The Semiotic Inspection Method

Clarisse Sieckenius de Souza[3], Carla Faria Leitão[3], Raquel Oliveira Prates[1,3], Elton José da Silva[2,3]

[1]Departamento de Ciência da Computação – UFMG
Av. Antônio Carlos 6627, Prédio do ICEx, sala 4010, Pampulha, Belo Horizonte - Minas Gerais, CEP 31270-010
rprates@dcc.ufmg.br

[2]Departamento de Computação – UFOP
Campus Universitário, Morro do Cruzeiro, Ouro Preto – MG, CEP 35400-000
elton@iceb.ufop.br

[3]SERG – Departamento de Informática – PUC-Rio
Rua Marquês de São Vicente, 225, Gávea, Rio de Janeiro – RJ, CEP: 22453-900
{clarisse, cfaria}@inf.puc-rio.br

ABSTRACT
This paper describes semiotic inspection, a semiotic engineering evaluation method. It briefly identifies the essence of theory-based evaluation methods in HCI. Then it provides a detailed description and illustration of this method, which is based on a semiotic theory of HCI. It discusses its theoretical stance in semiotic engineering compared to the communicability evaluation method, as well as the perceived advantages and disadvantages of semiotic inspection. Finally, it points at the next steps in the semiotic inspection research agenda.

Categories and Subject Descriptors
User Interfaces (D.2.2, H.1.2, I.3.6)

General Terms
Evaluation/methodology

Keywords
Semiotic Engineering, Inspection Methods, Qualitative Methods.

1. INTRODUCTION
The importance of HCI evaluation cannot be underestimated. Decisions made throughout design or re-design activities can and should be informed by iterative evaluation sessions. Since the choice of evaluation methods is very wide, it is difficult to detect, understand and compare their purpose, power and limitations. Theory-based methods are usually easier to examine and assess than heuristic methods, because theories have inherent explanatory and inferential power that heuristics don't have. Hence it is possible to anticipate many of the results, consequences and compromises of using such methods.

This paper describes semiotic inspection, a theory-based evaluation method. Like other inspection methods (e.g. Heuristic Evaluation [13] and Cognitive Walkthrough [17], the semiotic inspection does not involve actual users during evaluation. We provide a detailed account and illustration of the method's procedures, as well as critical considerations about its main characteristics and limitations in view of its foundational theory. Potential evaluators willing to use the method should be able to understand it, and to appreciate the kind of knowledge that can be obtained as a result of its application.

There is a tight but frequently underestimated connection between methods and theories. In very broad terms, a theory of HCI consists of a set of articulated concepts that express a given abstract characterization of what human-computer interaction is (or should be). A theory-based evaluation method refers to the possibility of transposing the theory's abstract characterization of its object to the empirical realm. Thus, in HCI, the purpose of theory-based evaluation methods is to assess the quality of interfaces and interaction in the light of a given perspective on human-computer interaction. The results that can be obtained when such methods are applied to the data are always limited and determined by the corresponding theoretical perspective, which creates an important feedback loop between theory and practice. The quality and consistency of evaluation results can indicate an opportunity or a need to improve the theory; and improved theories can increase the power and reach of evaluation methods. Cognitive theories of HCI, and cognitive evaluation methods, are more abundant than others [3]. Semiotic inspection however, like communicability evaluation [15], is based on semiotic engineering [7], a semiotic theory of HCI.

Unlike cognitive theories, which have tended to follow a generalization path, semiotic engineering views human-computer interaction as a set of unique and contingent instances of metacommunication from designer-to-user. More specifically, a system's interface, along with its interaction patterns, is characterized as a message that can itself exchange messages with the users. Hence a *meta*communication perspective. The gist of the metacommunication message content is: "*Here is my [the designer's] understanding of who you [users] are, what I've learned you want or need to do, in which preferred ways, and why. This is the system that I have therefore designed for you, and this is the way you can or should use it in order to fulfill a range of purposes that fall within this view*" [7]. A simple analogy with drama – a well known literary genre – helps illustrate the essence of this perspective. Just like the overall message of the playwright is captured and interpreted as we, as spectators, follow the dialogs between characters, the message of the designer is unfolded as we, as users, participate in the dialogs with the system.

IHC 2006 – VII Simpósio Sobre Fatores Humanos em Sistemas Computacionais. Novembro, 2006, Natal, Rio Grande do Norte, Brazil. ACM .2006 ISBN 1-59593-432-4/06/0011.

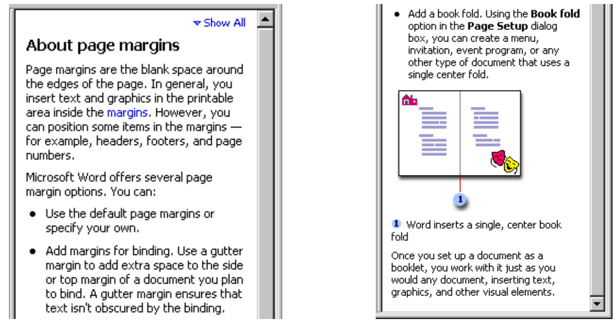
In semiotic engineering, communication is the top-level category that enables HCI experts to collect and integrate unique instances of user-system interaction. All evaluation methods derived from this theory seek to increase knowledge about designer-to-user metacommunication. Evaluators using semiotic engineering methods are the "producers" (and reporters) of knowledge referring to *unique cases* of HCI, not the "consumers" of knowledge about HCI *in general*, which is applied in a particular instance of evaluation. Semiotic engineering evaluation methods are qualitative and interpretive [9]. They provide tools to facilitate the evaluator's interpretation and assessment of the quality of metacommunication across an indefinitely wide scope of human-computer interaction instances. These include innovative technologies that *par excellence* constitute unique cases of evaluation.

