Input Methods for Notification Systems: *A design analysis technique with a focus on input for dual-task situations*

Chuck Holbrook

Submitted to the Faculty of
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
IN
COMPUTER SCIENCE

June 2003

Keywords: Notification Systems, In Vehicle Information Systems, Input Methods, Character Recognition, Scenario Based Design, Graffiti

INPUT METHODS FOR NOTIFICATION SYSTEMS: a design analysis technique with a focus on input for dual-task situations

Chuck Holbrook

Abstract

Design and evaluation of input methods for secondary tasks in dual-task systems presents specific challenges not covered by traditional human-computer interaction design techniques. Emerging trends in the fields of mobile, ubiquitous, and in-vehicle information systems demonstrate a desire for users to interact with information systems while engaging in other tasks. Research on interaction within these various fields has revealed input methods that perform well for a particular task. However, few focus on the tradeoffs of attention that must be made to react to this notification information. A design analysis technique for input methods is proposed focusing on the design objectives of interruption, reaction, and comprehension for the secondary task made at the cost of primary task attention. Through a study conducted using a reusable usability test platform constructed for this thesis, a typical in-vehicle information system is analyzed using the proposed design analysis. Three input methods were designed and compared: a graffiti character recognizer, a touch screen, and a remote control for their proficiency at selecting an item from a list while operating a driving simulator. The results of the study revealed similar task performance between the varied input methods; however, the design analysis enabled recommendations about future design directions, confirming the viability of the technique for notification systems research.

Acknowledgments

I would like to thank Professor Scott McCrickard for his supervision and his support for carputers. I would like to thank Christa Chewar for her guidance, support, and editorial skills that made this thesis readable. I would like to thank Maxim Moldenhouer for his contribution of the simulator used for the usability study. I would like to thank Ali Ndiwalana for his formatting advice. I would also like to thank my friends Catherine, Jon, and Ranjit for their support throughout this year.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

Imagine a world where computers are seamlessly integrated into all parts of our lives and interacting with them comes as naturally as thinking; you simply move your arm or say a simple phrase of what you want done. You do not have to contemplate the syntax that the computer requires, in fact you may not even know there is a computer processing your requests. You just know that if you need information, it will be there, without a second thought. This is a world where input methods have been perfectly matched with secondary task user goals. It may sound far-fetched and imaginary, but the technology involved to make this utopia a reality is currently available. What remains is understanding how to interact with these ubiquitous systems in a way that is unobtrusive and yet still functionally complete.

1.1 The Need for Input Methods for Notification Systems

Unlike input methods for standard computer use as a primary task, designers of input methods for systems with secondary tasks not only have to consider the goals presented by the system controlled by the input method, but also have to consider the user needs related to the primary task. The need for secondary information does not simply stop because another, and more significant, primary task begins. In fact the need for many forms of supplemental information increases as other tasks occur. For example, the desire for users to be appraised of road and traffic conditions increases as soon as they begin driving. This information is often then followed by a need for input to the system, such as entering a new destination for a Global Position Satellite (GPS) system. However, it is not likely that users will give up or even want to hinder their driving task just to interact with the GPS system. This illustrates the common challenge in secondary tasks to understand how a user can obtain the desired utility from a secondary task while not compromising the goals of the primary one. In such systems, not only does the delivery of content have to be carefully considered, but in many cases, there is also a need to react to and control the system on a peripheral level—presenting an even greater challenge to designers.

Traditional input methods are designed without consideration for the impact they may have on other tasks the user is engaged in and often fail to be as effective when used with notification systems. Past research specific to a variety of different domains has highlighted many of the challenges posed for input for notification systems and revealed some potential solutions. For example, research into ways of better interacting with mobile computers revealed that simply augmenting existing visual-manual input methods with audio feedback can hold significant improvements on the tradeoff of visual distraction for secondary task performance (Brewster and Walker 2002). Innovative designs incorporating exotic joystick-like input devices were also able to obtain superior performance for specific tasks done in the periphery by diminishing the cognitive demands and therefore the distraction posed by the secondary task (Kawachiya and Ishikawa 1997; Myers, Miller et al. 2000). Comparison of tradeoffs for existing input methods within the fields of mobile computing and in-vehicle information systems have also revealed that potential benefits and drawbacks exist when optimizing the input method for dual-task environment (Manes and Green 1997; Clausen and Pedersen 1998; Tsimhoni, Smith et al. 2002).

Although these research efforts have produced significant results in and of themselves, they have not yet resulted in a thorough understanding and general applicability of just how to interact with systems while maintaining awareness of another task. Specifically, research has not revealed at what level of interaction the utility gained from a secondary task begins detracting from the goals of the primary one. This attention utility theme is a main distinction between the study of notification systems and traditional HCI research. Wild innovation may solve some issues, however, the real contributions to the field will most likely come in the form of scientific methods that unify and form a foundation from which further research efforts can be based (McCrickard, Catrambone et al. 2003).

1.2 A Design Analysis

This thesis proposes an analytical method for designing and evaluating input methods used to interact with secondary tasks. It consists of using a scenario based design (SBD) technique to make claims about the inherent distraction caused by the specific interaction technique and to analyze the notification system critical parameters of interruption, reaction, and comprehension. This forms a description of the input method with respect to its potential impact on the primary task and ability to support the goals of the secondary task. An empirical validation phase is incorporated to validate the claims and appraise the role that they have on the notification system. The results of a study through this process will reveal the success or failure of the input method. The claims made earlier will provide a basis for making design recommendations, improving the notification system compatibility with a particular input method or suggesting a better method that meets the goals of the system.

Validation of the design analysis technique was supported with a practical application focused on an in-vehicle notification system. In a study conducted on the

system, a number of users were asked to perform two tasks simultaneously while using one of three different input methods. These included a scroll and select method implemented with two arrow buttons and a single select button located on *an infrared remote control*, a finger based touch pad *character recognizer*, loosely based on the graffiti language (Graffiti 2003), and a scroll and select method implemented with static buttons on a *touch screen*. A variety of different metrics were used to capture the various input methods' level of performance in relation to the primary task and the user's ability to comprehend and react to the secondary task. The results of the study were somewhat surprising in that the widely varied input methods resulted in remarkably similar task performance; however, the design analysis enabled recommendations about future design directions, confirming the viability of the technique for notification systems research.

Chapter 2: Notification Systems Background

This chapter will present a general introduction to notification systems. The theory will be defined and supported with various examples, demonstrating the need for a specific classification for notification system tasks.

2.1 What is a Notification System

Computer technology has given us the ability to access to a wealth of information ranging from data gathered with real world devices like a Global Positioning Satellite (GPS) device to real-time Internet-enabled information like stock prices or current events from around the world. This information often remains valuable even when another, more significant task *(primary task*) is ongoing and engaging the user's resources. In this case users may need a system that facilitates monitoring of information over a long period of time (*secondary task*) and supports simultaneously performance on other tasks. This presents a *dual task situation* in which a user must balance the attention needs of the primary task with gaining utility from the secondary source of information. The need to take into account the goals a user has for two simultaneously occurring tasks is what separates notification system design problems from traditional HCI research.

McCrickard has named systems that present information in a secondary process as notification systems. Formally, a notification system is defined by its characteristics that "attempt to deliver current, important information to the computer screen in an efficient and effective manner, while a user is engaged in other tasks." (McCrickard, Catrambone et al. 2003) These systems can present any type of information that a user desires and can be related to topics present in the primary system or topics having nothing to do with the user's primary goals. Although visual output to a computer screen is the typical information delivery mode for these types of systems, notification concerns include a much broader range of output methods, such as those that just targeting the aural or tactile senses of the user.

2.2 Why Notification Systems Matter

Notification systems are often found in use as mobile or other types of devices of a ubiquitous nature, allowing users to gain access to information while interacting in the real world. These types of devices have become exceptionally popular in recent years, and aside from fulfilling typical notification delivery needs, they also include a user interaction component while other tasks are ongoing. Personal Digital Assistants, mobile phones, and even in-vehicle information systems are all examples of these types of notification systems, which are increasing in popularity. All these devices may present information to a user when they are engaged in other more pressing matters, such as conversing with others, walking down the street, or driving. All these real world activities compete for the same resources used to access and absorb the information delivered by notification system; however, users still want to be able to interact with these devices. Designers must take this into consideration when developing applications for these usage requirements if they are to meet the goals of the users. Designing a notification system for a secondary task requires a designer to understand how to distribute information to the user in a manner which compliments the primary task and simultaneously allows gains in awareness, as well as how to support interaction with secondary information.

2.3 Example Notification Systems

Examples of notification systems can be found in a large variety of domains and a number of different platforms, however, desktop systems offer the most straightforward examples. The abundance of processing power and multitasking capabilities have allowed today's regular desktop computers to host a multitude of different notification systems. One of the more popular examples is America Online's instant messenger client (Figure 1)(AIM). The program primarily allows real-time text based communication with others from a preselected list of people (AIM 2003). From a notification systems standpoint it provides many functions that operate in the periphery of the user's focus such as indicating the status of people from the list and signifying whether they are available or not. It also provides a ticker interface that can display a variety of information from different sources, including news headlines or stock quotes. This system is characterized as notification system not because of its content, but rather because of the interface design techniques that it employs. For instance, knowing whether someone is online or monitoring stocks through the ticker is not a task that requires a significant amount of cognitive effort and be possible without much distraction to the user. The presence of the "always on" notification portion of AIM is one of the only major differences between this interface and more typical chat based ones such as MIRC (mIRC 2003). However, the popularity of AIM over these more traditional chat programs gives evidence to the burgeoning role of notification systems research in computing.

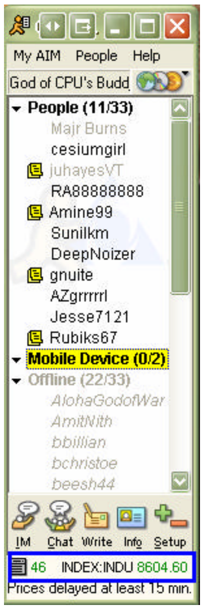


Figure 1. AOL Instant Messenger (AIM) client showing the list of people online and their status as away, idle, or active. The scrolling stock ticker is highlighted in blue.

The "always on" interruptive chatting capabilities of the AIM client, which allows users to receive instant messages from other users and then immediately begin carrying on a conversation with them, illustrates another point about notification systems: they are dynamic in that they can switch from being a notification system one minute to a primary task in the next. Any notification system ceases functioning as one as soon as the user makes it the dominant focus of attention for an extended period of time. After the switch, the tradeoffs for attention and utility that should prevail in the design of the interface no longer apply. This also applies to input methods used in conjunction with these systems. Using the keyboard to chat with another user in instant messenger is not an input method that would benefit much from dual task optimizations, since the user is now primarily concerned with the conversation taking place. However, if the user desires to interact with another task while continuing to work, such as changing the current song playing on an MP3 player in the background, this task would benefit greatly from an input method that took this into consideration.

2.4 Non-Desktop Notification Systems

Although input methods used for notification systems exist on the standard desktop platform (e.g. multimedia keyboards for controlling MP3 players (Figure 2)), less conventional platforms present a much greater need and more challenging environment for input. When personal digital assistants (PDAs) or other forms of mobile computers are used, it is often for complimentary secondary tasks to other more important primary activities. For example, as Clausen and Pedersen detailed in their comparison of input techniques for PDAs, the true power of the PDA comes from the ability to use it while focusing on another event, such as taking notes while listening to a lecture, working in a dynamic group environment, or looking up addresses to give to another person on the cell phone (Clausen and Pedersen 1998). For all of these tasks, it is imperative that a balance between the attention demands of interacting with the PDA not interfere with the needs of the primary task. It is important to be able to lookup a phone number quickly or take accurate notes on a lecture, but these are useless tasks if you lose contact with the person you are giving the number to or can no longer pay enough attention to the lecturer to comprehend the main points.

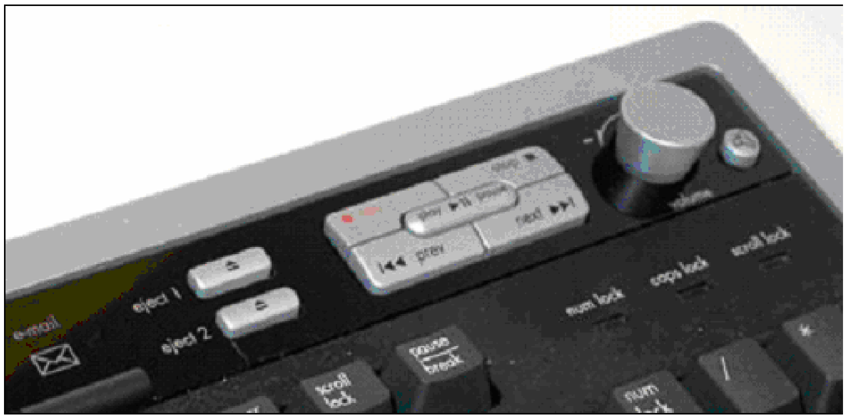


Figure 2. A Multimedia Enhanced Keyboard. The keys on the top of the keyboard can be used to control background tasks while working in another application (Shimpi and Anad Lal, 2003).

In some domains, the success of a task supported by a notification system plays an absolutely critical role. This is perhaps best illustrated by the domain of In-Vehicle Information Systems (IVIS), in which a variety of applications attempt to give drivers whatever information they desire while still maintaining a safe driving environment (Lee, Kantowitz et al. 1994; Anderson, Abdalla et al. 2001). Gallagher indicated the serious nature that the increased attention demands of an in vehicle information system can have on driver ability, noting the already staggering annual costs of 70 billion dollars and the estimated 42,800 deaths resulting from accidents in 2000 (Gallagher 2001). New and more complex computer technology entering the domain makes it more important than ever to develop a thorough scientific foundation for understanding how users can obtain the utility they need from an information source in a non-distracting way. Global Positioning Satellite (GPS) applications are one of the better known instances of these notification systems beginning to enter the in-vehicle market in considerable force. However, they pose a potentially significant safety risk (Nowakowski and Green 1998; Tijerina, Parmer et al. 1998; Anderson, Abdalla et al. 2001; Tsimhoni, Smith et al. 2002). This type of application not only places requirements on the user to understand complex visual map data while driving, but also requires a sophisticated input method to enter data for any one of possibly millions of destinations.

The popularity of the IVIS field has resulted in the production of a significant amount of research related to the domain; however, the field is still lacking in rigorous empirical studies to form a common foundation for future work to be built upon (Anderson, Abdalla et al. 2001). A brief literature review within the domain revealed that balancing attention demands are a common theme, although the way in which attention constraints are described varies from document to document, referring to the same problem of unwanted interruption as “overloaded,” “inattentive,” “unfocused,” or “distracted” driving. Few studies quantify the problem in a way that can be used for universal comparison (Green 1999).

2.5 Notification System Critical Parameters

Other wide ranging research efforts with a notification system focus include the study of tickers, large screen displays, and information visualization (McCrickard 2000; Zhao and Stasko 2000; Bartram 2001). Although the examples of notification systems vary greatly in their functionality and domain of applicability, they still share a number of design objectives in common. McCrickard et al. has suggested that these commonalties can be quantified through three different, often conflicting critical design parameters: *interruption* to primary tasks, *reaction* to specific notifications, and *comprehension* of information over time (McCrickard, Catrambone et al. 2003). These critical parameters, collectively referred to as *IRC*, promise to establish a metric for user goals present in a system by which all notification systems can be classified and compared, regardless of functionality. These parameters give designers a common framework to consider a design. They can also be used to construct evaluations for systems, highlighting all the tradeoffs made for different design decisions and, ideally, indicating all possible side effects for the independent variables under question. For instance, in the AIM example, interruption would be a necessary user goal for many features, such as for the notification of incoming messages from other users or when the status of another user changes. Reaction would be an important user goal when a user is concerned about changes in a certain stock being monitored or responding if someone comes online. The tradeoffs often come in when the levels of interruption and reaction are changed independently. Finally, it might be important for the system to help the user remember conversations with others, indicating a need for some level of comprehension support.

2.6 Notification System Applicability

The background just given described how various systems can be analyzed from a notification system perspective by translating user goals into the critical parameters of interruption, reaction, and comprehension. This theory has been demonstrated to work well in developing a system design that takes the context of the primary task and secondary task into consideration (McCrickard, Catrambone et al. 2003). However, the tradeoffs made for different ways of interacting with these systems are not formally addressed. The next chapter will introduce the concept of an input method and relate it to notification systems and introduce the specific problems that have not been addressed yet by existing research.

Chapter 3: Input Methods for Notification Systems

This chapter will focus specifically on input methods, describing what one is and giving examples, what past research has focused on, and, more to the point, why they need to be considered in a notification system design context.

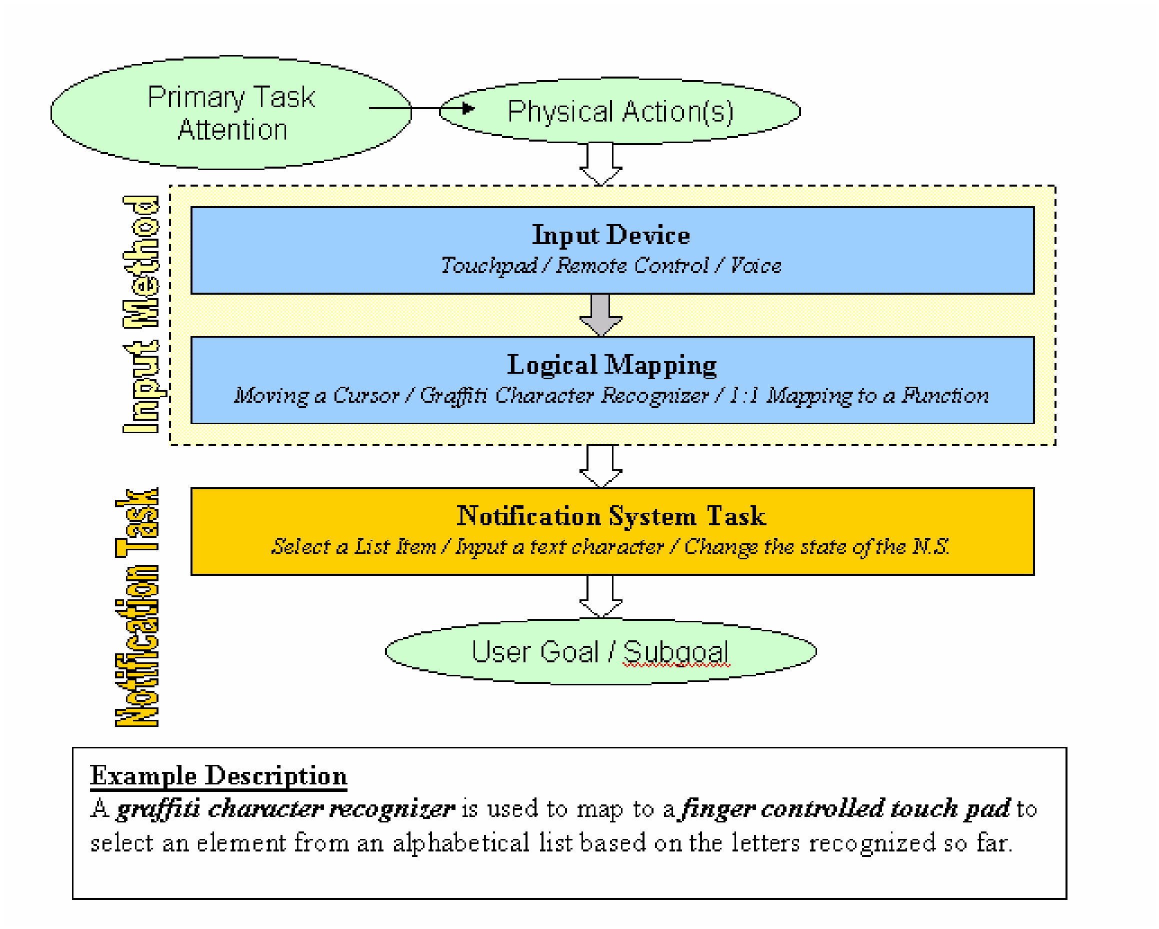
3.1 Impact on Notification Systems

Although as the term "notification" denotes, the systems classified as such are first and foremost responsible for delivering some form of information output to the user, most notification systems still require some sort of feedback from the user on the type of information to provide. In many instances this feedback is static and only needs to be specified once, such as stating which email server to monitor for incoming messages for an inbox notification system. At other times, more dynamic interaction is required, for instance allowing a user to react to a notification by replying to or dismissing an incoming email message. This can easily be accomplished on a desktop system through the use of a message box prompting the user to click a reply or dismiss button with a cursor controlled by the mouse. This may seem like an insignificant example, but slight changes in how the message box operates can drastically affect the performance of the system in terms of interruption and reaction. A bigger message box may make it easier to react to incoming mail by providing a larger area to click on, but as a tradeoff, may interrupt the user's primary task more if it blocks more of the foreground application.

Of even more significance is the role that input methods play in more constrained platforms that do not have a standard means of interacting like a mouse. Consider this same email notifications system operating in a vehicle—this raises many design decisions for the input method. A mouse is obviously not appropriate to use while driving, raising the question which input method can best meet the goals of the secondary task while not interfering with the primary task. How much more interruptive would it be if a button usually used for the primary task was now responsible for dismissing the notification when it was shown? Would it be better to simply make the notification disappear after a specific time than to make the user become aware of an additional system state? Also, what effect would each of these choices have on the user's reaction to and comprehension of the events? In exploring a solution methodology for these questions, the current state of the art in input methods must be covered.

3.2 What is an Input Method

The term *input method* has been used for a number of different concepts ranging from obtaining input in a standard way for users of different languages to actual hardware devices used for interaction. However, in this thesis, an input method refers only to the way in which a user interacts with a system. Formally, an input method characterizes both the device that the user interacts with and the logical mapping that the software uses to interpret the commands sent from the device (Figure 3). For example, on a desktop computer, a mouse could be the input device and a mapping would be a cursor, while on a PDA, a touchpad could be the input device and a mapping would be a graffiti character recognizer. These examples could also be reversed so that moving a finger on a touchpad controls a cursor and moving the mouse in a particular fashion generates a graffiti character.


Primary Task Attention
Physical Action(s)
Input Method
Input Device
Touchpad / Remote Control / Voice
Logical Mapping
Moving a Cursor / Graffiti Character Recognizer / 1:1 Mapping to a Function
Notification Task
Notification System Task
Select a List Item / Input a text character / Change the state of the N.S.
User Goal / Subgoal

<u>Example Description</u>
A ***graffiti character recognizer*** is used to map to a ***finger controlled touch pad*** to select an element from an alphabetical list based on the letters recognized so far.

Figure 3. An input method detracts from primary task attention when users make physical actions that send signals to a *logical mapping* from an *input device*. This causes a change in the state of the notification system task that completes a user goal.

The definition is meant to qualify an input method as a higher level term, combining the physical constraints that an input device imposes on a user as well as the logical meaning that is applied to the signals coming from it. An input method for a notification system should be stated in these specific terms as either a simple change in the hardware or the software characteristics that can drastically affect the way the user interacts with the notification system. For example, there are a number of different devices that can be interpreted in a way that results in a character being generated, the keyboard being the most popular and logically simple of these. However, there are many instances in which such a device would not be appropriate. Many times using a QWERTY keyboard may actually hinder performance or lugging a full-size keyboard around may just be too inconvenient. The PDA platform illustrates this fact through its varied number of text input methods, each with their own tradeoffs in design that make them better or worse then keyboards in certain aspects (Pavlovych and Stuerzlinger; Mankoff and Abowd 1998; Masui 1999; Ward 2001; Alpern and Minardo 2003). Examples of these include character recognizers such as Graffiti (Figure 4)(Graffiti 2003), mapping number keys to letters through the MultiTap algorithm (MacKenzie and Soukoreff 2002), and path based character selection demonstrated by Cirrin (Figure 5)(Mankoff and Abowd 1998).

