Elastic Metaphors: Expanding the Philosophy of Interface Design

Gerald R. Khoury
IBM Global Services
Gerald.R.Khoury@uts.edu.au

Simeon J. Simoff
Institute for Information and Communication Technologies
University of Technology Sydney
simeon@it.uts.edu.au

Abstract

Metaphors are generally accepted as essential to the design of effective human computer interfaces. However, "The generally assumed theoretical benefits of user interface metaphor are supported by surprisingly little empirical evidence." (Blackwell, 1998)

This paper discusses the concept of "concrete metaphor" and the problems that it presents in interface and interaction design. Concrete metaphors are composed of objects that users are familiar with from their everyday experience (L'Abbate and Hemmje, 1998). Since we live in a physical world, then it seems natural that computer interfaces should resemble as closely as possible – physical objects. We already know how these devices work, and so a metaphor based on the known should help us to understand the unknown. After all, "The essence of metaphor is understanding and experiencing one kind of thing in terms of another." (Lakoff and Johnson, 1980) Certainly, this has been the prevailing school of thought when discussing the application of metaphor to Human Computer Interface (HCI) design.

However, there is another school of thought that the use of metaphor is *detrimental* to HCI design. For example, Halasz and Moran (1982, p. 386) considered analogy as "dangerous when used for detailed reasoning about computer systems - this is much better done with abstract conceptual models."

Our argument is that metaphor can be used for the representation and explanation of abstract conceptual models. Recent work by Lakoff and Núňez (2001) describes the notion of conceptual metaphor – a cognitive mechanism that derives abstract thinking from the way we function in the everyday physical world[1].

The new approach towards the application of metaphor to human computer interactions, proposed in this paper, is based on the concept of 'elastic metaphors'. The paper presents the features of elastic metaphors and methods for its construction.

Keywords: Metaphor, elastic metaphor, HCI, human computer interaction.

1 Introduction

The approach presented in this paper parallels the use of metaphor in cognitive linguistics (Lakoff and Johnson, 1980), where the terms *source* and *target*[2] refer to the conceptual spaces connected by the metaphor. The structure of the source domain is projected onto the target domain in a way that is consistent with the inherent target domain structure (Lakoff, 1993; Turner, 1994).

Contemporary research has shown us that metaphor, far from being just a figure of speech, is central to everyday communication and learning. (Lakoff and Johnson, 1980) (Lakoff and Nunez, 2000) It has been argued that only by relating a new concept (the 'target') to a well understood, everyday object (the 'source'), can one develop new knowledge and understanding and that without this process, it is impossible for humans to carry out abstract thinking.

Similarly, it is widely believed that metaphor is essential, to the design of effective information systems. In this case, metaphor is believed to provide value by:

- Reducing the effort required to understand the conceptual system model by providing an

[1] In the research literature the target is variously referred to as the primary system or the topic, and the source is often called the secondary system or the vehicle.

analogy between the new (target) system and a known (source) system.

- Assisting in specific task problem solving by allowing the user to extend their working knowledge of the target system, based on their understanding of the source system.

However, while humans use metaphor quite naturally and instinctively, the explicit and overt application to I.T. domains *cannot* be assumed to carry the same benefits. Indeed, "The generally assumed theoretical benefits of user interface metaphor are supported by surprisingly little empirical evidence." (Blackwell, 1998)

Much of the evidence that does exist to support the use of metaphor appears to misattribute benefits from other sources onto the metaphor, or misattribute problems with the metaphor to other aspects of the system! For instance, the trend to incorporate metaphor into Graphical User Interfaces (GUI's), can easily lead one to misattribute the benefits of the GUI to the metaphor.

There are examples of experiments where the use of a metaphor has been empirically proven to be misleading and confusing. (Hammond and Allison, 1987) Yet the metaphor is declared successful on the basis that (a) the system probably wasn't being used correctly (b) the users *believed* the metaphor was valuable. Similarly, Blackwell (1998) cites several other examples where claims are made purporting to show the value of metaphor in user interface design, *despite* evaluation results that show the contrary. Blackwell's own empirical research shows that metaphor has little effect on problem solving.

The case then, for the value of metaphor in the realm of I.T. systems, is far from proven and more research is needed in this area. We know that metaphor is crucial to learning and yet the results showing the value of metaphor in information systems design are far from convincing. At the same time, the *pitfalls* of using metaphor are easily demonstrable.

