VoiceXML: Usability, Scalability, and the Future

By

Brad Touesnard

National Research Council, IIT

Fredericton, New Brunswick

20 December 2001

UNIVERSITY OF NEW BRUNSWICK

FACULTY OF COMPUTER SCIENCE

Executive Summary

MySQL is a relatively new relational database management system (RDBMS) that is constantly gaining strength, attempting to catch up with the industry leading Oracle Corporation. Lacking decades of Oracle's experience, it is expected that MySQL will take time to gain the full stature of its competitor. To effectively choose a RDBMS many domains must be explored including requirements, architecture, security and integrity, performance, and support.

In general, MySQL's requirements are equally as demanding as Oracle. However, Oracle's years of upgraded functionality makes for an impressive list of features, many of which are not yet included in MySQL. Additionally, Oracle's firm commitment to securing data and transactions is a difficult feat to match. In contrast, MySQL's open source initiative offers a support service like nothing that can ever be provided by Oracle, enabling administrators to debug the code directly.

All-in-all MySQL provides a great database solution for the majority of database applications.

More information on MySQL can be found at MySQL AB's home page (http://www.mysql.com). Equally, the best source of information on Oracle can be found at Oracle's home page (http://www.oracle.com).

Table of Contents

Table of Figures

1.0 Introduction

It took thirty-seven years for cellular phones to become commercially available from their conception in 1947. Today, wireless devices are quickly becoming a part of everyone's daily life. This year, the Strategis Group predicts that "more than 483 million [wireless devices] will be sold to end-users globally, and one third of the world's population will own a wireless device by 2008." (1) This increase in use of wireless devices has spawned a great need for wireless services and applications.

In March of 2000, the VoiceXML Forum composed of AT&T, IBM, Lucent Technologies, and Motorola created a draft of the VoiceXML 1.0 specification. They submitted the specification to the World Wide Web Consortium (W3C) with the aspiration that it would do for Interactive Voice Response (IVR) systems what HTML did for the Internet.

Today, VoiceXML has attained version 2.0 of its W3C specification and has been implemented by most voice browser software vendors. VoiceXML application developers are overwhelming free VoiceXML hosting platforms with processing requests.

Although it has been but three years, VoiceXML has matured considerably since its conception. There has never been a better time to expose the intricacies of VoiceXML to help vendors create improved voice browsers, to guide developers in building better applications, and to reassure decision-makers that they are making the right choice for their business.

The following document details studies conducted to expose scalability and usability issues of VoiceXML applications. A VoiceXML 1.0 application, VIMS (see Appendix A), developed by a co-op student with the NRC, Institute of Information Technology was used in this study and was the focus of all activities.

2.0 Scalability

When developing an application for deployment, developers must always keep scalability in mind. Scalability issues must be foreseen to prevent future problems. To explore scalability issues with the VIMS application, a large database of just over a thousand computer products was added to the previous VIMS database of only sixteen products.

2.1 Database Growth

The original version of VIMS, VIMS 1.0 had a searchable product listing. That is, when you enter the Product Information menu you can say a product and it will find it immediately without having to go through any product lists. This search interface worked exceptionally well with the small sixteen-product database but it was thought that it would not perform as well if additional products were added to the searchable list.

The first test involved comparing the performance of VIMS with a small database against VIMS with a large database. For consistency, a recording of the following script of phrases was used:

1. What is the price of the mouse?
2. Give me the description of the wood table.
3. Tell me everything about the ergonomic keyboard.
4. What is the price of the plastic bench?