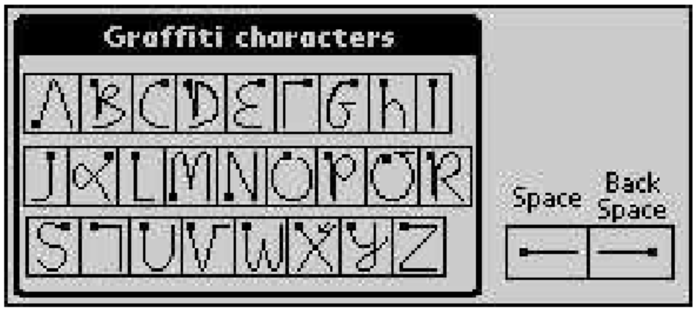


Figure 4. The Graffiti Alphabet (Ward 2001).

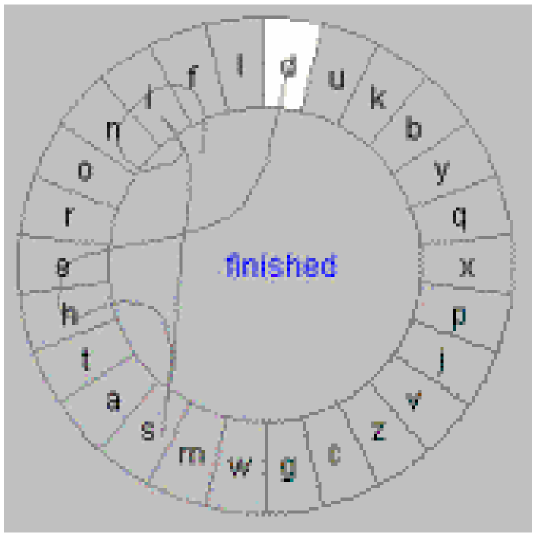


Figure 5. The word "finished" written on cirrin, a circular, word-level soft keyboard (Mankoff and Abowd 1998).

When considered in the context of notification systems, the input method can become of significant importance, as simple changes to how one interacts with a computer can alter the amount of attention that using the device requires. This is particularly true for input methods on constrained platforms (Pavlovych and Stuerzlinger; Mankoff and Abowd 1998) like cell phones which work with a limited 12 key keypad and require more concentration from the user to accomplish text input functions. Besides limiting speed and raw performance, the fewer keys limit auxiliary options that notification systems could possibly take advantage of. For instance, the disambiguity that exists through the number of input choices available on a typical desktop system with a 101 key keyboard allow for unused keys to be bound to a secondary interface. Similar to how the ctrl+alt+del keys perform a system function rather then an application function, key combinations not used by the primary task could allow a user to interact with a notification system when an interaction is necessary. Considering this fact and that short breaks from a primary task are usually acceptable for interacting with notification information on a desktop computer, the need for input methods there is relatively small.

3.3 Critical Parameters with Input Methods

Notification system theory attempts to identify user goals in terms of interruption, reaction, and comprehension of the secondary task. Input methods designed for such systems can take advantage of this already classified design knowledge to optimize interaction for the task as hand. It is possible to identify specific input methods that actually have a tendency to promote certain user goals, such as a speech recognition system improves comprehension since that users must remember the names of items within the system. However, input methods can not easily be classified as improving interruption, reaction, and comprehension on their own.

3.3.1 Interruption

For example the term interruption denotes reallocation of attention from the primary task to the secondary. For this to apply to an input method it would mean that the device is responsible for this reallocation. However, this introduces a contradiction. Users would not interact with a secondary task prior to redirecting their attention. Using the input method may cause more distraction from the primary task, which is similar to interruption, but serves a completely different and destructive purpose to the user's goals. The interruption goals inherent in a notification system can, however, be used when designing the interaction technique for an input method. If interruption is a user goal present with a secondary task, the attention shift provides an opportunity to support a more accommodating environment for interaction to take place. Speech recognition, for example, would benefit over many other input methods from the presence of an interruption user goal in a largely audio based notification system. The system could be automatically muted when attention is needed for the secondary task and compel the user to vocally interact with it.

3.3.2 Reaction

Similarly reaction may not be considered as directly related to input methods. This user goal denotes the accuracy and urgency to which a response to a particular notification is needed. A faster response time with a particular input method will allow the user goal of quick reaction to be met, but it would not make sense to use a slower input method for the opposite case where only gradual reaction is called for. Obviously, between two input methods that are the same in every respect but speed, the one supporting faster performance is going to be the better interaction technique. There are, however, cases where slower reaction time can allow for other beneficial tradeoffs. Classifying input methods that offer tradeoffs for reaction time can help identify the best input method for the reaction goal of a particular secondary task. For example, selecting an item from a very large list box will generally take longer then simply spelling an item out; however, if a slow reaction is all that is needed, the former method will offer the lesser cognitive demands of recognition over recall and facilitate lower interruption goals.

3.3.3 Comprehension

Lastly, comprehension could presumably be improved by certain input methods, like the speech recognition example given earlier, but logically this role would work better in reverse. Specifically a system designed for a given level of comprehension may be improved by an input method that enables a user to effectively interact with certain aspects of the information in the notification. The only difference between them will be how they reach their goal. Users do not remember nearly as much of the data encountered when interacting compared to the data that is the goal of their interaction. This will be validated later by the usability study conducted for this thesis. If comprehension is shown to be promoted by notification system design features, this information can be used to select an input method that, as with reaction, may have tradeoffs that only come into play when this user goal is supported. For example, a system that supports recall over recognition, such as the list example given for previously, will be more usable (less cognitively demanding) when users are familiar with the information in the system and as a benefit, support faster reaction times.

In summary, although some input methods may be built that change the level of interruption, reaction, and comprehension perceived by the user, this thesis will focus on how to design input methods that match the goals that the secondary task was designed for. This is a subtle difference, but, as was just illustrated, has wide ranging implications. The least of these implications is that the former approach, in the cases where it is possible, may promote changing an already designed notification system during the input method design process. For example, bigger fonts may be linked to promoting comprehension in a recognition based input method may be preferable over one that takes advantage of recall. However, changing this design feature of the notification system will obviously have implications for the other tradeoffs the system was designed to meet and therefore should be addressed in more typical notification system design methods.

3.4 Example Input Methods

A survey of input methods will now be presented. The following list of methods is neither complete nor does it achieve full depth, rather it attempts to describe only the input methods that illustrate a particular aspect important to notification systems. For a more exhaustive survey of input methods, please see *Text Entry for Mobile Computing* (MacKenzie and Soukoreff 2002), *Adaptive Computer Interfaces* (Ward 2001), *Gestures in Human-Computer Communication* (Kurtenbach and Hulteen 1990), *The World in your Hand website* (http://www.daimi.au.dk/~ehlers/pda/input.htm), and *A Morphological Analysis of the Design Space of Input Devices* (Card, Mackinlay et al. 1991).

3.4.1 Mice

Mice, when used in a traditional form to control a cursor, offer users a way of interacting with software that allows analogies to real world objects to be made, such as clicking on a button to cause an action to happen. They range in size, style, and operation from traditional mice, trackballs, and touch pads. Each hardware device offers slightly different benefits, and minor variations can have drastic effects (Accot and Zhai 2000). Traditional mice offer fast performance and nominal accuracy (Card, Mackinlay et al. 1991), but are limited in certain applications by their size and movement characteristics, other forms mentioned generally offer greater portability or make tradeoffs for some other optimization. When used to control a cursor, on a traditional PC, mice offer few benefits to notification systems. They are a visual-manual input method and their use requires the full visual resources of the user devoted to the input method in use. Touch screens are often used for the same purpose as mice, but offer slightly more benefits from a dual-task management standpoint. They allow for direct manipulation of an interface that can be done through quick glances without requiring prior orientation.

3.4.2 Keyboards

The keyboard can refer to any number of devices that have a mapping which results in a function for entering text. The fastest and most logical keyboards have a one to one mapping from character to key, but are rarely used in their traditional QWERTY form for notification system input. A variety of different optimizations for size or speed have been developed, most of which have to do with the context in which the keyboard needs to be used. For a notification systems design, traditional QWERTY tactile keyboards can offer many benefits, if they can be used exclusively by touch and limit the attention resources consumed. However, they are rarely used for any input aside from the primary task. Other devices have attempted to capitalize on the tactile benefits of traditional keyboards, but in smaller and more portable sizes that are appropriate for input during a secondary task. Examples of these include the chorded (Figure 6), which operates using key combinations to generate a single letter and has the ability to be used with one hand. The Half-QWERTY (Figure 7) offers one-handed operation, but is much more similar in layout to a regular keyboard that switches between inputting the right handed or left handed characters. The fitaly (Figure 8) is one example of a soft keyboard optimized for text input on PDAs, but it doesn't offer tactile benefits only one hand operation and a very small size.

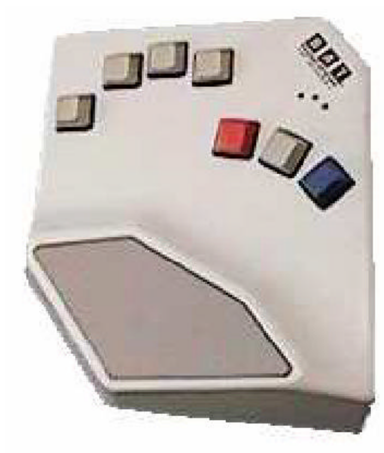


Figure 6. A chorded keyboard usable with only one hand (Ward 2001).

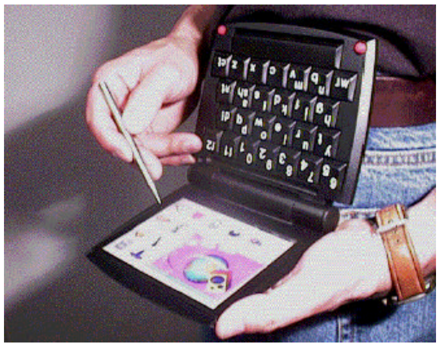


Figure 7. A Half-QWERTY keyboard usable with only one hand (Clausen and Pedersen 1998).

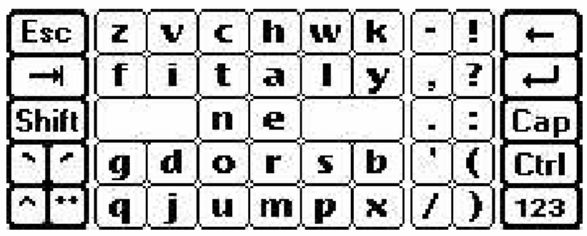


Figure 8. A Fitaly soft-keyboard (Ward 2001).

3.4.3 Gesturing

Gestures can define a wide variety of input methods, but all generally fit within the following definition by Hummels et al.

> *"A gesture is a movement of one's body that conveys meaning to oneself or to a partner in communication. That partner can be a human or a computer. Meaning is information that contributes to a specific goal. For gestural product design, describing the surface of an object or using the object are considered to be gestures, because these contribute to the creation of the final product...."*
>
> (Hummels, Smets et al. 1997)

Examples of gesturing input methods range from vision based devices that track the motion of hands or other physical movements (Hummels, Smets et al. 1997) to those that require touching a surface to receive input that maps to commands. Handwriting recognition using a pen based alphabet (Graffiti 2003). The benefits gained from using such methods vary. Vision based gestural systems allow freedom from being in contact with a hardware device which offers appeal in many situations for notification systems. These situations include initiating contact with the input device, which poses a significant distraction to the primary task and is typical of notification systems found in vehicles (Alpern and Minardo 2003), and with applications that run on large screen displays and offer information to a number of users at once (Zhao and Stasko 2000). However, even though vision based gestural interfaces open up a lot of possibilities, their success is largely limited by poor accuracy and the computationally intense algorithms needed for accurate recognition (Hummels, Smets et al. 1997).

Handwriting recognition using gestural interfaces can work well when size is a problem. Since they allow a varied number of inputs to be received on the same hardware device, they do not require a different hardware affordance for each input interaction they support and therefore can be significantly smaller then a standard 4”x12” keyboard. This allows them to work on platforms such as cellphones and PDAs which often are used in contexts where they support a secondary task. The gestural interfaces can follow a number of different alphabets that generally offer tradeoffs between efficiency of computer processing, such as Unistroke (Figure 9)(Goldberg and Richardson 1993) or ones that more closely resemble natural handwriting (iJot for Windows CE (Figure 10)). For a notification systems design either gestural alphabets may offer tradeoffs for secondary task use, Unistroke since it is faster and allows for less time away from the primary task, or iJot since it is closer to regular handwriting and may be less distracting from a cognitive standpoint. Both these types of gestural interfaces also may allow less distracting interfaces to be built as they have many of the features inherent in keyboards, including the ability for users to fixate on the primary task rather than on the input device during operation(MacKenzie and Zhang 1997).

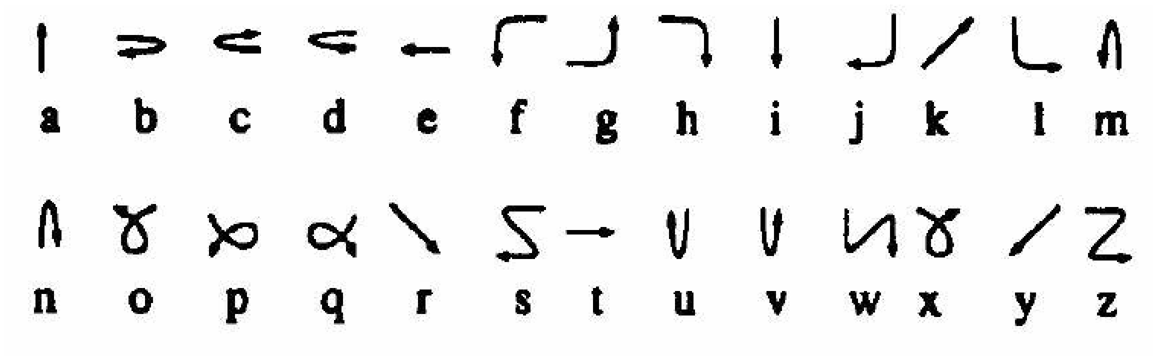


Figure 9. The Unistroke alphabet (Ward 2001).

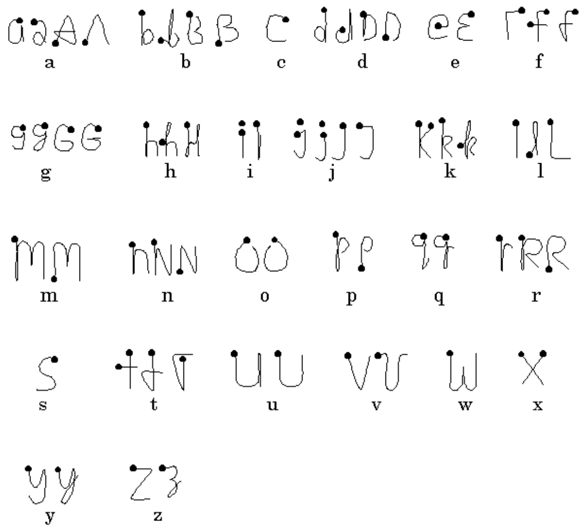


Figure 10 - The iJot alphabet (MacKenzie and Soukoreff 2002).

3.4.4 Speech Recognition

Speech recognition interfaces are thought by many to be the interface of the future. There strengths include the fact that they do not require tactile or visual interaction to use and are believed to interfere less with most typical primary tasks. Also, the interface is immediately usable and understandable for most novice users, often requiring only that users learn a command language similar to the user's spoken language. However, voice interfaces are not perfect. Accuracy rates are expected to be only around 90-95% for contemporary systems (Tsimhoni, Smith et al. 2002). Also, even though voice interfaces would seem to require less physical resources than visual-manual methods, they may require a similarly large amount of cognitive resources than other input methods. Studies have even linked voice based input methods to longer glances away from a primary task with the associated feedback for the secondary one (Tijerina, Parmer et al. 1998).

Voice input can be grouped into one of two different categories: dictation or command and control, each of which provides different services and associated benefits to the user. They are defined as follows by Microsoft:

> *"Dictation mode allows users to dictate memos, letters, and e-mail messages, as well as to enter data using a speech recognition dictation engine. The possibilities for what can be recognized are limited by the size of the recognizer's "grammar" or dictionary of words. Most recognizers that support dictation mode are speaker-dependent, meaning that accuracy varies on the basis of the user's speaking patterns and accent…In command and control mode, the grammar (or list of recognized words) can be limited to the list of available commands-a much more finite scope than that of continuous dictation grammars... This provides better accuracy and performance, and reduces the processing overhead required by the application. The limited grammar also enables speaker-independent processing, eliminating the need for speaker profiles or "training" of the recognizer." (Microsoft 2002)*

Both forms of voice input offer benefits specific to notification systems. They are an enabling technology that can allow smaller devices to be made without the need for hardware to allow for easy visual-manual input of complicated functions specifically sized on a wide variety of truly ubiquitous computing devices. Dictation interfaces are particularly useful in devices such as PDAs that are used to facilitate the recording of an infinitely wide number of possible events; however, they are much more limited by the technology that has yet to come of age (Grasso 1996). Command and control interfaces are currently much more feasible, and have become commercially available in many cell phone products as well as in in-vehicle information systems, but have yet to obtain wide ranging success. This is possibly due to a lack of social acceptance and well as the need to remember a language syntax that offers no more cognitive affordances than keyboard based command languages (shown to be disliked greatly by users through the success of GUI based interfaces).

3.4.5 Simple Tactile Devices

This last category of input methods applies to a large number of real world devices that map actions directly to a single application function and offer some form of tactile feedback to indicate an action has been performed. Keyboards can fall into this category if they are used to perform an action rather then input text, such as pressing an arrow key to move the cursor up or down on the screen. More typical tactile devices for notification systems, however, include things such as remote controls, which allow physical distance from a notification system and provide greater flexibility for primary tasks (Enns and MacKenzie 1998) or specialized joysticks that allow for one handed resource constrained input into systems (Kawachiya and Ishikawa 1997). Simple knobs or buttons located in places within grasp also provide special affordances to users in particular situations prone to distraction (Manes and Green 1997).

For a notification systems design, a simple tactile interface with a one to one mapping from device to application function offers a clear non-distracting method of input. Although they are potentially visual-manual methods, if the action the button performs does not require visual feedback, users can learn the position of and then use the buttons without visual requirements. This has been exemplified within the domain of in-vehicle information systems, in which radios are considered the base line method to compare to. Tactile buttons, however, create significantly more distraction issues when their function requires feedback, such as arrow keys that move a dynamic list box selection up or down.

3.5 Summary

The input methods just described demonstrated the need for interaction with notification systems and, more importantly, demonstrated many of the pitfalls with input methods for notification systems. The next chapter will expand upon these input methods by describing the context in which interaction with a notification system is typically needed.

Chapter 4: Related Work

This chapter will consider how input methods have been made to cope with the issues presented in certain domains and platforms. This includes consideration of the specific problems solved, or attempted to be solved by the device, as well as the performance and more on point, the design process and design analysis used to arrive at the performance metrics and their applicability to notification systems theory. Suggestions will be made on how well the notification system critical parameters (interruption, reaction, and comprehension) and the attention-utility theme were addressed, highlighting aspects relevant to user goals that may have been overlooked.

4.1 Using Input Methods with Notification Systems: Platforms and Domains of Interest

Three types of systems are particularly interesting to notification systems from an input perspective (Figure 11). Although notification system design considerations are certainly applicable to a much wider range of challenges, most other systems can be characterized according to one of the following three paradigms or have only a trivial input need. The three categorizations of systems all have one particular aspect that seems to permeate through decisions behind their design. In-Vehicle Information Systems (IVIS) are with the goal of being safe above all else, good mobile computing design minimizes size, and ubiquitous computing proposes to be as invisible as possible to the user. Although the emphasis on each of these aspects distinguishes these types of systems from one another, they still have many things in common. They are all still notification systems that must balance the attention of the primary task with the utility from the secondary task, they all share a significant need for input that dictates their information delivery capabilities, and the design rational behind each of the systems overlap. For example, in-vehicle information systems can and often are described as mobile or ubiquitous computers for their small size and attempt to merge the secondary functionality into the vehicle environment as a single undistinguishable presence.

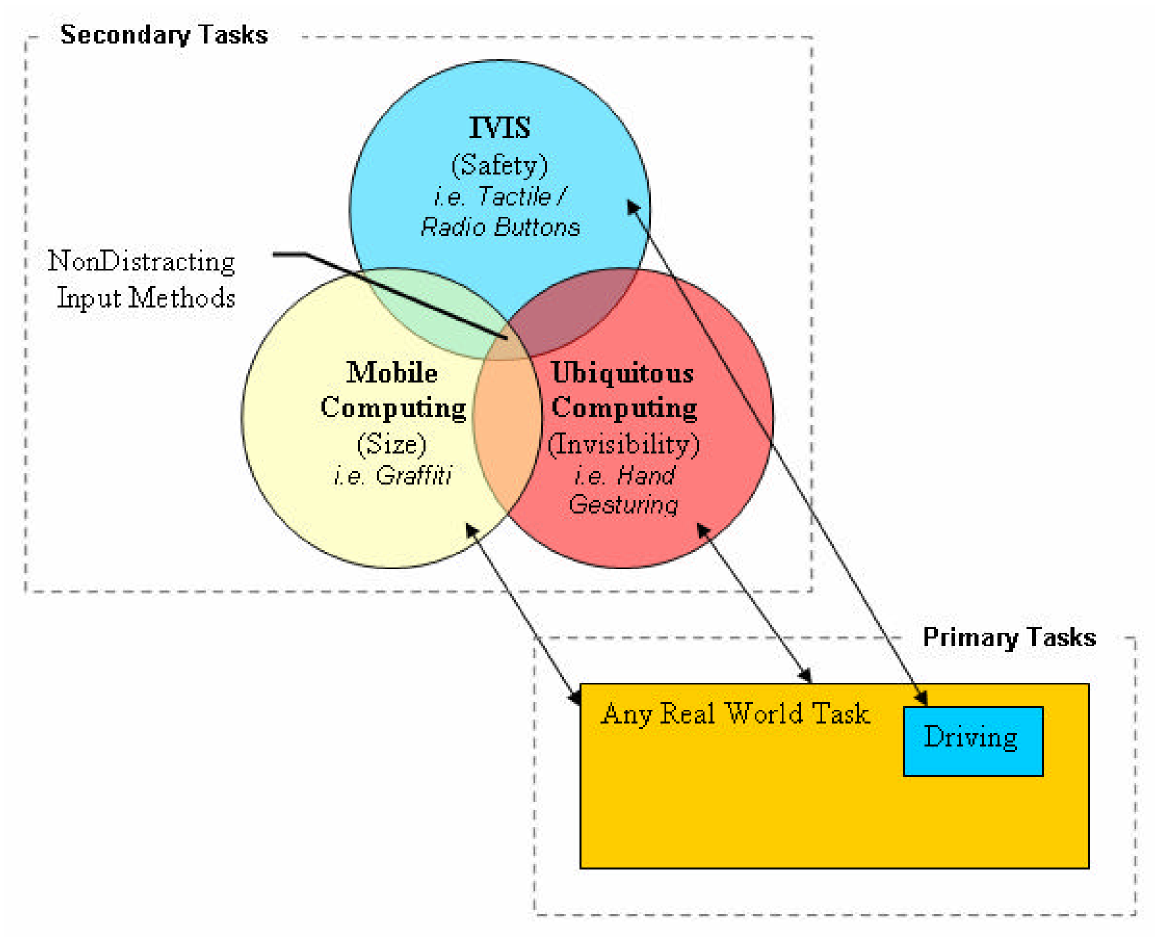


Figure 11. Diagram of the notification system domains with a significant input need. The major priority of each type is given followed by a typical example input method. The arrows represent the interaction between primary and secondary tasks. All types share the goal to be non-distracting, mobile computing emphasizes size first, ubiquitous computing emphasizes invisibility first, and IVIS emphasizes safety first.

