doi:10.1006/imms.1993.1082
Cite or Link Using DOI

Int. J. Man–Machine Studies (1993) **39**, 753–773

The semiotic engineering of user interface languages

CLARISSE SIECKENIUS DE SOUZA

Departamento de Informática, PUC-Rio, Rua Marquês de São Vicente, 225, 22453-Rio de Janeiro, RJ–Brazil

(Received 7 February 1992 and accepted in revised form 20 May 1993)

Semiotic approaches to design have recently shown that systems are messages sent from designers so users. In this paper we examine the nature of such messages and show that systems are messages that can send and receive other messages—they are metacommunication artefacts that should be engineered according to explicit semiotic principles. User interface languages are the primary expressive resource for such complex communication environments. Existing cognitively-based research has provided results which set the target interface designers should hit, but little is said about how to make successful decisions during the process of design itself. In an attempt to give theoretical support to the elaboration of user interface languages, we explore Eco's Theory of Sign Production (U. Eco, *A Theory of Semiotics*, Bloomington, IN: Indiana University Press, 1976) and build a semiotic framework within which many design issues can be explained and predicted.

1. Introduction

The design of good user interface languages for application programs, as with other design activities, is essentially a matter of experience and talent. Despite the availability of sound conclusions from careful analytical research work, no theoretical background has yet been provided to fully support the synthesis of communication codes and protocols between humans and computer systems. The body of knowledge which orients choices and decisions in this field is usually a loosely structured set of DOs and DON'Ts, derived from the examination of successful and unsuccessful attempts at building good interfaces (Booth, 1989).

The whole process of human–computer interaction (HCI) rests upon assumptions made about the roles played by users and systems. Kammersgaard (1988) has proposed that there are four basic perspectives according to which HCI can be seen: the systems perspective, where users are taken as data-entry components of systems; the dialogue-partner perspective, where users and systems are taken as equivalent parties in a conversation; the tool perspective, where systems are viewed as instruments to be wielded by users; and finally the *media* perspective, where systems are viewed as a communication *medium* through which messages are passed between human parties.

By concentrating our discussions around the dialogue-partner and the *media* perspectives mentioned by Kammersgaard, we examine HCI from a communications point of view and show that systems have two inherent communicative roles: they are message senders and receivers at the immediate interface level, but they are also achieved messages, themselves, sent from designers to users through the computational *medium*. Being such a peculiar type of metacommunication artefact, systems must be engineered according to explicit semiotic principles. Such principles apply to user interface languages (UIL)—the expressive system in which all messages are coded.

Recent semiotic approaches to interface design have been inspired by Peirce's (1931–1958) analytical framework (see also Nadin, 1988) and by the Scandinavian school of glossematics (Anderson, 1990), for instance. Our approach draws its theoretical foundations from Eco's Theory of Sign Production (TSP) (Eco, 1976). Compared to more analytical theories, Eco's TSP grants us a synthesis-oriented perspective which is paramount to all design activities, and allows us to sketch the basis of a theoretic approach to user interface language design (UILD).

We propose a set of theoretically-motivated guidelines for UILD. They provide for a series of predictions and explanations we contrast with cognitively-based research results. We show that semiotic principles do not contradict cognitive ones; quite contrarily, both indicate the same direction to be followed. Insights and contributions, however, are of a different nature; whereas Cognitive Science presents designers with the appropriate targets to be hit, Semiotics presents valuable guidance for making successful shots.

User Centered System Design (UCSD) (Norman & Draper, 1986) is taken as an exemplary cognitive approach to design. In particular, Hutchins, Hollan and Norman's (1986) views on conversational and direct manipulation interfaces, as well as Norman's (1986) Cognitive Engineering approach, serve as a contrastive setting for the presentation of Semiotic Engineering. Along a different line, Suchman's arguments against strict plan-based HCI, and for the incorporation of larger portions of situational information in computer models, are discussed and reinterpreted in order to clarify relevant distinctions we want to make between her ethnomethodological approach and our semiotic one.

Section 2 presents an analysis of the communicative setting in HCI, showing that systems have more than one role to be accounted for by UILD. Section 3 presents an overview of Eco's TSP, highlighting its most relevant aspects for the production of interface signs—textual and non-textual. Section 4 presents predictions and explanations derived from semiotic theory, contrasting them with existing cognitively-based research results.

The conclusion of this work, in Section 5, is that Semiotics can contribute in a very positive way to a theoretic approach to design. Specifically, we propose that if designers are led to conceive of systems as a distinctive type of message they are sending to users—an engineered metacommunication artefact—many of the misunderstandings and deadlocks possibly occurring in human–computer interaction can be avoided. When users realize they are not interacting with autonomous machines, but with a rational product of a human mind, they can resort to a wealth of beliefs and expectations they have regarding the intellectual and creative behavior of other people. Unless systems designers are brought forth onto the scene of HCI, all of these crucial interpretive resources may be left untapped.

2. Senders, receivers and messages in HCI

An intuitive understanding of communication processes naturally involves the notion of messages being passed from sender(s) to receiver(s). Messages are meaningful perceptive portions of a code some agent (the receiver) recognizes and interprets as the expression of another agent's (the sender's) mental, physical or

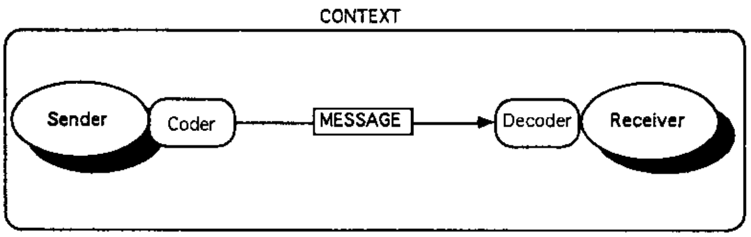

FIGURE 1. Basic elements of communication processes.

other state. A schematic representation of elements of a communicative act is shown in Figure 1. This may be used to characterize not only linguistic interchanges between humans, but also behavioral interactions between animals; broader interpretations may even cover communication between almost any kind of life forms (Sebeok, 1991).

Much of the discussion held in HCI about UILD points at the difficulty users find in (a) communicating intentions to systems and (b) understanding systems' behavior. The communicative setting may be conceived of as either one in which users are communicating with a computational agent (the dialogue-partner perspective), or one in which they are indirectly communicating with somebody else via a computational *medium* (the *media* perspective). These situations are represented in Figures 2(*a*) and 2(*b*), respectively.

