Chapter #

A SEMIOTIC MODEL OF USER-INTERFACE METAPHOR

Pippin Barr, Robert Biddle, and James Noble
Department of Mathematical and Computing Sciences, Victoria University of Wellington, New Zealand

Abstract: User-interface metaphor is an extremely popular technique for creating usable user-interfaces. Although there has been considerable research into its application in practical settings, there has been little investigation into the concept itself. This paper applies Peircean semiotics to user-interface metaphor in order to provide a structured model of the concept. As an end result, a far more detailed understanding of the technique and a useful vocabulary are made available to user-interface designers and researchers.

Key words: User-interface, metaphor

1. INTRODUCTION

Metaphors are, and have been, a very popular approach to user-interface design. Two of the largest interface design companies in the world, Apple and Microsoft, both strongly recommend the use of metaphor:

> You can take advantage of people's knowledge of the world around them by using metaphors to convey concepts and features of your application. Use metaphors involving concrete, familiar ideas and make the metaphors plain, so that users have a set of expectations to apply to computer environments. (Apple, 1992)

> Familiar metaphors provide a direct and intuitive interface to user tasks. By allowing users to transfer their knowledge and experience, metaphors

> make it easier to predict and learn the behaviors of software-based representations. (Microsoft, 1995)

Although there has been considerable research positing the advantages of metaphor in the user-interface (Carroll et al., 1988; Dertouzos, 2001; Erickson, 1990; Johnson, 1997), its problems (Carroll and Mack, 1995; Carroll et al., 1988; Carroll and Thomas, 1982; Halaz and Moran, 1982; Kay, 1990; Mountford, 1990; Nelson, 1990; Norman, 1998; Smyth et al., 1995; Wozny, 1989), and practical advice on its use (Carroll et al., 1988; Carroll and Thomas, 1982; Cataci et al., 1995; Erickson, 1990; Halasz and Moran, 1982; Lundell and Anderson, 1995; Madsen, 1994; Smyth et al., 1995; Väänänen and Schmidt, 1994), there has been little analysis of the concept itself. This lack of investigation into the underlying idea means that discussion is somewhat unguided and relies heavily on intuition and empirical studies. We believe that it is necessary to provide a structured examination of the *concept* of user-interface metaphor in order to fully understand its use.

As noted by Ronald Stamper, there is an important need to "build a science of information systems using the operationally secure, primitive notion of a *sign*." (Liu et al., 2002, p.xiv) Computer user-interfaces are a critical part of the study of information systems because they are the primary means of their representation. This is well reflected in the brief literature review in section 3. Within user-interfaces, the concept of a user-interface metaphor is very common. To this end, it is clear that a semiotic analysis of user-interface metaphor is an important contribution to organisational semiotics. Therefore, in this paper we use Peircean semiotics to explain the structure and function of user-interface metaphors. This involves the creation of a detailed semiotic model of user-interface metaphor, based on the Peircean triad.

In section 2 we will provide a basic introduction to the common perception of user-interface metaphors. Next, in section 3 we discuss the already existing application of semiotics to computers, and specifically to the user-interface. This completed, we begin to establish the background required for the semiotic model by outlining important aspects of Peircean semiotics, along with some research by Umberto Eco, in section 4. Finally, with the background in place, we produce models of both metaphor in general (section 5) and user-interface elements (section 6), before presenting our model of user-interface metaphor in section 7. Some thought is given to necessary future work in section 8 before we conclude in section 9.

2. WHAT IS USER-INTERFACE METAPHOR?

A metaphor is a device for explaining some concept or thing, x, by asserting its similarity to another concept or thing, y, in the form X IS Y.[1] The concept being explained is often referred to as the tenor of the metaphor, while the concept doing the explaining is called the vehicle. Thus, in a metaphor such as JULIET IS THE SUN (Shakespeare, 1933, Act II, Scene II, l.2), the tenor is “Juliet” and the vehicle is “the sun”. Juliet is described to the audience via their knowledge of the sun. Thus, we understand that Romeo believes her to be warm, radiant, high above him, and so forth.

User-interface metaphor is intuitively the application of this device to the user-interface. Thus, a user-interface metaphor is a device for explaining some system functionality or structure (the tenor) by asserting its similarity to another concept or thing already familiar to the user (the vehicle). The key here is that the chosen vehicle is something already *familiar* to the user and so the intention is to provide a base level of comfort and knowledge without necessarily understanding the underlying system.

Consider the traditional example of the “trashcan” used in most desktop environments today. In this case the tenor is the concept of “file deletion” while the vehicle is the concept of “using a trashcan.” The metaphor can thus be written out as FILE DELETION IS USING A TRASH CAN. It is intended to help the user understand how to interact with the system concept of file deletion which, without the metaphor, is an extremely complex process. Thus, the user interacts using concepts such as putting items into a trashcan and emptying the trashcan, rather than requiring detailed knowledge of directory tables, file flags, and so forth.

Although the intuitive idea of user-interface metaphor is simple and easy to understand, it is also clear that it is not sufficiently structured for detailed discussion of the concept. Particularly, it does not offer a strict way to characterise how the metaphor functions or what the metaphor really *means*. The use of semiotics in this paper will help to resolve these issues.

3. COMPUTER SEMIOTICS

Semiotics has been used extensively in the analysis of computer systems, and particularly concerning the user-interface. The treatment of the user-interface as a sign-system, then, is not out of the ordinary.

Joseph Goguen has performed detailed work on the concept of an “algebraic semiotics” (Goguen, 1999; Malcolm and Goguen, 1998). The theory behind this approach is that semiotics can be formally captured and thus become similar to mathematics in its application. Algebraic semiotics has been applied to the user-interface in considerable detail.

A second major researcher in the area of computer semiotics is Peter Bøgh Andersen. In his major work, *A Theory of Computer Semiotics*, Andersen shows in detail how semiotics can be applied to the understanding of computers as sign-systems (Andersen, 1997). In other papers Andersen investigates semiotics as regards programming (Andersen, 1993) and human-computer interaction (Andersen, 2000).

Other approaches to computer semiotics analyse their use for graphic design (Mullet and Sano, 1995), modelling user-interface elements (Nadin, 1988), informing the design process (Andersen and May, 2001; Connolly and Phillips, 2002; de Souza, 1993; May and Andersen, 2001), and analysing sign types in the interface (Barr et al., 2003).

It is clear from this brief survey that semiotics is an accepted approach to explaining computer systems. With this in mind, we now go on to show how Peircean semiotics can be used to analyse and explain user-interface metaphors and their purpose.

4. PEIRCEAN SEMIOTICS

In this section we will first outline the basics of the Peircean triad. Following this we will provide some discussion of the distinction between interpreting and generating signs, as well as the concept of unlimited semiosis. These three ideas from semiotics will be combined to create our model of user-interface metaphor.

4.1 The Peircean Triad

Charles Peirce proposed a triadic model of the sign (Peirce, 1934-1948). In his view, a sign is divided into these three parts: *object*, *representamen* and *interpretant*. Consider, for example, the triad applied to a stop-sign shown in figure 1.