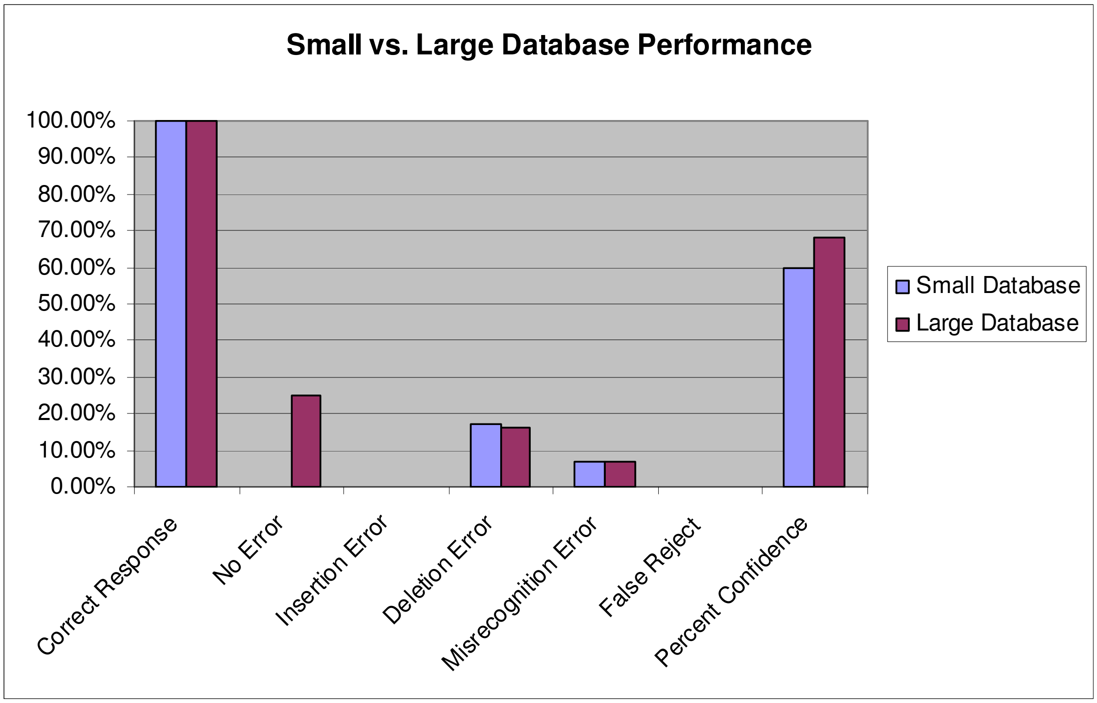


Figure 1: Small vs. Large Database Performance

The x-axis labels are described in Appendix B.

Figure 1 illustrates the results obtained from this experiment. These results show that the increased size of the database has little effect on performance. It is important to note that even though new products were present in the larger database test, the products from the original, smaller database were still targeted. So, if the amount of data does not affect performance, what does?

2.2 Phrase Size

The new data that was added to the original database was designed for the web and therefore a visual interface. Product names were very long and a lot of abbreviations were used which made the phrases difficult for the Text-to-Speech (TTS) engine to pronounce. For example the product name, “CHEETAH 36LP 36.7GB FC HD 10000RPM 40PIN SCA 4MB” would be pronounced “Cheetah thirty six L P thirty six point seven G B F C H D ten thousand R P M forty pin S

C A four M B". It is unreasonable to expect users to say such a long, fragmented phrase and in natural conversation, a person would generally not say such a phrase. Therefore, phrases that are to be verbalized by users should be practically spoken in natural speech. However, it would still be valuable to know the boundaries of phrase lengths.

The next investigation involved targeting the actual products individually and was based on the assumption that as product phrases increases in length recognition confidence decreases. To test this assumption, long product phrases were gradually decreased in size and tested after each decrease. The product phrases used are shown in Appendix C.