It may be that computing has presented society with such a 'radical novelty' and 'sharp discontinuity', that any use of metaphor and analogy to try to link new concepts to more familiar ones, is misguided. (Dijkstra, Denning et al., 1989) Perhaps we *should* throw away the metaphor, and "begin designing devices that have no metaphor, no real-world analogy." (Tristram, 2001) Yet, if everyday language is impossible without recourse to metaphor, how can we possibly hope to avoid using metaphor in a new field like information technology? Is there a better class of metaphor that could be used?

2 The Problem with Concrete Metaphors

Most contemporary computer system applications and interfaces are grounded in the world of metaphor. Generally, these are *concrete metaphors*.

Concrete metaphors are based on objects that users are familiar with from their everyday experience (L'Abbate and Hemmje, 1998). For instance, we have the desktop metaphor composed of filing cabinets, trash bins, and windows through which we can view the world of information. And thus, "… we typically conceptualize the non-physical in terms of the physical - that is, we conceptualize the less clearly delineated in terms of the more clearly delineated." (Lakoff and Johnson, 1980)

One of the problems we encounter is the assumption that, since many of our physical artefacts (upon which we base concrete metaphors) have developed over long periods, they are reasonably optimal. For instance, an office is the best way to organise a business work environment, a filing cabinet is the best way to store information, a 'clock face' dial is the best way to display variables, and so on. In a world bound by physical, engineering and economic limitations, this assumption is valid. However, in a computing environment, these limitations fade away and information can often be conveyed in a far more effective manner than physical devices.

> *"… by tying an interface to concepts which prevail in non-electronic environments, one is not taking full advantage of the benefits that can accrue from using the electronic medium. … For example, a 'filing cabinet' metaphor can be as restrictive as the real-life filing cabinet." (Gardiner and Christie, 1987) p.230.*

On the other hand, the information being conveyed by a computer is also often far more complex and diverse than would be conveyed by a single physical device. As this information cannot suitably be conveyed using a single concrete metaphor, we find convoluted aggregations of metaphors being created to cover the breadth of functionality that is required (Halasz and Moran 1982, p.384).

This "addenda" also causes other problems. One of the perceived benefits of metaphor is in assisting users to solve problems. However, it is precisely in this area that metaphor can cause the greatest of difficulties. When multiple metaphors need to be used to cover the target domain, then it becomes difficult for the user to work out which metaphor applies to their particular problem, or to anticipate new metaphors they have not yet encountered.

3 The Solution - Elastic Metaphors

We have discussed various forms of metaphor and have highlighted some of the difficulties they present. Yet, given that metaphor is so intrinsic to communicating and understanding (to the extent that perhaps *all* knowledge is based on metaphor) (Indurkhya, 1994) what alternatives are available?

There is another rich source of metaphor, and that is social life. After all, "Information technology is arguably, like society itself, an abstract concept." (Marakas, Johnson et al., 2000) Based on an analysis and deconstruction of social life, we have developed the concept of an 'elastic metaphor'.

While concrete metaphors focus on objects, elastic metaphors focus on societal structures. Elastic metaphors allow the invention of tailored, conceptually focussed metaphors that are not confined by physical world limitations, and which *are* optimised to make the best use of human cognitive and perceptive capabilities. The elastic metaphor is an artificial, cognitive construct that, although having pre-defined qualities and attributes, is malleable enough that it can be shaped to reflect the behaviour of a wide variety of systems.

Figure 1 shows the relationship between elastic and concrete metaphors with respect to "scope" and "level of description". "Scope refers to the number of concepts … that the metaphor addresses." (Hammond and Allison, 1987) (p.77) Level of description refers to the granularity of the knowledge being conveyed.

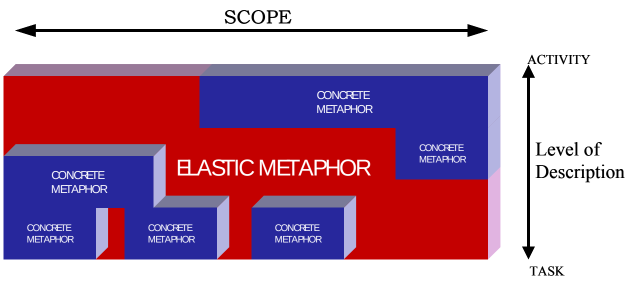

Figure 1 - Scope and Level of Metaphors (partially based on (Hammond and Allison 1987))

An analogy for elastic metaphors is the twelve-bar blues. The twelve-bar blues is a very tightly defined structure, and most tunes based on this make no variation to the basic structure. Despite this constraint, an infinite number of twelve-bar blues based tunes can still be produced. Rather than acting as an impediment to creativity, this structure has fostered immense creativity and has even given birth to new musical forms such as Jazz.