Given the above-mentioned highlights of the epistemological and methodological profile of semiotic inspection, in the next sections we begin by presenting a detailed account of the method. Then we illustrate the results of a semiotic inspection of Microsoft Word® [12]. In the last section we conclude by positioning the method within semiotic engineering and in comparison with communicability evaluation., We discuss its perceived advantages, disadvantages, and future work. Additionally, we indicate the next steps in the semiotic inspection research agenda.

2. THE METHOD

The semiotic inspection method proposed by semiotic engineering examines a large diversity of signs to which users are exposed as they interact with computing artifacts. The concept of *sign* is at the heart of semiotic inspection. The message sent from designers to users is expressed through signs of one or more *signification systems*. In the Peircean tradition, a sign is anything that stands for something (else) in someone's perspective [14]. Some of the most frequent signs in computer systems' interfaces are: widgets, images, words, colors, dialog structures, graphic layouts, etc. A signification system is the result of culturally (and, in the special case of HCI, also artificially) encoded associations between contents and expressions [10]. For example, words and images typically come from signification systems that exist in culture outside the specific context of human-computer interaction, whereas mouse pointers and dialog boxes belong to signification systems that are native to computer applications.

The semiotic inspection method is carried out in five distinct steps: (i) an inspection of online and offline documentation and help content; (ii) an inspection of static interface signs; (iii) an inspection of dynamic interaction signs[1]; (iv) a contrastive comparison of designer-to-user metacommunications identified in steps (i), (ii), (iii); and finally (v) a conclusive appreciation of the quality of the overall designer-to-user metacommunication. Through documentation and help, interface and interaction, designers *communicate* to users their design vision. Parts of this communication are more easily identified as *metacommunication* than others. For instance, Word online help adopts the 3rd person when referring to the system (see Figures 1a and 1b).



(1a) (1b)

Figure 1: Microsoft Word Help referring to the system as (1a) "Microsoft Word" and (1b) as "Word"

In Figure 1a the user is told:

> *"Page margins are the blank space around the edges of the page. In general, you insert text and graphics in the printable area inside the margins. However, you can position some items in the margins — for example, headers, footers, and page numbers.*
>
> *Microsoft Word offers several page margin options. You can:*
>
> - *Use the default page margins or specify your own.*
> - *Add margins for binding. Use a gutter margin to add extra space to the side or top margin of a document you plan to bind. A gutter margin ensures that text isn't obscured by the binding."*

And in Figure 1b the user is told:

> - *"Add a book fold. Using the Book fold option in the Page Setup dialog box, you can create a menu, invitation, event program, or any other type of document that uses a single center fold.*
>
> *Word inserts a single, center book fold*
>
> *Once you set up a document as a booklet, you work with it just as you would any document, inserting text, graphics, and other visual elements."*

Clearly, this must be someone *other* than "the system" speaking to the user, otherwise Word wouldn't refer to *itself* as Microsoft Word (1a) or simply Word (1b). Semiotic engineering proposes that the legitimate author of this message is the designer (or a spokesperson for the design team[2]).

Other parts, however, may suggest that "the system" is talking to the user. Such is the case of the dialog presented in Figure 2. The title "Microsoft Word" can be easily interpreted as the speaker's identity, especially in this kind of communication where the user is being addressed in 2nd person ("[You] Enter a valid drive letter"). Here again, according to semiotic engineering, it is the designer who is talking to the user *via* the system.

---

[1] *Static* signs express (and mean) the system state. They can be found in screen shots of the system's interface. *Dynamic* signs express (and mean) the system behavior. They can only be found as a user *interacts* with the system. They are best captured in video recordings of continuing user-system interaction.

[2] In this paper we will say that "the designer" communicates, or talks, or speaks, or means, and so on. This is just a metonymic rhetoric that, in most cases, should actually be interpreted as whoever speaks for "the design team".

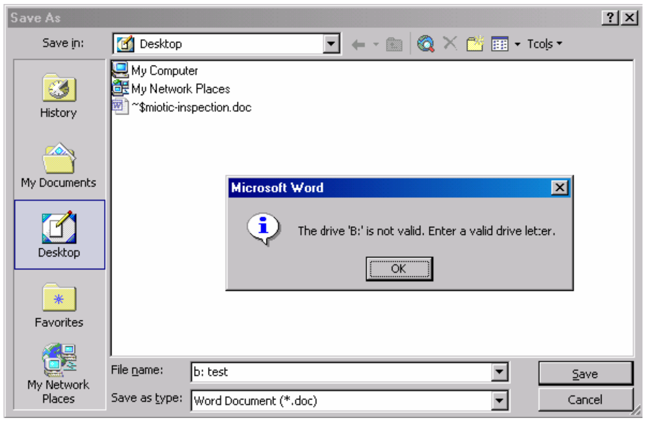


Figure 2: Dialog suggests that "Microsoft Word" is speaking

Nevertheless, since it is the designer (or the design team) that decides what to communicate to users, and what to expect/allow users to communicate in return in all interactive situations, the system is only delivering the designer's discourse and playing the designer's part in communication. The system is thus the *designer's deputy* at interaction time.