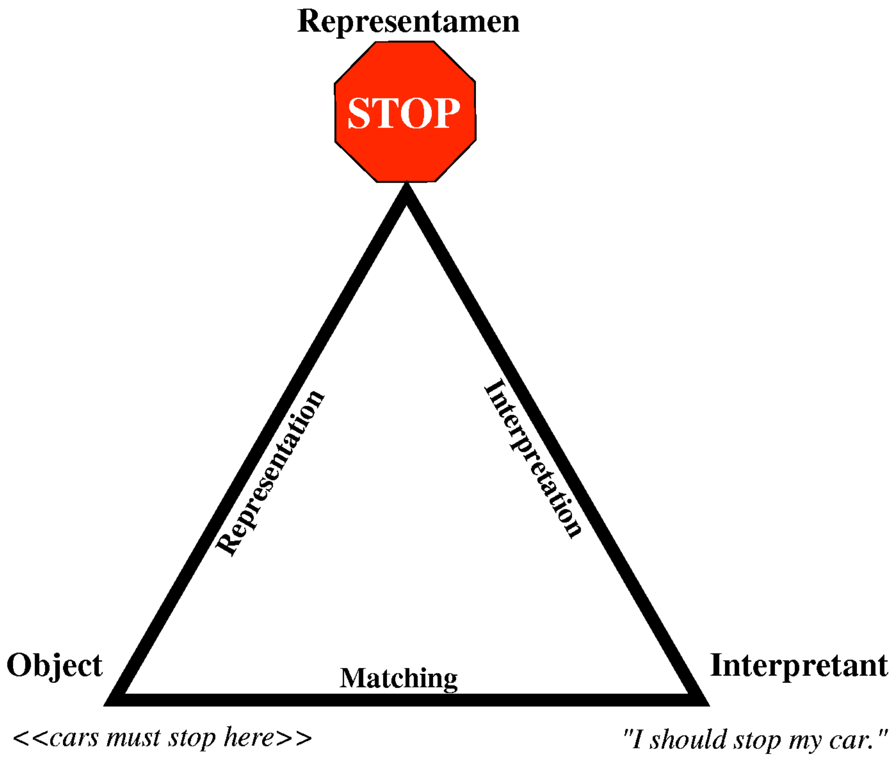


Figure #-1. A diagram of the Peircean triad as applied to a stop-sign.

In this example the *object* is the concept that “cars must stop here”, that is, it is the meaning of the sign. The *representamen* is the stop-sign itself, the sign's manifestation in the world. Finally, the *interpretant* or response to the sign is the successful one of the interpreter understanding they must stop their car.

In addition to the three parts of the sign, three relationships between the parts can be identified, as shown in the diagram. We suggest that these relations are revealing, despite Peirce's claim that semiosis is “not ... in any way resolvable into actions between pairs [of sign-components].” (Peirce, 1934-1948, v.5 p.488)

The relation of *representation* concerns the way in which a red sign with white lettering can represent the concept of stopping a car. In this case, it relates to linguistic and colour conventions, among other things. The *interpretation* relation describes the mental process that takes place between the interpreter seeing the representamen, and arriving at an interpretant. Finally, the *matching* relation concerns how successfully the interpretant arrived at matches the actual object of the sign.

Note that here, and for the rest of this paper, it is possible to talk of a "successful" sign because all signs in this paper are intentional ones. An intentional sign has a *specific* object which it is intended to convey. Interpretants which match this object can be considered successful or correct, and those that do not are unsuccessful in their intent.

4.2 Interpretation and Generation

The Peircean triad can be viewed as explaining either an *interpretive* process or a *generative* process. The interpretive process is the traditional view of the Peircean triad: the interpreter encounters the representamen and develops an interpretant which hopefully matches well with the unknown object.

From a generative perspective, the triad can be viewed as showing how a designer of a sign might view the intended process. Here, the object is known to the designer, who creates a representamen which is meant to lead to a successful interpretant, suitably matching the object.

Because this paper is intended for understanding the design of user-interface metaphors, a generative perspective will largely be taken. That said, the interpretive process is certainly useful for describing user reactions to interfaces.

Despite the clear importance of discussing designed signs, there is surprisingly little literature on the subject. Umberto Eco, however, provides useful insight into the process. In the remainder of this section we present some of Eco's ideas on intentionality which strengthen the case for a generative perspective on semiosis, as well provide useful concepts and terminology for use in the later model.

The key concept we can take from Eco's work is that of the "Model Reader," originally developed in his book *The Role of the Reader* (Eco, 1979). The Model Reader is the reader envisaged by the author of a text or sign who will understand precisely what it is that the author is trying to convey: the object of the text or sign.

The Model Reader helps us to bridge the divide between the generative and interpretive approaches to a sign:

> To make his text communicative, the author has to assume that the ensemble of codes he relies upon is the same as that shared by his possible reader. The author has thus to foresee a model of the possible reader (hereafter Model Reader) supposedly able to deal *interpretatively* with the expressions in the same way as the author deals *generatively* with them. (Eco, 1979, p.7)[2]

We use Eco's terminology to reinforce our explanation of the generative process. First of all, the author of the sign envisages some Model Reader, an interpreter of the sign who has certain beliefs and a certain context such that they will *correctly* interpret the sign. Next, the author of the sign seeks to *produce* the Model Reader "through the use of given ... strategies." (Eco, 1990, p.128) Thus, it is possible to *create* a Model Reader from an actual reader through various forms of influence. It is also clear that this influence must take place through the representamen, which is the point through which author/designer and reader/user are linked.

Based on this discussion, it is clearly justifiable to present a sign model that is from the point of view of a sign *designer*. In this case, the interpretant of the sign may be seen as the interpretant of the Model Reader, rather than a particular reader. Because of this, the interpretant in such a model will be a representation of the *correct* interpretant of the sign, and will thus match the object of the sign via the matching relation.

Eco's comment on the use of strategies to *produce* a Model Reader ties directly to the concept of "narrative strategies" in semiotics. The study of narrative strategies involves considering how a text or sign can produce desired effects in a reader. Clearly, this concept could be highly relevant to the design of user-interfaces, as there is a similar desire to produce particular reactions. For more on narrative strategies in semiotics see Algirdas Greimas (Greimas, 1983) or Gerald Prince (Prince, 1982), for example.

The view of an intentional sign taken in this paper is as follows. The author desires to convey a particular *object* to an audience of readers or interpreters. In order to do this, a *representamen* is created which is intended to produce a model *interpretant* which matches the object. In this way, a sign model can portray the intended effect of the sign, rather than one of the many possible effects. Note that, in order to actually produce the model interpretant, considerable thought as to who the audience of the sign is needed. This links well with the traditional human-computer interaction principle of "know the user" (Nielsen, 1993, pp.73-78).