3.1 Structure of an Elastic Metaphor

While concrete metaphors have *objects* as their source, elastic metaphors have *conceptual frameworks* as their source. Arguably, the conceptual structure most familiar to humans is the structure of society.

According to Giddens' Theory of Structuration (Giddens, 1984), there is an interdependency between humans (*actors*) and societal structures (*resources* and *rules*) that is manifest through specific *actions*. An actor is an individual who can exert power in order to produce an effect. Resources are "structured properties of social systems, drawn upon and reproduced by knowledgeable agents in the course of interaction." Rules refer to the sanctioned modes of conduct, and an action is an activity that is performed.

Figure 2 shows the elastic metaphor formalised in terms of entities and relationships.

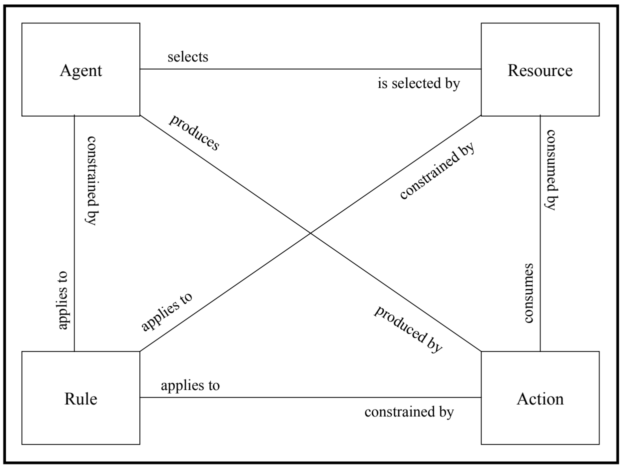

Figure 2 - Elastic Metaphor Entity-Relationship Diagram

Actors, Rules, Resources and Actions serve as the cornerstones of the Elastic Metaphor. Because these entities are universal, we can use them to develop a myriad of new elastic metaphors that cross social, cultural and educational boundaries.

The elastic metaphor can also be viewed as a *class* of metaphor. Within this class, we can form subclasses. For example, games, auctions, committees, etc. can each be viewed as Elastic metaphor subclasses. Table 1 shows the mapping between the Elastic Metaphor Class and these sample subclasses.

		ENTITIES			
Class	**Elastic Metaphor**	**Actors**	**Actions**	**Rules**	**Resources**
Subclass	Game Metaphor	*Players*	*Play to score points*	*Game Rules*	*Tokens*
	Auction Metaphor	*Buyers Sellers*	*Buy / Sell Decisions*	*Auction House Rules/ Legislation*	*Money / Communication Tools*
	Committee	*Members*	*Motion / Vote / Discuss*	*Constitution*	*Funding / Member Skills*

Table 1 – Elastic Metaphor Class-Subclass Mappings

3.2 Characteristics of an Elastic Metaphor

Elastic metaphors can be used to engineer an 'ideal' metaphor for the target system. We come to terms with the system "on account of its own internal consistency." (Dijkstra, 1985) rather than using a metaphor based on archaic, everyday objects. We can have a much tighter correspondence than any concrete metaphor can provide. Moreover, we liberate the ability of the computer to process and display information with far more power and flexibility than any other man made creations.

Elastic metaphors can be developed with a wide enough scope to cover all the functional elements of the system without introducing a large number of mixed metaphors. It is better to have a single metaphor that covers the domain, for “when discourse becomes full of conflicting metaphors, it may be difficult for the uninitiated to keep their bearings.” (Johnson, 1994)

The actor entity could be applied in several ways. In relation to traditional interfaces, the inclusion of an actor construct in the system model ensures that user attributes are taken into account even though they are actually ‘outside’ of the interface. In relation to virtual worlds or virtual reality, the actor may be explicitly represented by an avatar.

3.3 Applying Elastic Metaphors

Elastic metaphors have applicability to both the re-engineering of existing interfaces, as well as the construction of new interfaces. We deal with both cases in the following sections.