A semiotic inspection examines the various forms of designer-to-user communication, and collates the essence of metacommunication derived from explicit discourse about the system, typically found in documentation and help content, with that derived from implicit discourse, unfolding in interactive turns. For instance, Word help explicitly says that:

"Microsoft Word offers several page margin options. You can:

- *Use the default page margins or specify your own.*
- *Add margins for binding. Use a gutter margin to add extra space to the side or top margin of a document you plan to bind. A gutter margin ensures that text isn't obscured by the binding."* [12]

The way this content is conveyed through interaction is not always as effective. For example, the explicit help message above explains that "a gutter margin ensures that text isn't obscured by the binding." However, this idea is not so clearly communicated in the page layout view of a Word document. Compare the three images in Figures 3a,b,c. In Figure 3a, a visual representation of the *gutter* effect is clearly shown on the left-hand shaded margin of the page drawing (see detailed view). However, on the page layout view, the gutter is not shown (Figure 3b). The communication of the effects of the gutter in the margin could be made clearer on the page layout view, following the same visual design guidelines of the page setup dialog (Figure 3c). The communicative issue that a semiotic perspective can capture is that, as one moves from one signification system (textual) to another (visual), and from one context (help and explanation) to another (use and action), the criticality and purpose of interactive elements and strategies are not equally well-conveyed. This may be the cause of interaction breakdowns, or of sub-optimal user performance in formatting tasks.

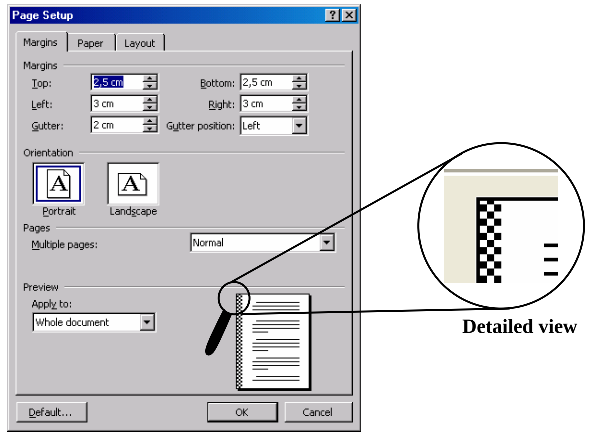


Figure 3(a) Page setup dialog – binding is clearly shown

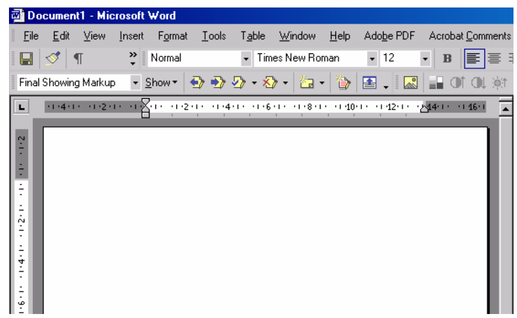


Figure 3(b) Page layout view – binding is not shown

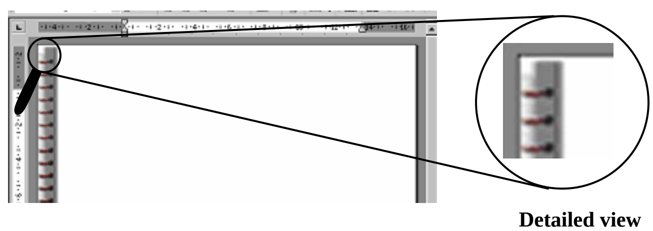


Figure 3(c) Suggestion for visual depiction of binding in page layout view

The aim of a semiotic inspection is to detect communicative problems (actual or potential) and opportunities for improvement. This should contribute to enhance the communicability of the interactive artifact, the distinctive quality of systems that communicate efficiently and effectively to users their underlying design intent and interactive principles [15].

Overall procedure

The overall procedure of the semiotic inspection method is depicted in Figure 4. The evaluator examines signs in three distinct communicative settings: explicit metacommunication in documentation and help contents; implicit metacommunication in static screen signs; and implicit metacommunication during exploratory interaction. Then, he or she performs a thorough contrastive analysis of the collected communicative material. The purpose of the analysis is to reconstruct an integrated version of the metacommunication message, tracking cases of inconsistency and ambiguity, if there are any.

Given the goal of evaluation, it is important that the evaluators assume a very specific perspective on the analysis, and keep to this perspective steadily throughout the whole process. Evaluators should be the users' advocates, a position that they can legitimately hold because of their solid technical knowledge in HCI. Although they are not genuine users, they can oversee the users' interest and benefit in a wide range of possible interactive scenarios. Just as a lawyer does not "replace" a client (but only "represents" the client in certain situations where the lawyer's technical knowledge benefits the client), the evaluator does not "replace" the user(s). The evaluator's technical knowledge about HCI and about the users' interests should benefit the user, just as that of a lawyer's should benefit the client.

Keeping to this perspective promotes the production of interpretations that users might plausibly produce themselves for signs in all inspection's steps. Such interpretations result from the combination of signs that belong to the *presumed users'* culture, and those that arise in the specific context of activities and tasks that are performed by the evaluator, on behalf of his or her users' constituency, throughout the inspection. Interpretations are, however, enriched by the *evaluator's own* interpretations of signs, which necessarily include elements of judgmental decisions about the quality of communication in human-computer interaction. For instance, a *user* that encounters the sign  on a screen will probably interpret it as a sign for "information". The evaluator, however, is able to enrich this interpretation with further meanings related to a particular *brand* of products, a particular operating system, and so on. This enriched perspective may play an important role when evaluating multi-platform software, for example, or when deciding about how to introduce new technology. A sign that will go unnoticed for some users will possibly be strange or unusual to others. The same applies to multi-cultural interfaces, and many other kinds of communication meant to cut across boundaries of different sorts, including temporal. In this case, an enriched perspective may allow the evaluator to foresee some interpretations that are plausible in different stages of the users' learning cycle, as they begin to develop their own personal strategies of interaction. Some signs may motivate strategies that will make users more productive (e.g. spend a smaller number of steps to achieve a particular goal), whereas others may never give users a hint that more productive strategies can be found.