These three types of systems will now be described in detail with a research focus. Particular attention should be paid to the description of the IVIS category, as this thesis will continue with in-vehicle theme to demonstrate the role of input methods. The other two categories are being covered for breadth, their direct application to IVIS, and their emphasis of a particular design challenge for notification system input methods.

4.1.1 In Vehicle Information Systems

The past few decades have seen more and more technology creeping into vehicles in an effort to make the most of a driver's time on the road. The most recent advances in the IVIS domain have raised more then just a few issues, prompting the need for legislation and new definitions of what is safe with these new devices. GPS and telematics (the merging of information and communication technologies) are two popular applications being developed for in-vehicle use that offer more, potentially distracting, features to drivers then ever before. The enormous magnitude of the domain and the critical nature that safety plays within it has resulted in a significant amount of research within the field; however, few hard and fast metrics have emerged for conclusively determining what is a safe and usable system (Anderson, Abdalla et al. 2001).

4.1.1.1 Notification Systems in Vehicles

Notification systems and IVIS are a perfect match. Any system that is to operate within a vehicle and be at the driver's disposal must be a very good notification system, the primary task of which should always be driving. The secondary task ranges from applications of GPS, cellular phones, or any number of other systems that offer some benefit for use in a vehicle, either related or not to the primary task of driving. More so than many other domains, the tradeoffs of attention for utility brought on by the secondary task are of the utmost importance to safety of the system. Unlike desktop systems in which usability issues simply cause an irritation, improper use of the driver's resources can results in harmful or fatal distraction. Almost all of existing research hints on this point in some way or another (Collins 1997; Tijerina, Parmer et al. 1998; Anderson, Abdalla et al. 2001; Tsimhoni, Smith et al. 2002). They have suggested a number of guidelines to help developers limit the amount of time spent interacting with secondary tasks, reduce the types of interactions with secondary tasks, particularly those of a highly visual or cognitive nature, and keep notifications from causing excessive interruption (Lee, Kantowitz et al. 1994).

The 15 second rule is one of the most well known and often criticized (Green 1999) measures of the attention demand of an IVIS. This often criticized method for its plain simplicity merely states that if a task can be *performed in a static vehicle environment by a typical user in less than 15 seconds, it can be declared safe.* This directly relates the safety of a system to only a measure of time. Although there is an obvious correlation between how much time a task takes away from the primary task of driving and the safety of a vehicle is it easily observable that attention demand is really a function of more than just time. Specifically, the cognitive demands of a system can affect a user's ability to perform the primary task even more so then the amount of time it takes. This becomes evident when considering that in using a notification system in a vehicle, the primary task is never completely ignored, rather it becomes a task requires a lesser portion of the user's attention resources. The amount of this portion is based directly on how demanding the secondary task is, and can be inversely proportional to the amount of time the task takes. Other more complicated metrics handle the task of rating distraction better, but often fail to consider the impact on the utility of the secondary task that minimal attention may have.

Other examples that illustrate the role of input on notification systems can be found in the IVIS platform. Many new applications have driven development of applications for use while driving, in particular, GPS systems have become a significant analysis of IVIS has been carried out particularly with respect to GPS. The University of

Michigan has a devoted research facility towards studying the effects of IVIS (UMTRI 2003) with portions of research devoted exclusively to the role that input methods play on safety. An evaluation of various input methods was undertaken that showed a direct relation to a the input method used and the number of lane excursions (a measure of driver distraction) that occurred (Manes and Green 1997).

Many other approaches to rating the safety of a vehicle have been proposed and used over the years. Most similar are attempts to rate the safety of a system by monitoring users of a prototype and responses that indicate the amount of attention focused on the road, and thereby the user's ability to react to the diverse road conditions presented in a typical driving environment. Such measures generally involve eye glance behavior and lane exceedances, among other very complex measures to accurately rate the user's attention level. The methods have been shown to work generally better then the less complex 15 second rule, as they more thoroughly account for the numerous variables affecting attention (Anderson, Abdalla et al. 2001). Various research has shown that accidents are caused by more than just a driver's lack of attention, including factors such as mechanical failures, weather and road conditions, the ability of a other drivers on the road, and ever other factors that can be directly attributed to the presence of IVIS. However, IVIS research has shown that attention to the road and the primary task of driving is the most relevant factor in explaining performance while driving.

Notification system research can be applied directly in vehicles. Interruption can often be a desirable user goal to inform a user of a missed turn with GPS Systems to notify or to notify of a critical safety issue with sensing systems. Although disruptive to the primary task of driving, these notifications will actually result in better safety by preventing potential accidents. Quick reaction is needed to adjust driving to a critical notification, but many applications often call for a more gradual reaction to respond to information of a less critical nature, such as information about new traffic patterns or new e-mail messages. Finally, with the wealth and complexity of information that many IVIS attempt to deliver, long term comprehension of routes or perhaps location based information delivered via telematics sources is often a desirable user goal.

4.1.2 Mobile Input Methods

Since the release of the popular Palm Pilot over 10 years ago, more and more applications have been moving to devices that users can carry and exploit while on the go. However, as more people use their PDAs for increasingly complex tasks, the problem of interacting becomes more difficult. Text input, which is almost a trivial task taken for granted on a regular computer workstation requires the development of new and creative input methods for performing certain tasks when small size is a major consideration.

Research within the field, fueled by strong industry demand, is easy to come by (Clausen and Pedersen 1998; Ward 2001; MacKenzie and Soukoreff 2002). However, no one input method has yet to answer the input need for mobile platforms similar to that of a standard QWERTY keyboard on a desktop computer. With such a wide range of devices and methods, there is an essential need for a design and evaluation process that allow tradeoffs to be measured in standard terms that designers can utilize to best meet the user goals for a product.

MacKenzie and Soukoreff have beseeched designers of text input methods for mobile computing to use evaluation methods that are both reproducible and generalizable so that experiments have implications outside of the narrow scope of the controlled environment in which they take place. They also hint at the strong role that notification systems play in devices of a mobile nature noting that in real life, people rarely focus on performing a single behavior. This presents a major challenge in designing accurate evaluations, as an experiment that tries to mimic this dual-task role for mobile devices must include actions not required by the input method. MacKenzie and Soukoreff also indicate the role of distraction for these input methods through a term they refer to as *focus of attention* (FOA). FOA refers to where the user must place his or her attention during an input task. For example, using a keyboard as a touch typist is just a one FOA task, since it only requires looking at the input source. However, if a "hunt and peck" method is used, such as that required for soft keyboards, it becomes a two FOA task. If the user makes a lot of mistakes while hunting and pecking, thus requiring verification of the entry on the screen, it now becomes a three FOA task (MacKenzie and Soukoreff 2002). The role that higher FOA plays is then directly related to the amount of attention required to use the input method; however, no tradeoffs are mentioned for using devices with different FOA, particularly as related to the types of secondary tasks. They do suggest paths that designers should take to develop non-distracting and usable devices, including analysis of the number of hands used and the focus of attention the design creates.

Size reduction plays the largest part in the motivation behind many mobile input methods. The increased power of mobile computers allows designers to cram more and more features into the same space; however, the available screen real estate to control them does not increase. This reduction has many negative implications for the notification system critical parameters, due to a number of factors. In general, distraction is increased while reaction time and comprehension is decreased (MacKenzie and Soukoreff 2002). To get around this problem and meet user goals, a number of different solutions have been proposed. Brewster and Cryer demonstrated that by adding sounds to correspond to input events (such as button presses and mouse overs) reaction and comprehension is improved (Brewster and Cryer 1999). Other research has gone one step further and explored the ways in which completely auditory user interfaces could be acceptably designed. These required tradeoffs made it clear that audible interactions are significantly different from visual ones (Stifelman, Arons et al. 1993; Hindus, Arrons et al. 1995); however, they still can be discussed according to the notification systems critical parameters. For example, more complex sounds (speech) were shown to increase comprehension over simpler ones (beeps), but at the same time the level of distraction increased due to the amount cognitive work that understanding speech requires (Shneiderman 2002). Input methods that work with these non-visual interfaces also have to be designed with special consideration, since a lack of visual feedback limits the types of methods that can be used.

4.1.2.1 Cellular Phones

Cellular phones are becoming more enriched by computing functionality making them similar to PDAs in many respects. However, they are even more beleaguered by smaller size requirements as users generally expect a phone to look and act like a regular phone regardless of its capabilities. Typical cell phones usually have screens significantly smaller than PDAs and are accompanied by only the 12 digits used for dialing numbers. This has not stopped many services from being offered that rely heavily on text input from users.   Short messaging service (SMS) is a feature for sending text messages to other phone users that is found in many new mobile phones and has shown incredible growth over the past few years. It offers an unobtrusive, inexpensive, and asynchronous way of communicating.

Existing interfaces that allow text input on cell phones generally rely only on the 12 key number pad that maps the numbers to letters, and while a number of similar methods exist to perform this mapping, the most prevalent is Multitap. This method requires one, two, three, or in some cases even 4 key presses to be made to enter a single character. Many methods have improved upon this simple concept through using language models that guess the correct word based on a subset of characters in the word, or through different methods altogether using handwriting recognition or voice interfaces (Butts and CockBurn; Pavlovych and Stuerzlinger; Silfverberg, MacKenzie et al. 2000). Most evaluations made on these different methods generally focus solely on an objective measure of WPM and subjective questions as a way to find the best interfaces. Although this evaluation technique may sometimes reveal surprising results that differ more than just in magnitude from expert analysis, they can be far from real world results, particularly when the differences between the test systems are minimal. A major problem with lab based studies is just that—they are lab based studies and can not hope to cover all the variables that would be encountered in the real world. Diversions present in real-world situations can adversely affect the speed of input, but may affect it in different ways from device to device. For example, input methods that use dictionaries to auto-complete words may be faster when the user is able to look at the screen, but if a visual distraction is present they will perform significantly worse than those that do not require looking at the screen (MacKenzie and Soukoreff 2002). Also, using merely speed does not tell anything about how or why a particular input method is faster than another one, making it impossible to compare new input methods without running a new test that shows some advantage as a tradeoff for raw speed. The IRC framework attempts to correct these problems by indicating a way to frame evaluations that will mimic real world situations closer, allow standardization between studies, and indicate the specific areas that one input method differs from another.

Issues related to size are not limited to mobile platforms hardware constraints often play a role in the IVIS domain. Cars can hold a lot more components, but they often need to be located amid a variety of already existing devices that support other features needed while driving. There are obviously solutions that will not work across the platforms, but most offer tradeoffs of function for size that meets similar user goals for both types of systems.

4.1.3 Ubiquitous Computing

Ubiquitous computing aims to improve the usage experience by making the machine invisible and making interaction more natural. This, however, is easier said then done. Interacting with a device, without actually knowing you are interacting with a device, presents problems for a full spectrum of computer science specialization from artificial intelligence to information visualization. Input methods also provide a whole range of challenges for ubiquitous computer developers. The input methods most applicable to the platform generally are very exotic compared to those found on typical desktop computers, i.e. voice and gesturing. Also, like mobile computing, methods for text input speeds cannot match that found on typical desktop computers, but fewer applications need such interactions. Input is usually needed for methods that allow communication with something in the real world. Ubiquitous computers also cannot as easily be tested in lab based studies, as they are comple tely dependant on interacting within a real world scenario even more so than other notification systems. Typical studies in the field generally involve the construction of "smart" homes or work environments that can be used for controlled field studies (K idd, Orr et al. 2000). However, these are very expensive to conduct and other evaluations take advantage of more typical field studies often employing a wizard of Oz technique or mobile data gathering (Mäkelä, Salonen et al. 2001). Regardless of the different contexts that ubiquitous computers are tested in, they are still clearly notification systems in which the primary task is the whole of the real world.

Scholtz et al. describe ubiquitous computing as capable of being characterized for evaluation purposes by 4 different distinct properties: *universality*, the applicability to different domains, *utility*, the expected benefits, *usability*, effort required to achieve a goal, and *ubiquity*, where and when (Scholtz, Burnett et al. 2002). These four qualities can be adequately described by the notification systems framework. Since the framework generalizes to non domain specific terminology, it is already universal. Utility and usability are both described by how close the designed system met the desired user goals for interruption, reaction, and comprehension. Ubiquity, is a largely domain specific property; however, the notification system critical parameters will still reveal how well this is achieved.

Technological limitations have thus far restricted the field of ubiquitous computers to only performing very simple tasks. Many other fields, such as wearable computing, are defined by many of the same principals of ubiquitous computing and attempt to be as unobtrusive as possible. Twiddler (Figure 12), one of the more practical and popular input methods for wearable computers only requires one hand to operate the device, although it must be correctly oriented, to use it (HandyKey 2003). Other glove type devices may keep both hands free, but may still hinder performance (VirtualTechnologies 2003). For most practical purposes, these devices get the job done well; however, they lack the finesse and usability that wearable computers should have to make it seem as though the real world is digitally enabled—beyond just giving users the ability to bring desktop workstations with them to more places. Rekimoto has suggested that the key to solving this problem lies in making devices that support hands-free operations or allow quick changes between normal and operation modes (Rekimoto 2001), a clear reference to interruption. To this end, he has proposed two distinct input methods that support this goal: GestureWrist and GesturePad (Figure 13); however, no usability tests were run to judge their performance. Like more traditional ubiquitous computers, the context or primary task in which the system used play a very important role. The true power of wearable computing comes from its ability to be used in dynamic environments and, as such, any design analysis should take this into account.

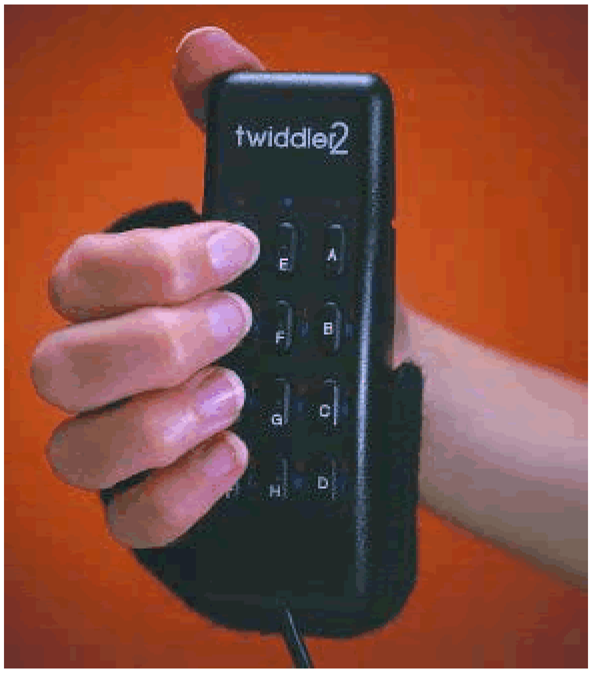


Figure 12 - Twiddler; a keyboard-like device used for wearable computers (HandyKey, 2003).

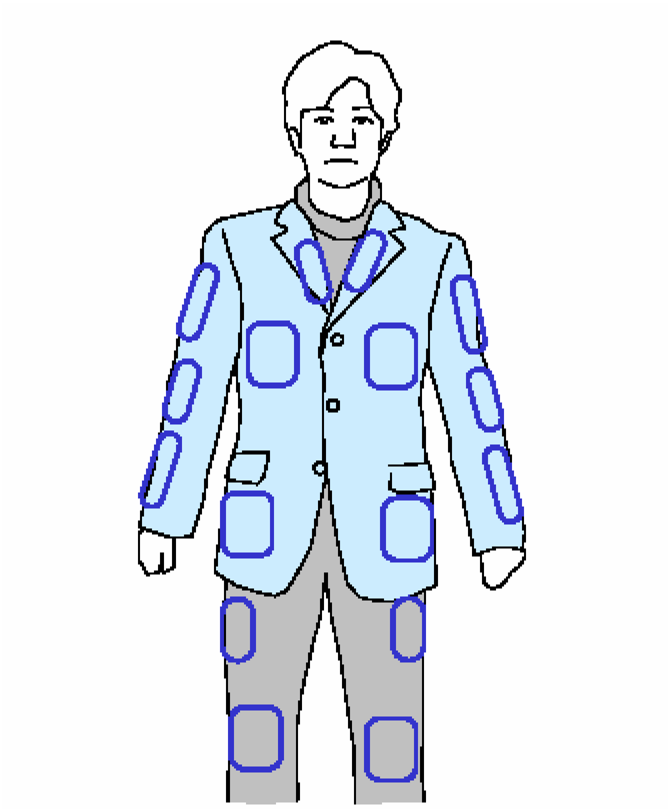


Figure 13. GesturePad—a wearable input method. The dark blue outlines show areas that are sensitive to touch (Rekimoto 2001).

Chapter 5: Carputer: A Narrative of a Design Focused on Input Methods

This chapter will present the design of a notification system for an in-vehicle information system with a vital need for efficient and non-distracting input. The chapter is meant to be a practical illustration of the role that input methods play with the success or failure of a notification system, as well as to establish a practical basis for understanding the relationship between the primary task, the secondary task, and the input method that must fit in with both.

The next few sections will describe a project to install a computer inside a car. The primary function for the computer was to allow for selecting and playing MP3 music files from a large database with features not possible on current stereo systems, mostly due to the lack of refined interaction methods. The design process taken was not conducted in a controlled environment or under the guise of a scientific experiment, rather it was a practical attempt to develop a working application. Over a 2 year design cycle in which various implementations of the system were built, iteratively tested, then redesigned, the system highlighted the issues inherent in the IVIS domain and encountered the roles that all the IRC critical parameters have on input methods. Although visual output posed a few problems, given the simplistic nature of the desired output data, song names etc., these solved fairly typically through limiting the amount of information viewable to the user and using large, easily readable fonts. The design may still pose some issues from an information delivery standpoint, but it is beyond the scope of this initial discussion and proposed as future work.

5.1 The Need for a Carputer

A carputer is simply defined as a single electronic device or composition of devices designed to host any number of different, potentially unrelated, applications in a vehicle. The need for a carputer, in this case, started about 2 years ago with a situation that has become more commonplace with the rise of vast, easily portable music databases popularized by the MP3 file format (allows for a virtually unnoticeable 10:1 compression rate of audio). The average laptop hard drive can hold thousands of songs, more than enough storage for most music collections, posing the problem of how to take advantage of such a collection in a vehicular environment. Although the portability of a laptop makes it possible to use for playing MP3s in a car, this is clearly by no means safe and even more detrimental, since it takes more effort to use the system than switching CDs and tuners. Typical applications for playing MP3 files are optimized only for playing and navigating songs on a typical desktop computer. They are built around the paradigm of mouse based computing, an input method that requires both visual and manual resources from the user, which are also resources that are sorely needed for the primary task of driving. These challenges, however, do not negate the user goal of listening to music with the same freedom that is enjoyed on other less restrictive platforms. Industry has tried to answer this need through CD players and changers that are capable of playing MP3 CDs, which allow for around 130 songs to fit on a single CD. This solves the problem of allowing for vast, portable music collections; however, navigating such a CD to find a specific song is quite difficult.

Stereo systems capable of playing MP3 CDs typically only allow for navigation through tactile buttons located on the console of the device. This works fine when only a few songs are available, or when it does not matter which song is played next. However, when a user desires a specific song, these devices can be very distracting. Typical input methods generally allow for file system navigation, in which a folder must first be selected by using arrow buttons to scroll up or down thorough an alphabetized list that displays only one highlighted item at a time on the stereo's small display. When the proper folder is highlighted, it can then be selected to show the contents. The scroll and select process is repeated until a music file is selected. Given a song database capable of holding over 130 songs, this method leaves a lot to be desired, particularly when the exact ordering of the songs are not known and with the small text size that these systems provide.

Few other input techniques have emerged in commercial MP3 based products, most likely for safety concerns. Speech recognition has had some limited success, but not wide acceptance. One of the most innovative products to use voice control is the CD-VC60 Voice commander by Pioneer electronics (PioneerElectronics 2003). It offers user programmable voice control to select common CD player options or play songs when the user names a CD and track that has been previously programmed in. Although no usability tests have been conducted on the exact system, it has significant obvious drawbacks that limit its potential, such as cost, a limited vocabulary (100 commands), speaker dependant control, as well as reliance on clear audio in a dynamic and noisy environment. However, performance for many tasks should still be better than with the tactile standards it replaces.

Noting the lack of good solutions for providing the input necessary for navigating a large MP3 collection, the solution to using a laptop in a car implies the need for a significant amount of interface design and software implementation. A few constraints were made by the hardware available on the laptop, but for the most part, customization was readily applied given the maturity and standards that laptop technology has come to enjoy. The system itself consists of a Dell Inspiron 3200 laptop with 12.1" screen, Intel Pentium 233 mhz processor, and 96 MB of RAM. Supporting hardware (Figure 14) for various input methods and other system functions, collectively referred to as *carputer*, has grown to include an infrared remote control, a webcam with microphone, a 15GB USB harddrive, a trackpad mounted in the center console where it is easily accessible by the driver (Figure 15) and a direct connection to the car's power supply and speaker system (Figure 16). The laptop screen was first reversed and placed back on the laptop base (Figure 17), It was then mounted in the passenger side of the car where the glove box once resided (Figure 18). This does not provide a complete heads up display, but allows for quick glances to be easily made without much movement of the eyes.

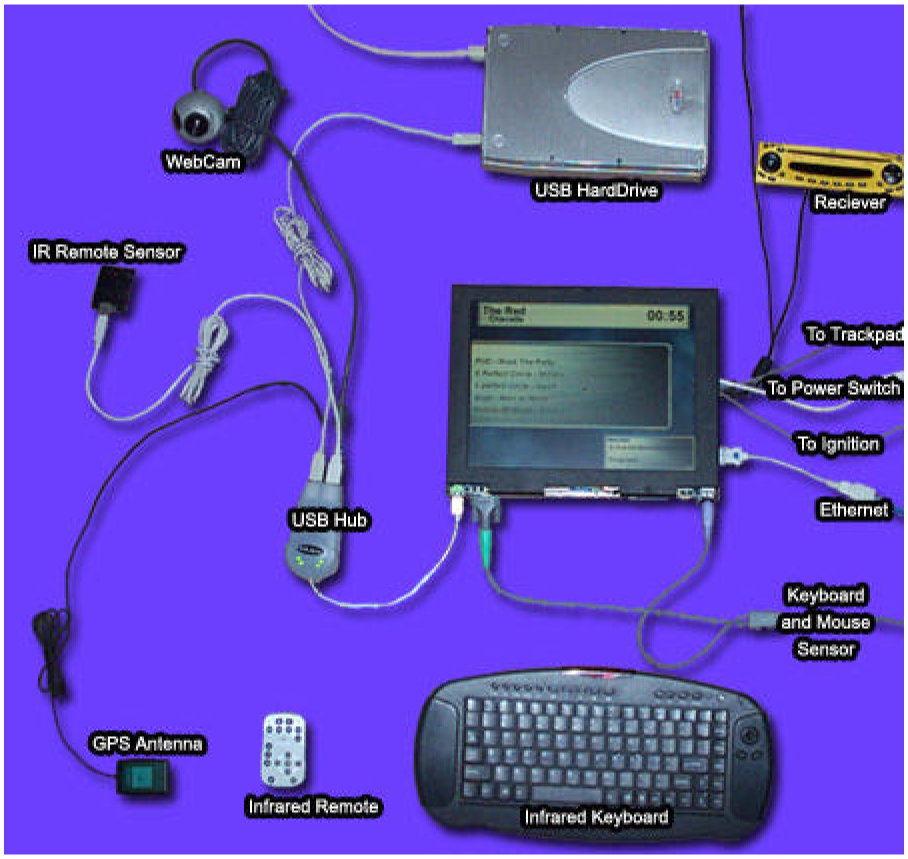


Figure 14. Carputer Hardware - All hardware shown is used to make the carputer function, the webcam is not yet implemented, but will allow for visual gesture control and speech recognition through its integrated microphone.

