VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Computer Science

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@mcs.vuw.ac.nz

Semiotic Explorations in User Interface Design

Jennifer Ferreira

Supervisors: James Noble, Robert Biddle

Submitted in partial fulfilment of the requirements for
Bachelor of Science with Honours in Computer Science.

Abstract

The user interface is a complex but fascinating phenomenon. There are signs to be discovered with semiotic tools that provide new and different perspectives on relevant user interface issues such as design, redesign and evaluation. This report focuses on the valuable insights that semiotic analysis can offer researchers in these fields.

Acknowledgments

My thanks go to

- **James Noble** and **Robert Biddle** for being super supervisors
- **Tim Wright** for helpful comments and advice
- My COMP311 group: **Daniela Mogin, Shane Morton** and **Mariam Yousif** for allowing me to refer to the work we did as part of the course.
- All the willing and eager volunteers who made the heuristic evaluation and the icon intuitiveness test for this report possible

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This project covers a very broad range of topics that challenge user interface designers and researchers today.

A major challenge is the nature of design. To design is to create, but it is not a rigorous activity that can be guided by strict rules and conventions. Thus, it is not an easy process, especially for user interface design. Designing the user interface is a creative activity and yet user interfaces are required by users for specialised and highly structured tasks. Therefore, user interface design processes face the task of attempting to impose structure on an inherently chaotic activity, while still allowing enough opportunity for user interface designers to produce novel but usable solutions. Usage-Centered Design has become a widely accepted user interface design process and has been successful in a number of projects of varying sizes. It is an attractive research opportunity to attempt to determine what makes this process so successful.

An approach that has started making more headway into all facets of computer science is semiotics — the study of signs. From a semiotic perspective, we can view the user interface as a collection of signs, each designed to convey an intended message. The user interface designer can be viewed as an *encoder* of signs and the user of the interface, as the *decoder*. It is clear that both the encoder and the decoder must agree on the system they are using in which to communicate, or else problems with interpretations of the message are likely to arise. Semiotics allow us to model user interface elements and study them as signs. Some kinds of signs are easier to interpret than others. It has been the goal of this research to try and determine how users interpret signs, why signs used in the user interface can be problematic and why others are so successful.

This project applied semiotic analysis to various aspects of a user interface and user interface design. We explored different types of signs and how redesigning an interface impacts on the type of the new sign. During the course of the year, we also conducted an icon intuitiveness study in order to learn how users interpret what they see, and we conducted a heuristic evaluation. Both these studies provided valuable insights into users and the their relationships with the computer systems they use.

Road Map

This report begins by giving an overview of the discipline of semiotics in chapter 2. Here we introduce the appropriate models and justify why applying a Peircean framework of semiotics is suitable for investigating user interface design issues. In chapter 3, we introduce two interesting studies about icon interpretations and then report the results of an icon intuitiveness test of our own. Chapter 4 investigates the semiotics of user interface redesigns, i.e., how the semiotics of a sign changes as the sign is redesigned in the interface. There is no evidence in the literature to suggest that the discussion in chapter 5 has been attempted before. This chapter explains the Usage-Centered Design process in terms of semiotics. Finally, our conclusions are presented in chapter 6 and any criticisms that need to be considered, are presented here also. Appendix A contains the paper that was accepted for publication by the Australian User Interface Conference 2005. Appendix B contains the documents for the icon intuitiveness test we carried out. Appendix C contains the documents for the heuristic evaluation we carried out and appendix D reports the results. Appendix E contains some earlier work done during the early stages of (mis)understanding semiotics.

Chapter 2

Semiotics

2.1 Introduction

This chapter presents a very basic introduction to semiotics. The most important semiotic concepts that will be referred to throughout the rest of the report are introduced, and we also include a discussion on how semiotics applies to the user interface.

2.2 A Definition

Semiotics is the doctrine of *signs*. The sign is the most important building block of semiotic study and it is defined as anything that stands for something else to some interpreter [42]. Hence, signs can be words, gestures, sounds, pictures, scents etc. Yet, a sign is not a sign unless it is interpreted as such [33] — a sign must be something that *means* something to someone. Semioticians break the sign up into parts and study the way these parts interact to form meaning.

2.3 Semiotic Models

The ancient Greeks from the 6th century BCE are the earliest recognised group who contributed to the theory of signs [29]. The two major figures in the recent history of semiotics, from which the modern European and American traditions are derived, are the Swiss linguist Ferdinand de Saussure (1857–1913)[1] and the American scientist and philosopher Charles Sanders Peirce (1839–1914).

2.3.1 The Saussurean Model

Ferdinand de Saussure's dyadic model maintains that a sign is a result of the union of a *signifier* — the form which a sign takes — and a *signified* — the concept it represents. Being interested in units and rules that combine them into meaningful systems, Saussure was the first to attempt to describe the *structure* of language, as opposed to describing its history or form [29]. His theories continue to be successfully applied today in many fields such as

[1]In fact, Saussure coined the term *semiology* to indicate the discipline of the study of signs according to his approach [48]. Both the terms 'semiotics' and 'semiology' originate from the Greek word for sign: *semeion* [49].

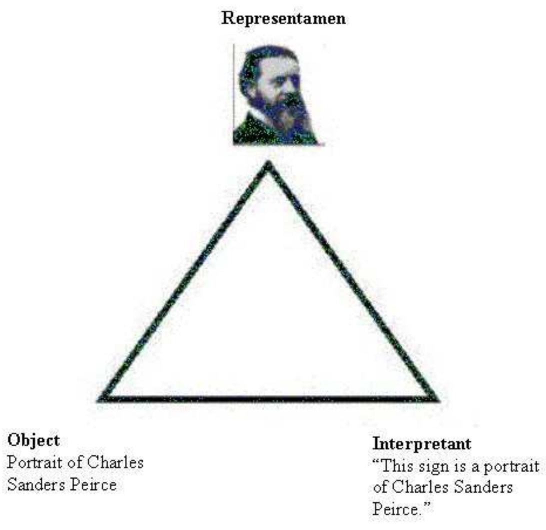

Representamen
Object
Portrait of Charles Sanders Peirce
Interpretant
"This sign is a portrait of Charles Sanders Peirce."

Figure 2.1: Peircean model: iconic sign

social studies, linguistics and religion. The Saussurean model will not be further discussed here, since we find the Peircean model (outlined below) to be more appropriate for our analysis.

2.3.2 The Peircean Model

In contrast to Saussure, Peirce's model consists of a triadic relationship containing: the *representamen*, the *object* and the *interpretant* (see figure 2.1). The representamen stands to somebody for something in some respect or capacity. It addresses somebody and creates in the mind of that person an equivalent, or perhaps more developed sign. The object is the actual thing the sign stands for [42]. The interpretant is therefore the sign created in the mind of the perceiver or the reaction caused by the object in the perceiver [4].

Peirce classified signs into thousands of categories, but acknowledged that the three most fundamental sign divisions are the *icon*, *index* and *symbol*. The category a sign belongs to depends on the relationship between the object and the representamen.

If the representamen resembles, or in some way imitates, the object then the sign can be interpreted as iconic. See figure 2.1: Here the representamen resembles the portrait of Charles Sanders Peirce and the perceiver of the sign can interpret this as such precisely because the representamen resembles him enough to be recognisable. According to the triadic model, this sign is only fully formed when the perceiver (interpreter) interprets the sign as standing for a portrait of Charles Sanders Peirce.

Indexical signs exist because of a causal relationship between representamen and object. In this case the sign does not represent its object but the representamen creates a link between it and the object in the mind of the perceiver. In figure 2.2, the time display on the wrist watch is an index of the time of day because the perceiver must perform a referential action: the time displayed on the watch must be understood as referring to the time of day.

If the relationship between the object and the representamen is a purely conventional one that must be learned by the perceiver, then the sign is symbolic. An example is given in

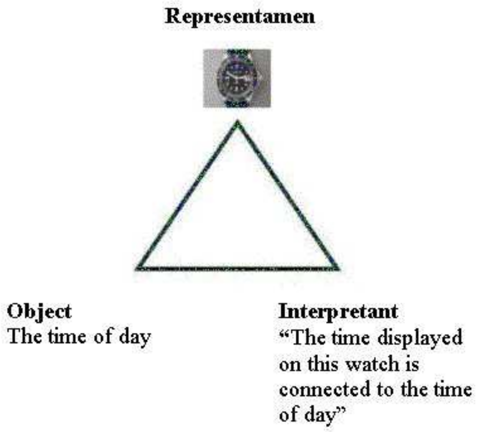

Representamen
Object
The time of day
Interpretant
"The time displayed on this watch is connected to the time of day"

Figure 2.2: Peircean model: indexical sign

figure 2.3. It is by learning to associate this symbol with a place where information can be obtained, that this specific interpretant is generated in the mind of the perceiver when this sign is encountered. Learning is necessary because there is nothing in the representamen that resembles or allows the perceiver to infer the notion of information. We can say that the relationship between the representamen and the object is arbitrary. At this point it must be stressed that the three divisions are not mutually exclusive. Most signs contain elements of iconicity, indexicality and symbolism in varying measures. It is very rare, and some argue impossible, to find signs in the real world that belong to solely one division. A well known example of a sign that can belong to all three categories is the photograph. While it is an icon in that it looks like the objects it represents, it is also an index of light on photographic emulsion [14], which is a sign of an event that has taken place at some point in time. Lindekens [32] would argue that the photograph is symbolic, as the camera can never make an exact replica of events, due to technological constraints.

Another instance where signs may belong to more than one category is in the case of indexical signs. An indexical sign may not be able to be interpreted as such unless the iconic representation of the sign is understood. For example, if the perceiver is unable to identify a footprint as that of a human being, it would be impossible to go on to infer that the footprint is an index of human presence. However, it is notable that some relationship between the object and representamen will tend to dominate in the sign and then it can be said that the sign is primarily of that relationship which dominates.

2.4 Semiosis

For a sign to exist, it must consist of all three parts (the object, representamen and the interpretant) and the interaction between them is a process Peirce termed *semiosis*. He described this process as "an action, an influence, which is, or involves, a cooperation of three subjects, such as a sign, its object and its interpretant, this tri-relative influence not being in any way resolvable into actions between pairs." [42] Goguen sees semiosis as the construction of meaning [28]. Clearly, then semiotics is concerned with 'how' signs mean [11], instead of 'what' they mean.

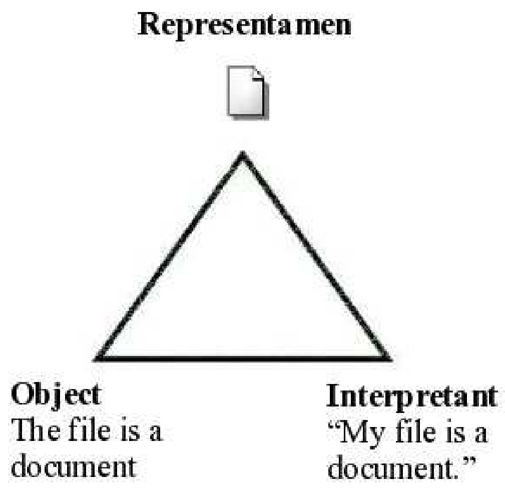


Figure 2.3: Peircean model: symbolic sign

Peirce did not name the relations between the three parts of the sign, but Barr [8] suggests a way of relating the object, representamen and interpretant that eases further discussion throughout this report. He proposes three relationships:

1. The *representation relation*, that occurs between the object and the representamen;
2. The *interpretation relation* that occurs between the representamen and the interpretant; and
3. The *matching relation* that occurs between the object and the interpretant.

Figure 2.4 taken from Barr's thesis [8] shows where the suggested relationships may appear on the Peircean triad.

To conclude this discussion of the theory of semiotics, we mention Eco's insistence that it is not enough that the sign simply represents something else. Semiosis must be allowed to be performed by the perceiver of the sign. Thus, for something to be a sign, it must adhere to two criteria : substitution and interpretability [26].

For the rest of this study, we use the Peircean model as a basis of the discussion, as several other authors have identified it as a good model for studying computer based signs and applied it successfully [35], [40], [8].

2.5 Semiotics and the User Interface

Semiotics is important to the general field of user interface design, since design is concerned with representation and semiotics provides tools for analysing these representations. The sign in the user interface is always an *intentional* sign, i.e., someone has created it in order to convey some message to the user. As Andersen notes, the designer builds the user interface so it can be used to tell people something [1]. So, the designer combines various signs to

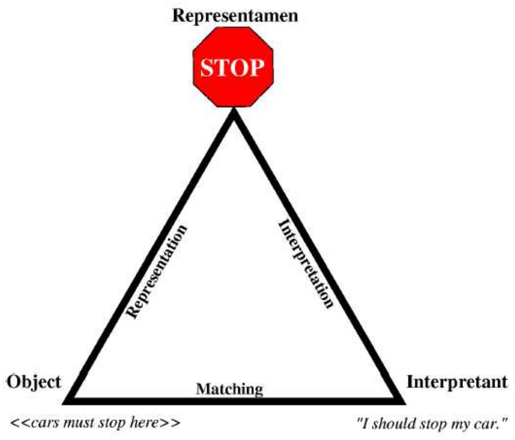


Figure 2.4: Barr's suggested relationships between the parts of the sign

make up the interface in order to convey its intended meaning to the user. Further, Nadin [36] maintains that to *design* means to structure systems of signs in such a way as to make possible the achievement of human goals, one of which is communication. The communication referred to here is that between the user and the designer [35].

The user interface can be seen as a complex sign made up of many smaller signs (buttons, scroll bars, images, etc.) all contributing to the process of communication, with each of the smaller signs having their own triadic relation. The representamen corresponds to the form the sign takes in the interface, the object corresponds to the underlying functionality of the sign and the interpretant corresponds to the sign generated in the mind of the user. This implies that users are required to guess at the object of the sign when interacting with the interface.

Due to signs in the interface being intentional signs as defined above, signs can be said to be successful when the user's interpretant matches the object of the sign, and unsuccessful otherwise []. This property allows us to evaluate the user interface, since the ideal interface would consist only of successful signs.

One potential problem with applying semiotic analysis to computer signs is imagining that all signs in the user interface are indexical, since all signs found in the interface necessarily have an underlying functionality. (This assumes that the interface is the most economic collection of signs that allows the user to perform all the tasks required.) Assuming indexicality is somewhat justified seeing as when the user activates[2] a sign in the interface this almost always results in some action on the part of the system — indicating a causal relation between the representamen and the object. But this would be ignoring the representation relationship [35] between these two; more specifically, the visual elements of the representamen and how this relates to what functionality it is signifying. An example is the document icon found in many desktop applications. Figure 2.3 shows the triadic relation between the representamen, object and interpretant of the document sign. Selecting this sign on the desktop results in a new document being created for the user to edit. Clearly there is a cause (the creation of a new document) and we may assume the sign is indexical, but when the visual elements of the representamen is considered in relation to its object, we realise that this sign

[2]Note that the user can *activate* a sign in various ways: single or double mouse click, keyboard input or any form of manipulation of what is presented in the user interface.

(the image of a paper based document) resembles the system concept of a document. Thus, it is an iconic sign. Only when this representational aspect is considered can the signs of the interface be classified as belonging to any of Peirce's three main divisions and not just as to the group of indexical signs.

2.6 Summary

This chapter briefly introduced semiotics as the study of signs and the different sign types as identified by Peirce. We explained that a sign must be both substitutable and interpretable for it to be called a sign. Finally we showed how user interface components can be seen as signs and warned of the danger of interpreting all computer based signs as being only indexical.

.

Chapter 3

User Interface Icons

3.1 Introduction

Graphical user interfaces make heavy use of icons to represent functionality required by users in performing their tasks. Icons are a popular method for visually representing functionality because they provide direct access (as opposed to functionality hidden away in menus), direct manipulation (one mouse click results in an action) and can save valuable user interface real estate. Studies have shown that icons are also faster and easier to recognise than text [16]. Good icon design should support the learnability and rememberability [21] of the user interface but of course, badly designed icons would have the opposite effect. Unfortunately icon design is not well understood and often usable icons are obtained simply by trial and error. This chapter presents a semiotic explanation for why some icons are likely to be better understood by users and others less so.

In Figure 3.1 from Schaffer and Sorflaten [46], it is easy to read the message. It is clear that all the pictographic images are iconic signs, i.e., they look like the things they are intended to represent. Yet this ease of interpretation of images does not always extend to computer icons. Therefore, if icons are to be used successfully in user interfaces they need to be well designed, but more importantly, thoroughly tested by the users. This chapter introduces an icon intuitiveness study done by Nielsen and Sano [39] (performed as part of the usability evaluation of Sun Microsystems' internal web), a study by Piamonte, Ohlsson and Abeysekera [44] of candidate icons for use in telecommunications software and an Icon Intuitiveness Test carried out by the authors. Our hypothesis was that users will interpret the icons as iconic signs and Eco [26] states that this is the case unless they already believe that it is appropriate to make further inferences about the icons. He further explains that "only if I already know the general rule which makes for 'if smoke, then fire' am I able to render the sensory datum *meaningful*."

3.2 Icon Semiotics

Using only icons (pictorial representations of functionality) in any application is discouraged but they can be a powerful and efficient way of communicating the underlying functionality to the user, provided they are well designed. Pictures are better in communication for some problems but worse for others. Thus it is not enough to rely only on pictorial representations.

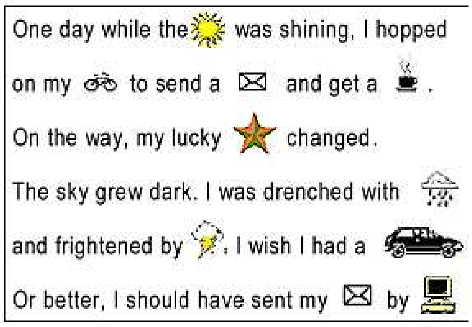

One day while the was shining, I hopped
on my to send a and get a .
On the way, my lucky changed.
The sky grew dark. I was drenched with
and frightened by . I wish I had a
Or better, I should have sent my by

Figure 3.1: Reading with pictures

The semiosis of using icons can be seen as consisting of three parts: the initial state, the process and the final state [2]. As illustration, we refer to a particular button found in the user interface of SAS/GRAPH v8.2 [50]. This button is an icon that the user can select in order to resize a graph. Supposing some user wishes to enlarge a graph, the initial state of this interaction consists of the application, e.g. SAS/GRAPH and the opened file in which the graph the user wishes to resize is located. Once the user has clicked on the resize graph icon, the final state is observed as an enlarged graph in the file the user was working on in the SAS/GRAPH application. The user can therefore, interpret the process as "I have enlarged my graph" although there is essentially a gap between the initial and final states that the user will never see. This gap is specified in a program and the user is required to make guesses at what this process does. Hence, the user creates an interpretant based on clues obtained from sensory input whereas the programmer intends an interpretant with the icon's design. If the user guesses correctly, i.e., creates the correct interpretant, then the final state is the desired final state. However if the user guesses wrongly, because the wrong interpretant is created in the mind of the user, the final state represents an undesired state. Andersen et. al. speak of this process as *interactional semiosis* [2, p 63].

Barr, Biddle and Noble [9], define an icon to be successful if the interpretant of the user matches the object that the designer had intended with that sign, and as unsuccessful when the user's interpretant does not match the designer's intended object. Hence, if the final state in Andersen's interactional semiosis is the desired state, then the sign used to obtain the final state is successful. Otherwise, if the final state is the undesired state, then the sign is unsuccessful.

3.3 SunWeb

In an icon intuitiveness study done by Nielsen and Sano [39], which was performed as part of the usability evaluation of Sun Microsystems' internal web, it was interesting to note the way the test users had interpreted the icons. The results can be seen in figure 4.6. In the study images were presented to users and then they were asked to indicate what functionality they thought the icons represented. For the most part the users seemed to respond to the question in a way that made it clear that they were interpreting the sign iconically. If the intended meaning of the icon is equated with the object of that sign and the Test User's Interpretation equated with its interpretant, then Peirce's triadic model can be reconstructed for each individual icon. Figure 3.4 is an example. Here the interpretation relationship between the Interpretant and the Representamen shows that the interpreter (in

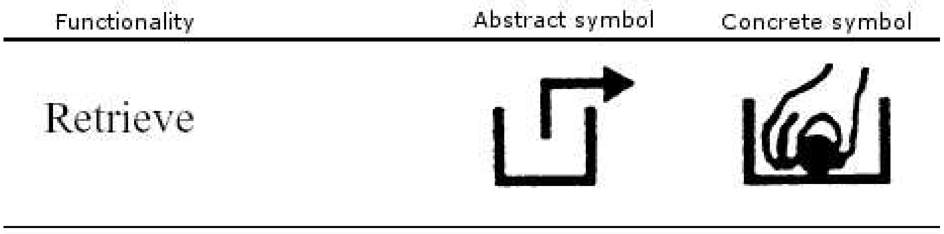
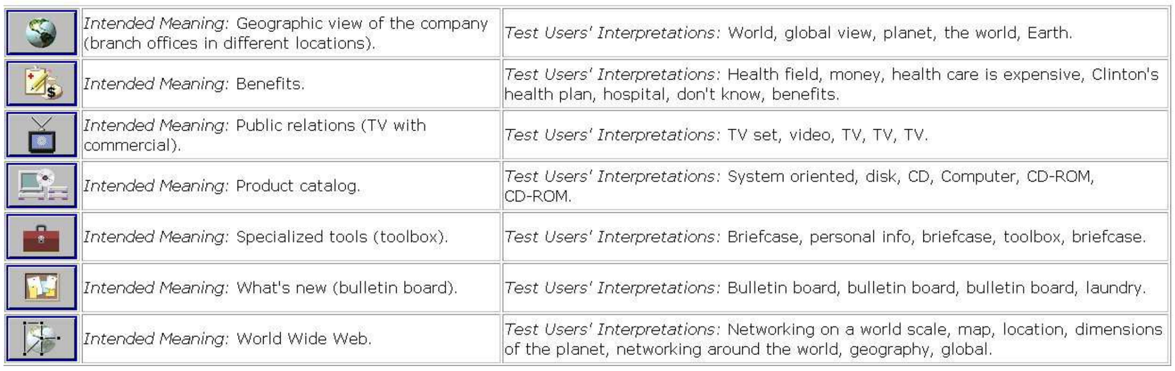


Figure 3.2: From Piamonte, Ohlsson and Abeysekera [44]: 'Abstract' and 'Concrete symbols' for the function 'Retrieve'



	Intended Meaning: Geographic view of the company (branch offices in different locations).	*Test Users' Interpretations:* World, global view, planet, the world, Earth.
	Intended Meaning: Benefits.	*Test Users' Interpretations:* Health field, money, health care is expensive, Clinton's health plan, hospital, don't know, benefits.
	Intended Meaning: Public relations (TV with commercial).	*Test Users' Interpretations:* TV set, video, TV, TV, TV.
	Intended Meaning: Product catalog.	*Test Users' Interpretations:* System oriented, disk, CD, Computer, CD-ROM, CD-ROM.
	Intended Meaning: Specialized tools (toolbox).	*Test Users' Interpretations:* Briefcase, personal info, briefcase, toolbox, briefcase.
	Intended Meaning: What's new (bulletin board).	*Test Users' Interpretations:* Bulletin board, bulletin board, bulletin board, laundry.
	Intended Meaning: World Wide Web.	*Test Users' Interpretations:* Networking on a world scale, map, location, dimensions of the planet, networking around the world, geography, global.

Figure 3.3: From Nielsen and Sano[39]: Results of the icon intuitiveness study

this case the Test User) expected the representation relationship between the Object and the Representamen to be iconic. However, it is not an iconic relationship but a symbolic one — the image of hardware (monitor, server and compact disc) does not look like a product catalog. Other icons with a high number of iconic interpretations were the icons representing the 'geographic view of the company', 'public relations', 'specialised tools' and 'what's new'.

3.4 Telecommunications

There has also been another study carried out by Piamonte, Ohlsson and Abeysekera [44]. They tested icons used in telecommunication-communication products that were divided into three sets of icons: what they call 'abstract symbols', 'concrete symbols' and 'proposed symbols'. The two types of symbols of interest are the 'abstract symbols', which correspond to Peirce's notion of symbolic signs and the 'concrete symbols', which correspond to the Peircean notion of iconic signs. Hence, the 'abstract symbols' were designed so that they did not look like the functionality they represented and the 'concrete symbols' were designed so that they represented their underlying functionality. Figure 3.2 shows an example of the 'abstract' and 'concrete symbols' for the function 'Retrieve'. Their study showed that participants were better at recognising the 'concrete symbols' than the 'symbolic symbols'.