Finally, as is the case for all inspection methods, an exhaustive inspection may be quite costly. For instance, think of the practical feasibility of an exhaustive heuristic evaluation [13] or cognitive walkthrough [17] for Microsoft Word. The semiotic inspection is, therefore, typically made for a *sample portion* of the artifact. As with any other inspection method knowing how to choose the sample is important. Different criteria can be used for this. One possibility is to choose portions that the design team thinks are most relevant, possibly because they have been hard to design, or because they are critical for the commercial success of the application. Another is to choose portions that are commercially advertised as the product's advantage over competitors. Yet another is to choose basic core functions which all users are likely to perform in any kind of anticipated use situation. A combination of criteria can also be used, if different facets of the product are expected to be evaluated.

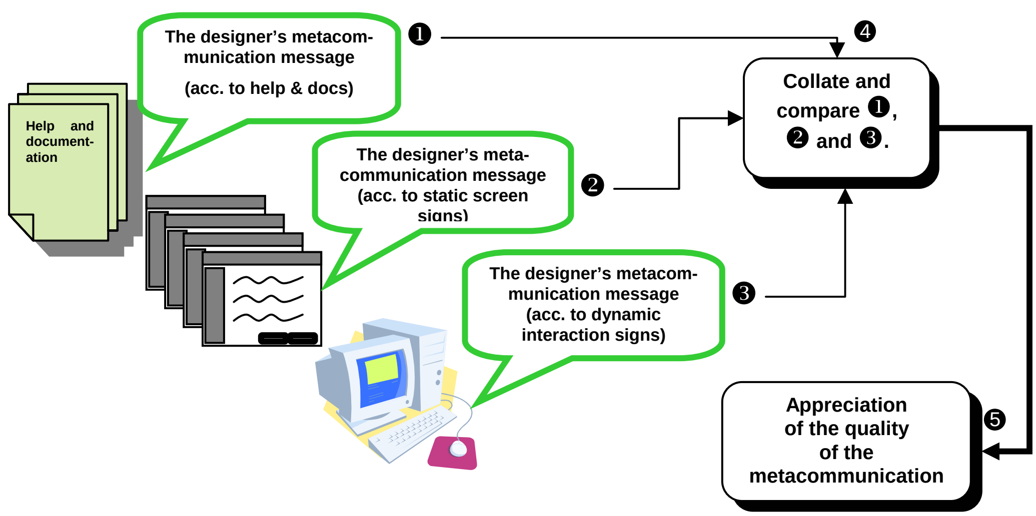


Figure 4: Overall procedure for the semiotic inspection method

Once the focus and portion of the evaluation is decided, the evaluator must plan the inspection carefully. The selected portions of documentation to be examined, as well as the focal activities to be performed must be organized and prepared. As part of the preparation, the evaluator should produce inspection scenarios [4]. Exploratory interaction *must* be guided by inspection scenarios. They are necessary because aimless interaction cannot provide the appropriate communicative *intent*, a crucial element for interpreting any kind of communication.

Step 1: Examining signs in documentation and help contents

The purpose of this examination is to reproduce the contents of metacommunication ❶ in Figure 4. A convenient *template* of this message's content is, as presented in the introduction: "Here is my understanding of who you are, what I've learned you want or need to do, in which preferred ways, and why. This is the system that I have therefore designed for you, and this is the way you can or should use it in order to fulfill a range of purposes that fall within this view".

Documentation and help provide abundant hints about who the expected users of the artifact are. For instance, Word documentation and online help say that "Word offers great new ways to polish the formatting in your document." The word choice for "polish" communicates the idea that quality formatting is important for its users. If not, Word may not be the appropriate tool. And, indeed, Microsoft provides simpler text editors for use situations where formatting is unnecessary, irrelevant, or impairing.

Detailed information about (presumed) users' needs is also present in documentation and help content. For instance, Word help says that: "[The] key features of Word in Microsoft Word [are]: easier formatting, collaborative document creation, and speech and handwriting recognition." The message is elaborated by detailed explanations about how to use each of these features. The elaboration tells users about the designer's interpretation of the users' preferences. For instance, when explaining how to perform everyday tasks such as table editing, the designer's message says that: "Word now offers drag-and-drop copying of tables." In other words, the designer communicates that the direct manipulation style for table editing is a bonus for (some? many? most?) users. Interestingly, in table editing instructions, help contents tell users how to "move or copy items in a table" saying that:

- *"To move the selection, drag it to the new location.*
- *To copy the selection, hold down CTRL while you drag the selection to the new location."* [12]

The choice of a direct manipulation method in "how to" instructions communicates the designer's expectation that this be a preferred method for interaction.

Communication about why users are expected to prefer to use the product and its key features is not always explicit (if present at all) in documentation and help contents. They are typically present, however, in advertisements, websites, and in the product's package itself. In the current website for Word, for instance, Microsoft tells users to: "Help protect your company assets by preventing recipients from forwarding, copying, or printing important documents by using information rights management (IRM) functionality. You can even specify an expiration date for the message, after which it cannot be viewed or changed."[3] This is one of the reasons *why* users should use Word collaborative document creation features.