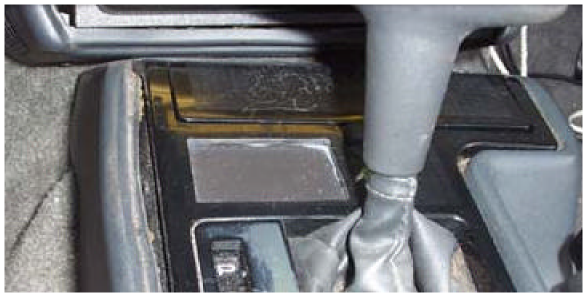


Figure 15. Image of the touchpad used in the carputer. Initially it functioned as a regular mouse, however, this was found to be unusable in a vehicle and was reimplemented as a graffiti character recognizer.

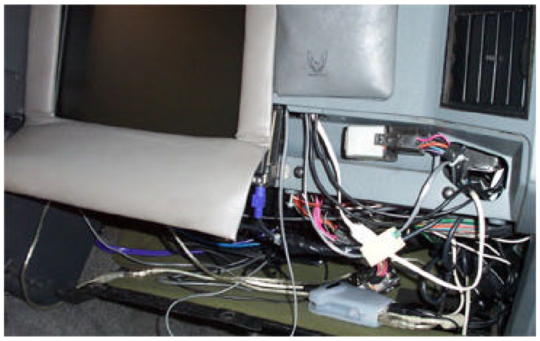


Figure 16. The carputer requires over 10 different wires to be connected to work with the vehicle, including a direct connection to the cars power supply and speaker system.

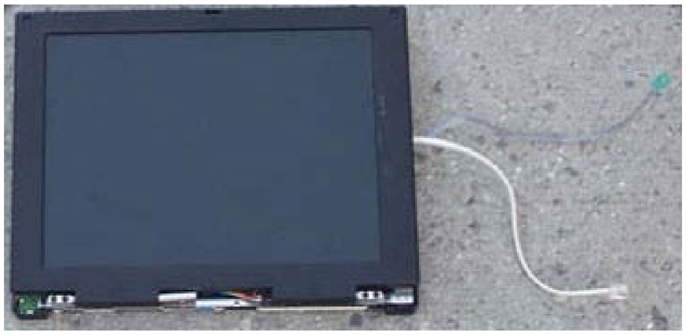


Figure 17. The carputer screen was reversed then placed back on the computer base.

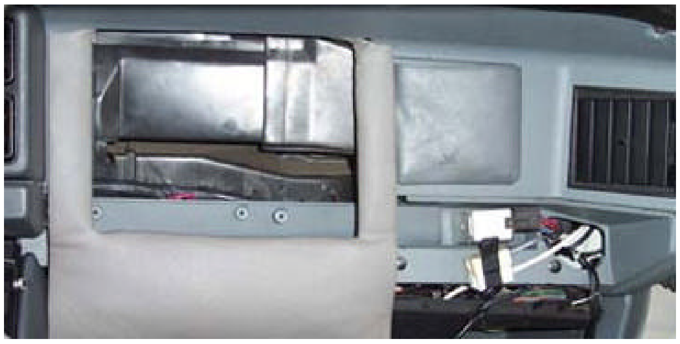


Figure 18. The carputer was installed on the passenger side of the car where the glove compartment once resided.

5.2 The Initial Interface

The software for the carputer system was built on Microsoft Windows 98; however, the operating system (OS) was greatly stripped down to only include the hardware abstraction layer and only those features necessary for the basic function of the

OS as a platform to run the software that would be useful in a car. The Windows Explorer shell was removed so that the Start Menu typically associated with the Windows platform was now no longer even run and was replaced with a simple program that started the MP3 player and interface. The backend for the MP3 player was powered by WinAmp (Nullsoft 2003), a popular MP3 player for desktop computers, but the interface was first augmented and eventually completely replaced with a program that organized all the MP3 files on the hard drive and controlled all the operations allowable on the system, to include selecting new files to play, from a randomly constructed play list or based on input from the user.

5.2.1 Mouse - Cursor Interface

The first version of the MP3 player interface was based exclusively on more traditional mouse / cursor input (Figure 19) through the installed touchpad. The interface for WinAmp (Figure 20) that was designed for desktop computers was still in place and used for the more complicated functions of selecting items from a play list. This included scrolling through a list box and selecting the desired song. Given the relatively small font and clickable areas of the list interface, this, obviously, was not even close to usable while driving. The interface was, however, designed with four large software buttons located at the bottom right of the screen that mapped directly to the most common functions of "play", "stop", "next song", and "previous song" (Figure 21). With some practice and luck, it was possible to click the appropriate button with the track pad to control the system. The key to using the interface, though, required leaving the cursor positioned over the next song button and just tapping the touch pad to perform a click and activate it. This allowed for a new random song to be selected and played without actually looking at the screen, just by tapping the touchpad (Figure 15) that was located at arms length from the driver. This feature was put to much better use in later revisions through a graffiti (Graffiti 2003) character recognizer.

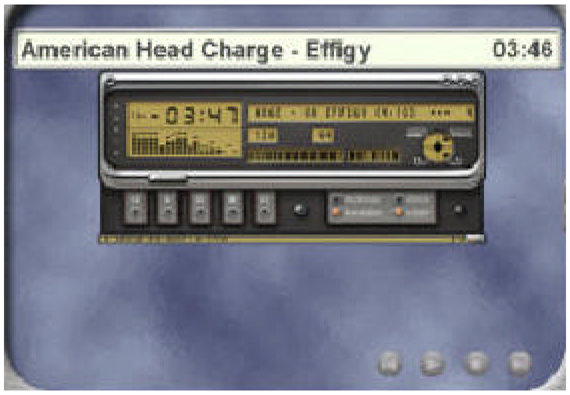
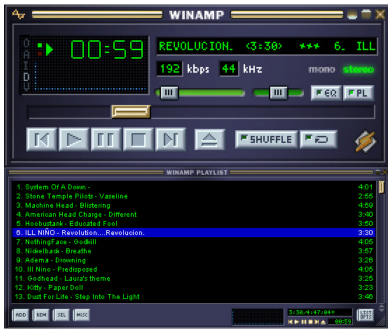


Figure 19. Initial interface for the carputer - winamp running in the center of the screen, the window at the very top shows the current playing song and time, the four buttons at the bottom control winamp. This interface offered litter in terms of usability in a vehicle.



```
1. System Of A Down -                          4:01
2. Stone Temple Pilots - Vaseline              2:55
3. Machine Head - Blistering                    4:59
4. American Head Charge - Different            3:40
5. Hoobustank - Educated Fool                  3:50
6. ILL NIÑO - Revolution....Revolucion.        3:30
7. NothingFace - Godkill                       4:05
8. Nickelback - Breathe                        3:57
9. Adema - Drowning                            3:26
10. Ill Nino - Predisposed                     4:05
11. Godhead - Laura's theme                    3:25
12. Kitty - Paper Doll                         3:23
13. Dust For Life - Step Into The Light        3:46
```

Figure 20. The WinAmp Interface. The bottom portion of the window lists the songs in the playlist for selection.

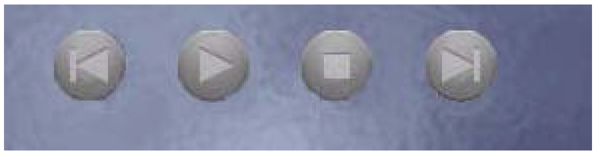


Figure 21. Cursor operated button interface for the initial version of carputer.

After using the system for a short period of time, many problems with it were noted and taken into consideration for a redesign. The most common operation done on the system was found to be using the next song button to select a new song to be played; however, even though this should have been a trivial task, it was complicated by the current system design. The problem has already been solved many times before through the standard stereo systems found in cars. To change the current song playing while in a vehicle, the simplest answer which has proven to be relatively safe and useful through decades of operation in typical stereo systems is just to have a hardware button mapped directly to the function. Although, mimicked by the software interface, this proved to be almost unusable, since it required looking at the screen to select it. Unlike, the tactile buttons found on stereo systems, it did not offer the ability to be located simply by touch.

5.3 The First Redesign

The first version of the carputer made one fact very clear: typical desktop interfaces do not work well in vehicles. In the next revision of the interface(Figure 22), hardware buttons were implemented in the carputer system through the inclusion of a remote control that was connected to the laptop's USB port. The remote device (Figure 23) itself featured a number of buttons many of which had multimedia symbols for common functions like "play", "stop", "pause", "fast forward", "next track", "volume up / down", etc. These were logically mapped to the same functions for the MP3 player, i.e. next track would select the next song in the play list and begin playing it automatically.

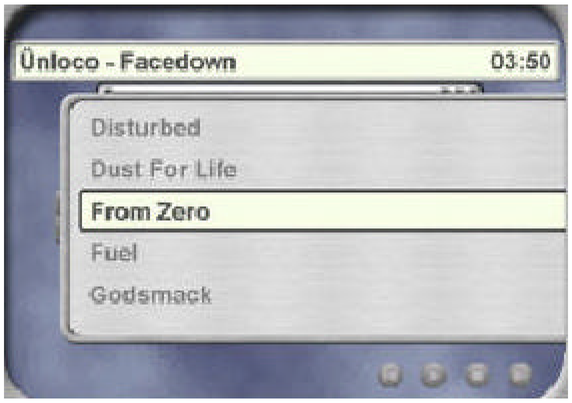


Figure 22. First revised interface for the carputer - Input from a remote control was now supported in addition to mouse control. The alphabetized list of songs shown in the center of the screen allows songs to be scrolled and selected with the remote.

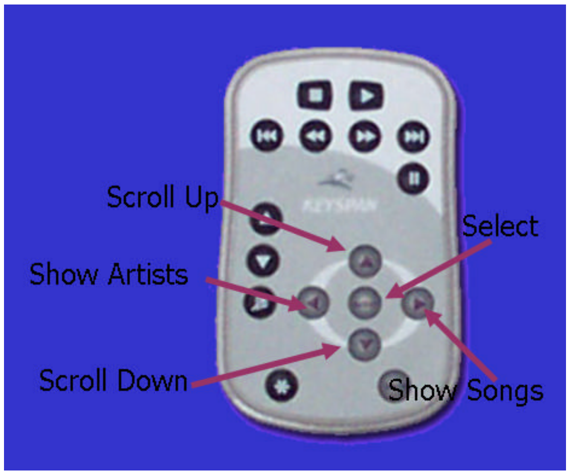


Figure 23. Remote control used to control the carputer interface.

5.3.1 List Selection with a Remote Control

The remote also featured arrow buttons and a select button that was used to implement a visual-manual scroll and select input method that allowed the user to choose a new song to play from the database as follows: Pressing the left arrow button shows an initial alphabetized list of all artists in the system. One of these artists must then be selected by pressing the buttons mapped to “up” or “down” to move the list and change the highlighted item. Then when the desired artist is highlighted, it can be selected by pressing the “select” button which constructs and displays a list of songs performed by that artist. A song can then be selected from this list in the same manner in which an artist is selected. When a new song was chosen by the user, in this manner it was enqueued at the top of the playlist, but after all previously enqueued songs. Each song would be played from the playlist as time ran out on the song currently being played. The queue was used as a safety factor so a user of the system could select a number of songs to be played when stopped at a light or times when distraction was not as much of an issue as it was when moving at high speeds. The software, however, did not monitor speed or disable the selection system when the car began to move. The top 5 items to be played next in the playlist were shown in a large font in an always visible list (Figure 24). Queued items are highlighted in blue. This input method, although similar to existing methods on radios, was thought to be much more usable due to the size and readability of the list items.

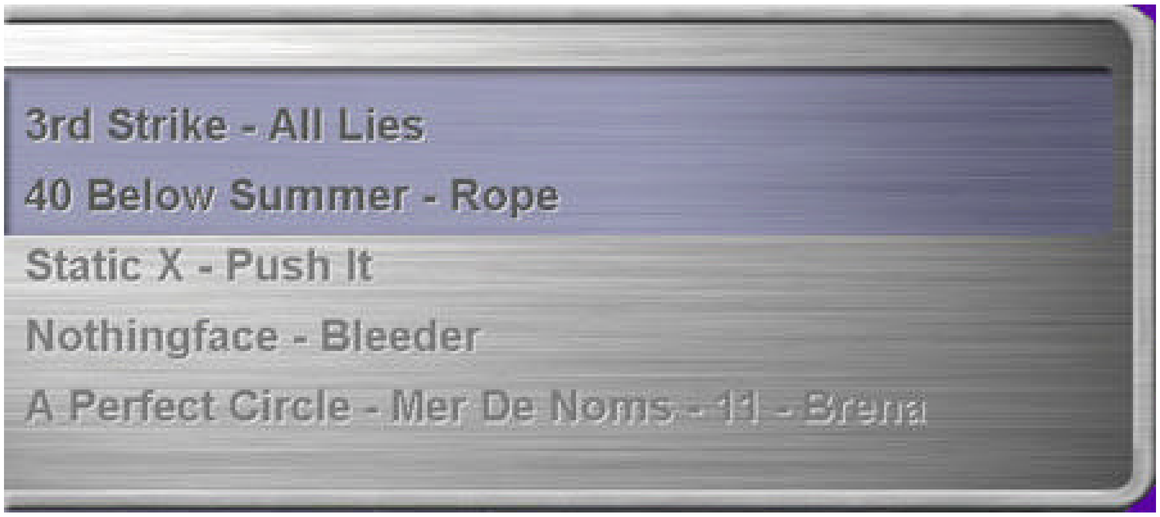

3rd Strike - All Lies
40 Below Summer - Rope
Static X - Push It
Nothingface - Bleeder
A Perfect Circle - Mer De Noms - 11 - Brena

Figure 24. The carputer playlist - the blue highlighted songs are ones recently selected by the user, each additional song selected is enqueued at the end of these.

All the functionality that WinAmp provided from an interface standpoint was now accessible from the new system and input methods. This allowed the backend to be completely hidden and only the tactile input methods based on the remote control were made available. This system was much more complete from an in-vehicle standpoint and was significantly more usable; however, the remote control list selection method is still a visual-manual input method that requires the driver to glance at the screen and take a hand off the wheel to use. Based solely on expert evaluatio n, it is, however, must less interruptive than mouse based control of the system and provides quicker reaction time for selecting a new song when an undesirable one is played. Also, the use of the large, easily readable lists enables greater comprehension of the songs within the database and should facilitate quicker selection times as the user becomes more familiar with the layout of the contents. These improvements make it much more possible to use the system while driving, even when the car is in motion, but safety still might pose a significant issue even for expert users. The amount of time required to select a song is dependant on the starting location of the previously selected song, but with a large database (50+ artists), it is quite possible to exceed the guidelines proposed by the 15-second rule (Green 1999). The next section will talk about further revisions to the system that attempted to solve this problem through implementation of alternate input methods.

5.4 The Addition of a Character Recognizer

At this point, the carputer was fully functioning and was at the point of a potentially safe and usable system that was at least as good as other commercially available products. It was not without its flaws though, and its features were limited to only playing MP3s. Two design directions for the system were now possible, either continuing to work on the MP3 system until it offered no more significant problems or to port more applications for in-vehicle use. The former approach was taken, as although satisfactory, the input methods still indicated there was much need for improvement and more sophisticated applications, such as GPS or telematics, would require even more interaction and have different tradeoffs of use. Reusable input methods became a large part of the motivation behind the new system in place. Also, the varied requirements of these other types of systems enabled the discovery of some creative solutions to the problem.

The touchpad present in the car offered the most potential for an additional input method to be deployed that would be significantly different from the current remote based approach. Touchpads have proven their worth on personal digital assistants when used in conjunction with handwriting recognizers and, as such, makes it a potentially non-visual manual input method suitable for in-vehicle operation. For the current system being designed the user goals for it were to limit the distraction to the primary task that interacting with the system causes; however, the speed of the input method to allow for quick reactions was also a notable tradeoff that was taken into consideration when searching for new input methods. The current remote-based method offers the ability to very quickly change songs or perform other system functions at the single stroke of a button. Although its task time for selecting a specific song is slow, its speed for reacting to and performing specific commands should be matched.

5.4.1 The Graffiti Language

Taking a cue from notification systems in mobile computing, a freeform, single stroke symbol recognizer was implemented that took a set of ordered points representing a drawn character and returned a vector representing the symbol drawn. The specific algorithm used for the recognizer can be found in the appendix.

To meet the user goals of the carputer MP3 system, an input method should be as non-distracting as possible. From an expert standpoint, Unistroke offers much potential since its strokes are short and precise; however, it is not immediately as usable to novice users who must first learn the language and results in significantly more errors being made (MacKenzie and Zhang 1997). Errors generated while using the recognizer will, no doubt, increase the distraction level substantially; therefore, the very popular graffiti character set (Figure 4) was programmed into the character recognizer. Graffiti is a compromise for its similarity to the Latin alphabet and ease of recognition. Most characters are easily distinguishable from one another by very simple algorithms, such as the one implemented, yet the characters in the set still closely resemble the characters they generate. The usability of graffiti has been shown to offer recognition rates to novice users with only 5 minutes of training with very good long term retention rates even through non-use (MacKenzie and Zhang 1997). This is exemplified by the incredible popularity that Palm Pilot devices using the Graffiti alphabet enjoyed early in their development.

The graffiti symbol set was programmed into the recognizer by simply running the software in a record mode that saved a unique vector generated for each stroke into a database under the character that was typed in upon recognition. Multiple vectors were programmed into the system for each letter. Simple letters like “l” had only one or two corresponding vectors for the different ways in which it could be drawn; however, more complicated ones, like “y” had as many as 100 associated vectors. Input of the characters was also augmented by visual output of the path taken so far by a blue path drawn on a black background, meant to duplicate the surface area of the touchpad on the screen (Figure 25). This seemed to make drawing characters easier, but may have the opposite of the intended effect and cause more visual distraction to the driving task, since it requires the use of the user’s visual resources. This was flagged as a potential issue to be answered later through actual user evaluation.

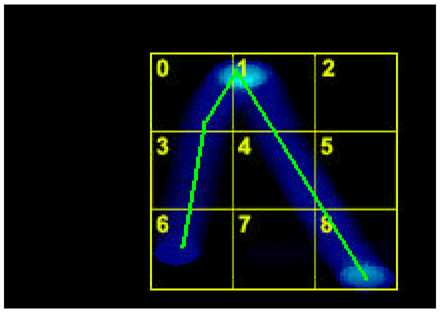


Figure 25. Touchpad symbol recognizer in action, the blue areas represent where the finger was moved across the touchpad, lighter areas indicate greater pressure or time spent in an area. The yellow grid encloses the areas interpolated to create a single vector representing the character.

5.4.2 List Selection with Graffiti

With the character recognizer built, the only remaining implementation issue was how to look up items from the song database using the recognized characters. This was solved through a query approach that used a search mechanism that would slowly get more and more specific as each additional letter was entered into the system, until only a unique selection existed, at which point it was chosen automatically by the system. Selection for this input method begins as soon as the user makes a recognizable gesture on the touch pad. The first alphabetically found artist starting with the recognized character is highlighted, and the list is shown with the first character for this highlighted artist specified in yellow to denote it was recognized. This item can be selected immediately through a simple horizontal left to right line gesture, or the user can enter a second letter into the system corresponding to the second letter of the artist's name to narrowing the results even closer to the desired selection (Figure 26). Selection continues like this until a character is input that forms query from the series of characters that either matches no items (causing list selection to end in error) or finds only one item (causing that item to be automatically selected and the corresponding song list for that selected artist to be shown). Song selection happens in a similar manner, allowing characters to be input to form a query until only one matching song exists. Error correction is supported though a simple “backspace” symbol, a horizontal right to left line gesture, that allows the user to undue the previous character input. Spaces and other non-alphanumeric symbols are ignored in all items within the database during matching, and numbers are converted to their word equivalent. As an example, to lookup “Apple” from a list which contains “Acorn”, “Apple”, “Ape”, and “Butter” you would need to draw the symbols for the characters “A”, “P”, “P.” After entering the second “P”, the item would be automatically selected, since only “Apple” starts with the characters “APP”.

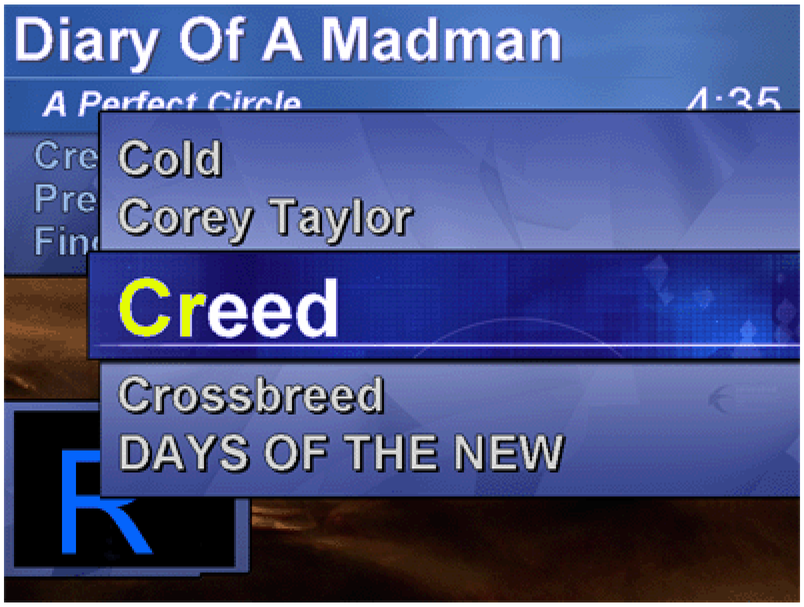


Figure 26. List selection in revision three of the carputer software. The graffiti recognizer in the bottom left of the screen shows the second recognized character as an "r". The list was shown when the first character "c" was received.

5.4.3 Critical Analysis of the Graffiti List Selection

Initial expert evaluation revealed that the graffiti method could, on average, be used to look up items much quicker than through using a remote to scroll the whole database, and most songs could be selected in about four characters. Some inefficiency still existed though. If two artists or songs categorized together both contain the same starting word, like “cross breed” and “cross multiply,” the method would become very awkward. This condition was handled by the addition of "next" and "previous" gestures to select the next or previous items in the list. So for the above example, if the letters "cr" were input into the system causing "crossbreed" to be highlighted, "cross multiply" could be then selected by simply using the "next" gesture follow by the "select" one. It is assumed that with time, expert users will be able to learn the series of gestures required to select a particular item from the list and perform them in order without having to look at the screen for confirmation. To support novice users in looking up items without using visual resources, audio feedback was added to the interface. Whenever a character was recognized, which highlighted a new item in the list, a sound was played. If a character was entered that made a search query that returned no items, a different error sound was played. One more additional type of sound was also played when an item was successfully selected. Expert evaluation revealed this method to be capable of selecting items without looking at the screen and made the whole process more usable in general. To allow for the graffiti method to compete on even ground with the remote method, command symbols were implemented for the common operations of "play", "stop", "next", and "previous" that created similar reaction times for those input methods.