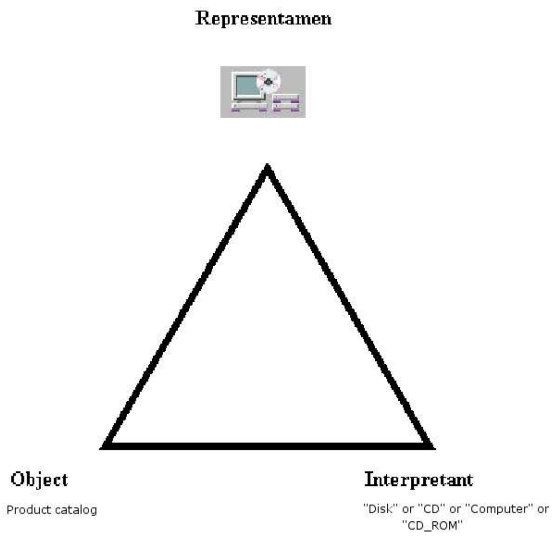


Figure 3.4: Peircean triad of a SunWeb icon

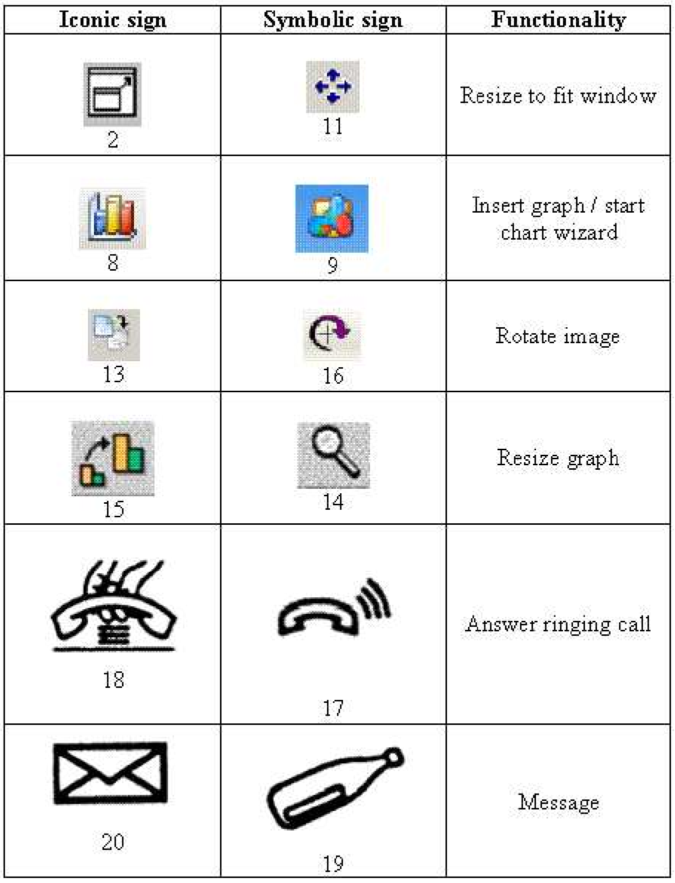
3.5 Icon Intuitiveness Test

To determine whether the results from the SunWeb and telecommunications studies would extend to other computer icons as well, we carried out an Icon Intuitiveness Test using a set of computer icons from a range of desktop applications as well as icons used in examples given by other icon researchers. The questionnaire for the Icon Intuitiveness Test can be found in appendix A of this report. In figures 3.6, 3.7 and 3.8 the column headed 'Origin' lists the application or study from which the icons were selected and below is a summary:

- Acrobat Reader 6.0
- Schaffer and Sorflaten [46]
- Microsoft Excel
- KSpread
- Maple 9.5
- Maple Worksheet 9.5
- SAS/GRAPH v8.1 and v8.2
- KView
- Piamonte, Ohlsson and Abeysekera [44].

3.5.1 Selecting the Icons

The icons selected for the study were icons that would not be too familiar to experienced computer users and they usually represented the less frequent functions in applications. This was done so that the participants' answers were not influenced by what they could



Iconic sign	Symbolic sign	Functionality
2	11	Resize to fit window
8	9	Insert graph / start chart wizard
13	16	Rotate image
15	14	Resize graph
18	17	Answer ringing call
20	19	Message

Figure 3.5: Iconic and Symbolic icons used in the Icon Intuitiveness Test

remember the icon to mean. Also, due to established conventions, many interfaces now have very similar icons for representing frequent actions such as opening a new document, saving a file to disk, copying, pasting and cutting. Therefore, these were not selected for the icon intuitiveness test — the fact that these icons have become conventional already attest to their success with users.

Further, some icons were selected that looked different but had the same functionality. These were the icons for expanding the current window to fit to the screen, inserting a graph, rotating an image clockwise, enlarging a graph, answering a ringing call and a message notification. These icons were chosen so that one representation was an iconic sign and the other a symbolic sign. Figure 3.5 shows the icons with the same functionality.

Thus, the icon was selected either because it was not a conventional icon, or it was a functionality that was represented iconically in one application and symbolically in another.

3.5.2 Organisation

In a similar way to the SunWeb study, the Icon Intuitiveness Test presented icons to users and they were asked to specify what they thought the icons meant. This was a paper based questionnaire. The icon set was divided into three groups depending on the type of ap-

Participant	Education	Context Provided
female1	Undergraduate	yes
male	Honours (Bio)	yes
female2	Honours (CS)	no

Table 3.1: Pilot study participants

plication from which they were obtained. The first group was called the 'Word processing and Spreadsheet icons' and consisted of the icons from Acrobat Reader 6.0, Schaffer and Sorflaten [46], Microsoft Excel and KSpread. The second group was called 'Maths/Statistics software and Graphics icons' and consisted of the icons from Maple 9.5, Maple Worksheet 9.5, SAS/GRAPH v8.2 and KView. The last group was called 'Telecommunications software icons' and consisted of icons from the telecommunications study [44].

The icons were organised into three tables of two columns each according to the group they belonged to. One column contained the icons and the other was left blank so the participant could fill in their interpretations. The icons within the tables had no special ordering Each table was headed by the context, i.e., the type of application the icons were obtained from. The contexts corresponded to the three groups mentioned before: 'Word processing and Spreadsheet icons', 'Maths/Statistics software and Graphics icons' and 'Telecommunications software icons'. The provision of the context has the effect of constraining the applicant's interpretation of the icon because a program embodies stable properties that constrain user activities[5].

3.5.3 Pilot Study

The pilot study was performed with three participants — two female and one male. One of the females and the male participant were both Honours students (Computer Science and Biology respectively) and the other female a final year undergraduate in a field other than Computer Science. The details of the participants can be summarised in table 3.1.

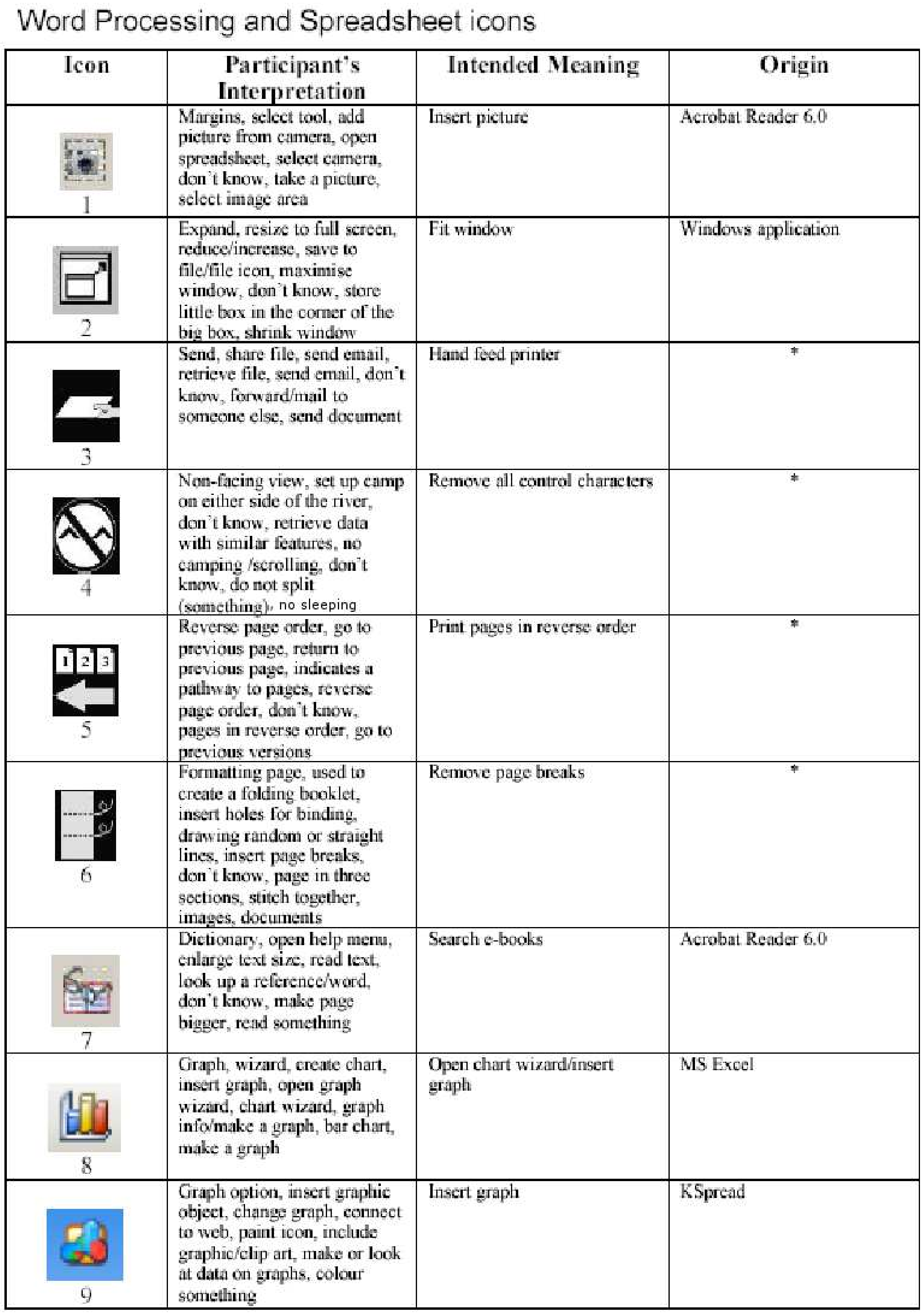
Consequently, the pilot study showed that those participants with a strong background in computing — either as Computer Science students or otherwise — could infer from the context what the icon meant.

In the case of the male participant, the guesses were almost all correct even though he admitted that he had never encountered some of the icons before. This led to the decision to omit the context from the questionnaire for Computer Science students who may participate. The context was retained for the students with no Computer Science background.

As a result, the Icon Intuitiveness Test consisted of two questionnaires in which the context was either omitted or included, depending on the computing background of the participant.

3.6 Characteristics of Test Participants

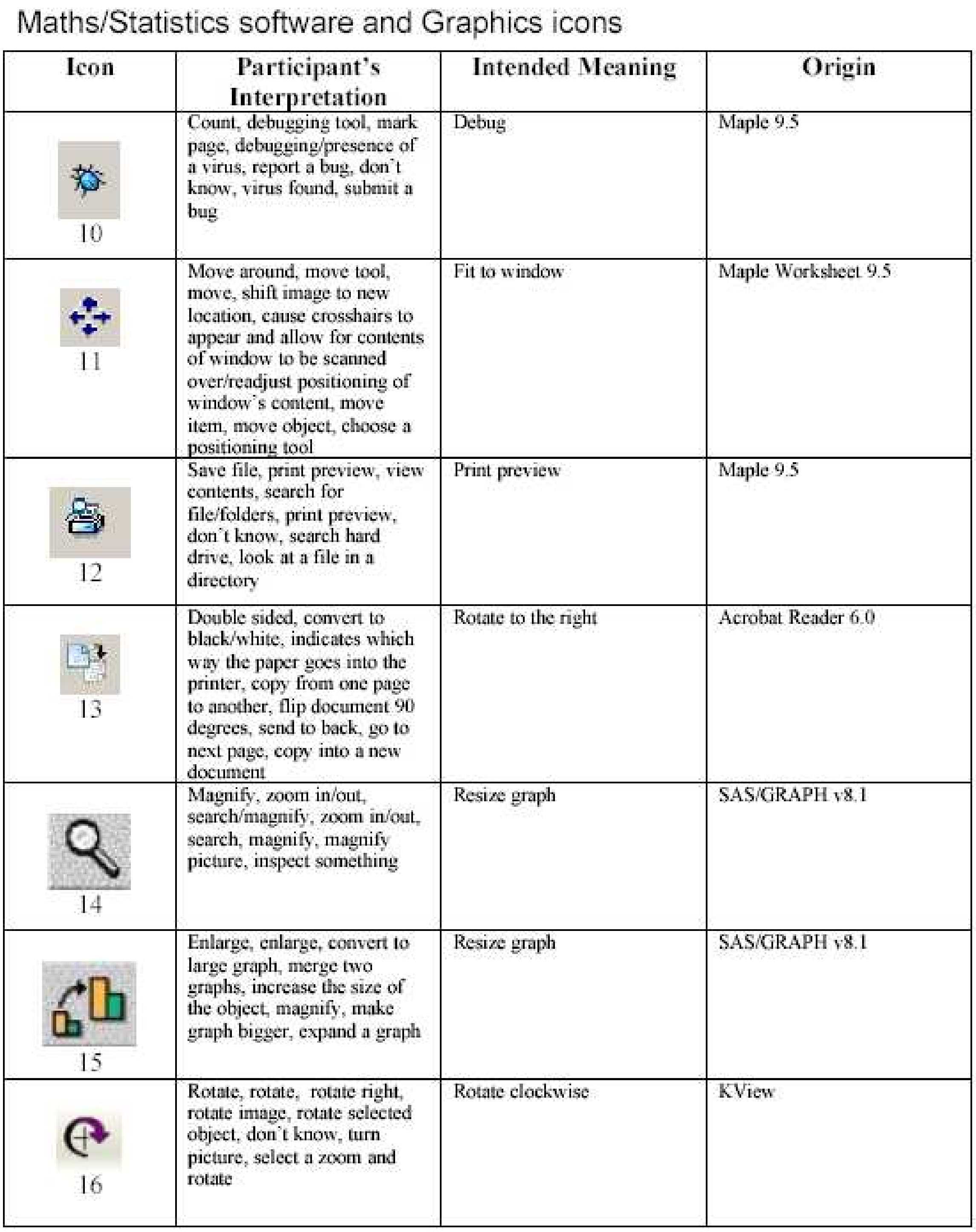
There were eight participants, two of which were Computer Science students. The others could be classified as using the computer for work or for university assignments. They were all aware that some icons had the same functionality, but were not told which.



Word Processing and Spreadsheet icons

Icon	Participant's Interpretation	Intended Meaning	Origin
1	Margins, select tool, add picture from camera, open spreadsheet, select camera, don't know, take a picture, select image area	Insert picture	Acrobat Reader 6.0
2	Expand, resize to full screen, reduce/increase, save to file/file icon, maximise window, don't know, store little box in the corner of the big box, shrink window	Fit window	Windows application
3	Send, share file, send email, retrieve file, send email, don't know, forward/mail to someone else, send document	Hand feed printer	*
4	Non-facing view, set up camp on either side of the river, don't know, retrieve data with similar features, no camping /scrolling, don't know, do not split (something), no sleeping	Remove all control characters	*
5	Reverse page order, go to previous page, return to previous page, indicates a pathway to pages, reverse page order, don't know, pages in reverse order, go to previous versions	Print pages in reverse order	*
6	Formatting page, used to create a folding booklet, insert holes for binding, drawing random or straight lines, insert page breaks, don't know, page in three sections, stitch together, images, documents	Remove page breaks	*
7	Dictionary, open help menu, enlarge text size, read text, look up a reference/word, don't know, make page bigger, read something	Search e-books	Acrobat Reader 6.0
8	Graph, wizard, create chart, insert graph, open graph wizard, chart wizard, graph info/make a graph, bar chart, make a graph	Open chart wizard/insert graph	MS Excel
9	Graph option, insert graphic object, change graph, connect to web, paint icon, include graphic/clip art, make or look at data on graphs, colour something	Insert graph	KSpread

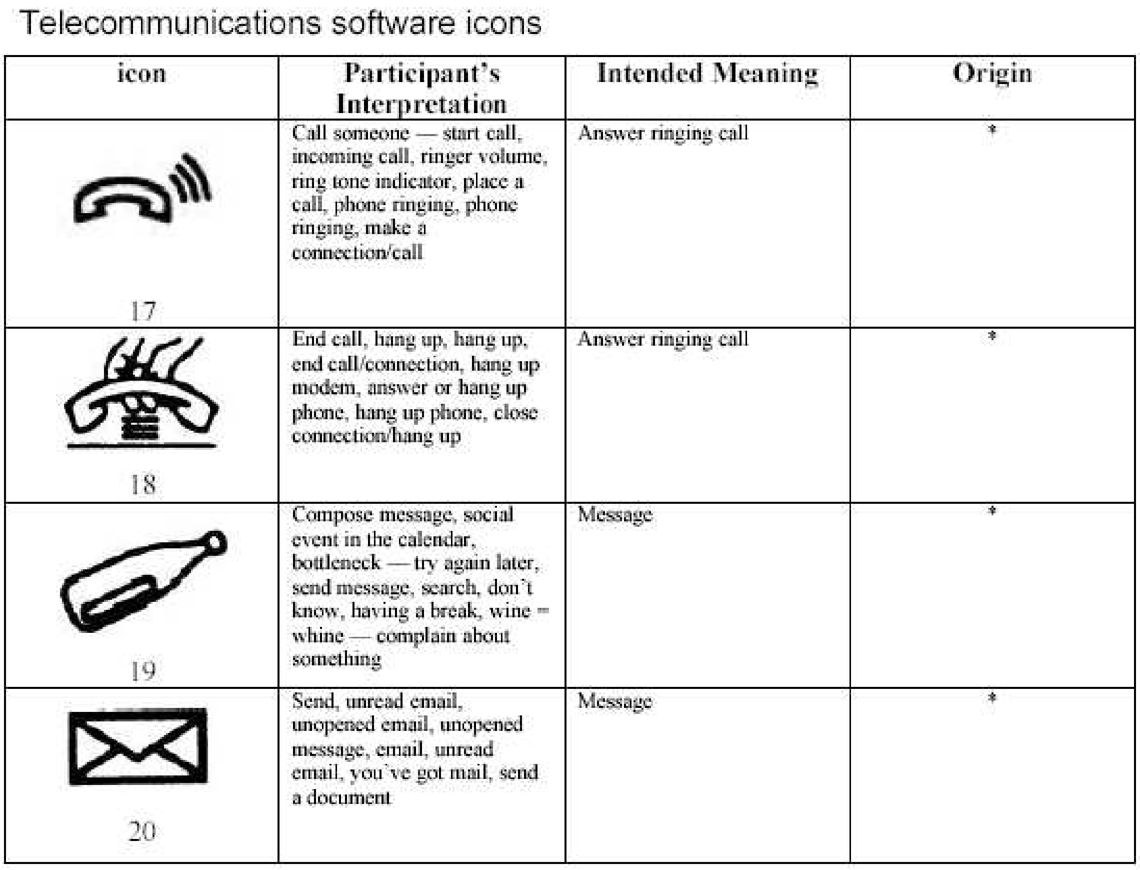
Figure 3.6: Icon intuitiveness test results. * are from Schaffer and Sorflaten[46].



Maths/Statistics software and Graphics icons

Icon	Participant's Interpretation	Intended Meaning	Origin
10	Count, debugging tool, mark page, debugging/presence of a virus, report a bug, don't know, virus found, submit a bug	Debug	Maple 9.5
11	Move around, move tool, move, shift image to new location, cause crosshairs to appear and allow for contents of window to be scanned over/readjust positioning of window's content, move item, move object, choose a positioning tool	Fit to window	Maple Worksheet 9.5
12	Save file, print preview, view contents, search for file/folders, print preview, don't know, search hard drive, look at a file in a directory	Print preview	Maple 9.5
13	Double sided, convert to black/white, indicates which way the paper goes into the printer, copy from one page to another, flip document 90 degrees, send to back, go to next page, copy into a new document	Rotate to the right	Acrobat Reader 6.0
14	Magnify, zoom in/out, search/magnify, zoom in/out, search, magnify, magnify picture, inspect something	Resize graph	SAS/GRAPH v8.1
15	Enlarge, enlarge, convert to large graph, merge two graphs, increase the size of the object, magnify, make graph bigger, expand a graph	Resize graph	SAS/GRAPH v8.1
16	Rotate, rotate, rotate right, rotate image, rotate selected object, don't know, turn picture, select a zoom and rotate	Rotate clockwise	KView

Figure 3.7: Icon intuitiveness test results continued…



Telecommunications software icons

icon	Participant's Interpretation	Intended Meaning	Origin
17	Call someone — start call, incoming call, ringer volume, ring tone indicator, place a call, phone ringing, phone ringing, make a connection/call	Answer ringing call	*
18	End call, hang up, hang up, end call/connection, hang up modem, answer or hang up phone, hang up phone, close connection/hang up	Answer ringing call	*
19	Compose message, social event in the calendar, bottleneck — try again later, send message, search, don't know, having a break, wine = whine — complain about something	Message	*
20	Send, unread email, unopened email, unopened message, email, unread email, you've got mail, send a document	Message	*

Figure 3.8: Icon intuitiveness test results continued. * are from Salasoo[44].

It was not important for the participants to be existing or prospective users of the applications that the icons were obtained from, since this test was to ascertain how users would interpret the icons. Hence participants who have not encountered these icons before would give a better indication of how well understood the icons would be in general.

3.7 Results

From figures 3.6, 3.7 and 3.8 it is obvious that a given picture definitely does not convey the same thousand words to all viewers! These figures show all the guesses by the participants. Some participants made more than one guess for an icon and some participants made identical guesses, which are shown as separate guesses[1].

Correct guesses were taken as guesses that were either exactly the functionality represented, or a close guess that would be unambiguated by the context of the specific application. For example, the guesses for the 'Fit window' icon (icon number 2) in figure 3.6 were taken as correct if they indicated that something could be expanded to fit the window, and almost correct if the guess implied some analogy to making something larger, expanding or increasing. Figure 3.10 shows that for icon 2 there were four correct/almost correct guesses. These would be the guesses 'expand', 'resize to full screen', 'increase' and 'maximise window'. The same figure shows that icon 5 had three correct/almost correct guesses. These would correspond to 'reverse page order', 'reverse page order' and 'pages in reverse order'. It is clear that the main concept of applying some functionality to pages in the reverse order was understood by the participants. An assumption we make is that in the context, i.e., in the actual application, the same participant would be able to deduce from the information available that this icon means 'print pages in reverse order'.

3.7.1 Icon Success

ISO (International Organisation for Standards) 9186 suggests a rate of 66% for icon *recognition rates* in order for the icon to be classified as successful. In this study, the recognition rate can be computed by taking the number of correct/almost correct guesses as a percentage of the total number of guesses. The values we are working with are rather small, so it is necessary to bear in mind that one guess more or one guess less will have the effect of increasing or reducing the recognition rate by more than 10%.

According to the benchmark of 66%, the most successful icons (in figures 3.10, 3.11 and 3.12) were:

- icon 8 *recognition rate*: 100%
- icon 15 *recognition rate*: 75%
- icon 16 *recognition rate*: 100%
- icon 20 *recognition rate*: 75%

Icon 9 is very close to being successful with 62.5%.

Noticeably, out of this set of four, three are iconic signs (icons 8, 15 and 20) and one is a symbolic sign (icon 16).

[1]No guess is taken as a 'don't know'

The least successful icons are those with a recognition rate of zero: icons 3, 4, 6, 11, 14, 19.

The participants had the most difficulty with icon 4. Although there were other icons in the set that had zero correct/almost correct guesses, this icon had only two guesses ('retrieve data with similar features' and 'do not split (something)') that could reasonably be functionality belonging to a word processing or spreadsheet application. The other participants either could not guess at all or simply ignored the given context and made a guess based on the appearance of the image — 'set up camp on either side of the river' is a particularly iconic interpretation of the icon, since the diagonal line through the image is not even recognised as a symbolic sign prohibiting something.

Another obviously problematic sign is icon 19. This image of a bottle triggered the idea of a message in exactly two participants but they disagreed on whether the message could be sent or composed using this icon. One participant expected the interpretation of this icon to require a referential action such as was required for the debugging icon (icon 10). Such icons are sometimes called *visual puns* and are prone to misunderstanding.

3.7.2 Iconic versus Symbolic Signs

The computer icons that had the same functionality were chosen such that one was an iconic representation and the other a symbolic representation (see figure 3.5).

Resize to Fit Window

Icons 2 and 11 both represent expanding the window to fit the screen, however, icon 2 is the iconic representation and icon 11 is the symbolic representation. In this study none of the participants could guess the symbolic icon 11 correctly, whereas icon 2 performed better with a recognition rate of 50%.

One reason for this result could be that on many desktops the mouse pointer changes to look like icon 11 when it is possible to move an object around. Hence explaining the reason almost all guesses referred to the action of moving an object.

Insert Graph

Icons 8 and 9 both represent the option of inserting a graph or starting a graph wizard to help create a graph. Icon 8 is the iconic representation and had a significantly better recognition rate (100%) than icon 9 (62.5%).