4.3 Unlimited Semiosis

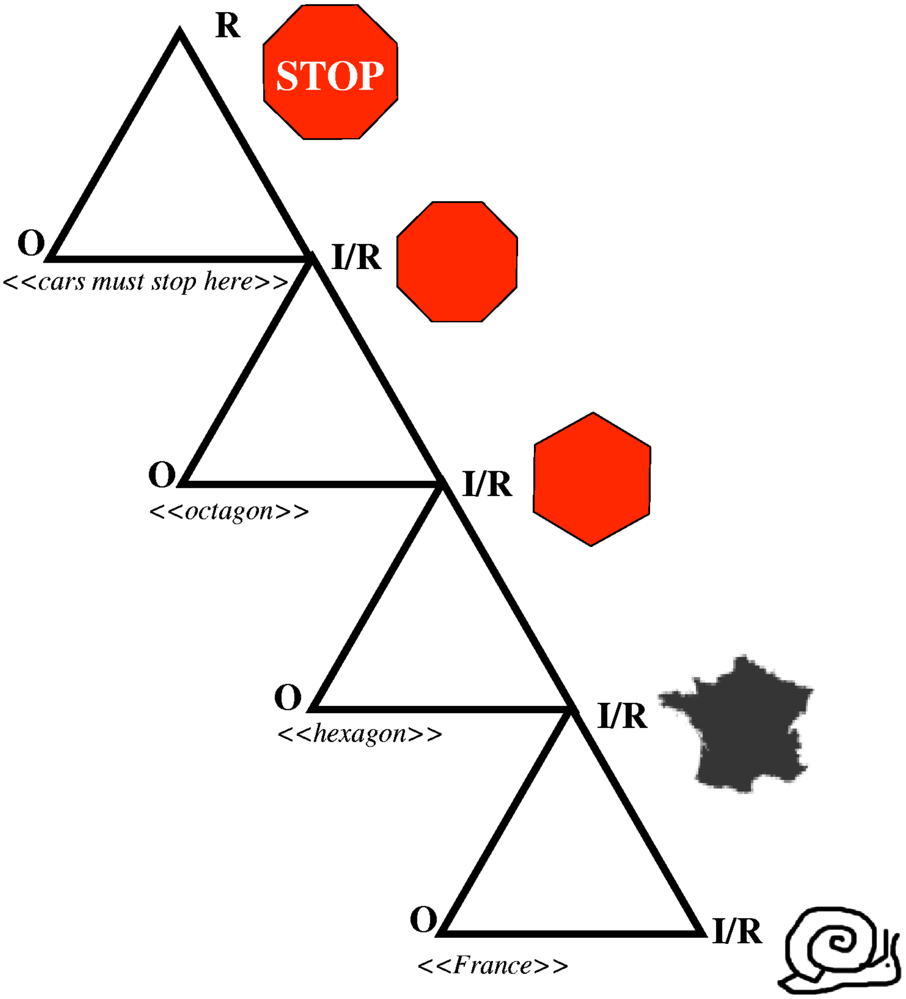


Figure #-2. A diagram of the process of unlimited semiosis.

Before discussing the process of modelling a user-interface metaphor semiotically, it is important to examine the process of semiosis in a little more detail. In particular, it was suggested by Peirce, and furthered by Umberto Eco, that an encounter with a sign is not quite as neat as a single triad. In fact, in Peircean semiotics it is commonly thought that an encounter with a sign involves many interpretants in a process known as *unlimited semiosis*. The term itself is due to Umberto Eco (Eco, 1990), but the idea is also apparent in Peirce's work. For example, Peirce describes a sign as "anything which determines something else (its interpretant) to refer to an object to which itself refers (its object) in the same way, this interpretant becoming in turn a sign, and so on ad infinitum." (Peirce, 1934-1948, v.2 p.300)

What this means in a basic sense is that the interpretant of a sign can be a sign itself. In particular, the interpretant of a sign can become the representamen of a new sign, with a new object and interpretant of its own. This process can go on forever. Hence Eco's terming it *unlimited* semiosis.

Figure 2 gives a possible example of unlimited semiosis. In this case, the initial sign is the sighting of a standard stop-sign. This makes the interpreter think of an octagon shape, which leads in turn to thinking about another polygon: the hexagon. The shape of a hexagon reminds the interpreter of the shape of France, and, when thinking about France, they think about eating snails. Clearly, this could lead to further representamens and interpretants forever.

Any interpretant can lead to further signs because the interpretant *itself* can represent something else. That is the basis of unlimited semiosis. The chief interest in unlimited semiosis for the purposes of this paper is the concept of joining two signs together. Note also that it is not completely necessary that this *joining* of signs occur strictly *in the mind*. In particular, when seeing a map of France someone might *draw* a hexagon in response. This drawn hexagon can be considered as an interpretant becoming a sign in an important sense. Therefore, we claim that the interpretant of a sign might have a *physical* manifestation as well as a mental one.

5. A SEMIOTIC MODEL OF METAPHOR

Although metaphor is considered important and is discussed in semiotics research, there is no definitive model of the concept readily available. The general view appears to be that a metaphor sign involves the interaction in some way of *two* signs, which are the tenor and the vehicle of the metaphor (Chandler, 2002; Thwaites et al., 1994). The stance we take in this paper is that a metaphor may well be composed of two signs, but can plausibly be treated as a sign in itself as well. Using this perspective, we apply the Peircean triad to metaphor as shown in figure 3, and will now explain the various parts.

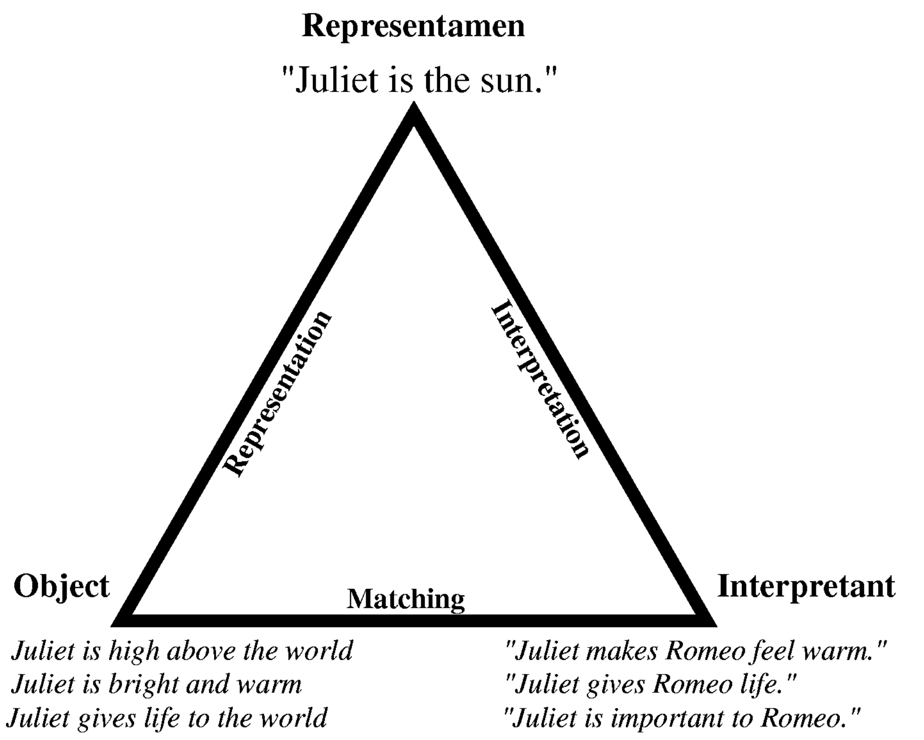

Representamen
"Juliet is the sun."
Representation
Interpretation
Object
Matching
Interpretant
Juliet is high above the world
Juliet is bright and warm
Juliet gives life to the world
"Juliet makes Romeo feel warm."
"Juliet gives Romeo life."
"Juliet is important to Romeo."

Figure #-3. A semiotic model of metaphor.

Essentially, the meaning of the metaphor intended by its author comprises the object, while the expression of the metaphor itself, usually in language, forms the *representamen*. An encounter with the representamen leads a reader to form an *interpretant*, which is what the metaphor is taken to mean by them.

The relations involved are all fairly similar to those already discussed above. The *representation* relation concerns the way the representamen conveys the object. In the case of metaphor this tends to be some linguistic statement which either directly ("Juliet is the sun") or more indirectly ("Arise fair sun and kill the envious moon") specifies the metaphor. The *interpretation* relation concerns the reader's encounter with the metaphor and how they think about it. Finally, the *matching* relation describes how well the final interpretant matches with the intention of the metaphor's author.

