-10: GUI global flow diagram

7.5.1 Login Screen

The first screen the user sees is a critical point in any web-based application. With so many sites vying for the user's attention, the first impression the user gets may well be the last impression ever, if the welcome screen is not made sufficiently captivating. There

exist several strategies for attracting users to a specific site, a discussion of which is outside of the scope of this major report (users could be redirected to the site by clicking on paid advertisements, the web address could be featured in "old media" advertisements, etc.). The most striking feature of the WishToys site is the Shopping Assistant agent (requirement 20) that appears as soon as the user visits the site.

Figure 7-11: Shopping Assistant agent character

Rather than using the off-the-shelf "Genie" character from the Microsoft Agent SDK, shown in Figure 7-11, the deployed application could use a customized agent character, i.e. some easily identifiable character that is also featured in advertisements for the site, both in new and old media, or a face familiar to the user. A new user would thus encounter a familiar face at the site, and would hopefully be intrigued by the character's capabilities. The agent appears to float on the screen, on a plane between the windowing environment and the user. The contents of the actual application window/web site are shown in Figure 7-12.

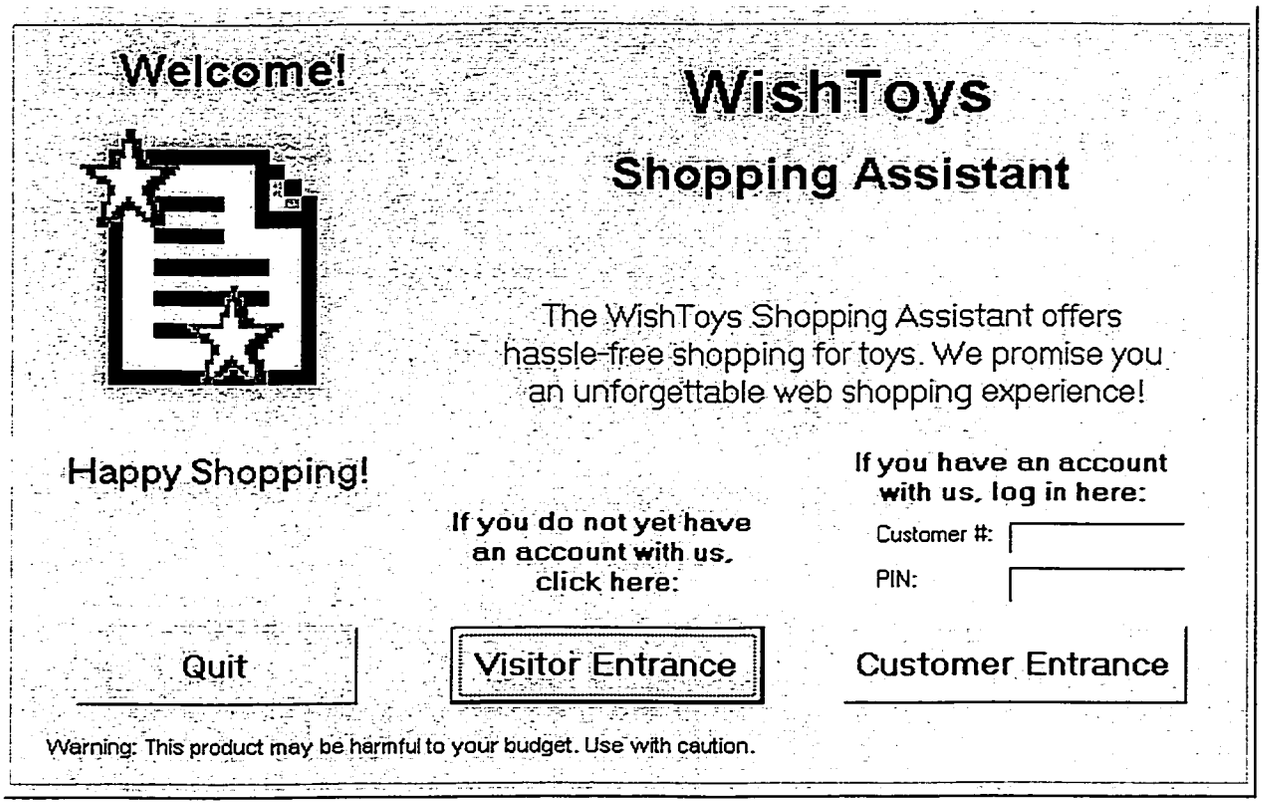

Figure 7-12: Login screen

The Quit button would be removed in a Web implementation, as users browsing the Web normally use other means to leave a given Web site. After the user chooses the Visitor's Entrance (which could be designed to resemble a real entrance), the user interface (UI) level selection screen appears. For users with an account, the next screen is the Shop screen.

7.5.2 UI Level Selection Screen

Users who are new to WishToys are offered the choice of user interface level (requirement 5). The main categories are child, adult, and senior. Only the design for the adult level is discussed at length in this report. After the Shopping Assistant has determined the user interface level, the agent's character would assume one of the three available sets of animations and behaviours. Assuming that the custom character is

initially in the "adult" mode, its behaviour would necessarily change for the "child" and "senior" modes. However desirable it may be, this switch of the agent's behaviour poses a usability problem. Hopefully, user interaction with the agent at the welcome screen would be short enough not to result in a noticeable and confusing change of the agent's behaviour within the application.

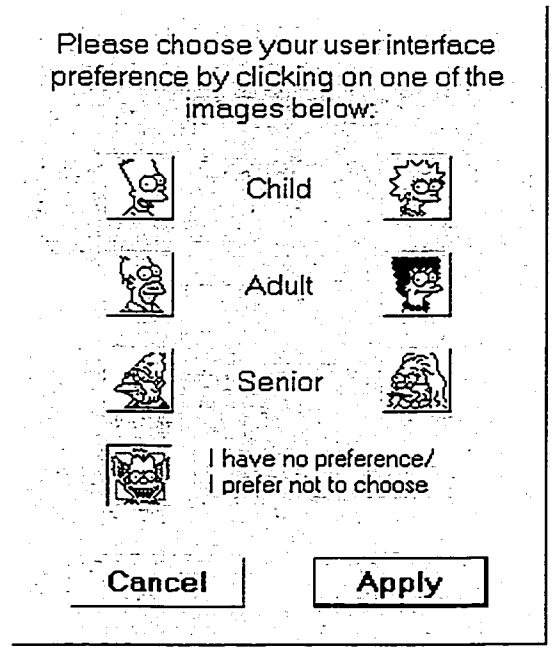

Figure 7-13: User Interface Level Selection screen

The corresponding screen, seen in Figure 7-13, offers the choice of seven images. The current images are borrowed from the Fox TV show "The Simpsons," with Crusty the Clown representing the "prefer not to answer" option (equivalent to the adult level). For obvious legal reasons, this choice of images needs to be replaced with custom artwork in the final application.

7.5.3 Shop Screen

The Shop screen is the first screen the user sees after logging in (not counting the user interface selection screen). The left part of the screen is the "Outlook bar" component

titled "Shop for..." The "Outlook bar" has the On Special icon, the Best-Sellers icon, and as many named icons as there are major toy brand names in the catalogue, such as Barbie, Matchbox, etc. The On Special icon refers to toys currently on sale (where the word "Special" was used to side-step the cheapening connotation of "Sale"), while the Best-Sellers group shows best-selling toys. This last icon could have been called Favorites, but Favorites is already used throughout the Windows 95/98/NT interface in a different manner. Following the Windows analogy, a Favorites icon would imply that it is a set of the user's favourite toys, while the Best-Sellers icon implies a set of toys that is generally not modifiable by the user. The Wish List functionality replaces the Favorites folder in this application, but there is still a subtle difference. A case can be made that the set of the user's truly favourite toys includes some toys that the user has already bought. Hence, these toys do not need to be stored in the Wish List.

To the right of the main Outlook bar there is another Outlook bar, "Subcategory," which usually contains named icons representing general toy categories such as dolls, games, etc. Depending on the context, the Subcategory bar may also contain the On Special, Best-Sellers, and brand icons. The fact that the second bar is shorter is an indicator of the relative importance of the two. To browse toys in a given combination of main and sub-categories, the user just needs to click on any icon in the Shop for... and then possibly on a Subcategory icon. By default, clicking on a Shop for... icon shows the set of corresponding Subcategory icons, and shows the matching toy(s) for the topmost subcategory. This results in the listing of "Best-Sellers On Special" being shown as soon as the user has logged in. The Shop for.../Subcategory hierarchy is shown in Figure 7-14,

while the Shop screen is shown in Figure 7-15. This design is based on research on the optimal number of menu items in Web page design. [18]

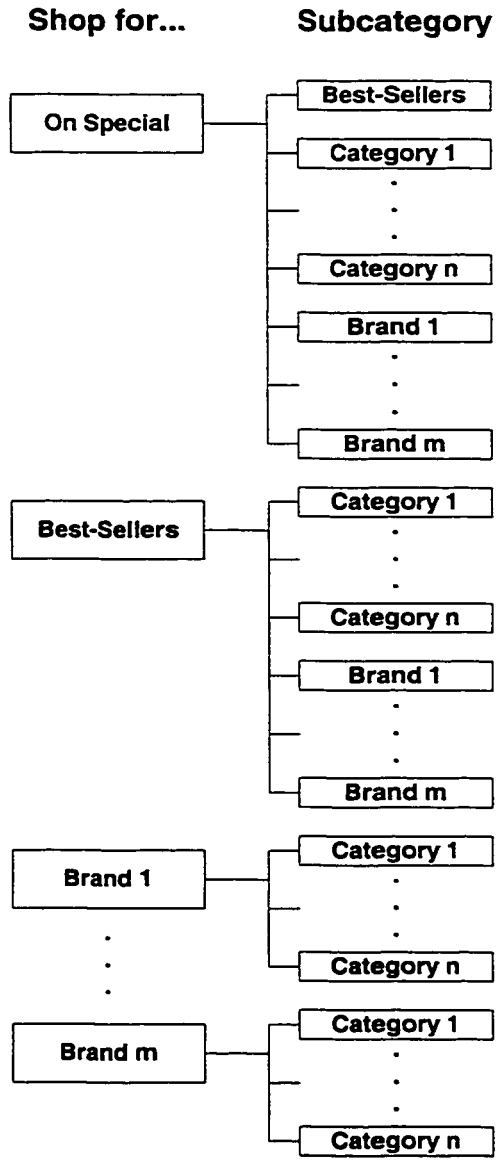

Figure 7-14: Shop for.../Subcategory icon hierarchy

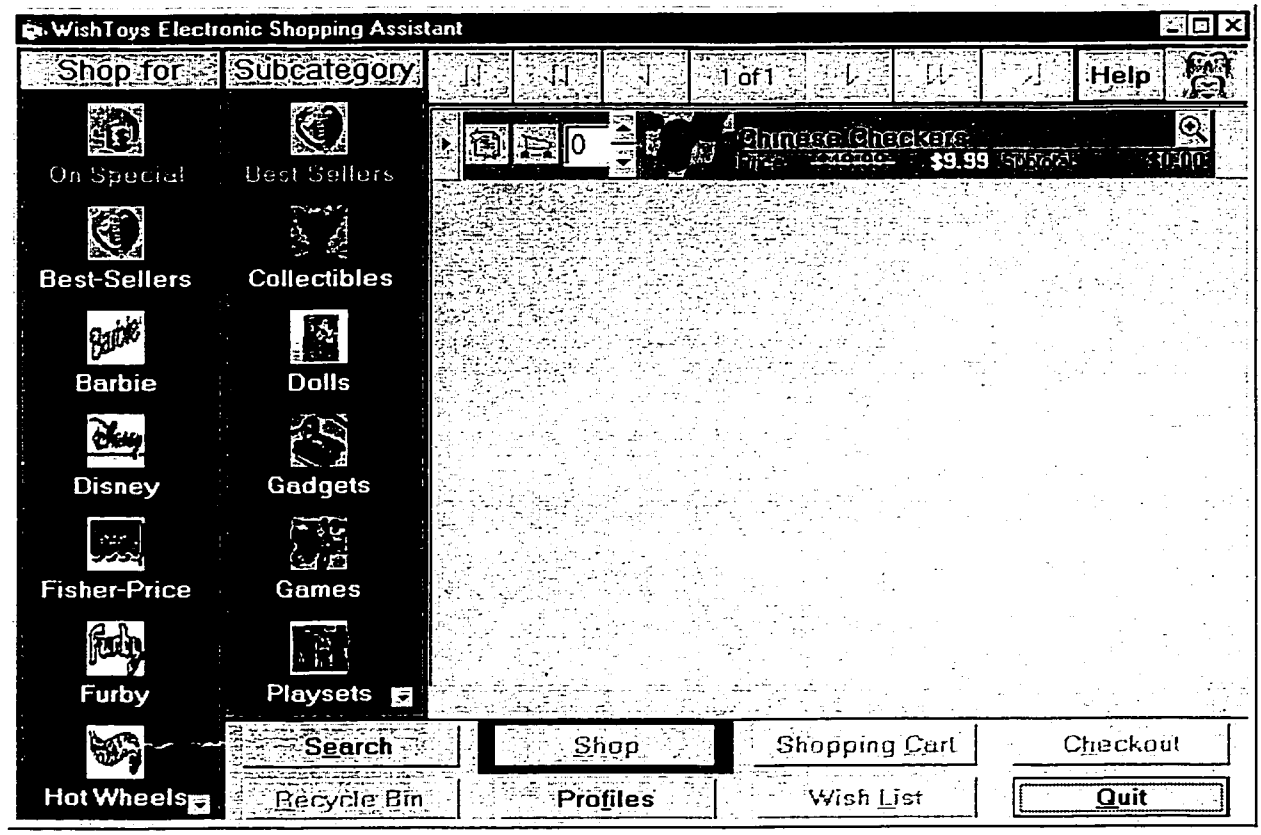

Figure 7-15: Shop screen

The available commands are all legal combinations of Shop for.../Subcategory, even if there are no matching toys. For instance, clicking on "On Special" and then on "Dolls" fetches the list of dolls currently on sale, while clicking on "On Special" and then on "Barbie" fetches the list of Barbie dolls on special. In the above examples, the "Best-Sellers" subcategory was the default, but this could be made to be individually configurable as per the user profile. If the user is more interested in another category of toys, this could become the default category, based on certain criteria enforced in the back office user behaviour tracking functionality. On the other hand, the store may want to change the default in order to interest the user in other toy categories, based on the store's marketing philosophy, in the same manner in which supermarkets rearrange their aisles every so often.

The right side of the Shop screen shows the results for the Shop for.../Subcategory selection. The results may be shown either in a list view (several items per screen, horizontally arranged) or a detailed view (one item per screen). The list view is the default view, but the user can force the detailed view to be the default view by clicking on a checkbox in the Profile screen. If there are no results, a helpful message appears in the space reserved for the results. The message will suggest an alternative course of action to get results, depending on the current context.

Every operation with the Outlook bar icons causes the Shopping Assistant agent character to speak the current Shop for.../Subcategory setting, e.g. "Showing best-selling collectibles." The text that is spoken also appears in a cartoon-like speech bubble. The agent sound control button in the upper right-hand corner of the screen, shown in Figure 7-16, can be used to toggle between the speaking and silent modes.

Figure 7-16: Speaking and silent agent graphical checkbox images

The issue of the order of the icons in the Shop for.../Subcategory icons, as well as the order of the toys in the corresponding lists of matching toys, merits a special mention. A suggestion for the ordering of items can be made as follows:

- Items on special are listed by price (highest price is first), for maximum savings;
- Best-selling items are listed by generated revenue (the number of units sold, multiplied by the unit price), to represent the real economical impact of a toy;
- Toy subcategories are listed in alphabetical order;
- Brand names are listed in alphabetical order.

However, only a few icons are visible at any time in the Outlook bars, making for strong competition for visibility. The same stands for the list of matching toys. The Shopping Assistant back office software could have a feature that imposes a specific order on the items in the various categories and lists. This would especially be important for brand names, where the economic model is the same as with paying for shelf space in supermarkets. Another possibility is to rank some of the items according to the respective profit margin, in an attempt to maximize profits for the store.

In designing the Shop screen, it became apparent that users wishing to see all dolls would encounter a usability issue. This is due to the Shopping Assistant's general strategy of minimizing the number of items in any list of matching toys obtained using the browse functionality. This strategy results in the "Show All" functionality being purposefully absent from the Shop screen, assuming that this functionality would not be requested as much as sale items, best-sellers, and brand name/toy category combinations. Instead, the "Show All" functionality is found only in the Search screen described further below. As presented, the double Outlook bar browsing mechanism could be adapted to any merchandise (clothing, tools, etc.) that can be classified into a limited set of categories and brands.