In this case icon 8 was taken from a more popular application — Microsoft Excel — than icon 9 from KSpread. Additionally, the size of icon 9 makes it appear incomprehensible but at a higher resolution as in figure 3.9 it is much clearer what this icon is supposed to represent. If the icon in this form was used in the icon test it would have been an iconic sign. It was, however, the lower resolution image that was used since this is closer to the way it appears in the actual application.

Rotate Image

Icons 13 and 16 both represent the functionality of rotating an image to the right or clockwise. In the case of icon 13, only one participant correctly guessed the functionality. All

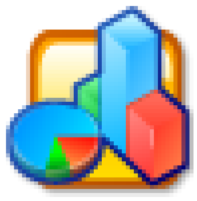


Figure 3.9: Higher resolution image of icon 9

participants understood icon 16's functionality even though this is the symbolic representation of the notion of rotating an image.

Icon 13 is more iconic since it shows the before and after state of the object to be rotated, as well as the direction indicated by the arrow. While participants understood that the arrow indicated direction, they were confused as to what the arrow was indicating. One interpretation was obviously influenced by the fact that the before state of the object is indicated in yellow and the after state in grey. Hence, they focused on the fact that the before state appeared differently to the after state due mainly to its colour, and concluded that the icon indicates that the object is converted to black and white.

Resize Graph

Icon 14 is a tentative icon that would have been used in SAS/GRAPH v8.1, but Wimmer's usability study[50] showed that this image did not represent the graph resizing functionality the designers had intended. The designers then changed this icon to icon 15 but did not report whether the users did indeed understand this icon better.

If the results of this icon intuitiveness test under discussion is to be believed, then the results here do support icon 15 as a better icon for the graph resizing functionality. All participants guessed that the magnifying glass represented some zoom functionality, but gave no indication that it could be used to resize an object, i.e., change its actual size. Interpretations for icon 15 seem to indicate that this better represents the graph resizing functionality, since many applications (Acrobat and Konqueror, for example) use the magnifying glass as a zoom tool. This result is an indication of the already accepted convention of the magnifying glass as a zoom tool by the participants in this study.

Answer Ringing Call

Icons 17 and 18 both represent the functionality of answering a call indicated by ringing. Icon 18 can be seen as a more iconic sign than icon 17 due to the hand on the phone that is supposed to look like the action of someone answering a phone. However, the participants all agreed that icon 18 could represent the functionality of ending a call. One participant specified that it could mean either answering or ending a call. Although the iconic sign was not interpreted correctly it is notable that the participants' guesses were quite similar and consistent in their interpretation of this icon. For icon 17 the participants were more divided about whether it represented the tone of the call or making a call. Only two participants guessed the correct functionality of this icon.

The reason participants guessed icon 17 to be the icon for answering or placing a call and icon 18 to be the icon for ending a call is not very clear. The guesses could be attributed to the order in which the icons appeared, or due to the three black lines appearing under the phone in icon 18 that could signal movement of the phone, which the participants interpreted as a downward movement. Another reason could be that the grip of the fingers of the hand is rather loose. This may have been interpreted as a 'letting go' action rather than a 'picking up' action.

Message

Icons 19 and 20 indicate that there is a message to be read. Icon 19 is the symbolic sign and also the only icon with unique interpretations supplied by each participant. Again the iconic representation of the message (icon 20) resulted in fairly consistent interpretations, i.e., unread/unopened mail message, by most participants.

3.7.3 Icons For Actions

All of the icons except for 12, 19, and 20 all represent actions, i.e., they are all icons for doing something like inserting, feeding, searching, creating, etc. Icons 12, 19, and 20 represent system objects on which functions are performed. Icon 12 represents a print preview of a document for inspecting and perhaps printing, icons 19 and 20 represent a message in the system that can be read, forwarded, or replied to. However, participants attributed actions even to these icons, alluding to the notion that in a test situation, participants may naturally expect the icons they are presented with to be icons for performing actions.

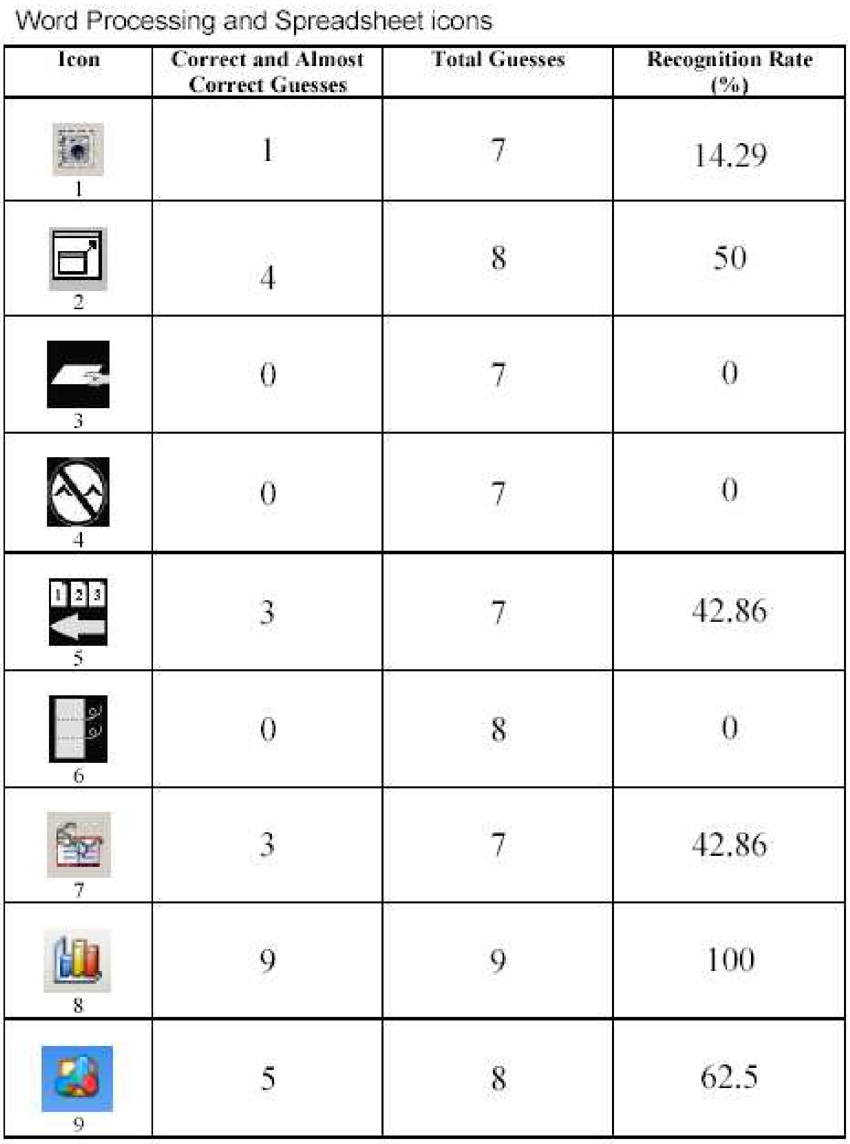
3.8 Important Considerations

There are some issues worth noting about the design of the Icon Intuitiveness Test:

1. It was clear that the participant's experience with computer applications affected how many icons were interpreted correctly, as was made evident in the pilot study. Additionally, participants' guesses could have been influenced by their knowledge of what functionalities are available in certain types of computer applications. The Icon Intuitiveness Test made no further distinctions between computer users other than the two groups defined as those with a known strong background in computing and those without.

2. Cultural aspects were not taken into account in devising the test or interpreting the results. Icon 5 could be problematic for cultures that read based on a right-left orientation. For Western cultures, that read from left to right, the pages appear in numerical order from left to right and the arrow indicates that the functionality of this icon (printing in this case) will be applied to the pages in the opposite direction. This design could be problematic for a person who reads right to left, since the arrow will be pointing in the logical direction but the page numbers will be in reverse order. The Icon Intuitiveness Test could not produce results to determine the difference in recognition rates due to the reading directions of different cultures.

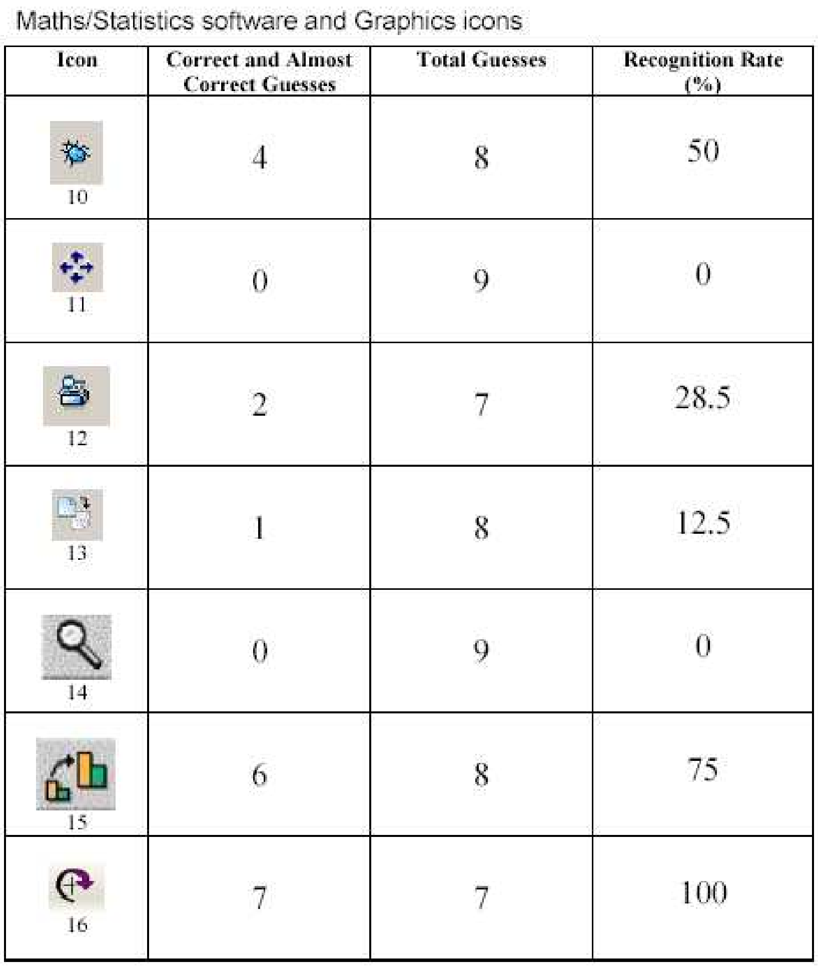
3. The icons were presented on paper. Therefore the participants could not compare them to any other icons in the same application in order to make a guess at their



Word Processing and Spreadsheet icons

Icon	Correct and Almost Correct Guesses	Total Guesses	Recognition Rate (%)
1	1	7	14.29
2	4	8	50
3	0	7	0
4	0	7	0
5	3	7	42.86
6	0	8	0
7	3	7	42.86
8	9	9	100
9	5	8	62.5

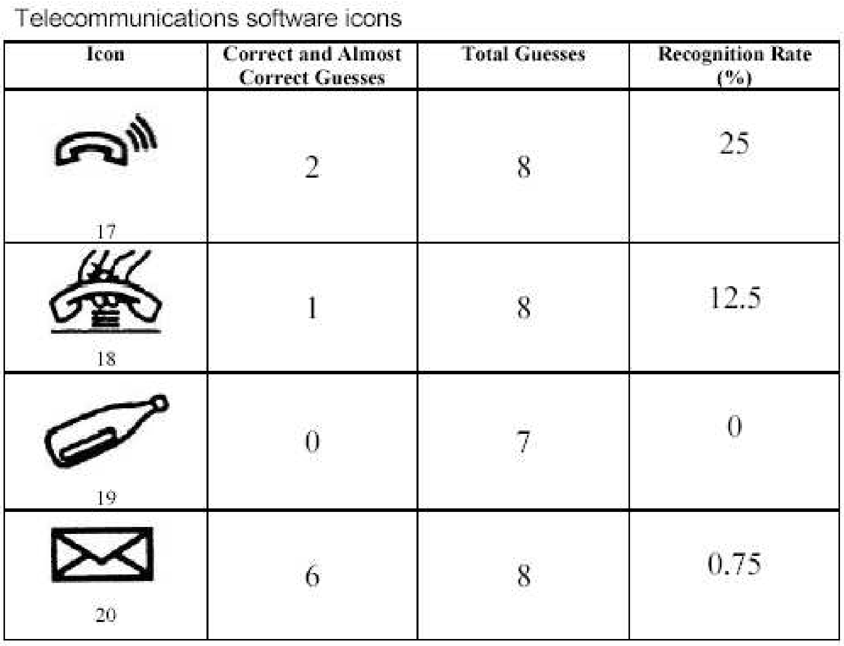
Figure 3.10: Guessing results



Maths/Statistics software and Graphics icons

Icon	Correct and Almost Correct Guesses	Total Guesses	Recognition Rate (%)
10	4	8	50
11	0	9	0
12	2	7	28.5
13	1	8	12.5
14	0	9	0
15	6	8	75
16	7	7	100

Figure 3.11: Guessing results continued. . .



Telecommunications software icons

Icon	Correct and Almost Correct Guesses	Total Guesses	Recognition Rate (%)
17	2	8	25
18	1	8	12.5
19	0	7	0
20	6	8	0.75

Figure 3.12: Guessing results continued

functionality. This is not a reflection of reality, where users of an application may be in a better position to guess the icon correctly because there are others to compare and contrast it with.

3.9 Summary

Goguen claims that "all else being equal: Icons are better than indices, and indices are better than symbols" [27], but provides no evidence for this claim. In this chapter we have shown that several studies do support the notion that icons are better than symbols, however icon evaluation is a very difficult task. There are many factors and limitations that have to be taken into account such as the medium in which the icons are presented and tested, and the background of the participants.

The important observations from this study can be summarised as follows:

1. Some concepts are better explained by symbolic signs, for example, rotation of an object.
2. The pairs of opposite actions of sending and receiving, increasing and decreasing, answering and hanging up are very difficult to convey.
3. Participants are more likely to attribute an action to an icon than guess that it represents a system object.
4. When participants can not think of a suitable functionality that fits into the given context, they will base their guess on the visual clues that they can see and so, interpret the icon based on what they think they see. This was most clearly illustrated by icon 4.

5. Iconic signs generally are better recognised than their symbolic counterparts and in the case where participants interpret the iconic signs incorrectly, they are surprisingly consistent in their interpretations of the image.

Chapter 4

Semiotic Analysis of User Interface Redesign

The work presented in this chapter was compiled as a paper and has been accepted for publishing by the Australian User Interface Conference 2005 (see appendix A).

4.1 Introduction

Given that designing the user interface is a semiotic activity, it makes sense to examine usability problems and subsequent redesigns of the user interface in terms of semiotics. User interfaces that are good at communicating to the user what it is used for, should need less redesigning and less resources as a consequence. Redesigning an interface usually takes place once it has been tested or been exposed to expert evaluation, but once these have been performed it is only clear that parts of the interface need to improve. There is very little theory about why the usability problems exist and how to redesign to improve the situation. Semiotics can help the designer improve their communication power [23]. In this chapter we show how semiotic analysis can give insights into user interface design issues and why some designs do a better job of communicating than others. This chapter examines interface redesigns that we have come across in the literature and they are analysed according to the Peircean model of the sign. It was striking how few examples of comparative studies of the interface before and after redesign exists. Hence, the examples discussed below had to be picked from a very small body of adequate studies. The first two are from a usability case study of the graphical user interface of the V9 Graphing Tools in SAS/GRAPH®— a component of SAS software used for client side data visualisation[50]. The third example compares the redesigns of Microsoft®Word user interfaces, available in the vast number of sources that deal with this popular software.

4.2 Cascading Menus

Figure 4.1 shows the cascading menus that the user needed to navigate through when making changes to a graph. In the figure, the user has selected the 'Decrease' menu option in order to decrease the width of the bars of the graph.

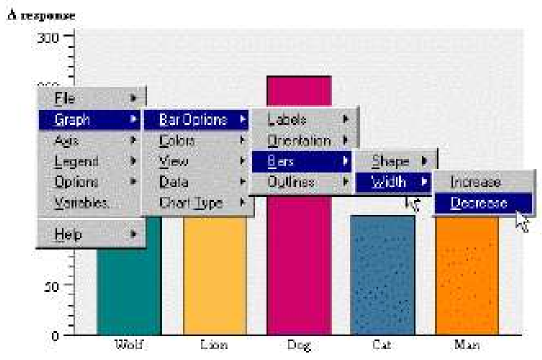


Figure 4.1: SAS Version 8.1 Cascading menus

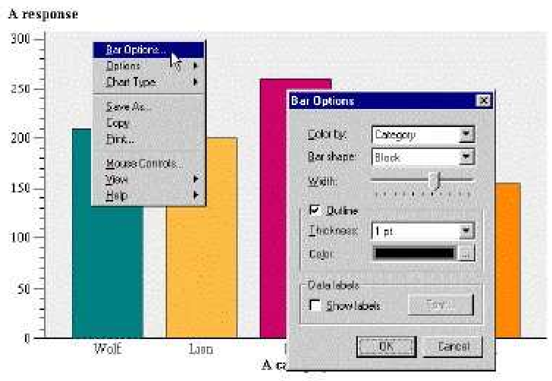
4.2.1 Usability Problems

The usability study showed two problems with this menu design. The first was that the user was required to be quite accurate in the mouse movements. If the mouse pointer was to venture too far from the region surrounding the menu, the menu would collapse and the process would have to start all over again. Navigating through five layers of submenus is also quite cumbersome. The second problem was that the amount by which the bars on the graph could be increased or decreased could not be specified by the user. Therefore, the user was required to navigate through five levels of cascading menus repeatedly until the satisfactory width had been achieved.

Another usability issue with a cascading menu, although not mentioned in Wimmer's report, is that the menu items are hidden. Only when the mouse pointer moves over the menu item is the next level displayed and this violates Constantine and Lockwood's Visibility Principle and Jakob Nielsen's heuristic of recognition rather than recall. Cooper[22] discourages the use of menus with several layers especially in the case of items that are required frequently.

4.2.2 Semiotic Analysis

Applying Peirce's notion of sign categories to this interface sign, it is clearly both symbolic and indexical. This sign is strongly symbolic because it is by pure convention among computer users that cascading menus are used to allow access to certain functionality within an interface. Users learn that there are certain choices offered by the menus and over time they can memorise how the menu is structured. Novice users may not be able to navigate directly to the functionality they require (the option to decrease the bar widths in this example) because it must be learned. The symbolic sign requires them to experiment and learn by trial and error how to perform their task using this sign. The cascading menu is indexical because once the user has cascaded through the menu and selected the decrease option this causes a decrease in the widths of the bars. This change appears visually to the user, who may then decide that the width has decreased by an adequate amount, or they may decide to decrease again or to change back to the size it was before.



Color by: Category
Bar shape: Block
☑ Outline
Thickness: 1 pt
Color: ■
☐ Show labels

Figure 4.2: SAS Version 8.2 Slider

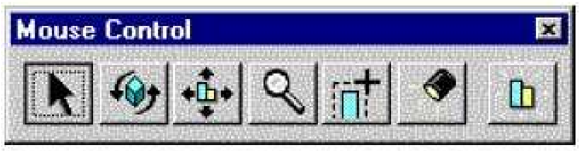


Figure 4.3: SAS Version 8.2 Magnifying Glass Icon

4.2.3 Redesign

After the usability study, the cascading menus were replaced by adjusting a slider in the dialogue box in figure 4.2. This immediately frees the user to determine the width of the bars in a more hands on fashion — as they adjust the slider, the bars widen or become narrower. Now the process of changing bar widths involves a strongly indexical sign, in the from of the slider. The user can perceive the changing bar widths as the position of the slider changes, so the effects perceived by the user are more immediate and the sign is more interactive. Additionally, the results of the interaction are visible during the interaction, not after it as in the cascading menus.

The cascading menu example can be thought of as required semiosis. The user needs to navigate through 5 menus before getting to the desired functionality. As the first menu option is selected, the reaction/interpretant is the appearance of the next menu. This menu becomes the representamen of the next sign and this process repeats itself 5 times until the decrease option becomes the interpretant of the last sign.

4.3 Magnifying Glass

In figure 4.3, the magnifying glass icon represented the graph resizing functionality, supposedly suggesting that the graph was made smaller by zooming out, and larger by zooming in[50].

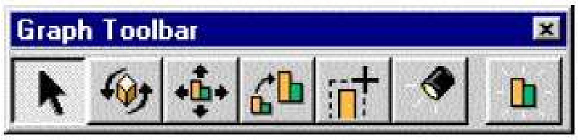


Figure 4.4: SAS Version 8.2 Resize Graph Icon

4.3.1 Usability Problems

The usability study showed that users were confused by the zoom metaphor. Many applications, such as Acrobat Reader and Konqueror use the magnifying glass to represent functionality that allows a user to zoom in or out of a document, without changing its actual size. This may be the behaviour the participants in the usability test expected and it clashed with the behaviour implemented by the designers of SAS/GRAPH®.

4.3.2 Semiotic Analysis

The user's interpretant of the magnifying glass icon did not match the object that the designer had intended with this sign. Even though both the designer and the user would agree that the picture on the button fourth from the left on the 'Mouse Control' tool bar is a magnifying glass, the confusion lies with what functionality the user interprets the magnifying glass to represent. Since many popular applications (Acrobat Reader and Konqueror for example) make use of the magnifying glass to represent the functionality of zooming in and out of objects on screen, this may be the functionality the users taking part in the usability test would have assumed the magnifying glass in the SAS/GRAPH®interface to have. The magnifying glass is a symbolic sign in this case because the link between the magnifying glass and the resizing functionality is made arbitrary by the fact that a magnifying glass can not change the actual size of an object.

4.3.3 Redesign

The fourth button on the left in figure 4.4 shows the replacement button for the graph resizing functionality. This is a more iconic representation of resizing the graph due to the resemblance that exists between the image on the button and the resizing functionality it represents. If the size of a graph can be seen as a quality, this is in accordance with Barr et. al.'s[9] heuristic that proposes that icons representing qualities should be iconic signs. This helps to ensure that there is a better chance of the designer and the user agreeing on what functionality the icon represents. In fact, Peirce makes the following statement about qualities: "Since a quality is whatever it is positively in itself, a quality can only denote an object by virtue of some common ingredient or similarity." [42].

4.4 Microsoft®Word Text Styles

In Microsoft®Word Version 3[30] the options for setting the font style in a document were structured as part of a meny system. There were no usability studies available to explain

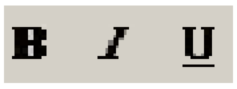


Figure 4.5: Microsoft®Word (2002) text format styles: bold, italic, underline

whether users experienced difficulties with this menu system, but since the successive versions of Microsoft®Word are easily accessed, either in the literature or on a desktop computer, it is still possible to do an analysis of the redesigns.

4.4.1 Semiotic Analysis

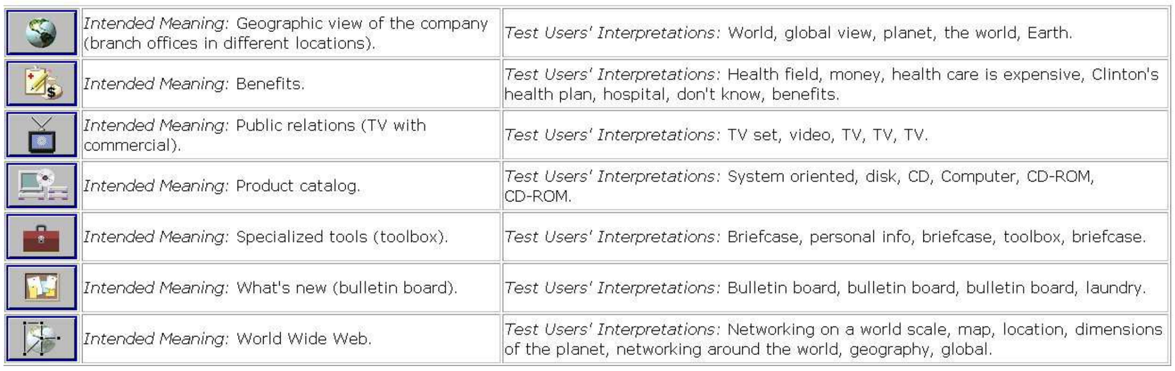
As discussed in section 4.2, text based menu systems are symbolic signs. Not only do users have to learn to associate the terminology used by the interface designer with certain functionality, they are also required to learn the structure of the menu system.

4.4.2 Redesign

By version 5 the icons in figure 4.5 were available for placing on the toolbar. These are iconic signs because the button for making text bold, resembles bold text. Similarly the italicised *I* and the underlined U resemble the result when applied to selected text in the document. This is another instance of redesigning a representation of functionality from being a symbolic sign to becoming a more iconic sign. These format styles can be seen as qualities of the text in a document and the same discussion as in section 4.3.3 applies here. The reason for the bold, italic and underline options to be represented on the toolbar may be many, but obviously the user would benefit by having a shorter route to functionality that is used frequently. Considering that Microsoft®Word is used as a word processor, the actions of bolding, italicising and underlining text are frequent actions.

