Semiotic Proposals for Software Design: Problems and Prospects

Osvaldo Luiz de Oliveira e
M. Cecília Calani Baranauskas

Relatório Técnico IC-98-10

Abril de 1998

Semiotic Proposals for Software Design: Problems and Prospects

Osvaldo Luiz de Oliveira e M. Cecília Calani Baranauskas

Abstract

Recently, the wide use of computers raised the need for better interface design. As a complement to the traditional cognitive approaches to interface design some proposals based on Semiotics are being addressed in the HCI literature. In such approaches the interface is understood as a collection of messages sent by the designer to the user. With a focus on the relationship designer-message-user these approaches do not make explicit the dialogue between the user, the interface entities and among the entities themselves. This paper discusses the main current proposals for applying Semiotics to software design, their underlying drawbacks, and proposes using of Semiotics to make explicit the design of the communication among the entities that inhabit the interface.

Keywords: interface, design, semiotics, user.

1 – INTRODUCTION

When the first computers appeared on the commercial market in the fifties, they were extremely difficult to use. Many reasons exists for this as quoted by Preece, J. et al.(1994, p. 4):

- They were huge and very expensive machines. Therefore, destined for a small number of users;
- They were only used by experts, scientists and engineers that knew the programming "secrets" through punch cards;
- Since these were for a small number of people, there was no interest to make them easy to use.

Nowadays, these conditions are not worth anymore: the cost of computers has been decreasing more and more. Ordinary users are working day by day with them and, as a consequence, we must to understand how to adapt them for people and their needs. It took forty years, to change computers from enormous machines housed in large air-conditioned rooms to small personal machines in offices or even hand-carried by the user.

The development of the first personal computers in the seventies was a great mark, because these machines provided capacity of individual users interactive processing at a low cost. Thus, people from different areas (business, agriculture, education, manufacture, entertainment etc.) began using computerized systems. Now, the diffusion of the computers has been such that, young or senior, expert or novice, people are using computers, direct or indirectly, in one way or another. Globalisation, the convergence between computers and telecommunications has been linking people throughout the entire world.

In order to provide na efficient use of the environment, his tools and his machines, its design needs to be adequate. Our everyday objects (steering wheel of an automobile, handle of a door, faucet, glass, traffic light) have been specifically designed having in mind their use. The handle of a door has to fit the hand that will manipulate it, and at the same time, it should be resistant enough to be pulled (Laurel and Mountford, 1990, p. xii).

One must ask what is design? Kapor (1996, p. 4) comments that architects are the responsible for studying and proposing solutions for the space allocated to humans. Architects are interested in proposing the design of a house in a such way that, for example, bedrooms are located in a silent place, dining rooms are close to the kitchen, etc.. Automobiles have a design that is function, among other things, of the purpose for which they are built. In the same way, the effective use of computers demands an adequate design.

Software design is a new area that studies the intersection between people and computers together with the several interfaces (physics, sensorial, psychological) connecting them (Winograd, 1996, p. xv). Software design is an area involving several disciplines such as: Software Engineering, Hardware Engineering, Psychology, Artificial Intelligence, Linguistics, Anthropology, Sociology, Philosophy, Ergonomics and Human Factors (Draper and Norman, 1986, p. 1; Preece et al., 1994, p. 38). When we say that the design of a software works, we mean that it works in a context of values and people's needs. To practice design it is necessary to create structures supporting the Human-computer interaction. Different from a software planner, a software designer should surpass the barrier of work stations and conceive structures adapted for the user's thought (Winograd, 1995, p. 68).

We could say that, in the beginning, the computer was considered a tool to facilitate human tasks with reliability and speed. The development of cognitive theories on the Human-computer interaction brought us a vision of the computer as a cognitive tool to enable people enlarging their understanding capacity, memorization and decision making. Considering the semiotics perspective, computers are seen as media (Andersen et al., 1993) in the same way as books, cinema, the theatre and television. The computers is a medium through which messages should be interpreted. We understand that the cognitive and semiotics points of view are not antagonistic but, taken under different angles, represent different pictures of the same piece.