Filling out the *template* for the metacommunication message at this stage is made possible by following the same interpretive procedure. The portion about the system, and how it should be used to fulfill a number of purposes, should refer only to the portions of the system under examination. Notice that the message integrates various *speakers*' messages: that of technical designers, that of manufacturers, that of marketing professionals, and so on. This integrated perspective is a bonus of the semiotic inspection method compared to other methods where task-oriented interaction is the main or sole focus of analysis. Users get all these messages. Their experience will therefore be affected by all such meanings, and not just by their particular goals and tasks in any given interactive scenario.

Step 2: Examining static interface signs

The purpose of this examination is to reproduce the gist of metacommunication ❷ in Figure 4. The same message content *template* used in step 1 should be used again. An inspection of static screen signs should ideally confirm, and possibly elaborate on, the contents derived from documentation and help. For instance, it is very difficult to communicate direct manipulation editing opportunities statically. As a consequence, an examination of Word's static screens for table editing like the one in Figure 5 shows that interaction based on toolbox button-pressing and menu selection is much more effectively communicated than drag-and-drop interaction opportunities. Notice that there is no suggestion that the selected text can be dragged and dropped to another cell (like the one below its current location, for example) in order to achieve a move or copy effect.

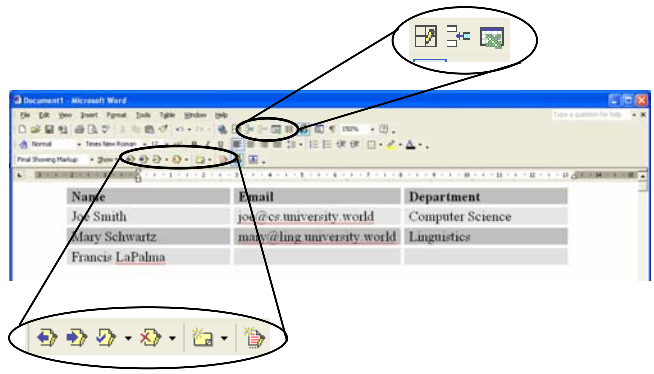


Figure 5: Communicating table editing tasks in Word interface's standard configuration

Also, some of the key features advertised for Word, in our examples, are not as clearly communicated statically as others. For instance, signs for collaborative text editing are clearly present on the screen in Figure 5 – see the reviewing tools ostensibly shown in the lowest toolbox on top of the screen. Signs for handwriting input, however, are not visible on this screen. They are embedded rather deeply in a specific toolbar (the language toolbar), and thus the communication that this feature is there to be used is quite obscure.

As the evaluator fills in the *template* for metacommunication ❷ in Figure 4, he or she may occasionally find mismatches, contradictions, or ambiguities, which may have a negative impact on the presumed user's interpretation of the artifact. The example just mentioned above – the location of tools for processing hand written input – illustrates how static signs may not convey the intent expressed in explicit metacommunication messages of documentation and help material.

Step 3: Examining dynamic interaction signs

The purpose of this examination is to reproduce the gist of metacommunication ❸ in Figure 4. Dynamic signs play a crucially relevant role in designer-to-user communication. In essence, they confirm or elucidate the anticipation of meanings from static screen signs. A classical issue in interface design – choosing menu (option) words or phrases – illustrates this point. For example, a menu labeled "table" suggests that all options beneath it refer to table elements. It is therefore curious to verify during interaction that list elements, and even paragraphs and headings, can be *sorted* if the Table > Sort … menu selection path is used.

[3] Material shown at http://www.microsoft.com/office/word/prodinfo/default.mspx on June 18, 2006

Other anticipation issues can be evaluated in communication through dynamic signs. For example, pressing the button "cancel" in most dialogs in Word does not effect the final result of the dialog, neither does it leave traces of intermediary parameter settings commanded before the cancel button was pressed. For example, if the user selects File > Print … and fills up various configuration parameters for the printer, but ends up canceling the operation, the parameters are typically reset to default values. So, the next time the user enters the dialog, the context of conversation does not show a trace of the previously canceled dialog. The recurrence of this pattern helps the user interpret "canceling" as an action that sweeps away all the modifications equivocally provoked by the wrong path of interaction. The same is not true, however, if the user chooses File > Save as … and, in the follow-up dialog he or she decides to create a new folder to store the copy to be saved. After pressing the  icon which causes the new folder to be created, canceling the "Save as …" activity will not cause the deletion of the new folder. This ambiguity is likely to cause communicative breakdowns during interaction, when the user tries to anticipate whether or not side effects will linger after canceling the current action.

The *template* for the metacommunication message can thus be filled out once again, based on the interpretations that the evaluator derives from actually interacting with the artifact. The next two steps in the method finalize the analysis and produce the kind of result illustrated later in this paper.

Step 4: Collating and comparing metacommunication messages

As hinted in previous steps, metacommunication messages ❶, ❷ and ❸ are not identical. In fact, they *should not* be identical because they are expressed in different signification systems. They have distinct expressive power, and may thus serve different communicative purposes. For example, documentation and help use extensive natural language text to explain and illustrate interactive opportunities and their corresponding effects. The same is not true of communication that is restricted to static interface elements like icons, menus, buttons, pointer shapes, etc. Likewise, the meanings conveyed by signs that appear in the dynamic sequence of input-output turns in user-system interaction communicate aspects of the technology that are difficult if not impossible to convey otherwise. For example, the sense of precision and control associated to moving drawings with a combination of the CTRL and the arrow keys compared to drag-and-drop interaction with the mouse is difficult to express with words, and virtually impossible to communicate with typical interface widgets. The user has to *experience* this precision and control dynamically in order to understand what designers mean when they say, in online help, that: "You can also "nudge" an object in small increments by selecting it and pressing the arrow keys."