4.5 Discussion

In the above examples, we discussed the shifts between Peirce's different sign divisions in the user interface as it is redesigned. It is interesting to note how the shift tended to be away from symbolic signs toward indexical signs, in the case of the slider for adjusting the widths of the bars in SAS/GRAPH®, and towards iconic signs in the other examples. There is evidence that users tend to interpret computer signs as being iconic, i.e., that they resemble their underlying functionality. Nielsen and Sano[39] present an "Icon Intuitiveness Test" where images were presented to users and then asked to indicate what functionality they thought the icons represented. For the most part the users seemed to respond to the question in a way that made it clear that they were interpreting the sign iconically (see figure 4.6). If the intended meaning of the icon is equated with the object of that sign and the Test User's Interpretation equated with its interpretant, then Peirce's triadic model can be reconstructed for each individual icon. On a larger scale there has been research in the area of making a whole interface resemble the object in the real world whose functionality it represents. IBM aims to create interfaces that are more natural and intuitive by designing the interface to resemble a real world artefact[31]. Their term for these interfaces is RealThings and one



	Intended Meaning: Geographic view of the company (branch offices in different locations).	*Test Users' Interpretations:* World, global view, planet, the world, Earth.
	Intended Meaning: Benefits.	*Test Users' Interpretations:* Health field, money, health care is expensive, Clinton's health plan, hospital, don't know, benefits.
	Intended Meaning: Public relations (TV with commercial).	*Test Users' Interpretations:* TV set, video, TV, TV, TV.
	Intended Meaning: Product catalog.	*Test Users' Interpretations:* System oriented, disk, CD, Computer, CD-ROM, CD-ROM.
	Intended Meaning: Specialized tools (toolbox).	*Test Users' Interpretations:* Briefcase, personal info, briefcase, toolbox, briefcase.
	Intended Meaning: What's new (bulletin board).	*Test Users' Interpretations:* Bulletin board, bulletin board, bulletin board, laundry.
	Intended Meaning: World Wide Web.	*Test Users' Interpretations:* Networking on a world scale, map, location, dimensions of the planet, networking around the world, geography, global.

Figure 4.6: SunWeb's Icon Usability Test Results

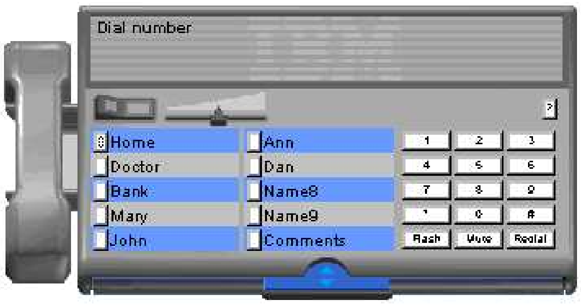


Figure 4.7: Example of IBM's RealThing

example from their website is shown in figure 4.7. Whether this iconic representation inherent in RealThings interfaces has the same benefits as iconic signs in the interface will not be known until results of usability studies are made available.

4.6 Summary

In the words of de Souza [24], "our conclusions at this point is more a matter of speculation and sense than of thorough testing and analysis". Based on the idea that the user interface is a complex sign built up from many smaller signs, we propose that examining interface redesigns can provide meaningful insights to designers. Semiotic analysis is an effective tool for analysing the communicability and interpretability of signs in the user interface, so we applied the Peircean model to some redesigns. This analysis provides a better understanding of redesigns and can potentially aid designers in designing better interfaces in the future. The major insight gained from this study is that the redesign tended to change the existing sign from a symbolic into an indexical or iconic sign. Research into the intuitiveness of computer icons tends to support this move and indicates that users tend to interpret the signs they see as iconic signs.

Chapter 5

Usage-Centered Design

5.1 Introduction

The success of Usage-Centered Design (UCD) has been well documented in a number of software projects [41], [51]. UCD is a design methodology that produces a user interface that is the result of several derivations of successive models [20]. In UCD the abstract models describe the processes in the domain [2]. Constantine [20] talks of the advantage of modeling at the abstract level as providing the user interface designer with more 'creative leverage' than a design process that moves to the concrete level very early on. Another important aspect that sets UCD apart from other interface design methodologies is the focus on usage, i.e., user tasks. This is, however, not a novel idea and was introduced in 1987 by Bodker [12]. Yet, in a personal conversation with James Noble in 2004, Constantine admitted that it is still not well understood *why* the process is so successful.

The aim of this chapter is to form a tentative response to Constantine's musing, basing our argument on the Peircean model of semiotics. We investigate the semiotic processes involved in UCD's core models and employ the use of a running example as illustration of the UCD process in practice. All models discussed are taken from the book by Constantine and Lockwood[21].

5.2 What is Usage-Centered Design?

In a talk given by Constantine, he referred to the UCD process as "successive translations or derivations" [19] of various models. In an interview with David J. Anderson, he further explained that through the use of models, we can communicate understanding and skills[6]. The UCD process can then be described as understanding and skills that are successively translated and derived through the use of models. Biddle, Constantine and Noble propose models as ideal tools for answering questions like "What capabilities must be present within the user interface for to solve the users' problems? How should they be organised into handy collections? How should the work flow within and between the various parts of the interface?" [10] Indeed, UCD makes heavy use of modeling in an attempt to integrate usability into the design of the user interface. Usability, of course being an important characteristic of any interface, and the incorporation of usability into the design can ensure that resources are not wasted in the process of developing designs that have to be changed at a later stage. There are five models that form the complete UCD process [21] and these appear in figure 5.1:

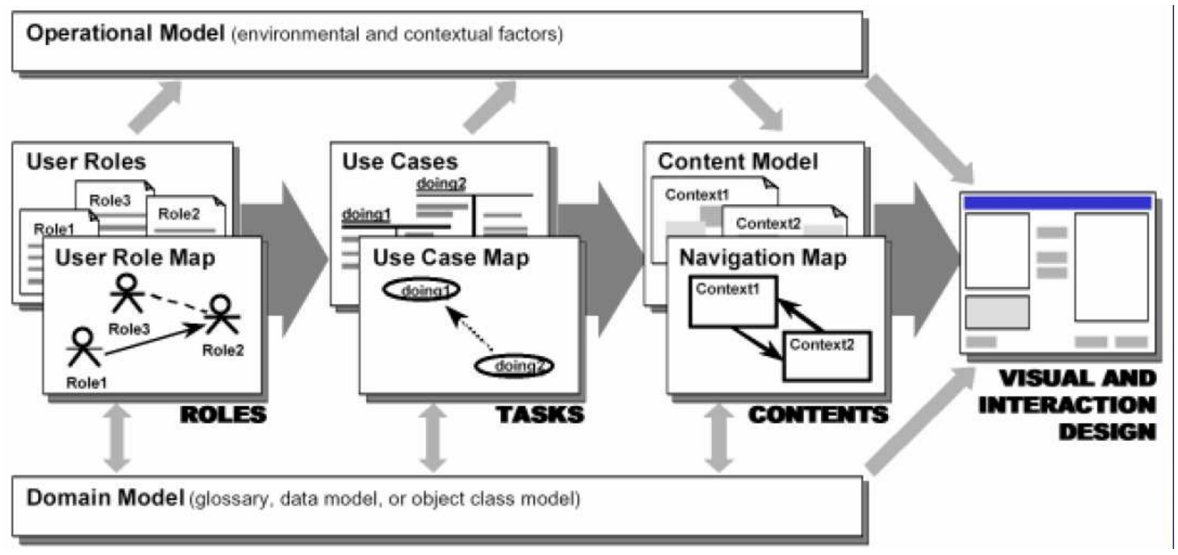


Figure 5.1: Schematic Outline of Usage-Centered Design Process [25]

1. Role model — the relationships between users and the system
2. Task model — the structure of tasks that users will need to accomplish
3. Content model — the tool and materials to be supplied by the user interface
4. Operational model — the operational context in which the system is to be deployed and used
5. Implementation model — the visual design of the user interface and description of its operation

Since the time of publishing their influential book titled *Software for Use* [21], Constantine and Lockwood have added the Canonical Abstract Prototype (CAP) to their UCD process, because practical experience with the UCD process showed that there existed a substantial conceptual gap between the core models provided and the realistic implementation of the interface. With the introduction of the CAP, the transition from the abstract content model to the concrete prototype is much smoother [20]. Due to its addition, the CAP will also be discussed along with the role, task and content models.

5.3 Modeling

Modeling is not peculiar to UCD, it is also an Object-Oriented (OO) Design activity. OO-modeling recognises two domains that require modeling — the *application domain* and the *solution domain* [13]. To Bruegge and Dutoit the application domain represents all aspects of the user's problem, whereas Andersen refers to this as the *problem domain*, defined as "that part of the real world the system is required to control, administer, regulate, etc." [4]. These domains are also applicable to user interface design. Andersen and Nowack suggest 4 different domain models and Constantine and Lockwood believe it is outside the scope of the UCD process [21], and do not discuss it further. We will use Bruegge and Dutoit's term and also use their term for the space containing all possible implementations — the solution domain. Domains are an important consideration because, as we have pointed out in the introduction, Andersen, Hage and Brandt believe that models describe a domain [2]. In order to apply semiotics sensibly, we need to determine what the model as sign is representing, i.e., what it is modeling. This helps to pin down the object of the model as sign.

The user interface designers have their own special roles to play in the modeling process. There are two roles: the *model-constructer* and the *model-interpreter* [4]. These roles do not necessarily belong to different people, since small projects may have only a single interface designer and we occasionally refer to these terms when it is necessary to distinguish user interface designers that interpret the models and those that construct them.

This chapter will investigate the semiotic processes involved in the following core models of the UCD Process:

- Role model
- Task model
- Content model

The information from these models are the main drivers behind the design decisions regarding the tasks supported by the user interface and its architecture.

Adapting Liu, Liao and Chong's [15] description of the software engineering process, the UCD process can be seen as "a series of transformations between requirements and solutions at different levels." This implies that the solution that models a domain becomes the requirements of the next solution. We see this occurring in the UCD process where the role model is a solution that becomes the requirements for the task model, and the task model then becomes the requirements for content model. The content model becomes the requirements for the CAP, and the CAP in turn becomes a model for the real world implementation of the prototype.

5.4 Semiotic Discussion

Andersen and Nowack explain a triadic concept of models as signs — the model represented by the special notation that inherently belongs to it, and renders it visually is the representamen. This representamen refers to some other tangible thing, which is the concept the model represents. The interpretant is what can be called a *referent system*. This referent system is "the particular way we choose to relate the model to reality." [4] Hence, each model will have a different referent system as its interpretant, since each model employed by a process such as UCD is intended to model a unique aspect of the domain, but UCD designers should agree on what the referent system of each model is so that effective communication of the models can take place.

Closely following Andersen and Nowack's semiotic analysis of the Object-Oriented Analysis and Design process [4], we realise that the UCD models listed above are signs. These models can be related to Peirce's triadic model of the sign, where the actual models that are constructed during the user interface design process become the representamen that stands for concepts in the underlying domain, the way the model is interpreted becomes the interpretant, and the object relates to the corresponding real world phenomenon in the domain.

5.5 Running Example

Throughout the rest of this chapter, we will refer to an example of the design of a user interface to a computer system for use in a small pizza business. This example was developed as part of a course in User Interface Design[1], teaching the methods of the UCD process. The models were produced as part of the set tasks that were carried out as group work.

5.6 User Role Model

The first important UCD component is the *user role model*. This essential model involves identifying who the most important users of the system are and what tasks they will be performing with the system. Identification of the user roles and the relationships between them, guides decisions about what functionality will be implemented in the interface.

5.6.1 Structured Role Model and Focal Roles

Users are the people who would or could use the system. Therefore, they can be sorted into groups according to their needs, interests, expectations, behaviours and responsibilities and the different groupings that result are the *user roles*. In contrast to the UML notion of *actor*, which can be a role played by a human or a non-human system, the UCD term *role* relates only to the human users with respect to the system.

Focal roles are a collection of user roles selected from all the possible user roles, which are deemed as the most important. The usage, i.e., tasks, of the focal roles will then drive the rest of the design process. User interface designers will ensure that the user interface supports all tasks belonging to the focal roles first before turning their attention to implementing functionality for the other roles.

The *user role map* is a graphical representation of the user roles and their relationships.

Structured Role Models order user information, i.e., the needs, interests, expectations, behaviours and responsibilities of the users, in a more detailed way in order to capture characteristics of their interaction with the system. Among other information, structured role models can include descriptions of the patterns of usage of a given role, usability criteria that are required by a role and the level of domain knowledge (of the application area) someone in a specific role can be expected to have.

In our example of the pizza business, we based our roles on all the people who could be involved in the ordering, making and delivering of pizzas. Figure 5.2 shows the user role map with the roles we identified. These were the customer, the person behind the counter who takes the orders, the pizza maker, the delivery person who delivers the pizzas, and the manager of the store. These also correspond to the basic roles of the people who would use the system. However, this is not the complete picture. A customer is a very broad and general concept. In this simple example we can differentiate between several different types of customers: Some customers order their pizza over the phone and some come into the store to order their pizza; Some customers require their pizzas to be delivered, but others will pick them up from the store when they are ready. Further, there are customers whose information is held by the pizza store for reference when their pizzas need to be delivered. Clearly, one human being can be any of the types of customer we have just named, depending on

[1]COMP311: User Interface Design, Victoria University of Wellington, New Zealand (2004)

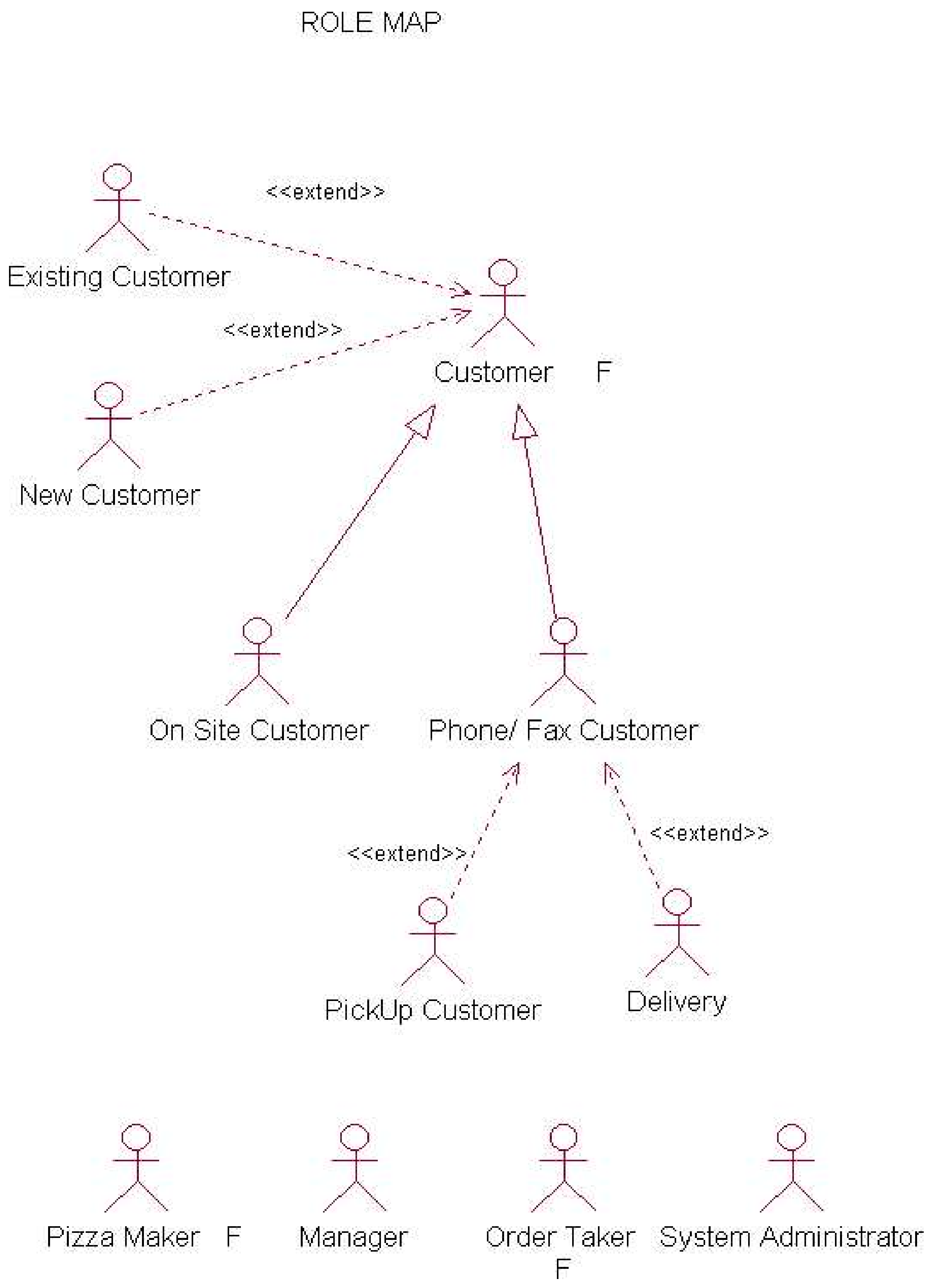
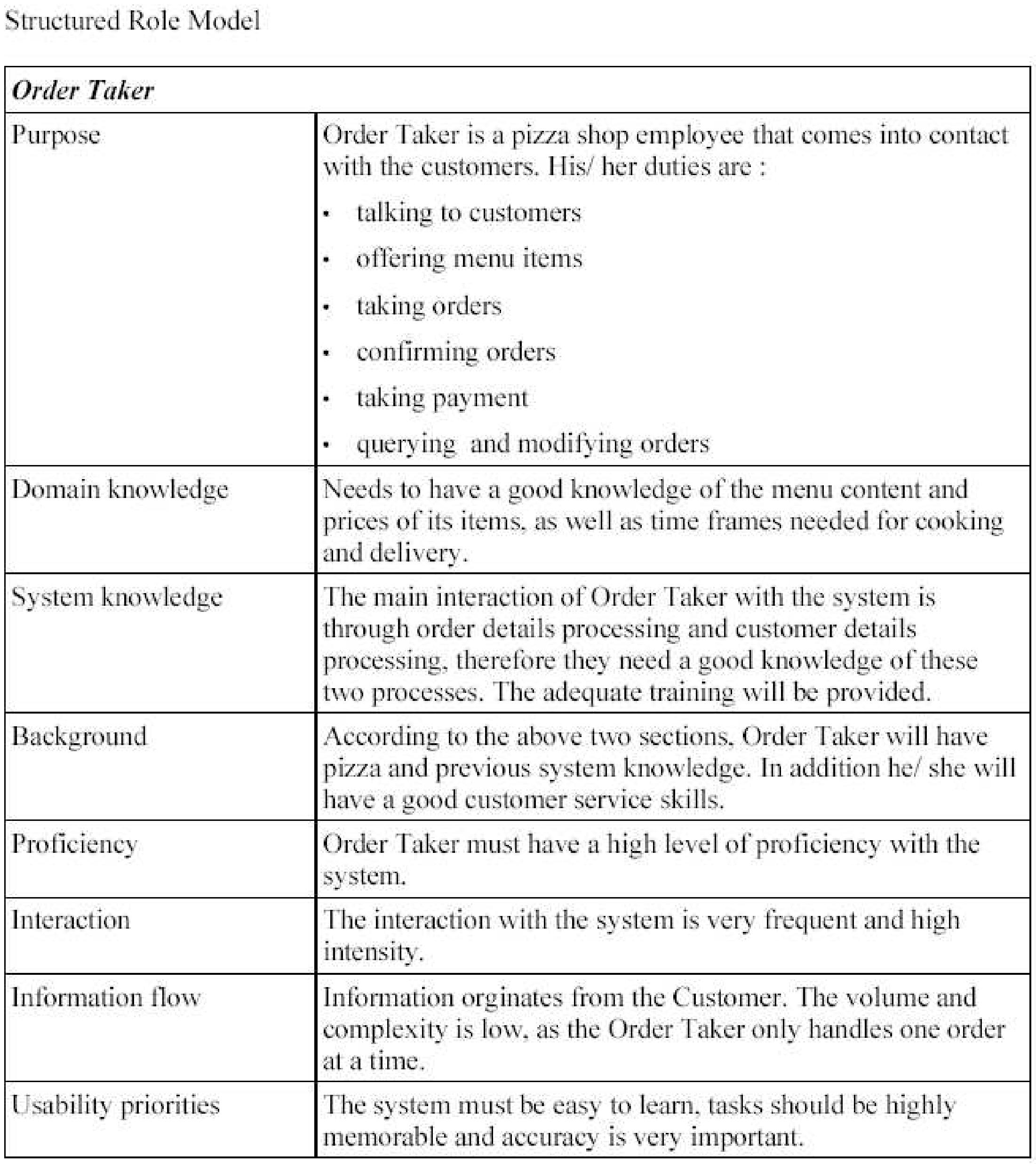
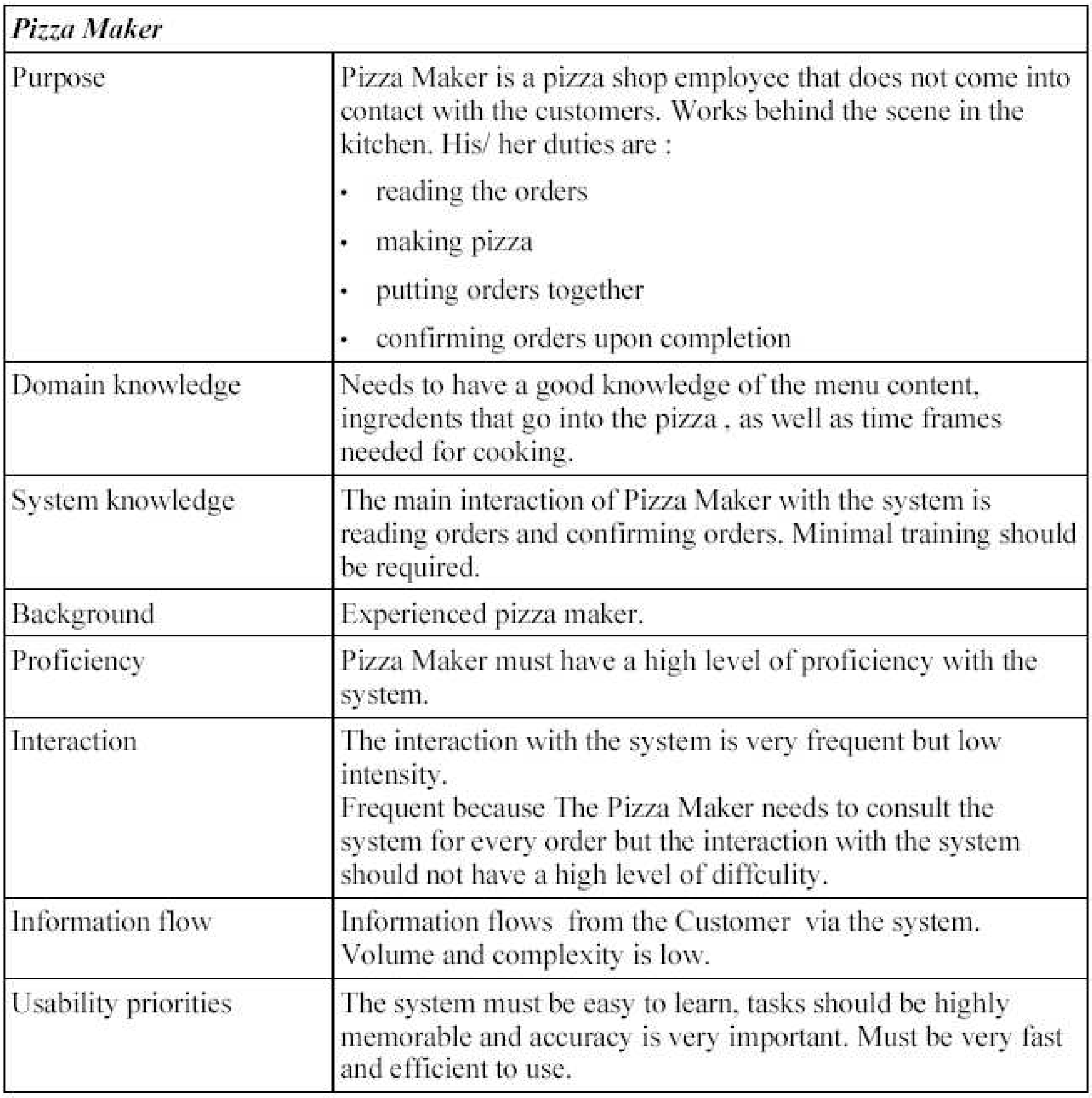


Figure 5.2: User Role Map (F = Focal Role)


Structured Role Model

Order Taker	
Purpose	Order Taker is a pizza shop employee that comes into contact with the customers. His/ her duties are : • talking to customers • offering menu items • taking orders • confirming orders • taking payment • querying and modifying orders
Domain knowledge	Needs to have a good knowledge of the menu content and prices of its items, as well as time frames needed for cooking and delivery.
System knowledge	The main interaction of Order Taker with the system is through order details processing and customer details processing, therefore they need a good knowledge of these two processes. The adequate training will be provided.
Background	According to the above two sections, Order Taker will have pizza and previous system knowledge. In addition he/ she will have a good customer service skills.
Proficiency	Order Taker must have a high level of proficiency with the system.
Interaction	The interaction with the system is very frequent and high intensity.
Information flow	Information orginates from the Customer. The volume and complexity is low, as the Order Taker only handles one order at a time.
Usability priorities	The system must be easy to learn, tasks should be highly memorable and accuracy is very important.