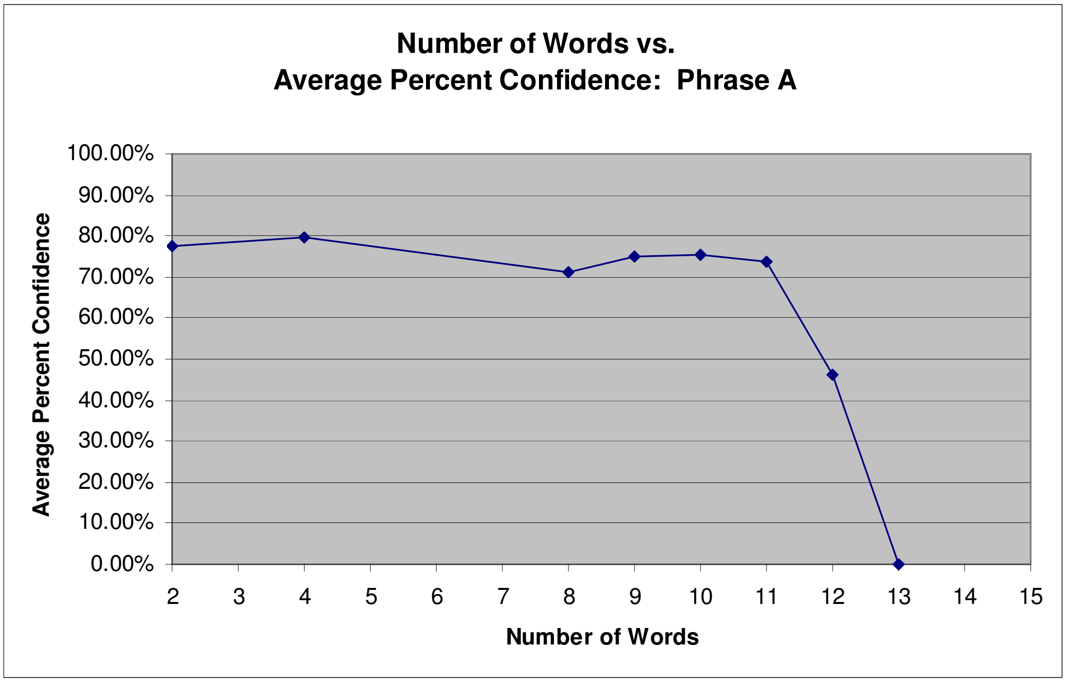


Figure 2.1: Number of Words vs. Average Percent Confidence: Phrase A

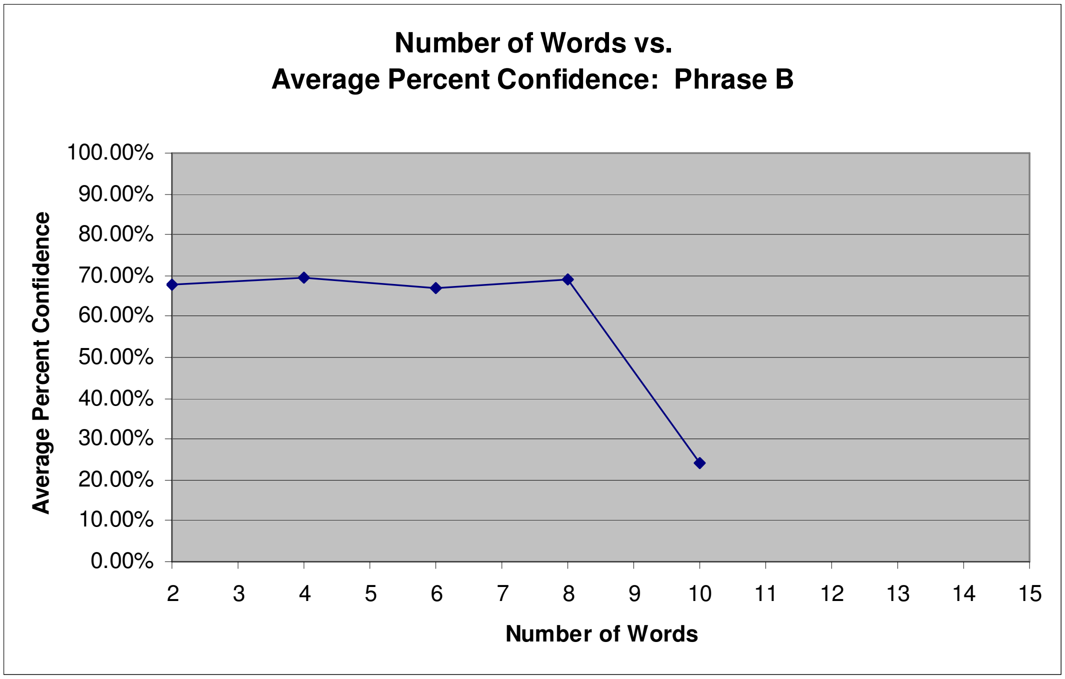


Figure 2.2: Number of Words vs. Average Percent Confidence: Phrase B

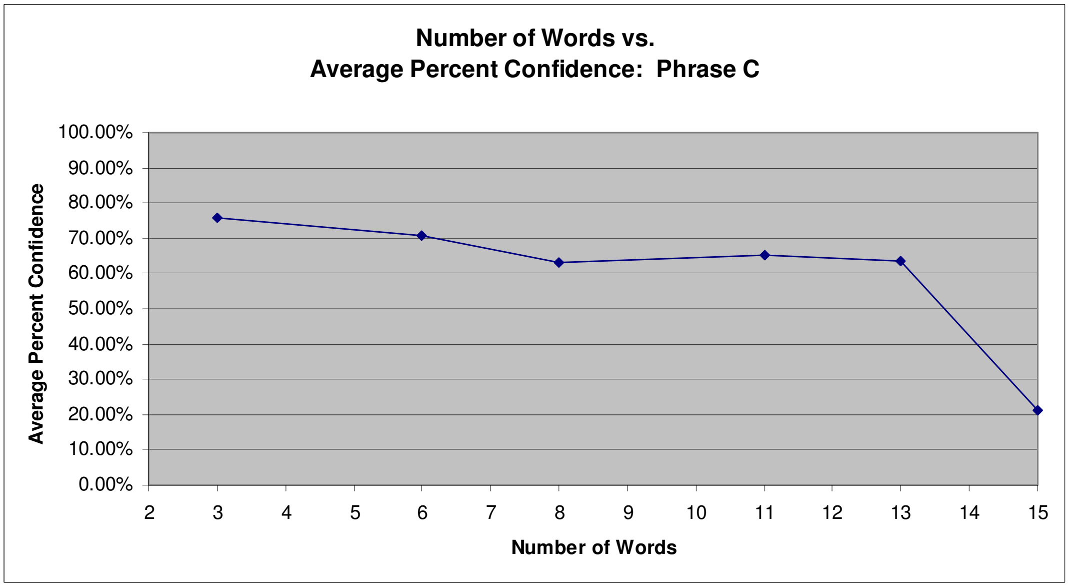


Figure 2.3: Number of Words vs. Average Percent Confidence: Phrase C

From the results illustrated in Figures 2.1-2.3, it is obvious that the assumption was correct, that sorter product phrases are more confidently recognized than longer ones. Furthermore, it is good practice to keep phrases sorter than 6 words to yield better recognition and to keep the dialog natural for the user to pronounce.

2.3 Grammar Issues

The most time consuming activity in this scalability study was formatting the data. Since the data from the large database was originally formatted for a graphical interface (i.e. a web site) with no intentions of ever being used for a voice interface, it required a considerable amount of formatting to agree with the grammar compiler. In fact, every record had to be manually edited.