However, although metacommunication messages should not be expected to be identical, they should certainly not be inconsistent with each other. Here the users' advocacy perspective plays an important role. The evaluator should intentfully explore the possibility of assigning plausible contradictory or ambiguous meanings to the signs that constitute the three messages. Just like the questions proposed to guide Cognitive Walkthroughs [17], questions such as 1-5 below can serve as scaffolds for the evaluators' analysis.

1. Would the user plausibly be able to interpret this (sign, message) differently? How? Why?
2. Would this interpretation still be consistent with the design intent?
3. Does the current interpretive path remind me (the evaluator) of other interpretive paths I have used in the semiotic inspection? Which one(s)? Why?
4. Can classes of static and dynamic signs be drawn from the semiotic analysis? Which ones?
5. Are there static or dynamic signs that are apparently misclassified according to the classes proposed in 4? Can this affect communication with or through the system? How?

These questions are not the only ones that the evaluator can or should ask, but they provide useful guidance and input for productive semiotic inspection. Because interactive artifacts have unique signification systems, which are especially designed to convey their unique qualities and features compared to competing artifacts (if there are any), consistency and regularity play a major role in creating or evoking patterns of signification that are familiar to the user. Existing signification systems can be encountered in the users' cultural environment. Such is the case of natural language words and expressions, of images, and even gestures or sounds used in multimodal and multimedia interaction. Some ambiguities or strict inconsistencies can be tolerated in many cases. Culture provides numerous examples of ambiguities and inconsistencies with which everyone deals without much trouble on a regular basis. For example, it is more natural in our culture to *tell* a system to close an unsaved file and then be asked whether we want to keep changes or not, than to have the system not understand a simple "close" command, but only "save and close" or "close without saving". The ambiguity does not offend communication if it is recognized by the system. If it isn't, then this may cause important breakdowns.

However, especially in the case of innovative technologies, interaction and interface design may require innovative signs and signification "systems", which in this case are not only artificial, but unfamiliar to the user. Here users will have to learn the system as they learn a new language, hence the importance of analogies (on a more concrete or a more abstract level) and regularity (syntactic and semantic). Ambiguities of any sort are likely to represent learning obstacles, as users will be able to interpret signs in different ways and not make the appropriate anticipation in critical situations.

Step 5: Appreciating the quality of the overall designer-to-user metacommunication

This step produces the final result of a semiotic inspection. A conclusive appreciation will typically contain the following parts:

i. A brief description of the method (helps the reader perceive how the evaluator understands and uses it)
ii. The criteria for selecting which portions of the artifact to inspect
iii. For each of the three communicative settings – documentation and help, static interface signs, and dynamic interaction signs – provide:

a. An identification of relevant signs (list & justify their relevance)

b. An identification of sign systems and sign classes in use

c. A (revised) version of the designer-to-user metacommunication message

iv. A substantiated judgment of communicative problems, actual or potential, that may prevent users from getting the designer's message, and interacting productively with the artifact.

In the next section we will illustrate item **iv** above in order to show the value of the semiotic inspection method.

3. AN EXAMPLE OF SEMIOTIC INSPECTION RESULTS

Keeping with Microsoft Word, for sake of convenience, we will illustrate the results of the semiotic inspection method applied to collaborative reviewing activities. As mentioned in the previous section, this is one of the main features of the artifact. The metacommunication message derived from documentation and help content tells us that collaborative reviewing allows users to send files for review, track changes and comments to an existing file, review (accept, reject) changes, read comments, and review version information.

An inspection of static signs on screens related to collaborative reviewing reveals that this activity is communicated by static signs spread throughout different parts of the interface. Switching the "track changes" feature on, tracking changes, accepting and rejecting them, commenting, editing and deleting comments, and viewing the reviewing pane are communicated by means of a structured sign: the "reviewing toolbar" (Figure 6).

Figure 6: Word's reviewing toolbar

Parts of this structure are also signified as another class of sign – as menu options. Such is the case for switching the "track changes" features on and off and for creating comments. However, "track changes" appears in the "Tools" menus, whereas "comments" appears in the "Insert" menu. The possibility of moving to the next or previous change/comment is also communicated as a menu option, but neither in the "Tools" nor in the "Insert" menu. It appears in the "Edit" menu, associated to the "Go to …" option (Figure 7). Finally, sending a file with a review form to collaborators is achieved through features communicated in the "File" menu, and viewing markup is an option in the "View" menu.

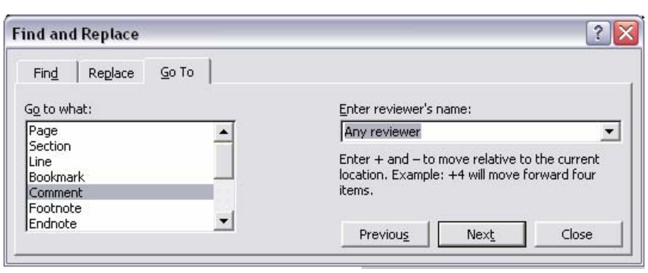


Figure 7: Accessing Word's comments through menu options and dialogs

An inspection of dynamic signs reveals that the full benefits of Word's collaborative text editing requires further understanding and learning. Although the features communicated in the reviewing toolbar are reasonably easy to interpret and to use effectively in communicating with the system, the other features may lead to major communicative breakdowns, including certain system crashes. Starting with the goal of sending a file to a collaborator for reviewing, the inspector will find that activating the "`Send to > Routing recipient …`" option in the File menu may cause part of the system to crash (See Figure 8).