In the *media* perspective, as Nadin puts it, "once the user accepts a language, he will apply it according to the rules the designer embedded in the interface, and their communication, mediated by a certain machine, will take place" (Nadin, 1988:

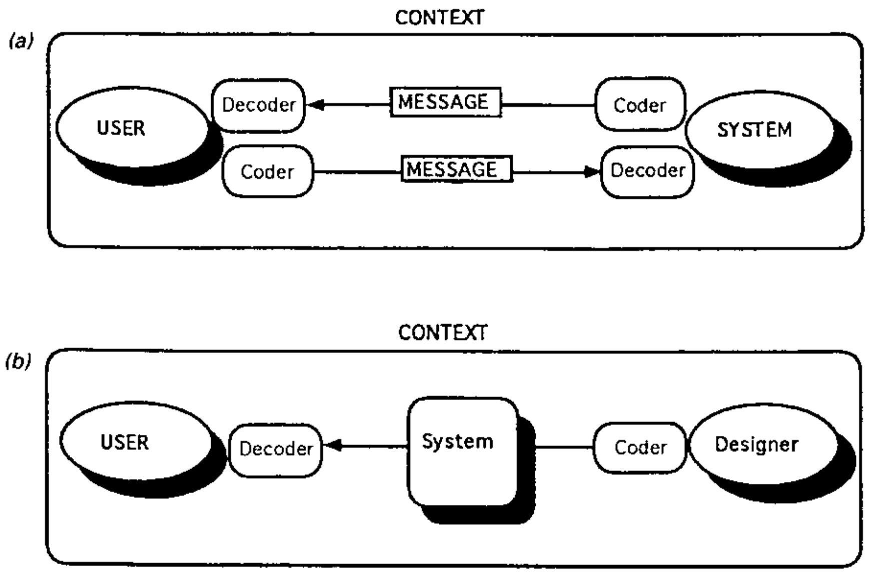

FIGURE 2. Possible communicative frameworks in HCI. (*a*) User in communication with system; (*b*) user in communication with system's designer.

p. 284). The mediation is such that the user cannot send back a response to the designer, except perhaps in test, maintenance or evaluation circumstances which are outside the scope of the actual interfacing process. This setting, depicted in Figure 2(*b*), is similar to that of author–reader communication, composer–listener, and the like: the two agents do not necessarily share the same space-time dimensions, and thus feedback is difficult or impossible.

The one-shot messages designers send to users through systems are of a peculiar type. Firstly, they are **performing messages** (unlike books, for instance); and secondly, what they perform is itself a communicative act, in which the message plays the role of sender and receiver of other messages. Consequently, systems can be rightfully taken as **metacommunication artefacts**. A parallel in art may be found with some situations in drama, where the audience is led to identify play and characters with real life (Laurel, 1986, 1991). Despite strong similarities with art, however, HCI diverges considerably from it inasmuch as computer messages are not typically intended to elicit a variety of emotions in users, but quite contrarily a reduced number of rational alternative meanings for each message.†

The actual communicative setting in HCI is thus one in which there are two levels of message-passing: (a) a one-shot message sent from designer to user (i.e. the system); and (b) an interchange between the user and the designer's performing message. Such setting is characterized in Figure 3. Designer–user communication is non-situated as opposed to the situated system–user communication.

In an extensive analysis of the role of plans and goals in HCI, Suchman (1987) underlines the fact that there is a deep asymmetry in such kinds of interaction, due

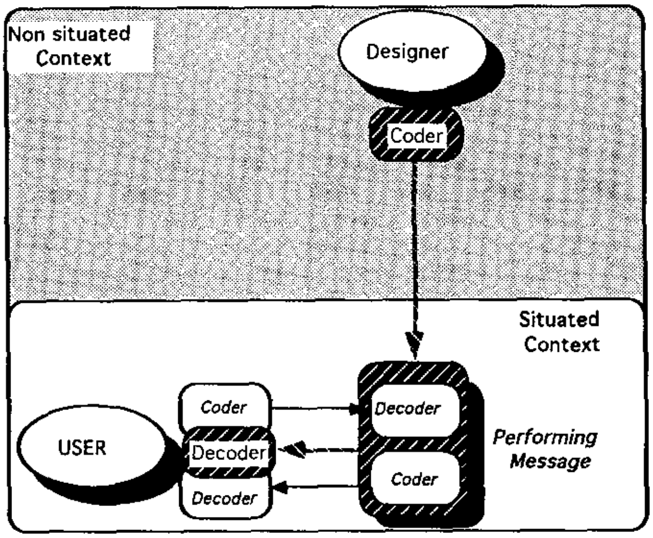

FIGURE 3. Two levels of communication in HCI.

† It should be emphasized that we ideally think of a reduced number of *meanings* for every message, but not a reduced number of *uses* or *purposes*. Users should feel they understand systems' behavior, no matter how numerous and varied the applications of such behavior can be. This seems to be the ultimate target of most approaches to interface design.

to fundamental differences in the level of access each agent has to situational aspects of the actual communicative setting. She criticizes the Cognitive Science approaches which take plan-based behavior as the basis for a theory of action, showing that ethnomethodological research provides a number of evidences that plans are very vague representational resources for mutual intelligibility. Her arguments suggest that research efforts aiming at specifying plans to a high level of detail and using them in interfaces to guide productive user–system communication may be frustrated. She concludes that:

> as long as machine actions are determined by stipulated conditions, machine interaction with the world, and with people in particular, will be limited to the intentions of designers and their ability to anticipate and constrain the user's actions. (Suchman, 1987: p. 189.)

Further examination of the issue reveals some other very important aspects, apparently left outside the scope of Suchman's discussion. Firstly, as stated by Adler and Winograd (1992), one of the crucial aspects of HCI is the continuous interpretation process in which users are engaged in their interaction with systems over time. Consequently, as new situations and needs emerge, creative usage may be the outcome of new interpretations. An example of this is the possibility of experienced text editor users wielding the software implicit information retrieval functions to make it play the role of a rudimentary information handling system. A list of people and their phone numbers may be easily processed by the **find** function of any text editing application. If this is the case, then it is unlikely that users "will be limited to the intentions of designers and their ability to anticipate and constrain the user's actions".

Secondly, plan-based behavior may be inadequate as the basis for a sound theoretical account of human action, but it is nevertheless perfectly adequate for an account of the metacommunication artefact's (i.e. the system's) action. This point is of major importance in the present discussion, since successful coding and decoding of messages at the user–system communication level certainly requires that users have an appropriate set of beliefs and expectations about the system's structure and functioning. Moreover, at the designer–user communication level, situational aspects of the nature Suchman refers to are necessarily abstracted away, due to the non-situated context of their interaction.

Because users have to interpret two-fold messages conveyed by systems' interfaces, and because successful interaction is fully dependent on correct interpretations, the design of message content *and* expression has to be optimally achieved. Semiotic theory plays an important part in supporting UILD at the *expression* level; the design of message *content* is substantially supported by cognitive psychology. Norman and Draper's (1986) User Centered System Design has been chosen as a complementary theoretic framework to our proposal due to its obvious congruence with the major points discussed in this paper.

In a user-centered approach to systems design, there are actually seven stages of user activities involved in the performance of a task by means of a computer system (see Figure 4). Such stages are found along the path that leads from users' psychological conceptions of the task to the system's actual performance of it. The central part of the interaction is the establishment of a goal. Once with a goal in

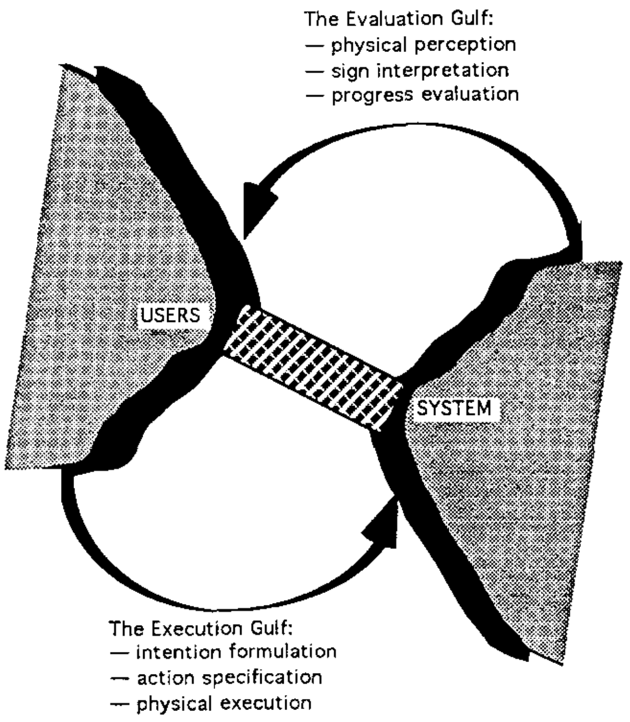

FIGURE 4. Norman's execution and evaluation gulfs.

mind, users must be able to "engage in a planning process" (Norman, 1986: p. 45)† in which they'll form an intention, specify the action sequence, and execute it. This amounts to bridging the EXECUTION GULF from the users' point of view (i.e. getting a system to work).