The traditional cognitive approaches focus the human interacting with the interface, his motor system, his perception, learning and other mental processes. The semiotics approaches facilitate an interpersonal, social, cultural perspective, focusing the expression and interpretation of the elements in the software interface.

This paper aims to discuss the problems presented by the main semiotic approaches to software design. It is organised as follows: section 2 presents the contribution brought by semiotics to the software design, in relation to the cognitive tradition that has marked the works in this field. Sections 3 and 4 discuss the main semiotics approaches highlighting the problems raised by these approaches. Section 5 signals a possible solution to problems set out in the section 4. Section 6 is the conclusion.

2. THE COGNITIVE TRADITION TO SOFTWARE DESIGN

The tradition that characterized the research in Human-computer interaction and, mainly, the software design, has been the cognitive. Cognition refers to the process by which we became conscious of things or, in another way, the way we acquire knowledge. This includes understanding, ability to memorize, reasoning, attention, learning, creation of new ideas, etc.. The main objective of the research in this field has been to understand and to represent the way human interacts with computers, in terms of the way knowledge is exchanged among the two.

The theoretical basis for this approach rests on Cognitive Psychology and explains how humans reach their objectives. Such goal-oriented activity comprizes performing cognitive tasks which involve information processing. Humans are characterized as information processors. Anything felt (seen, heard, smelled, tasted, touched) is considered an information processed by the mind. The basic idea is that the information enters and leaves the human mind through an orderly series of processing steps (Preece et al., 1994, p. 62).

The human model as processor of information has influenced the development of several models of Human-computer interaction as, for example, GOMS (Card, Moran and Newell, 1983) and the Theory of Action (Norman, 1986). The Human-computer interaction is seen as composing two processing units: the human and the computer. The output of a processing unit is the input of the other. In other words, the Human-computer interaction can be described as a loop as shown in the figure 1 (Kaptelinin, 1996, p. 104) with several advantages: first it provides a coherent description of the whole interaction system in the perspective of information processing. Second, aspects of the Human-computer interaction, as presentation of information to the user, the user's perception, his mental model of the task, the user's attention on the system and I/O devices can be easily located in this outline.

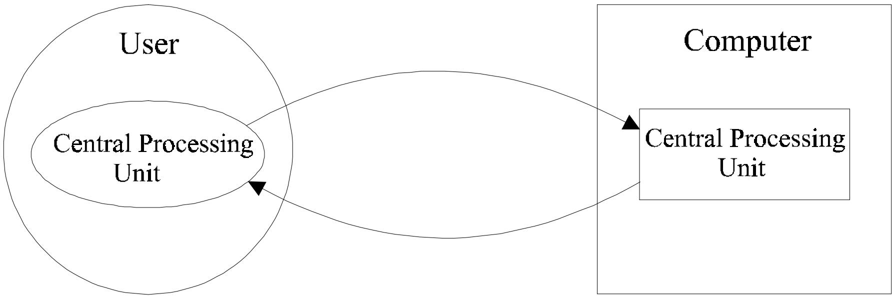

Fig. 1: Human-Computer interaction in the cognitive tradition

The cognitive approach can be applied with success to many problems of Human-computer interaction. However, there is an emergent consensus that this approach has its limitations. The information loop is closed, so it is difficult to take into consideration phenomena that are out of it (Kaptelinin, 1996, p. 105). For example the purely cognitive approach doesn't provide basis to take into consideration phenomena of linguistic nature and of interaction among people's groups. On the other hand, semiotic approaches to software design allows to consider not only the immediate aspects of the Human-computer interaction but, also, the underlying aspects of the cultural and social context where the interaction happens.