Figure 8: A communicative breakdown with Word

Notice that the system response to the interaction gives no hint about the cause of the problem, neither does it suggest what the user could try to do in order to repair the situation. Some may rightfully argue that the crash in case is probably not caused by Word, itself, but by some plugin, add-on, or other component that is missing in the user's environment or operating system. This is true. However, since it is a design strategy to rely on plugins, addons and components, the designers "should" let users know about the perils of this strategy, and ideally trap component-related errors and communicate them more appropriately. A further problem is that the user cannot understand the meaning of "routing recipient" during interaction precisely because the system, in its current configuration, cannot handle exploratory interaction – it crashes on the way.

Another communicative barrier that becomes clearer in dynamic sign inspection is that *reviewing* is not a unified sign in an important signification system: the menu structures. The importance of this signification system stands out when accessibility is an issue. Toolbars are designed to be activated by mouse clicks. Menus and menu options, however, can be accessed either by mouse clicks or by key stroke combinations (and keyboard shortcuts). Thus, if a particular user is not able to communicate with mouse clicks, then he or she is unlikely to understand how to express reviewing intents that are scattered throughout various menu structures, apparently unrelated with each other.

Moreover, menus have stronger expressiveness than toolbars for communicating structured meanings. Menus can be grouped and recursively organized. Thus, they can help users infer the meaning of certain sequences or classes of related interactive sign. For this reason, menus are important for beginners, or for experts who are trying to use some particular feature for the first time (or long after they did it for the last time).

Finally, a number of messages encountered during the dynamic sign inspection belong to other signification systems – namely other artifacts that are part of the Microsoft® Office® package (like Outlook®, for example), or the Windows® operating system itself. The success of communication across Windows artifacts depends, to a great extent, on the installation options selected by the user, possibly a long time ago. It also depends on modifications that the user may have decided to do later on, like removing Outlook from his or her computer, or setting another mailing tool as the default one. Communicating such dependencies is not only a highly complex task *per se* (note that even help contents don't provide the full range of possibilities that can explain why the user obtained the message in Figure 8), but it can be overwhelming for most Word users.

So, choosing to communicate only part of the whole set of available reviewing tasks by means of a toolbar may be a good alternative. First, it bundles the core reviewing tasks into a single sign, and communicates that commenting/editing some previously edited text and accepting/rejecting changes are strategies that can be used in collaborative writing work. Second, it breaks the complete collaborative text editing workflow into increasingly complex parts – users may well benefit from never going beyond the basic steps.

However, choosing not to represent reviewing as a separate menu option may cause some communicative problems. First, as already mentioned, it may bring about accessibility obstacles. Second, and perhaps more importantly from a return-on-investment point of view, the user may never realize that there is more to collaborative reviewing than is communicated in the toolbar. This is a problem for advanced users who would not have much trouble configuring Office components or mail systems in such a way as to benefit from more agile communication with collaborators. They are not clearly *told* about such possibilities. Hence, they may never wonder if they exist, and end up being less productive than they could be. This possibility becomes highly probable in view of another design feature that comes across in the dynamic sign inspection step – the communicative effects of retracting menus. If the user never suspects that a particular menu option is useful, he or she will never choose it. As a consequence, this option will not show up in quick menu browsing interaction. The result is similar to that of a self-accomplished prophecy – users will *not need* to use these options, because not suspecting that they exist they will learn to get by without them. The point is that all the resources invested in designing and programming this portion of the artifact may have been wasted. Also, if these poorly communicated features are marketed as key features of the system, consumers may be frustrated and angry that they *are not* part of the product (although they are, as we know).

4. CONCLUSION

As is the case with qualitative methods in general [9], semiotic inspections performed by different individuals, or even by the same individual at different times, are not necessarily homogeneous. According to semiotic theory [10][14], the interpretation of signs is always affected by that of previous and co-occurring signs. Therefore, there are always a number of factors that can slant one's interpretation in novel directions.

As with any theory-based method, the higher the theory knowledge of the evaluator, the better the results. In the specific case of semiotic inspection, the semiotic awareness of the evaluator improves quality to assign plausible contradictory or ambiguous meanings to the signs that constitute the messages in the three communicative settings. The method, however, provides scaffolds that can guide the inspection, which is especially useful for less experienced evaluators. A set of questions that can be asked about relevant signs is proposed to support analysis in step 4. Scaffolds are not girders, though, and as such these questions can (and should) be replaced by more productive ones if they are available or emerge during the inspection. The girders are the theory, semiotic engineering,[7] which supports the whole structure of the method, and warrants rigorous analyses even in the absence of predictions and metrics.

Also because of its theoretic foundation, semiotic inspection inherits from semiotic engineering the ability to account for different modes and codes of communication, to identify different interlocutors and their respective intent, as well as to classify the various signification systems designed to be used in user-system communication in different contexts of interaction anticipated at design time. This is an important advantage of the method compared to methods based on cognitive theories. These tend to focus on the cognizing subject(s), but to abstract away the semiotic environment where cognitive demands arise, are expressed, and satisfied. Our semiotic approach, in turn, abstracts away deeper cognitive details, but by accounting for users' interpretations, it provides a handle for cognitive methods to take over. Whether both kinds of methods can be theoretically articulated with each other in higher meta-theoretical levels is a stimulating long-term research question.

The semiotic inspection method, like the communicability evaluation [15] – the other evaluation method associated to semiotic engineering, aims to assess how well the designer-to-user metacommunication message gets across to users through interactive systems' interfaces. Thus, it also evaluates the *communicability* of the artifact. However, unlike the communicability evaluation, semiotic inspection does not involve observing users' interaction with systems. As a consequence, the two methods capture different communicability aspects and perspectives.