An issue, arises, however, when considering the object of a metaphor. Simply suggesting it is the "meaning" of the metaphor is not specific enough for our purposes. To this end, we have turned to the work of George Lakoff and Mark Johnson in *Metaphors We Live By* to provide an answer (Lakoff and Johnson, 1980). In the book, Lakoff and Johnson introduce the concept of *metaphorical entailments*. A metaphorical entailment is the application of some fact about the vehicle to the tenor. Thus, in the example of JULIET IS THE SUN, a metaphorical entailment might be that “Juliet is warm,” because the sun is warm and this quality is transferred to Juliet. These metaphorical entailments “characterize the *internal* systematicity of the metaphor ... that is, they make coherent all the examples that fall under that metaphor.” (Lakoff and Johnson, 1980, p.91) In other words, a set of metaphorical entailments can be said to be the meaning of a metaphor: the object of a metaphor sign.

The question remains as to *which* possible set of metaphorical entailments we should consider to be the object of a metaphor. The standpoint taken in this paper is that this depends on the position from which the sign is being considered. Because our chief occupation is to examine a generative view of user-interface metaphor, the object of a metaphor is considered to be those metaphorical entailments taken to be the meaning of the metaphor by its designer or author. This will be discussed in more detail when the full model of user-interface metaphor is presented. Note also that the metaphorical entailments provide a more specific means of describing the *matching* relation of a metaphor: the relation is strengthened by aspects of the interpretant which match the metaphorical entailments, and weakened by those that do not.

6. A SEMIOTIC MODEL OF USER-INTERFACE ELEMENTS

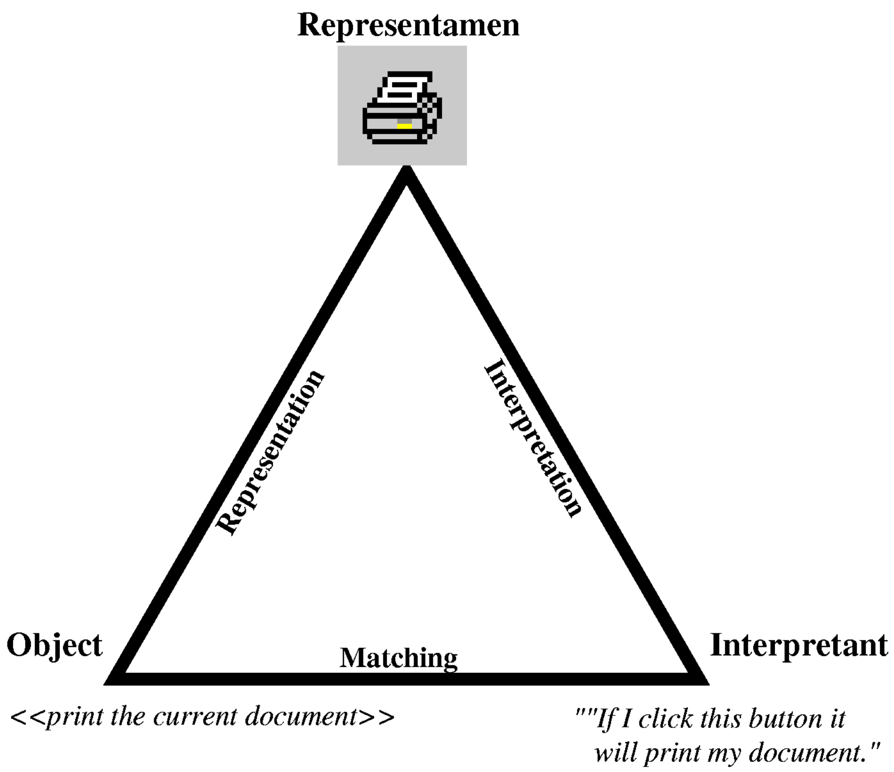

Representamen
Representation
Interpretation
Object
Matching
Interpretant
<<print the current document>>
""If I click this button it will print my document."

Figure #-4. A semiotic model of a user-interface sign.

It is actually quite straightforward to create a triadic model of a user-interface sign. Much of this section utilises the work on a semiotic view of computer icons covered by the present authors in a previous paper (Barr et al., 2003). Figure 4 provides an example application of the model which will guide the following discussion. The example is of the print button presented by Microsoft Word which is clicked in order to print the currently viewed document.

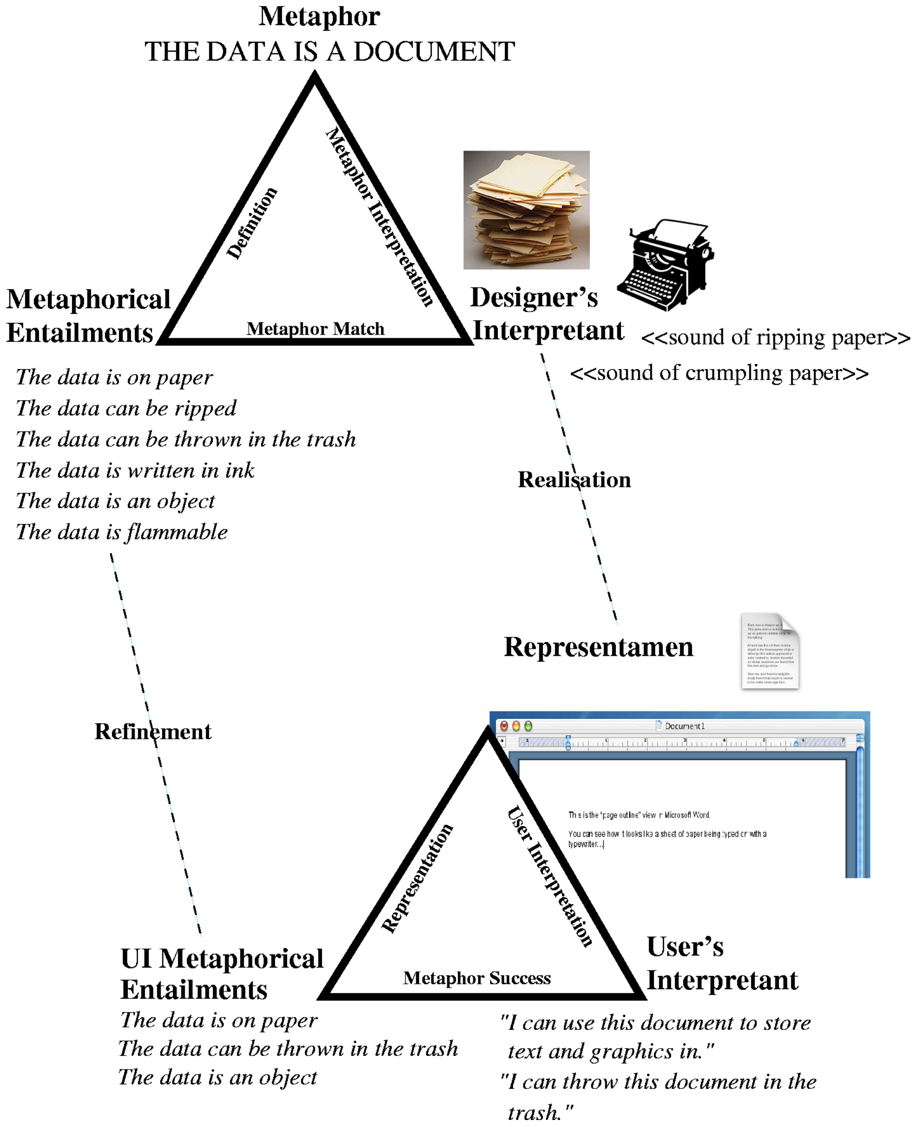
Essentially, the *object* of a user-interface sign is the functionality underlying it, in this case the ability to print a document. The *representamen* is the perceivable aspect of the interface that is intended to convey the object to a user, who, in turn, develops an interpretant. The interpretant can be thought of as the user's mental response to the sign. Note that it could additionally be thought of as the user's general response to the sign, their reactions to it. It therefore might involve physical reactions, for example.