5.5 Voice: The Ultimate Input Method?

The graffiti based input method offered a distinct advantage over the remote control method when used for large databases, proving the power of looking up items based on spelling rather than location in a list. However, on rare occasions, long character series would need to be input, which was difficult and time consuming. The interface also required the manual and sometimes visual resources of the user. Speech recognition solves both these problems. However, speech recognition is difficult, which is why it was not yet implemented. The technical challenges are significant, but even more discouraging is that audio only interfaces cannot be designed like typical visual-manual ones (Hindus, Arrons et al. 1995). Take these problems and add to that the fact that in a vehicle, the audible environment can change significantly based on many events such as open windows, passengers, outside traffic, or, particularly on point for the primary focus of the carputer system, music being played loudly from a stereo system. The answer is clear why voice based systems have yet to gain wide spread usage, particularly in cars. However, these problems aside, the input method offers a very enticing solution to the problem at hand: Select an item from a potentially huge list without actually having to go through the whole list. This was accomplished in part by the graffiti based touchpad approach, but from just brief initial use of the system, this was not as elegant of a solution, as universally usable, or as safe as voice could potentially be.

Microsoft, like many others, has also realized the power that voice interfaces offer, and to this end, has developed low level operating system support for the technology in the form of a software development kit (SDK) for Windows (Microsoft 2002). The SDK offers two modes of interface support: command and control (in which only a limited vocabulary is given to be recognized from) and dictation mode (which allows users to speak any word and have a text representation of it constructed). Command and control mode was used exclusively for the carputer, as it offers significant accuracy advantages and does not require any training.

Building a command and control voice interface requires first constructing a grammar. This is akin to laying out the controls on a window for a more typical visual application. Support for all the standard multimedia controls were statically constructed into this grammar that started with the command word “carputer” followed by one or more action words such as “play”, “stop”, “next”, or “previous”. The real power of the voice interface, came into play for song selection. Selection was done through a dynamic grammar that was initiated through the command “carputer select” followed by the artist then by the song name. When recognized, this resulted in the quickest reaction times for song selection and offered little distraction in visual or tactile forms to the primary task. Many other unobvious benefits were also achieved from the interface through its much larger command space. This allowed the system to support many more functions without cluttering a visual interface. For example, selecting a song could be achieved through the aforementioned method, which would then immediately play it, but it could also be enqueued in the play list for later listening by using the command “enqueue” in place of select.

5.5.1 Critical Analysis of the Voice Interface

Expert evaluation revealed the performance of the speech recognition system to be above expectations. In a controlled lab environment, recognition accuracy was close to 100%, when the command word that was spoken was in the database. The system was even able to pick a voice out from the background noise generated from the playing MP3 file in the background in all but the loudest of circumstances. In short, the only obvious technical problem based on recognition accuracy was that the system would sometimes perform commands if random words were casually spoken even if they were not in the database. This basically makes the system almost unusable as currently implemented if background speech is present, since even if it only picks up a misread command for 1 out of every 100 random words said, the system would almost always perform a command in error over the words in a typical conversation.

The system also had other issues, which were mainly not technology limitations, but rather based on unavoidable user errors. Many artists and songs have names that are pronounced or spelled strangely (e.g. Lincoln spelled Linkin) or use foreign words. This in turn may generate a correct grammar that will be recognized, but only if the user pronounces the name according to the correct English spelling. An error like this is exceptionally critical for the system, since if the user does not know how to pronounce the name properly, there is no way to access the song, leaving the user frustrated and believing the song is not in the database or that a recognition error has occurred. Another problem with the interface comes as a result of the system's ability to be used without visual resources. This requires that the user must know all the songs in the system in order to access them. This is particularly interesting to note, since it may well be worth the attention tradeoffs, as long as they are still low, to use a visual system for this added utility that it can provide. Finally, the voice interface was extremely processor intensive and did not work with accuracy above 50% when it was moved to the significantly slower carputer platform from the development system that prevented further evaluation.

Table 1 – Comparison of input methods for the carputer application. Each input method is listed with the relevant issues and benefits associated with it.

Input Method - Mapping	Benefits	Issues
Microphone – Speech Recognizer	Hands free, eyes free input method!	Use for an application primary used for audio output creates a conflict. Pronunciation errors were particularly prevalent given the unique names of many songs and artists and inability to

		pronounce a name right meant the song could not be played. Remember exact song names was sometimes difficult, basically only works with favorite songs (Recall vs. Recognition). Precludes the benefits of a visual system.
Touch Pad – Graffiti Character Recognizer	Potentially eyes free input method. Query based method potentially faster then scroll and select ones.	Requires exact spelling to be remembered. The number of letters needed to select a song is dynamic so the user may not be aware of when a selection occurs. Similar spellings result in long lookup times.
Remote Control – Visual Interface Changes	Extremely natural and easy to use. Allows any song to be found without knowing as much about it. (Recognition vs. Recall) Very good for performing simple function like play.	Highly visual. Potentially slower then Graffiti and voice.
Touch Pad – Cursor Control	Similar usage to desktop counterparts. No implementation required.	Extremely highly visual, highly manual input method. Breaks in the secondary task needed to attend to the primary one greatly decrease performance. Bad for both list selection and simple functions.

5.6 The Need for a Design Analysis

This chapter has demonstrated the design of a typical in-vehicle notification system and multiple input methods that were proposed to control it. The design methodology used up to this point for the carputer system has worked well to propose and validate a few input methods; however, many assumptions were made about how typical users would perform and unsubstantiated criticisms were made of all the input methods, but the true tradeoffs behind them were not fully explored. The IRC focus in which the input methods were described helped illicit the problems with the interfaces, particularly the unobvious ones. However, the only evaluations performed were by expert analysis. This method was very helpful and found a number of basic problems, but it is lacking in scientific validity and indicates only issues with the system that may be apparent for a very specific subset of expert users. Continuing on the notification systems theme and using the IRC critical parameters to frame the problems, a user based design analysis will be proposed in the next chapter that will help to identify the best input methods for the secondary task at hand, drawing out the particular aspects of each input method that makes it succeed, as well as the tradeoffs it allow it to do so. While this design analysis was tested with this design case, it is expected that it can be generally useful to any input method evaluation for notification systems.

Chapter 6: Methodology

This chapter will introduce an analysis technique for the design and evaluation of input methods for notification systems. The method was developed to augment existing formal evaluation processes for notification systems to place an emphasis on the role that input methods play in overall system design. It was supported in this thesis with an empirical evaluation focusing on the tradeoffs inherent in the notification system design; however, the analysis technique could have similarly been applied to other types of studies. Even though, the proposed method incorporates this empirical study for validation purposes, the real contribution of the evaluation lies in the framework for analysis that attempts to indicate the properties of the input method in relation to the primary and secondary tasks within the notification system. It is meant to force developers into thinking about the attention-utility theme that dictates the performance of a notification system when designing an input method.

6.1 The Need for a Notification System Evaluation for Input Methods

Unlike designing traditional input methods, designing an input method for a notification system is not about creating the best way to interact with the task at hand, rather it is about finding the input method that most closely matches the amount of distraction allowable from the primary task while meeting the user goals from the secondary one. However, an input method for a notification system is still an input method. Assuming its raw task performance should not be very different than an evaluation for an input method not supporting a dual task situation. In terms of benchmarking scores for fastest system, the same performance with an input method on a single task system should be achievable on a notification system. The tradeoff is that primary task performance may suffer greatly, rarely the most desirable user goal. This suggests that all the evaluations used to rate input methods in a single task role can still apply, but only if the context of the primary task is used to calibrate this rating in a way that reflects any undesirable attention tradeoffs made for high utility.

A notification system where input is particularly challenging, such as in mobile computing and IVIS, is often characterized by significant hardware constraints. This limits the types of devices that can be used. For example, on cell phones, where the input device is often restricted to just a 12 button keypad, a number of different input methods have been shown to be possible. However, they vary quite a bit in actual performance. This is where many existing taxonomies and some other well known types of analyses sometimes fail by not making a distinction. The input device design space suggested by Card et al. (Card, Mackinlay et al. 1991) would classify all of them as absolute linear input devices and fail to differentiate them by any objective means. The problem of the Card et al. design space comes from the categorization of input methods based on coordinate space alone (i.e. a mouse is distinguished by performance in x,y coordinate space where a button is distinguished by performance in a single z coordinate space). This thesis will suggest an analysis that takes into consideration the end result of the input method (primary task and secondary task performance) over the subtleties of how the input method works to avoid this problem.

6.2 What Makes a Good Evaluation for Input Method?

Many objective evaluations already exist for rating input methods, some even specific to devices that are often used in a notification system role (Grasso 1996; Clausen and Pedersen 1998; Gallagher 2001). The problem with a lot of these traditional input method evaluations is that they are only focused on input. They do work well for evaluations when the only consideration is the task the input method is working with; however, when used for a dual task environment, the proper emphasis on distraction to the primary task is not always expressed in the results and they may use a performance metric in a very specific context to the device it was designed for, not notification systems in general. More importantly, is that these simple performance metrics, although useful as a benchmarks in single task situations when there is the only one consideration, (e.g. speed), rarely provide and insight in exactly what makes one device better then another or how to fix problems indicated by poor performance (Griffen 2001). The important question for a design analysis to answer when used in an iterative design process is not so much whether an input method simply performs badly in some way, but rather how and why it failed.

6.2.1 Criteria for an Input Method Design Analysis

This list of criteria necessary for a good design analysis was developed from a review of related literature (Butts and CockBurn; Clausen and Pedersen 1998; Turner 1998; Kidd, Orr et al. 2000; Anderson, Abdalla et al. 2001; Gallagher 2001; Mäkelä, Salonen et al. 2001; MacKenzie and Soukoreff 2002) and an analysis of the design process used for the carputer system in the previous chapter.

A notification system input method evaluation should be able to:

- consider the simultaneous effects caused by the primary and secondary tasks.
- distinguish between input methods based on the same input devices, but with slightly different mappings for example iTap and MultiTap for text input.

- distinguish between input methods that use different devices, but perform the same mapping function such as a touchpad with a character recognizer and a keyboard for text input.
- compare broadly different input methods like a voice interface and a remote control interface for selecting list items.
- work with the full range of notification systems from more traditional ones, like those found on desktops, to more exotic ones, like those found on mobile devices.
- suggest and justify design decisions by offering the how and why of performance, rather then just a benchmark.
- express specific problems with the input method in a way that is understandable outside of the domain it was applied on.

6.3 Scenario Based Design for Evaluation

When humans interact with computers, they rarely do so in simple and unambiguous terms. This is particularly true with notification systems. Whenever a user does interact with such a system it is always done while another task is ongoing. Although the primary task does not necessarily have anything to do with the user goals for secondary task, it still always provides important context about the device. For example, imagine how the context of being in a vehicle affects the design of an IVIS type of notification system, or that of being able to use a mobile system anywhere affects the design of a PDA one. The two platforms have many things in common, but small differences can adversely affect the system's performance. Scenarios help designers envision the context that creates these differences and reason about the situations in which the system will be used. This context is particularly important for input methods, as it not only provides grounded reasons for why the goals of the primary and secondary task should be met, but also shows a direct cause and effect relationship between the input method and both tasks.

6.3.1 What is a Scenario?

Simply put, a scenario is just a story told about how the designer envisions a typical user operating the system. Scenarios give designers a way to document complex issues with use that may occur within a system (Carroll 1999). Scenarios include characteristic concepts such as the setting, goal, plot, and agent. These elements elicit details about the specific program design, such as where the application is typically used, what users expect to get from it, how they expect to achieve their goals, and who will be typically performing the task. Extracting these elements about a very specific context makes it easier to determine certain aspects of the system and make general assumptions about constraints placed on the users. For example knowing the agent is a driver and the setting is a vehicle greatly restricts the number of options that developers should use when designing their systems. These elements are also typical of the primary and secondary task demands that characterize notification systems, so by acknowledging them, a designer will be able to reason about the composition of the input method's environment.

6.3.2 A Scenario for Developing a Design Analysis for Input Methods

Using the scenario-based theme, a question and answer scenario describing a hypothesized input method design phase for the carputer system from chapter 5. This type of scenario approach has many benefits, but it is being used here illustrate some of the finer points that a design analysis must consider, including resource usage and user goals. Please note that even though this scenario shares many things in common with one used for an evaluation of an input method, it is related from a system designer's perspective rather than from a system user's.

Question:

> *I have this notification system task. It's basically selecting an item from a list. It will be used within an in-vehicle information system in conjunction with an MP3 player to select a song to be played by the stereo system of the car with a large screen for visual information delivery. The system should never have to interrupt the user, and the only time they will use the input method is when they want to hear a different song than the one currently playing, so reaction will need to be supported, but it should not cause a interruption. The number of songs in the database will probably number at least in the hundreds to make the system worth it. The name of the song and other information about its current status will be output to a screen, so the user should learn the names of the songs. The primary user of the system will be the driver, so the user will need the use of both eyes to view the road most of the time, but quick glances are acceptable. At least one hand will need to be kept on the steering wheel; both will be needed when turning is necessary, and obviously the task must not overwhelm the driver's attention span and cause him to lose concentration on the road. What is the best input method for this task?*

Answer:

> *From the description, it is important that the input method not consume a lot of the driver's visual resources or be cognitively demanding. Tactile*

resources seem pretty important too, but a lot of input methods exist that only require using one hand so this should not pose a major problem. Audio resources will also, no doubt, be in use by the secondary task. This still brings to mind a number of different input methods, some that can be ruled out immediately.

A mouse only uses one hand, but it seems to be very inconvenient to use in a car. A touch pad mouse may work better from a hardware integration standpoint, but when used with a cursor it is a highly tactile and visual input method. I cannot see it not competing for the driver's consumed attention. The same goes for any other input method that moves a cursor. Touch screens may offer better performance, but the actual location of it will play a large role. It's also a manual method that does not offer tactile feedback, so it may be too visually demanding. A remote control would offer better tactile response, but would still be a visual method and may take a long time to select a list element. Voice would seem to be a good, non-visual choice. Good speech recognition can pose some significant cognitive demands when users have to think of a song name, but the system supports comprehension of the song names with its visual output, so this may get better with time. Also, noise from the car and even from the secondary task may inhibit this method. You may want to consider why voice recognition is an otherwise good choice and use another input method that capitalizes on these same things, but maybe isn't so audio dependant.

6.3.3 Scenario Analysis

The previous scenario described the problem and revealed that the notification system was constrained in a number of ways by the primary task, articulated by the solution portion of the scenario. These constraints fell into four different categories (*Tactile, Visual, Audible,* and *Cognitive*) that fully described the attention demands the input method needed to compete with. These categories should be used as elements in any scenario describing an input method. The other discussions on the input methods focused on how the secondary task user goals, described in terms of the notification system critical parameters could be used to suggest the best input method. Interruption was not considered, as it was not a user goal in the notification system. However, reaction was touched upon when it was suggested that the remote control might be too slow. Comprehension played the largest role, since it offered significant tangible benefits to voice control.

These elements make up the foundation of the design analysis used in the proposed evaluation methodology. Input methods are categorized according to their effect on attention and according to their fit with the user goals of the secondary task. This classifies the input method in a way that allows designers to focus on the aspect of an input method that will result in poor performance. Classification is based on a high-medium-low scale that is explained in Table 2. Resource usage metrics are only valid if the secondary task with which it is used is also defined, since a few of the metrics require knowledge of the notification system.

Table 2 – Table for calculating resource usage metrics for the input device.

Tactile	
High	Requires complex interactions with more than one axis of movement (*i.e. a mouse*) used for both input and interactive feedback.
Medium	Requires a single axis of movement. (*i.e. button, slider, rotary knob*).

Low	Not tactile; used for simple feedback only

Audible	
High	Requires both audible input and interactive feedback to be used effectively.
Medium	Requires audible input only or interactive feedback only.
Low	Not audible or used for redundant feedback only

Visual	
High	Requires continuous visual feedback to be used successfully used. Pauses needed to attend to the primary task significantly reduces efficiency.
Medium	Input method greatly aided by visual feedback, but it is not required for many situations or the visual feedback can be disturbed by pauses created by the primary task.
Low	Not visual or used for completely redundant feedback.

Cognitive	
High	Requires the user to recall a specific name or thing.
Medium	Recall aided by confirmation of items.
Low	Instinctive or natural action, (i.e. repetitive action like pressing a button).

6.3.3.1 IRC User Goal Mapping Metrics for Input Methods

Classification according to the user goals that an input method can be gauged from is also based on a discrete selection scale explained in Table 3. These metrics can only be applied to an input method when the secondary task IRC critical parameters are known. The goal mappings are meant to highlight ways that different input methods take advantage of the properties present in the notification system and indicate why one particular input method could achieve better performance over another. These questions focus designers toward thinking about the tradeoffs for certain input method and promote the creation of new input methods that would broaden the design space of input devices.

Table 3 – Table for calculating an input method's compatibility with a notification system's user goals.

Interruption *Will the input method work better if it is only used when the secondary task shifts into focus?*	
Critical	An interruption of the primary task is required for successful operation. (*i.e. a keyboard being used for the primary task is part of the input method for the secondary task*)
Helpful	Interruption of the primary task makes the input method more efficient, but is not required (*i.e. An interruptive prompt could queue speech recognition and mute any audible system output*).
Low Importance	The input method works just as well regardless of whether an interruption occurs before use or not.

Reaction *How fast will the user be able to react to an incoming notification or a need to change a system state?*	
Fast	Requires only a single action with an immediate response to perform the task on the notification system. (*i.e. Reaction can be accomplished through a single button press*)
Average	Simple action, but reaction is not immediate (*i.e. Giving a voice command that requires processing*).
Slow	Complicated multistep action accompanied by high attention demand. Reacting with such a method can only be done if the secondary task has the user's focus of attention. (*i.e. responding to an email inbox notification requires using a*

	mouse to switch to the mail program)

Comprehension *Will the input method work better if the user becomes aware of the information within the system?*	
Critical	Comprehension of notification information is necessary to achieve good usability of the input method. (*i.e. the name of the list element must be known to look it up with a voice interface*)
Helpful	Some comprehension of notification information is required but most of the time it just makes the process more efficient (*i.e. looking up list elements with a keyboard, knowing the spelling of the word helps, but is aided by visual feedback provided by showing the list elements*)
Low Importance	Comprehension of notification information has almost no effect on the usability of the system.

6.4 Scenario Based Evaluation of Input Methods

Scenarios have been proven to add benefit to the design process if used to capture and reason about context (Turner 1998; Carroll 1999). This is particularly important to note for input methods for notification systems. Context describes the role input will play on the secondary task and the side effects it could have on the primary activities. This context allows designers to better envision the dynamics of the interaction between the input device, the notification system, and the primary task through concrete examples of use. Assumptions about the tradeoffs inherent in these interactions can be formally stated through the resource usage and goal mapping elements within the scenario description. Implied assumptions about design artifacts can be made explicit through making general claims extracted from the scenario (Erskine, Carter-Tod et al. 1997). The scenario grounds the study in the usage constraints encountered in the real world. The incorporation of the critical parameters for notification systems into the claims makes the study implications generalized and replicable in a similar manner for varied input methods, and the inclusion of resource usage elements allows the role of the primary task to be taken into consideration.

Scenarios can be used in a number of ways, from finding faults within a design logic before a system is even built, to a formal narrative of an actual usability study, which can be used to retool the project design after a prototype has already been built (Turner 1998). In this case, scenarios are used to help guide an analysis of an input method for an already designed notification system.

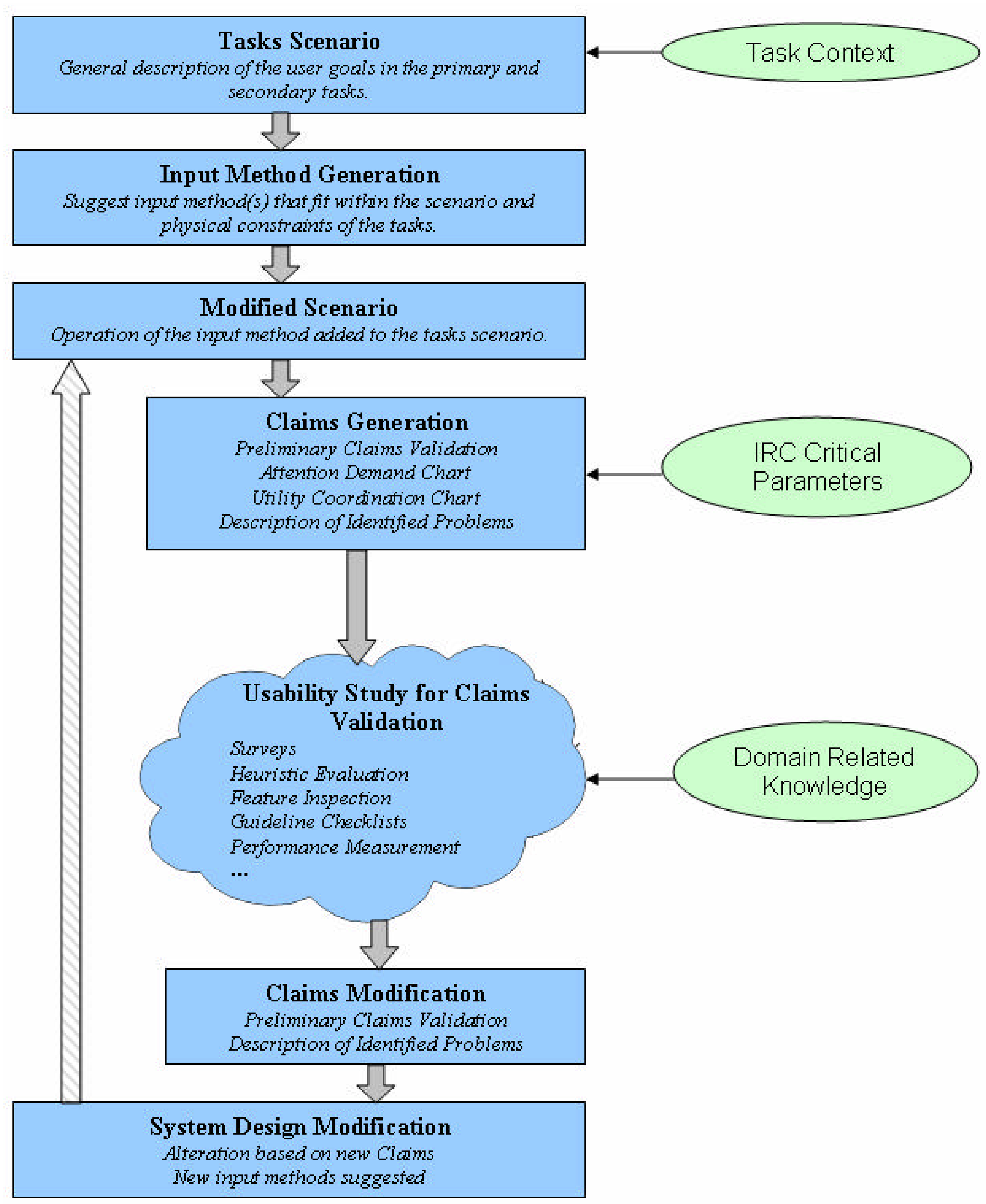
6.5 The Proposed Design Analysis

The goal of any notification system is to properly leverage the right amount of a user's attention in exchange for utility. Distraction should be kept to a minimum, and the secondary task's user goals sho uld be supported through tradeoffs made between interruption, reaction, and comprehension. This design philosophy was the guiding principle for the design analysis proposed. It is achieved through the use of claims extracted from scenarios that hypothesize about the input method for the specific notification system being considered. Results are expressed in terms of the attention demands from the primary task and the utility basis for the secondary information. For any possible notification system with an input need, the design anlysis (Figure 25) works as follows:

1. Develop a task scenario, leaving out any specifics of the input method, but with artifacts that highlight the main user goals and tradeoffs between the two tasks. This should be a purely user goal-oriented description that does not go into detail about how an input method works in the system, but does show an input need.