The GSL grammar format has a limited set of acceptable characters that can be used in its phrases (also know as “tokens”). The characters include lowercase letters, digits, hyphen, underscore, single quote, at sign, and period. When presented with a character outside this set, the GSL grammar parser will throw an error. This means that common characters such as commas and quotations cannot be included in grammar tokens.

When faced with a dilemma such as this, there are two options: manually edit every product or create a filter to strip unsupported characters. The choice depends on whether or not the product names are in need of extra formatting such as replacing abbreviations with explicit meanings and relocating unnecessary information to the description field. In the case where extra formatting is required, manual editing seems to be the only option.

In terms of scalability, grammar can become a major problem over time as the database grows. Every time a VoiceXML page is accessed, attached grammars must be compiled. The compile time is directly related to the size and complexity of the grammar. Therefore, with VIMS 1.0, which wrote all the products to a single grammar file, the compile time of the grammar would increase with the growth of the database. Although it may seem like a minor deficiency at first, this type of scalability issue can grow to completely disable an application. Once the grammar becomes large enough, one of two things will occur: the compile-time delay will take longer than the user is willing to wait, or the VoiceXML browser will timeout waiting for the grammar and the application will exit. How do you prevent these scalability nightmares?

2.4 Preventing Scalability Problems

To avoid the disastrous consequences of bloated grammars, developers must look to architecture and data segregation. Developing solid architecture can help disperse data over several grammars where it would previously have been dumped into one.

The VIMS 1.0 application presented very simple architecture that worked well for the small-scale database it originally accompanied. However when introduced to the larger database it became apparent that this architecture would have to be reinvented in order to decrease the compile time of the grammar.

Because products often have common attributes such as company name and/or category, segregating the product names into several portions and storing the portions in different fields in the database can prove very beneficial to the scalability of the system. Once the data has been separated, the common product attributes can be written to a grammar allowing users to target a group of products. For example, the product name “3Com HomeConnect Etherfast 10 Base-T NIC” could be segregated to the company “3Com”, the product “HomeConnect Etherfast”, the category “NIC”, and the sub-category “10 Base-T” (Figure .

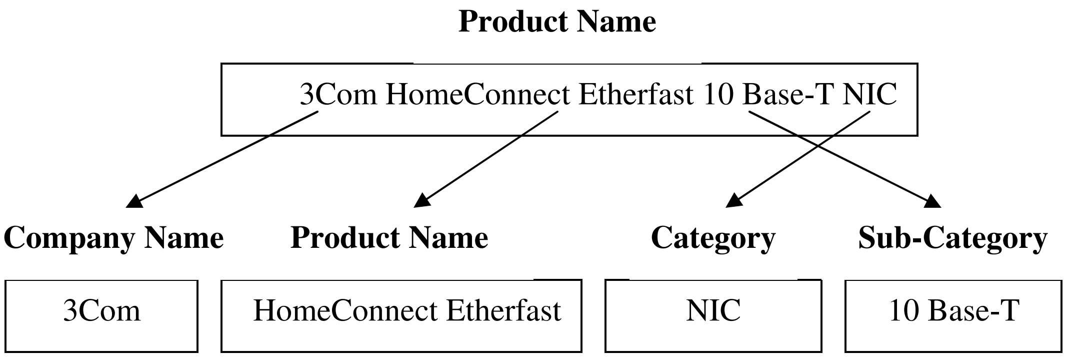


Figure 3: Data Segregation Example

In short, the segregation of data can enhance the flexibility of system architecture thus enabling a decrease in grammar sizes. Fortunately, data segregation also allows developers to create cross-references with attributes to precisely target a given product. Therefore, the steps taken to ensure scalability can also have a favorable impact on usability.

3.0 Usability

"Among speech interface designers, there's a credo: A good GUI and a good VUI are both a pleasure to use, a bad GUI is hard to use, but a bad VUI isn't used at all." (2) Voice user interfaces (VUIs) have several significant disadvantages over graphical user interfaces (GUIs). Among those disadvantages, users have to rely on their memory to remember their choices and visualize the navigation of the system. To explore usability in depth with an application, one of the best approaches is to go to the source, the users.

3.1 Surveying the Users

The survey's purpose is to expose the intricacies of the VoiceXML system from the user's perspective, to realize whether the system will gain public acceptance, and to assure the developer's employer that the system is ready to deploy. The VIMS survey consisted of 12 students from the University of New Brunswick, 10 male and 2 female, ages eighteen to twenty-one. The faculties represented by the students are shown in Figure 4.

The students proved to familiar with the movie and video game products in the database, however few were familiar with novels with only twenty five percent reading at least once per week (Figure 5). It is possible that this low percentage of students reading novels could explain why one student mistaken the directions to find the fictional novel "Lord of the Rings: The Two Towers" for the movie. Only two of the twelve students had never used an automated telephone system in the past. Of the ten other students that had used an automated telephone system, eight said they found VIMS to be equally as good or better than their favorite system.