The relations of the triad are similarly easy to apply. The *representation* relation concerns the way in which the representamen actually depicts the underlying functionality. The *interpretation* relation captures the process gone through by a user on encountering the sign, leading to the interpretant. Finally, the *matching* relation concerns how well the user's interpretation of the sign matches the actual functionality it represents.

7. A SEMIOTIC MODEL OF USER-INTERFACE METAPHOR

In order to create a semiotic view of a user-interface metaphor it is necessary to combine the previously discussed views of a metaphor sign and a user-interface sign. As noted in section 4.3, the concept of unlimited semiosis can be used to join signs together. Traditionally, this transition from one sign to another occurs in the mind of an interpreter as they consider a sign and find that their consideration leads to other signs. As this model is based on the designer's viewpoint, however, the joining of the two signs occurs through the designer instead. What is more, the designer is consciously aware of the joining of the two signs as it is done *intentionally*.

The key point of unlimited semiosis we utilise, then, is the notion that the interpretant of one sign can become the representamen of another. If this is considered as a conscious process, done intentionally, then it is similar to using one concept to fuel another in some useful way. This matches well with the idea of using a metaphor to fuel the design of some part of the user-interface. Therefore, the user-interface metaphor sign will involve a metaphor sign which is linked with a user-interface sign. This reflects the necessity of viewing a user-interface metaphor as a cohesive unit, while also recognising that it is divisible into a base metaphor along with a sign in the user-interface.



The data is on paper
The data can be ripped
The data can be thrown in the trash
The data is written in ink
The data is an object
The data is flammable

This is the "page outline" view in Microsoft Word.

You can see how it looks like a sheet of paper being typed on with a typewriter...

The data is on paper
The data can be thrown in the trash
The data is an object

"I can use this document to store text and graphics in."
"I can throw this document in the trash."

Figure #-5. A semiotic model of user-interface metaphor.

Figure 5 displays the model that involves linking the two types of signs already discussed. In particular, the interpretant of the metaphor sign becomes the representamen of the user-interface sign. This presentation of a user-interface metaphor as *two* signs linked together allows discussion of both the perceivable interface elements, and also the underlying metaphor.

The various parts of the sign have been renamed to avoid confusion due to the fact there are now *two* distinct signs involved, the metaphor sign and the user-interface element sign. The following discussion of the model will be guided by the example of the *document* metaphor common to many modern user-interfaces. This example is reflected in figure 5, although note that the figure itself is not intended to convey a *complete* model of that particular sign.

7.1 The Parts of a User-Interface Metaphor

7.1.1 Metaphor

The representamen of the metaphor part of the sign is now simply called the *metaphor*. This is in keeping with Peirce's suggestion that the representamen is the sign in some sense. This part reflects the representation of the metaphor that the designer considers while creating the user-interface metaphor sign. It might be a sign in the designer's mind, or more likely should be written down in the project documentation. Note also that by having the representamen be the metaphor, this raises the possibility of having a more detailed model of the metaphor itself, possibly involving two separate signs, as discussed in section 5. This low-level representation could then be linked through its interpretant to the metaphor which is the representamen of the metaphor sign.

In the context of the document example, the metaphor is that THE DATA IS A DOCUMENT. The idea behind this metaphor is that, instead of viewing information input into the computer as amorphous, raw data, it can be cast as a document instead. The metaphor is especially relevant when the data entered into a computer serves a similar purpose as a written document. Although the data is clearly not *really* a document, the similarity of function allows the metaphor to function.

7.1.2 Metaphorical Entailments

The object of the metaphor part of the sign is now referred to as the *metaphorical entailments* because those are literally what the metaphor is considered to mean. These are the entailments of the metaphor that the overall user-interface metaphor is based on. In particular, these are the entailments the *designer* believes the metaphor to have. They are also *independent* of the user-interface at this point, and are simply entailments of the metaphor in general.

In the context of the document metaphor, the metaphorical entailments are any considerations of documents that might be useful in describing data. Thus a possible (and incomplete) list might be something like:

- The data is an object.
- The data can be written on.
- The data can be read.
- The data contains text, and possibly images and graphs, etc.
- The data can be ripped.
- The data can be typed up.
- The data is usually on white paper.
- The data can be photocopied.
- The data can be written in pencil or pen or ink.
- The data can (sometimes) be edited using twink or an eraser.
- The data contains information.
- The data can be set on fire.
- The data can be picked up and moved from place to place.
- The data can be thrown in a trashcan.
- The data can give you a paper-cut.

It is worth noting here that it is obvious not all of these entailments will be appropriate for use in the final user-interface. Naturally, there are facts about "documents" which are will not be used because they are not applicable in the context of a user-interface. An argument could be made that the metaphorical entailments are those entailments that the designer assumes will be shared by the user. This is similar to Eco's claim that "the author has to assume that the ensemble of codes he relies upon is the same as that shared by his possible reader." (Eco, 1979, p.7)

7.1.3 Designer's Interpretant

The final part of the metaphor half of the overall sign is the *designer's interpretant*. This reflects the result of the designer's consideration of the metaphor. It consists of the various thoughts the designer has about the metaphor overall, while thinking about how to design an aspect of the user-interface. These will be very much related to the metaphorical entailments which are the object of this sign. Note that this interpretant can include mental images, as well as concepts about motion, words, sounds and so forth. In fact, it can be considered as the overall mental reaction to the metaphor. What is more, the designer's interpretant could include any actual work done by the designer prior to actual implementation; it is the thought process which leads to the final implementation and encompasses all that design work.

In the example of the document metaphor, the designer's interpretant is any of the work done by the designer while attempting to figure out the representamen of the user-interface metaphor. Possible aspects of the designer's interpretant might be:

- The designer's mental images of documents, perhaps set down on paper or on the computer for documentation purposes. Specifically, this might involve images of typed documents, documents with images set in them, stacks of paper, and so forth.
- Thoughts and recordings of the sounds involved with documents, such as the sound of a typewriter working, paper tearing, or pieces of paper moving against each other.
- Actions involved with documents, such as typing one up, or throwing one into the wastepaper basket, for instance.
- Thoughts on the tactile sensations involved with documents, such as the feel of paper, or the feeling of hitting typewriter keys.

Note how all of these aspects of the designer's interpretant can be physically documented as a part of the design process. This would make the retroactive identification of the different parts of the user-interface metaphor sign considerably more simple. Note also that all of the aspects of the designer's interpretant can be linked back to the metaphorical entailments that were established. In fact, the designer's interpretant is, in some ways, the realisation of those metaphorical entailments. By comparing the two an idea can be gained of the particular perspective the designer took regarding the entailments of the metaphor.