7.5.4 Search Screen

In addition to offering browsing by categories and by brands of toys, which is in itself a form of searching, the Shopping Assistant provides an explicit search functionality with any combination of the following five criteria:

- Category (e.g. Dolls, Games, etc.);
- Price range;

- Age;
- Gender;
- Word(s);

In the usability evaluations performed for COMP 773, there were many comments on the search functionality. In particular, the parameters appeared complicated to set and the controls were obscure. The present interface is an attempt to simplify the interface as much as possible while retaining the above five search criteria.

The Search screen, shown in Figure 7-17, is complementary to the Shop screen, offering quick searching according to the first four criteria. The "search by keyword" feature is reserved for situations where those four criteria are still not sufficiently precise or are not applicable. As in the Shop screen, there is an Outlook bar on the far left, titled "Search for ...," but in this case the bar offers named icons for "All", as well as for individual categories, such as Dolls, Games, etc. The choice of categories doesn't include brands, for two reasons. Firstly, the Brands are available in the Shop for.../Subcategory bars in the Shop screen, and the user can browse the brands by category, which narrows down considerably the selection of matching toys. Secondly, the brand names inherently imply information about the price range, age and gender of the toys, thereby rendering the search criteria mostly redundant. If the user really needs to search for a specific brand of toy while enforcing other criteria and/or keywords, the brand name becomes just another keyword.

To the right of the Search for... bar, a search form with four controls and two buttons replaces the Subcategories bar of the Shop screen. The search form has a keyword text

box, a custom-designed double slider for the price range, a custom-designed slider for the age, and a hybrid "check-at-least-one" checkbox pair for the gender.

7.5.5 Price Range Control

The price range double slider is particular in that the extreme settings are 0 and 100 $, where the 100 $ mark takes on the meaning of "Any" price. There are two reasons for this: in the context of toys, the percentage of toys costing more than 100 $ doesn't warrant a finer gradation of the scale, because the incremental gain in scope reduction would not justify this. The other reason is that, for all intents and purposes, a user interested in toys above the 100 $ mark is likely to be interested in toys at any price. The upper value could be made configurable, the value being a function of market data.

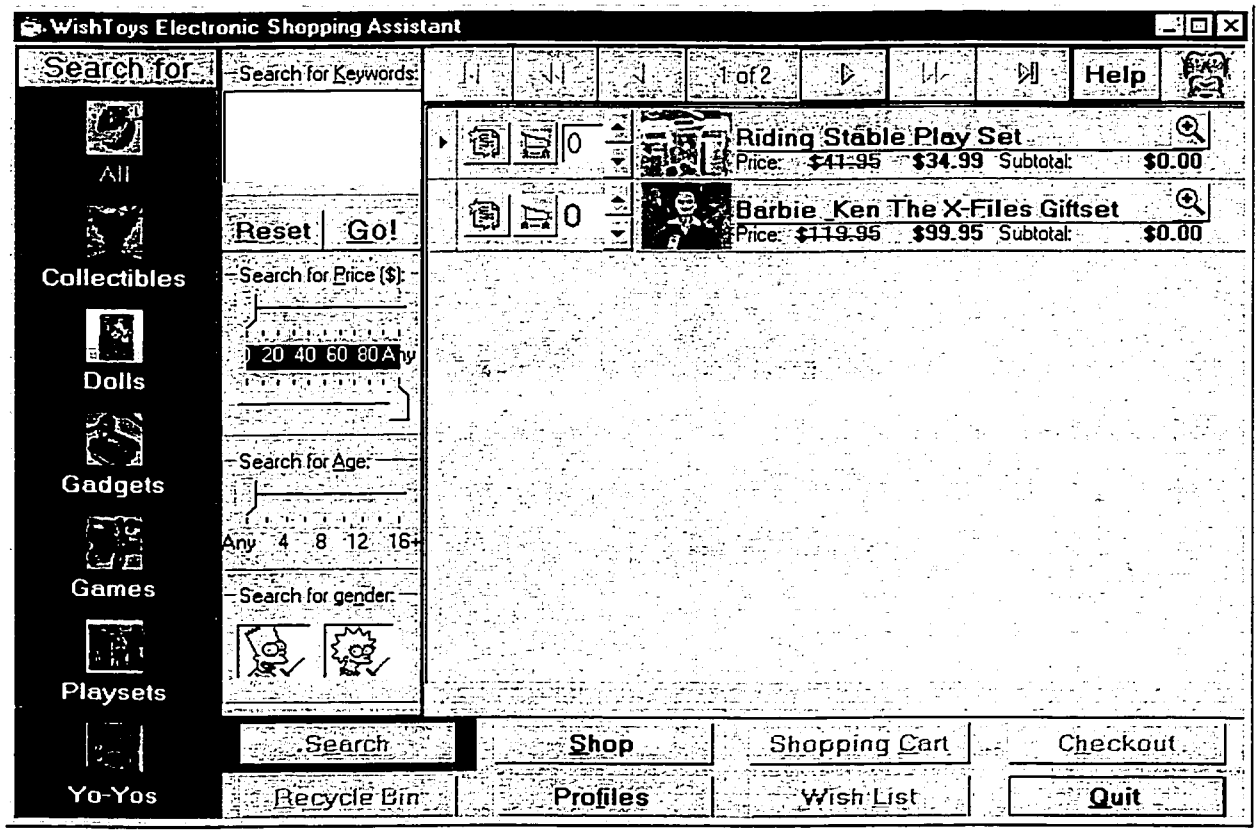

Figure 7-17: Search screen

The double slider has another feature: the two sliders cannot be reversed, as they are programmatically linked, and the minimal distance between them is configurable (the value used in the current version is 10 $). This eliminates impossible searches (where the lower limit is below the higher limit), and reduces the occurrence of empty searches, as the price range cannot be too narrow. The actual query string takes into account another peculiarity of the application domain. A search for toys in the 10 to 20 $ price range will actually return matching toys in the 9.95 to 20 $ range, compensating for the effect of "psychological" pricing.

7.5.6 Age Slider Control

The age slider is specific in two ways. The lower limit is "Any" rather than 0, while the upper limit is "16+", accounting for toys that would be interesting for adults. If a precise age is given, the user having a particular child (or self) in mind, the search will return toys where the toy's specified age range includes the target age. This is an opposite type of search to the search by price range, where the toy has a given price, while the user more often has a price range in mind.

7.5.7 Gender Selection Control

In previous versions of the Shopping Assistant, the gender selection control attempted to fine-tune the gender selection in differentiating between toys that are just for one or the other gender, and toys that are not gender specific, such as chess or playing cards. Usability evaluation participants deemed this to be too complex and ultimately confusing. Therefore, the gender selection defaults to both toys for boys and for girls, while the choice can be restrained to either toys just for boys or toys just for girls. In all cases, the results include toys suitable both for boys and for girls. As there is no generic check-at-

least-one checkbox control, additional code had to be written around default checkbox controls.

Compared to the Shop for... functionality, the search functionality is specifically adapted to toys, but it too could be made more generic and adapted to other domains. Other merchandise could have different price ranges, may or may not be gender-specific, or could have other criteria, such as height, weight, etc. However, the general idea behind this implementation of the search interface is that a user is able to easily restrict the selection of matching toys using a minimum of mouse-and-keyboard and/or voice commands, by setting one or more of the most pertinent domain-specific criteria.

In the occurrence there are no toys that match the search criteria, the Shopping Assistant could suggest enlarging the scope of the search, according to the following rules:

- If an age is specified, the age could be increased by a year. The age criterion has the least impact on the suitability of the toy. The reason is twofold. First, the toy may well be suited for the child, even though the age range doesn't match perfectly. In any case, the child will eventually reach the right age for the toy. Second, the limiting factor is most frequently the price of the toy;
- If a price range (or a maximum price) is specified, the upper limit could be increased by 5$. The price criterion has a positive impact on the suitability of the toy, because the general rule is that a better toy will carry a higher price. Nevertheless, the price criterion should be modified only if the age criterion is not modifiable;
- If the gender is specified, there is not much to be done, because the toys for one gender already include toys suitable for both genders (e.g. the game of chess). The Shopping Assistant will not suggest changing the gender criterion.

In any case, if the search doesn't yield results after a second attempt, the Shopping Assistant could offer to notify the user by e-mail when a toy matching the criteria becomes available. The marketing department could then analyze the search criteria and introduce appropriate new offerings.

7.5.8 List View Mode

The results of the browsing operations are displayed on the right side of the respective Shop or Search screen. There are two modes for the display: List View and Detailed View. In List View mode, the screen shows basic data for several toys at a time, while the Detailed View mode presents all available data for an individual toy. Basic data displayed in List View, as can be seen in Figure 7-17, includes the following:

- Name of the item (or shortened name, where applicable);
- Image (or a detail of the full image);
- Price;
- Sale price (if applicable);
- Indication whether the toy is in the Shopping Cart;
- Quantity of this toy in the Shopping Cart;
- Subtotal (Quantity multiplied by the regular or sale price);
- Indication whether the toy is in the Wish List;
- Indication whether the toy is a best-seller;

Each row of the List View also offers Wish List and Shopping Cart management operations for the corresponding item. Users can easily add/remove the item to/from the Wish List, as well as add/remove any number of items to/from the Shopping Cart. If an item is removed from the Shopping Cart or the Wish List, it can be later recovered in the

Recycle Bin screen. The Wish List management feature appears in the form of a graphical checkbox button, while the Shopping Cart management feature can be accessed both using a graphical checkbox button and a textbox with an up/down button combination control. Individual subtotals are dynamically updated on every change of the Shopping Cart contents. Each row also has a "Zoom In" button, which switches the interface to the Detailed View mode.

The available commands for navigation and for Wish List and Shopping Cart management are shared with the Detailed View. The navigation commands are designed to work under the following assumptions:

- The first item is always selected as soon as a list is displayed.
- Multiple selections in lists are disallowed.
- There is always an item selected in any list.

An exception to this rule is the order item summary grid in the Checkout screen, which is not considered to be a list in the sense that is used in this section. The grid is a static representation of the contents of the shopping cart, and the contents of the grid can only be modified indirectly by modifying the shopping cart contents.

7.5.9 Detailed View Mode

As already mentioned, the Detailed View mode shows all the available information for one toy at a time. The increased screen real estate is used mostly to display the full image of the toy, in an attempt to maximize the visual effect. Compared to the List View, the additional data displayed is as follows:

- Age range;
- Gender;

- Catalog No.;
- Product No.;
- Year;
- Whether batteries are required;
- Whether assembly is required;
- Contents and/or description;

Also, in this mode, the Shopping Assistant will speak the accompanying marketing description (the so-called "blurb") for the toy, if the agent is displayed. The screen also has a "Zoom Out" button to switch back to the List View mode.

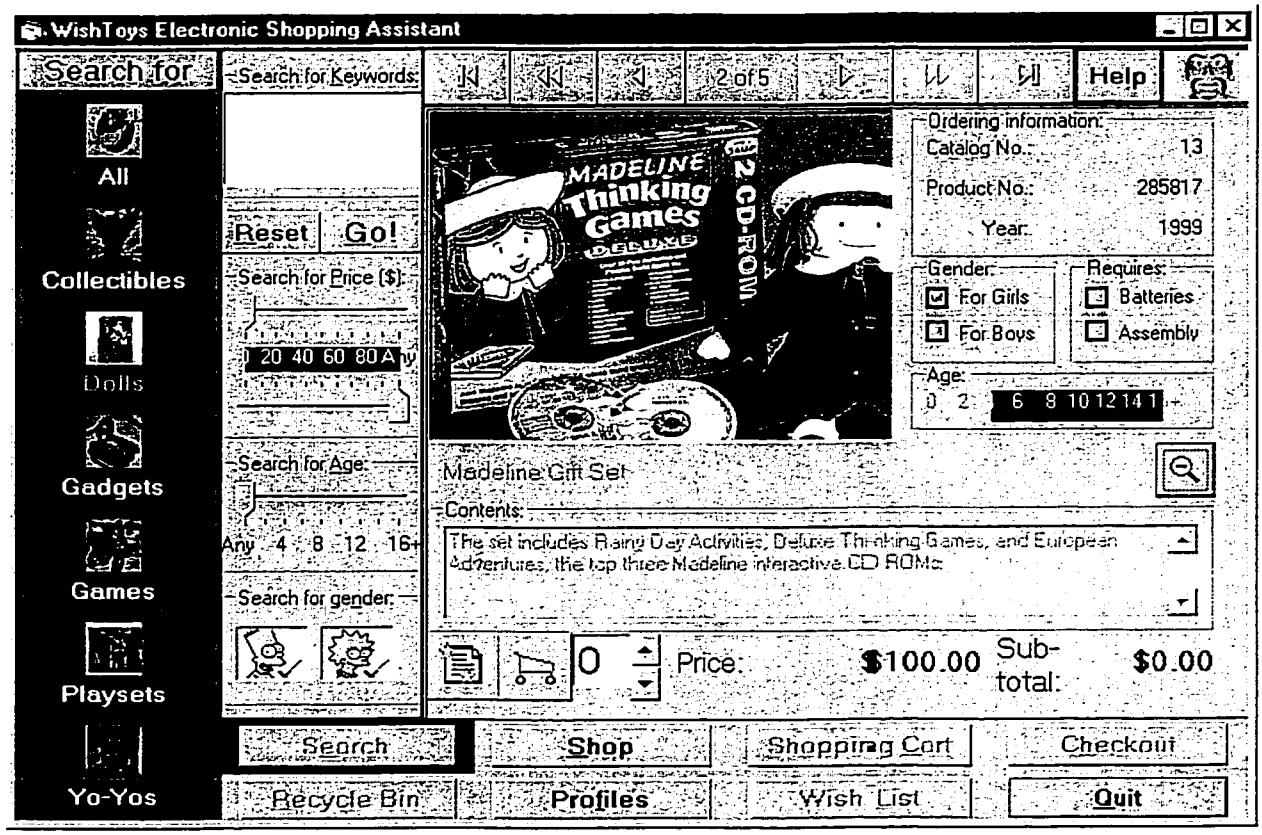

Figure 7-18: Detailed View mode

The design of the user interface of the individual rows of the List View and the design of the Detailed View screen is different for each user interface level. In the child level, the emphasis is on the graphical aspect, and the Shopping Cart management features may be entirely absent if the profile does not have any purchasing privileges. There are relatively less items on display in the List View because the image is bigger. In the adult mode, the full functionality is on offer, with the most items displayed in the List View. In the senior mode, the emphasis is on easier mouse manipulation, as well as on making the displayed text more readable. Therefore, both the buttons and the text are larger, causing the number of displayed items to fall compared to the adult interface level.

7.5.10 Profiles Screen

When the user selects the Profiles screen for the first time, the application generates a new customer number and informs the user about this through the agent. The Profile's name is Master User, and it cannot be changed. The user is required to fill in the necessary information (the field names in bold) for the Master User profile, before creating Family Member profiles. The Add Family Member... button is not available as long as the Master User profile is not created. The credit card information is common to all profiles under the same Master User. All other information is specific to each user: shipping address (if not same as the Master User's), spending limit, contact information, customer number, PIN, preference for the default viewing mode, and user interface level.

Although it was entirely within the capabilities of the technology, the choice as to which toy categories and/or brands should be displayed at the top of the list is not offered. This information can be inferred from the user's habits in using the interface, and it would become part of the internal user model for that user. On the other hand, the storefront

must reserve the right to rearrange the categories/brands as it sees fit, as discussed elsewhere in this report.