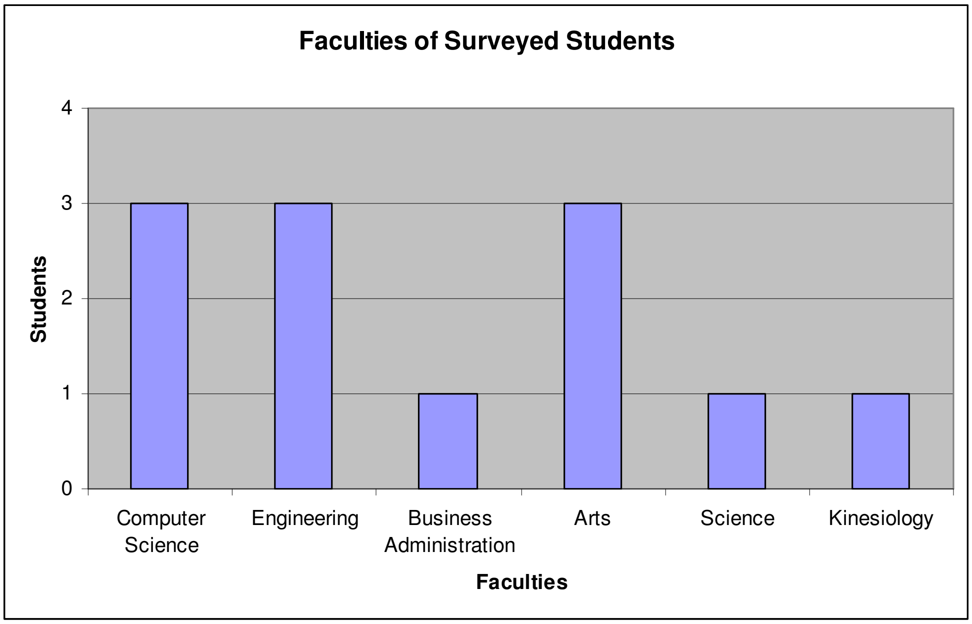


Figure 4: Faculties of Surveyed Students

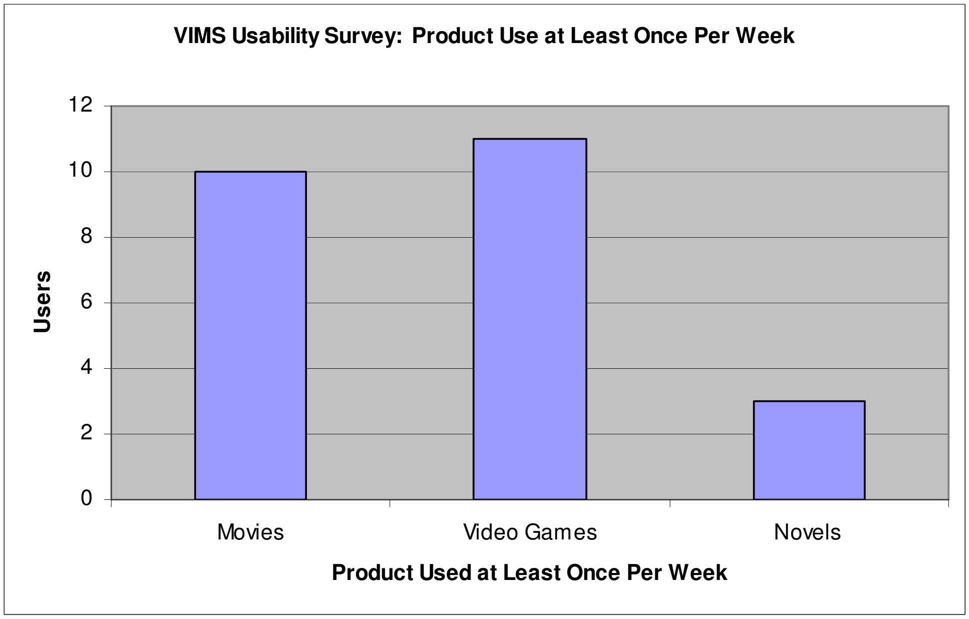


Figure 5: Product Use at Least Once Per Week

To prepare the students for system use, a brief five-minute presentation was given about VoiceXML and VIMS. To ensure all students had the same level of understanding of the system, a brief one to two-minute quiz was given individually before each student used the system. Each student was given an instruction sheet and the same cordless telephone. No further instructions were given until the survey was completed. The average time that users spent on the telephone was 8 minutes and 37 secondes.

Most students found it easy to get the information for which they were searching (Figure 6). Although, the majority found that the Product Information menu was the most frustrating and/or confusing while the Main Menu and Warehouse Information were the least frustrating and/or confusing. Most of the students were attempting to chose a product in a listing by saying the number of the product, but the voice browser did not support this function and would choose the product that was the closest match to the number said. This can be very frustrating at first, but once realized it is not likely to cause much frustration with future use of the system.