However, the Execution Gulf presents only half of the obstacles involved in human–computer interaction. The other half has to do with the system's reaction. Again from the users' point of view, this involves an evaluation of the system's behavior in terms of the original goal, which amounts to bridging the EVALUATION GULF. The stages of such a process are the physical perception of the system's output, an interpretation of the output's meaning and a final evaluation of how much of the initial goal has been furthered.

In User Centered System Design, interface languages must help users bridge the two gulfs with maximal ease. Hutchins, Hollan and Norman (1986) suggest that there is a metrics to support the analysis of the interface language. It should be noticed that their views are oriented towards analysis, rather than synthesis, given that distances can only be measured once certain expressive choices have already been made. In their own words,

> Every expression in the interface language has a meaning and a form. Semantic distance reflects the relationship between the user intentions and the meaning of the expressions in the interface languages both for input and for output. Articulatory distance reflects the relationship between the physical form of an expression in the interaction language and its meaning, again, both for input and for output. (Hutchins *et al.*, 1986: p. 100.)

† Confront this view with Suchman's (1987) critique.

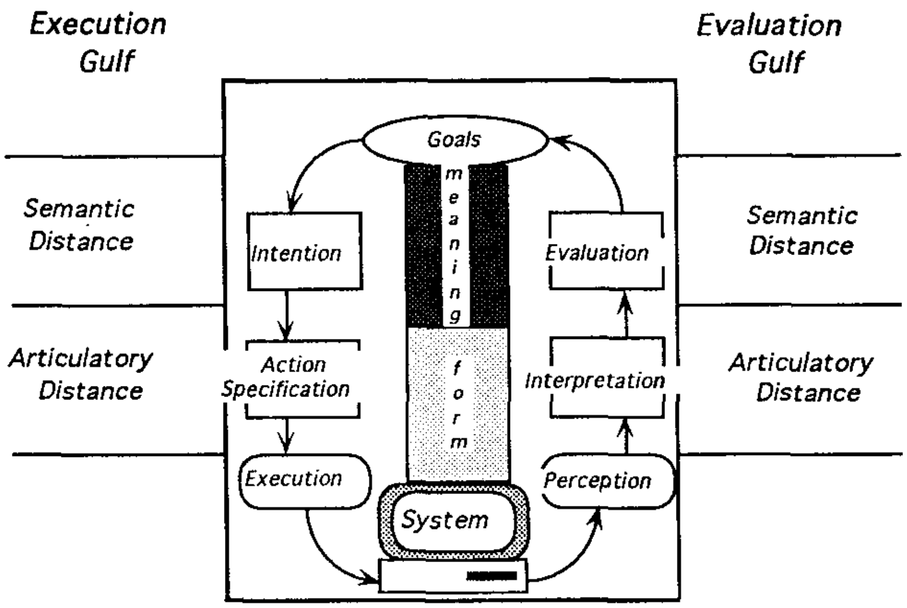

FIGURE 5. Semantic and articulatory distances in the execution and evaluation gulfs.

These authors show a correspondence between the Semantic and Articulatory Distance, on one side, and the steps involved in the Execution and the Evaluation Gulfs. Schematically, this correspondence is depicted in Figure 5, adapted from Hutchins *et al.* (1986: p. 111). Once intentions have been formulated, semantic distance spans from intentions to the meanings of expressions in the interface language. Such meanings are related to the set of functions the system can perform. Once the intended system's functions have been identified with the interface language meanings, the articulatory distance spans from there to the language-perceptible forms capable of triggering execution. In the inverse direction, once the user perceives the system's signalling, articulatory directness is related to how distant perceived forms are from the language meanings. Semantic directness, in its turn, is related to how distant such meanings are from the user's intended effects to further his or her original goals.

Hutchins *et al.'s* semantic and articulatory metrics emerge from their attempt to identify where the feeling of directness, present in the XEROX Star interface (Smith, Irby, Kimball, Verplank & Harslem, 1982), for example, comes from. By comparing direct manipulation interfaces to conversational ones, they conclude that directness is a matter of distance and engagement: it is achieved when distances are small and engagement is strong. Conversational interfaces tend to offer greater distances and weaker engagement than direct manipulation ones.

Brennan (1990) argues that the dichotomy between conversational and direct manipulation interfaces is actually a false one. She notes that *human language* supports direct manipulation in a highly effective way, by means of elaborate discourse structures. *Computer languages,* however, typically do not include such structures. In her view, the problem with the conversational metaphor is that computer systems "treat all user utterances as if they were *nonsequiturs*" (Brennan, 1990: p. 396). Her argument reinforces Draper's (1986) point for intereferential input and output in optimal HCI. According to him, the elements being manipulated by the user when expressing his or her needs and activating the system (bridging the Execution Gulf) have to be present in the system's output, when the user is trying to see what has been done with intentions expressed in the input (bridging the Evaluation Gulf). This is parallel to what takes place in intelligent conversations between humans: a "direct manipulation" of abstract (conceptual) structures by means of a linguistic system which extensively supports references to previously mentioned textual objects.

A conversational style in the interface reinforces the system's role as a dialogue partner. In semiotic terms, this type of interface is one in which the metacommunication artefact is a stylized replica, or even an impersonation, of a human interlocutor. The direct manipulation interface, in its turn, favors an interpretation of the metacommunication artefact as a physical object, whose properties resemble that of real objects in the world. Thus, semiotically speaking, the two types of interfaces seem to belong to two different expressive systems; one is that of what human beings can *mean* (in an active sense), the other that of what things can *mean* (in a passive sense).

So, what seems to be conflicting in this dichotomy is not a matter of what sorts of manipulations can or cannot be made in either style of interfaces, as Brennan puts it, but rather of how the message—i.e. the system—is to be interpreted by users in each case. This shifts the focus from what can be done with (or commanded to) systems to how the system itself is to be interpreted. It gears discussions towards the expressive power of interfaces, shifting the fulcrum of debate from Cognitive Engineering to Semiotic Engineering.

Semiotic Engineering is the process of engendering messages in an appropriate code—i.e. with emphasis on message expression—so that the content interpretation targets suggested by cognitively-oriented research can be met. The articulation between Semiotic and Cognitive Engineering is one in which the latter provides experimental data, explanations and predictions that are in accordance with the constitutive principles of the former. They operate at different stages of design, as shown in Figure 6. Cognitive Engineering helps designers engender the content elements of interaction, whereas Semiotic Engineering helps them engender the expressive elements of messages.