3.3.1 Developing New Interfaces

Elastic metaphors can be used to develop new interfaces. In this situation, elastic metaphors are applied in the following stages:

1. Construct the elastic metaphor subclass
2. Identify the required functions
3. Map the required functions onto the elastic metaphor subclass
4. Construct the interface

In the following example, a new, improved ‘email’ interface is constructed.[2] For the purposes of this exercise, we define an email system as a system that allows asynchronous communications between two or more hosts using text and files.

3.3.1.1 Elastic Metaphor Subclass Construction

The first step in applying an elastic metaphor to the design of an interface is to choose an appropriate elastic metaphor subclass. Once a potential candidate has been identified, an entity mapping as demonstrated in Table 1 will easily show if the elastic metaphor is appropriate. The game metaphor will be used in this example however many other metaphors could also be used.

3.3.1.2 Function Identification

The commonly provided user functions of an email system typically include:

- Mailing messages
- Filing messages
- Deleting messages
- Printing messages
- Selecting a message from a list

3.3.1.3 Mapping

The required functions can be mapped onto the game metaphor as shown in figure 3.

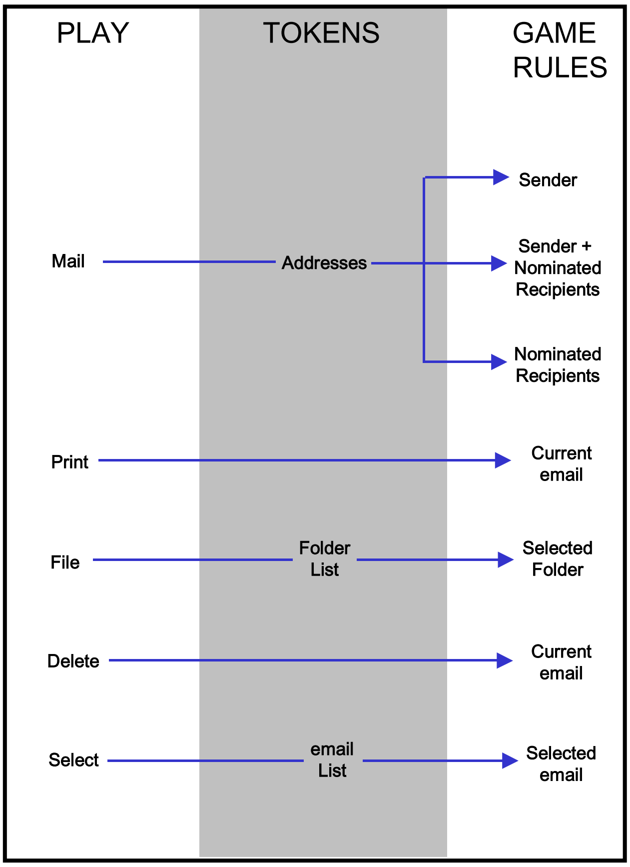

Figure 3 - New Interface Function Mapping

3.3.1.4 Construction

The set of functions shown in figure 4 is required to represent each of the Actions (Plays) and would be provided as part of the new email interface.

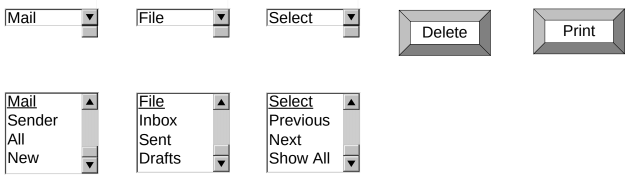

Figure 4 - new interface functions

---

[2] In this context, the term ‘email’ really becomes an anachronism, as the name itself suggests a concrete metaphor.

A significant advantage of elastic metaphors is flexibility and expandability. When the first email systems were developed, technologies for synchronous communication (e.g. instant messaging) did not exist. Thus, the term "email" appeared rather appropriate, as there is a strong analogy between asynchronous messaging and traditional mail.

Instant messaging has now become popular, but the email metaphor does not readily accommodate it. Hence, suppliers offering both email and instant messaging services usually provide them as separate products. By building the email interface using an elastic metaphor, we have used a metaphor that can easily be expanded to incorporate other forms of human communications such as instant messaging, groupware, game playing etc.

3.3.2 Reengineering Existing Interfaces

To reengineer existing interfaces, elastic metaphors are applied in the following stages:

1. Construct the elastic metaphor subclass
2. Analyse the existing interface
3. Decompose the existing interface
4. Perform a functional transformation onto the elastic metaphor subclass
5. Recompose the interface

3.3.2.1 Elastic Metaphor Subclass Construction

The game metaphor will be used again for the sake of comparison.