3. SEMIOTICS AND COMPUTATIONAL SEMIOTICS

The discipline of Semiotics has been known for approximately two thousand years (Echo, 1976, trad. port, p. xi). However, its development as it is actually understood had its start with the work of Charles Sanders Peirce (1839-1914), a North American philosopher, and with the work of the Swiss linguist Ferdinand de Saussure (1857-1915). Semiotics aims to study signs and signs systems. A sign is something that stands for another thing for somebody under certain aspects or capacities (Peirce, CP 2.228)[1]. Any mark, physical movement, symbol, token etc., used to indicate and convey thoughts,

[1] CP is a short name for "Collected Papers of Charles Sanders Peirce" and "2.228" is the paragraph 228 in the vol. 2.

information and commands constitute a sign (Sebeok, 1994, p. xi). A photo is a sign which, stands for the elements represented in it, for somebody who interprets it. And, if somebody‘s interpretation, the word "yellow" stands for the yellow colour, red spots in the face stands for the measles, the "horse" pronunciation stands for the animal horse, smoke stands for fire, the drawing of a printer in the computer screen stands for print, waving the hand in a certain way stands for good-bye, a certain perfume stands for a rose, then, yellow, red spots in the face, the horse pronunciation, smoke, the drawing of the printer, waving the hand and the perfume of a rose are all examples of signs. Observe that, without signs, our communication would be very poor, once we would be forced to communicate making use of the specific objects to which we want to refer.

For Peirce (CP, 2.274) the sign is a genuine triadic relationship among the elements: the representamen, the object and the interpretant (fig. 2). The representamen refers to the material aspect of the sign and represents the object under certain aspects or “capacities”. The sign only means so because the representamen can represent another thing: the object. The interpretant doesn't refer to the interpreter of the sign but it refers to a relational process occurring in the interpreter’s mind, associating representamen and object.

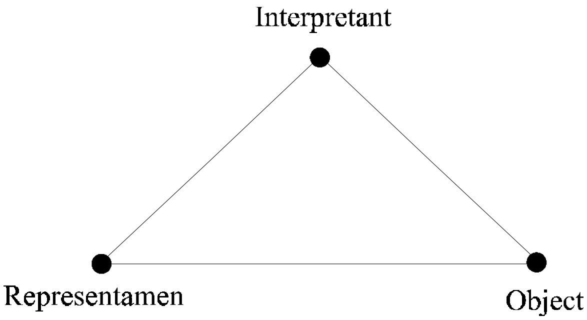

Fig. 2: The sign as a triadic relation

Santaella (1996, p. 7) observes that Semiotics proposes to view the world as a language. She doesn't only refer to the spoken or written verbal languages, but to all other types of language: the deaf-and-dumb language, the dance, the cookery, the fashion, the rituals of primitive tribes, the music, the sculptures, the scenography, the hieroglyph, the dreams, the wind etc.. Semiotics investigates of all possible languages as a phenomenon of producing significance and sense. Its field of interests is vast, going from the study of the communication in non human communities to the social study of ideologies; the study of moral, political, economic, religious and military codes. Today we know that, even our notion of life as something with certain characteristics and capable to reproduce, presupposes the existence of information coded inside the biological system: the DNA. It is language and it would not be absurd to say that without language there is no life. It is a semiotics subject any sign produced or interpreted by us, human, or by other animals, plants, protozoa, fungi and bacteria or by an engine developed by some entity like a robot, or supernatural entity such as when a boy kneels down and prays to divinity *I pray the Lord my soul to take* (Sebeok, 1994, p. 6). The notion of culture itself (and of society) needs to be reformulated when one tries to study the universe phenomena, from its very beginning to the present animals and plants, as manifestation of languages, i.e., under a semiotics sphere - the semiosphere (Hoffmeyer, J., 1996, trad. Ing., p. viii).

Computers can be understood as media just as the book, the theatre and the cinema. We are not referring only to communication-based applications like electronic mail, video-conference, or applications classified as Groupware and CSCW (Computer Supported Cooperative Work), for which the media role becomes evident. We mean any type of software: for example, a paintbrush of a drawing software, represented by a collection of pixels in the screen, is a sign for a paintbrush. Under certain capacities it stands for a paintbrush, it transmits the meaning of a paintbrush as the word "paintbrush" written in a book does.