Figure 5.3: Structured Role Model for the Order Taker



Pizza Maker	
Purpose	Pizza Maker is a pizza shop employee that does not come into contact with the customers. Works behind the scene in the kitchen. His/ her duties are : • reading the orders • making pizza • putting orders together • confirming orders upon completion
Domain knowledge	Needs to have a good knowledge of the menu content, ingredents that go into the pizza , as well as time frames needed for cooking.
System knowledge	The main interaction of Pizza Maker with the system is reading orders and confirming orders. Minimal training should be required.
Background	Experienced pizza maker.
Proficiency	Pizza Maker must have a high level of proficiency with the system.
Interaction	The interaction with the system is very frequent but low intensity. Frequent because The Pizza Maker needs to consult the system for every order but the interaction with the system should not have a high level of diffculity.
Information flow	Information flows from the Customer via the system. Volume and complexity is low.
Usability priorities	The system must be easy to learn, tasks should be highly memorable and accuracy is very important. Must be very fast and efficient to use.

Figure 5.4: Structured Role Model for the Pizza Maker

what their particular preferences happen to be on the day they order their pizzas. It is necessary to differentiate between the types of customers in this way because the system under development and the user interface of that system will behave differently for each of these types. For example, if a customer requires their pizzas to be delivered, then the system needs to have their information available for the delivery person. This could be stored in a data base containing customer information. On the other hand, a customer who will pick their pizzas up from the store, do not need to supply their information to the system because it will not be required. The user interface should then allow the delivery customer's information to be entered into the system, but not insist on the information of the pick-up customer to be entered.

The next step was to decide what the focal roles of the pizza system was to be. Originally we identified the customer, the order taker and the pizza maker for our focal roles. This meant that during the design of the interface, we would implement all the tasks of the customer, order taker and pizza maker first, before considering implementing tasks for other roles. Our decision to use the customer, the order taker and the pizza maker for our focal roles was based on the consideration that these are the most vital people in a pizza business and that without them, the business could not exist or be recognised as a pizza business. The business can exist without a delivery person, or without a manager or system administrator (however chaotic that might be).

Finally, the structured role models were constructed. Since the focus from here on is now on the focal roles, these will be dealt with first and the structured role models for the other roles can be returned to once the implementation of the focal roles have been completed.

For constructing the focal roles for the pizza business, it was necessary to consider the purpose, domain knowledge, system knowledge, background, level of proficiency, type of interaction, information flow and usability priorities for each focal role. As it turned out, the customer, although chosen as a focal role, was never required to directly interact with the system. The order taker was in effect the 'interface' between the customer and the system. This insight was quickly realised when it became time to model the structured role models. Since nothing in the structured role model applied to the customer role, we quickly realised that the customer is not a role that would interact with the system directly. All customer interaction with the system would be mediated by the order taker role.

5.6.2 Semiotics of the User Role Model

As a whole, the user role model is a complex sign made up of smaller signs: the user roles, the user role map and the structured role models. The user role model gives an abstract overview of the users, their characteristics and the relationships among them. Thus, it acts as a sign that is used to convey information about the users of the system, to the model interpreters, who will use it to construct the next essential model of the UCD process — the *task model*.

In Peircean terms, the representamen of the user role map is the notation used to represent the user roles and the relationships between them. That role map provides a visual representation of this information. The object of the user role map is the actual potential human users of the system. The interpretant is the agreed understanding between designers of the UCD process of what the visual elements in the role map represent, i.e., the referent system for a role map.

The representamen of the structured role model is the table containing the text that describes the characteristics of that particular role. Thus, there is a likeness between the description of the role and the actual role itself. This would imply that the relationship between the representamen (the model system) and the object (the user roles of the application domain) is an iconic relationship and hence, the sign is an iconic sign. The guidelines given by Constantine and Lockwood [21] for constructing the profiles of the structured role model, are in effect guiding the iconic construction of roles as they exist in the application domain. These profiles aid the designers in building strong iconic models on which to base the rest of the user interface design process.

The user role map and the structured user role model both belong to the application domain of the UCD process, as it models the requirements or characteristics of the users. The role model is then the solution that becomes the requirements for the task model. This is strictly considering the role model as a sign belonging to the application domain.

If we allow no distinction between the solution and application domains, then the role model can be considered to have the interface under development as the object of the sign. Now the sign can be considered a symbolic sign, since it does not bear any likeness or similarity to its object — the interface under development.

5.7 Task Model

The advice given by Constantine and Lockwood is that the user interface design should closely fit the task model [21]. Basically, the task model is a representation of both *what* and *how* user tasks are performed. This is done by developing *essential use cases*[2], which are descriptions of one single interaction between a user and a system. From the user's point of view, they relate what steps are required to achieve a goal with the system. They are called essential use cases, rather than concrete use cases, because they steer clear of making assumptions about the concrete implementation details of the user interface involved in the user interaction. We turn to Constantine and Lockwood's definition of an essential use case: "An essential use case is a structured narrative, expressed in the language of the application domain and of users, comprising a simplified, generalized, abstract, technology-free Adin implementation-independent description of one task or interaction that is complete, meaningful and well-defined from the point of view of users in some role or roles in relation to a system and that embodies the purpose or intentions underlying the interaction." [21, p 103][3] They have found that essential use cases allow for the design of more creative solutions in interfaces, since they do not limit or restrict the designer in any way.

To make explicit the relationships between the essential use cases, the UCD process enlists the help of a UML (Unified Modeling Language)[4] notation — the use case diagram. In UCD it is called the *use case map*. The four relationships in the use case map and the use case diagram are described in very similar ways.

The result of task modeling is a picture of user-computer interactions couched in the context of user roles and tasks spelled out in design-free interaction narratives, and illustrating the inter-relationships among all of the individual use cases [45]. This 'picture' is then what designers will use to model a content model of the solution domain.

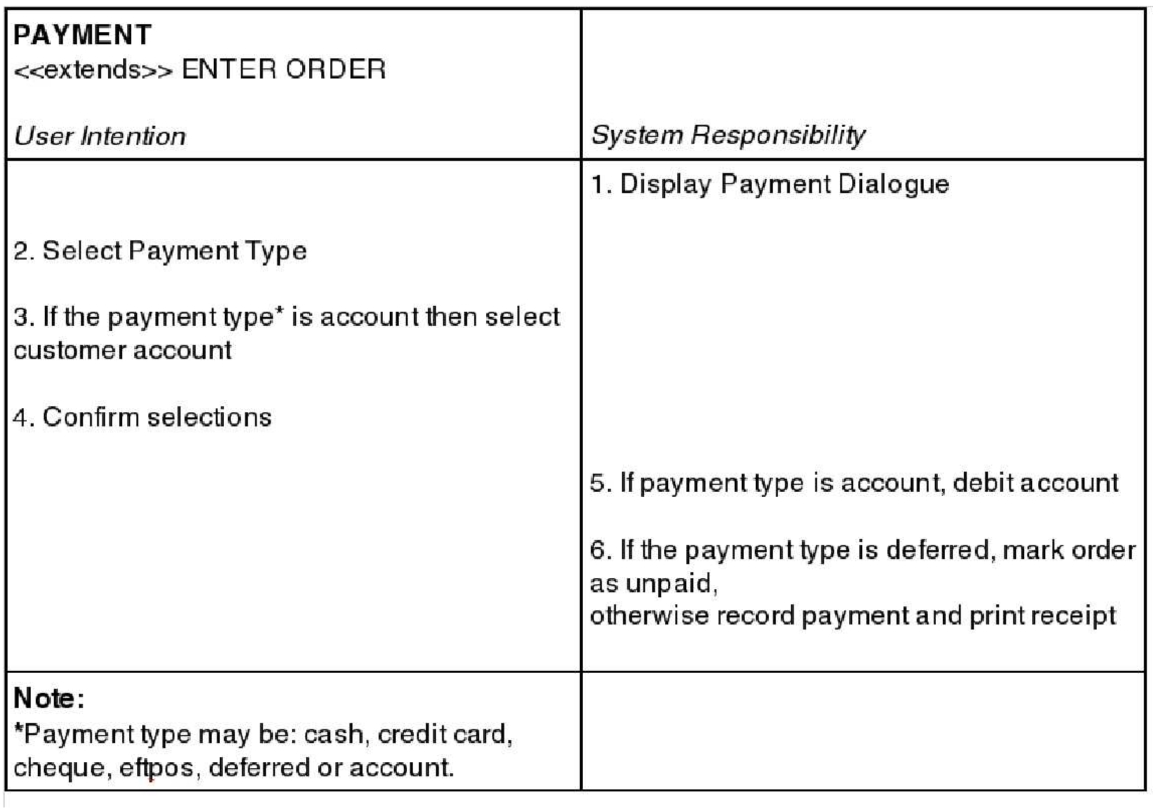
Returning to the running example, for brevity, we will focus on one user task required to be

[2]Both the terms 'use case' and 'essential use cases' are used interchangeably throughout this chapter

[3]There are several well-known methods for obtaining the use cases, e.g. textual analysis, observations, but these are not central to the discussion here, so they are omitted.

[4]A popular method for specifying, visualising and documenting the artifacts of an object-oriented system under development.

supported by the pizza business's user interface. We can choose any task belonging to the Order Taker or Pizza Maker, so we arbitrarily pick the Order Taker's task of handling the payment of an order. This task corresponds to a task mentioned in the structured role model in the 'Purpose' section of figure 5.3. The PAYMENT use case appears in figure 5.7. This use



PAYMENT <<extends>> ENTER ORDER *User Intention*	 *System Responsibility*
	1. Display Payment Dialogue
2. Select Payment Type	
3. If the payment type* is account then select customer account	
4. Confirm selections	
	5. If payment type is account, debit account
	6. If the payment type is deferred, mark order as unpaid, otherwise record payment and print receipt
Note: *Payment type may be: cash, credit card, cheque, eftpos, deferred or account.	

Figure 5.5: The Payment use case

case is invoked when the Order Taker confirms the customer's order (in Enter Order use case) for a pick-up order type.

The system displays the payment dialogue with payment type options as well as the payment value (determined automatically in the system by the order value). The Order Taker selects a payment type and confirms the selection. If the selected payment type is account, the account is credited; if the payment type is deferred (for Pick-up Customers) the order is marked as unpaid, and the flow of control is returned to Enter Order use case. In any successful case the payment is recorded and the receipt is printed.

Canceling a payment can be performed at any point during this use case and the flow of control will be returned to Enter Order use case.

If the customer is a Delivery Customer, payment type option cannot be eftpos. Payment types for all other customers are: cash, eftpos, credit card, cheque, account or deferred.

The use case map shows the relation of the PAYMENT use case with respect to the other possible use cases for the system. It is related to the Enter Order use case, by extending this use case. This use case can be interacted with directly by the Order Taker and communicates with the Query Order use case.

5.7.1 Semiotics of the Task Model

The essential use case as artifact is the representamen, the user task that is described by the use case is the object and once again, as with all the models of the UCD process, the interpretant is the referent system for essential use cases.

Considering the essential use case as an artifact of the task model, the essential use case is an indexical sign because the use cases contain references to the tasks that users will be performing with the interface. The essential use case is not iconic because the interactions are not concrete and do not resemble the actual interactions that will be performed with the eventual user interface. For example, in the PAYMENT use case in figure 5.7, step 2 on the User Intention side of the use case, 'Select Payment Type', resembles the same action that will eventually be implemented in the user interface only by reference. Since interpreters of the use case are not told *how* the user performs the selection,the interpreters can not generate an equivalent sign in their mind although they understand that it is an action referring to a later implementation.

OO-design places the use case firmly in the application domain but there is obviously a problem here. The essential use case is the link between the application domain and the solution domain. The User Intention part of the use case belongs to the application domain but the System Responsibility relates to the system that is to be implemented (which resides in the solution domain). The language of the essential use case is natural and intended to be easily understood by the potential users of the system, so that the use cases can be used as a way for designers to verify with the users that their tasks are represented correctly and that the system is responding to them in they way they expect. Use cases are also intended for unambiguous interpretation by the designers who require the use case to create the content model. This presents a problem: what domain does the use case model? There may be two answers to this question — either the use case is a model of another domain that may intersect the application and solution domain, or it is a model of both.

Considering the use case map as an artifact of the user tasks (represented by the use cases) and the relationships among them, this sign is a combination of indexical signs — the essential use cases — and symbolic signs — their interrelationships. Therefore the task model as a whole may be a sign existing somewhere between an indexical and symbolic sign.

5.8 Content Model

The intermediary step between designing the Task Model and a CAP, is the creation of a *content inventory*, designed using paper and sticky notes. One content inventory can consist of many *interaction contexts* and each interaction context (See figure 5.7 taken from [10]) contains *materials* and *tools*. The materials are the stuff the users want to see and manipulate and are represented by sticky notes in cool colours (blue, green). The tools enable users to do things with the materials and are represented by sticky notes in hot colours (pink, yellow, orange).

Roughly, each line in the use case corresponds to one sticky note. The following questions can help designers decide what materials are required in each interaction context:

1. "What information will the user need to create, examine or change?"

2. "What information will the user need to present or offer to the system?"

3. "What information will the user require from the system?"
4. "What views, perspectives or forms of presentation will be of value to users?"
5. "What conditions or events will the user need to convey to the system?"
6. "About what conditions or events will the user need to be informed?"

Whereas the following questions can help designers decide what tools are required in each interaction context:

1. "How will users need to operate on or change the information of interest on the user interface?"
2. "What capabilities or facilities will need to be supplied by the user interface for users to be able to accomplish their tasks?"

Using sticky notes for representing *materials* and *tools*, and sheets of paper for the interaction context, the sticky notes act as placeholders for actual user interface components, so this process helps designers decide how to organise the user interface features. The interaction contexts combine tools and materials together into useful collections[10] and asking the appropriate questions listed above, along with the essential use cases determine what the content inventory will consist of. Hence, designers select materials and tools so that the tasks in the essential use cases can be performed. These are then organised in such a way that each use case can be performed in roughly one interaction context, although it may be more sensible to allow closely related use cases to be performed in the same interaction context. Closely related use cases can be identified by the use case map, which illustrates the relationships among the use cases.

In the process of designing the pizza business's user interface, this artifact was not expected to remain a permanent model intended to be used for future reference like the previous models we have discussed. In our group experience, the content inventory served the function of a prototype of the prototype — or CAP. It allows for the quick and cheap exploration of possible prototype designs and was discarded when the final prototype design had been decided on.

5.8.1 Semiotics of the Interaction Context

The interaction context is the first UCD model that can be used to explore the solution domain, before the CAP is created. With the interaction context model as the representamen, the interpretant the referent system for the interaction context and the object a possible realisation of a user interface — with widgets and functionality that correspond to the tools and materials that appear in the context model — this sign is a symbolic sign. The representamen does not resemble the object in this sign.

Not only can the text on the sticky notes convey messages to the model interpreters, but also the colours used for tools and materials. The value of having materials and tools each represented by sticky notes of colours which appear next to each other on the colour wheel (figure 5.8), creates a sense of unity among the components. Greens and blues appear next to each other on the colour wheel and are known as *analogous colours*. These colours have the effect of creating unity among the components and conveys to the viewer that the sticky notes of analogous colours belong to the same category of component, i.e., materials. On the other hand, the fact that hot and cold colours appear on the opposite sides of the colour wheel reinforces the idea that they are two separate categories [34]. Therefore, although the interaction context as artefact will probably only exist until a satisfactory CAP has been designed, the information that it needs to convey to the designers of the CAP should still be clearly represented. Making use of analogous colours for materials and tools but complementary colours to distinguish between the material and tool components, visually aids the information the interaction context is designed to convey.

The content inventory is obviously not ideal as a technical visual language but its advantage is that it is easy to change and assemble. There is a sense of temporariness about using paper models and this is ideal in an environment that requires the exploration of different options as quickly and cheaply as possible.

5.9 From Interaction Context to Canonical Abstract Prototype

Canonical Abstract Prototypes (CAPs) were developed in order to provide a better transition for designers who have to move from the abstract design model to the actual interface (See figure 5.9). As part of the UCD process, they are the "bridge" between the Task Model and the realisation of the design[20]. Constantine[19] lists the main strength of CAPs as being able to separate decisions about:

- What information and functions are presented
- How the UI is arranged and organised
- What widgets are used
- Precise appearance and behaviour

The CAP is composed of *canonical abstract components*. There are three types and these are: a *generic tool* glyph and a *material glyph*. These two can be combined into a third component called a *active materials*. These tools and materials have been designed to cover much functionality and the freedom with which they can be combined to form new components makes them a very expressive notation. Constantine, Windl, Noble and Lockwood [18] provide these canonical abstract components in figures **??**, 5.11 and 5.12.

The CAP also has all the other well-known benefits of prototyping as part of the software design process, namely, the quick and cheap exploration of possible solutions without heavy resource investments.

The pizza business's interface required a way of recording payments for orders. The PAYMENT use case contains all the requirements that would be needed to design some kind of interface element that would have the required features. From inspecting the use case it was clear that a user would need a tool for selecting the type of payment that the customer presented, an element material to display the amount owed by the customer, an active selectable collection in order to select the customer's account from a list of customer accounts kept in a database, a tool to exit (delete/erase tool in figure 5.11 and finally, a select tool that would confirm any choices made in the payment dialogue. Figure 5.9 is the CAP form of the Payment Dialogue and figure 5.13.

5.9.1 Semiotics of Canonical Abstract Prototypes

The CAP is composed completely of symbolic signs (canonical asbtract components) and yet, the CAP is closer to iconicity on the icon-index-symbol continuum than any of the other models discussed above. This is because new information is added to the model — size and positioning of user interface features. This is achieved by still using symbolic representations of the features but their size and position are more accurately determined. Precise details such as colour and borders etc still remain undetermined, and so it is not yet fully iconic.

Practice has shown that this is usually a one-to-one mapping between the canonical abstract component and the concrete user interface element, unless some external factor such as testing shows that additions are required in order to make the interface more usable.Once the CAP has been designed, a realistic prototype can be developed — which is iconic in the same way that a thumbnail is an iconic representation of a photograph — and subsequently the actual user interface.

From a semiotic perspective, it is not important whether we the UCD process as ending with the realistic prototype, which exactly resembles the user interface to be implemented from its design, or whether the UCD process concludes once the actual final user interface has been implemented. This is because both are iconic representations of the final user interface.

5.10 Discussion

It has become clear that the UCD process makes use of models in order to first of all model the application domain in the form of an iconic sign — the role model. Then the solution to the application domain, i.e., the role model, is used to construct a new model, the task model. This task model is an indexical sign of the user tasks that will be supported in the interface under development. The task model bridges the gap between the application domain and the as yet unexplored solution domain. It is not very surprising then that Constantine and Lockwood call the task model "the very core of the usage-centered design process." [21, p 35] From the task model, the abstract content model and the less abstract CAP are constructed, both of which model the solution domain. The content model is a way of organising the interaction between the user and the system and by the time the content model is created, the user tasks have already been elicited and modeled, so that it is just a matter of ensuring that the required interaction does not make unreasonable demands on the user. The content model and the canonical abstract prototype allow designers to explore the solution domain quickly and cheaply. If the derived realistic prototype is a close fit to the task model, then it is obvious that the realistic prototype should adhere to all the user requirements elicited during the design of the role model.

Through our semiotic analysis we have observed two different characteristics of the transitions between the signs in each of the domains during the UCD process. If we take the movement from icon to index to symbol as a movement from firstness, to secondness, to thirdness, we can make the following observations: In the solution domain we begin with establishing a well defined iconic representation of the user roles the interface is required to support, and observe a movement in the next model toward an indexical representation of the task model. Movement is continued in the same direction from the indexical task model, to the symbolic content model. However, the solution domain requires the designer to move in the opposite direction to the movement between models in the application domain, i.e., from symbolic models to more iconic representations of the realistic user interface.

Thus, Usage-Centered Design is a process that takes the designer from an iconic representation of the user roles toward indexical signs of the tasks in the user interface further toward the symbolic interaction contexts. Then there is a movement back toward the iconic representation of the interface in the form of a realistic prototype.

Semiotically, as explained by the studies in chapter 3, people seem to understand iconic signs better and when they are in doubt they revert to an iconic representation. This would seem to support the success of the UCD process due to the movement from the well understood iconic representation of user role models, on which the future models are based. This implies that a sound basis for future modeling is ensured.

Before concluding it is worth noting that all the models can be seen as indexical signs — they all refer to the interface that is being developed and thus, their presence conveys to interpreters of the models that an interface is under development. Yet, as noted before, each model's domain has to be taken into account, so that the underlying object of the peircean triadic relationship can be established.

5.11 Summary

We conclude this chapter with the observation that the success of the UCD process lies with the iconic first model that gives a well understood basis on which to base further more refined models.