2. Using the elements of the scenario, construct a resource chart for the primary system using a high-medium-low scale over the four types of resource metrics and an IRC chart for the secondary task indicating the user goals for the secondary task.
3. Suggest a new input method or a common method for another type of task that can answer the basic needs of the system highlighted by the task scenario.
4. The original tasks scenario can now be modified using the suggested input method to reflect the operation of the input method. The effects that it has on the primary and secondary tasks, and vice versa, should be clear.
5. A resource usage metrics chart and IRC goal mapping chart should be generated for the input method. Claims should be made for each metric and mapped to express the rational behind the choice based on a specific input method characteristic.
6. A form of usability test can now be run to verify the merit of the claims. This can range from a number of different methodologies as long as it encompasses the interaction of the users with the system. This usability test can be one that primarily measures the input method speed or tests the actual notification system as a whole, but should record the role of the input method and simulate the input method being used for the secondary task while a primary task is ongoing.
7. The original claims made for the system should then be modified according to the results of the usability study.
8. Design decisions can now be made based on an understanding of these claims to modify the original input method.

9. The input method can then be related with the new designs implemented in the iterative design process.



Tasks Scenario
General description of the user goals in the primary and secondary tasks.
Task Context
Input Method Generation
Suggest input method(s) that fit within the scenario and physical constraints of the tasks.
Modified Scenario
Operation of the input method added to the tasks scenario.
Claims Generation
Preliminary Claims Validation
Attention Demand Chart
Utility Coordination Chart
Description of Identified Problems
IRC Critical Parameters
Usability Study for Claims Validation
Surveys
Heuristic Evaluation
Feature Inspection
Guideline Checklists
Performance Measurement
...
Domain Related Knowledge
Claims Modification
Preliminary Claims Validation
Description of Identified Problems
System Design Modification
Alteration based on new Claims
New input methods suggested

Figure 27. Diagram of the proposed design analysis.

6.5.1 Limitations and Benefits of the Design Analysis

The design analysis has benefits over simple usability testing, in which a strict performance metric is used, or when usability testing is only done on the notification system itself. The scenario based approach allows context to be used to shape the design of an input method. However, this method may not necessarily reveal definitively the better input method for a certain task. The method relies on a usability test to determine a better or worse design. Although a notification system-specific usability test is recommended to judge the system while using the input method, input method-specific studies can also be conducted to reveal raw performance. The design analysis will still give the tradeoffs of each method that make it better or worse than others in certain ways, but they may not be empirically validated in this case.

The structured claims generation portion that is the heart of the design analysis will allow designers to improve upon existing ideas and give suggestions as to when to use a particular input method. Traditional usability testing alone that offers an objective best / worst approach to a particular problem can not be correct for all situations that an input method could be used in. For example, performance metrics for the IBM track point mouse revealed it to be just as efficient as a regular mouse (Zhai, Smith et al. 1997), but subjective tests revealed people still substantially preferred to use the track point mouse. Only analysis of the mouse's other effects could reveal why. The next chapter will demonstrate the design analysis further through an empirical experiment.

Chapter 7: Experiments

This chapter will discuss a usability study conducted to improve the carputer application discussed in chapter 5. Studies of a similar nature are common for IVIS, and many of the metrics used were based on other experiments conducted from the domain. The study's aim was finding the effects that three different input devices had on a notification system in terms of interruption, reaction, and comprehension. The only variation between experiments was the input method.

7.1 The Design Analysis in Practice

The design analysis described in the previous chapter was put into practice with an empirical usability study, starting with the first step in the process formulating a general tasks scenario. Discussion of the design analysis will continue in the next chapter with a presentation of the experiment results and the new design decisions suggested based on them.

7.1.1 Tasks Scenario

Ed Driver's new car is equipped with an MP3 carputer music system. He is currently driving the speed limit of 45 mph. The bus in front of him has a sign for the new CD by his favorite artist, which Ed just uploaded to his in car system. Ed now wants to hear John Doe's new hit single, but he cannot quite remember the title of the song. Ed selects the artist from the system, which shows the list of songs by John Doe. However, while looking at the song list on the carputer's screen located in the passenger side of the car, Ed sees the brake lights for the bus in front of him come on and his attention is brought back to the primary task of driving. After adjusting his speed to match that of the bus, he glances back at his screen, sees the name of the new single, and selects it.

The scenario resulted in the following tables of the primary task resource usage (Table 4 – Carputer MP3 player: Resource usage of the application calculated based on assumptions from use and intended design and documented through claims.Table 4) and secondary task user goal description (Table 5), based on the elements in the description. The values for these two charts may be arrived at in different ways, applying knowledge of the primary task revealed through past research or updated notification system's design philosophy for the secondary task description. General methods can not be successfully applied to the primary task, since the metric of high, medium, or low resource usage will vary from system to system. The resource usage was generated based on a general understanding of the IVIS domain, but different methods may be used in a similar manner if they are found to be more accurate. Also, the suggested resource consumption is meant to be accurate only from a scenario standpoint that will help guide design decisions and selection of critical factors for the usability test to cover. It is not meant to be scientific standard for all in-vehicle information systems.

Table 4 – Carputer MP3 player: Resource usage of the application calculated based on assumptions from use and intended design and documented through claims.

Carputer MP3 Player Resource Usage			
While driving, one hand is normally free to perform other tasks, so **tactile** resource usage is *medium*, Although **audible** resources are often present during driving, they are rarely required, so usage is *low*. **Cognitive** resource usage may be continuous while on the road, but it is rarely demanding of the full attention of the user so usage is *medium*. Finally, driving is a *highly* **visually** demanding task requiring both eyes to be on the road at most times and at least peripheral vision at others.			
Tactile	**Audible**	**Cognitive**	**Visual**
Medium	Low	Medium	High

Table 5 – Carputer IRC user goals: Calculated for the application based on assumptions from use and intended design and documented through claims.

Carputer IRC User Goals		
The carputer system does not offer any prompts to the user, so **interruption** is *low*. Quick **reaction** is desirable to minimize distraction, but not required to take the most advantage of the system, and the user may want to learn the songs in the database so **comprehension** is *medium*.		
Interruption	**Reaction**	**Comprehension**
Low	Medium	Medium

7.1.2 Input Method Generation

The scenario of use, resource usage, IRC goals, and general knowledge of existing input devices resulted in the following three input methods being generated. Two have already been described in chapter 5 with the carputer system. The first of these two is the remote control based method—a tactile method that usually requires visual resources to use and can be found under section 5.3.1 with the heading *List Selection with a Remote Control*. The second graffiti based method is also tactile, but can be used without visual resources and employs a fundamentally different input device and mapping. It can be found under section 5.4.2 with the heading *List Selection with Graffiti*. The third input method was not implemented yet in the carputer system.

The final method uses a similar mapping to the remote based method, with a different input device (a touch screen) to perform the same task. The interface is shown in (Figure 28). If the item is visible, it can be selected by just clicking on it. Scrolling is achieved through the scroll up or down arrows. The arrows may be held down to move the list progressively faster or they can be clicked once to move through the list one item at a time. When the item is highlighted as the center element, it is selected by clicking the select button. If an unwanted category is already selected, the back button can be pressed once to return to the category list.

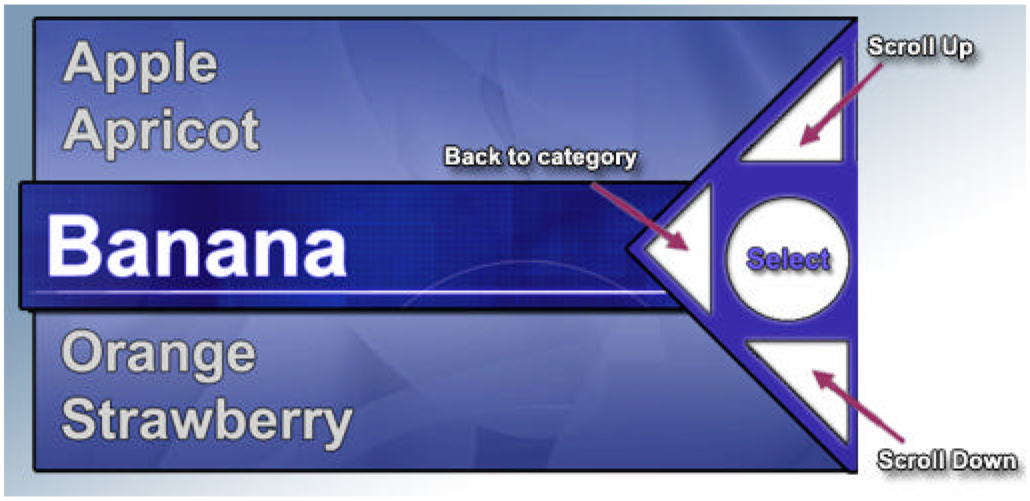


Figure 28. Touch screen interface for the carputer application. The scroll up and down arrows allow the list to be browsed, selections are made by tapping on the center select area. The list categories can be shown (if applicable) by pressing the back button.

7.1.3 Modified Scenarios

Three new scenarios were generated based on the original one, but with plot elements added detailing each input method's role in the system. Only the remote control scenario is shown here for brevity. The other two scenarios can be found in the appendix.

7.1.3.1 Remote Control Scenario

Ed Driver's new car is equipped with an MP3 carputer music system. He is currently driving the speed limit of 45 mph. The bus in front of him has a sign for the new CD by his favorite artist, which Ed just uploaded to his in-car system. Ed now wants to hear John Doe's new hit single, but he cannot quite remember the title of the song. Ed first shows the list of available artists by pressing the left arrow on his remote control. He then scrolls down to the artist "John Doe," by pressing and holding down the down arrow button then presses the select button, which brings up the list of songs. However, while looking at the song list on the carputer's screen located in the passenger side of the car, Ed sees the brake lights for the bus in front of him come on, and his attention is brought back to the primary task of driving. After adjusting his speed to match that of the bus, he glances back at his screen, sees the name of the new single and uses the remote to scroll up and highlight it. He then selects it using the select button, and it begins playing.

7.1.4 Claims Generation

The input method scenarios lead to claims (Table 8) being generated with respect to attention demand (Table 6) and IRC agreement (Table 7). The general claims about the input methods should provide validation and rational for non-obvious values in the chart. See table 3 for a complete definition of the meanings of the values assigned to the different input methods.

Table 6 – Resource usage for the individual input methods used with the carputer MP3 application.

Input Method	Tactile	Audible	Cognitive	Visual
Touch Pad – Graffiti Character Recognizer	High	Low	High	Low
Remote Control – Visual Interface Manipulation	Medium	Low	Low	Medium
Touch screen – Direct Interface Manipulation	Medium	Low	Low	High

Table 7 – Input method utility coordination for the individual input methods used with the carputer MP3 application.

Input Method	Interruption	Reaction	Comprehension
Touch Pad – Graffiti Character Recognizer	Low Importance	Average	Critical
Remote Control – Visual Interface Changes	Low Importance	Slow	Helpful
Touch screen – Direct Interface Manipulation	Low Importance	Slow	Helpful

Table 8 – Claims generated from the scenario of use on the carputer MP3 application.

General System Claims
Visual input feedback on the screen changes the position of the highlighted list element which   + Enables the user to react to the new selection.

- + Enables comprehension of the other elements in the list seen while browsing
- + Should allow for quicker browsing then a spoken list.
- – May cause unwanted interruption to the primary task by consuming the visual resources of the driver.

Touch Pad – Graffiti Character Interpreter

A graffiti character interpreter allows a user to look up items in a list using a single physical interface.

- + Lessens visual distraction by providing a method that does not require locating items on the screen.
- + Lessens tactile distraction by providing a hardware device location that never changes.
- – May cause cognitive distraction for user's to remember the spelling of the item.
- – May cause cognitive and tactile distraction for users unfamiliar with the Graffiti language.
- – May cause slower reaction times compared to a scroll and select method when the list size is small.

Remote Control – Visual Interface Manipulation

The portability of a remote control allows a user to remain in contact with the interface while performing other tasks which

- + Uses a scroll and select interface with minimal cognitive demands.
- – May increase distraction if the user has to locate the remote.
- – Increases reaction time to select items as the list gets larger.

Touch Screen – Direct Interface Manipulation

The soft interface of a touch screen

- + Uses a scroll and select interface with minimal cognitive demands.
- + Eases cognitive demands via a direct manipulation method.
- + Can change to better support a current state of the system.
- + Minimizes distraction by providing a hardware device location that never changes.
- + Large list sizes increase reaction time.
- – May increase distraction if the user has to locate the remote.
- – Decreases reaction time to select items as the list gets larger.

7.2 Experiment Platform

A usability study constructed around the scenario and claims will now be described. The study used a constant primary and secondary task, varying only the three input methods (Graffiti, Remote Control, and Touch screen). A total of 33 users were run, 11 for each condition. Each participant was only asked to use one of the three different input methods since the study was a between subjects design. Quantitative as well as qualitative data in the form of surveys was collected from the users to measure the different effects the input method had on interruption, reaction, and comprehension. The test consisted of a number of components automatically controlled by a usability tester application. The application walked the user through the testing process, controlled the synchronization and running of the simulator and benchmark tasks, and collected all quantitative and qualitative data in a central location. Each experiment was conducted on two computers (a desktop and a laptop (Figure 29)) The desktop ran a driving simulator used to mimic the primary task environment encountered when driving, while the laptop computer ran the testing application and benchmark task, recorded all metrics received from the simulator, and was connected directly to the input device. The testing time varied from 15 to 45 minutes depending on the user and the input method being tested.

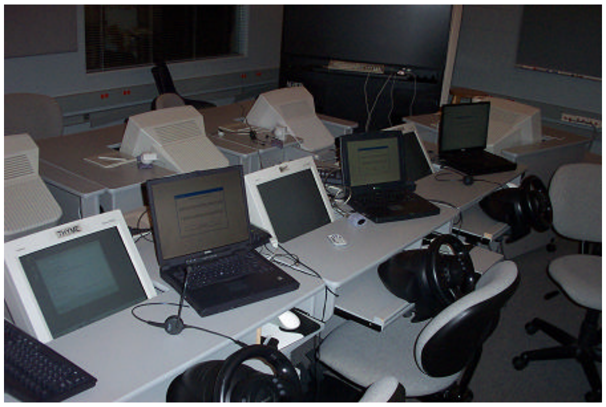


Figure 29. Image of the lab showing the three desktop computers used to run the simulator along side the three laptops used for the primary tasks.

7.2.1 The Primary Task

The driving simulator used for the primary task was meant to provide an attention demanding environment with similar demands in terms of cognition and vision to a real vehicle. The simulator was controlled through a steering wheel and pedals hooked up to the desktop computer (Figure 30). One button on the steering wheel was operational and reversed the vehicle when depressed simultaneously with the accelerator pedal. Operating the simulator was otherwise similar to operating a real car always in the proper gear. The only major problem of the testing platform indicated by many of the participants, was that the physics guiding the vehicle were not accurate and it often felt like the vehicle was driving on ice.

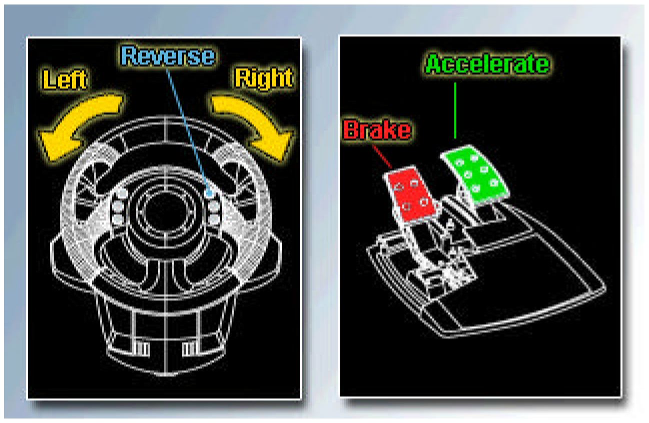


Figure 30. Illustration of the Steering wheel and pedals used to control the simulator. This image was given to participants as instructions for the simulator task.

To complete the primary task, drivers had to follow a large circular course that ran along the borders of a simulated town. The course was the same for all tests. Participants were guided by checkpoints that were indicated by 3D numbers placed on the road around the corners of every turn that needed to made in the course. No checkpoint required turning more than 90 degrees. An arrow at the top of screen always pointed to next checkpoint. During the test, participants were asked to maintain a constant speed as close to 25 mph as possible and within their lane for the duration of the experiment, but were not asked to obey any other traffic laws such as stop signs. The driver's speed was indicated in large letters in the upper left of the simulator screen with a color that indicated the driver's compliance with the speed limit. Green numbers indicated the participant was driving within tolerance, yellow indicated too slow, and red indicated too fast (Figure 31). No other vehicles were on the road which was flat and completely unobstructed.

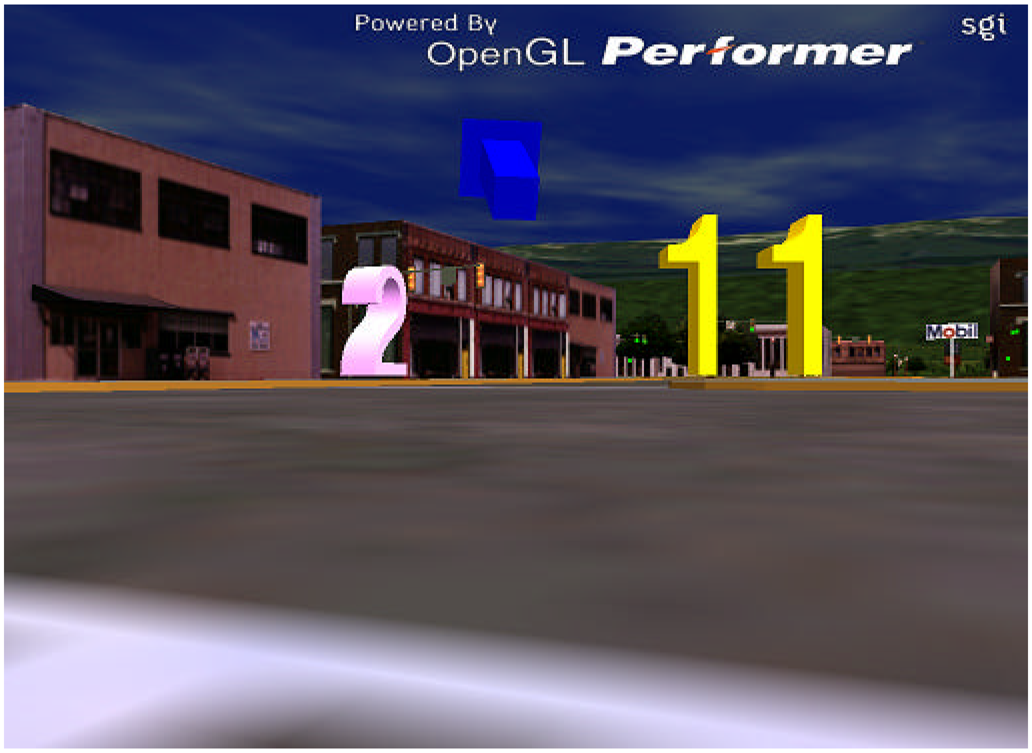


Figure 31. Screen shot from the driving simulator. The yellow numbers represent the speed of the driver, they turn green when the driver is going between 20 and 30 mph and red if 30 mph is exceeded. The pink 2 represents a checkpoint marker that the users must drive over to complete the course (Moldenhauer, 2003).

A brief training session provided for the participant to drive around a short course that covered approximately the first 25% of the evaluation course the system would be evaluated on. The training session automatically ended early if the participant appeared to be driving "badly." This was measured by a number of metrics, including exceeding 35 mph or driving off the road too many times as well as other indicators. The driver was required to start the course over again and complete all the checkpoints before being allowed to continue if one of these metrics was exceeded.

The test portion of the evaluation required the participant to drive the simulator in the same manner as they drove in the training session. The only difference now was that the simulator would continue regardless of the actions the participant made and a large text prompt was shown on the bottom of the screen when a category and item was to be selected for the secondary task. The simulator continued to run until it received a stop message from the laptop running the secondary task.

7.2.2 The Secondary Task

While driving the simulator, the participants were required to lookup an item in a list. This task was modeled after the task for the carputer application of finding a song in an MP3 database to play. It varied very slightly between input methods, but all used a visual list of names that was browsed when input was received.

Rather than songs (which might not be known by all participants), the data to lookup for the test was semi-random in nature; it was simply items that are common and were chosen on the basis that they should be known by most participants (See the appendix for a complete list of these items). This was done since the task that the usability test is modeling requires looking up songs to play from the system, which obviously the song names must be known to the user before they can be found. Also, just as the songs were categorized by artist, the list data was categorized into a single parent category, i.e. "Beagle" was an item under the category "Dogs".

Like the primary task, a brief training session was given to the participant prior to the actual usability study. For the remote control test condition, users were told the function of every button and then told to press it. The touch screen training session simply showed an image of the interface and described the touchable areas on it and their function. The Graffiti test condition was the most complicated and took the participants significantly longer to complete. It consisted of a lengthy training and validation period, since prior knowledge of the Graffiti language was not required to participate in the study and slight variations in the implementation compared to what some participants were use to needed to be controlled. Participants were first asked to make the Graffiti symbol for every letter in the alphabet on the laptop touchpad when both the Latin alphabet representation and its symbolic Graffiti form were shown. They were only allowed to continue to the next letter when the proper current letter was recognized. This was followed by a task that required the participants to spell out the sentence “The quick brown fox jumped over the lazy dog” in Graffiti without the use of the alphabet representation or spaces. If the participant made five or more errors while spelling the sentence, they were required to start over again and reenter the alphabet.

During the testing phase of the study, the list items that the user was required to lookup were shown as text in the simulator. They were automatically displayed on the screen at random intervals between 25 and 50 seconds. If the user was not complete with selecting the previous item when a new one was shown, they were instructed to begin looking up the new item and forget about the old one. The intervals between displaying items were sufficiently large though that this was rarely a problem. Average selection time for all input methods is under 15 seconds.

7.3 The Testing Software

There were three major components to the software used during the test: the driving simulator (Figure 31) used for the primary task, the list selection software that implemented the benchmark task, and the usability tester that coordinated both tasks and conducted the test. The simulator was run on Linux and based on OpenGL Performer

(SGI 2003), a 3D application programming interface complete with sample models and landscapes. The code and models for a town sample were modified with a physics engine to simulate driving, support a Microsoft SideWinder Precision Pro Racing Wheel and pedals (http://www.microsoft.com/hardware/sidewinder/), and collect data such as speed and position. The simulator was a modified version of the program used for another experiment (Moldenhauer, 2003) run examining information modality aspects of notification systems.