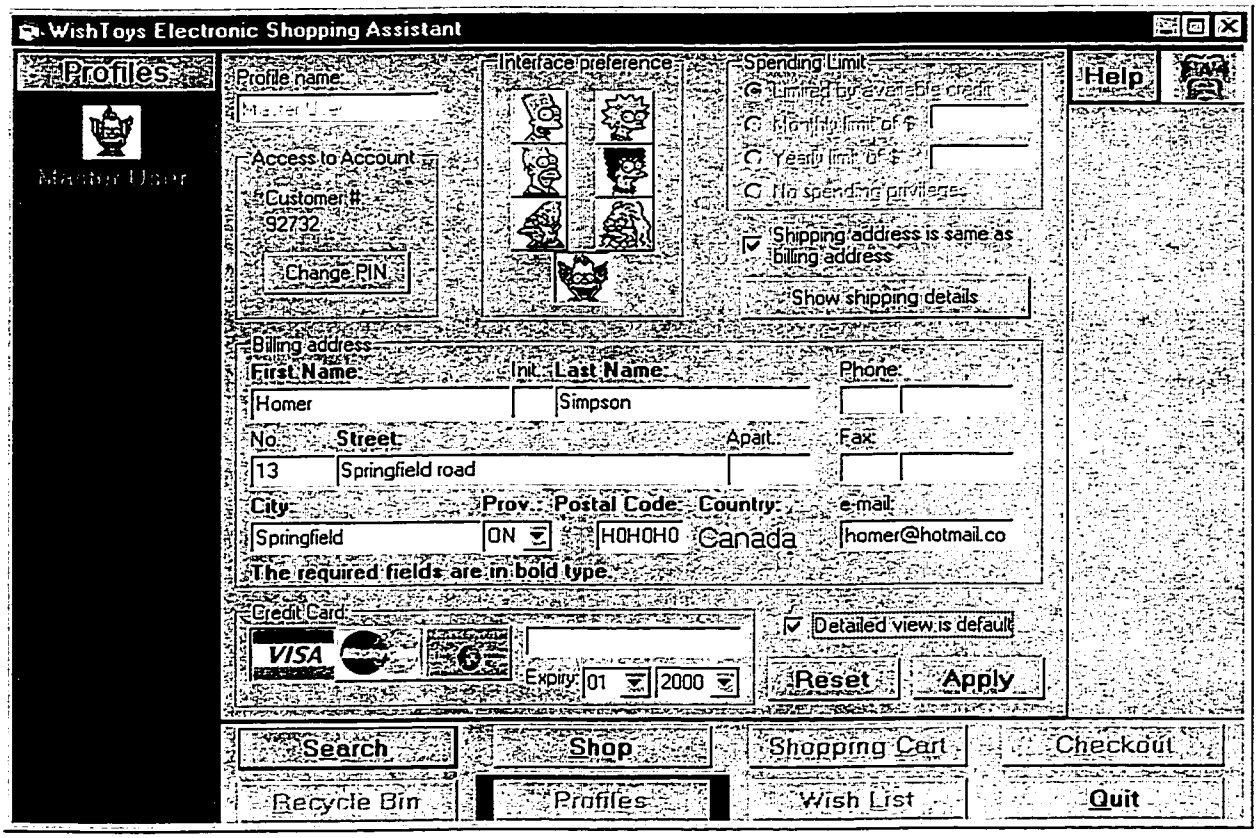

Figure 7-19: Profiles screen

7.5.11 Checkout Screen

A mock-up of the Checkout screen is shown in Figure 7-20. The main features of the screen are the Address Book that uses the Outlook bar component, the invoice-like summary of the user's order with a calculation for the total cost of the order, and the two shipping-related sets of controls. The two shipping-related controls determine the confirmation mode and the shipping mode. In the example in Figure 7-20, the only confirmation mode is by e-mail, because the user only provided an e-mail address. Otherwise, a set of radio buttons would have appeared. The shipping mode is selected

through the use of radio buttons, and the corresponding shipping cost appears in the Shipping text box in the group of text boxes showing the total cost of the order.

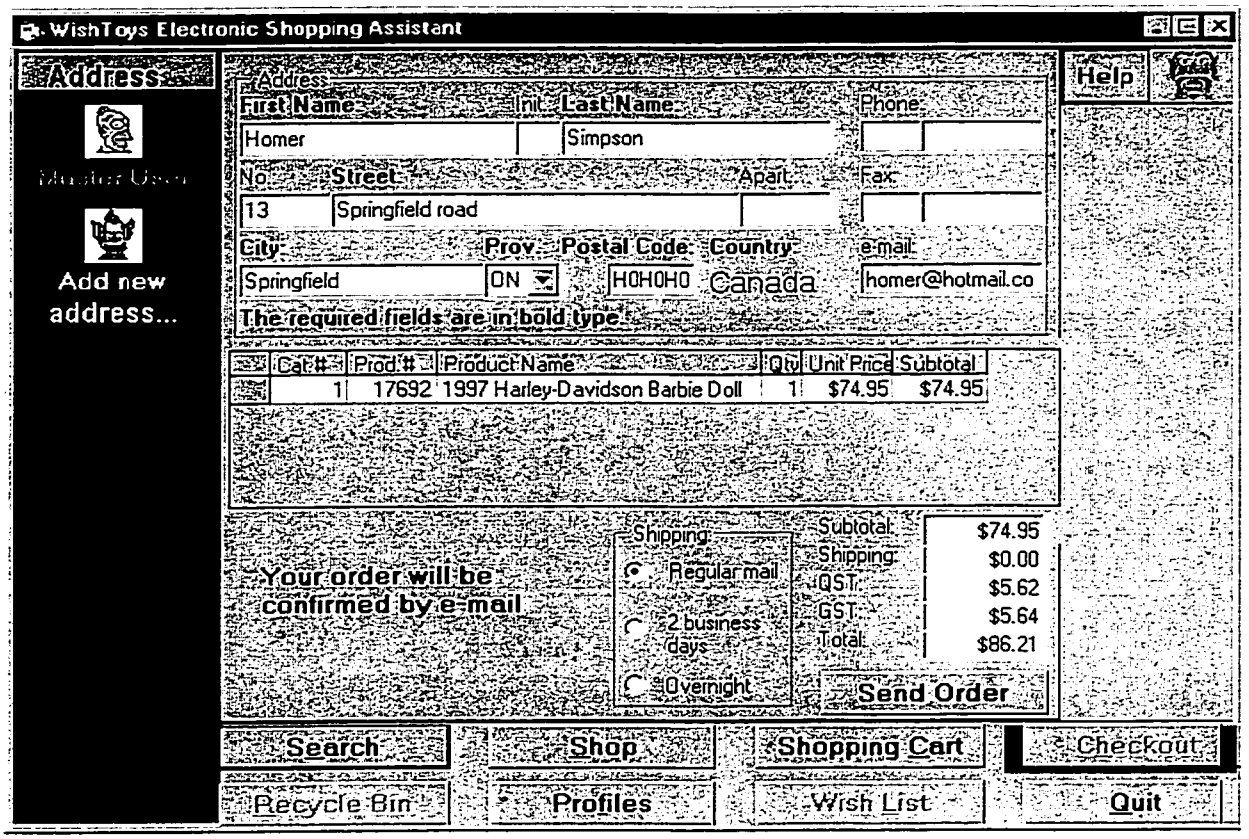

Figure 7-20: Checkout screen

7.5.12 Address Validation

As with all similar e-commerce applications, the billing address is verified by a remote authentication server along with the credit card information. The billing address has to match the billing address associated to the credit card. As for the shipping address, some sort of validation should also be required, to reduce the possibility of a delayed delivery (and related shipping costs/drop in customer satisfaction) caused by an incorrect address. If the Shopping Assistant back office is connected to an up-to-date database of Canadian addresses, the address validation feature can be greatly improved. For lack of access to

such a database, this functionality is not implemented in the demo application. The suggested order of validation would be:

- Postal code (The postal code can be validated with the existing set of postal codes, as well as using rules for postal code generation. If the postal code is in the database, it forms a key with the range of addresses, city, and province);
- Province (The choices are restricted by the available options in the list);
- City (This can be validated from knowing the province and/or postal code);
- Apartment number (This is separate from the street number, so as to prevent mistakes where the apartment and street number are inversed);
- Street number and street (The possible choices are limited by the postal code or by the city, although previsions must be made for new street names);
- First and Last Name (This validation step comes last, because by now it is possible that a name is already associated with the address. Here especially, the application should provide for new names)

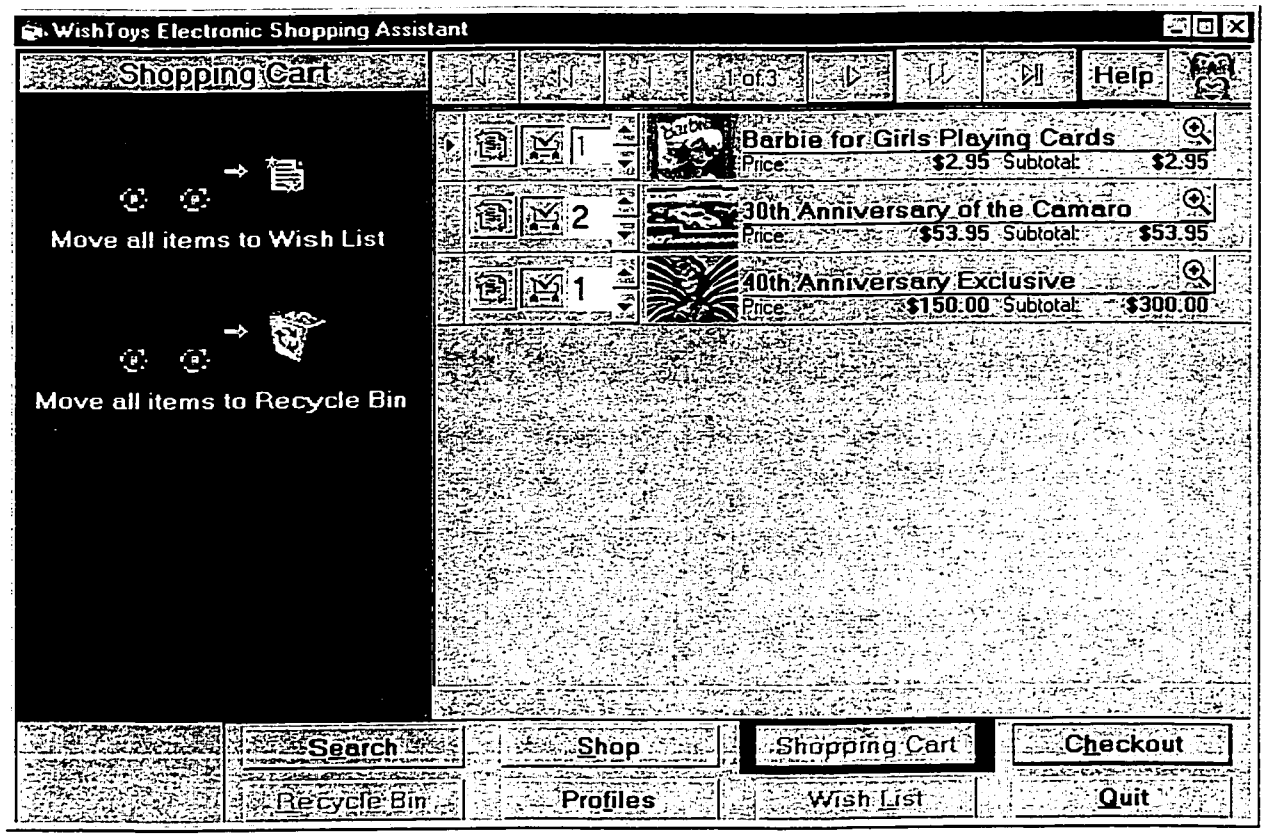

Figure 7-21: Shopping Cart screen

7.5.13 Shopping Cart Screen

The following three screens, the Shopping Cart screen, the Wish List screen, and the Recycle Bin screen share the same general design: the Outlook bar on the left has been widened to accommodate new double-width action button icons, while the toy display is the same as in the Shop and Search screens. The Shopping Cart screen, shown in Figure 7-21, allows the user to transfer all items to the Wish List or to the Recycle Bin in one click. Items can be transferred individually to either of the two using the Shopping Cart and Wish List management operations in the toy display area.

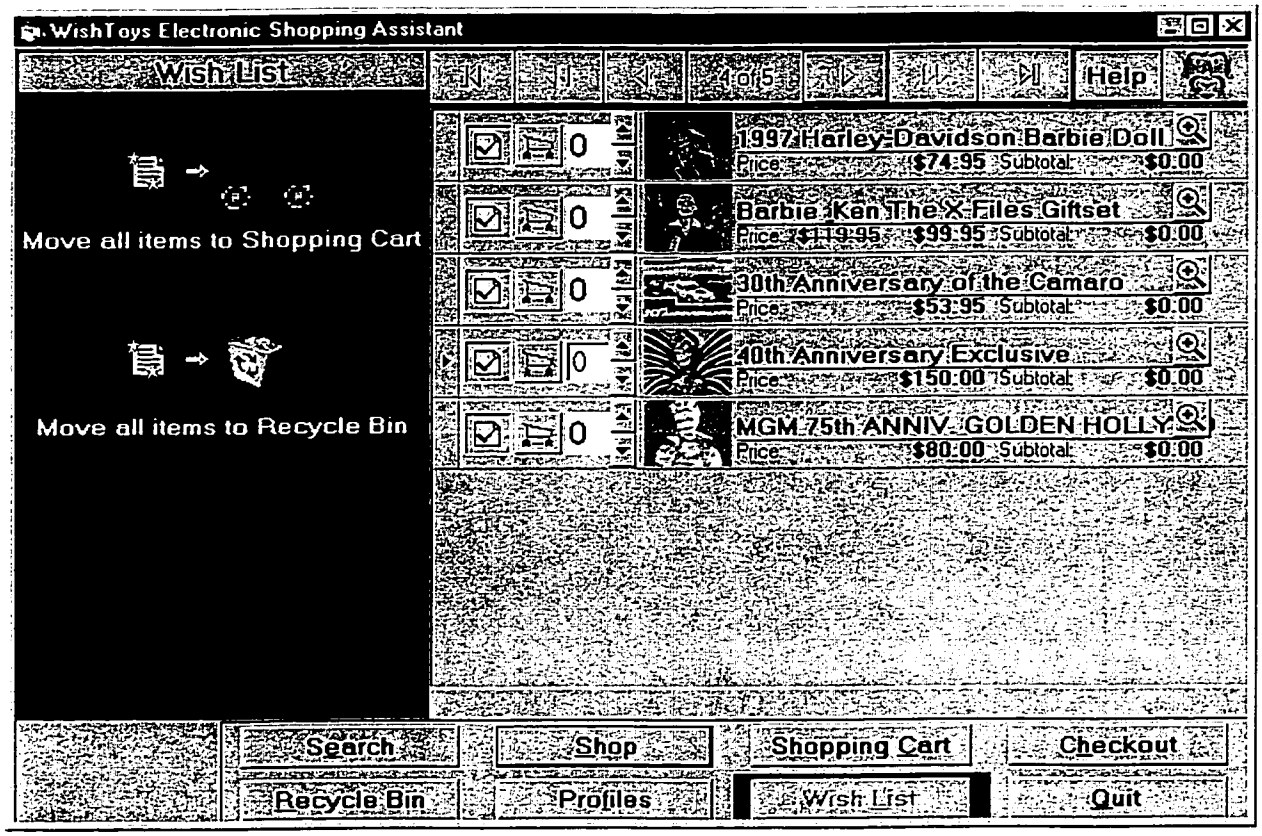

Figure 7-22: Wish List screen

7.5.14 Wish List Screen

The Wish List screen, shown in Figure 7-22, is very similar to the Shopping Cart screen. A possible addition to the Outlook bar could be a "Copy all items to the Shopping Cart"

action. It is unclear whether an item should be removed from the Wish List when it is added to the Shopping Cart. The present design allows for the item to be both in the Shopping Cart and Wish List, but this could be changed after obtaining user feedback.