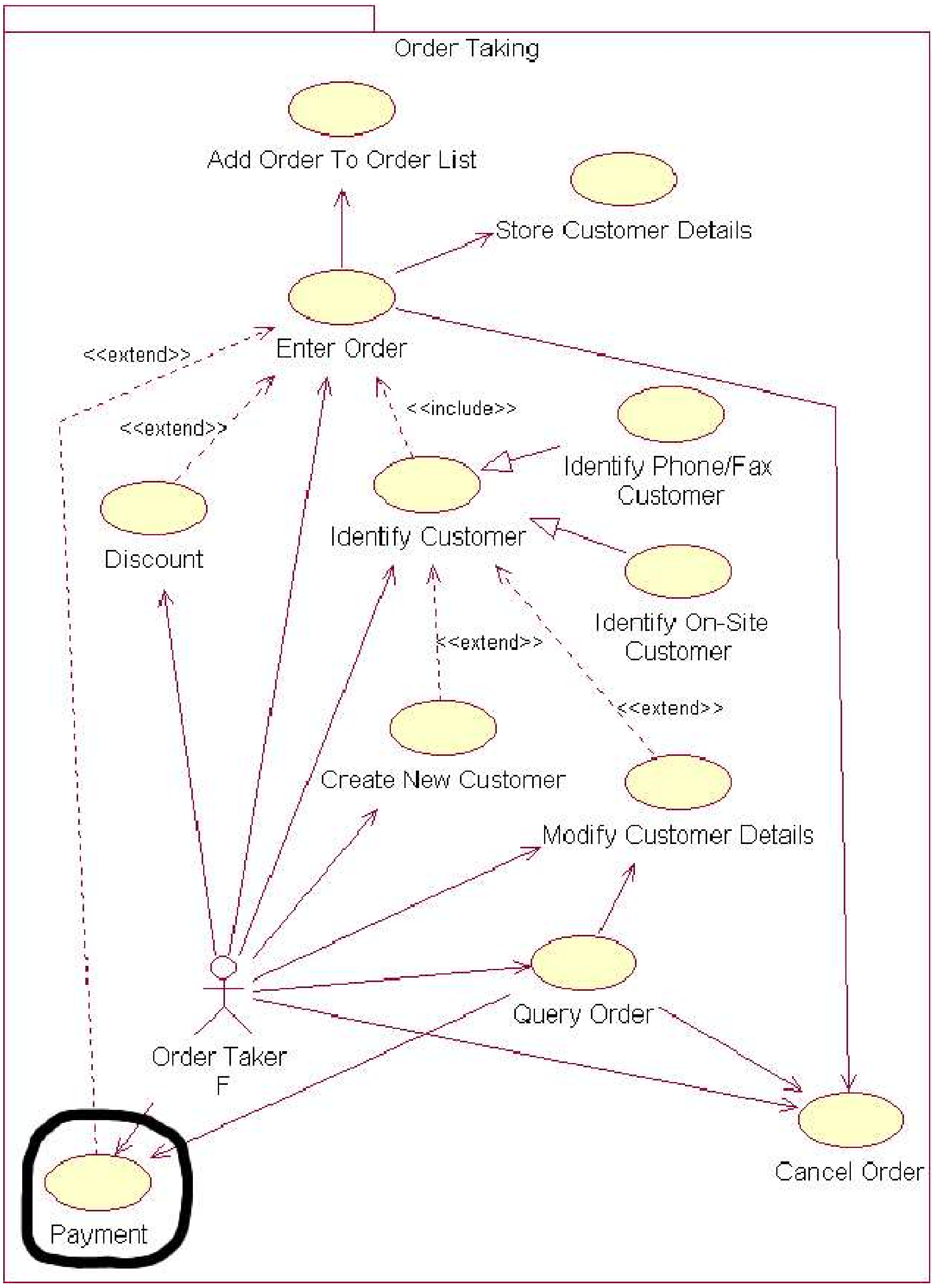


Figure 5.6: Use case map for the Order Taker

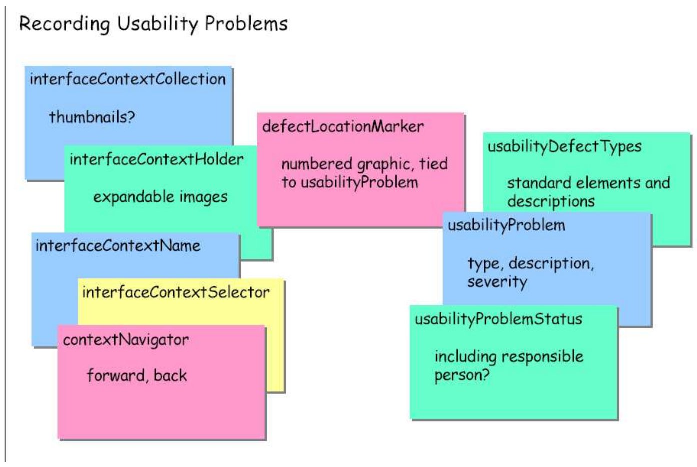


Figure 5.7: From [18]: Example of an Interaction Context

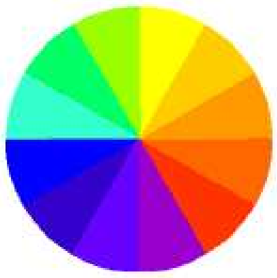


Figure 5.8: From [34]: The colour wheel

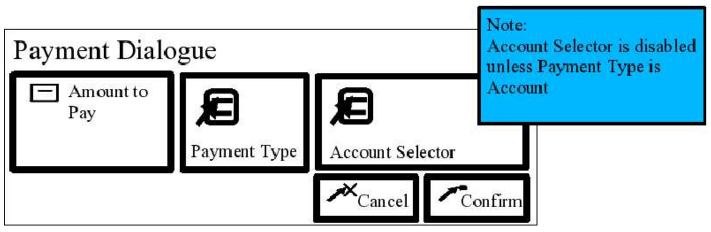
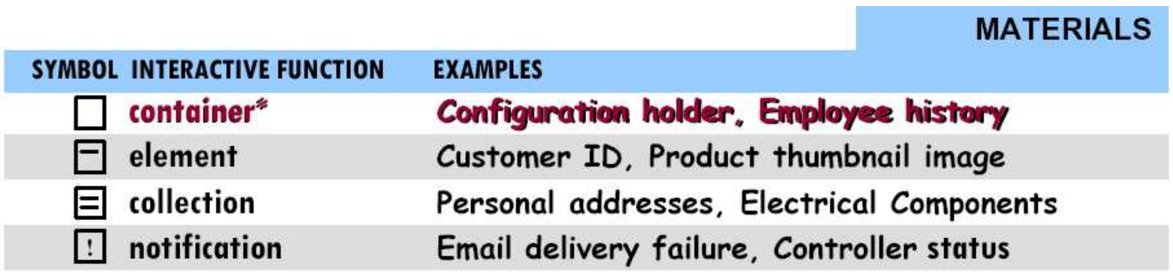
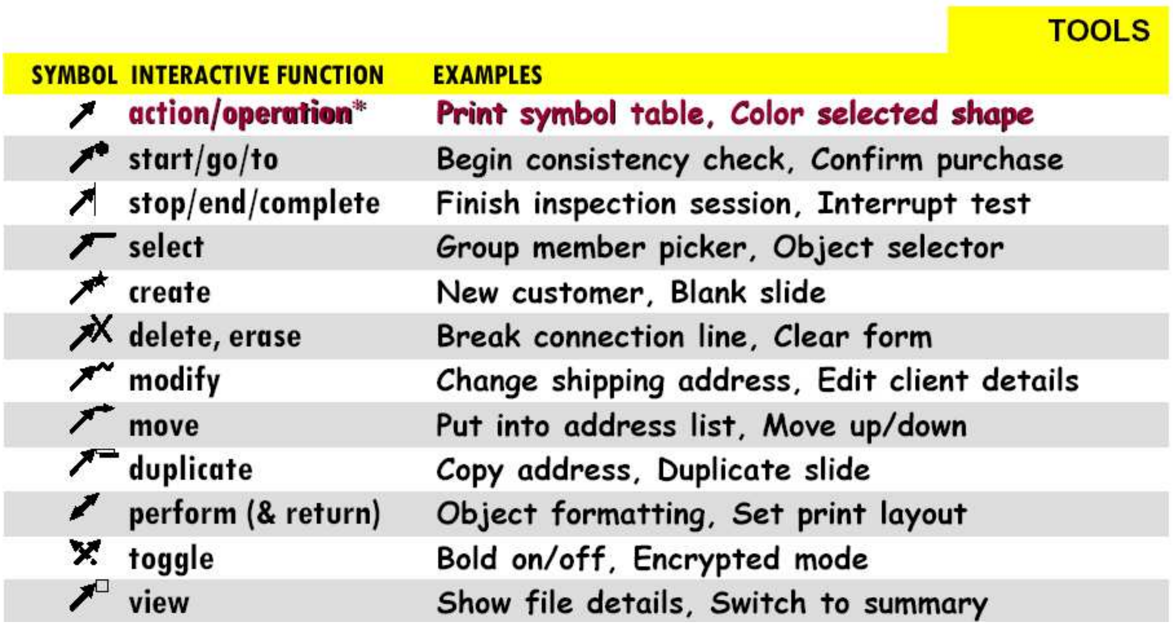


Figure 5.9: Canonical Abstract Prototype of the Payment Dialogue



		MATERIALS
SYMBOL	INTERACTIVE FUNCTION	EXAMPLES
☐	container*	Configuration holder, Employee history
⊟	element	Customer ID, Product thumbnail image
☰	collection	Personal addresses, Electrical Components
[!]	notification	Email delivery failure, Controller status

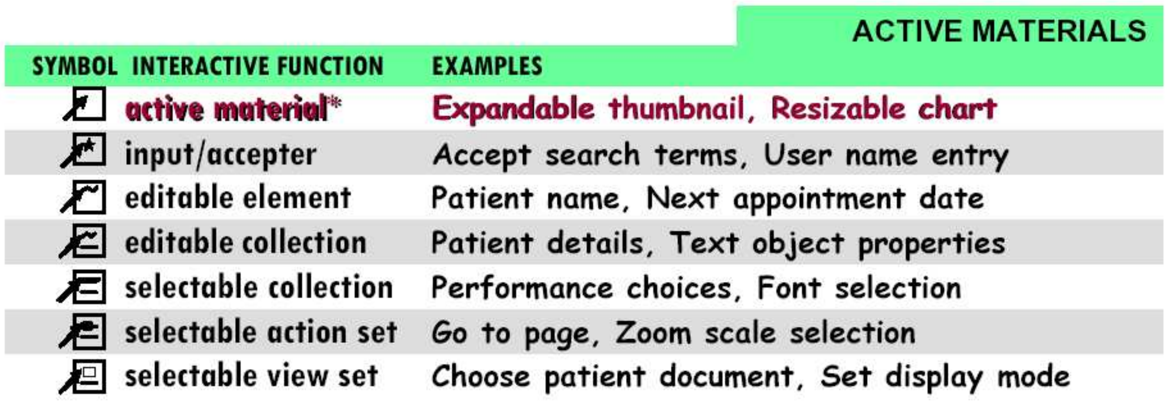
Figure 5.10: From [18]Canonical Abstract Components — Materials



TOOLS

SYMBOL	INTERACTIVE FUNCTION	EXAMPLES
	action/operation*	Print symbol table, Color selected shape
	start/go/to	Begin consistency check, Confirm purchase
	stop/end/complete	Finish inspection session, Interrupt test
	select	Group member picker, Object selector
	create	New customer, Blank slide
	delete, erase	Break connection line, Clear form
	modify	Change shipping address, Edit client details
	move	Put into address list, Move up/down
	duplicate	Copy address, Duplicate slide
	perform (& return)	Object formatting, Set print layout
	toggle	Bold on/off, Encrypted mode
	view	Show file details, Switch to summary

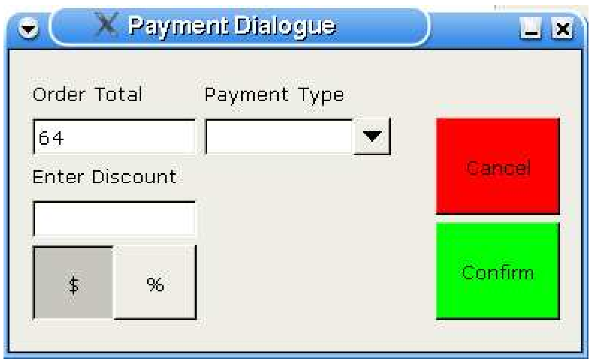
Figure 5.11: From [18]Canonical Abstract Components — Tools



ACTIVE MATERIALS

SYMBOL	INTERACTIVE FUNCTION	EXAMPLES
	active material*	Expandable thumbnail, Resizable chart
	input/accepter	Accept search terms, User name entry
	editable element	Patient name, Next appointment date
	editable collection	Patient details, Text object properties
	selectable collection	Performance choices, Font selection
	selectable action set	Go to page, Zoom scale selection
	selectable view set	Choose patient document, Set display mode

Figure 5.12: From [18]Canonical Abstract Components — Active Materials



Order Total: 64

Figure 5.13: Concrete implementation of the Payment Dialogue

Chapter 6

Conclusions

In the introductory chapter to semiotics and user interface design (chapter 2), we briefly introduced semiotics as the study of signs and selected Peirce's model for the rest of the semiotic analysis that was performed. This decision was based on two criteria: first, Peirce's model is a highly structured model that takes into account three parts of the sign; second, other semioticians have selected it as a good model to use when considering user interface signs.

In chapter 3 we quote Goguen: "all else being equal: Icons are better than indices, and indices are better than symbols" [27], but he provides no evidence for this claim. In this chapter we showed that several studies do support the notion that icons are better than symbols, however icon evaluation is a very difficult task. We summarise our most important findings as follows:

1. Some concepts are better explained by symbolic signs, for example, rotation of an object.

2. The pairs of opposite actions of sending and receiving, increasing and decreasing, answering and hanging up are very difficult to convey.

3. Participants are more likely to attribute an action to an icon than guess that it represents a system object.

4. When participants can not think of a suitable functionality that fits into the given context, they will base their guess on the visual clues that they can see and so, interpret the icon based on what they think they see. This was most clearly illustrated by icon 4.

5. Iconic signs generally are better recognised than their symbolic counterparts and in the case where participants interpret the iconic signs incorrectly, they are surprisingly consistent in their interpretations of the image.

In the words of de Souza [24], "our conclusions at this point is more a matter of speculation and sense than of thorough testing and analysis". But the major insight gained from modeling user interface redesigns semiotically, was that the redesign tended to change the existing sign from a symbolic into an indexical or iconic sign. Research into the intuitiveness of computer icons from chapter 3ends to support this move and indicates that users tend to interpret the signs they see as iconic signs.

We concluded chapter 5 with the observation that the success of the Usage-Centered Design process lies with the iconic first model that gives a well understood basis on which to base further more refined models. Again, this observation is substantiated by the results from the studies in 3.

From the project as a whole, it is possible to conclude that if the results from the studies do support that users generally understand iconic signs more easily, then any user interface concept, be it a functionality represented in the interface or a model of some domain, is consequently better understood when modeled as an iconic sign. We can also further use this information to find explanations for the reasons user interface redesigns happen the way they do and why some design processes, such as Usage-Centered Design is so successful at producing usable interfaces. Overall, then the conclusions are valuable for user interface design.

Although not directly related to the research in the main body of the project report, we conducted a heuristic evaluation, the results of which would help to improve the usability of the stereo system in the Memphis computer lab.

Possible Criticisms of the Semiotic Approach

There are two major criticisms that could be levelled at the semiotic approach applied in computer science:

- Predicting another person's interpretant based on a given object and representamen is near impossible. Each individual will make their interpretation based on the context in which they find themselves at that point in time. Their specific context is based on their culture, background and education level. This makes any analysis involving the interpretant merely speculative. While it is true that some interpretants are more likely to be formed than others given the representamen, this will still only be valid within a culture, community group or individual [40].
- Semiotic analysis is not a technique that can stand on its own in a technical field such as computer science. It must always be backed up by empirical evidence, otherwise it is just a subjective theories.

Appendix A

Paper accepted for publishing by the Australian User Interface Conference 2005

FERREIRA, J., NOBLE, J., and BARR, P. The Semiotics of User Interface Redesign to appear at the 6th Australasian User Interface Conference (AUIC2005), Newcastle. Conferences in Research and Practice in Information Technology, Vol. 40. M. Billinghurst and A. Cockburn, Eds.

This paper is primarily my own work and presents some interim results of the project described in this report.

Appendix B

Icon Intuitiveness Test

Appendix C

Heuristic Evaluation Documents for the Stereo System in Memphis

Appendix D

Heuristic Evaluation of the Stereo System in Memphis

D.1 Introduction

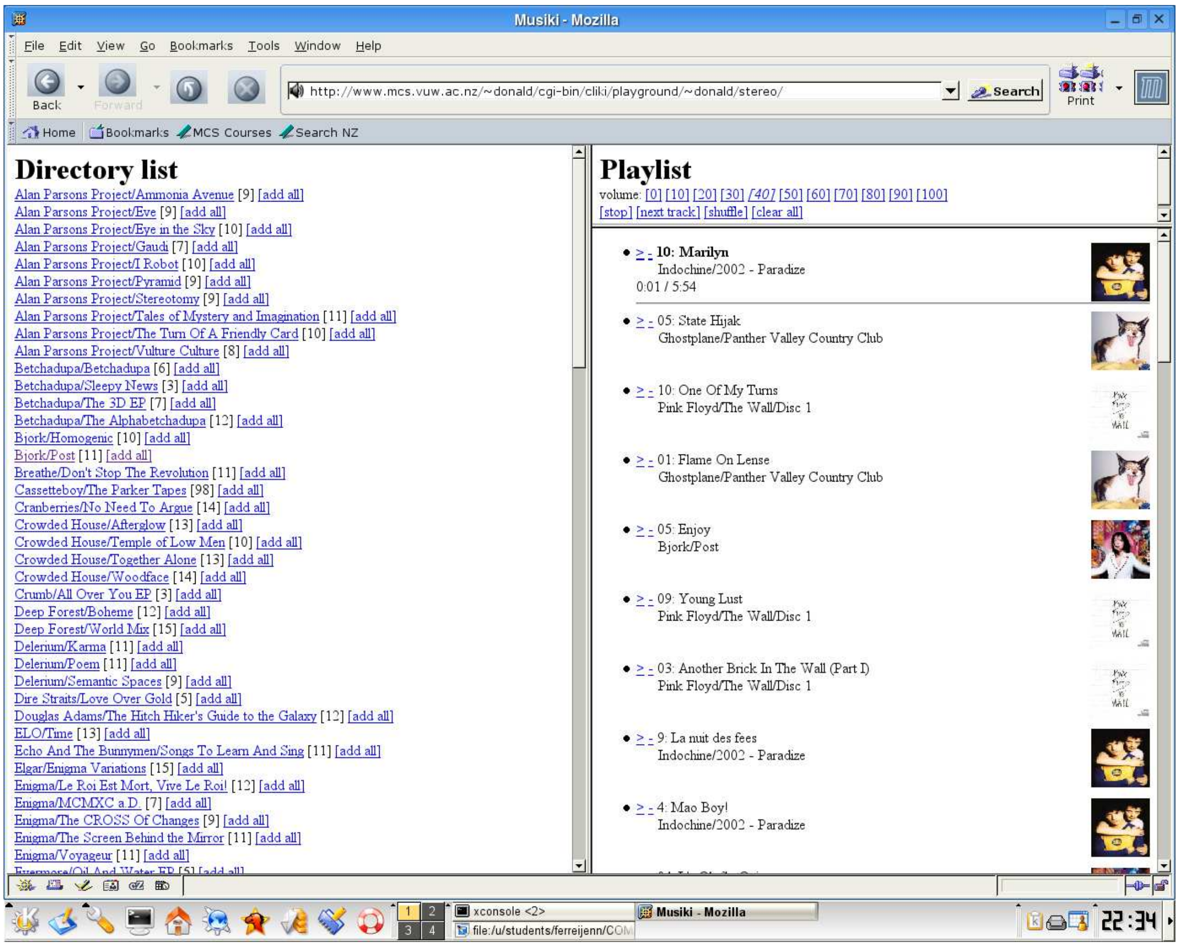
A heuristic evaluation is an excellent method for finding usability problems in a user interface when the results are required quickly, cheaply and requires no intensive training in order to be able to conduct one. It is known as a *discount engineering method* and was first advocated by Nielsen, to be performed as part of the user interface design process [38]. A heuristic evaluation is usually conducted by selecting usability experts, supplying them with the heuristics[1] the interface to be evaluated should conform to and then allowing them to inspect the interface and evaluate which heuristics are violated. This process can be very useful to user interface designers because user interface designers are not always usability experts and they may not have access to the future users of their system in order to conduct user testing.

According to studies carried out by Nielsen [38], the ideal number of evaluators for a heuristic evaluation is three to five, which makes it more likely that all the major usability problems will be identified. This is due to the fact that not all evaluators find the same problems in a given interface. He further suggests that evaluators do not communicate during the evaluation, as this also helps evaluators find distinct usability problems. However, after the evaluation has been carried out, the evaluators can have a debriefing session in which design advice can be exchanged.

We carried out a heuristic evaluation of the stereo system interface in the Memphis computer lab. Figure D.1 shows a screenshot of the interface as it appeared at the time of the evaluation. The evaluators were required to evaluate the interface using Nielsen's well-known ten usability heuristics and then rate the severity of the problem according to a given severity scale.

This chapter gives a brief introduction to the stereo system interface, describes the methodology used in the heuristic evaluation and then summarises the results. The document distributed to the evaluators during the evaluation, along with Nielsen's ten usability heuristics, can be found in appendix B.

[1]Guidelines



http://www.mcs.vuw.ac.nz/~donald/cgi-bin/cliki/playground/~donald/stereo/

Directory list

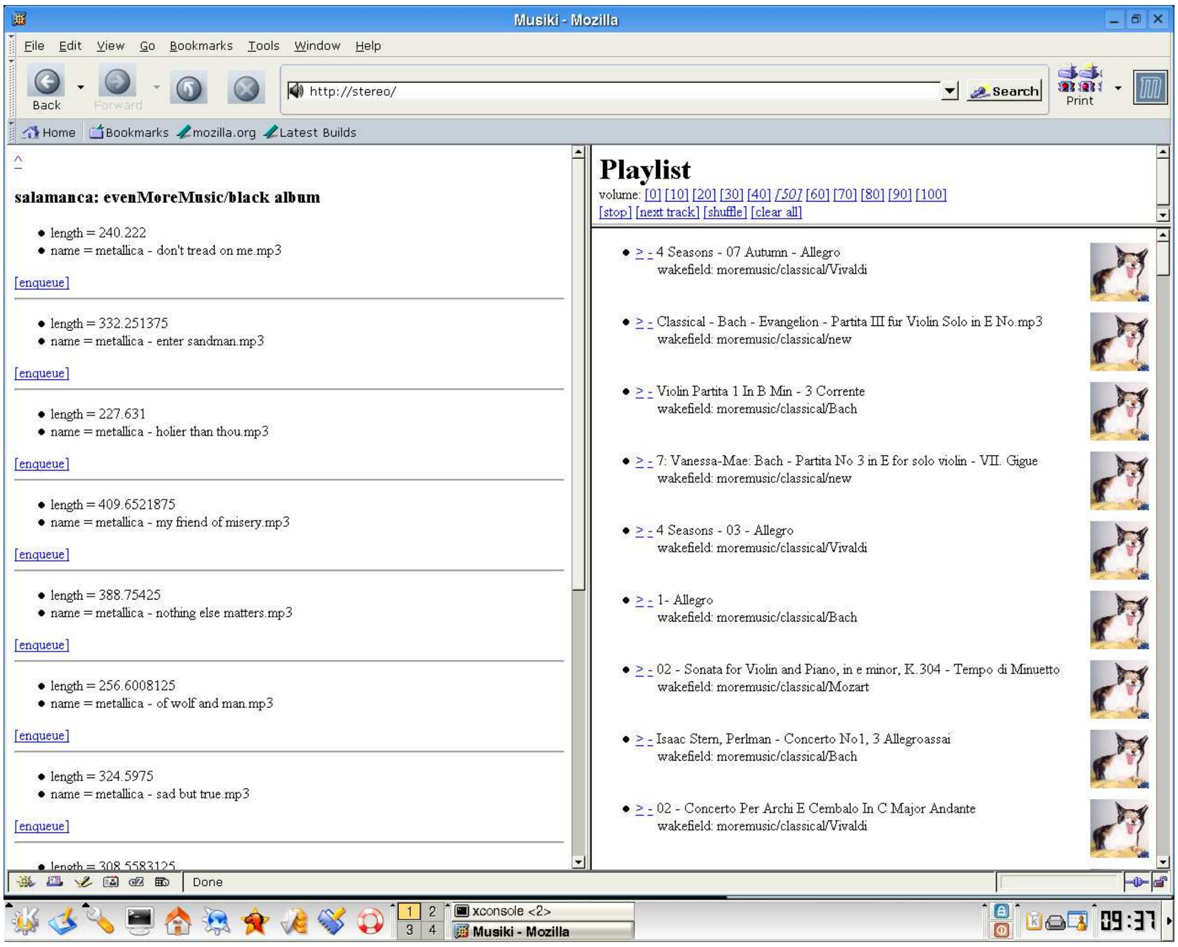
Alan Parsons Project/Ammonia Avenue [9] [add all]
Alan Parsons Project/Eve [9] [add all]
Alan Parsons Project/Eye in the Sky [10] [add all]
Alan Parsons Project/Gaudi [7] [add all]
Alan Parsons Project/I Robot [10] [add all]
Alan Parsons Project/Pyramid [9] [add all]
Alan Parsons Project/Stereotomy [9] [add all]
Alan Parsons Project/Tales of Mystery and Imagination [11] [add all]
Alan Parsons Project/The Turn Of A Friendly Card [10] [add all]
Alan Parsons Project/Vulture Culture [8] [add all]
Betchadupa/Betchadupa [6] [add all]
Betchadupa/Sleepy News [3] [add all]
Betchadupa/The 3D EP [7] [add all]
Betchadupa/The Alphabetchadupa [12] [add all]
Bjork/Homogenic [10] [add all]
Bjork/Post [11] [add all]
Breathe/Don't Stop The Revolution [11] [add all]
Cassetteboy/The Parker Tapes [98] [add all]
Cranberries/No Need To Argue [14] [add all]
Crowded House/Afterglow [13] [add all]
Crowded House/Temple of Low Men [10] [add all]
Crowded House/Together Alone [13] [add all]
Crowded House/Woodface [14] [add all]
Crumb/All Over You EP [3] [add all]
Deep Forest/Boheme [12] [add all]
Deep Forest/World Mix [15] [add all]
Delerium/Karma [11] [add all]
Delerium/Poem [11] [add all]
Delerium/Semantic Spaces [9] [add all]
Dire Straits/Love Over Gold [5] [add all]
Douglas Adams/The Hitch Hiker's Guide to the Galaxy [12] [add all]
ELO/Time [13] [add all]
Echo And The Bunnymen/Songs To Learn And Sing [11] [add all]
Elgar/Enigma Variations [15] [add all]
Enigma/Le Roi Est Mort, Vive Le Roi! [12] [add all]
Enigma/MCMXC a.D. [7] [add all]
Enigma/The CROSS Of Changes [9] [add all]
Enigma/The Screen Behind the Mirror [11] [add all]
Enigma/Voyageur [11] [add all]

Playlist

volume: [0] [10] [20] [30] *[40]* [50] [60] [70] [80] [90] [100]
[stop] [next track] [shuffle] [clear all]

- > - **10: Marilyn**
  Indochine/2002 - Paradize
  0:01 / 5:54
- > - 05: State Hijak
  Ghostplane/Panther Valley Country Club
- > - 10: One Of My Turns
  Pink Floyd/The Wall/Disc 1
- > - 01: Flame On Lense
  Ghostplane/Panther Valley Country Club
- > - 05: Enjoy
  Bjork/Post
- > - 09: Young Lust
  Pink Floyd/The Wall/Disc 1
- > - 03: Another Brick In The Wall (Part I)
  Pink Floyd/The Wall/Disc 1
- > - 9: La nuit des fees
  Indochine/2002 - Paradize
- > - 4: Mao Boy!
  Indochine/2002 - Paradize

Figure D.1: Screen 1 of the stereo system in Memphis



^

salamanca: evenMoreMusic/black album

- length = 240.222
- name = metallica - don't tread on me.mp3

[enqueue]

- length = 332.251375
- name = metallica - enter sandman.mp3

[enqueue]

- length = 227.631
- name = metallica - holier than thou.mp3

[enqueue]

- length = 409.6521875
- name = metallica - my friend of misery.mp3

[enqueue]

- length = 388.75425
- name = metallica - nothing else matters.mp3

[enqueue]

- length = 256.6008125
- name = metallica - of wolf and man.mp3

[enqueue]

- length = 324.5975
- name = metallica - sad but true.mp3

[enqueue]

- length = 308.5583125

Playlist

volume: [0] [10] [20] [30] [40] *[50]* [60] [70] [80] [90] [100]
[stop] [next track] [shuffle] [clear all]

- > - 4 Seasons - 07 Autumn - Allegro
  wakefield: moremusic/classical/Vivaldi
- > - Classical - Bach - Evangelion - Partita III fur Violin Solo in E No.mp3
  wakefield: moremusic/classical/new
- > - Violin Partita 1 In B Min - 3 Corrente
  wakefield: moremusic/classical/Bach
- > - 7: Vanessa-Mae: Bach - Partita No 3 in E for solo violin - VII. Gigue
  wakefield: moremusic/classical/new
- > - 4 Seasons - 03 - Allegro
  wakefield: moremusic/classical/Vivaldi
- > - 1- Allegro
  wakefield: moremusic/classical/Bach
- > - 02 - Sonata for Violin and Piano, in e minor, K.304 - Tempo di Minuetto
  wakefield: moremusic/classical/Mozart
- > - Isaac Stern, Perlman - Concerto No1, 3 Allegroassai
  wakefield: moremusic/classical/Bach
- > - 02 - Concerto Per Archi E Cembalo In C Major Andante
  wakefield: moremusic/classical/Vivaldi

Figure D.2: Screen 2 of the stereo system in Memphis

D.2 Stereo System Interface

The interface that was evaluated in the heuristic evaluation is the interface for the stereo system in the Memphis computer lab. This is a web-based interface that allows students in the lab to play music through the sound system. The interface allows access to albums, consisting of individual songs, stored in a database. Whole albums or individual songs can be selected to be added to a playlist which plays the songs and when it reaches the last song on the list it loops back to the first song and continues in this way.