Semiotics is an old science, at least when compared to Computer Science, and a lot has been written about its application in several fields of the human knowledge. When we associate to the computer the media role, we won the possibility to transfer to Computer Science a strong theoretical basis known from Semiotics for analysis and creation of another media: theatre (Guisburg and Coelho Netto, 1988), cinema (Metz, trad. ing., 1991), analysis of paintings (Schapiro, 1996), magazines, newspapers and television (Bignell, 1997). After all, computers are media in which signs can be produced and transmitted: machines which generate interpretation. This has motivated the development of a Computational Semiotics.

Computational Semiotics is a discipline that studies the nature and use of the signs based on the computer (Andersen, 1997, p. 1). The object of study of Computational Semiotics is not only directed to software design (that is our interest in this work). For example, system programming and modelling can also be understood using the semiotics point of view. Programming and modelling are not only engineering but also activities of creation of signs, comparable the writer's activities, the director’s activities (of theatre, cinema, television) or the painter, whose objective is to create meaning.

4. SEMIOTIC APPROACHES AND HOW THEY "INFORM" THE DESIGN OF SOFTWARE

Nadin (1988) introduces one of the first attempts of applying Semiotics to the interfaces design, based on Peirce’s theory. Andersen (1997) found in the European School, created by Saussure and developed by Hjelmslev, the theoretical substratum that allowed to propose a Computacional Semiotics. Andersen (1997) is a renewed edition of a vast work, first published on 1990, that applies Semiotics not only to the design of interfaces but also to software programming, analysis and project.

The interface is defined by Andersen (1997, p. 143) as a collection of computer-based signs, i. e., the software parts which can be seen or heard, used and interpreted by a community of users. The interface design should emerge of its patterns of uses, i. e., from the way the user makes use of the dialect based on the computer (Andersen 1997, p. 246). The design is seen as an interactive process in which proposals are continually developed, used and evaluated (Andersen, 1997, p. 171). It is presupposed that in each iteration of the design cycle, there is a body of signs to be analyzed (Andersen, 1997, p. 247). The structuralist Semiotics of Hjelmslev supports this design method because it is descriptive and analytic. Thus, the relationships among the units composing the language can be analysed and, based in this analysis, possible modifications can be proposed with the objective of adapting the design of the interface signs to the patterns of uses of the computer-based dialect.

Souza (1993) proposes a Semiotic Engineering for the design of user interface languages. In her approach the interface is a meta-communication artefact. The interface is composed of messages from the designer to the user in such way that each message itself can send and receive the user's messages. In this sense, the interface executes two roles: (1) to communicate the application functionality (what things the interface represents, which types of problems it is prepared to solve) and the interaction model (how a problem can be solved); (2) to facilitate the exchange of messages between user and application.

Souza uses Eco's theory of the sign production (1976, trad. port.) and, specially, its four parameters of sign production to define four basic guidelines to develop the signs of the user's interface. The design process involves: (1) an analysis of the user (profile, knowledge about the domain and about computational systems); (2) analysis of the tasks and of the application domain; (3) design of the interface messages in agreement with the basic guidelines derived from the theory of the sign production (Souza and Leite, 1997, p. 22). It can be said that, while Andersen's approach is descriptive and analytic, Souza's approach is generative, focusing on the sign production.

In another work with theoretical substratum in the Semiotic Engineering, Leite and Souza (1997) propose a method for the user's interface language design, based on the sign definition of Pierce and on the Eco's Theory of the Sign Production (1976, trad. port.). The main point which deserves attention in this design method is the understanding that the model of usability of the interface is learned by users when they interpret messages coded in a User Interface Design Language (UIDL) (Leite and Souza, 1997, p. 2).