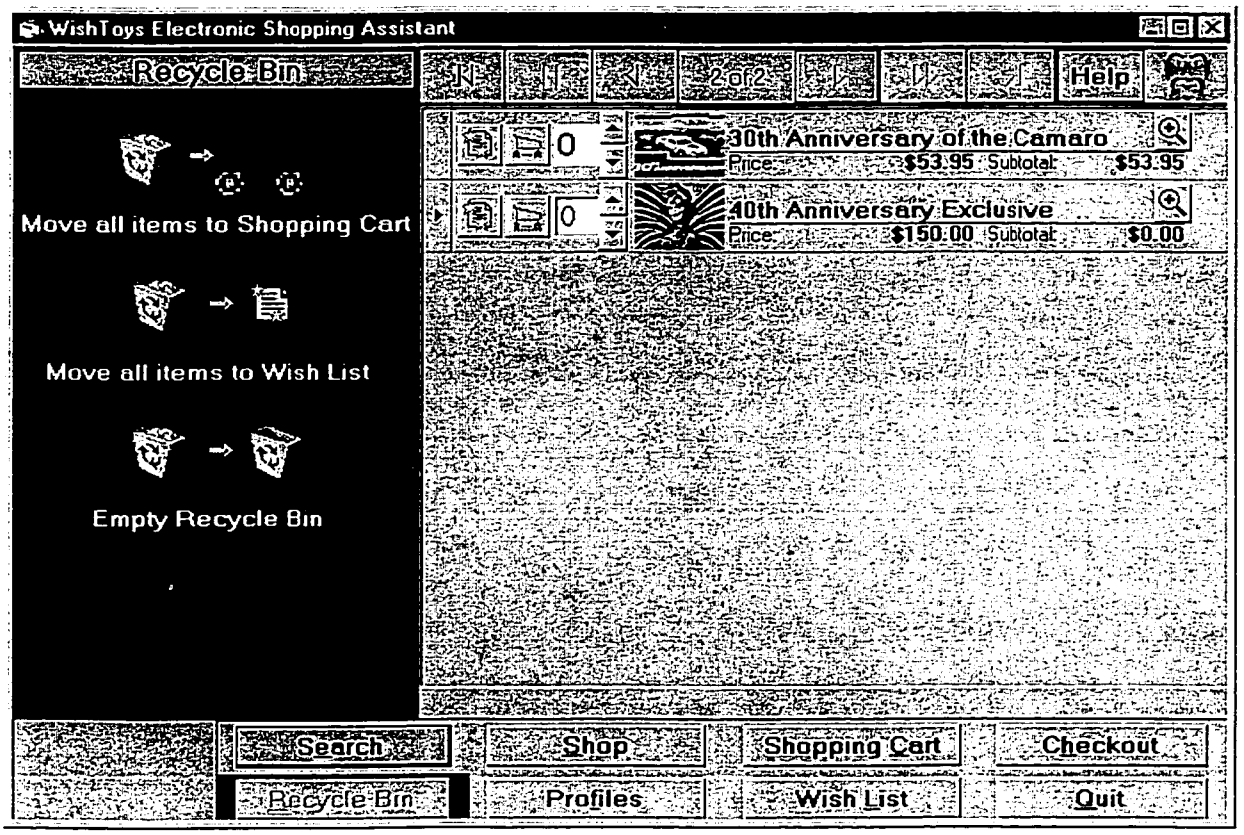

Figure 7-23: Recycle Bin screen

7.5.15 Recycle Bin Screen

Whenever an item is removed from the Shopping Cart or Wish List, it can be retrieved from the Recycle Bin screen, shown in Figure 7-23. Items are removed from the Recycle Bin through the one-click actions available in the Outlook bar: Move all to Shopping Cart, Move all to Wish List, or Empty Recycle Bin. The issue whether individual items should disappear automatically from the Recycle Bin when they are individually added to the Shopping Cart or Wish List can be reviewed after obtaining user feedback.

CHAPTER 8: Simulating, Evaluating, and Revising

8.1 Simulating, Evaluating, and Revising the VUI

As the DDP methodology was developed for a large-scale project where resources were not spared, and at a time when rapid speech application development tools were not available, the simulation phase was performed using the Wizard of Oz technique. With the advent of the VoxML speech recognition application markup language (see Appendix C) and the freely available Mobile Application Development Kit from Motorola, the simulation phase of the development of a speech recognition application can in theory be performed with far less cost and effort.

8.1.1 Wizard of Oz Technique

In the Wizard of Oz (WOZ) technique, a human (the wizard) simulates the whole or a part of the interaction model of the system to be developed, carrying out spoken interactions with users who are led to believe that they are interacting with a real system. WOZ is a relatively costly technique because: (i) the wizard needs a significant amount of training and support; (ii) involving experimental subjects, WOZ experiments require careful planning and preparation and take time to run; and (iii) experimental results have to be transcribed and analysed, which takes time and requires skill to benefit further system development. On the other hand, by producing data on the interaction between a (fully or partially) simulated system and its users, WOZ provides the basis for early tests of the system and its feasibility, as well as of the coverage and adequacy of the requirements prior to implementation. [1]

8.1.2 WOZ Set-up and Minimum Requirements

Figure 8-1 below shows the elements that are potentially included in a WOZ set-up. Some of these elements are mandatory while others are optional. The mandatory elements are: a person (the wizard) simulating the interaction model of the system, a subject acting as a user and a subject-wizard interface which conceals the fact that the subject is interacting with a human rather than with a real system. Usually, a simulation support tool representing the interaction model to be simulated, and one or more data collection tools are also considered mandatory. If these are absent, the wizard will have to know the interaction model by heart and data collection can be done by observation only so that no track record will be left after the experiment. All other elements are optional but may provide useful support during simulation. [1]

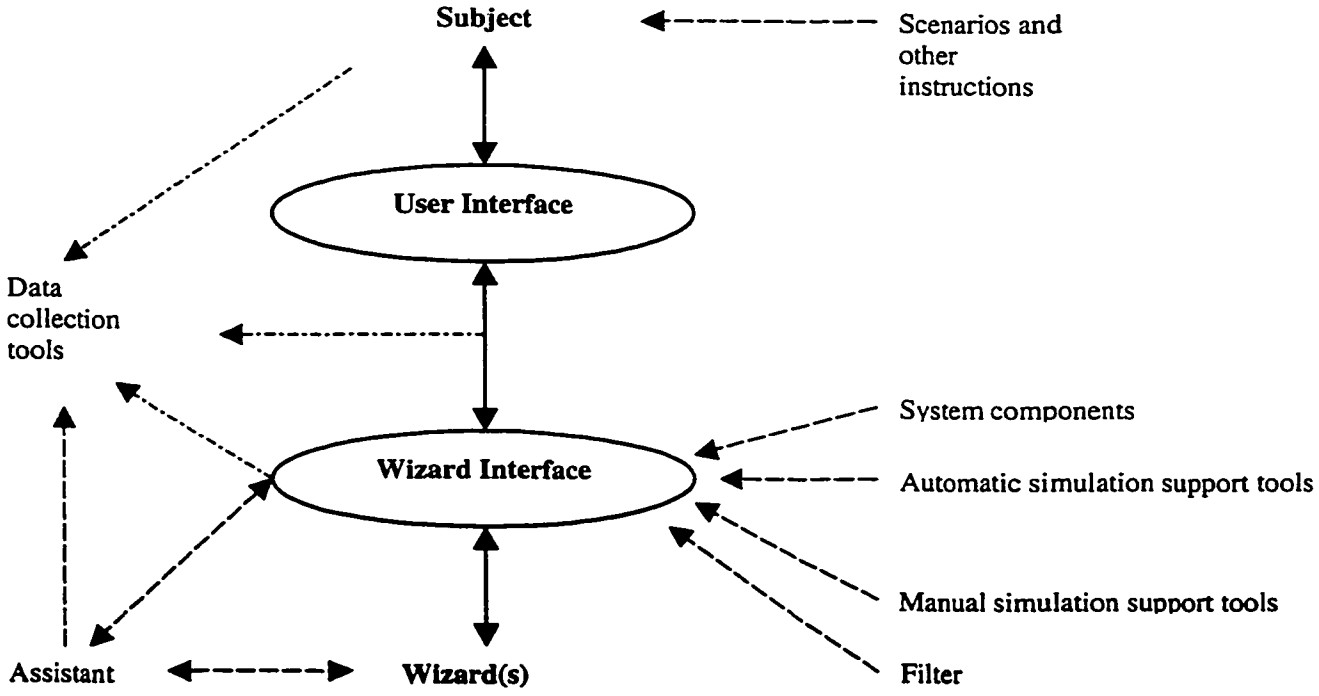

Figure 8-1: Wizard of Oz elements

8.1.3 Mobile Application Development Kit Simulator

From the description of the WOZ technique, it is easily seen how costly it is in manpower and equipment and to what degree the users must be misled to believe they are interacting with a computer system in order for the technique to be properly applied. By contrast, the Mobile Application Development Kit (MADK) simulator from Motorola offered the promise of rapid prototyping for speech recognition applications written in VoxML. The MADK runs on the Windows platform and uses Microsoft's speech recognizer and the Microsoft Agent objects in the simulator user interface, shown in Figure 8-2.

8.1.4 Simulating Speech Recognition Applications with the MADK Simulator

The application to be simulated in the simulator must first be written in VoxML, following the guidelines provided in Motorola documentation. [19] The application can be static, where the possible options at any dialog turn are known in advance (i.e. hard-coded), or dynamic, where the option lists are built in real-time using database access code written in VBScript and executed using the Application Server Pages (ASP) server-side scripting technology. Only static VoxML code, provided in Appendix D, was written for the purposes of this major report. Once the static and dynamic implementations are tested in the simulator, a request can be made for the deployment of the VoxML application on a VoxML-enabled server operated by Motorola's developer support group. This would allow for performing acceptance tests in a real-world setting by having users dial the telephone number associated with the application.

Figure 8-2: MADK simulator screen

When a simulation is started in the MADK simulator (Figure 8-2), a Microsoft Agent character ("Merlin") appears and starts speaking prompts from the VoxML file(s). This is very important because, in order to be successful, the simulation has to be audible in most of the cases. In general, subjects need to hear the utterances (i.e. the prosodic cues), not merely read a verbatim transcript, to be able to distinguish between turn-medial and turn-final utterances. [20]

The speech is generated by Lernout & Hauspie text-to-speech software included with Microsoft Agent, and the computer must have appropriate hardware for speech reproduction/acquisition, such as one or more sound cards, speakers or headphones, and a microphone. The text of the prompts also appears in a cartoon bubble beside the agent character. Impressive as it is, the cellular phone representation at the far right of the screen (with functional display and face and side buttons) is not actually used in the VoxML simulation, but the text box and list box below the image are. The user can interact with the simulator in one of three ways: (i) by speaking into a microphone attached to the computer; (ii) by typing the responses into the text box below the cellular phone image; or (iii) by choosing an option among those listed in the list box beneath the text box. A fourth interaction modality, selecting the response from a list that appears when right-clicking on the agent character, doesn't offer the full range of options, and therefore cannot be used consistently.

The list box provides several options besides the valid responses at any given dialog turn. These additional options, such as "invalid response," "no response" and "too little input," are used for debugging error handling in the VoxML file(s).

8.1.5 Speech Recognition Functionality

The simulator's speech recognition functionality is so closely tied to the Microsoft Agent functionality that there can be no recognition if the agent is not visible. Recognition is always initiated by the user, by pressing the Scroll Lock key in order to activate the recognition for a preset duration. The Scroll Lock key can also be depressed for the entire duration of the utterance, overriding the preset recognition duration. When the Scroll Lock key is first pressed, the agent will display a "tool tip" message saying that it is preparing to recognize speech. Another tool tip message will appear after a short delay, informing the user that the agent is ready. On subsequent use of Scroll Lock, the agent accepts speech without delay.

The agent's tool tip feature is useful for providing the user with immediate feedback on the agent's state. The user understands easily that it is necessary to wait for the agent to signal that it is ready to accept speech input. The tool tip's other purpose is to display the recognized utterance, which gives an immediate feedback to the user. In a sense, this may be counter-productive, because one of the goals of the simulation/evaluation would be to see how well the system gives feedback to the user about the recognition results, i.e. what is the user's mental image of the words the system has recognized.

The fact that the agent has to be visible has one positive aspect in the sense that it provides an anthropomorphic interlocutor instead of letting the user speak to an inanimate object, the computer. The agent can encourage the user to speak by simulating facial expressions as if it were listening. On the other hand, the user can be distracted by the agent's appearance and "behaviour," which have no relation to the real speech application under evaluation.

8.1.6 MADK Simulator Findings

The first official release (version 1.0) of the MADK simulator had a number of limitations, which slowed down considerably the simulation/evaluation phase. Given the amount of time and effort spent on analysing and working around these limitations, they are worth discussing, although they have been largely corrected in the version 2 of the MADK, released six months later:

1. Simulator Doesn't Implement the Yes-or-No (YORN) Input Element Correctly.

This is quite an issue knowing that the yes-or-no acquisition step might be the most frequently used of all possible steps in an application. In the simulator, the list of possible inputs for the YORN input type doesn't contain any options except the built-in commands such as "cancel" and "help". This indicates that the command set grammar for the YORN input type is not being built, and the agent doesn't recognize "yes" or "no" utterances. Given the fact the simulator doesn't support timeouts (see below), the only way to get past the YORN step is to type a "yes" or "no" in the text box. Therefore, a work-around is needed in the form of an option list with possible choices equivalent to "yes" and "no," such as "OK," "yeah," etc. (Limitation specific to version 1.0)

2. Recognizer Doesn't Support the INPUT DIGITS Element.

The WishToys application requires the acquisition of a digit sequence for user identification, which is not possible using the simulator's implementation of the VoxML specification. In order to be able to collect a series of digits from the user, it is necessary to develop a custom work-around dialog equivalent to the INPUT DIGITS step. The dialog consists of a cascading series of identical steps capable of collecting a single digit. The collected digits are subsequently concatenated to form the multi-digit sequence and the user is asked to confirm the correctness of the digit sequence. Due to the fact that the

recognizer returns only the first digit of a multi-digit utterance, resulting in a high probability of losing digits between steps, the user needs to be explicitly prompted for each digit. The prompting is performed using a brief beeping sound stored in an audio file. (Limitation specific to version 1.0)

3. Simulator Introduces a Usability Issue by Forcing the User to Initiate the Recognition Operation.

At any point in a given VoxML application, it is possible to ascertain (deterministic) whether a user utterance is expected or not, i.e. whether the recognition engine should be active or not. The simulator could conceivably activate the speech recognizer in those instances, much as the real VoxML application server would behave. In the 1.0 implementation, however, it is up to the user to activate the speech engine at each and every turn. In the case of the Microsoft Agent user interface, the user needs to press the Scroll Lock key. This poses a problem for two distinct sets of reasons: First and foremost, the simulator doesn't allow for proper usability evaluations because the user has an unspecified amount of time to think about how to react to the prompt, as opposed to the real-time experience with the VoxML application server. Secondly, the task of pressing the Scroll Lock is pure cognitive overhead: it may be hard to find on the keyboard, the user has to focus on a repetitive manual operation rather than a purely vocal exercise, and on certain laptop computers pressing the Scroll Lock key requires the use of both hands (because the key is activated by holding down two keys on opposite sides of the keyboard). All this easily leads to frustration on the part of the user, which reduces both the general satisfaction rating and the recognition accuracy. (Limitation specific to version 1.0)

4. Simulator Doesn't Support Timeouts.

This limitation is similar in effect to the preceding limitation, in the sense that the user has an unlimited time to phrase a response. Moreover, the simulator can't be used to test the timeout repair steps, as there is no way even to force a timeout. (Limitation specific to version 1.0)

5. Microsoft Agent Object is Initialized when the User Presses the Scroll Lock Key for the First Time.

This introduces a delay in the start of recognition for the first user turn, starting from the moment the user presses Scroll Lock, thus increasing significantly the probability of a failed recognition of the first user utterance. As shown in a separate proof of concept, the agent object could be initialized beforehand, i.e. at the start of the simulation, and the delay could be greatly reduced.

6. Microsoft Agent Object Freezes Up Frequently.

Even when the VoxML application doesn't contain calls to unsupported features, the performance of the speech recognition through the agent object is unreliable, with the agent object frequently remaining blocked for no apparent reason. As demonstrated in a separate proof of concept, the agent object, and by extension the speech recognition engine, can function quite reliably when properly called from the client application. Because of this reliability factor, it is seldom possible to complete a simulation without reverting to mouse and keyboard operations for option selection in the simulator. This of course in no way simulates the user's experience with the real system.

7. Built-in "Help" Keyword Activates the Help Prompt in the CLASS Element Definition (VoxML Limitation).

In other words, any time the user requests help, the response coded in the CLASS element definition will override any other response that may have been intended at that point. This is a definite convenience for the designer in the sense that help authoring is simplified by the "inheritance" of CLASS element definitions. On the other hand, designing a custom multi-level help system within a VoxML application is all but impossible. A possible work-around is to offer choices in the help prompt and to include the help-specific options into the current OPTIONLIST, together with valid application options. If the user speaks a help-specific option, the system would branch to STEP elements specifically created for help output. This poses a problem of how to return to the original STEP element, which would require additional workarounds. Also, the user could accidentally trigger a branching to the help-specific STEP, which would be rather confusing and unhelpful. Lastly, this solution would only work for steps with INPUT OPTIONLIST elements.