3. The production of iconic and non-iconic signs

Conversational and direct manipulation interfaces tend to privilege, respectively, two different kinds of UIL: textual and iconic. As their names say, textual UILs are based on words and sentences, whereas iconic UILs are based on pictures and graphics. In pure state, neither of the two types is intrinsically better than the other. The value of words in conveying complex abstract meanings and the value of pictures and graphics in being rapidly recognized and often more economically represented on loaded screens has been repeatedly invoked in technical and commercial literature. Since a combination of both types is frequently the best

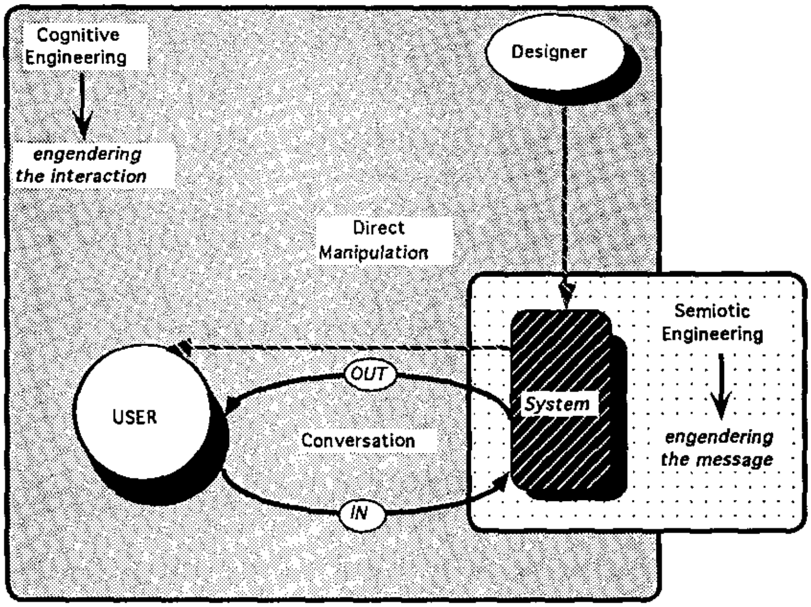

FIGURE 6. Cognitive and semiotic engineering.

option available for designers, research could aim at identifying theoretically motivated guidelines to support the style of combination between iconic and non-iconic signs.

Although researchers in graphic interfaces may have acknowledged the potential contribution of Semiotics for UILD, some confusion still remains as to what, exactly, this contribution is. Part of the problem may be due to a terminological confusion—even if tentatively dispelled—with the word **icon**. Marcus (1992), for example, borrows Peirce's basic taxonomical trichotomy for what he calls Visual Semiotics and defines sign types as follows:

An icon is something that looks like what it means... An index is a sign that was "caused" by the thing or process to which it refers... A symbol is a sign that may be completely arbitrary in appearance. (Marcus, 1992: p. 52.)

Marcus acknowledges that the meaning of the term icon in Computer Science is loose, and "strictly speaking, the so-called icons in iconic interfaces are a mixture of icons, indices and symbols." (Marcus, 1992: p. 52.)

Eco (1976) presents an extensive critique of iconism, showing by means of various examples that the most popular notions about icons as signs whose form resembles or approximates meaning, by way of some sort of perceptual attribute like shape, sound or other, is quite equivocal. He argues that a considerable portion of the problems with these notions may spring from an attempt to concentrate semiotic analysis on a typology of signs, such as that proposed by Peirce (1931–1958). Although the most popular of Peircian classifications is the icon-index-symbol one, there are dozens of others in his original proposal, many of which overlap, leading to a labyrinth of types and criteria. This multiplicity is due to Peirce's attempt to account for the process of communication, and not just for the analysis of signs after they have been selected and produced by interlocutors.

By drawing a parallel with Linguistics, we can see that analytical approaches typically fail to provide explanations and predictions regarding the conditions of actual generation of language(s). Structuralist schools such as that led by de Saussure (1916), for example, contributed considerably to describe natural language in terms of a system of opposing units that combine with each other in principled ways. However, generative approaches (Chomsky, 1965) have been needed to account for the underlying processes that make surface structures appear the way they do. The same applies to Semiotics, as Eco shows. A taxonomy of signs contributes to describing semiotic systems, but not to specifying the processes by which signs are used in communication.

Taxonomical approaches based on the icon-index-symbol distinction are by nature structure-oriented. They may support testing and evaluation of designed codes, but they fail to support the design process of metacommunication artefacts, since they do not explain or predict what happens when signs must be put together to create messages. Eco's TSP, on the other hand, with its explicit account of sign production processes, makes a number of potential contributions to design (Anderson, 1990).

Significant communication occurs whenever a human being (i.e. an agent capable of interpreting a code) perceives a message sent by a transmitter (Eco, 1976). The fundamental requirement for communication, in this framework, is that the message be coded in a signification system. Communication can only take place if the interpreter recognizes the sign(s) being passed as part of a culturally established code. The passage below reveals the centrality of interpretation in significant symbolic exchanges, which is in full accordance with UCSD cognitive framework.

> So, let us define a communicative process as the passage of a signal (not necessarily a sign) from a source (through a transmitter, along a channel) to a destination... When the destination is a human being, or 'addressee' (it is not necessary that the source or the transmitter be human, provided that they emit the signal following a system of rules known by the human addressee), we are on the contrary witnessing a process of signification—provided that the signal is not merely a stimulus but arouses an interpretive response in the addressee. This process is made possible by the existence of a code. (Eco, 1976: pp. 8–9.)

As Eco illustrates in his work, if the drawing of a dog is placed on a fence, the sign is taken to mean "*beware of the dog*". Such meaning is derived because cultural experience has taught us the fact that people can have dogs trained to protect their houses. The drawing (or icon, in Peircian terms) should not be taken to mean that the owner of the house has a kennel or is willing to have a dog. The same applies to indices. Symptoms, for example, are inferentially associated with diseases because "this association is culturally recognized and systematically coded" (Eco, 1976: p. 17). When a physician associates red spots on the skin with measles, for instance, this sign is the result of cultural knowledge and practice—for the first physician who ever saw them when measles was still an unknown disease, they did not constitute a sign, but a physical signal.

It is not by coincidence that the icon of a dog is taken to mean what it means, instead of inspiring in each person a different motivated meaning, and that red spots on somebody's skin are taken to mean a disease, instead of a fortuitous coloring that can go unnoticed. These things are signs, and not just visual signals, because they have a culturally codified meaning. Note that if some individual suddenly wants to express the meaning "*beware of the dog*" by hanging a picture of a wounded (bitten) foot on his or her garden fence, other people are not likely to take it for what it means. The interpretability of signs is fundamentally dependent on the recognition that they are coded. This is why Eco proposes that Semiotic Theory is composed of two parts: a Theory of Codes, accounting for phenomena related to signification (structure-oriented), and a Theory of Sign Production, accounting for phenomena related to communication (process-oriented).

In order to produce signs, the semiotic agent has to carry out tasks involving physical and mental labor. There exists a continuum for expression, ranging from words to gestures, to drawings, to sounds, and so on, from which the agent chooses the ones to convey message contents. Some contents are better conveyed by words, for example, others are better conveyed by gestures. Some can be conveyed by verbal and non-verbal expressions, whereas others can only be conveyed by one or the other.

Eco identifies four parameters which provide for a classification of the modes of production of signs: the physical labor performed by the agent, the token/type ratio of the sign, the type of continuum selected for expression and the mode and complexity of articulation. It should be noticed that the expressive continuum essentially ranges from a structured codified end to another unstructured uncodified end.

The physical labor to produce a sign ranges from the mere recognition of something as the existing codified expression of a content to the invention of a new, previously uncodified, expression (for example, an artistic piece), whose codification will have to be figured out by a human interpreter.