An interface requests from the designer a process of messages production and, it requires from the users, the interpretation of these messages. Thus, the UIDL is a semiotics system which allows the designer to compose the messages of the user's interface for a certain application. For the user, the fact of having the interface messages coded in a well-known UIDL, brings the possibility of knowing what the application does and how to interact with it. In the proposed method the UIDL should be defined as a computable hypercode: a code that is computable and which supports the meta-communication function of the interface.

Drawbacks

In the semiotics approaches which understand the interface as a intermediary, the designer develops signs to compose the interface and the user interprets these signs. Under this view, besides the user-system interaction, everything happens as if the user's role was to match an indirect monologue of the designer. The interface is the "office boy" of the designer.

Semiotics Proposals which understand the interface strictly as messages sent from the designer to the users, transmitted through computational media, have its limitations. The communication is function of the time and of the person who interprets it: the communication cannot, in general, be described as a message of the designer to the users through a channel, because a same sign can induce different interpretations in different people at different times. The communication is dependent on time and state of the receiver.

Interfaces design processes which consider as one shot the transmission of messages from designer to the user, fail in not considering the exchange of information between the designer and the user. The growing prototyping culture (Schrage, 1996) and the awareness for the need of participatory design (Kuhn, 1996) in the organisations, assigns to the user a great influence in the design process. Therefore it is more realistic to consider, also, the messages transmitted by the user to the designer.

Good computer systems create in the user a feeling of first person when he is interacting with the computer (Laurel, 1986). The user projects herself in the interface and he finds herself dialoguing with the entities of the interface, creating the feeling which Hutchins et al. (1986) called *directness*. For the user, to feel dialoguing with the entities of the interface and to be capable to attend and to understand the established dialogue among the entities of the interface is fundamental condition to reach *directness*. However, the current semiotics approaches are not interested in explaining the relationship among the entities of the interface and, consequently, they don't accommodate the current phenomena of the interrelation among them.

5 – PROSPECTS

Winograd defines software not only as a device for users interaction with the computer; software is, also, the space in which the user lives. *When an architect designs a home or* *an office building, a structure is being specified. More significantly, though, the patterns of life for its inhabitants are being shaped. People are thought of as inhabitants rather than as users of buildings. ... we (should think) software users as inhabitants, focusing on how they live in the spaces that designers create* (Winograd, 1996, p. xvii). From the semiotics point of view, to "inhabit" means to be quickly understood by the software (or by the entities that compose the software interface) and, at the same time, to interpret easily what the software or any entity of the software interface "says".

Thus, the interface should be understood as being composed of entities with communicative capacity, one or more of the which human (users). About software design we agree with Andersen (1997) about the need for developing the interface design in agreement with the patterns found the dialect people use. However, this is not enough: we need, also, to analyse the "language" of each non human entity that participates in the interface. Starting from the knowledge of these languages we can design the software observing the communication among the entities. We thought that a design process like this one, in which there is a strong purpose of accommodating the underlying phenomena of the communication among the entities, leads to the *directness* feeling, and it configures the interface as a space to be inhabited by its users.

The semiotics theories motivate us to understand the interface as an environment inhabited by entities that communicate. *As we are naturally symbolic, our life in the world and our sensitive access to it is always hindered by a crust of signs which, although supplies us with a mean of understanding, transforming, and planning the world, at the same time it usurps us of a direct, immediate, sensual, tangible existence, with the sensitive* (Santaella, 1996, p. 52). But, if on one side we are condemned not to taking direct contact with the world, on the other hand we can feel it in different manners. When we see an orange, we interpreted what we see as an orange. When we look at the drawing of an orange, we also recognise an orange. In computer games it is very common to hear sentences like: "I skidded in that curve and I beat in the wall" or "I caught the sword and I killed the dragon". To be in the pilot's feet in a car racing of "F1 Grand Prix" game or in the place of the hero in the "Dark Castle" game is a demonstration that a layer of signs intermediate our relationship with the world. The wall, in the F1 Grand Prix game is not a wall but it stands for it; the dragon in Dark Castle is not an animal but stands for it. The moment in which we put ourselves in the pilot's place we start to live his life and live with the other pilots and their cars, racing tracks and curves of a F1 racing environment. The interface becomes an environment we users inhabit with other pilots and several entities.