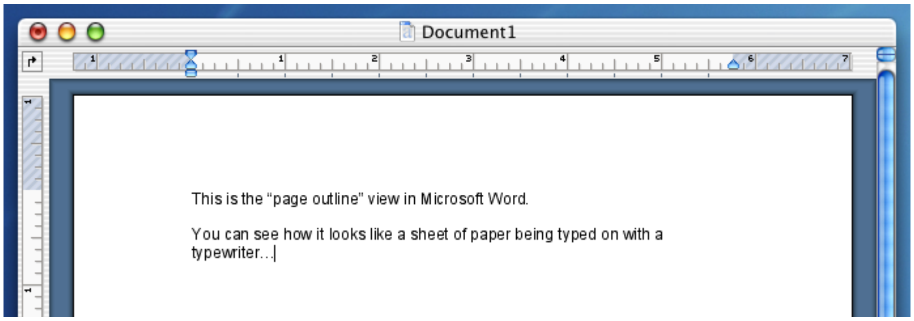
7.1.4 Representamen

Because of the process of unlimited semiosis, the designer's interpretant now becomes the representamen of a new sign. This makes a considerable amount of sense in the context of interface design: the designer's interpretant is effectively the design process which leads up to the actual implementation of the user-interface metaphor. This implementation is the *representamen* of the user-interface sign discussed in section 6. The representamen is arrived at when the designer turns their interpretations of the metaphor into an actual user-interface element, and therefore can be considered as the realisation of that design. Here, the designer's interpretant, which is an interpretant of the metaphor sign, becomes the representamen in the user-interface via implementation. The representamen consists of all perceivable aspects of a user-interface pertaining to the particular metaphor.

This can be made clear by considering the example of the document metaphor. The designer's interpretant for this metaphor involves the designer's ideas about the sorts of ways a document behaves, based on the metaphor. The representamen of the document metaphor consists of all aspects in the system that are used to represent these ideas. A few examples of the representamen are as follows:

- The document icon, as shown in figure 6. This visually represents the way that a document might look from afar, in a fairly abstract manner.
- The "Page Outline" view in Microsoft Word, which presents the current document being worked on as a sheet of A4 paper which can be typed on as in figure 7.
- The ability to pick up and move document icons on the desktop, reflecting the concept that documents are physical objects.
- The ability to "throw" a document into the trashcan.
- Explicit referral to collections of data as documents using language, as shown in figure 8.
- The ability to store a document in a folder.

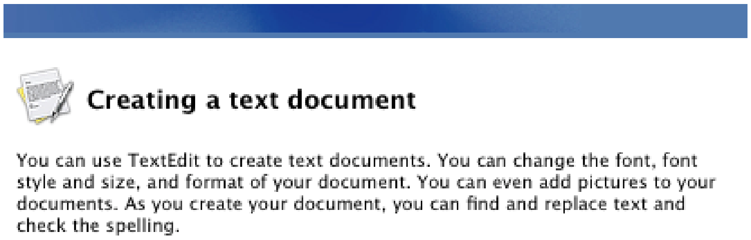
Figure #-6. The standard document icon in MacOS X.



This is the "page outline" view in Microsoft Word.

You can see how it looks like a sheet of paper being typed on with a typewriter...|

Figure #-7. The standard view of the "paper" in Microsoft Word.



Creating a text document

You can use TextEdit to create text documents. You can change the font, font style and size, and format of your document. You can even add pictures to your documents. As you create your document, you can find and replace text and check the spelling.

Figure #-8. Example of explicit referral to a collection of data as a "document" in the MacOS X help system.

Note how, in representing this metaphor, other possible metaphors are invoked, such as a trashcan and a folder. By combining metaphors, the overall illusion of metaphorical interaction is strengthened.

7.1.5 UI Metaphorical Entailments

Given that a new sign is now being discussed, there is therefore a new object of the sign. It is quite interesting to note that the lower (user-interface) sign mirrors the upper (metaphor) sign quite closely. In fact, the user-interface sign can be considered as the realisation of the metaphor sign in a different format: a user-interface implementation. To that end, the object of the user-interface sign is also a set of metaphorical entailments. This new set of metaphorical entailments will be called the *UI metaphorical entailments*, to distinguish it from the earlier set. The UI metaphorical entailments differ from the metaphorical entailments in that they are a set specifically tailored to the user-interface. In particular, they contain only those entailments that are relevant to the implementation of the interface. Basically, the point is that not all of the entailments of a metaphor will be appropriate for actual implementation. Therefore, the set of metaphorical entailments must be refined to the set of UI metaphorical entailments.

In the document example, not every one of the metaphorical entailments is useful for implementation. The designer realises this over the course of establishing the designer's interpretant. Therefore, a new set of metaphorical entailments is established when the representamen is being created. These UI metaphorical entailments reflect the actually applicable entailments about documents which will work in the user-interface. A sample listing could be as follows:

- The data is an object.
- The data can be written on.
- The data can be read.

- The data contains text, and possibly images and graphs, etc.
- The data can be typed up.
- The data is usually on white paper.
- The data contains information.
- The data can be picked up and moved from place to place.
- The data can be thrown in a trashcan.

Note that various metaphorical entailments have been omitted in this listing. For example, entailments such as “the data can be ripped in two,” and “the data can be set on fire” are no longer present because such details are not implemented. The complete set of UI metaphorical entailments defines the functionality made available by the implemented document metaphor. Note, however, that they may not define all the functionality associated with the element in the user-interface. For example, it is possible to change the colour of the text in a document at will. This is *not* a metaphorical entailment because you cannot easily do such a thing to a real world document. The ability to combine metaphorical and non-metaphorical functionality is one of the powerful aspects of user-interface metaphors.

7.1.6 User's Interpretant

The final part of the user-interface metaphor model is the *user's interpretant*. As already discussed, this interpretant might be the Model User's interpretant, or a real user's interpretant, depending on what the model is being applied to. If it is the Model User's interpretant, then the model represents the ideal process through which the sign goes, culminating in its successful interpretation. If a real user's interpretant is used, then the model represents a kind of user testing, where the final interpretant can be compared with the UI metaphorical entailments to establish how successful the sign was in conveying the correct way of interacting with it. Note also that the process of unlimited semiosis can be continued upward from the user's interpretant. In this case it can be used to model possible or actual thought processes undergone when interpreting the user-interface. In particular, it might be used as a means to show how the user must “reverse engineer” the perceivable representamen in order to establish the underpinnings of the sign, including the underlying metaphor.

For the document example, the Model User's interpretant will simply be a collection of thoughts which are similar to the UI metaphorical entailments. That is, the Model User will understand, through the representamen, the kinds of things they can expect to do with the user-interface via the document metaphor. This means they will think things such as “I can throw a document into the trash” or “I can type into this document.” In a user testing context, the user's interpretant will be established by the examination of real users. In this case, the interpretant must be somehow elicited by observation, or by direct questioning, for example. In this case, the designers must have users interact with an implementation of the document metaphor, and then find out what it is they think. If the users think similarly to the Model User, then the metaphor is successful. If the users think things such as "I hope I don't accidentally knock this document off the desktop," then there are issues in the representamen to be resolved.

7.2 The Relations of the User-Interface Metaphor

The relations of this model of a user-interface metaphor are somewhat more complex than those already discussed in sections 5 and 6. This is because of the linking provided by the concept of unlimited semiosis. Not only must the interpretant of the metaphor sign be linked with the representamen of the user-interface sign in a kind of relation, but the objects of the two signs, both sets of metaphorical entailments, have a special relationship too. This section will outline each of the relations in turn and explain what they represent in terms of the design process.