The token/type ratio regards the expression system corresponding to a content: when there is a recorded expression type for a given content, the expression token (i.e. the expressive instance used by the agent) is a case of *ratio facilis*. If we pronounce the word /dog/ to mean a *dog*, for example, our utterance is a token of a type of expression codified for the meaning we want to convey. The *ratio difficilis* happens when there is no previously codified type of expression for the desired content (or when the type is the content, itself). An example of *ratio difficilis* could be the use of objects to stand for themselves.

The type of continuum to be shaped in expressing a message can be either homomaterial (i.e. be made of the same matter as the content) or heteromaterial (i.e. made of a different matter). An example of a homomaterial continuum is found in cases where we quote somebody else's writing in our own piece of writing. All other passages referring to the ideas we want to present are typically expressed in a heteromaterial continuum (i.e. the content is some mental construct and the expression is written language).

Finally, the mode and complexity of articulation refers to the expression system having precise combinatorial rules or being a text "whose possible compositional units have not yet been further analyzed" (Eco, 1976: p. 217). Eco himself mentions iconic systems as examples of the latter case. The compositional units of such systems do not offer clear distinctions and organizational rules, applicable to some sort of iconic discourse. This is why Eco considers icons as "*visual text*", complex meaningful units in themselves, not further analysable (Eco, 1976: p. 215).

4. TSP predictions and explanations

As Winograd and Flores put it, computer systems are actually a cascade of representation systems ranging from the "natural" human representation of the task performed in the social environment to the "machine" representation of data and procedures involved in solving task-related problems. A similar view has been proposed by Requicha (1980) in a paper whose goal is to present an organized view of the whole field of computer graphics. Essentially, what these authors say in different ways and occasions is that Computer Science has a major linguistic component, given that computer systems are implementations of a series of models of real life objects and events. Such models are represented by means of symbolic systems, which can be natural language, formal logic, mathematics or other, outside the machine, but are always embedded in an artificial language system inside the machine, so that implementations of models can be built. Consequently, an articulated symbolic system lies inevitably at the core of any digital computer application.

The production of signs is central to the design of computer systems. The issue is to identify semiotic principles that will make possible better design. We suggest, as an initial framework for Semiotic Engineering, a set of four guidelines emerging from Eco's parameters for modes of sign production. They all converge to the goal of getting the message across at the interface level, and apply both to textual (Scott & de Souza, 1990) and non-textual interaction (Norman, 1991).

Guideline 1:
User interface language designers should produce signs they recognize as existing codified expressions of the intended contents.

Guideline 2:
User interface language designers should try to select expressions that are recognized as a token of an established type of expression system which accounts for the intended contents (*ratio facilis*).

Guideline 3:
User interface signs referring to domain objects and to computer-modeled solutions for problems should be heteromaterial, whereas the representation of I/O elements should be homomaterial and subject to direct manipulation.

Guideline 4:
User interface language designers should always resort to expressions derived from a recognizably codified (rule-based) system.

Guideline 1 predicts that, given a set of contents to be conveyed, existing recognizable signs make better expressions than invented ones. An example of this is mentioned in Section 3: the picture of a dog expresses the idea of "*beware of the dog*" better than a bitten foot. Reported experimental results (Black & Moran, 1982) have shown that users achieve clearly better performance when menus present words like INSERT or DELETE than when menus present nonwords like GAC or MIK.

Guideline 2 predicts that, given an established type of expression for the content at hand, designers should select a token of such type (*ratio facilis*) and not signs whose shape is in direct correlation with content, by way of causality or semantic entailment alone. Examples of *ratio facilis* are words, road signs, and conventional symbols in general. This aspect is supported by Hemenway's (1982) experimental results in favor of familiar symbols and canonical views in iconic interfaces. Examples of *ratio difficilis* are footprints (standing for a human being), dark smoke (standing for pollution) and other indexial signs. Again, Hemenway (1982) provides us with experimental support: she reports experiments with the interface of a system named OKRA, in which a wilted flower icon expresses the idea of DELETE FILE, whereas a living flower stands for CREATE FLOWER. As expected, users had a lot of difficulty figuring out the meaning of the two flowers. Yet another example of *ratio difficilis* is that of contents for which there are no established types of expressions. In these situations, token-type distinctions are often absent. When the hand icon first appeared in direct manipulation interfaces, to mean GRAB, no convention was available and it was clearly a case of *ratio difficilis*.

Guideline 3 approaches the variability of objects (or referents) to be represented by expressions in user interface languages. It suggests it is not the case that all objects should undergo the same semiotic treatment as far as the expressive continuum to be shaped is concerned. Firstly, domain objects are necessarily heteromaterial, since the meaning users assign for them is selected from real world situations, and expression is necessarily picked up from the set of possibilities computer I/O devices can offer on screens, speakers, and the like. This point finds support in Maissel's (1990) experimental results showing that direct depictions of world objects are easily interpreted by subjects.

Secondly, problem-solving steps, whose meaning is actually found in the underlying computer model of the task solution, should again be heteromaterial. In homomaterial representations, the expression units are picked up from the same realm as their contents. So, in this case, homomaterial expression should be computer world entities, not real world entities: users would have to interpret programming language code spans, for instance. Heteromaterial representations, however, draw their expression units from a different realm than that of contents. Consequently, in this case, real world entities would be taken to stand for computer data and procedures. This is where analogies play a major role, as for example in the desktop and filing system metaphors encountered in many operating systems interfaces.

Thirdly, I/O device components are yet another kind of content, totally distinct from the two above. They have no meaning or referent except inside the physical machine users are interacting with. This is a case for homomaterial signs. Windows, for example, despite their names which refer to things we find in buildings, are computer entities of a peculiar type. Users can size them, hide them, move them, and so on. The homomaterial nature of the sign is at the basis of such rich

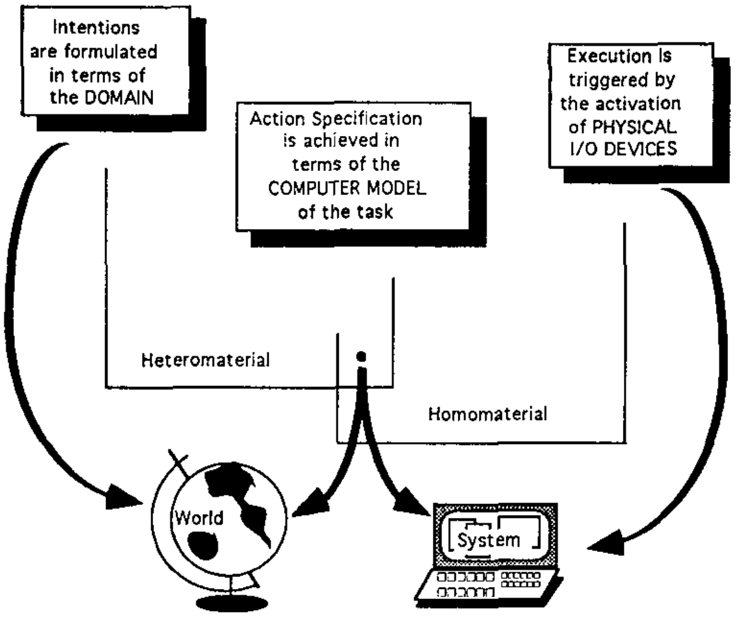
Intentions are formulated in terms of the DOMAIN
Action Specification is achieved in terms of the COMPUTER MODEL of the task
Execution Is triggered by the activation of PHYSICAL I/O DEVICES
Heteromaterial
Homomaterial
World
System

FIGURE 7. Homomaterial and heteromaterial and the execution gulf.

manipulation: outside world windows are not manipulated in this way, and unless such signs are understood as computer objects, they will be possibly mistaken or misused for what they are not. Figure 7 depicts the homomaterial vs. heteromaterial stances of signs in terms of Norman's (1986) steps for the traversal of the Execution Gulf.