The interface consists of two screens: screen 1 (see figure D.1) is the screen the user sees when the url is entered into the browser. Screen 2 (see figure D.2) is the screen displayed once the user has clicked on the name of an album.

Access to the interface is unrestricted but the sound system is only set up for playing music in the lab.

D.3 Methodology

The heuristics used in this evaluation were Nielsen's ten usability heuristics. In light of Nielsen's suggested number of evaluators, four evaluators who had never encountered the stereo system before, took part in the evaluation. The evaluators were asked to list the problems they found with the interface, i.e., the heuristic violations, and then to rate the severity of the problems on a severity scale that was supplied. This was paper based. The advantages of supplying a severity scale is so that problems can be prioritised in a convenient manner and if it is not possible to have a discussion after the evaluation with all evaluators involved (as was the case in this evaluation) then the severity ratings can be used to identify the major problems. The heuristic evaluation was also non-task orientated, which means that evaluators were required to check for heuristic violations on every screen and on every action. This is only appropriate if the interface is very simple. We felt that the stereo system interface was simple enough for the evaluators to explore and evaluate all the functionality within a reasonable time frame[2].

Not all the evaluations were performed at the same time but there was an observer present in all cases. The observer was mainly required to distribute the evaluation documents, help the evaluators access the interface on the web and ensure that the evaluations were running smoothly.

We feel confident that the major usability problems were identified during this evaluation and present the results in the following section.

D.4 Results

Not all the heuristic violations found by each evaluator are listed below, only those violations with an average severity rating of 7 or above. The average was simply calculated by taking the mean of the four evaluators' severity ratings. The problems in the table are ranked according to severity, starting with the most severe.

---

[2]The evaluators were occupied for roughly an hour.

	Problem	Average severity	Heuristic violated
1	No help or documentation	9	H10
2	Accidentally enqueued albums can not be removed — the songs must be dequeued individually	9	H3, H9
3	The number of songs on the playlist is unknown	9	H1
4	The numbering of songs only apply to the album, not its position in the playlist	9	H1, H4
5	Selecting shuffle changes the currently playing song shown in the playlist, but only when '>' is selected, does it play	9	H1, H2, H4
6	Clicking with the mouse pointer in the white space of the playlist or highlighting the song crashes the interface	8.5	H9
7	There is a weak connection between the term directory list and as alist of audio selections and the term directory used in the real world.	8	H2
8	Functionality of the glyph '-' is unclear	8	H2
9	Volume control does not conform to well-known interface conventions	8	H4, H2
10	There is no way of enqueing all songs on an album from the screen showing the individual songs	8	H3
11	As the playlist becomes large it is not clear where in the playlist the songs are added	7.67	H1, H4
12	No way of recovering from a user interface crash	7.5	H9
13	No instructions exist as to how to proceed or use the interface	7.5	H10
14	Functionality of the glyph '$\wedge$' is unclear	7	H2
15	The currently playing song is not visible at all times	7	H1
16	If a song, say on the second album is selected, that songs that appear before that one on the album move to the end of the playlist, as though they have recently been played.	7	H7

Table D.1: Results of the heuristic evaluation of the stereo system

D.5 Areas for Improvement

This heuristic evaluation could have been improved in the following ways:

- Require that evaluators not only specify the heuristics that are violated, but also the heuristics that the interface conforms to. This would give the user interface designer an indication of what appeals to the users in the interface, ensuring that subsequent redesigns of the interface do not destroy or change the appealing features.
- Hold a post-evaluation discussion after the evaluation. While this is only possible when the evaluators can perform the evaluations simultaneously, such a discussion may result in valuable suggestions for improvements to the user interface. In this study, time constraints on the part of the evaluators and the coordinator did not allow for the evaluations to be conducted simultaneously. One way of overcoming this

problem could have been to ask evaluators to suggest ways in which the usability problems could be improved upon — seeing as they are considered experts in the field and should therefore have experience with usable user interface features.

D.6 Conclusion

We are confident the methodology followed in this heuristic evaluation allowed for the identification of the major user interface problems of the stereo system, however we are aware of the areas in which the evaluation could be improved. We conclude that it would be beneficial for the user interface designer and the users of this system to take the results of this evaluation into account, since improvements based on this evaluation would greatly enhance the usability of the stereo system.

Appendix E

Earlier analysis work: Modeling Heuristic Violations Semiotically

E.1 Introduction

This was a semiotic application to the problematic signs in th user interface that was done very early in the year. There is no guarantee that these categories are sound or complete, however they were based on three case studies. Two were from the literature and the third was the heuristic evaluation we conducted. The results from the heuristic evaluations, were isolated as heuristic violations and the signs, i.e., user interface components that were involved in the violations were analysed. These problematic signs that were causing heuristics to be violated were divided up into categories of signs showing related problems. While this is an interesting idea, not much was gained from it because the heuristics that were derived from the analysis were not novel.

This chapter proceeds to introduce the case studies, and then proceeds to group or categorise signs involved in heuristic violations together.

E.2 Case Studies

There were three case studies involved in this categorisation of sign problems: the stereo system, the TRAVELweather GUI and Labscape 0.2. These were chosen because they were all evaluated according to Nielsen's ten usability heuristics. The stereo system evaluation was held by the author and the other two were selected because their results were available and very complete.

E.3 Categories

The advantage of modelling the heuristic violations on semiotics is that we can now ask the question: "Why does this interface violate a particular heuristic?" To answer it, we must isolate the particular signs involved in the violation, and investigate their properties. This is how the different sign categories further discussed below, were discovered. The two basic categories of sign problems identified were:

1. Representamen Problems
2. Matching Problems

E.4 Representamen Problems

This category of problem is concerned with the properties of the representamen. It can also be seen as a category of cosmetic problems and can be further subdivided into signs that are:

1. Too similar
2. Obscured
3. Distracting
4. Too close together

Representamen problems do not impact the perceived meaning of the sign - if the sign is visible the user will generally interpret the sign correctly. However, the way it is represented could cause the user to make errors or waste time identifying the sign. Representamen problems can occur in four ways. The representamens of two different signs may be so similar that the user under pressure may mistake one interpretant for the other. This causes an error because the user unintentionally invokes the functionality of the wrong sign. Alternatively, the representamen can be obscured - the sign does exist but it is not visible to the user. This can be the result of either the way signs like windows or text boxes are layered, or colour contrasts that make them difficult to see. Then there may be signs that are well understood by the user, but are less important than what their appearance make out to be. This subcategory is called distracting signs. They distract attention from more important signs in the interface. The last subcategory is a problem of the signs being too close together. This can also cause the user to make errors. Closely situated signs can be mistakenly selected when the user is working at speed because increased speed sacrifices accuracy.

E.4.1 Too Similar

This category of problems involves signs that look so similar that the user possibly makes the error of interpreting one for the other.

Examples

- Labscape

Once the user has selected a check box in dialog box in figure E.1, the system automatically starts to log in the user. There is no chance for the user to confirm the selection or cancel if the wrong check box has been selected. This sequence of actions by does not help to prevent the user from making errors. IN fact, the errors can not be easily undone. But why has the user made the error in the first place? The window allowing users to check in can be seen in figure E.1. The signs of interest here are seen in figure E.2. In this case, it is obvious that if User2 was not selected, then there is very little distinguishing the two signs. This will make

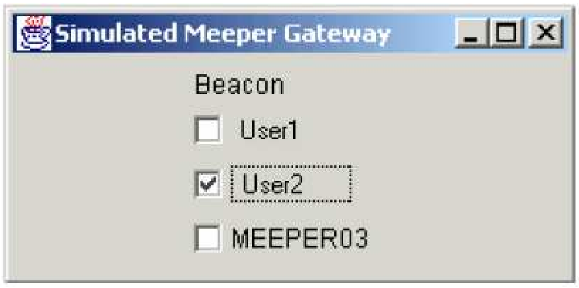


Figure E.1: User check-in window

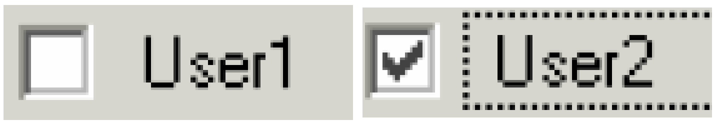


Figure E.2: Two similar representamens

it easy for a user to mistake User1's check box for User2's. Selecting the wrong check box begins the log in for the mistaken user right away.

- TRAVELweather GUI

The same problem occurs in the TravelWeather GUI. Here the user's understanding of the map in figure E.4 is hampered by the representation of the water features, i.e., the ocean and lake. The same representamen (see fig E.3) stands for two different objects. Zoom factors and latitude and longitude specifications can render the map in a way that makes it difficult to tell what is sea and what is lake. Telling the difference is made even more difficult by there being no place names to give clues as to what part of North America the map is displaying.

- Stereo System

There is also an example of this in the stereo interface, figure E.5. The [add all] options appearing next to the album names all have different underlying objects, yet they all look the same (see figure E.6). A better design may perhaps let the user select the album with a check box and then have one add all button to add all the songs of the checked albums to the playlist. Users working at speed may select the wrong [add all] option, and add the

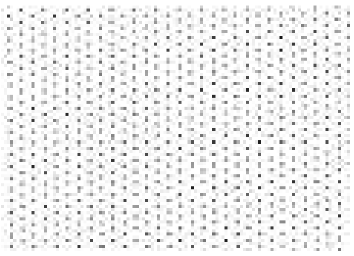
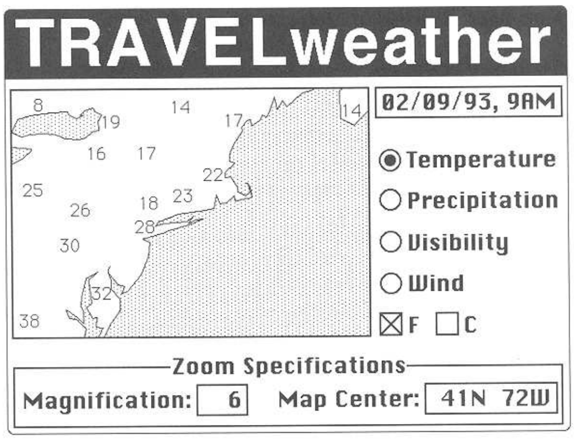


Figure E.3: Representation of Water


TRAVELweather

02/09/93, 9AM

[✓ Temperature] Precipitation Visibility Wind

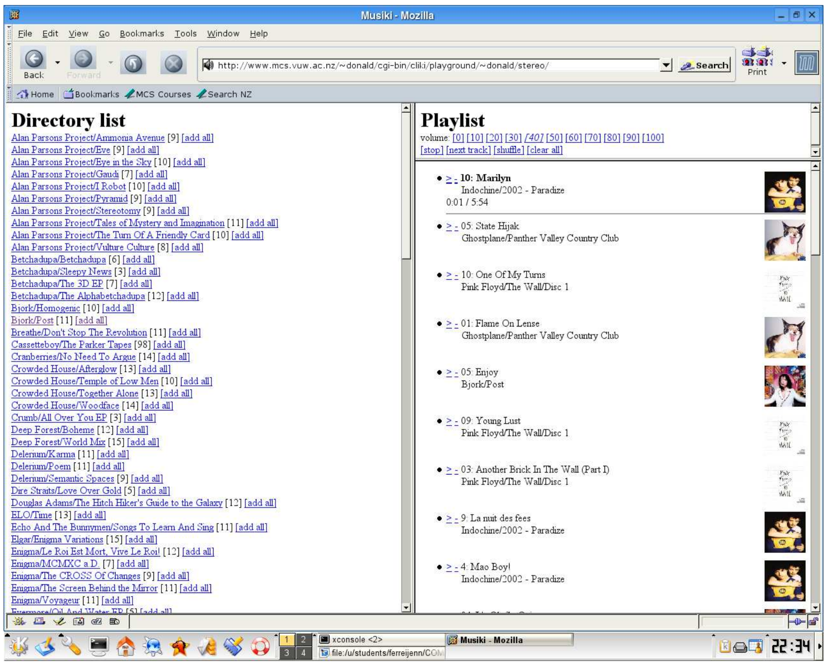
[✓ F] C

Zoom Specifications

Magnification: 6

Map Center: 41N 72W

Figure E.4: TravelWeather GUI



http://www.mcs.vuw.ac.nz/~donald/cgi-bin/cliki/playground/~donald/stereo/

Directory list

Alan Parsons Project/Ammonia Avenue [9] [add all]
Alan Parsons Project/Eve [9] [add all]
Alan Parsons Project/Eye in the Sky [10] [add all]
Alan Parsons Project/Gaudi [7] [add all]
Alan Parsons Project/I Robot [10] [add all]
Alan Parsons Project/Pyramid [9] [add all]
Alan Parsons Project/Stereotomy [9] [add all]
Alan Parsons Project/Tales of Mystery and Imagination [11] [add all]
Alan Parsons Project/The Turn Of A Friendly Card [10] [add all]
Alan Parsons Project/Vulture Culture [8] [add all]
Betchadupa/Betchadupa [6] [add all]
Betchadupa/Sleepy News [3] [add all]
Betchadupa/The 3D EP [7] [add all]
Betchadupa/The Alphabetchadupa [12] [add all]
Bjork/Homogenic [10] [add all]
Bjork/Post [11] [add all]
Breathe/Don't Stop The Revolution [11] [add all]
Cassetteboy/The Parker Tapes [98] [add all]
Cranberries/No Need To Argue [14] [add all]
Crowded House/Afterglow [13] [add all]
Crowded House/Temple of Low Men [10] [add all]
Crowded House/Together Alone [13] [add all]
Crowded House/Woodface [14] [add all]
Crumb/All Over You EP [3] [add all]
Deep Forest/Boheme [12] [add all]
Deep Forest/World Mix [15] [add all]
Delerium/Karma [11] [add all]
Delerium/Poem [11] [add all]
Delerium/Semantic Spaces [9] [add all]
Dire Straits/Love Over Gold [5] [add all]
Douglas Adams/The Hitch Hiker's Guide to the Galaxy [12] [add all]
ELO/Time [13] [add all]
Echo And The Bunnymen/Songs To Learn And Sing [11] [add all]
Elgar/Enigma Variations [15] [add all]
Enigma/Le Roi Est Mort, Vive Le Roi! [12] [add all]
Enigma/MCMXC a.D. [7] [add all]
Enigma/The CROSS Of Changes [9] [add all]
Enigma/The Screen Behind the Mirror [11] [add all]
Enigma/Voyageur [11] [add all]
Evermore/Oil And Water EP [5] [add all]

Playlist

volume: [0] [10] [20] [30] *[40]* [50] [60] [70] [80] [90] [100]
[stop] [next track] [shuffle] [clear all]

- > - **10: Marilyn**
  Indochine/2002 - Paradize
  0:01 / 5:54
- > - 05: State Hijak
  Ghostplane/Panther Valley Country Club
- > - 10: One Of My Turns
  Pink Floyd/The Wall/Disc 1
- > - 01: Flame On Lense
  Ghostplane/Panther Valley Country Club
- > - 05: Enjoy
  Bjork/Post
- > - 09: Young Lust
  Pink Floyd/The Wall/Disc 1
- > - 03: Another Brick In The Wall (Part I)
  Pink Floyd/The Wall/Disc 1
- > - 9: La nuit des fees
  Indochine/2002 - Paradize
- > - 4: Mao Boy!
  Indochine/2002 - Paradize

Figure E.5: UI of the stereo system

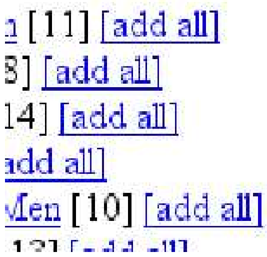
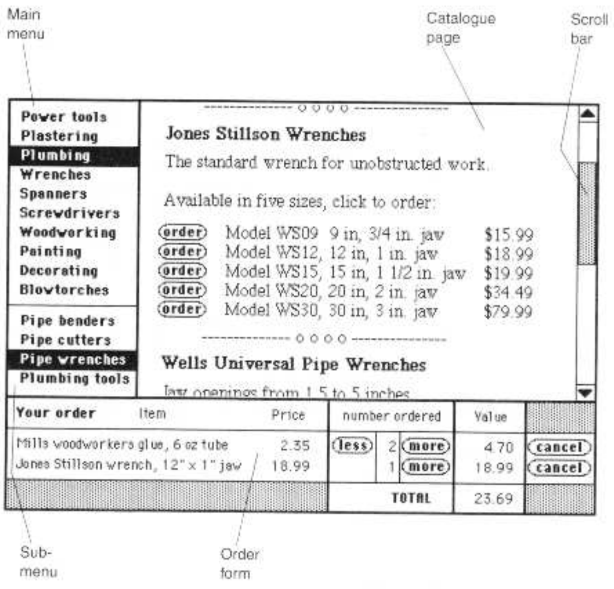

n [11] [add all]
3] [add all]
14] [add all]
add all]
Men [10] [add all]

Figure E.6: Option to add all songs of the corresponding album



Power tools
Plastering
Plumbing
Wrenches
Spanners
Screwdrivers
Woodworking
Painting
Decorating
Blowtorches

Pipe benders
Pipe cutters
Pipe wrenches
Plumbing tools

-------------- o o o o --------------

Jones Stillson Wrenches

The standard wrench for unobstructed work.

Available in five sizes, click to order:

order	Model WS09 9 in, 3/4 in. jaw	$15.99
order	Model WS12, 12 in, 1 in. jaw	$18.99
order	Model WS15, 15 in, 1 1/2 in. jaw	$19.99
order	Model WS20, 20 in, 2 in. jaw	$34.49
order	Model WS30, 30 in, 3 in. jaw	$79.99

-------------- o o o o --------------

Wells Universal Pipe Wrenches

Jaw openings from 1.5 to 5 inches

Your order Item	Price	number ordered			Value	
Mills woodworkers glue, 6 oz tube	2.35	less	2	more	4.70	cancel
Jones Stillson wrench, 12" x 1" jaw	18.99		1	more	18.99	cancel
				TOTAL	23.69	

Figure E.7: Online catalogue

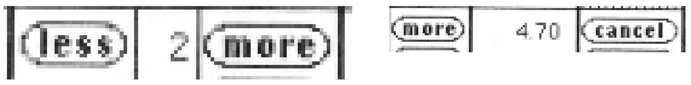


Figure E.8: Accidental cancellation

wrong album to the playlist, not only because the sign is duplicated for every album, but also because of how closely they are situated to one another - this problem is discussed separately under the too close category. So far we have looked at single signs that are too similar. However, this category is not restricted to only single signs. The next example shows that a combination of signs (or complex sign) that are too similar, can also have the same effect. As seen in the interface in figure E.7. The problem is that the user may accidentally cancel an order and there is no way of undoing the last operation. Remembering that the question is: "Why is the user able to make the error in the first place?", when looking at the signs involved in more detail, it becomes obvious. See figure E.8. Here is a common visual pattern which may lead to erroroneous selections. At speed the user may mistake the cancel button for the more button. Motivating the fact that users don't read what's on the screen is in an article I still need to get[43]. Suggestions for improvement would be to substitute the less and more buttons for a sign which looks different from the cancel button. For example see figure E.9. The point is that the pattern should be broken, so not all the buttons need to be replaced.

Proposed Heuristic: Make representamens with different objects look different.

When this is not possible, as in the case with the user check-in box and the [add all] of the stereo, ask the user for confirmation of the choice or combine the repeating sign into one, or do both.

In this section, the possible improvements to the signs are improvements that address a major issue: that of error prevention.

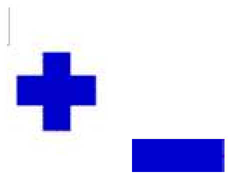


Figure E.9: Suggested improvements

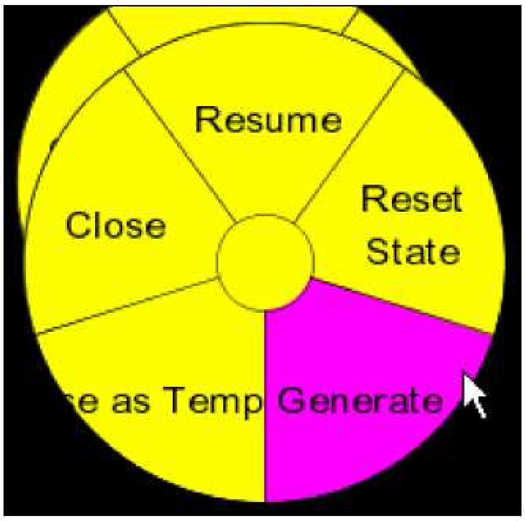


Figure E.10: Pie menu with highlighted magenta

E.4.2 Obscured Signs

Evaluators felt theses signs were not clear enough for the user to easily recognise their existence. With this comes the danger that the user may feel a certain functionality is missing even though it does exist. Obscured signs are signs that exist in the interface, but for various reasons, the user can not see them. There may be several reasons why the user can not see a sign. One reason could be that the colour contrast makes them difficult to see.

Examples

- Labscape

Heuristic evaluators in the Labscape study noted that black text on a magenta background is difficult to read. Andersen[3] states that in a semiotic approach, each part of the system is a sign for somebody, so the default is that *everything should be visible and interpretable while invisibility should be explicitly decided*. Figure E.10 shows the pie menu used in the application and figure E.11 shows the problematic sign.

It is also possible that the layering of signs like windows can cause an important sign to become hidden. In the labscape evaluation, the user check-in window was obscured by the debug window (See figure E.12). If a sign is not displayed completely, this is also an example of an obscured sign. The window in figure E.1 was originally not fully expanded for the whole title to be visible - as in figure E.13.

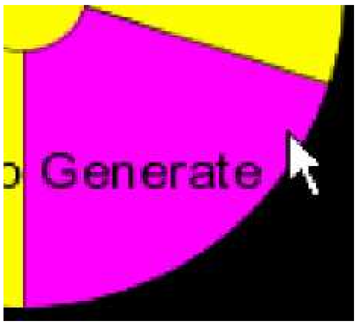


Figure E.11: Highlighted magenta

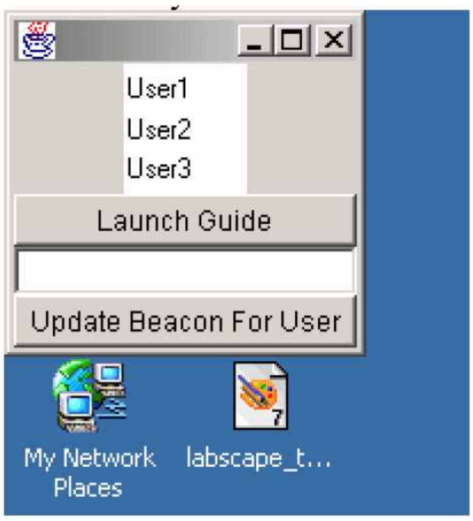


Figure E.12: Debug window covers user check-in window

- Stereo System

If the designer picks a very unusual representation for some functionality, the user may not find the sign for the task they have in mind. This was found in the heuristic evaluation of the stereo system (see figure E.5). Figure E.14 shows the obscure volume control. Several evaluators thought the volume control was difficult to locate.