8. Simulator Doesn't Support the INPUT RECORD Element.

The INPUT RECORD element should be capable of recording the user's utterance to a wave file. This would be used for the personal voiceprint key acquisition dialog in a future implementation of WishToys. With the version 1.0 of the simulator, a "not yet implemented" error is displayed on parsing a VoxML file that contains the INPUT RECORD element. (Limitation specific to version 1.0)

8.1.7 Evaluating and Revising the VUI

For all the above reasons, the first version of the simulator was utterly inadequate for the purposes of the WishToys VUI. With version 2, a possibility opened up with the added

support of many features lacking in the first version, but the reliability issue was insurmountable. In fact, a simulation could never be completed properly, rendering any result of a usability evaluation inherently flawed. As a consequence of the unsatisfactory experience with the MADK simulator, VUI usability was evaluated using the heuristics approach, which resulted in the dialogues shown in Figure 7-2 to Figure 7-9. The design was frozen after a varying number of iterations, depending on the complexity of the dialog. In order to preserve clarity, the dialog flow diagrams in Figure 7-2 to Figure 7-9 for the most part do not show the many options for error handling in VoxML, the usage of which can be seen in the code listings in Appendix D.

8.2 Simulating, Evaluating and Revising the GUI

Thanks to the rapid application development environment offered by Visual Basic, the cycle of simulating, evaluating and revising the GUI can be repeated at a relatively small cost. As the GUI had already undergone usability evaluations, the user evaluation part was performed in a less formal manner described in Gomoll (1990). [21] The steps followed were:

- Set up the observation: Write the tasks. Recruit the users. Set up a realistic situation.
- Describe the purpose of the observation (in general terms). Emphasize that you are testing the product, not the users.
- Tell the user that it's OK to quit at any time.
- Talk about and demonstrate the equipment in the room.
- Explain how to "think aloud."
- Explain that you will not provide help.

- Describe the tasks and introduce the product.
- Ask if there are any questions before you start; then begin the observation.
- Conclude the observation.
- Use the results.

8.2.1 Usage Scenarios

As suggested by Nielsen (1994), [22] the tasks presented to the test users were written so that the first task would be the easiest to complete:

Scenario 1: You are casually browsing the Web and you stumble on the WishToys site. Your interest is piqued and you check out the price and description for an expensive Harley-Davidson Barbie doll.

Scenario 2: You want to buy a playhouse for your niece, and you want to ship it to a different address from your home address.

Scenario 3: You are a parent and wish to set up an account for your child with a monthly allowance of $30 for toys at the WishToys site.

Scenario 4: Your nephew has a Wish List and you are provided with the child's account number. You want to choose a toy from the Wish List not exceeding $50 and send it to the child's address.

8.2.2 Evaluation Results

Four potential users with various levels of computer experience and knowledge were asked to use the demo GUI in a relaxed environment. The number of potential users was kept to a minimum, following the line of thought that increasing the number of users past 3-5 people brings diminished returns. [23] The users were given a brief overview of the

system within the context of Web shopping applications in general, but specific instructions for use of the different features were not provided. The results were captured in the form of notes, captured in Table 8-1, divided into usability problems and software issues on the one side and future improvements and additional feature requests on the other.

Table 8-1: Usability evaluation notes

User 1:
Usability Problem/Software Issue:
1. On clicking to add an item to the cart, "doesn't see where stuff went" (to cart): possibly add animated feedback (with option to turn it off);
2. Suggested explaining the customer number and PIN on the Login screen;
3. Move the "required fields" notice above the fields;
4. "On special" is confusing terminology (use special offer?)
5. Double "Outlook bar" menu is confusing (e.g. "Best sellers on special") - seems that all items in "Shop for ..." are on special, because "On special" is the first item (i.e. most prominent);
6. The double-menu system with the changes in displayed items was all the more confusing because the test database contained a small number of entries;
7. Need a visual indication that the second Outlook bar menu has changed (as a function of the first menu), which currently is not clear enough;
8. Add animation to the second Outlook bar menu as it opens/closes;
9. "Best sellers" and "On special" icons in "Shop for ..." should be visibly separated from others;
10. Confusion for the Quit button, seems like it just applies to the Profiles screen;
11. Remove from cart operation should have immediate feedback: the item should disappear from the list;
12. Total sum should be calculated in the Shopping Cart screen;
Future Improvements and Features:
13. Implement Web browser cookies for automated login/usage tracking;
14. Provide e-mail reminder feature for forgotten PIN;
15. Get the PIN through e-mail on first registration;

16. The UI level selection step seems to be a "pre-search" – make it into a pre-search for kids (do not provide a search feature in the kids' UI), whereby kids logging on would only be able to browse toys for kids (e.g. age < 10);
17. Profiles should have the birthday or some other dates, and new features could be developed to use that information;
User 2:
Usability Problem/Software Issue:
18. Add to list/add to cart operations need feedback (flying toys?)
19. "Add to wish list" check box/button looks like the windows properties icon (before revision);
20. Kids shouldn't see the price of toys;
21. Navigation is difficult because of non-standard and text-only buttons (before revision);
22. Couldn't find the "un-zoom" button (before revision which changed its place on the toolbar nearer to the large picture);
23. Quit button is confusing: does it apply to the current screen or to the entire application?
24. Profile dialog needs add/apply button (before revision);
25. Change name of "browse" button to "shop for" (before revision);
26. Unclear what the Profile feature is used for;
27. Price search is not intuitive: suggestion to cluster the visuals (think of airplane dashboard), as it would be simpler with just two text boxes for the low/high price;
28. Need to find better icons for gender (boy/girl) (before revision);
29. Make "search" text box bigger, with bigger font (before revision);
30. Rename search button to "go" or "search" instead of "go search" (before revision);
31. Navigation buttons should be changed to icons (before revision);
32. Grayed-out text in the detailed view is inappropriate;
33. Price for items "on special" should be bigger, and the old price should be struck out;
Future Improvements and Features:
34. Provide configuration setting to enable/disable add to list/add to cart animated feedback (e.g. it can be turned off using a checkbox in the dialog that pops up) (how can it be re-enabled when the user wants it?);
35. Make the overall presentation more aesthetically pleasing with rounded corners/buttons/different "skins," etc. (examples: www.radioone.com.lb, www.iga.net, saharasoftware, digitalriver, toysrus);

36. Suggestion to be able to drag and drop pictures to the shopping cart;
37. Need a "print" feature for toy's picture and other info;
38. Need to be able to e-mail info about a toy to others;
User 3:
Usability Problem/Software Issue:
39. Browse buttons' button frame color coding (selection indicator) is unclear, should use checkbox with graphics;
40. Minimizing errors: every GUI operation should be recoverable;
Future Improvements and Features:
41. Suggestions for profiles: improvement to child profiles: - Age is dynamically updated; - Memorize favorite categories/brands - Memorize price range?
42. Pop up a wizard (agent) if significant mouse movement is detected between clicks;
User 4:
Usability Problem/Software Issue:
43. Shopping Cart, Wish List need subtotals: when toy is added/removed to/from cart, the total number of toys, and the total cost, should be calculated;
44. UI selection level screen not clear;
45. Add a "legend" in lower left corner of Shopping Cart, Wish List, Recycle Bin screens;
46. Family member users should be able to see only their own profile;
47. "Remove family member profile" button should only be available in the dialog for existing family member profiles, and not in the dialog for "add new";
48. Only the master user should be able to override the PIN of other users;
49. Ask the user to repeat their old PIN in the "change PIN" dialog (except for the first time when the PIN is not yet set);
50. Should implement case-sensitive equality in Profiles so that the change/lack of change can be detected and the "cancel" and "apply" buttons can be enabled/disabled;
51. Incorrect tabbing order in address panel;
52. Need a help screen for Profiles which would say: "fill in the required information (in bold) for yourself, and then create as many additional profiles as you wish, which will all share the same credit card information";

53. Need to disable labels in shipping address dialog when shipping address is same as billing address;
54. Need to display new profile after it is created (same for addresses);
55. Clicking on checkout should display help message: "please fill in shipping and billing information in the master user profile first";
56. Naming issues: Master User (me, myself, I); Family Member/Auxiliary User; Wish List; Profile/Preferences; Child/Adult/Senior; Visitor/Guest; Customer Account; Toy Assistant/Advisor/Recommend-a-Toy;
57. Need to display appropriate text in list view/detailed view screen when no results for the search can be found;
Future Improvements and Features:
58. Suggestion to combine the Login and UI level selection screens for faster access;
59. Add ability to review current/previous orders, both made through GUI or VUI;
60. Add an icon/action button in the Wish List menu to access another user's Wish List by typing the other user's customer number in a pop-up dialog (actually, should be able to send a link to another person by e-mail, clicking on which would allow this other person to view your wish list);
61. Feature to display remaining credit when credit limits are applied. Advantages: parents teaching children how to manage money/how to budget; Disadvantages: less parental involvement into child's spending, missed educational opportunity;
62. Implement flashing labels to indicate that field validation failed (possibly could click on label to get context help);

As far as scenarios are concerned, the users were able to complete the first three scenarios, with varying levels of expressed satisfaction. The functionality necessary for scenario 4 was not implemented, and therefore the users were not asked to attempt it. Asking the users to unwittingly attempt an impossible scenario would be contrary to the philosophy of utmost respect for the test user that is essential for a successful usability evaluation.

8.2.3 Usability Criteria

With more time and effort, quantitative data could have been gathered, such as rating the five main usability criteria on a scale from 1-5 and calculating average scores. The scores

would be tracked between successive cycles of revising and evaluating in order to measure improvements brought by changes to the interface. [22] For a toy shopping application, the order of importance of the five main usability criteria would be as follows:

Subjective Satisfaction – User satisfaction must be the driving force of the application, in order to generate repeat business and priceless word-of-mouth advertising.

Few and Non-catastrophic Errors – The application must not crash, and errors must be handled in a fool-proof manner, otherwise user confidence in the shopping application will plummet, and the user will shop elsewhere.

Learnability – After the previous two criteria, learnability comes as a close third, so as to achieve a walk-up ease-of-use character in the interface. The last thing a user is looking for is a struggle with a difficult interface.

Memorability – Once the user is hooked by the application's rate of satisfaction, lack of errors and learnability, repeat business will be easier to retain if the user interface operations are easily memorized.

Efficiency – In the toy-shopping context, purchases occur fairly infrequently, so user interface efficiency would seem to be the least important criterion among the five main ones. This would no longer be true for an electronic commerce application dealing with items purchased on a daily basis, such as an on-line grocery store.

8.2.4 Summary

Testing the demo GUI with potential users in different stages of development was crucially important. A number of issues were identified and corrected, and these corrections are included in the GUI screen captures (see Figure 7-12 to Figure 7-23). The gathered comments and suggestions for future features are a good indication that the design is on the right track, with more work to be done on improving the overall usability of the application, keeping in mind the relative importance of different usability criteria in the toy shopping application domain.

CHAPTER 9: Conclusion and Further Work

The WishToys application described in this major report, a Web-based electronic shopping application with a separate voice access interface, proved to be a challenging project. The demo application was developed following a combined methodology incorporating the best elements of the DDP methodology for the VUI component, [1] and a rapid-prototyping approach for the GUI component. As such, the WishToys dual user interface application offers a number of advantages over either a Web-only or a voice-only application.

The Web-based interface offers easy browsing/searching, large and colorful pictures of merchandise, secure ordering, etc. With the current penetration of Web-based shopping sites, the interface shouldn't present a problem for a large majority of users.

The voice access interface ensures that the customer can always make a purchase from the on-line store, no matter whether a computer is available or not. The voice interface is admittedly not the most natural one for a shopping application, but it can be made usable by applying a number of usability techniques discussed in this report. The main reason for the voice access interface is not usability but rather accessibility, for guaranteed 24/7 shopping site availability.

On the other hand, the dual interface presents several great challenges, stemming from the radically different natures of the two modes of interaction. A main concern is the sharing of terms and concepts between the two interfaces, so that the user's experience with the Web interface can be reused when interacting with the voice access interface.

Compared to this issue, all other usability concerns are secondary, because a failure to co-ordinate the visual and verbal language between the two interfaces would represent an enormous, perhaps even insurmountable, usability issue in itself.

Another important concern is how to handle the first step of interaction: how users can identify themselves to the shopping application in the most similar and hassle-free way in both interfaces.

These and other concerns are of such importance that it is no surprise that a general-availability dual user interface shopping application does not yet exist on the North American market. In spite of the difficulties encountered during the writing of this major report, the WishToys demo could be further developed into a working application, assuming the availability of human and technological resources as for any other large-scale electronic commerce application. Still, the finished application would have to be aggressively marketed in order to persuade users to try out the voice access component, which will probably seem quite unfamiliar and unusual to the majority of people.

With the increasing penetration of personal digital assistant (PDA) computers with both Web and voice support, perhaps a third type of user interface, where voice access is combined with graphics displayed on a small screen, will be the first type of voice-enabled shopping application to achieve widespread popular acceptance. Such an interface would be modeless, so that the user would feel in control, a very important characteristic that voice-only interfaces cannot yet offer.

9.1 Further Work

Work on the WishToys application could be continued in a number of different areas, including the following:

9.1.1 Further Work on the Application as a Whole

1. Quantifying the Effect of UI Features on Sales

A study to quantify the effect of UI design on traffic and sales can only be performed on a relatively large sample population and under real-life conditions. Simulation results on this topic would be worthless, as subjects would not be using their own money and valuable time. The work by Lohse and Spiller [24] is an excellent example of the counter-intuitive results that a small study sample can produce when looking at these issues.

2. Improvements for the Shopping Assistant Agent Functionality

The Shopping Assistant could be improved by introducing a "recommender" system functionality, both in the GUI and the VUI. [25] This functionality would be part of a multi-agent networked system.

9.1.2 Further Work on the VUI

3. Introductory Explanation

Although the VUI has been designed with usability in mind, it is overly optimistic to expect users to be at ease with a voice recognition interface. In particular, users may need to hear a short explanatory message before they first use the system. The message can be delivered in two different ways: either every caller hears a yes/no question as to whether they are familiar with the system, or the message is played to those users that successfully log into the VUI application for the first time. Even combining the two strategies, with the explanatory messages covering different aspects of the VUI application wouldn't

hurt. The second (post-login) message would be played only at the first login, but it could be repeated if the user logs in after not using the system for a given time interval.

4. Multiple Voices

The use of several different voices may make it easier for the user to discern various separate system functions, such as a command from data or a narrative mode from a multiple-choice mode. [10] As a marketing tie-in, toy manufacturers could provide toy descriptions as audio files with voices used in TV ads for the same toy. Also, the voice chosen for the agent character in the GUI could be reused for certain prompts in the VUI.

9.1.3 Further Work on the GUI

5. Developing a Custom Agent Character

As mentioned previously, a custom designed agent character would ideally be featured in all forms of advertising for the toy store. Some of the tools required to develop the custom designed agent are available for download from Microsoft, but the real challenge lies in the artistic design, marketing concept development, and digital character animation domains, which are well outside the scope of this report.

6. Interfacing with an Intelligent Toy

If a toy store chooses an agent character that can be represented in the form of a toy, it could be sold as an intelligent toy. In the manner of ActiMates Barney, [26] the toy would interface with the GUI to provide friendly comments and suggestions while the child is visiting the site. Other forms of entertainment could also be added – songs, stories, etc. – all delivered through the toy.

7. Adding More Feedback with Animation

As pointed out in the usability evaluation, the "Shop for" menu would benefit from animation to clearly indicate that the menu to the right is dependent on the left-most menu. Similarly, the "Add to ..." operations could also be reinforced with animated feedback of flying icons, which the user can disable once it becomes tiresome. Research on children has shown that these kinds of animated feedback would be helpful to that category of users as well. [27]

8. Adding After-sales Features

It would be useful to have the ability to cancel or modify an order once it is submitted. The HSN site already offers that feature, up to 10 p.m. on the day of the order. After the modification period has expired, the order-tracking feature would take over: similar to what is found at an overnight-delivery company's Web site, the customer would be able to track the order on-line during the course of shipping.

REFERENCES

1 Bernsen, Niels Ole, Hans Dybkjær, and Laila Dybkjær, Designing Interactive Speech Systems – From First Ideas to User Testing, Springer, London, 1998.