What are the implications of this? Considering the way the interface is understood, it switches the interface understanding from "the thing in between", of many interface characterisations[2] (Laurel and Mountford, 1990, p. xii; Nadin, 1988, p. 272), and of "the thing to be read", of Souza (1993) and of Andersen (1997), to "the thing to be inhabited".

[2] Interface has been defined, for several researches, as "the contact surface of a thing".

With reference to the design process, we intend to focus and to give special importance to the semiotics aspects that permeates the communication among the entities. We intend to change the application of semiotics from the designer-user relationship as it is addressed by Andersen (1997), Souza (1993) and Leite and Souza (1997), to the interaction among the interface entities.

The fact of understanding the user (or users) as entities that inhabit the interface, our proposal would naturally include, the phenomena happening in the new interfaces for groupware, CSCW and virtual reality, without leaving outside phenomena of the traditional interfaces.

6. CONCLUSION

The traditional theories of interface design, based on Semiotics, focus on the designer-user relationship and fail in making explicit the communication among the interface entities. Focusing on the designer-user relationship these theories tend to consider the designer as a writer and the user as a reader, and they do not respect the user's experience of interaction with the world. Because of the lack of making explicit the communication among the interface entities, these theories have difficult in explaining: (1) the user's actions on the interface, which change the way of the history - the user is not a simple reader, but a character acting in the history played in the interface; (2) the dialogue among the interface entities which, as in a play, generates possibilities of interpretations.

We pointed out the need for a semiotics approach to software design that understands the user as an inhabitant of the interface and the interface as a group of communicating entities. The user interprets what is happening by observing the communication among the other entities that inhabit the interface. At the same time, the user interferes in the running history, by communicating with one or more of the other entities. Thus, we proposed to understand the interface as a function of the semiotics relationships among the entities that inhabit the interface, and as such we intend to apply the semiotic theories in the communication design among these entities. This approach moves the focus from the designer-user relationship to the relationship among such entities.

We are now working in a methodology of software design that applies Semiotics in the proposed way. In the near future we are planning to use this methodology in the design of tools for collaborative work.

7 - REFERENCES

Andersen, P. B. (1997). *A Theory of Computer Semiotics*. Updated ed. of 1990. New York: Cambridge University Press.

Andersen, P. B., Holmqvist, B., Jensen, J. F. (1993). *Computer as media*. New York: Cambridge University Press.

Bignell, J. (1997). *Media Semiotics – An Introduction*. Manchester: Manchester University Press.

Card, S. K., Moran, T. P., Newell, A. (1983). *The Psycology of Human-Computer Interaction*. Hillsdale: Lawrence Erlbaum Associates.

Draper, S. W., Norman, D. A. (1986). Introduction. In Norman, D. A., Draper, S. W. (eds), *User Centered System Design – New Perspectives on Human-Computer Interaction.* Hillsdale: Lawrence Erlbaum Associates, p. 1-6.

Eco, U. (1976). *Tratado Geral de Semiótica*. Trad. Antônio de Pádua Danesi e Gilson Cézar Cardoso de Souza (A Theory of Semiotics. Bloomington: Indiana University Press). São Paulo: Editora Perspectiva, 1980.

Guinsburg, J. e Coelho Netto (1988). *Semiologia do Teatro*. São Paulo: Editora Perspectiva.

Hoffmeyer, J. (1996). *Sings of Meaning in Universe.* Trad. Barbara J. Haveland (En Snegl På Vejen: Betydningens Naturhistorie, 1993). Bloomington: Indiana University Press.

Hutchins, E. L., Hollan, J. D., Norman, D. A. (1986). Direct Manipulation Interfaces. In Norman, D. A., Draper, S. W. (eds), *User Centered System Design in New Perspectives on Human-Computer Interaction*. Hillsdale: Lawrence Erlbaum Associates, p. 87-124.