Proposed Heuristic: Make representamens visible.

E.4.3 Distracting signs

These signs do not influence the user's understanding of the other signs in the interface, but only detract the user's attention or cause minor unease for various reasons. One reason may be that these signs are not as important as their visual representation make them out to be - they might visually dominate the interface for no particular reason. Other distracting signs may appear as superfluous and unnecessary, or take on an unusual form so that the user perhaps focuses too much attention on them. Elizabeth Krupinski's research on gaze fixations could support this point.

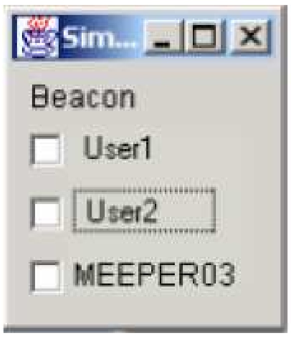


Figure E.13: Window title not visible

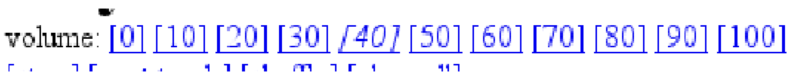

volume: [0] [10] [20] [30] [40] [50] [60] [70] [80] [90] [100]

Figure E.14: Volume control of the stereo system

- TRAVELweather GUI

The TRAVELweather interface in Figure E.4 has a very large title in relation to the whole, which wastes valuable space that could be used for better representing other signs. The unexplained dominance of the title makes this a distracting sign. The evaluators also found that the representation of the number zero, in the date and time textbox is too much of a computer typeface. While this is not an impediment to the overall interpretation of the date or time, if the user notices it, it could cause minor unease.

- Stereo System

The scroll bar on the top right hand window of the stereo interface (figure E.5) seems superfluous. The user may waste time by trying to use the scroll bar, only to find out that the window already displays everything it contains.

Proposed Heuristic: Do not draw unwarranted attention to the representamen and if it is not required do not create it in the first place.

E.4.4 Signs too close together

This problem is characteristic of how groups of signs have been positioned in the interface. When signs are positioned too close together, the user may mistakenly select the wrong one. Use Fitt's Law as applied to UI design to motivate this point.

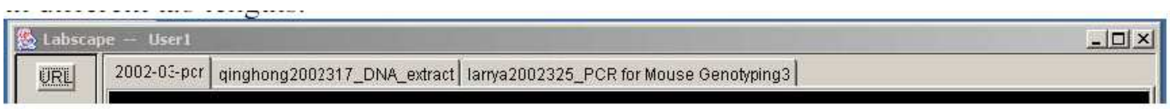
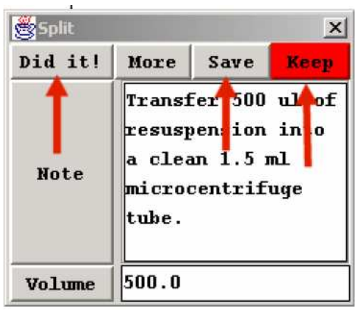

Labscape -- User1
URL
2002-03-pcr
qinghong2002317_DNA_extract
larrya2002325_PCR for Mouse Genotyping3

Figure E.15: Tablength is inconsistent



Transfer 500 ul of resuspension into a clean 1.5 ml microcentrifuge tube.

500.0

Figure E.16: Did It!, Save and Keep are ambiguous

Examples

- Stereo System

An example is the [add all] option (see figure E.6) of the stereo system. As mentioned previously, users working at speed may accidentally select the wrong [add all] option and add the wrong album to the playlist. Since there is no undo option, this is a dangerous state of affairs.

- Labscape

There is a danger in Labscape of losing unsaved work due to the proximity of certain buttons. Figure E.16 shows one of the dialog windows of Labscape. The Save and Keep buttons are ambiguous in meaning, making it harder for the user to quickly determine which functionality is actually required. Further, the Save and Keep buttons are right next to each other, since selecting the Keep button results in the loss of modifications to the data.

Proposed Heuristic: Keep representamens with different objects at a safe distance.

E.5 Matching Problems

This category of problems is concerned with the mismatch between the designer's intent with the representamen, and the user's interpretation of it. This distinguishes the problems in this category from those of the representamen problems in section E.4. Here the meanings are actually influenced by the representamen and can be further divided into:

1. Interpretant-Object mismatch
   - lying signs
   - convention
2. required semiosis

E.5.1 Interpretant-Object Mismatch

This is quite a broad category and it includes all problems where there is a mismatch between the designer's object and the user's interpretant. For example, a user may interpret a sign as representing some functionality, when the designer has in fact attributed a different functionality to the sign. There are two special cases of problems of this type. One case is when the designer has violated some convention. The user, through convention will create their interpretant of the sign, but unfortunately it will not match the designer's object because the designer has not adhered to the convention. Another case is when the designer misrepresents some underlying reality. The user may be confused when reality is misrepresented in the interface. These will be discussed in more detail in sections E.5.2 and E.5.3. However, not all Object-Interpretant problems have conventions or underlying realities. Some signs are simply ambiguous due to factors such as inconsistent use of terminology or colours by the designer, important information about what the sign represents is missing, or the user simply does not understand what the sign is supposed to mean.

Examples

- Stereo System

The functionality to add an album to the PlayList is represented by the [add all] link in the Directory list. This is inconsistent with the functionality to add an individual song to the PlayList. To add an individual song, the user must select [enqueue]. Although the underlying functionality is the same, i.e., the song or songs are added to the PlayList, the different terminology makes this less obvious to the user. This is an interpretant-object mismatch, since the user is left guessing at the correct object of the [add all] and [enqueue] features.

- TRAVELweather GUI

The date and time text box in the TRAVELweather GUI can be edited by the user. But this is not obvious from merely looking at it. It requires extra information for the user to know right from the first encounter that the information in the date and time text box can be edited. This can be done either in the form of a drop down list, which would change the appearance slightly, or by inviting the user to enter a new date and time with some form of textual message in or near the text box.

- Labscape

In the Labscape dialog window in figure E.16 the buttons called Keep, Save and Did it! are ambiguous. They may be very close in meaning, but the designer intended slightly different actions to be taken when selecting them. According to the report[17],

> when Keep is pressed, another dialog can be opened without the current window automatically closing, but any data that has not been saved is lost. When Save is pressed, the information in the dialog window is saved to the database and the window closes. When Did it! is pressed, the step is marked as completed, but any data that has not been saved is lost.

Again, the user is left guessing at the correct objects of the buttons. They could be left wondering whether the Keep button also saves the changes to the data, or whether the Did it! button will save the data and then close the window. It is not clear what the designer intended.

E.5.2 Convention

In this category, signs have a certain convention and users expect them to behave accordingly. However, as seen in the case studies, designers sometimes do not adhere to the existing conventions.

Examples

- Stereo System

The appearance of the signs used for playing and stopping a song have not been implemented in the conventional manner. The > sign plays a song and [stop] stops the song playing. There are well-known conventions that every user is aware of for these functions. Although the [stop] sign does not necessarily confuse the user or cause a misinterpretation of the underlying functionality, it is better to implement the convention simply because it exists and because that way the user feels more comfortable with using the system. Perhaps it is a different story with the > sign. Users may not understand the functionality of the sign at a first encounter, whereas the conventional sign for the play functionality would be instantly recognisable.

- TRAVELweather GUI

It is convention to represent mutually exclusive concepts as radio buttons. Check boxes, as used in the TRAVELweather GUI, suggest that both Celcius and Fahrenheit can be selected at the same time. However, these check boxes behave the way radio buttons do, i.e., only one can be selected at a time. This contradicts the conventional behaviour of check boxes and therefore the designer's object of this sign is likely to contradict the interpretant of the user.

- Labscape

In some cases the user's previous experience has an impact on what can be regarded as convention. The application with which the Labscape users are familiar with is MS Excel. In a sense, the behaviour in Excel becomes the expected convention of the users of Labscape. The Labscape system does not notify the user when a duplicate session name has been entered. Instead, it allows the user to continue working as though it is a new session, thereby confusing data and effectively ruining the experiments. Because opening a session is similar to opening a file, the user would expect to be notified if there already exists such a session/file name. This is what would happen in Excel, so this is what the user expects from Labscape.

Proposed Heuristic: Adhere to convention if it exists.

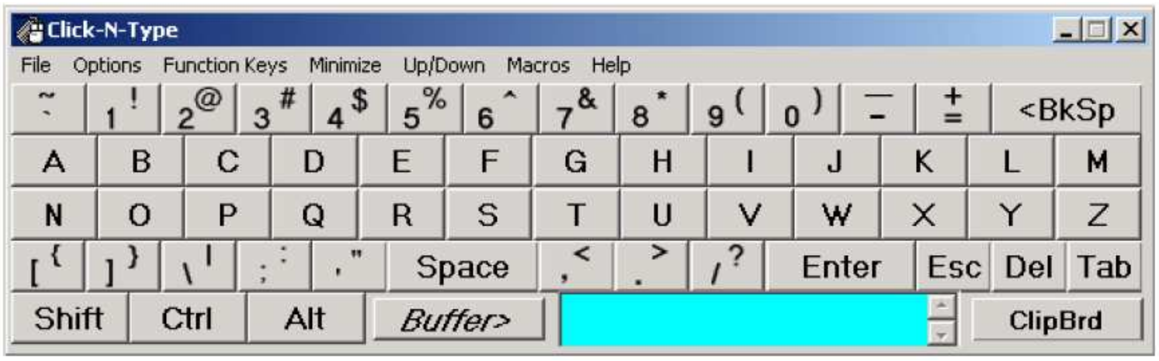

Click-N-Type
File Options Function Keys Minimize Up/Down Macros Help
~ ` 1 ! 2 @ 3 # 4 $ 5 % 6 ^ 7 & 8 * 9 (0) — - + = <BkSp
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z
[{] } \ | ; : ' " Space , < . > / ? Enter Esc Del Tab
Shift Ctrl Alt Buffer> ClipBrd

Figure E.17: Pop-up keyboard provided in Labscape

E.5.3 Lying Signs

There is sometimes a reality which the sign is a representation of. This reality is usually known to the user so that the way the designer represents this will determine whether the user understands the sign or not. For example, the underlying reality of a map is the actual geography which it is supposed to represent. If the map does not agree with the geography, the map is then a misrepresentation of the reality and has minimal value to the user. The same can be said of signs on the computer screen.

Examples

- Stereo System

The display of the time elapsed is not shown in real time. This becomes a lying sign because it does not accurately represent the underlying reality, i.e., the actual time that has passed.

- TRAVELweather GUI

On the map of the TRAVELweather GUI, Rhode Island is not shown to be an island. It appears as connected with the mainland and so the map is lying about the fact that it is an island. In this case, this lying sign makes the map more difficult to read, given that only a small portion of North America is shown and no place names are provided.

- Labscape

The pop-up keyboard in the Labscape system is laid out alphabetically (See figure E.17). Users in English speaking countries will most likely use the QWERTY keyboard (See figure E.18. Therefore it would be a better match for the pop-up keyboard to also follow this layout. Perhaps the alphabetical layout should instead be viewed as a break with convention, however considering the fact that actual QWERTY keyboards do exist, there is necessarily a reality on which the sign of the keyboard on the screen could be modeled.

Proposed Heuristic: Adhere to reality if it exists.

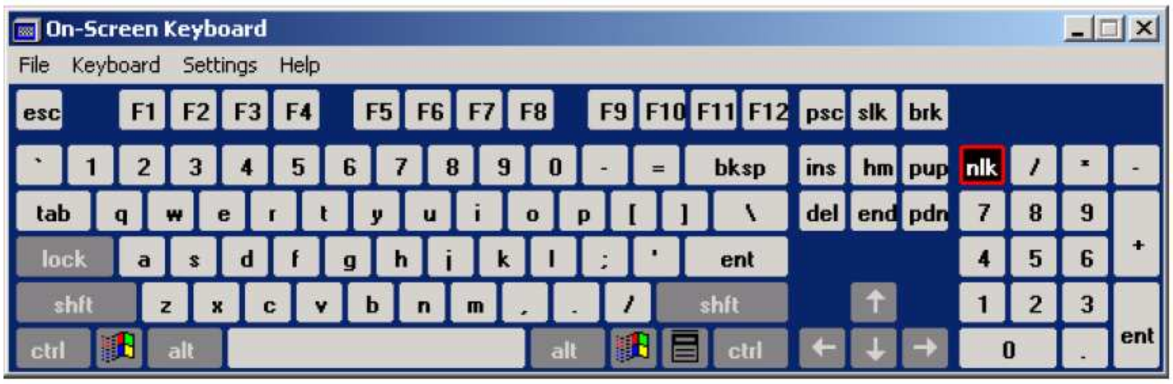

On-Screen Keyboard
File Keyboard Settings Help
esc F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 psc slk brk
` 1 2 3 4 5 6 7 8 9 0 - = bksp ins hm pup nlk / * -
tab q w e r t y u i o p [ ] \ del end pdn 7 8 9 +
lock a s d f g h j k l ; ' ent 4 5 6
shft z x c v b n m , . / shft 1 2 3
ctrl alt alt ctrl 0 . ent

Figure E.18: MS Windows on-screen keyboard

E.5.4 Required Semiosis

Sometimes the interface requires the user to go through a series of steps in order to access the information or the functionality required and when there is no support for remembering what steps have been taken the result is a required semiosis problem. This could happen when the user is required to select options from a cascading menu, for example. Only once all the required selections have been made, can the user access the needed information or functionality. Alternatively, when certain settings are required by the user, and the system can not store the user's settings, then the user will be required to repeat the same steps every time the system is used. This indirectness results in a waste of time and is also taxing on the user's memory. See `http://developer.kde.org/documentation/design/ui/simplify.html` for a discussion.

Examples

- TRAVELweather GUI

Regular users who check the same area's weather, could benefit by being provided with the option of directly accessing that area's weather information. This will save the user time because the various selections do not have to be repeated every time the system is used. The user will then also not have to remember the exact steps involved in attaining the specific area's weather information.

- Labscape

Users are forced to make selections from the pie menus in the system. There are no shortcuts for advanced users, or for actions that need to be performed often. Regular users will soon find it tedious to use the pie menus when a few keystrokes could speed up their tasks.

Proposed Heuristic: Allow the user the most direct access route to information as possible.

E.6 Summary

The following table summarises the categories discussed above:

$RepresentamenProblems$	$MatchingProblems$
1. Similarity	1. Interpretant-Object mismatch
2. Obscured signs	- lying signs
3. Distracting signs	- convention
4. Physical limits	- user-Designer mismatch
	2. continuous semiosis

Bibliography

[1] ANDERSEN, P. Computer Semiotics. *Scandinavian Journal of Information systems 4* (1992), 3–30.

[2] ANDERSEN, P., HASLE, P., AND BRANDT, P. *Machine Semiosis*. de Gruyter, forthcoming, ch. To Appear in Handbook of Semiotics.

[3] ANDERSEN, P., HOLMQVIST, B., AND JENSEN, J., Eds. *The computer as medium*. Cambridge University Press, 1993.

[4] ANDERSEN, P., AND NOWACK, P. Tangible Objects: Connecting Informational and Physical Space. Tech. rep., Department of Computer Science, Aalborg University and Maersk Institute, University of Southern Denmark, 2003?

[5] ANDERSEN, P. B. Elastic Systems. In *Human-Computer Interaction, Interact '01. IFIP TC.13 International Conference on Human-Computer Interaction* (2001), M. Hirose, Ed., Ansterdam. IOS Press, pp. 367–374.

[6] ANDERSON, D. J. In www.uidesign.net/2000/interviews/larry1.html, 2000.

[7] ANDERSON, M., AND MERRELL, F., Eds. *On Semiotic Modelling*. Mouton de Gruyter, 1991.

[8] BARR, P. User-Interface Metaphors in Theory and Practice. Master's thesis, Victoria University of Wellington, 2003.

[9] BARR, P. BIDDLE, R., AND NOBLE, J. Icons R Icons: User interface icons, metaphor and metonymy. Tech. Rep. CS-TR-02/20, Victoria Unversity of Wellington, September 2002.

[10] BIDDLE, R., CONSTANTINE, L. L., AND NOBLE, J. Usage-Centered Design and Software Engineering: Models for Integration. In *IFIP Working Group 2.7/13.4*, ICSE 2003 Workshop on Bridging the Gap Between Software Engineering and Human-Computer Interaction. Portland Oregon,. 2003.

[11] BLONSKY, M. Introduction: The Agony of Semiotics. In *On Signs*, M. Blonsky, Ed. Baltimore, Maryland. John Hopkins University Press, 1985.

[12] BODKER, S. *Through the Interface a Human Activity Approach to User Interface Design. DAIMI PB-224*. PhD thesis, Aarhus Universitet, 1987.

[13] BRUEGGE, B., AND DUTOIT, A. H. *Object-Oriented Software Engineering: Conquering Complex and Changing Systems*. New Jersey, Prentice Hall, 2000.

[14] CHANDLER, D. *Semiotics: The Basics*. Routledge, 2001.

[15] CHONG, S., LIAO, S. Y., AND K., L. Semiotics for Information Systems Engineering: re-use of high-level artefacts. In URL www.scit.wlv.ac.uk/~in8189/CSNDSP2002/Papers/A1/A1.1.pdf, 2002.

[16] COLLINS, B. L., AND LERNER, N. D. Assessment of fire safety symbols. *Human Factors 24*, 1 (1982), 75–84.

[17] CONSOLVO, S., AND TOWLE, J. Heuristic Evaluation of Labscape Version 0.2. Tech. Rep. IRS-TR-02-014, Intel Research Seattle and University of Washington Information School, August 2002. In URL "www.seattleweb.intel-research.net/projects/labscape/heuristic_eval.htm%l". Accessed 28 July 2004.

[18] CONSTANTINE, L., WINDL, H., NOBLE, J., AND LOCKWOOD, L. From Abstraction to Realization: Abstract Prototypes Based on Canonical Components—REVISED. In URL http://www.foruse.com/articles/canonical.htm, July 2003.

[19] CONSTANTINE, L. L. Abstract prototyping. In *Talk Given on 23 March 2004 at Victoria University of Wellington*.

[20] CONSTANTINE, L. L. Canonical Abstract Prototypes for Abstract Visual and Interaction Design. In *Proceedings of DSV -IS'2003- 10th International Workshop on Design, Specification and Verification of Inter-active Systems, Lecture Notes in Computer Science* (2003), J. Jorge, N. Nunes, and J. e Cunha, Eds., Berlin, Springer-Verlag.

[21] CONSTANTINE, L. L., AND LOCKWOOD, L. A. D. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design*. ACM Press, 1999.

[22] COOPER, A. *About Face: The Essentials of Interaction Design*. John Wiley and Sons, 1995.

[23] DE SOUZA, C., BARBOSA, S., AND PRATES, R. A Semiotic Engineering Approach to HCI. In *Proceedings of CHI'01 extended abstracts on Human factors in computing systems* (Seattle, Washington, 2001), ACM Press, pp. 55–56.

[24] DE SOUZA, C. S. The semiotic engineering of concreteness and abstractness: from user interface languages to end user programming languages. Tech. Rep. MCC08/96, PUC-Rio, Rio da Janeiro, 1996.

[25] DROSCHL, G., AND KARNER, H. Usage-Centered Interface Design for Knowledge Management Software. *Journal of Universal Computer Science 8*, 6 (June 2002). In URL http://www.jucs.org/jucs_8_6/usage_centered_interface_design. Accessed 21 October 2004.

[26] ECO, U. *Semiotics and the Philosophy of Language*. Southampton, UK. Camelot Press Ltd., 1984.

[27] GOGUEN, J. *On Notation*. Prentice-Hall, 1993, ch. TOOLS 10: Technologyof Object-Oriented Languages and Systems.

[28] GOGUEN, J. Semiotic Morphisms. Written for CSE 271: User Interface Design: Social and Technical Issues. In URL http://www.cs.ucsd.edu/users/goguen/papers/sm/smm.html, 1996. Revised 2004. Accessed 19 October 2004.

[29] GRODEN, M., AND KREISWIRTH, M., Eds. *The John Hopkins Guide to Literary Theory and Criticism*. The John Hopkins University Press, 1997. In URL http://www.press.jhu.edu/books/hopkins_guide_to_literary_theory/semioti%cs.html. Accessed 19 October 2004.

[30] HOFFMAN, P. *Microsoft Word for the Macintosh: Made Easy, Version 3*, 2nd ed. Berkeley, California. Osborne McGraw-Hill, 1987.

[31] IBM Corporation. RealThings design guide. In URL `http://www-3.ibm.com/ibm/easy/eou_ext.nsf/publish/581`. Accessed 9 Sep 2004.

[32] LINDEKENS, R. *Eléments pour une sémiotique de la photographie*. Paris and Bruxelles, Didier/Aimav, 1971.

[33] MORRIS, C. *Foundations of a Theory of Signs*. Chicago. University of Chicago Press, 1938.

[34] MORTON, J. L. Colour Matters ®—Design Art. In URL `http://www.colormatters.com/colortheory.html`,1995–2002. Accessed 22 October 2004.

[35] NADIN, M. Interface design: A semiotic paradigm. *Semiotica 69* (1988), 269–302.

[36] NADIN, M. *Semiotics in the Individual Sciences*, vol. 2. Bochum: Brockmeyer, 1990, ch. Design und Semiotics, pp. 418–436.

[37] NASS, C., AND REEVES, B. *The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places*. Cambridge University Press, 1996.

[38] NIELSEN, J., AND MACK, R. L., Eds. *Usability Inspection Methods*. New York, John Wiley and Sons, 1994.

[39] NIELSEN, J., AND SANO, D. SunWeb: User Interface Design for Sun Microsystem's Internal Web. In *Proc. 2nd World Wide Web Conf. '94: Mosaic and the Web. Chicago, IL*, pp. 547–557. Also available in URL `http://archive.ncsa.uiuc.edu/SDG/IT94/Proceedings/HCI/nielsen/sunweb.ht%m`.

[40] ORLIAGUET, J. M. Prolegomenon to a Semiotic of Digital Media. In URL `www.ckk.chalmers.se/people/jmo/semiotics/semiotic_of_digital_media.pdf`, 2002. Accessed 5 Sep 2004.

[41] PATTON, J. Extreme Design: Usage-Centered Design in XP and Agile Development. In *forUSE2002: Proceedings of the First International Conference on Usage-Centered, Task-Centered, and Performance Centered Design* (2002), L. Constantine, Ed., Rowley, MA. Ampersand Press.

[42] PEIRCE, C. *Collected Papers of Charles Sanders Peirce*, vol. II of *Elements of Logic*. Harvard University Press, 1932.

[43] ROBERTSON, G., MCCRACKEN, D., AND NEWELL, A. The zog approach to man-machine communication. *Int. J. Hum.-Comput. Stud. 51*, 2 (1999), 279–306.

[44] SALASOO, A. Towards usable icon sets: A case study from telecommunications engineering. In *Proceedings of the Human Fatcors Society 34th Annual Meeting* (Santa Monica, CA, 1990), pp. 203–207.

[45] SCANLON, J., AND PERCIVAL, L. UCD for Different Project Types, Part 1: Overview of Core Design Activities. In URL `http://www-106.ibm.com/developerworks/usability/library/us-ucd/`,March 2002. Accessed 21 October 2004.

[46] SCHAFFER, E., AND SORFLATEN, J. Icons: Much Ado about Something. *The X Journal* (January/February 1996). Published by SIGS Publications, Inc.

[47] SCOLLON, R., AND WONG SCOLLON, S. *Discourses in Place: Language in the Material World*. Routledge, 2003.

[48] TOMASELLI, K. Semiotics, Semiology and Film. *Communicare 1*, 2 (1981), 42–61.

[49] UNDERWOOD, M. J. Introductory models and basic concepts: semiotics. In URL `http://www.cultsock.ndirect.co.uk/MUHome/cshtml/semiomean/semio1.html`. Accessed 19 October 2004.

[50] WIMMER, F. The New User Interface of V9 Graphing Tools: A Usability Case Study. `http://www2.sas.com/proceedings/sugi26/p177-26.pdf`. Accessed 25 Aug 2004.

[51] WINDL, H. Designing a Winner: Creating STEP 7 Lite with Usage-Centered Design. In *forUSE2002: Proceedings of the First International Conference on Usage-Centered, Task-Centered, and Performance Centered Design* (2002), L. Constantine, Ed., Rowley, MA. Ampersand Press.