2 Cosic, Miroslav and Xiaoman Song, "COMP 675 Programming Project – Part 1: User Interface Specification", Concordia University, Montreal, October 1998 (unpublished)

3 Cosic, Miroslav and Xiaoman Song, "COMP 675 Programming Project – Part 2: Software Implementation and Testing", Concordia University, Montreal, January 1999 (unpublished)

4 Le, Cam Chuong and Miroslav Cosic, "COMP 773 User Interface Evaluation Report," Concordia University, Montreal, July 1999 (unpublished)

5 Cosic, Miroslav and Cam Chuong Le, "COMP 773 User Interface Evaluation Project (Final Report)," Concordia University, Montreal, August 1999 (unpublished)

6 VoxML Developer Web site, Motorola Inc., http://www.voxml.com/voxml.html

7 ActiveListBar Web site, Sheridan Software, http://www.shersoft.com/products/

8 Taylor, M.M. and M.J. Hunt, "Flexibility versus Formality," in *The Structure of Multimodal Dialogue*, M.M. Taylor et al. (eds.), Elsevier Science Publishers, North-Holland, 1989, pp. 435-453.

9 Walker, Marilyn A., et al., "What Can I Say?: Evaluating a Spoken Language Interface to Email," in *Proceedings of CHI '98*, ACM Press, Los Angeles, 1998, pp. 582-589.

10 Schmandt, Christopher, Voice Communications with Computers – Conversational Systems, Van Nostrand Reinhold, New York, 1994.

11 Mandel, Theo, Elements of User Interface Design, Wiley Computer Publishing, New York, 1997

12 Suhm, Bernhard, "Multimodal Interactive Error Recovery for Non-Conversational Speech User Interfaces," Ph.D. Thesis, Universitat Karlsruhe (Technische Hochschule), 1998

13 Yankelovich, Nicole, "How Do Users Know What to Say?," ACM interactions, November/December 1996, pp. 33-43

14 Denecke, Matthias, "An Information-based Approach for Guiding Multi-Modal Human-Computer Interaction," in *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, IJCAI 97, Nagoya, Japan, August 23-29, 1997*, Morgan Kaufmann, 1997, pp. 1036-1041.

15 Wolf, Catherine G. and Wlodek Zadrozny, "Evolution of the Conversation Machine: A Case Study of Bringing Advanced Technology to the Marketplace," in *Proceedings of CHI '98*, ACM Press, Los Angeles, 1998, pp. 488-495.

16 Baber, C., Stammers, R.B. and D.M. Usher, "Error Correction Requirements in Automatic Speech Recognition," in *Contemporary Ergonomics 1990*, E.J. Lovesey (ed.), Taylor & Francis, London, 1990, pp. 454-459.

17 Proctor, C. and S. Young, "Dialogue Control in Conversational Speech Interfaces," in *The Structure of Multimodal Dialogue*, M.M. Taylor et al. (eds.), Elsevier Science Publishers, North-Holland, 1989, pp. 385-398.

18 Larson, Kevin and Mary Czerwinski, "Web Page Design: Implications of Memory, Structure and Scent for Information Retrieval," in *Proceedings of CHI '98*, ACM Press, Los Angeles, 1998, pp. 25-32.

19 VoxML 1.1 Language Reference, Revision 1.1, Motorola Inc., Internet and Connectivity Solutions Division, April 1999, http://www0.motorola.com/ developers/ wireless/ products/ mix/ download/ voxml_lg.pdf

20 Waterworth, J.A., "Interactive Strategies for Conversational Computer Systems," in *The Structure of Multimodal Dialogue*, M.M. Taylor et al. (eds.), Elsevier Science Publishers, North-Holland, 1989, pp. 331-340.

21 Gomoll, Kathleen, "Some Techniques for Observing Users," in *The Art of Human-Computer Interface Design*, by Brenda Laurel (ed.), Addison-Wesley, Reading, 1990

22 Nielsen, Jakob, Usability Engineering, Morgan Kaufmann, San Diego, 1994

23 Nielsen, Jakob and Thomas K. Landauer: "A Mathematical Model of the Finding of Usability Problems," in *Proceedings of ACM INTERCHI'93 Conference*, Amsterdam, 24-29 April 1993, pp. 206-213.

24 Lohse, Gerald L. and Peter Spiller, "Quantifying the Effect of User Interface Design Features on Cyberstore Traffic and Sales," in *Proceedings of CHI '98*, ACM Press, Los Angeles, 1998, pp. 211-218.

25 Lueg, Christopher, "Supporting Situated Actions in High Volume Conversational Data Situations," in *Proceedings of CHI '98*, ACM Press, Los Angeles, 1998, pp. 472-479.

26 Alexander, Kristin and Erik Strommen, "Evolution of the Talking Dinosaur: The (Not So) Natural History of a New Interface for Children," in *Proceedings of CHI '98*, ACM Press, Los Angeles, 1998, pp. 7-8.

27 Druin, Allison, "Cooperative Inquiry: Developing New Technologies for Children with Children," in *Proceedings of CHI '99*, ACM Press, Pittsburgh, 1999, pp. 592-599.

APPENDIX A: Overview of Existing Applications

This section contains brief overviews of similar applications in the toy-shopping domain. A search for an offering that fully integrates web and voice access in a manner similar to WishToys was fruitless. The existing offerings can be divided into two sets: Web sites operated by companies that make some use of voice recognition technologies, and purely Web-based toy shopping sites.

A.1 Web Sites from Companies that use Speech Technology

A.1.1 Home Shopping Network - Dolls & Bears

The Home Shopping Network (HSN) (http://www.hsn.com/store/) is planning to introduce natural language understanding speech recognition software into its order-taking process in the near future. Currently, HSN is already using speaker verification software (provided by Nuance Communications) to accelerate and simplify customer identification over the phone. The user interface is much simpler for the customer, as the only information to memorize is the customer's telephone number.

Among the many products offered by HSN, customers can buy specialty dolls and bears (including Barbie). HSN focuses on the doll collector community, featuring tie-ins with doll designers, collectors' meetings, doll signings, etc.

A.1.2 Sears – wishbook.com

Sears (http://www.wishbook.com/wishbook6/) is using voice recognition technology to reduce the time its telephone operators spend redirecting customer calls to specific departments. Plans for further voice recognition applications are not available.

Through their wishbook.com Web site, Sears allows customers to set up wish lists (must be over 13) and to send a special wish list ID by e-mail to a friend or family member. Customers access the system by providing combinations of e-mail address, password, wish list ID, etc.

A.2 Other Toy-shopping Web Sites

A.2.1 eToys

The eToys Web site (http://www.etoys.com) has a wish list similar to one in WishToys. Access to the site requires an e-mail address, a name, and a password. Another similarity between eToys and WishToys is the Shop-by-Age feature, that is an implicit "search by age." For some reason, eToys doesn't allow searches for "kids" older than 12, i.e. toys that would be of interest to adults.

A.2.2 iBaby

The iVillage iBaby site (http://www.ibaby.com) targets parents rather than children, but it also has some of the features common to wishbook, WishToys, etc. The wish list feature is offered as a gift registry for new parents. Shopping assistant functionality suggests related items while a customer is browsing.

A.2.3 Toys'R'Us

The Toys'R'Us Web site (http://www.toysrus.com) has both main browse/search features described in WishToys: the browse by brand/category and the search. Toys'R'Us call these features "How do you want to shop today?" and "EZ Toy Finder" respectively. The site also has a baby registry and gift certificates, but no wish list per se.

APPENDIX B: Overview of Competing Technologies

The following technologies are currently competing with Motorola's VoxML-based voice application servers:

B.1 Microsoft Web Telephony Engine

The Microsoft Web telephony engine (http://www.microsoft.com/ WINDOWS2000/ library/ howitworks/ communications/ telephony/ webtel.asp) is a Windows 2000 run-time service that renders Web-based interactive voice response (IVR) applications. This service lets developers use Web-development tools such as HTML, Microsoft ActiveX controls, scripting languages, and Active Server Pages (ASP) to create IVR applications that are accessible by both telephone and Internet browser. Figure B-1 shows a diagram of a system using the Web Telephony engine.

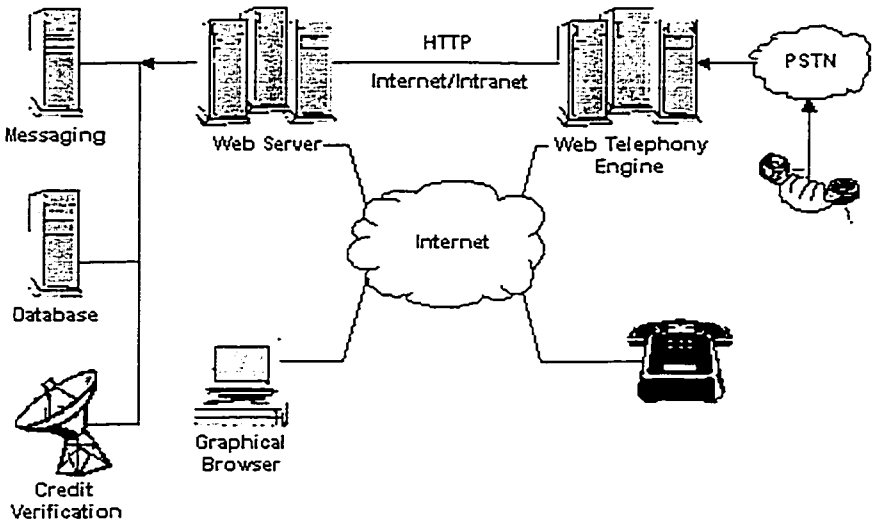

Figure B-1: Web Telephony system diagram

B.2 Nuance Communications Voyager

The Nuance Voyager network-based voice browser (http://www.nuance.com/ partners/ voiceweb.html) provides a standardized user interface into any voice-driven application and is modeled after the functionality provided by web browsers on the desktop. The Voyager user interface includes:

- **Enrollment** - mechanism for new account creation.
- **Entry Logic** - for user identification and login.
- **Main Menu** - customizable main menu that provides the entry point into an aggregated set of applications or voice portal services.
- **Navigation** - commands like "Go Forward" and "Go Back" for navigating voice web content.
- **Bookmarks** - personal list of frequently visited sites.
- **Profiles** - personal information stored by the browser that is used to speed up transactions and enable personalized content delivery.
- **Active Help** - Context sensitive help provided as it is needed based on the user's experience.

B.3 IBM WebSphere Voice Server

IBM's WebSphere Voice Server (http://www-4.ibm.com/ software/ speech/ enterprise/ ep_1.html) provides the structure for voice-enabled applications, using mixed-initiative dialog and VoiceXML programming. It contains the following components: Voice XML Browser, IBM's Voice Recognition Engine, IBM's Text-To-Speech engine and basic command-line and system management tools. IBM WebSphere Voice Server tools allow developers to create a natural and conversational user interface for Web applications.

B.4 SpeechWorks SpeechSite

SpeechWorks SpeechSite (http://www.speechworks.com/ productsservices/ products/ speechsite/ index.cfm) builds on the web model of self-service by providing a way for customers to reach a company to contact employees or retrieve the information they need without relying on operators or waiting for business hours. The SpeechSite architecture, shown in Figure B-2, takes advantage of the enormous investments that corporations have made, and are making, in their web site infrastructures for self-service. With significant amounts of corporate data already organized and published on the web, companies can easily add on a SpeechSite for telephone access.

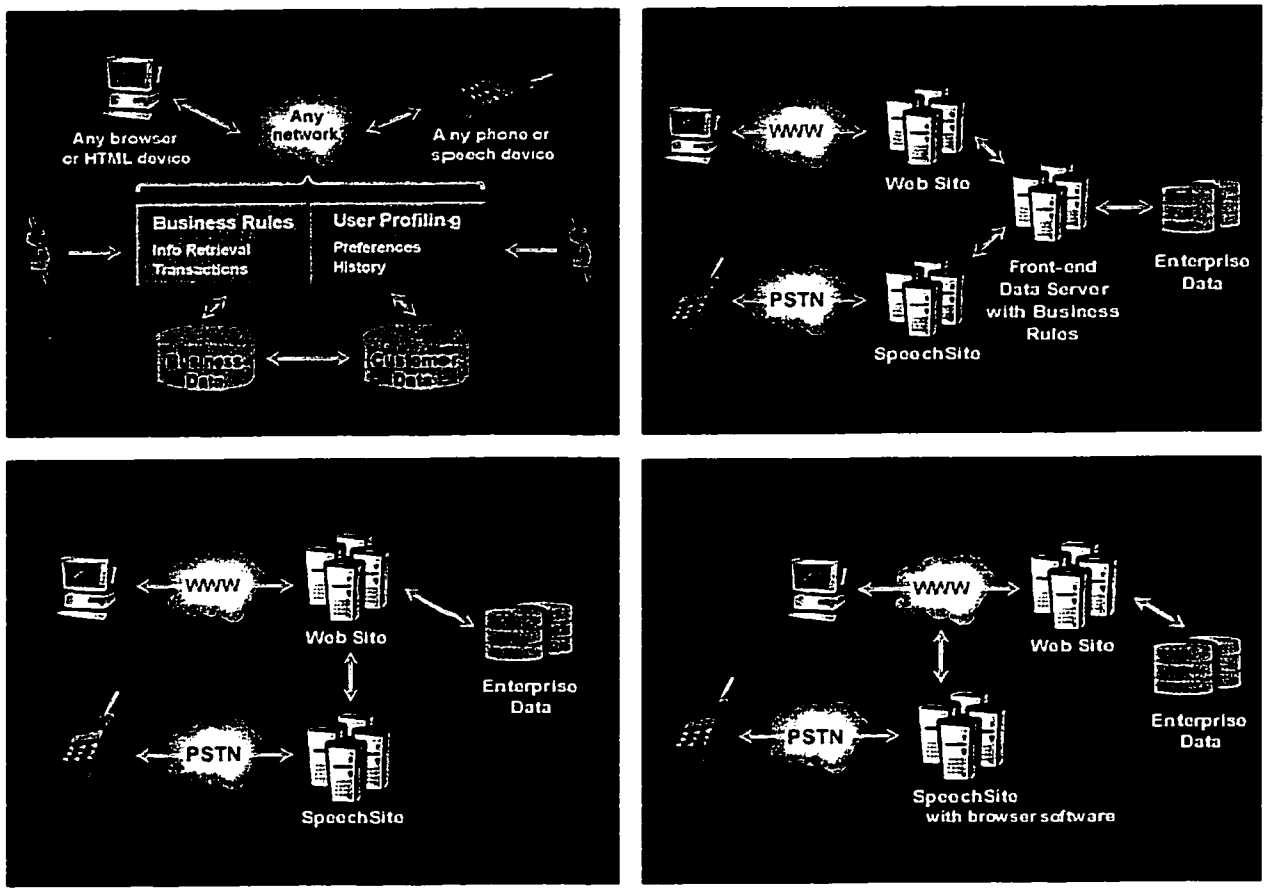

Figure B-2: SpeechWorks SpeechSite architecture

APPENDIX C: Voice Markup Language (VoxML)

C.1 Overview

VoxML is a dialog-based language, where applications are delivered in whole-dialog units. The elementary components of a dialog are called steps, which are roughly equivalent to the more common notion of dialog turns. As VoxML is based on the eXtensible Markup Language (XML), VoxML code has to comply with the XML Document Type Definition (DTD) defined by the VoxML specification. The current version of the specification is 1.2. Examples of VoxML code are provided in Appendix D.

C.2 Language Features

For a quick introduction to VoxML features, it is interesting to look at the mapping between VoxML language elements and speech application features, listed in Table C-1:

Table C-1: VoxML language features

VoxML Element	Purpose	Speech Application Usage (with a focus on usability improvements)
ACK	Provide acknowledgement with or without confirmation.	Provide positive recognition feedback while transitioning to the next step in the dialog.
AUDIO	Play back pre-recorded audio files.	Provide for a more natural interface as opposed to text-to-speech generation.
BREAK	Introduce a pause in the text-to-speech output.	Improve comprehension of text-to-speech output by introducing pauses between unrelated speech segments. BREAK can also be used with audio files, if the files do not already contain the appropriate periods of silence.