The list selection program for the secondary task was based on the same source as the carputer application (refer to chapter 5 for a more in depth description). It was changed to support loading of a list from a text file with the random elements, the status window that showed the status of the playing song was removed and the queue window was modified to show only list selections rather then the top three songs in the playlist.

7.3.1 Usability Testing Framework

The amount of test data required to validate the preliminary claims about the input methods required a number of different metrics to be collected. This included subjective data from the participants about their attitudes during the usability study as well as survey questions for demographics. Also all the relevant data needed to reconstruct the participant's actions during the study needed to be collected, including data from both the primary and secondary tasks, such as a lane exceedance. To accurately collect and organize all this data in a single place and administer the test, a tool was created that walked the participant through the different steps of the test. The usability testing tool used a wizard based approach that walked the participant through the test based on figure 32.

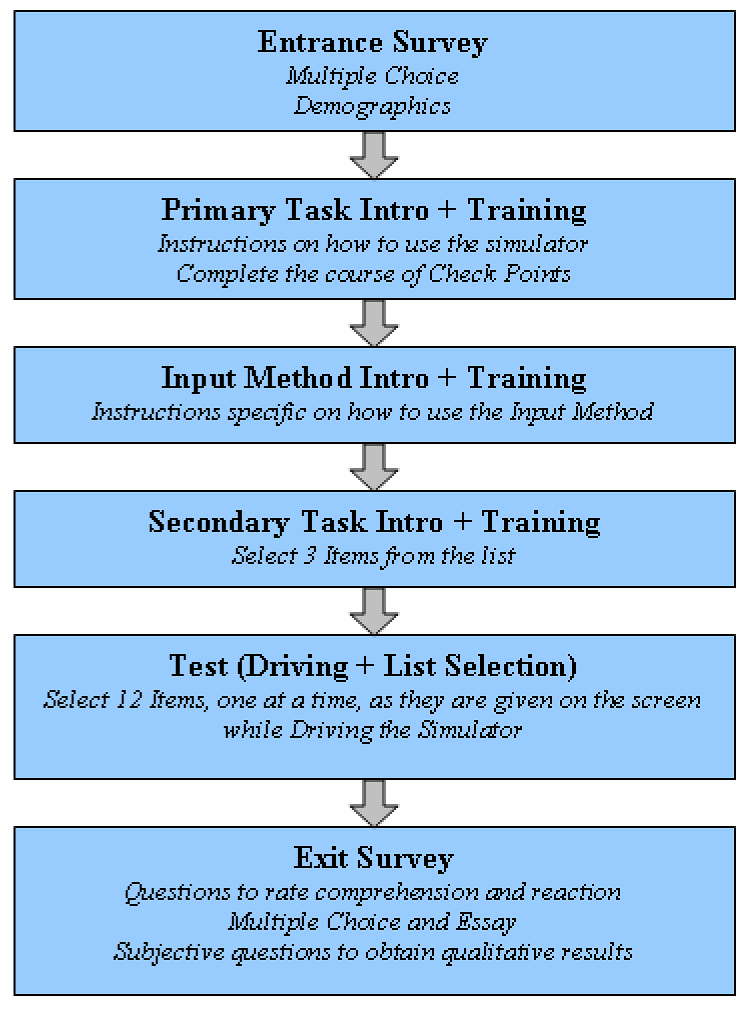

Entrance Survey
Multiple Choice
Demographics
Primary Task Intro + Training
Instructions on how to use the simulator
Complete the course of Check Points
Input Method Intro + Training
Instructions specific on how to use the Input Method
Secondary Task Intro + Training
Select 3 Items from the list
Test (Driving + List Selection)
Select 12 Items, one at a time, as they are given on the screen while Driving the Simulator
Exit Survey
Questions to rate comprehension and reaction
Multiple Choice and Essay
Subjective questions to obtain qualitative results

Figure 32. Diagram of the usability study - shows the different portions of the study participants were walked through to complete the full experiment.

The tool began with a start page to greet each participant and tell them how to begin the test. After pressing a start button, a wizard style window was shown to the user with previous and next buttons, disabled when appropriate, this guided the user through all the phases of the test. The first few wizard pages consisted of multiple choice survey questions used for demographics. Next the participant was introduced to the primary task of driving. The page contained instructions on how to operate the simulator as well as a command button that started it on the desktop computer by sending a signal over the network. A programming API called DIVERSE (http://diverse.vt.edu/) for sharing memory between two computers was used for this communication and collecting all metrics that needed to be recorded during the entire trial. The simulator was monitored for metrics as shown in Table 9. If one of these values was exceeded, the simulator was stopped and a descriptive error message was shown to the participant on the laptop.

Table 9 – Metrics monitored during the simulator training session. The maximum tolerance indicates the maximum value for the metric. If this value was reached the simulator would exit, tell the user that they failed and why, and give them a chance to retry to training session until they passed.

Metric	Maximum Tolerance (>= for Failure)
Maximum Speed	35 mph
Making the car spin out while turning	2 times
Times exceeding 35 mph	3 times
Collisions	once
Driving off the road	3 times

The last training session taught the participants how to use the input method with the list selection interface. General instructions were given to the participant on the process of list selection, then a category and an item were shown to be selected. Once the participant used the input method to properly select this list item, then next item was shown. This continued until three items were selected, at which time the participant was allowed to continue.

The testing phase of the usability study began when the participant pressed a button. This started the simulator on the desktop computer and the list selection application on the laptop. As the participant selected items from the list, the relevant metrics were recorded, including task times and simulator performance. After the participant selected the 12 items shown to them, the test ended and a final exit survey was given.

Chapter 8: Usability Study Results

An analysis of the results of the usability study revealed a rather startling fact: all three input methods resulted in no apparent performance difference. Although the input methods were chosen partially on the basis of their difference from one another, this was not apparent in the usability study conducted. Results for the usability test for the notification system were largely positive from a quantitative and qualitative approach though. All three input methods were within the parameters to be considered non-distracting, based on the metrics collected, and participants indicated they would feel safe using the input method on the road.

8.1 Distraction from Primary Task

Distraction caused by the input method while performing the secondary task was measured through judging the participants performance in completing the primary task. These metrics included the number of times the participant exceeded 30 mph, the number of the times the car spun out (angular velocity), the number of times the participant drove off the road, the number of times the participant went below the minimum speed of 20 mph, and the number of collisions the participant had with objects off the road. Metrics from the secondary task were also collected for correlation with events in the primary task. These metrics included when the item for selection was presented to the participant, when the participant started list selection, when the participant selected the proper category and item, and the time and type of any incorrect selections. Position data was also collected from the simulator at intervals of 1 second. This data included: time, position, speed, and direction.

Analysis of the data was done through statistical and visual analysis. Statistical analysis of the metrics revealed no significant differences between the input methods. A summary of this analysis is given below. The percentages shown are based on the amount of time users spent selecting items while driving as compared to just driving alone.

Table 10 – Participant performance while performing both the secondary task (list selection) and the primary task (driving the simulator). The percentages indicate the amount of time the participants were operating under the heading condition out of the total time they were operating the simulator.

Input Method	% Time Spent Selecting Items	Avg. Time Spent Selecting Items	Avg. Speed	% Time Speeding	% Time Too Slow
Touch Pad – Graffiti Character Recognizer	51.4% std. dev: 8.3%	11.4 s	14.9 mph	0.2%	35.4%
Remote Control – Visual Interface Manipulation	51.4% std. dev.: 16.1%	12.3 s	15 mph	0.5%	36.0%
Touch screen – Direct Interface Manipulation	51.4% std. dev.: 14.3%	11.3 s	13.6 mph	0.1%	35.0%

Table 11 – Participant performance while performing *only* the primary task (driving the simulator).

Input Method	Avg. Speed	% Time Speeding	% Time Too Slow
Touch Pad – Graffiti Character Recognizer	18.6	0.7%	18.5%
Remote Control – Visual Interface Manipulation	17.1	0.3%	22.6%
Touch screen – Direct Interface Manipulation	15.9	0.6%	23.4%

As shown by the results summary, the time users spent on selecting items in a divided task environment was exactly the same, at 51.4% for all three input methods. The individual average amount of time spent selecting per user varied from 28% to 79% within a single condition. However, the graffiti method resulted in much less of this variance from the mean value, with a standard deviation (Figure 33) of 8.3% compared to 16.2% and 14.3% for the Touch screen and remote methods respectively. This may indicate that graffiti is more consistently less distracting for the average user.

Participants also slowed down consistently by about 15% in all three input methods, but *all three conditions provided average task times under 15 seconds, indicating suitable use for driving performance*.

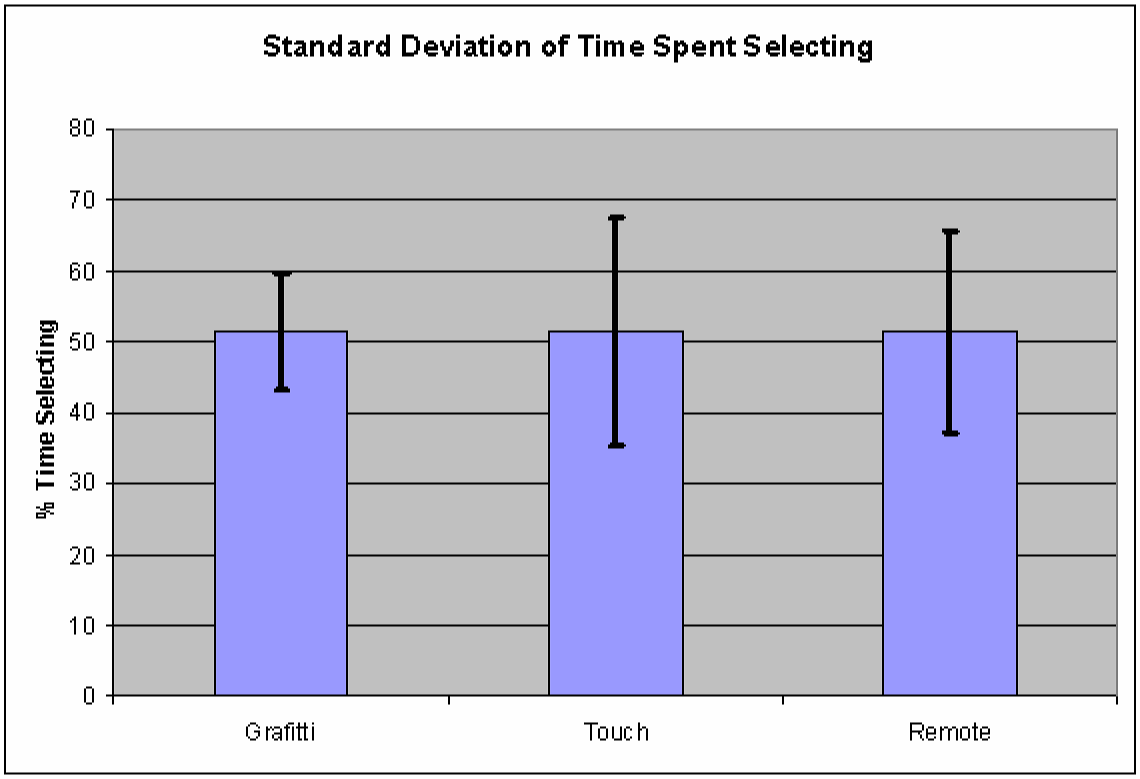


Figure 33. Standard deviation from the mean percentage of time spent selecting items during the usability study. The graffiti method showed the least variation from the mean value.

A couple of survey questions were also asked about distraction incurred from using the secondary task. They asked whether the participants felt the secondary task with the input method would be more distracting than a radio, and whether it would be safe to use in a car. The results were as follows. The two numbers indicate the number of users who answered affirmatively to the question versus the total number of users surveyed.

Table 12 – Results of survey questions asked during the exit survey relating to the distraction level of the secondary task (list selection). The first question asked whether participants found using the carputer application more distracting than operating a radio. The second question asked if participants thought that performing the secondary task in an actual vehicle should be illegal while driving. Answers are in terms of the number of participants that replied yes to the question vs. the total number that answered.

Input Method	More distracting than a radio	Safe to use
Touch Pad – Graffiti Character Recognizer	6/11	9/11
Remote Control – Visual Interface Manipulation	5/11	11/11
Touch screen – Direct Interface Manipulation	9/11	10/11

Although the subjective surveys were not enough evidence to draw any significant conclusions about the individual input methods, they did reveal that most participants found the task to be more distracting than using a radio in a car, but most did not think it would be unsafe to use while driving. These opinions cannot support a significant statement about safety, but they can be used to show that users may accept using the notification system in their cars.

Finally, the positional data collected from the usability study was analyzed visually to reveal if any complex trends based on input method could be found. To visualize the data, a tool was developed that reconstructed the path that a participant took during the test. All the data was overlaid on a top down map of the simulated town used in the experiment. Speed, selection, errors, as well as driving metrics were all encoded into this map. A line yellow, green, or red represented the driver's path and speed on it as either too slow, acceptable, or too fast, respectively. The times at which the user was selecting items was represented by surrounding the path line in blue. It was surrounded by light blue when the user should have been selecting an item, but had not started yet it and was depicted in dark blue after the user began to select the item, but had not finished.

Errors and driving metrics (e.g. driving off the road or collisions) were represented with a colored circle enclosing a letter to indicate the type of metric or error that occurred. These were placed where the incident occurred. Totals for the metrics collected were also shown in a list box to use as general numeric user performance scores.

To make the visual data analysis tool more useful, controls were used to select which data to view. This included specifying the intervals in the experiment the path should be depicted. The path, times of selection, as well as the metrics could also be shown or hidden individually. The background on which the paths and metrics were drawn could be changed from the actual map, to all roads, to just roads that were on the course the participant was required to follow. Individual participants could be selected to be viewed one at a time by a combo box that listed all participants in the experiment by input method.

Although the tool allowed the participant's test to be reconstructed, it did not reveal any more interesting results than the statistical analysis did. The lane exceedances and driving performance in relation to selection was completely visible. Some obvious trends were seen, such as when selection resulted in more erratic driving behavior, but these were repeated throughout the three input methods.

These results, indicating rather similar performance between the input methods, were by no means expected. Significant differences were expected to be found between the input methods, particularly between the graffiti based method and the other two input methods, since they varied greatly on approach. Although all three input methods could be used as visual-manual methods, graffiti characters can be gestured and used to select items without ever looking at the screen. The remote and touch screen requires at least quick glances to scroll and select the proper list item, as well as develop an orientation to the current place in the list. The remote control offers portability and tactile feedback over the other two. All these factors were expected to contribute to more varied results in some way. Possible reasons for these differences are suggested in the following discussion section.

8.2 Reaction Time for Selecting Items

Support for reaction was measured by how quickly a list item could be selected by a user after it was displayed on the simulator screen. Like interruption, the collected data revealed no difference between the input methods.

8.3 Comprehension of Selected Items

Six questions were asked immediately following the test portion of the study to gauge how well the participants remembered the information (list items) encountered in the system. Most questions were on list items that were not selected, but should have been seen during the process of selecting list items. The questions were similar to the following: "How many forms of weather were in the weather category?" and "Which cereal was NOT an item?" Participants did particularly bad on these types of questions; less then 20% of the questions were answered correctly. The participants were able to answer the questions about the items they selected (Question 6) and about items all within the same group (Question 1). However, no input method showed a significant difference on the answers received from participants. The 10% difference between the touch screen and other two methods appeared different, but t-tests were not significant.

The comprehension questions did, however, reveal an interesting thing about the list selection task by the varied performance of the different types of questions. Five of the six questions asked about items that users may have seen while selecting another item, the sixth question asked about the items that were actually selected. More correct answers were received on this last question then any of the other questions, indicating greater comprehension on items that are goal oriented (Question 6).

Table 13 – Results of the survey questions on comprehension asked to the usability study participants. The actual questions that were asked can be found in appendix A.6. The results are in terms of the percentage of the 11 participants for each input method who answered the particular question right. The blue column indicates the goal oriented questions.

	1	2	3	4	5	6	Totals
Graffiti	27%	9%	0%	9%	27%	**82%**	25.6%
Touch screen	64%	18%	0%	18%	9%	**100%**	34.8%
Remote	38%	8%	8%	8%	15%	**69%**	24.3%

8.4 Discussion of Results

The similar performance of all the input methods did not reveal one to be superior over the others for the particular notification task, even though the claims made about the methods indicated tradeoffs that should have been apparent in the study. This discontinuity has a few possible explanations. First, the study may not have adequately measured the attention of the primary task or the completion of secondary task user goals. This can be at least partially ruled out, since the study was effective in revealing that the notification task caused more distraction when used then when not. Second, changes in the input method for the studied notification task have a much more subtle effect on overall system performance than was previously assumed. This does necessarily support that the statement that the performance of the three input methods will always be the same for every notification system, but it may indicate that the three are much more dependant on the particular operation of the secondary task than was previously thought.

8.4.1 Suggestion and Considerations for Future Systems

Suggestions and considerations based on the initial claims and on the results of the usability study will now be described.

- *Comprehension is not significantly affected by differences in input methods that deal with the same data.* This fact was indicated by the similar usability study results that indicated that spelling a partial item using the graffiti method did not hinder nor help remembering item names over the scroll and select methods.
- *Users retain information pertaining to their goal much more frequently than information encountered on the way to their goal.* Results of the comprehension questions revealed that users could answer more questions about items they were required to select then those that they were not, but may have seen.
- *List selection should be augmented with little or no visual feedback or confirmation of selected items, when used with a non-visual input method.* The graffiti input method was chosen largely because it offered low visual interruption compared to the other methods. The primary task was also a large consumer of the user's visual resources; however, the list selection task still showed the list on the screen and gave users the ability to view the contents of the database. Although this may be a useful feature, it comes at the expense of the user's visual resources.
- *Touch screen controlled interfaces should use changing state to lessen visual demands of the system.* The touch screen interface was implemented to be controlled in much the same way as the remote control interface, without taking advantage of its ability to offer a better interface based on the state of the system. With the list selection task modeled in the usability test, list items in categories

are fewer than categories themselves. The touch screen should take advantage of this and offer a different interface to work better with fewer items when choosing a song from the artist.

- *Audio should augment list selection when the task is performed in a visually demanding environment.* Analyzing the table of resource usage for the input methods revealed that none were using audio resources, even though the primary task was not a large consumer.

Chapter 9: Conclusions

The application of computer technology to areas less and less like the typical desktop workstation has created a need for input methods that can be used with ongoing tasks. The usability study conducted exemplifies the type of research and problems that need to be addressed in order for notification systems to be designed that allow operation in real-world environments. The proposed evaluation incorporated an input method analysis that focus the design process on the critical parameters that probe the specific needs imposed by notification systems and provide guidance for design recommendations.

The contributions of this thesis can be summarized as the following:

- Three types of notification systems (IVIS, Mobile, and Ubiquitous) were identified as significantly effected by input; this was followed by a case study in the IVIS domain that demonstrated the need for better designed input methods for these types of systems.
- Three different input methods were developed for use in vehicles.
- Indications suggested that notification systems may benefit more from input methods that are matched with the user goals of the secondary task rather than by developing input methods that meet user goals of a secondary task.
- A framework was proposed to allow the design of an input method to match the needs of the notification system and allow parameters of the input method to guide design of the notification system to be optimized for a relevant input method. This will help with finding relevant input methods to match a

notification system need or modifying a notification system to take advantage of the upsides and downsides of an available input method.

- A reusable testing platform and analysis tool was developed that should be reusable for future experiments.

9.1 Future Work

This thesis described a design analysis method for notification systems, it compared three different input methods under this analysis and indicated possible design recommendations for them. Implementation and further iterative study of these methods will only reveal if these recommendations were correct. Additionally, the proof of this design analysis's utility is potentially limited to the single notification system and input methods tested. However, it is expected that a similar process would be applicable to other types of related systems, such as those in mobile or ubiquitous computing. It is hoped that future studies will further test the design analysis and suggest problems with the method and offer improvements that will extend and refine the applicability of it to other domains and input methods. Specifically what benefits do input that are compatible with interruption, reaction, and comprehension have.

Also, other input methods, particularly voice based ones, were highlighted as offering a lot of promise for notification systems, but were not formally examined. Specifically VoiceXML is an emerging standard that offers great potential for in-vehicle information systems and notification systems in general (http://www.voicexml.org), however, the cognitive demands and tradeoffs made for using the interaction technique are not well understood yet. The results of the usability also indicated that goal oriented tasks provide more comprehension then non-goal oriented tasks. General applicability of this concept seems likely, but should be verified by future studies.

Finally, the software (test walkthrough and simulator) constructed for the usability study was built with a strong emphasis on reusability; it is hoped that future studies will take advantage of this work and expand upon their functionality to lessen the cost of constructing a usability study and standardize the process.

9.1.1 Secondary Input Benchmarking Tasks

The secondary task examined in depth a particular task of list selection. This task is common throughout many different domains and offers a common challenge for designers of a wide variety of systems. McCrickard suggested that focusing research on a specific set of benchmark tasks with common goals will benefit construct validity and expedite research progress (McCrickard, Catrambone et al. 2003). The task of list selection is offered up as such a task that other designers can leverage and build upon in future studies, additionally, the following other tasks are also as offering a specific input problem.

Table 14 – Benchmark tasks for comparatively rating input methods.

Benchmark Task	Description	Desktop Example
Point and Click	Select a location from a two dimensional coordinate space.	Selecting an X,Y position on the screen.
Slider	Selection from a one dimensional space.	Selecting a continuous value from a slider control.
Text Input	Enter an undefined set of characters into a system.	Typing into an edit box.
List Selection	Choose an item from a predefined database of terms.	Selecting an item from a listbox.
Simple function	The complete execution of a single application function.	Keyboard function keys or bound key

	combinations (i.e. Ctrl + C) that perform some action.

9.2 Closing

Growing popularity of the information systems—noted to be in demand of human interaction presents a wealth of challenges for designers of notification systems. Simple performance metrics will reveal the answer to some of these problems, but only after thorough and scientific analysis of all the elements within the systems will the answers to the more complex questions be revealed. The evaluation methodology proposed attempts to do just that; however, further study and an in depth awareness of notification systems in terms of universally accepted parameters will allow this goal to be reached.

Appendix A: Usability Study Support

A.1 Informed Consent Form

Informed Consent for Participant of Investigative Project

Virginia Polytechnic Institute and State University

Title of Project: *Input Methods for Improving Secondary Task Performance*

I. THE PURPOSE OF THIS RESEARCH/PROJECT

You are invited to participate in a study of input methods for improving secondary task performance. You will be testing a specific input method that can be used for controlling tasks occurring the periphery. This study involves experimentation for the purpose of evaluating and improving the user interface and the quality of information provided by the system.

II. PROCEDURES

You will be asked to perform a set of tasks using the input method that will be explained to you. These tasks consist of finding an item in a list while driving a simulator. Your role in these tests is that of evaluator of the software.

We are not evaluating you or your performance in any way; you are helping us to evaluate our system. All information that you help us attain will remain anonymous. Your actions will be noted and you will be asked to describe verbally your learning process. You may be asked questions during and after the evaluation, in order to clarify our understanding of your evaluation. You may also be asked to fill out a questionnaire relating to your usage of the system.

The session will last about 30 minutes. There are no risks to you. The tasks are not very tiring, but you are welcome to take rest breaks as needed. If you prefer, the session may be divided into two shorter sessions. You may also terminate your participation at any time, for any reason.

III. RISKS

There are no known risks to the subjects of this study.