3.3.2.2 Interface Analysis

A functional analysis of the original user interface is performed. What buttons, menus or other devices are presented and what path is taken when these functions are selected?

3.3.2.3 Decomposition

The elastic metaphor subclass entities are identified e.g. for a Game subclass, Actions become Plays (or Moves). Attributes are also identified e.g. if the metaphor is being applied to an email interface, then the play attributes may be 'mail' and 'print'.

The original interface is then decomposed by mapping it against the subclass entities and attributes. This can be done using a mapping such as that shown in figure 6.

3.3.2.4 Transformation

Using the mapping, the interface functions are transformed according to the following rules:

Each discrete Action class entity transforms to a user function.

If the Action requires the user to select a Resource, then the user function is presented as a 'menu'. Otherwise, it is presented as a simple 'button'.[3]

3.3.2.5 Recomposition

The user interface is reconstructed with the new user functions.

3.3.2.6 Example

Assume an interface provides the email commands shown in figure 5.

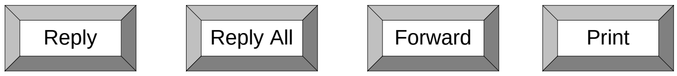

Figure 5 - sample email commands

Using the game metaphor, which is mapped as per table 1, these functions decompose as shown in figure 6.

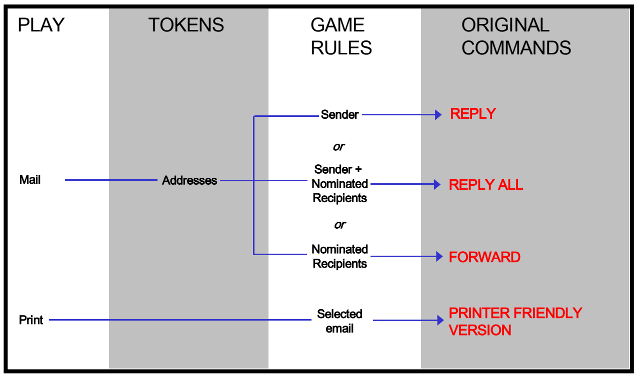

Figure 6 - sample decomposition mapping

According to our transformation rules, the Mail command should be represented as a menu, since a choice of Resources (Tokens) must be made. The Print command has no associated resources, so it should be represented as a simple button. Therefore the following set of functions are required on the recomposed interface are as shown in figure 7.

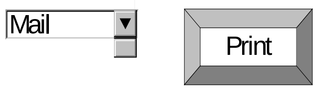

Figure 7 - sample recomposition - 1st level functions

[3] Assuming these are the concrete metaphors being used at the task level

If a Mail Action (Play) were made, then following Resources (Tokens) shown in figure 8 would be offered.

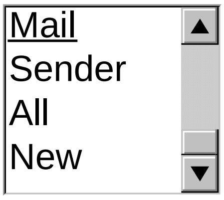

Figure 8 - sample recomposition - 2nd level functions

3.3.2.7 Results

This method has been applied to a number of popular interfaces, consistently resulting in significant interface improvements. This is achieved by reducing what is often a haphazard and arbitrary collection of functions into a concise, logical and consistent set of user functions. Indeed, in one case the number of initial screen elements was reduced from thirty-one to just seven without the loss of any functionality!

As the elastic metaphor can be applied to each part of a system (and potentially, every system) it is possible to provide the user with a level of interface consistency that the use of concrete metaphors can never provide. The result is an interface that is much more attuned to the way humans experience the world based upon a consistent and coherent metaphor that can be applied any interface.

3.4 Evaluating the Value of Elastic Metaphors

3.4.1 Source-Target Interaction

Using the concepts introduced by Anderson, Smyth et al (Anderson, Smyth et al., 1994) we can measure the effectiveness of the Source-Target mapping (Anderson, Smyth et al used the terms "Vehicle" and "Topic" equating to Source and Target). The size of the intersection of the two sets representing the Source and Target features indicates the effectiveness of the particular metaphor in use (figure 9).

We previously defined four source features upon which an elastic metaphor is based: Actors, Actions, Resources and Rules. Of these four features, one (Actors) was not directly used in the interface functional mapping (the actor, being the user, is *outside* of the interface). However, this feature is still part of the newly developed systems. Therefore, we can conclude that all the features supported by the source are provided by the target (T- S+ is zero).

In addition, all features provided by the target systems are supported by the source (we found no system functions that couldn't be mapped onto an elastic metaphor characteristic), so (T+ S- is zero).