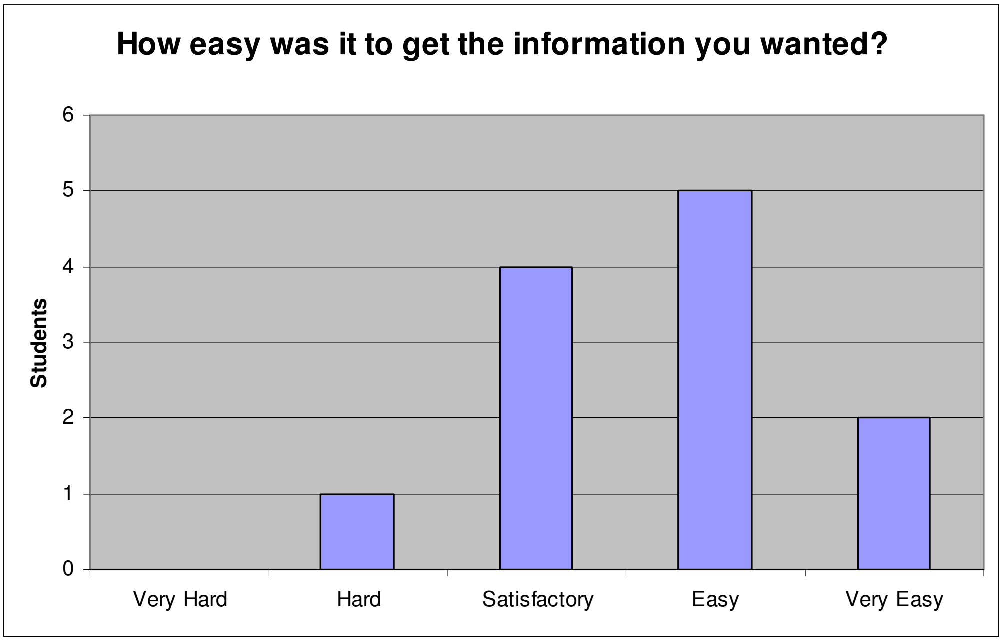


Figure 6: How easy was it to get the information?

Although there were many frustrations with their first use of a VoiceXML system, the students gave generally positive feedback. The majority wanted VoiceXML applications to replace some of the automated telephone systems they currently use. Figure 7 shows the popularity of some of the VoiceXML applications the students would like to see in the future.

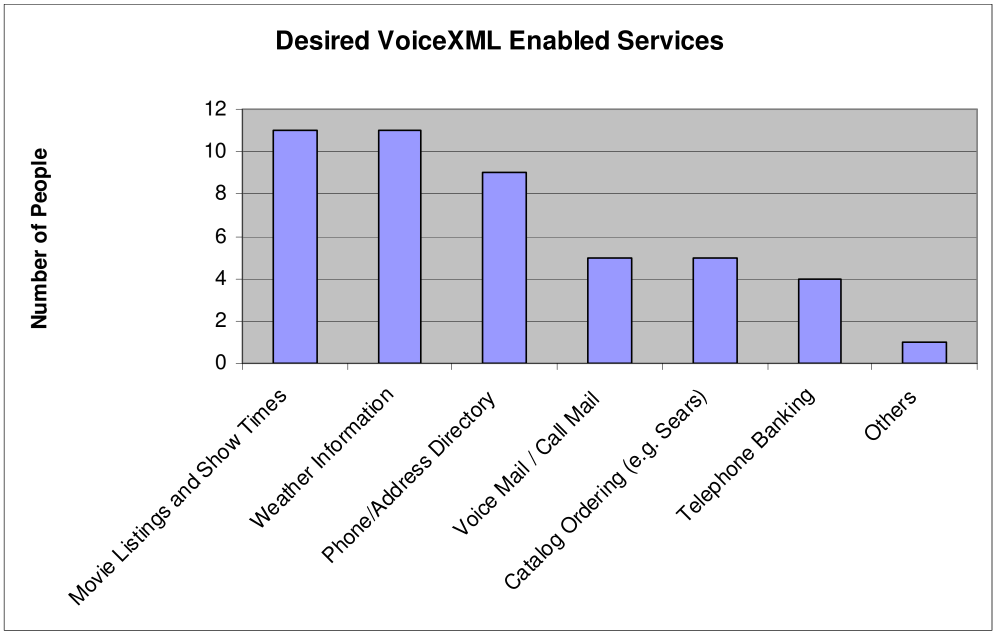


Figure 7: Desired VoiceXML Enabled Services

To discover how well the students could picture the navigation in their mind, they were asked to sketch a diagram of the navigation hierarchy. Students were expected to draw a diagram similar to the one in Appendix A from the “Main Menu” and below. Two of the students misinterpreted the question and drew the VoiceXML diagram that was shown in the presentation. All of the students who interpreted the question correctly managed to draw the Main Menu, Product Information, and Warehouse Information. Most of the students drew detailed diagrams with other system information as well. Figure 8 is an example of a diagram sketched by one of the students.