7.2.1 Definition

Between the metaphorical entailments and the metaphor is the relation of *definition*. That is, the set of metaphorical entailments can be considered as the definition or meaning of the metaphor itself. This is reinforced by the position of Lakoff and Johnson, as discussed in section 5. In fact, they even go so far as to list a large number of metaphorical entailments for a metaphor and then write that these entailments "form a coherent whole as instances of the metaphor." (Lakoff and Johnson, 1980, p.140) It can be taken from Lakoff and Johnson that the metaphorical entailments, along with their overall structure and coherence, define the content or meaning of a metaphor.

Thus, in the context of the example, the metaphorical entailments listed (along with the rest which would complete the set) are the definition of the metaphor THE DATA IS A DOCUMENT.

7.2.2 Metaphor Interpretation

The relation of *metaphor interpretation* holds between the metaphor and the designer's interpretant. This relation represents the designer's thought process as they consider the metaphor and ponder what it means. The process leads to the designer's overall thoughts as embodied in their interpretant. This relation, therefore, concerns the designer's brainstorm as to what the metaphor might mean *in general* and independently of any strict design and implementation considerations.

In the example, this relation concerns the process of the designer thinking about the document metaphor, and then coming up with possible ideas for its use in the user-interface. This ultimately yields the designer's interpretant.

7.2.3 Metaphor Match

The *metaphor match* relation embodies how the designer's thoughts on the metaphor tie in to the metaphor's entailments. Because the designer effectively defines these entailments, the match ought to be quite close. It is important to note, however, that this relation can be actively examined. This can be done by having the designer's interpretant assessed for specific matches between it and the metaphorical entailments. This means that it can be discovered whether all of the entailments have been taken into account.

In the example, it can be seen that the ideas in the designer's interpretant do realise various of the metaphorical entailments. Thus, the aspect of the designer's interpretant which concerns the visual aspects of a document realises metaphorical entailments such as “the data is on paper,” “the data can have a fold in it,” “the data can have text typed on it,” and so forth.

7.2.4 Realisation and Refinement

Now that we have discussed the relations relating strictly to the metaphor sign, we must address the important issue of the relations *linking* the two signs involved in the model. There are two links here, one between designer's interpretant and representamen, and one between the two sets of metaphorical entailments. The *realisation* relation concerns the process of the designer's interpretant becoming a real implementation in a user-interface: the representamen. This is the process of actual implementation of the ideas about the metaphor. Thus, in the example, the realisation concerns the transformation of the designers concepts about how documents look and behave into actual interface elements and functionality.

The process of realisation is intricately linked with the parallel relation of *refinement*. The refinement relation concerns the process of narrowing down the set of metaphorical entailments to just those that will be true of the actual user-interface implementation: the UI metaphorical entailments. Because it is so tied to the implementation, this refinement will take place simultaneously with the realisation of the designer's interpretant as representamen in the user-interface. In the example, the refinement relation reflects the designer's consideration of the exact metaphorical entailments that will be used to define the functionality of the document metaphor in the user-interface. This is a culling process, where useful entailments, such as "the data contains text," are retained while inappropriate entailments such as "the data can give you a paper-cut" are jettisoned.

7.2.5 Representation

The *representation* relation concerns how the representamen links with the UI metaphorical entailments. Once again, this can be formalised by specifically indicating what these links are. In this way, it becomes possible to check on the coverage of the interface metaphor. That is, it can be established whether all the UI metaphorical entailments are indicated in some specific way by the representamen, and whether the representamen indicates any non-existent entailments.

In the document example, the representation relation concerns how the implementation of the document metaphor conveys the UI metaphorical entailments already discussed. An example linking between the two is that visual representation of the document icon looks like a piece of paper. This ties to the UI metaphorical entailments which suggest a document can be moved, and even thrown in the trash.

7.2.6 User Interpretation

The *user interpretation* relation is the process of the user interpreting the representamen, as is traditional in Peircean semiotics. If the interpretant is that of the Model User then this interpretive process will be the ideal one, making all the correct inferences. If the interpretant is that of a real user, the user interpretation relation will have to be elicited from the user via interviews, thinking-aloud and other techniques.

In the document example, the user interpretation relation occurs while the user is interacting with the implementation of the document metaphor in the user-interface. The relation ends with the user's interpretation of how that aspect of the interface functions.

7.2.7 Metaphor Success

The final relation is that of *metaphor success*. This relation concerns how well the user's interpretant is matched with the UI metaphorical entailments. In other words, this relation is a measure of how well understood the underlying meaning of the user-interface metaphor is by the user. The more the user has made correct inferences about how to interact with the system, the stronger this relation will be.

In the document metaphor example, this relation will concern how well the user has established the entailments about the DATA IS A DOCUMENT metaphor by interacting with the representamen. Instances where the user's interpretant matches with the UI metaphorical entailments indicate a degree of success of the metaphor. Instances where the user imagines functionality not present, or fails to see functionality which *is* present indicate a degree of failure.

8. FURTHER WORK

Although the semiotic model of user-interface metaphor just presented has been shown practically applicable to some degree via the document example, there is clearly more work to be done in that area.

One possibility is to use the semiotic model during user testing. By presenting user responses within the model it may prove possible to trace problems with an interface design back to their origin. Clearly, problems can arise anywhere between the representamen and the metaphorical entailments of a metaphor. For example, a representamen may indicate a UI metaphorical entailment that is not present. Alternatively, this problem might be traced back to the process of refinement, where a metaphorical entailments was removed from the set of entailments, but this was not reflected in the final representamen. Being able to pin-point the cause of a problem for users would be a valuable asset. The work concerning the use of metaphorical entailments to analyse metaphor content in (Barr et al., 2002) partly concerns this sort of analysis.

A further important avenue of future research is the application of the semiotic model to the design process. Specifically, it would be desirable to teach the model to designers about to begin an interface design, and to see how this knowledge influenced their activities. It could be predicted that the more structured vocabulary would generally aid the designers in discussion, for example.

The analysis of icons using a semiotic model presented in (Barr et al., 2003) goes some way to suggesting that, once a semiotic model is in place, other semiotic techniques can be applied. In the paper, Peircean sign-types are used to classify computer icons. This kind of transference of traditional semiotic analysis techniques to user-interface metaphors should be made possible by our model.

One more possible step is to perform a detailed case study of actual user-interface metaphors in some software. This has been done to some extent with three major metaphors from Microsoft Office's Project Gallery in (Barr, 2003). It will be a major undertaking to analyse an entire interface's metaphors semiotically, but it may well prove very valuable to understanding the metaphors' individual properties, as well as their interactions with each other.

9. CONCLUSIONS

In this paper we have identified a particular issue with the current approach to user-interface metaphors: the concept itself is not well understood. In response to this we have shown in considerable detail how Peircean semiotics can be used to produce a highly structured model of user-interface metaphor. We claim that this model provides interface designers with a consistent vocabulary for discussion, as well as a strong analytical approach to the concept.

Additionally, the application of semiotics to user-interface metaphor clearly allows the possibility of applying further semiotic analysis. In other words, the semiotic model provides a foot in the door of analysis normally applied to other disciplines. Because we now have a semiotics of user-interface metaphors, these analyses can be directed at that concept also.

NOTES

1. This paper will follow the convention of George Lakoff and Mark Johnson's book Metaphors We Live By (Lakoff and Johnson, 1980) in presenting metaphors in a small-caps font.
2. Emphasis added by authors.

REFERENCES

Peter B. Andersen and Michael May. Tearing up interfaces. In Kecheng Liu, Rodney J. Clarke, Peter Bøgh Andersen, and Ronald K. Stamper, editors, *Information, Organisation and Technology: Studies in Organisational Semiotics*, pages 299-337. Kluwer Academic Publishers, 2001.