While the set S- T- is always infinite (by definition), it can be seen that an elastic metaphor will always provided

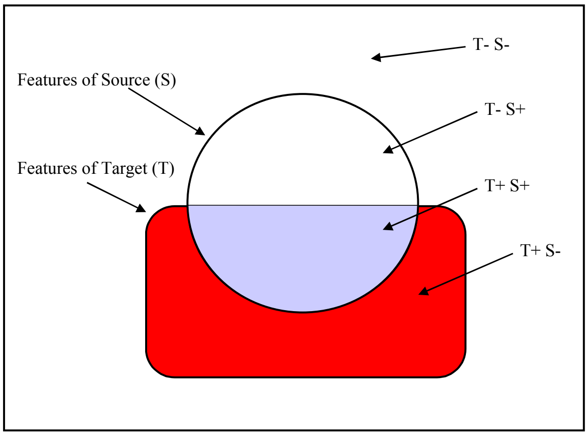

Figure 9 - Source-Target Interaction (based on Anderson, Smyth et al)

a smaller T- S- than a concrete metaphor by definition: i.e. elastic metaphors are inherently expandable and always have potential to grow larger than the system currently represented. In other words, T- S- is minimised.

This leads to the conclusion that elastic metaphors can provide an *optimal* mapping from source to target domains. This is particularly important in relation to emerging technologies such as mobile computing. The above-described techniques for developing interfaces using elastic metaphors produce a 'minimalist' interface. This systematic approach ensures that no functions are duplicated, and that a minimal number of functions are presented on each screen. This could be particularly valuable when designing interfaces that need to be represented on small output devices such as handheld computers or mobile phones.

3.5 Conclusion

Metaphor is an essential and fundamental part of language and thought. Indeed, "… it is the basic means by which abstract thought is made possible." (Lakoff and Nunez, 2000) It seems unlikely that metaphor can be eliminated from computer discourse.

However, the reliance upon concrete metaphor is particularly problematic, especially for wide domains. The "referents of computer metaphors are often ghostly abstractions, not things one can point to or see or touch." (Johnson, 1994) Yet when we anchor these abstractions in concrete metaphor we throw away the opportunity to rise above the world of gross physicality.

Elastic metaphors provide a vehicle for rising above these limitations.

4 References

Anderson, B., M. Smyth, et al. (1994). Minimising Conceptual Baggage: Making Choices about Metaphor. People and Computers IX, Proceedings of HCI '94. G. Cockton, S. W. Draper and G. R. S. Weir. Glasgow, Cambridge University Press**:** 179-194.

Blackwell, A. F. (1998). Metaphor in Diagrams. Darwin College. Cambridge, University of Cambridge**:** 188.

Dijkstra, E. W. (1985). "Fruits of Misunderstanding." Datamation **31**(4): 86-87.

Dijkstra, E. W., P. J. Denning, et al. (1989). "A debate on teaching computing science." Communications of the ACM **32**(12): 1397 (18).

Gardiner, M. M. and B. Christie, Eds. (1987). Applying Cognitive Psychology to User-Interface Design. Chichester, John Wiley & Sons.

Giddens, A. (1984). The Constitution of Society: Outline of the Theory of Structuration. Cambridge, Polity Press.

Hammond, N. V. and L. J. Allison (1987). The Travel Metaphor as Design Principle and Training Aid for Navigating Around Complex Systems. Proceedings of the Third Conference of the British Computer Society, University of Exeter, Cambridge University Press.

Indurkhya, B. (1994). "The Thesis That All Knowledge Is Metaphorical and Meanings of Metaphor." Metaphor and Symbolic Activity **9**(1): 61-63.

Johnson, G. J. (1994). "Of metaphor and the difficulty of computer discourse." Communications of the ACM **37**(12): 97-102.

L'Abbate, M. and M. Hemmje (1998). "Virgillio - The metaphor definition tool." GMD Report **15**: pp.47.

Lakoff, G. and M. Johnson (1980). Metaphors we live by. Chicago, University of Chicago Press.

Lakoff, G. and R. E. Nunez (2000). Where Mathematics Comes From. New York, Basic Books.

Marakas, G. M., R. D. Johnson, et al. (2000). "A theoretical model of differential social attributions toward computing technology: when the metaphor becomes the model." International Journal Human-Computer Studies **52**: 719-750.

Tristram, C. (2001). "The next computer interface." Technology Review **104**(10): 52-59.