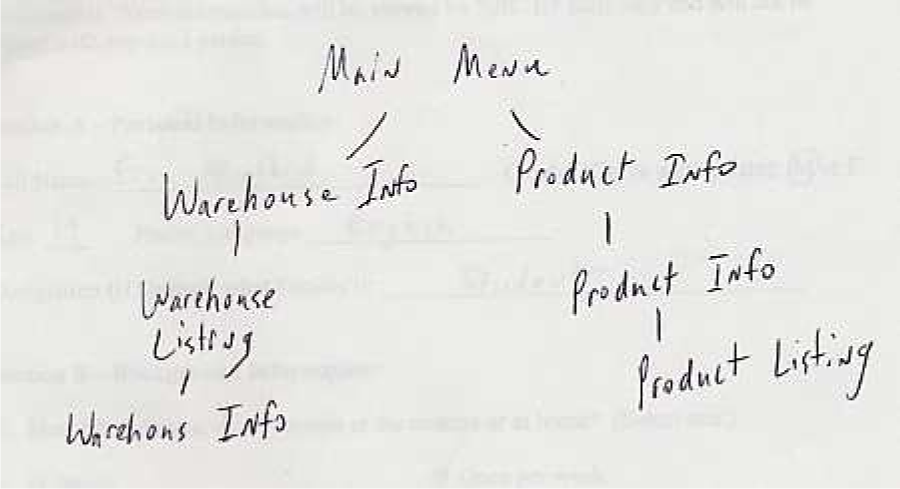


Figure 8: Student Sketch of VIMS Navigation Hierarchy

In summary of the survey, the students delivered mainly positive feedback with negative comments targeting Active Speech Recognition (ASR), Text-to-Speech (TTS), and the response time of the system. These components of a VoiceXML system are developed by third-party vendors and are out of the control parameters of a VoiceXML application. With time, these components will only improve which will improve the overall system. However, at this time there are measures that can be taken to get the best possible performance from ASR and TTS.

3.2 Knowing the Users

It is important to know the audience before building any application, this rule is even more important when developing a VoiceXML application. If it is known that users are familiar with an interface such as the World Wide Web (WWW), the use of “back”, “forward”, and “home” in your voice application would be a natural transition and users would learn the system very quickly. Likewise, if users are not generally familiar with the WWW a different navigation scheme may be better suited. Demographics may suggest that users will be mainly of age fifty or older for a given system. In this case, it would be wise to keep menus concise, as it is known that memories normally become poorer with age.

Knowing the users promotes demographic awareness for VoiceXML developers giving them a clearer picture of the potential problems that can arise with a specific group of users. Not knowing the user base, developers may create a general interface that does not accommodate the majority of users.

3.3 Acronyms, Abbreviations and Names

When importing data from a database created for a visual interface, there were many acronyms, abbreviations and names with which the TTS engine did not agree.

Acronyms such as "CGI" and "KGB" were pronounced correctly. However, acronyms such as "SCSI", which should be pronounced "Scuzzi", were generated incorrectly. The TTS engine that was used speaks the letters in uppercase words as though they were in a list. So, "CGI" is pronounced "C G I". Unfortunately, the TTS engine currently does not recognize acronyms such as "SCSI" which are pronounced as a word. Similarly, the engine will not recognize abbreviations such as "ft." for "feet". These acronyms and abbreviations do not promote high usability, as they are not pronounced by TTS like they would in natural dialog.

To remedy this problem, each of the problematic words must be replaced with a word that will convey natural dialog. Hence, the database records have to be edited manually or a filter must be applied before these words reach the TTS engine.

3.4 Error Recovery

It is a fact that users commonly make mistakes when interacting with a system. Mistakes generally lead to frustration and high tension. To minimize the frustration generated from user error, it is extremely important to have a comprehensive recovery system so users can go back to where they made the mistake and continue with minimal trouble. For example, when you click a link on the WWW and it happens to be the incorrect link you can hit the back button and return to the previous page. Similar navigation can be implemented quite easily with VoiceXML.

Minimizing the instances of user frustration is a key to enhancing the usability of VoiceXML applications.

4.0 Conclusions

Although it is as easy to develop as WWW applications, there are issues with VoiceXML that require extra attention as a result of the sensitive user interface.

Neglect of database and system architecture design can lead to serious scalability problems. It is for this reason that data segregation and architecture is extremely important and must be carefully analyzed. Scalability problems lead directly to usability dilemmas. Scalability neglect can slow down an application, make it more confusing, or in worst case can completely disable the application.