Peter Bøgh Andersen. A semiotic approach to programming. In *The Computer as Medium*, pages 16-67. Cambridge University Press, 1993.

Peter Bøgh Andersen. *A Theory of Computer Semiotics*. Cambridge Series on Human-Computer Interaction. Cambridge University Press, 1997.

Peter Bøgh Andersen. What semiotics can and cannot do for HCI. In CHI'2000 Workshop on Semiotic Approaches to User Interface Design., 2000.

Apple Computer, Inc. Staff. *Macintosh Human Interface Guidelines*. Addison-Wesley, 1992.

Pippin Barr, Robert Biddle, and James Noble. A taxonomy of user-interface metaphor. In Steve Jones and Masood Masodian, editors, *Proceedings SIGCHI-NZ Symposium on Computer-Human Interaction*. The New Zealand Chapter of ACM SIGCHI, 2002.

Pippin Barr. User-interface metaphor in theory and practice. MSc. Thesis, Victoria University of Wellington, Department of Mathematical and Computing Sciences, 2003.

Pippin Barr, Robert Biddle, and James Noble. Icons R Icons. In Robert Biddle and Bruce Thomas, editors, *User Interfaces 2003: Fourth Australian User Interface Conference*, pages 25-32, 2003.

John M. Carroll and Robert L. Mack. Learning to use a word processor: By doing, by thinking, and by knowing. In Ronald M. Baecker, Jonathan Grudin, William A. S. Buxtin, and Saul Greenberg, editors, *Readings in Human-Computer Interaction: Toward the Year 2000*, pages 698-717. Morgan Kaufmann Publishers, Inc., 1995.

John M. Carroll, Robert L. Mack, and Wendy A. Kellogg. Interface metaphors and user interface design. In M. Helander, editor, *Handbook of Human-Computer Interaction*, pages 67-85. Elsevier Science Publishers, 1988.

John M. Carroll and John C. Thomas. Metaphor and the cognitive representation of computing systems. *IEEE Transactions on Systems, Man. and Cybernetics*, 12(2):107-116, March/April 1982.

Tiziana Cataci, Maria F. Costabile, and Maristella Matera. Which metaphor for which database? In M. A. R. Kirby, A. J. Dix, and J. E. Finlay, editors, *People and Computers X: Proceedings of HCI'95*. Cambridge University Press, 1995.

Daniel Chandler. *Semiotics: The Basics*. Routledge, 2002.

John H. Connolly and Iain W. Phillips. User-system interface design. In Kecheng Liu, Rodney J. Clarke, Peter Bøgh Andersen, Ronald K. Stamper, and El-Sayed Abou-Zeid, editors, *Organisational Semiotics: Evolving a Science of Information Systems*, pages 119-132. Kluwer Academic Publishers, 2002.

Clarisse Sieckenius de Souza. The semiotic engineering of user interface languages. *International Journal of Man-Machine Studies*, 39(5):753-773, 1993.

Michael Dertouzos. *The Unfinished Revolution*. HarperCollins Publishers, Inc., 2001.

Umberto Eco. *The Role of the Reader*. Indiana University Press, 1979.

Umberto Eco. *The Limits of Interpretation*. Indiana University Press, 1990.

Thomas D. Erickson. Working with interface metaphors. In Brenda Laurel, editor, *The Art of Human-Computer Interface Design*, pages 65-73. Addison-Wesley Publishing Company, 1990.

Joseph Goguen. An introduction to algebraic semotics, with applications to user interface design. In Chrystopher Nehaniv, editor, *Computation for Metaphor, Analogy and Agents*, volume 1562 of LNAI, pages 242-291. Springer-Verlag, 1999.

A. J. Greimas. *Structural Semantics*. University of Nebraska Press, 1983.

F. Halasz and T. P. Moran. Analogy considered harmful. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 383-386, 1982.

Steven Johnson. *Interface Culture: How New Technology Transforms the Way We Create and Communicate*. Harper SanFrancisco, 1997.

Alan Kay. User interface: A personal view. In Brenda Laurel, editor, *The Art of Human-Computer Interface Design*, pages 191-207. Addison-Wesley Publishing Company, 1990.

George Lakoff and Mark Johnson. *Metaphors We Live By*. The University of Chicago Press, 1980.

Kechend Liu, Rodney J. Clarke, Peter Bøgh Andersen, Ronald K. Stamper, and El-Sayed Abou-Zeid, editors. *Organisational Semiotics: Evolving a Science of Information Systems*. Kluwer Academic Publishers, 2002.

Jay Lundell and Steve Anderson. Designing a `Front Panel' for Unix: The evolution of a metaphor. In Irvin R. Katz, Robert Mack, and Linn Marks, editors, *Proceedings of CHI'95 Conference on Human Factors in Computing Systems*, pages 573-580, New York, 1995. Addison-Wesley Publishing Co.

Kim Halskov Madsen. A guide to metaphorical design. *Communications of the ACM*, 37(12):57-62, December 1994.

Grant Malcolm and Joseph A. Goguen. Signs and representations: Semiotics for user interface design. In Ray Paton and Irene Nielson, editors, *Visual Representations and Interpretations*, pages 163-172. Springer, 1998.

Michael May and Peter B. Andersen. Instrument semiotics. In Kecheng Liu, Rodney J. Clarke, Peter Bøgh Andersen, and Ronald K. Stamper, editors, *Information, Organisation and Technology: Studies in Organisational Semiotics*, pages 271-298. Kluwer Academic Publishers, 2001.

Microsoft Corporation. *The Windows Interface Guidelines for Software Design: An Application Design Guide*. Microsoft Press, 1995.

S. Joy Mountford. Tools and techniques for creative design. In Brenda Laurel, editor, *The Art of Human-Computer Interface Design*, pages 17-30. Addison-Wesley Publishing Company, 1990.

Kevin Mullet and Darrell Sano. *Designing Visual Interfaces: Communication Oriented Techniques*. SunSoft Press, 1995.

Mihai Nadin. Interface design: A semiotic paradigm. *Semiotica*, 69(3):269-302, 1988.

Theodor Holm Nelson. The right way to think about software design. In Brenda Laurel, editor, *The Art of Human-Computer Interface Design*, pages 235-243. Addison-Wesley Publishing Company, 1990.

Jakob Nielsen. *Usability Engineering*. Academic Press, 1993.

Donald A. Norman. *The Invisible Computer*. The MIT Press, 1998.

Charles Sanders Peirce. *Collected Papers*. four volumes. Harvard University Press, 1934-1948.

Gerald Prince. *Narratology: The Form and Functioning of Narrative*. Mouton de Gruyter, 1982.

William Shakespeare. *Romeo and Juliet*. Morrison and Gibb, Ltd., 1933.

Michael Smyth, Ben Anderson, and James L. Alty. Metaphor reflections and a tool for thought. In M. A. R. Kirby, A. J. Dix, and J. E. Finlay, editors, *People and Computers X: Proceedings of HCI'95*. Cambridge University Press, 1995.

Tony Thwaites, Lloyd Davis, and Warwick Mules. *Tools for cultural studies: an introduction*. Macmillan Education, 1994.

Kaisa Väänänen and Jens Schmidt. User interface for hypermedia: How to find good metaphors? In Catherine Plaisant, editor, *Proceedings of the CHI'94 Conference Companion on Human Factors in Computing Systems*, pages 263-264, 1994.

Lucy Anne Wozny. The application of metaphor, analogy, and conceptual models in computer systems. *Interacting with Computers*, 1(3):273-283, 1989.