Finally, Guideline 4 is intrinsically related to the fact that computation is symbolic manipulation. From the user interface level down to the hardware physical response, everything is layer upon layer of formal codes. Consequently, it is the very nature of computing systems that signs are articulated and rule-governed. Moreover, rule-based communication codes allow for generalizations made from perceived articulation between parts of the overall system. For example, if the UIL has a rule that the selection of objects precedes that of the operation they undergo, users can infer that this general principle also applies for objects and operations they select for the first time (not trying to select the new operation first, and then its object).

The above guidelines for Semiotic Engineering have two major underlying principles: UIL signs should be systematically coded and deeply embedded in the users' culture. If either of the two is overlooked in UILD, the interpretability of signs is severely challenged, given that metacommunication with systems is limited (if systems are taken as dialogue partners) and accessibility to designers is often impossible (if systems are taken as messages sent by designers).

Guidelines 1, 2 and 4 apply to all signs appearing at the interface. Guideline 3, however, applies differently for different kinds of signs. Thus, there is an optimal resource space for the selection or creation of all UIL expressions, whose dimensions are defined by the physical labor needed to produce the expression (from Guideline 1), the token/type ratio (from Guideline 2) and the level of articulation and regularity of the expressive system (from Guideline 4). Such space

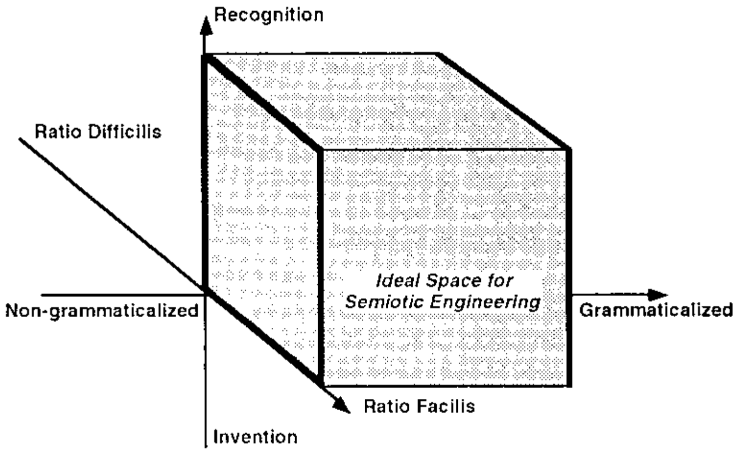

FIGURE 8. Space for the creation of signs in semiotic engineering.

is shown in Figure 8. Once inside this space, designers should select between homomaterial or heteromaterial signs according to what has been shown in Figure 7, above.

An examination of existing research shows that TSP can support semiotic explanation for a number of reported results. The adequacy of natural language text, for example, to express knowledge-based systems reasoning in explanations (Moore & Swartout, 1991), is a case of designers (a) recognizing that natural language is an available highly coded expressive system, (b) selecting a preferential token/type ratio for representing reasoning in general (*ratio facilis*), (c) resorting to heteromaterial domain-dependent objects for interface signs, and finally (d) using a greatly sophisticated type of articulation between content and form.

In contrast to natural language, if a pictorial expressive system were to be used to express artificial reasoning, designers would probably have to select icons representing objects, operations or both (Hemenway, 1982). Iconic expression would be difficult in this case because reasoning is itself a process in which other processes are involved as objects (abstract or concrete). This inevitably leads to at least one degree of articulation, where the recursive structure of processes within processes can be accounted for. Since icons are themselves primary units (i.e. they are not further analysable into other units), iconic expression is inadequate to represent the unfolding of reasoning processes. One exception might be raised for animated icons (Baecker, Small & Mander, 1991). In this particular type of expression, a possible representation of an explanation for a process could be its replication in an animated sequence of performing icons.

Some further consideration is called for, however. First, to explain a process by replicating its occurrence is a very special type of explanation, not necessarily effective to clarify users' doubts. Second, replication is usually a case of *ratio difficilis* in that objects stand for what they are, with no recourse to a meta-level of communication from where different perspectives can be invoked by means of different features brought about in a different expressive system. In order to clarify this point, it should suffice to compare two kinds of explanations for the fact that icons dragged into the trash in the Macintosh Desktop interface cannot be recovered after the trash is emptied. One is a short text saying that the memory addresses for *accessing* the objects referred to by the icons are deleted, and thus the objects cannot be found again. The other is a short cartoon showing files, folders and floppy disk icons being dragged into the trash. The trash icon is activated and a window is opened on the screen, showing all the trashed items. They are taken out, opened, closed, and put back into the trash. After that, the **special** option in the menu bar is activated, **empty trash** is selected, and again the trash is opened. This time, however, there is nothing inside it.

By comparing the two sketches of explanations above, it is clear that the textual one is more concise and precise, whereas the animated one is longer and metaphorical. If the meaning of **empty trash** is the issue, it seems that the icon animation reinforcing the analogy between file management and desktop operation accounts for the *effects* of the process, but not for its *causes.* Hemenway (1982) observed that the pictorial representation of processes by means of features that change as objects undergo the operation is more effective than others. As we have said before, signs of this type are close to what Peirce (1931–1958) calls indices, which can only be *ratio facilis* if there is a culturally coded association between signal and meaning (otherwise, they are *ratio difficilis*).

This example leads us naturally to an examination of analogies and metaphors appearing in UILs. Halasz and Moran have shown that analogies, though claimed to be "the best way to introduce a new user to a computer system", can bring about harmful consequences "preventing new users from developing an effective understanding of systems" (Halasz & Moran, 1982). If Macintosh Desktop users internalize the analogy too literally, they will be at a loss with applications whose function is to **undelete** files, for example. There is no reference in the interface to any garbage dump space where users can go in search of unduly trashed objects. However, correct understanding of the system's functioning should accommodate the idea of some memory addresses being recovered with the data kept in them, provided that no writing operation has been performed in that area. A deletion is a writing operation, and when files are deleted what is overwritten is only a small portion of the memory the whole file occupies. So, much of the data is left untouched and good for further recovery.

Previous experimental research by Svendsen (1991) has shown that interface style may influence problem-solving strategies adopted by users. Given two different modes of learning—insight learning, characterized by "a sudden leap from error prone to error free performance", and trial and error learning, characterized by "a gradual reduction in the number of errors made" (Svendson, 1991: p. 379)—users of command driven interfaces present clearly differing learning profiles from users of direct manipulation interfaces. The former tend to be able to report what they have learned, contrary to what is observable in the latter. However, command-driven interface users make much more effort than those of direct manipulation interfaces. The author concludes that, since users typically do not like to make conscious efforts in dealing with computers, direct manipulation interfaces might be taken to be more user-friendly than command-driven ones. Nevertheless, he points out, because of the lower degree of mental elaboration associated with trial and error learning, direct manipulation interfaces "are so supportive of thoughtless action that the user neglects to look for rules where these are called for" (Svendson, 1991: p. 395). The result is an apparent tension between user friendliness and usefulness of such interfaces.