Kapor, M. (1996). A Software Design Manifesto. In Winograd, T. (ed.), *Bringing Design to Software.* New York: Addison-Wesley Publishing Company, p. 1-9.

Kaptelinin, V. (1996). Activity Theory: Implications for Human-Computer Interaction. In Nardi, B. A. (ed.), *Context and Consciousness – Activity Theory and Human-Computer Interaction*. Cambridge: The MIT Press, p. 103-116.

Kuhn, S. (1996). Design for People at Work. In Winograd, T. (ed.), *Bringing Design to Software.* New York: Addison-Wesley Publishing Company, p. 273-289.

Laurel, B. K. (1986). Interface as Mimeis. In Norman, D. A., Draper, S. W. (eds), *User Centered System Design – New Perspectives on Human-Computer Interaction.* Hillsdale: Lawrence Erlbaum Associates, p. 67-86.

Laurel, B. K., Mountford, J. (1990). Introduction. In Laurel, B. K. (ed.), *The art of Human-Computer Interface Design.* Reading: Addison-Wesley Publishing Company, p. xi-xvi.

Leite, J. C., Souza, C. S. (1997). A framework for the Semiotic Engineering of User Interface Languages. In Lucena, C. J. P. (ed.), M*onografias em Ciência da Computação.* Rio de Janeiro: PUC Rio.

Metz, C. (1991). *Film Language – A Semiotics of the Cinema*. Trad. Michael Taylor (Essais Sur la Signification au Cinéma. Paris: Editions Klincksieck, 1968). Chicago: The University of Chicago Press.

Nadin, M. (1988). Interface Design. *Semiotica.* Amsterdam, v. 69, n. 3/4, p. 269-302.

Norman, D. A. (1986). Cognitive Engineering. In Norman, D. A., Draper, S. W. (eds), *User Centered System Design in New Perspectives on Human-Computer Interaction*. Hillsdale: Lawrence Erlbaum Associates, p. 31-61.

Peirce, C. S. (1931-1935). *Collected Papers of Charles Sanders Peirce.* Vols. 1-6: Charles Hartshorme e Paul Weiss (ed.). Vol I: Principles of Philosophy; Vol. II: Elements of Logic; Vol. III: Exact Logic; Vol. IV: The Simplest Mathematics; Vol. V: Pragmatism and Pragmaticism; Vol VI: Scientific Metaphysics. Thrird Printing. Cambridge: Harvard University Press, 1974.

Preece, J. et al. (1994). *Human-Computer Interaction*. Reading: Addison-Wesley Publishing Company.

Santaella, L. (1996). *O que é Semiótica*. 12.ed.. São Paulo: Editora Brasiliense.

Schapiro, M. (1996). *Words, Script and Pictures: Semiotics of Visual Language*. New York: George Braziller, inc..

Schrage, M. (1996). Cultures of Prototyping. In Winograd, T. (ed.), *Bringing Design to Software.* New York: Addison-Wesley Publishing Company, p. 191-205.

Sebeok, T. A. (1994). *Signs – An Introduction to Semiotics*. Toronto: University of Toronto Press Incorporated.

Souza, C. S. (1993). The Semiotic Engineering of User Interface Languages. *International Journal of Man-Machine Studies,* v. 39, p. 753-773.

Souza, C. S., Leite, J. C. (1997). Projeto de Interfaces de Usuário. In Lucena, C. J. P. (ed.), M*onografias em Ciência da Computação.* Rio de Janeiro: PUC Rio.

Winograd, T. (1995). From Programming Environments to Environments for Designing. *Communications of The ACM*, New York, v. 38, n. 6, p. 65-74.

Winograd, T. (1996). Introduction. In Winograd, T. (ed.), *Bringing Design to Software* New York: Addison-Wesley Publishing Company, p. xiii-xxv.

Instituto de Computação
Universidade Estadual de Campinas
Caixa Postal 6176
13083-970 Campinas, SP
BRASIL
reltec@dcc.unicamp.br