IV. BENEFITS OF THIS PROJECT

Your participation in this project will provide information that may be used to improve input methods and how they are evaluated. No guarantee of benefits has been made to encourage you to participate. You may receive a synopsis summarizing this research when completed. Please leave a self-addressed envelope with the experimenter and a copy of the results will be sent to you.

You are requested to refrain from discussing the evaluation with other people who might be in the candidate pool from which other participants might be drawn.

V. EXTENT OF ANONYMITY AND CONFIDENTIALITY

The results of this study will be kept strictly confidential. Your written consent is required for the researchers to release any data identified with you as an individual to anyone other than personnel working on the project. The information you provide will have your name removed and only a subject number will identify you during analyses and any written reports of the research.

VI. COMPENSATION

Your participation is voluntary and unpaid.

VII. FREEDOM TO WITHDRAW

You are free to withdraw from this study at any time for any reason.

VIII. APPROVAL OF RESEARCH

This research has been approved, as required, by the Institutional Review Board for projects involving human subjects at Virginia Polytechnic Institute and State University, and by the Department of Computer Science.

IX. SUBJECT'S RESPONSIBILITIES AND PERMISSION

I voluntarily agree to participate in this study, and I know of no reason I cannot participate. I have read and understand the informed consent and conditions of this project. I have had all my questions answered. I hereby acknowledge the above and give my voluntary consent for participation in this project. If I participate, I may withdraw at any time without penalty. I agree to abide by the rules of this project

____ _______________

Signature Date

____ ______________________________

Name (please print) Contact: phone or address or

____

email address (OPTIONAL)

To the participant/subject: Should you have any questions about this research or its conduct, you may contact:

Investigator: Charles Holbrook Phone (540) 552-9467
Graduate student, Department of Computer Science
email: cholbroo@vt.edu

Instructor:	Dr. Scott McCrickard Phone (540) 231-6075 Assistant Professor, Computer Science Department (231-6931) email: mccricks@vt.edu

X. EXPERIMENTERS' RESPONSIBILITIES

As the person(s) responsible for conducting the empirical study described in this form, we agree that we are response for protecting participant/subject rights and meeting the conditions described here, especially for erasing any videotapes by the date specified.

______________________________________ ______________________________ ____
______ Name (please print) Signature
Date

A.2 Graffiti Scenario

Ed Driver's new car is equipped with an MP3 carputer music system. He is currently driving doing the speed limit of 45 mph. The bus in front of him has a sign for the new CD by his favorite artist, John Doe, that Ed just uploaded to his in car system and now wants to hear John Doe's new hit single, but he cannot quite remember the name. Ed gestures on the touchpad mounted in the center console of his car to make the Graffiti symbol for the letter J. This highlights the artist Jeep, so Ed continues by gesturing an O, the next letter in the desired artist's name. This selects the artist John Doe and brings up the list of songs for him. However, while looking at the song list on the carputer's screen located in the passenger side of the car, Ed sees the brake lights for the bus in front of him come on and his attention is brought back to the primary task of driving. After adjusting his speed to match that of the bus, he glances back at his screen, sees the name of the new single and remembers that it is called Sing. Ed now gestures an S and since it is the only song by John Doe starting with an S, it is selected it and begins playing.

A.3 Touch screen Scenario

Ed Driver's new car is equipped with an MP3 carputer music system. He is currently driving doing the speed limit of 45 mph. The bus in front of him has a sign for the new CD by his favorite artist, John Doe, that Ed just uploaded to his in car system and now wants to hear John Doe's new hit single, but he can't quite remember the name. Ed first shows the list of available artists by tapping on an open area of the screen. He then scrolls down to the artist John Doe by pressing down on the image of a down arrow on the screen until he sees the entry for John Doe become highlighted, he then presses the select button which brings up the list of songs for the artist. However, while looking at the song list on the carputer's screen located in the passenger side of the car, Ed sees the brake lights for the bus in front of him come on and his attention is brought back to the primary task of driving. After adjusting his speed to match that of the bus, he glances back at his screen, sees the name of the new single and selects it by simply pressing on the entry and it begins playing.

A.4 List Categories and Item Participants chose From

ToothPastes
- Colgate
- Crest
- AquaFresh
- Rembrant

Candy
- Snickers
- Milky Way
- Reeses Pieces
- Twix
- Mars
- Skittles
- StarBurst

Money
- Dollar
- Quarter
- Nickel
- Dime
- Penny

Dogs
- Akita
- Alopekis
- Boxer
- Dalmatian
- Cocker Spaniel
- Eskimo Dog
- Fox Terrier
- German Shepherd
- Golden Retriever
- Hawaiian Poi Dog
- Irish Setter
- Jack Russell Terrier
- Kangaroo Dog
- Lapphunds

- Miniature Poodle
- Norfolk Terrier
- Otterhound
- Poodle
- Queensland Heeler
- Rottweiler
- Schnauzers
- Toy Poodle
- Whippet
- Xoloitzcuintle
- Yugoslavian Hounds

Insects
- Fly
- Mosquito
- Beetle
- Moth
- Butterfly
- Wasp
- Bee
- Ant
- Grasshopper
- Mantid
- Aphid
- Cicada

Colors
- Red
- Yellow
- Orange
- Green
- Pink
- Purple
- Blue
- Black
- White
- Maroon
- Magenta
- Indigo
- Violet
- Fushia

Sports
- Cricket
- Football
- Golf
- Soccer
- Tennis
- Rugby

- Ping Pong

Languages
- French
- German
- Italian
- English
- spainish

Music
- Blues
- Soul
- Rock
- Jazz
- Pop
- Classical

Movies
- Star Wars
- Aliens
- Blade Runner
- Blues Brothers
- It
- The Terminator
- Indiana Jones
- Gone with the wind

Programming Languages
- Java
- Perl
- C
- COBAL
- C Sharp
- Lisp
- Pascal
- Visual Basic
- Delphi

United States Presidents
- George Washington
- Abraham Lincoln
- George Bush
- Ronald Reagan
- John Adams
- Chester Arthur
- James Buchanan
- Jimmy Carter
- Grover Cleveland
- Bill Clinton
- Calvin Coolidge
- Dwight Eisenhower

- Millard Fillmore
- Gerald Ford
- James Garfield
- Ulysses Grant
- Warren Harding
- Benjamin Harrison
- William Harrison
- Rutherford Hayes
- Herbert Hoover
- Andrew Jackson
- Tomas Jefferson
- Anderw Johnson
- Lyndon Johnson
- John Kennedy
- James Madison
- William McKinly
- James Monroe
- Richard Nixon
- Franklin Pierce
- James Polk
- Franklin Roosevelt
- Theodore Roosevelt
- William Taft
- Zachary Taylor
- Harry Trueman
- John Tyler
- Martin Van Buren
- Woodrow Wilson

Rivers
- Mississippi
- Yangtze
- Amazon
- Danube
- Nile
- Yellow
- Congo
- Rio Grande
- Tigris
- Euphrates
- Ganges
- Rhine

Tools
- Wrench
- Rachet
- Screw Driver
- Hammer

- Saw

Countries
- United States
- Paraguay
- Mexico
- Canada
- France
- Spain
- England
- Russia

Continents
- North America
- South America
- Asia
- Europe
- Austrailia
- Antarctica
- Africa

Weather
- Sunny
- Cloudy

Board Games
- Yatzee
- Chess
- Checkers
- Shoots and ladders

Dinosaurs
- Tyranosaurus Rex
- Brontosaurus
- Diplodicaus

Flowers
- Rose
- Petuinia
- Daisy
- SunFlower
- Tulip
- Dogwood
- Impatients

Animals
- Dog
- Cat
- Fish
- Horse
- Bat
- Bird

Cereals

- Corn Flakes
- Rasin Bran
- Cinnamon Toast Crunch
- Cheerios
- Life
- Wheaties
- Apple Jacks

Jewelery
- Ring
- Necklace
- Bracelet
- Anklet
- Earring

Gum
- Wrigleys
- Trident
- Bubble Yum
- Juicy Fruit
- WinterFresh
- BigRed
- Extra
- DoubleMint

Fruit
- Banana
- Apricot
- Apple
- Orange
- Peach

A.5 Entrance Survey Questions

The following questions were given to the participants before completing any other portion of the study.

What is your gender?

1. Male
2. Female

What is your major?

1. Computer Science
2. Other
3. Not Applicable

What year are you in college?

1. 1st
2. 2nd
3. 3rd
4. 4th
5. 5th
6. Graduate
7. Faculty
8. Not Applicable

How old are you?

1. 17 or younger
2. 18
3. 19
4. 20
5. 21
6. 22
7. Between 23 and 30
8. Between 30 and 40
9. Over 40

How would you rate your computer literacy?

1. 1 (Computer Novice)
2. 2
3. 3
4. 4
5. 5 (Very Proficient)

How familiar are you with notification systems?

1. I Don't really know what a notification system is
2. Somewhat familiar
3. Familiar

4. Very Familiar

What kind of stereo equipment do you have installed in your car?
1. I don’t have a car\stereo
2. Radio and\or Cassette Desk
3. Standard CD Player
4. MP3 CD Player
5. Other MP3 device

Do you use a remote control with your in car stereo?
1. Yes
2. No

Do you own or use a Palm Pilot or similar type of device with character recognition?
1. Yes
2. No

Are you familiar with the grafitti alphabet for the palm pilot, do you know how to write all the letters?
1. Yes
2. No

Do you regularly (one or more times a week) use a cell phone while driving?
1. Yes, but only with a handsfree device
2. Yes, using just the regular handset
3. No

If you were going to buy a new electronic device, which of the following would be the most important to you?
1. That it supports the latest features
2. That it gets its primary job done very well
3. That it gets its primary job done and is as cheap as possible

A.6 Exit Survey Questions

These questions were given to the participants after completing the evaluation portion of the study. The following paragraphs introduced them to the survey process and what was expected of them. The questions were meant to be mostly an objective measure of how the input method effected how well the users were able to complete the given task.

> *You have just completed the test portions of this experiment; you will now be asked a few questions about your experience meant to gauge the performance of the systems.*
>
> *Please try to answer the questions to the best of your ability and remember it is not you being tested, but rather the system and wrong answers tell us just as much as right ones do so do NOT guess randomly or select an answer that you did not learn from this evaluation, but if your at least pretty sure of an answer from performing the evaluation go ahead and select it.*

How many forms of Weather were in the weather category?

1. 0
2. 1
3. 2
4. 3
5. 4
6. 5
7. I don't know

What is an Alopekis?

1. A country
2. A cat breed
3. A dog breed
4. The Bolivian equivalent of an American dollar
5. An insect
6. A candy bar
7. I don't know

Which cereal was NOT an item?

1. Corn Flakes
2. Raisin Bran
3. Captain Crunch
4. Cinnamon Toast Crunch
5. Apple Jacks
6. Life
7. Wheaties
8. I don't know

Which of the following was NOT a category?

1. Gum

2. Fruit
3. Vegetables
4. Car
5. Jewelry
6. Money
7. Flowers
8. Dinosaurs
9. I don’t know

Which category was the word yellow listed under?
1. Color
2. River
3. Both
4. I don’t know

Which item were you NOT required to select?
Dollar
1. Wheaties
2. SunFlower
3. Canada
4. Moth
5. Trident
6. Brontosaurus
7. I dont know

Please write any comments you have on just the driving simulator, input method, or anything else about the evaluation that you may think might be helpful.

A.7 Metrics Monitored During the Simulator Training Session

Metric	Maximum Tolerance (>= for Failure)
Maximum Speed	35 mph
Making the car spin out while turning	2 times
Times exceeding 35 mph	3 times
Collisions	once
Driving off the road	3 times

A.8 List Items to be Selected During the Training Session

Item #	Category	Item
1	Colors	Red
2	Insects	Mosquito
3	Programming Languages	COBOL

Appendix B: Carputer Design Specifications

B.1 Graffiti Character Recognizer Algorithm

The algorithm for character recognition used for the touchpad interface was based very loosely on the XEmacs strokes matching algorithm (Bakhash 2003). The implemented algorithm was optimized to use a Symantec's touchpad to allow for absolute positioning and the ability to start stroke recording when the user placed a finger on the touchpad and recognize it as soon as it was lifted rather then based on mouse clicks. To ease recognition, increase accuracy, and decrease reaction time, the implementation for the carputer was designed to only match single stroke characters. Recording of the finger coordinates begins as soon as the user places a finger on the touchpad and continues while the user draws a symbol and ends as soon as the finger is lifted. All the points that were recorded are then sent to the algorithm which first runs a line filter function on it. This filter recognizes extremely linear sets of points that would not normalize well in both directions. This filter works by first retrieving the equation of the line that starts at the upper left point of the bounding rectangle and ends at the lower right point, the x value for every recorded point is then checked against the value for the point that lies on the generated line at the same y coordinate of the recorded point and if the points x values vary by more then 10% of the line length, the whole set of points is not considered a line. However, if it is considered a line, the slope of the generated line is then used to determine the type of line it is, horizontal, vertical, or diagonal and the direction of the line is determined by the start and end points. A special type of vector is then returned for each of these line types denoting it as such. If the set of points is not considered a line, the points are then normalized from the coordinate space of the touchpad to a 3x3 grid (Figure 25). The minimum and maximum x and y values from all the points recorded are used as the bounding region for this grid. Each position within the grid is labeled with a number starting at 0 and going up to 8 from left to right, then from top to bottom. Any consecutive point that falls into the same grid position as the previous point is discarded. A normalized vector of the recorded strokes is then constructed using all the points still left in the grid, each point is added to the vector in the same order that it was recorded from the touchpad. This vector is then returned for processing by the calling function. The vectors returned by the character recognizer can be used for any type of symbol recognition, however, better accuracy results with symbol sets that vary more from character to character.

References

Accot, J. and S. Zhai (2000) Scale Effects in Steering Law Tasks. Toulouse cedex, France, Centre d' ´ Etudes de la Navigation A´erienne.

AIM (2003). The America Online Instant Messenger System.

Alpern, M. and K. Minardo (2003). Developing a Car Gesture Interface For Use as a Secondary Task. Pittsburgh, Human-Computer Interaction Institute (HCII), Carnegie Mellon University: 2.

Anderson, D., A. Abdalla, et al. (2001). Distracted Driving: Review of Current Needs, Efforts, and Recommended Strategies. Fairfax, Virginia, George Mason University.

Bartram, L. R. (2001). Enhancing information visualization with motion. Canada, University, Canada.

Brewster, S. and A. Walker (2002). Non-Visual Interfaces for Wearable Computers. Glasgow, UK, Department of Computing Science, University of Glasgow: 4.

Brewster, S. A. and P. G. Cryer (1999). "Maximising Screen-Space on Mobile Computing Devices." Proceeding of the ACM CHI'99.

Butts, L. and A. CockBurn An Evaluation of Mobile Phone Text Input Methods. Christchurch, New Zealand, Human-Computer Interaction Lab, Department of Computer Science, University of Canterbury.

Card, S. K., J. D. Mackinlay, et al. (1991). "A Morphological Analysis of the Design Space of Input Devices." ACM Transactions on Information Systems, **9**(2): 99-122.

Carroll, J. M. (1999). Five Reasons for Scenario-Based Design. 32nd Hawaii International Conference on System Sciences, Hawaii.

Clausen, C. S. and J. E. Pedersen (1998). Mobile Text Input - a Scenario-based Assessment of Text Input Techniques for PDAs, Institute of Information- and Media Studies.

Collins, D. J. (1997). An Examination of Driver Performance Under Reduced Visibility Conditions when Using an In-Vehicle Signing Information System. Industrial and Systems Engineering. Blacksburg, VA, Virginia Polytechnic Institute and State University.

Enns, N. R. N. and I. S. MacKenzie (1998). Touchpad-Based Remote Control Devices. Toronto, Canada, University of Toronto.

Erskine, L. E., D. R. N. Carter-Tod, et al. (1997). "Dialogical techniques for the design of web sites." Int. J. Human–Computer Studies 47: 169-195.

Gallagher, J. P. (2001). An Assessment of the Attention Demand Associa ted with the Processing of Information for In-Vehicle Information Systems (IVIS). Computer Science. Blacksburg, VA, Virginia Tech.

Goldberg, D. and C. Richardson (1993). Touch-typing with a stylus. INTERCHI '93 Conference on Human Factors in Computing Systems, New York.

Graffiti (2003). The Graffiti System.

Grasso, M. A. (1996). Speech Input in Multimodal Environments: A Proposal to Study the Effects of Reference Visibility, Reference Number, and Task Integration. Baltimore, Maryland, Department of Computer Science and Electrical Engineering University of Maryland, Baltimore Campus.

Green, P. (1999). Estimating Compliance with the 15 Second Rule for Driver-Interface Usability and Safety. Proceedings of the Human Factors and Ergonomics Society 43rd Annual Meeting-1999. University of Michigan Transportation Research Institute, University of Michigan Transportation Research Institute.

Griffen, T. (2001). Usability Testing in Input Device Design. 2003.

HandyKey (2003). The Twiddler Website, HandyKey Corporation. 2003.

Hindus, D., B. Arrons, et al. (1995). "Designing Auditory Interactions for PDAs." Proceedings of the 8th ACM symposium on User interface and software technology: 143-146.

Hummels, C., G. Smets, et al. (1997). An Intuitive Two-handed Gestural Interface for Computer Supported Product Design. International Gesture Workshop, Bielefeld, Germany.

Kawachiya, K. and H. Ishikawa (1997). ScrollPoint: An Input Device for Mobile Information Browsing. Kanagawa 242, Japan, IBM Research, Tokyo Research Laboratory.

Kidd, C. D., R. Orr, et al. (2000). The Aware Home: A Living Laboratory for Ubiquitous Computing Research. Atlanta, GA, College of Computing and GVU Center, Georgia Institute of Technology.

Kurtenbach, G. and E. A. Hulteen (1990). "Gestures in Human-Computer Communication." The art of human Computer Interface Design: 309-317.

Lee, J. D., B. H. Kantowitz, et al. (1994). Functional description of advanced in-vehicle information systems: Development and application, Proceedings of the First World Congress on Applications of Transport Telematics & Intelligent Vehicle-Highway Systems. Artech House. Boston, MA**:** 2369-2376.

MacKenzie, I. S. and R. W. Soukoreff (2002). "Text Entry for Mobile Computing: Models and Methods, Theory and Practice." Human-Computer Interaction 17: 147-198.

MacKenzie, I. S. and S. Zhang (1997). "The Immediate Usability of Graffiti." Proceedings of Graphics Interface '97: 129-137.

Mäkelä, K., E.-P. Salonen, et al. (2001). Evaluating the User Interface of a Ubiquitous Computing System Doorman. Finland, Tampere Unit for Computer-Human Interaction (TAUCHI)U iversity of Tampere.

Manes, D. and P. Green (1997). Evaluation of a Driver Interface: Effects of Control Type (Knob Versus Buttons) and Menu Structure (Depth Versus Breadth). Dearborn, MI, The University of Michigan Transportation Research Institute (UMTRI)**:** 93.

Mankoff, J. and G. D. Abowd (1998). Cirrin: A word-level unistroke keyboard for pen input. Atlanta, GA, GVU Center, College of Computing, Georgia Institute of Technology.

Masui, T. (1999). POBox: An Efficient Text Input Method for Handheld and Ubiquitous Computers. Shinagawa, Tokyo, Sony Computer Science Laboratories, Inc.

McCrickard, Scott, Richard Catrambone, C. M. Chewar, and John T. Stasko (2003). "Establishing Tradeoffs that Leverage Attention for Utility: Empirically Evaluating Information Display in Notification Systems." *International Journal of Human-Computer Studies*, volume 8, issue 5, May 2003, pages 547-582.

McCrickard, Scott. (2000). Maintaining information awareness in a dynamic environment: Assessing animation as a communication mechanism. Computer Science. Atlanta, GA, Georgia Institute of Technology.

Microsoft (2002). Microsoft Speech - Technology Overview, Microsoft. 2003.

mIRC (2003). mIRC.

Moldenhauer, Maxim and Scott McCrickard (2003). "Effect of Information Modality on Geographic Cognition in Car Navigation Systems." Short paper in *Proceedings of the IFIP TC.13 Conference on Human-Computer Interaction (INTERACT 2001)*, Zurich Switzerland, September 2003.

Myers, B. A., R. C. Miller, et al. (2000). Individual Use of Hand-Held and Desktop Computers Simultaneously. Pittsburgh, PA, Human Computer Interaction Institute School of Computer Science Carnegie Mellon University.

Nowakowski, C. and P. Green (1998). Map Design: An On-the-Road Evaluation of the Time to Read Electronic Navigation Displays. Ann Arbor, MI, The University of Michigan Transportation Research Institute (UMTRI).

Nullsoft (2003). WinAmp MP3 Player Software, NullSoft.

Pavlovych, A. and W. Stuerzlinger Less-Tap: A Fast and Easy-to-learn Text Input Technique for Phones. Ontario, Canada, Department of Computer Science York University.

PioneerElectronics (2003). Pioneer CD-VC60 Voice Commander, Pioneer. 2003.

Rekimoto, J. (2001). GestureWrist and GesturePad: UnobtrusiveWearable Interaction Devices. Shinagawa-ku, Interaction Laboratory Sony Computer Science Laboratories, Inc.

Scholtz, J., M. Burnett, et al. (2002). User-Centered Evaluation of Ubiquitous Computing Enviorments.

SGI (2003). OpenGL Performer: Home Page, SGI. 2003.

Shimpi, Anad Lal. "Windows XP Media Center Edition: Exposed." January 8, 2003. <http://www.anandtech.com/showdoc.html?i=1766&p=26>

Shneiderman, B. (2002). "The Limits of Speech Recognition." Communications of the ACM 43(9).

Silfverberg, M., I. S. MacKenzie, et al. (2000). Predicting Text Entry Speed on Mobile Phones. Finland, Nokia Research Center.

Stifelman, L. J., B. Arons, et al. (1993). "VoiceNotes: A Speech Interface for a Hand-Held Voice Notetaker." Proceedings of INTERCHI: 179-186.

Tijerina, L., E. Parmer, et al. (1998). "Driver Workload Assessment of Route Guidance System Destination Entry while Driving: A Test Track Study." Proceedings of the 5th ITS World Congress.

Tsimhoni, O., D. Smith, et al. (2002). Destination Entry while Driving: Speech Recognition versus a Touch-Screen Keyboard. Ann Arbor, The University of Michigan Transportation Research Institute (UMTRI).

Turner, S. G. (1998). “A Case Study Using Scenario-Based Design Tools and Techniques in the Formative Evaluation Stage of Instructional Design: Prototype Evaluation and Redesign of a Web-Enhanced Course Interface”. Instructional Technology. Blacksburg, Virginia Tech.

UMTRI (2003). University of Michigan Transportation Research Institute. 2003.

VirtualTechnologies (2003). CyberGlove, Virtual Technologies. May 17, 2003.

Ward, D. J. (2001). Adaptive Computer Interfaces. Cambridge, University of Cambridge.

Zhai, S., B. A. Smith, et al. (1997). "Improving Browsing Performance: A Study of Four Input Devices for Scrolling and Pointing Tasks." Proceedings of INTERACT '97: The 6th IFIP Conference on Human-Computer Interaction: 286-292.

Zhao, Q. A. and J. T. Stasko (2000). "What's Happening? the community awareness application." Conference Companion of the ACM Conference on Human Factors in Computing Systems (CHI '00). 253-254.

Vita

Chuck Holbrook has been developing software for in-vehicle information systems for three years prior to this document. He completed his Bachelor in Computer Science at Virginia Polytechnic Institute and State University in May 2002. He continued on there in order to complete a Masters in Computer Science. He is currently employed as a developer at Lockheed Martin.