This point raises an interesting issue in semiotic terms. What is the message users have to understand? In the above example, if the message is "*things can be recovered from the trash provided that it has not been emptied*", direct manipulation, with its iconic style of expression, may be sufficient for an adequate user conceptualization about the system. Contrarily, if deeper understanding of a message like "*items stored in memory can be partially recovered after deletion*" is desired, command-driven interfaces where users can type or select an **undelete** instruction suggest a more resourceful model of the system's functioning. In particular, the level of semantic and morphological articulation in the delete vs. undelete pair of lexical items in the command language provides more insight about the system model than the desktop analogy alone.

Iconic interfaces present enormous difficulties in terms of Guideline 4. Since icons are unanalysable stand-alone units, it is unlikely that a grammar can be written for something that would approximate *iconic language*. Since the fundamental role of grammars is to allow for generalizations and predictions about new sentences in the interface language, a substitute has to be found for users to predict how they can accomplish new tasks if only icons are visible. The way around this barrier is analogy. As shown in the desktop example, the substitute for a grammar in the iconic system is the metaphor (the expressive equivalent of analogy) along which deletion is related to trashing and recovery is related towards taking something out of the trash before it's emptied.

In their work, Halasz and Moran (1982) warn designers against the danger of analogies being used to guide reasoning instead of being used just as a means of expression—a literary metaphor. Lakoff and Johnson (1980) show that metaphors are mechanisms to structure concepts, but they also show that one concept may be structured by more than one metaphor. Thus, direct manipulation interfaces implementing only one metaphor may obstruct the path to a complete understanding of computer functioning.

TSP-based UILD assumes that designers clearly know what is the content of their message to users. The discussion about the look and feel of interfaces should not blur the reason why interfaces are needed in the first place. In the words of Adler and Winograd, "the key criterion of a system's usability is the extent to which it supports the potential for people who work with it to understand it, to learn it and to change it" (Adler & Winograd, 1992: p. 7). Interfaces cannot stand in the way of usability's ultimate criterion. If users are to change systems, they must have a satisfactory understanding of what systems do, and for this end message content and expression should be optimally engineered. As we have remarked earlier, message content is more cognitively engineered, whereas message expression is more semiotically engineered.

Svendsen's (1991) results suggest that direct manipulation interfaces, with preferential iconic expressive systems, are well suited for tasks that do not require much intellectual effort. Such tasks do exist, and should not be forgotten. Moreover, as pointed out by Lakoff and Johnson (1980), metaphors are powerful mechanisms for selective hiding and highlighting of features. So, for example, the fact that a character on a screen is actually an iconic representation for a certain pattern of bits at the machine level may well be hidden from users who are left free to conceive of the sign **1** as the digital *1* itself, and not as a sequence of zeros and ones or as an ASCII code number. Is users' intellectual understanding of bit strings or ASCII codes needed or desired? The answer is that application programmers (i.e. programming language users) should know this, whereas secretarial staff in an office typically shouldn't.

Focus is another crucial expressive resource in any message building activity. If the code used is natural language, focus is a highly structured and articulated process embedded in grammar and discourse. If the code is pictorial, then focus must be signaled by other means. Norman (1991) reports a number of psychological experiments with iconic interfaces, showing that saliency of distinctive features (focal information) is of capital importance in effective pictorial communication. In particular, Arend, Muthig & Wardmacher (1987) have shown that icons depicting global, more general, features of the content they express are more effective than those depicting local, more detailed features.

The direction indicated by TSP is that for long complex messages, where concepts follow a certain thread of events, the selected expressive mode should provide for focal manipulation. Baecker *et al.* (1991) report successful experiments with animated icons to portray complex processes evolving over time. An interesting comparison is found in Hypertext applications, where focal structure is typically difficult to implement (Gygi, 1990). The expressive system of visual buttons and anchors to be selected and visited, *per se,* lacks articulation at discourse level and has to resort to superimposed structuring systems to account for focal manipulation. Thus, when focus is an important part of the message, the expressive system has to account for it.

Finally, we would like to address the homomaterial aspect of some computer signs by means of two examples: text editing and 3D object manipulation. Text editors are interesting examples of expressive systems. If we take direct manipulation interfaces to such applications, we can see that words (as well as sentences, paragraphs, or characters) have something of an iconic flavor. They are not words in the linguistic or grammatical sense, but rather visual entities that can be shaped and moved in a variety of ways. This is only possible because they are computer entities. When contrasted with handwritten words, for instance, the operations they can undergo are of a totally different nature. They have a plastic characteristic which must be grasped to ensure full usability of the application by its users. The interface language in this case is composed of homomaterial signs which, if interpreted in the heteromaterial sense, do not warrant full usability of the system. This prediction is in accordance with Halasz and Moran's (1982) objections to analogical reasoning at the interface level.

3D manipulators, in their turn, present very distinctive problems, since 3D objects are represented in 2D. Experiments reported by Houde (1992) compare different strategies for easy 3D direct manipulation. Given a visual model of a furnished room, subjects have been asked to rearrange the objects in space by lifting, turning and sliding them. The first prototyped interface offered users hand-shaped cursors, each hand (or pair of hands) being represented in positions that suggested one of the three spatial operations. Interestingly, most users complained about the fact that the hand-shaped cursors "did not behave like a real hand" (Houde, 1992: p. 137). In other words, the icons were not realistic enough. Ongoing iterative design eventually portrayed a bounding box around objects ["a chair enclosed by a glass box" (Houde, 1992: p. 140)], with narrative handles (i.e. handles + hands making a particular gesture). This proved to be the best design option for intuitive 3D direct manipulation. As in the preceding text editor example, heteromaterial signs for this application presented problems (the unrealistic hand). Homomaterial signs, interpreted as computer entities (not real glass boxes, for example), warranted much better results. An additional support for homomateriality in Houde's research is provided by a special feature of the interface: the glass box was only used for lifting and turning operations. Sliding narrative handles were ignored by users, who performed sliding tasks as in 2D (click and drag). The author supposes this was due to users' previous experience with 2D manipulations in computer environments.

5. Concluding remarks

Existing theoretical research in human–computer interaction has not yet provided a coherent body of knowledge to support the design of user interface languages. Cognitively-based studies offer designers a wealth of reported experiments and evaluation data, pointing at a series of guidelines derived from the analysis of successful and unsuccessful interfaces.

Typically, cognitively-based research centers on theories of human (user) comprehension and purposeful action, and contributions to design lie in setting the right targets to be hit. Nevertheless, such investigations do not offer a theoretical framework to support designing. In other words, cognitive research tells designers what to do, but not exactly how to do it.

This paper is an attempt to contribute to the process of user interface language design by resorting to semiotic theory. As a complement to Norman's Cognitive Engineering, we propose a Semiotic Engineering process, in which systems are messages designers send to users. Such messages are constructed according to semiotic principles which should ensure that the target of Cognitive Engineering is met—i.e. that users understand systems and can use them productively.

Four UILD guidelines have been proposed as a basis for Semiotic Engineering. They emerge from Eco's four parameters involved in the process of sign production. The predictions entailed by our guidelines, as well as the explanations they provide, are in tune with a variety of reported results from cognitively-based research. The most relevant point in our study is not the guidelines, themselves, but the promising fact that semiotic theory has supported a theoretic approach to computer language design and may eventually support a theory of design itself, since interfacing is a crucial step in all engineering processes. As Simon says, "the artificial world is centered precisely on this interface between the inner and the outer environments: it is concerned with attaining goals by adapting the former to the latter" (Simon, 1981: p. 132).