VoxML Element	Purpose	Speech Application Usage (with a focus on usability improvements)
CANCEL	Define behaviour if a cancel command is issued.	Provide a meta-command or a regular command to exit from the current dialog step.
CASE	Define the flow of control of the application.	Provide for a number of possible choices to be presented to the user at a dialog step.
CLASS	Define the behaviour of ERROR, HELP and CANCEL within a given dialog.	Through inheritance, provide for a consistent application-wide behaviour for meta-commands.
DIALOG	Root element in a VoxML page.	Allows the application to be organized in reusable dialog units, improving consistency.
EMP	Introduce emphasis variation in text-to-speech output.	Improve comprehension of text-to-speech output by introducing emphasis for certain words.
ERROR	Define the behaviour of the application when an error is detected.	Provide differentiated error handling based on the nature of the error.
EXIT	Define the behaviour for a user-requested exit operation.	Provide a meta-command or a regular command to jump to the top level of the dialog.
GOODBYE	Define the behaviour for a user-requested goodbye operation.	Provide a meta-command or a regular command to quit the dialog.
HANGUP	Define the behaviour when the user hangs up.	Provide for a clean exit from the application.
HELP	Define the behaviour when the user requests help.	Provide for a user-friendly help system with an increasing level of detail in the explanations.
INPUT DATE	Used to collect a calendar date from the user.	Provide for a consistent application-wide behaviour for date input.
INPUT DIGITS	Used to collect a series of digits from the user.	Provide for a consistent application-wide behaviour for digit input.

VoxML Element	Purpose	Speech Application Usage (with a focus on usability improvements)
INPUT GRAMMAR	Allows for a custom input grammar to be specified for a given step.	Provides for the substitution of a more user-friendly grammar instead of the built-in grammar for a given data type, or a custom grammar for a given dialog step.
INPUT HIDDEN	Used to assign a value to a VoxML variable without user participation.	
INPUT MONEY	Used to collect a monetary value from the user.	Provide for a consistent application-wide behaviour for monetary value input.
INPUT NONE	Used to specify the next location for the voice browser to go to continue execution.	
INPUT NUMBER	Used to collect numbers from the user.	Provide for a consistent application-wide behaviour for number input.
INPUT OBJECT	Used to access some platform-specific feature that is not part of VoxML.	Provides for the substitution of a more user-friendly speech objects instead of the built-in grammar for a given data type, or a custom speech object for a given dialog step.
INPUT OPTIONLIST	Used to specify a list of options among which the user can choose.	Provide for a number of possible choices to be presented to the user at a dialog step.
INPUT PHONE	Used to collect a telephone number from the user.	Provide for a consistent application-wide behaviour for phone number input.
INPUT RECORD	Used to record and store an audio sample.	Provide the user with a means to leave a voice message in the application, or a way to acquire a voice sample from the user for speaker-verification purposes.
INPUT TIME	Used to collect a time of day from the user.	Provide for a consistent application-wide behaviour for time input.

VoxML Element	Purpose	Speech Application Usage (with a focus on usability improvements)
INPUT YORN	Used to collect "yes or no" responses from the user.	Provide for a consistent application-wide behaviour for "yes or no" response input.
SWITCH	Define the behaviour dependent on the value of a specified recognition slot.	Provide for a number of possible choices to be presented to the user at a dialog step.
TRANSFER	Allow for the transfer to another phone number.	Provide for live telephone support in case of problems or if the user requests to talk to a representative. Also, may provide a link to a non-VoxML voice application.
VALUE	Allows for presenting the value of a VoxML variable to the user via text-to-speech.	Provide dynamically modified user prompts.

C.3 VoxML Shortcomings

As described in the chapter on simulating the VUI, a custom help system is all but impossible to design in VoxML due to the presence of built-in commands and their precedence over other choices at any given dialog turn. Other perceived VoxML shortcomings are as follows:

- **Cannot Control Rejection Threshold in VoxML:** The recognizer settings are external to the VoxML code and the VoxML application cannot modify them in an intelligent manner.
- **Cannot Control Confusability in VoxML:** The recognition operation can only have one result in VoxML, making it impossible to code for synonym/conflict resolution.
- **Grammar Definitions in VoxML are Recognizer-dependent:** A VoxML application has to be written and tested for one make of speech recognizer because the recognizer settings and grammars are not normalized and portable.

C.4 Future Evolution

VoxML is being phased out in favour of VoiceXML, a more standardized voice applications markup language. Consequently, version 2 of Motorola's Mobile Application Development Kit (MADK) handles both VoxML and VoiceXML. Originally, Motorola created a single ADK for both VoxML and the Wireless Markup Language (WML), in an attempt to present a unified suite of application development tools for wireless applications. It is unclear whether this has really helped industry adoption of VoxML; while VoxML is a speech application development language, WML is a language for text-based very-low-bandwidth applications, and the two actually have very little in common. The downside of the association is the fact that while VoxML/VoiceXML may have a certain future, WML is more of a transitional phase until mobile appliances with sufficiently high bandwidths (e.g. 3G) become widely available. At that point, mobile appliances will probably revert to standard HTML/XHTML, where the pages may have to be specially designed for smaller screens, but not necessarily in a dedicated language such as WML. It would appear that, in information technology's never-ending striving for the latest cutting-edge technology, VoxML loses out by being associated with a has-been technology such as WML.

APPENDIX D: VoxML Code for the WishToys VUI

The current VoxML implementation of the WishToys VUI is purely static in character. A considerable amount of code (VoxML, ASP and VBScript) needs to be added to obtain a working application that interfaces with a database and reuses business logic components from the GUI side of the application. Note: The List dialogue, the Shopping Cart dialogue, the Checkout dialogue, and the Toy Advisor dialogue are not implemented.

D.1 Login Dialogue

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
       Concordia University
       COMP 793 Major Report
       WishToys - login.vml
       Login dialogue
       by Miroslav Cosic
-->
<DIALOG>
       <CLASS
              BARGEIN="Y"
              NAME="help_generic"
       >
              <HELP
                     NEXTMETHOD="GET"
                     REPROMPT="N"
              >You are at the login dialog.</HELP>
              <ERROR
                     NEXTMETHOD="GET"
                     REPROMPT="N"
                     TYPE="ALL"
              >You are at the login dialog.</ERROR>
       </CLASS>
       <CLASS
              BARGEIN="Y"
              NAME="help_customer_number"
       >
              <HELP
                     NEXTMETHOD="GET"
                     REPROMPT="N"
              >Please say your seven digit customer number.</HELP>
              <ERROR
                     NEXTMETHOD="GET"
                     REPROMPT="N"
                     TYPE="NOMATCH"
              >I&amp;apos;m sorry, I didn&amp;apos;t get that. Please say your seven
              digit customer number.</ERROR>
              <ERROR
                     NEXTMETHOD="GET"
                     ORDINAL="2"
                     REPROMPT="N"
```

```
       TYPE="NOMATCH"
>I&amp;apos;m sorry, I still can&amp;apos;t understand you. Please say
your seven digit customer number digit by digit.</ERROR>
<ERROR
       NEXT="transfer.vml"
       NEXTMETHOD="GET"
       ORDINAL="3"
       REPROMPT="N"
       TYPE="NOMATCH"
/>
<ERROR
       NEXTMETHOD="GET"
       REPROMPT="N"
       TYPE="NOSPEECH"
>Please say your seven digit customer number.</ERROR>
<ERROR
       NEXTMETHOD="GET"
       ORDINAL="2"
       REPROMPT="N"
       TYPE="NOSPEECH"
>Please say your seven digit customer number digit by
digit.</ERROR>
<ERROR
       NEXT="transfer.vml"
       NEXTMETHOD="GET"
       ORDINAL="3"
       REPROMPT="N"
       TYPE="NOSPEECH"
/>
<ERROR
       NEXTMETHOD="GET"
       REPROMPT="N"
       TYPE="TOOLITTLE"
>Please say all seven digits of your customer number.</ERROR>
<ERROR
       NEXTMETHOD="GET"
       ORDINAL="2"
       REPROMPT="N"
       TYPE="TOOLITTLE"
>Please say all seven digits of your customer number digit by
digit.</ERROR>
<ERROR
       NEXT="transfer.vml"
       NEXTMETHOD="GET"
       ORDINAL="3"
       REPROMPT="N"
       TYPE="TOOLITTLE"
/>
<ERROR
       NEXTMETHOD="GET"
       REPROMPT="N"
       TYPE="TOOMUCH"
>Please say just the seven digits of your customer number.</ERROR>
<ERROR
       NEXTMETHOD="GET"
       ORDINAL="2"
       REPROMPT="N"
       TYPE="TOOMUCH"
>Please say just the seven digits of your customer number digit by
digit.</ERROR>
<ERROR
       NEXT="transfer.vml"
       NEXTMETHOD="GET"
```

```
                ORDINAL="3"
                REPROMPT="N"
                TYPE="TOOMUCH"
        />
        <ERROR
                NEXT="transfer.vml"
                NEXTMETHOD="GET"
                REPROMPT="N"
                TYPE="BADNEXT"
        >An internal error has occurred.</ERROR>
        <ERROR
                NEXTMETHOD="GET"
                REPROMPT="N"
                TYPE="ALL"
        >Please say your seven digit customer number.</ERROR>
</CLASS>
<!-- Transition to get customer number step -->
<STEP
        BARGEIN="Y"
        NAME="init"
        PARENT="help_generic"
>
        <PROMPT>Welcome to WishToys.</PROMPT>
        <INPUT
                NEXT="#get_customer_number"
                NEXTMETHOD="GET"
                TYPE="NONE"
        />
</STEP>
<!-- Ask for customer number -->
<STEP
        BARGEIN="Y"
        NAME="get_customer_number"
        PARENT="help_customer_number"
>
        <PROMPT>Customer number.</PROMPT>
        <INPUT
                NEXT="#validate_customer_number"
                NEXTMETHOD="GET"
                NAME="customer_number"
                TIMEOUT="5000"
                MIN="7"
                MAX="7"
                TYPE="DIGITS"
        />
</STEP>
<!-- Validate customer number   Note: the help context is not used -->
<STEP
        BARGEIN="Y"
        NAME="validate_customer_number"
        PARENT="help_generic"
>
        <INPUT
                NEXT="#unknown_customer_number"
                NEXTMETHOD="GET"
                TYPE="NONE"
        >
                <SWITCH FIELD="customer_number">
                        <CASE VALUE="1234567" NEXT="#get_pin"/>
                </SWITCH>
        </INPUT>
</STEP>
<!-- Unknown customer number -->
```

```
<STEP
        BARGEIN="Y"
        NAME="unknown_customer_number"
        PARENT="help_customer_number"
>
        <PROMPT>I&apos;m sorry, I don't have a customer number <VALUE
        NAME="customer_number"/>. Please say your customer number digit by
        digit.</PROMPT>
        <INPUT
                NEXT="#validate_customer_number"
                NEXTMETHOD="GET"
                NAME="customer_number"
                TIMEOUT="5000"
                MIN="7"
                MAX="7"
                TYPE="DIGITS"
        />
</STEP>
<CLASS
        BARGEIN="Y"
        NAME="help_pin"
>
        <HELP
                NEXTMETHOD="GET"
                REPROMPT="N"
        >Please say your six to fourteen digit pin.</HELP>
        <ERROR
                NEXTMETHOD="GET"
                REPROMPT="N"
                TYPE="NOMATCH"
        >I&apos;m sorry, I didn&apos;t get that. Please say your six to
        fourteen digit pin.</ERROR>
        <ERROR
                NEXTMETHOD="GET"
                ORDINAL="2"
                REPROMPT="N"
                TYPE="NOMATCH"
        >I&apos;m sorry, I still can&apos;t understand you. Please say
        your six to fourteen digit pin digit by digit.</ERROR>
        <ERROR
                NEXT="transfer.vml"
                NEXTMETHOD="GET"
                ORDINAL="3"
                REPROMPT="N"
                TYPE="NOMATCH"
        />
        <ERROR
                NEXTMETHOD="GET"
                REPROMPT="N"
                TYPE="NOSPEECH"
        >Please say your six to fourteen digit pin.</ERROR>
        <ERROR
                NEXTMETHOD="GET"
                ORDINAL="2"
                REPROMPT="N"
                TYPE="NOSPEECH"
        >Please say your six to fourteen digit pin digit by digit.</ERROR>
        <ERROR
                NEXT="transfer.vml"
                NEXTMETHOD="GET"
                ORDINAL="3"
                REPROMPT="N"
                TYPE="NOSPEECH"
```

```
        />
        <ERROR
               NEXTMETHOD="GET"
               REPROMPT="N"
               TYPE="TOOLITTLE"
        >Please say all the digits of your pin. Your pin has at least six
        digits.</ERROR>
        <ERROR
               NEXTMETHOD="GET"
               ORDINAL="2"
               REPROMPT="N"
               TYPE="TOOLITTLE"
        >Please say all the digits of your pin digit by digit. Your pin
        has at least six digits.</ERROR>
        <ERROR
               NEXT="transfer.vml"
               NEXTMETHOD="GET"
               ORDINAL="3"
               REPROMPT="N"
               TYPE="TOOLITTLE"
        />
        <ERROR
               NEXTMETHOD="GET"
               REPROMPT="N"
               TYPE="TOOMUCH"
        >Please say just the digits of your pin. Your pin has at most
        fourteen digits.</ERROR>
        <ERROR
               NEXTMETHOD="GET"
               ORDINAL="2"
               REPROMPT="N"
               TYPE="TOOMUCH"
        >Please say just the digits of your pin digit by digit. Your pin
        has at most fourteen digits.</ERROR>
        <ERROR
               NEXT="transfer.vml"
               NEXTMETHOD="GET"
               ORDINAL="3"
               REPROMPT="N"
               TYPE="TOOMUCH"
        />
        <ERROR
               NEXT="transfer.vml"
               NEXTMETHOD="GET"
               REPROMPT="N"
               TYPE="BADNEXT"
        >An internal error has occurred.</ERROR>
        <ERROR
               NEXTMETHOD="GET"
               REPROMPT="N"
               TYPE="ALL"
        >Please say your six to fourteen digit pin.</ERROR>
</CLASS>
<!-- Ask for PIN -->
<!-- NOTE: At this point, the system may have understood a valid customer
number, but it may not have been what the user said. The user can use
CANCEL to return to the previous step -->
<STEP
        BARGEIN="Y"
        NAME="get_pin"
        PARENT="help_pin"
>
```

```
        <PROMPT>Pin for customer number <VALUE
        NAME="customer_number"/>.</PROMPT>
        <INPUT
                NEXT="#validate_pin"
                NEXTMETHOD="GET"
                NAME="pin"
                TIMEOUT="5000"
                MIN="6"
                MAX="14"
                TYPE="DIGITS"
        />
        <CANCEL
                NEXT="#get_customer_number"
                NEXTMETHOD="GET"
        />
</STEP>
<!-- Validate pin    Note: the help context is not used -->
<STEP
        BARGEIN="Y"
        NAME="validate_pin"
        PARENT="help_generic"
>
        <INPUT
                NEXT="#invalid_pin"
                NEXTMETHOD="GET"
                TYPE="NONE"
        >
                <SWITCH FIELD="pin">
                        <CASE VALUE="654321" NEXT="#salutation"/>
                </SWITCH>
        </INPUT>
</STEP>
<!-- Invalid PIN -->
<STEP
        BARGEIN="Y"
        NAME="invalid_pin"
        PARENT="help_pin"
>
        <PROMPT>I&amp;apos;m sorry, pin <VALUE NAME="pin"/> doesn&amp;apos;t match
        customer number <VALUE NAME="customer_number"/>. Please say your
        pin digit by digit.</PROMPT>
        <INPUT
                NEXT="#validate_pin"
                NEXTMETHOD="GET"
                NAME="customer_number"
                TIMEOUT="5000"
                MIN="6"
                MAX="14"
                TYPE="DIGITS"
        />
        <CANCEL
                NEXT="#get_customer_number"
                NEXTMETHOD="GET"
        />
</STEP>
<!-- Salutation step before switching to main menu -->
<STEP
        BARGEIN="Y"
        NAME="salutation"
        PARENT="help_generic"
>
        <!-- TBD use database access method to obtain name of customer
        with customer_number -->
```