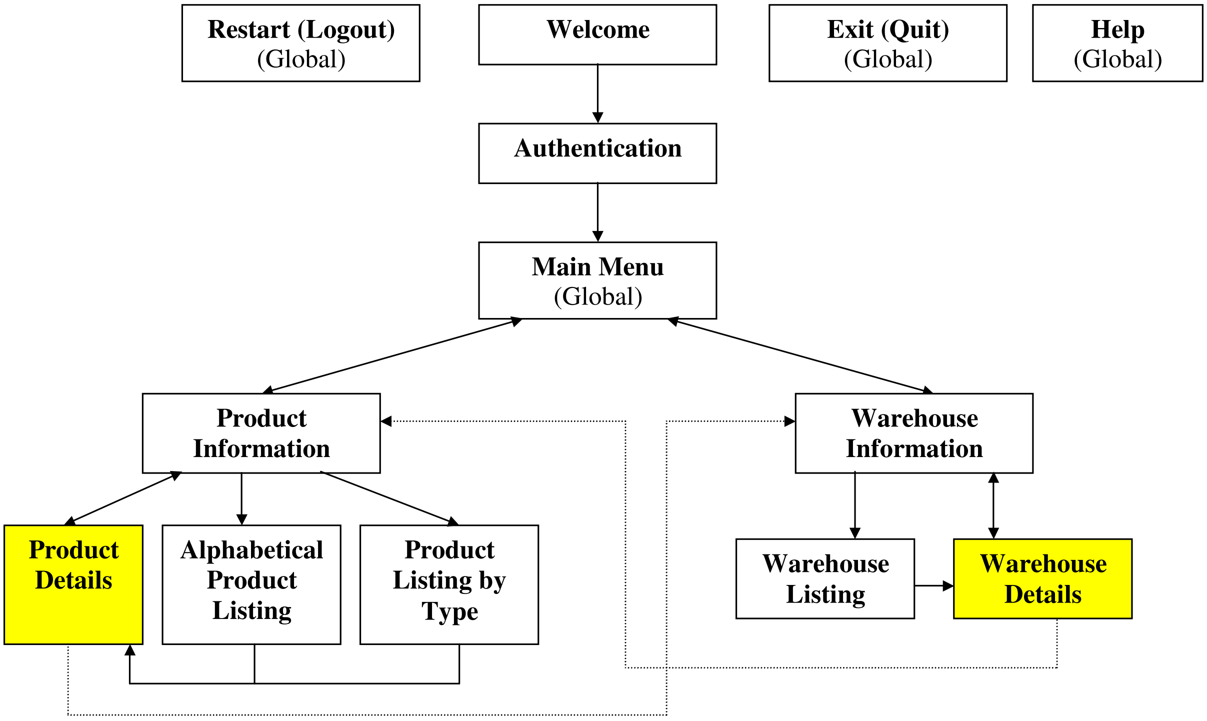
Creating a user-friendly interface can be a challenge for graphical interfaces but for voice interfaces the challenge is gravely increased. Investing a lot of time and effort into exploring the user base of a voice application can prove its worth. By creating a system that suits the specific needs of its users the instance and level of user-frustration can be minimized for optimal usability.

For a voice system that can be developed and maintained with little training, VoiceXML is a high prospect for the future. With the increasing demand of wireless applications, VoiceXML boasts architecture that enables developers to build the flexible and dynamic applications required. Overall, VoiceXML presents an optimistic future.

Appendix A

VIMS, the Voice Inventory Management System, was designed to allow users to access inventory information through the use of speech. Managers and salespeople "in the field" could access VIMS with a cellular phone, providing a quick and easy way to stay up to date on their inventory.

VIMS 2.0 Control Flow Diagram



Broken Lines indicate that the path may not be available depending on previous choices.

Appendix B

Testing conducted for this study often involved the following criteria:

- Input gives correct response.
- Recognized speech is word for word the same as input.

  If the recognition is not correct, the resulting errors are put into three categories:

  Insertion error: extra words in recognized speech.

  Deletion error: words missing from recognized speech.

  Misrecognition error: recognizes a word different from one that was spoken.
- False reject: the system does not accept input that is in the grammar.
- Percent confidence: This is level of certainty that the Active Speech Recognition (ASR) believes it has interpreted the input correctly.

Appendix C

Phrase A

1. Joe Frank Bob Sue Jim Sam Mike George Chris Robert Steve Tim Bruce
2. Joe Frank Bob Sue Jim Sam Mike George Chris Robert Steve Tim
3. Joe Frank Bob Sue Jim Sam Mike George Chris Robert Steve
4. Joe Frank Bob Sue Jim Sam Mike George Chris Robert
5. Joe Frank Bob Sue Jim Sam Mike George Chris
6. Joe Frank Bob Sue Jim Sam Mike George
7. Joe Frank Bob Sue
8. Joe Frank

Phrase B

1. International Business Machines WebSphere Voice Server Development Package Enterprise Edition
2. International Business Machines WebSphere Voice Server Development Package
3. International Business Machines WebSphere Voice Server
4. International Business Machines WebSphere
5. International Business

Phrase C

1. 3 com Etherfast Ten Base T Ethernet Adapter with Wake Up On Lan Retail Only
2. 3 com Etherfast Ten Base T Ethernet Adapter with Wake Up On Lan
3. 3 com Etherfast Ten Base T Ethernet Adapter with Wake Up
4. 3 com Etherfast Ten Base T Ethernet Adapter
5. 3 com Etherfast Ten Base T
6. 3 com Etherfast

Bibliography

1. Abbott, Kenneth R. *Voice Enabling Web Applications: VoiceXML and Beyond.* Berkley, CA, US: Apress, 2002.

2. unknown. The Strategis Group Predicts One Third of Global Population will have a Wireless Device by 2008. Strategis Group. [On-Line]. Available: http://www.strategisgroup.com/press/PressReleaseDetail.asp?ObjectId=49416 [Accessed on 2002-11-25]