Our conclusion is that HCI will possibly be more successful if designers are brought forth as the actual senders of a highly peculiar type of message: a message which sends and receives messages (a metacommunication artefact). For designers, this perspective should provide greater awareness of message contents and sharper understanding of the expressive power of semiotic systems available for computer processing. For users, this perspective should provide a wealthier and more accurate set of beliefs and expectations about systems' behavior: they are correctly taken as products of a human mind, and not as autonomous performing agents whose logic is totally impenetrable.

The next necessary step in the long research path ahead of us is to design experiments to test the theoretical results proposed here. Though psychologically-oriented research studies have provided valuable data in support of our hypotheses, experimental studies carried out within a consistent semiotic framework are clearly called for at this point.

The author is supported by the Brazilian Secretariat for Science and Technology (SCT). Research for this paper was done with valuable support from CNPq (Brazilian Council for the Development of Science and Technology) and CSLI (Center for the Study of Language and Information), Stanford University.

This author is grateful to her colleagues in the *ARETÊ Project* team and to Graça Nunes, for their constant incentive and support. A first version of this paper was written at CSLI (Stanford University), in a most challenging intellectual environment for research. Much is due to Terry Winograd's insightful suggestions about what points were relevant for design and to Bill Verplank's indication of an extensive bibliography in the field of graphic interfaces. Many thanks go to the valuable comments of two anonymous reviewers, whose ideas have helped me much in the writing of the present version.

References

ADLER, P. S. & WINOGRAD, T. A. (1992). *Usability: Turning Technology into Tools.* New York: Oxford University Press.

ANDERSEN, P. B. (1990). *A Theory of Computer Semiotics.* Cambridge: Cambridge University Press.

AREND, U., MUTHIG, K. P. & WANDMACHER, J. (1987). Evidence for global feature superiority in menu selection by icons. *Behaviour and Information Technology,* **6,** 411–426.

BAECKER, R., SMALL, I. & MANDER, R. (1991). Bringing icons to life. *Proceedings of CHI'91,* ACM Press, pp. 1–6.

BLACK, J. B. & MORAN, T. P. (1982). Learning and remembering command names. In T. P. MORAN, Ed. *Eight Short Papers in User Psychology*, pp. 11–14. Palo Alto, CA: Xerox Corporation.

BOOTH, P. A. (1989). *Introduction to Human–Computer Interaction.* Hove, UK: Lawrence Erlbaum.

BRENNAN, S. (1990). Conversation as direct manipulation: an iconoclastic view. In B. K. LAUREL, Ed. *The Art of Human–Computer Interface Design.* Reading, MA: Addison-Wesley.

CHOMSKY, N. (1965). *Aspects of the Theory of Syntax.* The Hague: Mouton.

DRAPER, S. W. (1986). Display managers as the basis for user–machine communication. In D. A. NORMAN & S. W. DRAPER, Eds. *User Centered System Design,* pp. 339–352. Hillsdale, NJ: Lawrence Erlbaum.

ECO, U. (1976). *A Theory of Semiotics.* Bloomington, IN: Indiana University Press. (First Midland Book Edition, 1979.)

GYGI, K. (1990). Recognizing the symptoms of hypertext.... and what to do about it. In B. K. LAUREL, Ed. *The Art of Human–Computer Interface Design,* pp. 279–287. Reading, MA: Addison-Wesley.

HALASZ, F. & MORAN, T. P. (1982). Analogy considered harmful. In T. P. MORAN, Ed. *Eight Short Papers in User Psychology,* pp. 33–36. Palo Alto, CA: Xerox Corporation.

HEMENWAY, K. (1982). Psychological issues in the use of icons in command menus. In T. P. MORAN, Ed. *Eight Short Papers in User Psychology,* pp. 21–24. Palo Alto, CA: Xerox Corporation.

HOUDE, S. (1992). Iterative design of an interface for easy 3-D direct manipulation. *Proceedings of CHI'92,* ACM Press, pp. 135–142.

HUTCHINS, E. I., HOLLAN, J. D. & NORMAN, D. A. (1986). Direct manipulation interfaces. In D. A. NORMAN & S. W. DRAPER, Eds. *User Centered System Design,* pp. 87–124. Hillsdale, NJ: Lawrence Erlbaum.

KAMMERSGAARD, J. (1988). Four different perspectives on human–computer interaction. *International Journal of Man–Machine Studies,* **28,** 343–362.

LAKOFF, G. & JOHNSON, M. (1980). *Metaphors We Live By.* Chicago, IL: University of Chicago Press.

LAUREL, B. K. (1986). Interface as mimesis. In D. A. NORMAN & S. W. DRAPER, Eds. *User Centered System Design,* pp. 76–85. Hillsdale, NJ: Lawrence Erlbaum.

LAUREL, B. K. (1990). *The Art of Human–Computer Interface Design.* Reading, M.A: Addison-Wesley.

LAUREL, B. K. (1991). *Computers as Theater.* Reading, MA: Addison-Wesley.

MAISSEL, J. (1990). *Development of a methodology for icon evaluation.* NPL Report DITC 159/90. National Physical Laboratory, Teddington, Middlesex, UK.

MARCUS, A. (1992). *Graphic Design for Electronic Documents and User Interfaces.* New York: ACM Press.

NADIN, M. (1988). Interface design: a semiotic paradigm. *Semiotica 69*, **3/4,** 269–302.

NORMAN, D. A. (191986). Cognitive engineering. In D. A. NORMAN & S. W. DRAPER, Eds. *User Centered System Design,* pp. 31–61. Hillsdale, NJ: Lawrence Erlbaum.

NORMAN, D. A. & DRAPER, S. W. (1986). *User Centered System Design.* Hillsdale, NJ: Lawrence Erlbaum.

NORMAN, K. L. (1991). *The Psychology of Menu Selection: Designing Cognitive Control of the Human/Computer Interface.* Norwood, NJ: Ablex.

MOORE, J. D. & SWARTOUT, W. R. (1991). A reactive approach to explanation: taking the user's feedback into account. In C. L. PARIS, W. R. SWARTOUT & W. C. MANN, Eds. *Natural Language Generation in Artificial Intelligence and Computational Linguistics,* pp. 3–48. Boston: Kluwer Academic Publishers.

PEIRCE, C. S. (1931–1958). *Collected Papers.* Cambridge, MA: Harvard University Press.

REQUICHA, A. A. G. (1980). Representations for rigid solids: theory, methods and systems. *Computing Surveys,* **12,** 437–464.

SAUSSURE, F. DE (1916). *Cours de Linguistique Générale.* Paris: Payot.

SCOTT, D. R. & DE SOUZA C. S. (1990). Getting the message across in RST-based text generation. In R. DALE, C. MELLISH & M. ZOCK, Eds. *Current Research in Natural Language Generation.* London: Academic Press.

SEBEOK, T. (1991). *A Sign is just a Sign.* Bloomington, IN: Indiana University Press.

SIMON, H. (1981). *The Sciences of the Artificial.* Cambridge, MA: MIT Press. (2nd ed.)

SMITH, D.C., IRBY, C., KIMBALL, R., VERPLANK, W. & HARSLEM, E. (1982). Designing the Star user interface. *Byte,* **7,** 242–282.

SUCHMAN, L. A. (1987). *Plans and Situated Actions.* Cambridge: Cambridge University Press.

SVENDSEN, G. B. (1991). The influence of interface style on problem solving. *International Journal of Man–Machine Studies,* **35,** 379–397.

WINOGRAD, T. & FLORES, F. (1986). *Understanding Computers and Cognition: New Foundations for Design.* Norwood, NJ: Ablex.

Paper accepted for publication by Associate Editor Professor. H. Thimbleby.