```
            <PROMPT>Welcome, Friend.</PROMPT>
            <INPUT
                  NEXT="main_menu.vml"
                  NEXTMETHOD="GET"
                  TYPE="NONE"
            />
      </STEP>
</DIALOG>
```

D.2 Main Menu Dialogue

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
      Concordia University
      COMP 793 Major Report
      WishToys - main_menu.vml
      Main application menu
      by Miroslav Cosic
-->
<DIALOG>
      <!-- The REPROMPT feature is used rather than the OPTIONS feature because
      of the special status of the checkout and shopping cart vocabulary
      entries. The entries should always be recognized but they should not be
      mentioned in the list of valid options unless there are toys in the cart.
      Rather than verifying this again in the help section, we reuse the
      original prompt. -->
      <CLASS
            BARGEIN="Y"
            NAME="help_generic"
      >
            <HELP
                  NEXTMETHOD="GET"
                  REPROMPT="Y"
            >You are at the main menu.</HELP>
            <ERROR
                  NEXTMETHOD="GET"
                  REPROMPT="Y"
                  TYPE="NOMATCH"
            >I&amp;apos;m sorry, I didn&amp;apos;t get that.</ERROR>
            <ERROR
                  NEXTMETHOD="GET"
                  ORDINAL="2"
                  REPROMPT="Y"
                  TYPE="NOMATCH"
            >I&amp;apos;m sorry, I still can&amp;apos;t understand you.</ERROR>
            <ERROR
                  NEXT="transfer.vml"
                  NEXTMETHOD="GET"
                  ORDINAL="3"
                  REPROMPT="N"
                  TYPE="NOMATCH"
            />
            <ERROR
                  NEXTMETHOD="GET"
                  REPROMPT="Y"
                  TYPE="NOSPEECH"
            />
            <ERROR
                  NEXTMETHOD="GET"
                  ORDINAL="2"
                  REPROMPT="Y"
```

```
            TYPE="NOSPEECH"
        />
        <ERROR
            NEXT="transfer.vml"
            NEXTMETHOD="GET"
            ORDINAL="3"
            REPROMPT="N"
            TYPE="NOSPEECH"
        />
        <ERROR
            NEXT="transfer.vml"
            NEXTMETHOD="GET"
            REPROMPT="N"
            TYPE="BADNEXT"
        >An internal error has occurred.</ERROR>
        <ERROR
            NEXTMETHOD="GET"
            REPROMPT="Y"
            TYPE="ALL"
        />
</CLASS>
<!-- Note: Help info is not used here -->
<STEP
        BARGEIN="Y"
        NAME="init"
        PARENT="help_generic"
>
        <PROMPT>You are at the main menu.</PROMPT>
        <INPUT
            TYPE="NONE"
            NEXT="#main_menu"
        />
</STEP>
<STEP
        BARGEIN="Y"
        NAME="main_menu"
        PARENT="help_generic"
>
        <!-- TBD Add database access to determine whether to include
        checkout and shopping cart among the choices -->
        <PROMPT>Your choices are wish list and toy assistant.</PROMPT>
        <INPUT
            TYPE="OPTIONLIST"
            TIMEOUT="5000"
        >
            <!-- TBD Add database access to determine whether to
            include checkout and shopping cart among the choices -->
            <!-- Note: Even if there are no toys in the shopping cart,
            the choices for checkout and shopping cart must be seen as
            valid recognition results - just the NEXT attribute changes
            in value -->
            <OPTION NEXT="#no_toys">checkout</OPTION>
            <!-- <OPTION NEXT="checkout.vml">checkout</OPTION> -->
            <OPTION NEXT="#no_toys">shopping cart</OPTION>
            <!-- <OPTION NEXT="shopping_cart.vml">shopping
            cart</OPTION> -->
            <OPTION NEXT="wish_list.vml">wish list</OPTION>
            <OPTION NEXT="toy_assistant.vml">toy assistant</OPTION>
        </INPUT>
</STEP>
<!-- Special step for the "valid but invalid" options in the main_menu
step -->
<STEP
```

```
            BARGEIN="Y"
            NAME="no_toys"
            PARENT="help_generic"
      >
            <PROMPT>Sorry, there is nothing in your shopping cart. Please
            choose wish list or toy assistant.</PROMPT>
            <INPUT
                  TYPE="OPTIONLIST"
                  TIMEOUT="5000"
            >
                  <OPTION NEXT="wish_list.vml">wish list</OPTION>
                  <OPTION NEXT="toy_assistant.vml">toy assistant</OPTION>
            </INPUT>
      </STEP>
</DIALOG>
```

D.3 Wish List Dialogue

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
      Concordia University
      COMP 793 Major Report
      WishToys - wish_list.vml
      Wish List feature
      by Miroslav Cosic
-->
<DIALOG>
      <CLASS
            BARGEIN="Y"
            NAME="help_generic"
      >
            <HELP
                  NEXTMETHOD="GET"
                  REPROMPT="N"
            >You are at the wish list dialog.</HELP>
            <ERROR
                  NEXT="transfer.vml"
                  NEXTMETHOD="GET"
                  REPROMPT="N"
                  TYPE="BADNEXT"
            >An internal error has occurred.</ERROR>
            <ERROR
                  NEXTMETHOD="GET"
                  REPROMPT="N"
                  TYPE="ALL"
            >You are at the wish list dialog.</ERROR>
      </CLASS>
      <CLASS
            BARGEIN="Y"
            NAME="help_customer_number"
      >
            <HELP
                  NEXTMETHOD="GET"
                  REPROMPT="N"
            > Please say a profile name or a seven digit customer
            number.</HELP>
            <ERROR
                  NEXTMETHOD="GET"
                  REPROMPT="N"
                  TYPE="NOMATCH"
```

```
> I&amp;apos;m sorry, I didn&amp;apos;t get that. Please say a profile
name or a seven digit customer number.</ERROR>
<ERROR
        NEXTMETHOD="GET"
        ORDINAL="2"
        REPROMPT="N"
        TYPE="NOMATCH"
> I&amp;apos;m sorry, I still can&amp;apos;t understand you. Please say a
profile name or say a seven digit customer number digit by
digit.</ERROR>
<ERROR
        NEXT="transfer.vml"
        NEXTMETHOD="GET"
        ORDINAL="3"
        REPROMPT="N"
        TYPE="NOMATCH"
/>
<ERROR
        NEXTMETHOD="GET"
        REPROMPT="N"
        TYPE="NOSPEECH"
> Please say a profile name or a seven digit customer
number.</ERROR>
<ERROR
        NEXTMETHOD="GET"
        ORDINAL="2"
        REPROMPT="N"
        TYPE="NOSPEECH"
> Please say a profile name or say a seven digit customer number
digit by digit.</ERROR>
<ERROR
        NEXT="transfer.vml"
        NEXTMETHOD="GET"
        ORDINAL="3"
        REPROMPT="N"
        TYPE="NOSPEECH"
/>
<ERROR
        NEXTMETHOD="GET"
        REPROMPT="N"
        TYPE="TOOLITTLE"
> Please say all seven digits of the customer number.</ERROR>
<ERROR
        NEXTMETHOD="GET"
        ORDINAL="2"
        REPROMPT="N"
        TYPE="TOOLITTLE"
> Please say all seven digits of the customer number digit by
digit.</ERROR>
<ERROR
        NEXT="transfer.vml"
        NEXTMETHOD="GET"
        ORDINAL="3"
        REPROMPT="N"
        TYPE="TOOLITTLE"
/>
<ERROR
        NEXTMETHOD="GET"
        REPROMPT="N"
        TYPE="TOOMUCH"
> Please say just the seven digits of the customer number.</ERROR>
<ERROR
        NEXTMETHOD="GET"
```

```
            ORDINAL="2"
            REPROMPT="N"
            TYPE="TOOMUCH"
        > Please say just the seven digits of the customer number digit by
        digit.</ERROR>
        <ERROR
            NEXT="transfer.vml"
            NEXTMETHOD="GET"
            ORDINAL="3"
            REPROMPT="N"
            TYPE="TOOMUCH"
        />
        <ERROR
            NEXT="transfer.vml"
            NEXTMETHOD="GET"
            REPROMPT="N"
            TYPE="BADNEXT"
        >An internal error has occurred.</ERROR>
        <ERROR
            NEXTMETHOD="GET"
            REPROMPT="N"
            TYPE="ALL"
        >Please say your seven digit customer number.</ERROR>
</CLASS>
<!-- Transition to get customer number step -->
<STEP
        BARGEIN="Y"
        NAME="init"
        PARENT="help_generic"
>
        <PROMPT>You are in the wish list dialogue.</PROMPT>
        <INPUT
            NEXT="#get_customer_number"
            NEXTMETHOD="GET"
            TYPE="NONE"
        />
</STEP>
<!-- Ask for customer number -->
<STEP
        BARGEIN="Y"
        NAME="get_customer_number"
        PARENT="help_customer_number"
>
        <PROMPT>Say a profile name or say a customer number.</PROMPT>
        <INPUT
            NEXT="#validate_customer_number"
            NEXTMETHOD="GET"
            NAME="customer_number"
            TIMEOUT="5000"
            MIN="7"
            MAX="7"
            TYPE="DIGITS"
        />
        <CANCEL
            NEXT="main_menu.vml"
            NEXTMETHOD="GET"
        />
</STEP>
<!-- Validate customer number -->
<!-- TBD this step needs to be implemented using an external grammar that
is capable of  supporting a finite list of profile names and an
"infinite" number of customer numbers. If this cannot be implemented,
then there should be a new step in the dialogue before this step, where
```

```
the user chooses whether the next information will be a profile name or a
customer number, and the current step would be replaced by two steps, one
for profiles (OPTIONS) and another for customer numbers (DIGITS) -->
<STEP
        BARGEIN="Y"
        NAME="validate_customer_number"
        PARENT="help_generic"
>
        <INPUT
                NEXT="#unknown_customer_number"
                NEXTMETHOD="GET"
                TYPE="NONE"
        >
                <SWITCH FIELD="customer_number">
                        <CASE VALUE="7654567" NEXT="#check_toys_in_list"/>
                </SWITCH>
        </INPUT>
</STEP>
<!-- Unknown customer number -->
<!-- TBD this step needs to be implemented using an external grammar that
is capable of supporting a finite list of profile names and an "infinite"
number of customer numbers. The response from the recognition operation
should provide information whether the recognized utterance is a number
or not (an additional step may be necessary). If this cannot be
implemented, then there should two steps in the dialogue, one for the
profile error message and another for the customer number error message -
->
<STEP
        BARGEIN="Y"
        NAME="unknown_customer_number"
        PARENT="help_customer_number"
>
        <PROMPT>I&apos;m sorry, I don't have a customer number <VALUE
        NAME="customer_number"/>. Please say a profile name or say a
        customer number digit by digit.</PROMPT>
        <INPUT
                NEXT="#validate_customer_number"
                NEXTMETHOD="GET"
                NAME="customer_number"
                TIMEOUT="5000"
                MIN="7"
                MAX="7"
                TYPE="DIGITS"
        />
        <CANCEL
                NEXT="main_menu.vml"
                NEXTMETHOD="GET"
        />
</STEP>
<!-- Note: The help info will not be used -->
<STEP
        BARGEIN="Y"
        NAME="check_toys_in_list"
        PARENT="help_generic"
>
        <!-- TBD add database access code to check whether the wish list
        for the acquired profile/customer number has any toys -->
        <INPUT
                TYPE="HIDDEN"
                NAME="toys_in_list"
                VALUE="1"
        />
        <INPUT
```

```
                TYPE="HIDDEN"
                NAME="total_in_list"
                VALUE="fifteen dollars"
        />
        <INPUT
                NEXT="#has_toys_in_list"
                NEXTMETHOD="GET"
                TYPE="NONE"
        >
                <SWITCH FIELD="toys_in_list">
                        <CASE VALUE="0" NEXT="#no_toys_in_list"/>
                </SWITCH>
        </INPUT>
</STEP>
<!-- No toys in list -->
 <!-- TBD this step needs to be implemented in such a way that it can
handle both profiles and customer numbers. If this cannot be implemented,
then there should be two steps in the dialogue, one for the profile error
message and another for the customer number error message -->
<STEP
        BARGEIN="Y"
        NAME="no_toys_in_list"
        PARENT="help_customer_number"
>
        <PROMPT>I&apos;m sorry, the wish list for customer number <VALUE
        NAME="customer_number"/> is empty. Please say a profile name or
        say a customer number.</PROMPT>
        <INPUT
                NEXT="#validate_customer_number"
                NEXTMETHOD="GET"
                NAME="customer_number"
                TIMEOUT="5000"
                MIN="7"
                MAX="7"
                TYPE="DIGITS"
        />
</STEP>
<CLASS
        BARGEIN="Y"
        NAME="help_add_or_review"
>
        <HELP
                NEXTMETHOD="GET"
                REPROMPT="N"
        > Please say add all or review list.</HELP>
        <ERROR
                NEXTMETHOD="GET"
                REPROMPT="N"
                TYPE="NOMATCH"
        > I&apos;m sorry, I didn&apos;t get that. Please say add all or
        review list.</ERROR>
        <ERROR
                NEXTMETHOD="GET"
                ORDINAL="2"
                REPROMPT="N"
                TYPE="NOMATCH"
        > I&apos;m sorry, I still can&apos;t understand you. Please say
        add all or review list.</ERROR>
        <ERROR
                NEXT="transfer.vml"
                NEXTMETHOD="GET"
                ORDINAL="3"
                REPROMPT="N"
```

```
                TYPE="NOMATCH"
            />
                <ERROR
                NEXTMETHOD="GET"
                REPROMPT="N"
                TYPE="NOSPEECH"
            > Please say add all or review list.</ERROR>
            <ERROR
                NEXTMETHOD="GET"
                ORDINAL="2"
                REPROMPT="N"
                TYPE="NOSPEECH"
            > Please say add all or review list.</ERROR>
            <ERROR
                NEXT="transfer.vml"
                NEXTMETHOD="GET"
                ORDINAL="3"
                REPROMPT="N"
                TYPE="NOSPEECH"
            />
            <ERROR
                NEXT="transfer.vml"
                NEXTMETHOD="GET"
                REPROMPT="N"
                TYPE="BADNEXT"
            >An internal error has occurred.</ERROR>
            <ERROR
                NEXTMETHOD="GET"
                REPROMPT="N"
                TYPE="ALL"
            >Please say add all or review list.</ERROR>
        </CLASS>
        <!-- Has toys in list -->
        <!-- This step has a number of equivalent vocabulary entries, only one of
        which is mentioned in the help -->
        <STEP
            BARGEIN="Y"
            NAME="has_toys_in_list"
            PARENT="help_add_or_review"
        >
            <PROMPT>The wish list for customer number <VALUE
            NAME="customer_number"/> has <VALUE NAME="toys_in_list"/> items
            for a total of <VALUE NAME="total_in_list"/>. You can add all to
            the cart or review the list.</PROMPT>
            <INPUT
                TYPE="OPTIONLIST"
                TIMEOUT="5000"
            >
                <OPTION NEXT="list.vml">review list</OPTION>
                <OPTION NEXT="checkout.vml">add all</OPTION>
                <OPTION NEXT="checkout.vml">checkout</OPTION>
                <OPTION NEXT="checkout.vml">buy all</OPTION>
            </INPUT>
            <CANCEL
                NEXT="#get_customer_number"
                NEXTMETHOD="GET"
            />
        </STEP>
</DIALOG>
```

D.4 Transfer Dialogue

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
       Concordia University
       COMP 793 Major Report
       WishToys - transfer.vml
       Transfer to customer rep
       by Miroslav Cosic
-->
<DIALOG>
       <CLASS
              BARGEIN="Y"
              NAME="help_generic"
       >
              <HELP
                     NEXTMETHOD="GET"
                     REPROMPT="N"
              > You call is being transferred.</HELP>
              <ERROR
                     NEXT="#end"
                     NEXTMETHOD="GET"
                     REPROMPT="N"
                     TYPE="ALL"
              >I&apos;m sorry, an error has occurred. Please try again.</ERROR>
       </CLASS>
       <STEP
              BARGEIN="N"
              NAME="init"
              PARENT="help_generic"
       >
              <PROMPT>A customer care representative will be better able to help
              you.</PROMPT>
              <INPUT
                     NEXT="#transfer"
                     NEXTMETHOD="GET"
                     TYPE="NONE"
              />
       </STEP>
       <!-- Transfer to customer rep and exit application -->
       <STEP
              BARGEIN="Y"
              NAME="transfer"
              PARENT="help_generic"
       >
              <PROMPT><AUDIO SRC="muzak.wav"/></PROMPT>
              <!-- Default timeout is 15 secs, after which the "NOANSWER" error
              is raised -->
              <TRANSFER
                     NEXT="#end"
                     NEXTMETHOD="GET"
                     TELNO="18005445555"
                     TIMEOUT="20000"
                     WAIT="Y"
              />
              <ERROR
                     NEXT="#end"
                     NEXTMETHOD="GET"
                     REPROMPT="N"
                     TYPE="BUSY"
              >I'm sorry, our customer care representatives are busy serving
              other customers. Please try again later.</ERROR>
              <ERROR
```

```
                    NEXT="#end"
                    NEXTMETHOD="GET"
                    REPROMPT="N"
                    TYPE="NOANSWER"
                >I'm sorry, our customer care representatives are not available.
                Please try again later.</ERROR>
        </STEP>
</DIALOG>
```