During a semiotic inspection, there is a thorough and lengthy exploration of the interface, which produces a much more detailed definition of the designer's metacommunication message than the one produced during communicability evaluation. The main reason for this is twofold. First, the evaluator can spend a much longer time in any particular scenario than communicability evaluation test participants can. For instance, whereas an evaluator may reasonably spend 30 minutes or more exploring a single activity, test participants typically get tired and demotivated when a single lab activity spans for that long. Second, an evaluator will be intentfully looking for alternatives, details, possibilities and motivations for signs brought about during his or her inspection (i.e. he or she will be consciously engaged in a reflective path), whereas test participants are typically focused on getting to a successful end of the activity, whichever it may be. Thus, during communicability evaluation a smaller number and variety of scenarios is explored by the evaluator. As a result, the semiotic inspection method produces a more detailed and comprehensive account of the metacommunication message.

The communicability evaluation method, in its turn, focuses primarily on the diverse modes and means by which (potential) users capture and interpret the designer's metacommunication message. By involving various interpreters (participants and evaluator(s)), the level of communicability analysis achieved by communicability evaluation tend to be deeper than that achieved by semiotic inspection.

Both methods are thus complementary. Ideally, the analysis of a particular software's communicability should start by a semiotic inspection, and then proceed with a communicability evaluation, based on the inspection's findings. Whereas semiotic inspection concentrates on the emission of the designer's metacommunication message, communicability evaluation concentrates on the process of reception and interpretation. Because semiotic inspection produces an explicit representation of the overall metacommunication message, it can help elaborate rich scenarios for user observation sessions. However, a number of factors (e.g. time constraints and technical quality/fidelity of prototypes) may prevent both methods from being applied in the ideal conditions. Consequently, evaluators will have to decide which method to apply.

In temporally critical situations, for instance, semiotic inspection consumes fewer resources than communicability evaluation, and is likely to bring about good cost/benefit ratios. Also, because it is performed by expert evaluators, semiotic inspection may provide better results than communicability evaluation in the formative stages of interaction design. Users may not always interact naturally with prototypes, especially with low-fidelity ones. But expert evaluators can handle abstract or sketched representations of design much more easily. For example, they can infer certain dimensions of dynamic signs even if the design representation is no more than a set of drawings on paper, or a series of diagrams on the wall.

In the near future we intend to investigate more closely the relations between semiotic inspection and communicability evaluation. Is the semiotic inspection the first and necessary step of the communicability evaluation? At its current stage of maturity, we say that an informal semiotic inspection of the system under communicability evaluation must be carried out as a preliminary step of the method. Is the informal inspection sufficiently informative for a communicability evaluator to grasp the essence of the designer-to-user metacommunication message? Or, are there flaws in the informal procedure suggesting that a full-fledged semiotic inspection should be always performed? Or else, would an intermediary version of the semiotic inspection be more appropriate for communicability evaluation?

Once the boundary between both methods is more clearly established, it is also relevant to semiotic engineering that we investigate and define criteria to guide the choice of methods in different contexts. When is one better than the other? When are both equally good? Also for the interest of the theory, we want to compare the results of our method to the results of semiotic analyses of information and communication technology artifacts carried out with classic and non-classic semiotic methods (for example, [1], [11], [5]).

Finally, we intend to pursue a more endogenic path. We have applied preliminary versions of the semiotic inspection method in other contexts ([6], [8], [16], [2]). But it needs to be systematically and repeatedly applied to different classes of systems, in order to reveal its strengths and weakness more clearly. This should also allow us to investigate if there are variants of the method for specific evaluation contexts. In the near future it is also important to contrast results achieved with semiotic inspection and classic usability inspections, like Heuristic Evaluation and Cognitive Walkthrough.

5. ACKNOWLEDGMENTS

Clarisse S. de Souza thanks CNPq for Grant 305730/2003-7, and Raquel O. Prates thanks CNPq for Grant 475992/2003-2.

6. REFERENCES

[1] Andersen, P. B. A theory of computer semiotics. Cambridge. Cambridge University Press, 1997.

[2] Barbosa, C. M. A.; Leitão, C. F.; de Souza, C. S. Why Understanding Culture is Important for Developing Effective Online Health Support: The Brazilian Context. HCI International 2005 (Las Vegas, July 23-27, 2005).

[3] Carroll, J. M. HCI Models, Theories, and Frameworks: Toward a Multidisciplinary Science. Menlo Park, Ca., Morgan Kaufmann, 2003.

[4] Carroll, J. M. Making Use: Scenario-Based Design of Human-Computer Interactions, MIT Press, Cambridge, MA, 2000.

[5] Condon, C.; Perry, M.; O'Keefe, R. Denotation and connotation in the human – computer interface: The 'Save as . . .' command. Behaviour & Information Technology.. 23, 1 (January–February 2004), 21–31.

[6] da Silva, E. J. de Souza, C. S., Prates, R. O., Nicolaci-da-Costa, A. M. What They Want and What They Get: a study of light-weight technologies for online communities. In Proceedings of CLIHC2003 (Rio de Janeiro, Brazil, August 17-20, 2003), 135-146.

[7] de Souza, C. S. The semiotic engineering of human-computer interaction, Cambridge MA, MIT Press, 2005.

[8] de Souza, C. S. ; Nicolaci-da-Costa, A. M. ; da Silva, E. J.; Prates, R. O. Compulsory institutionalization: investigating the paradox of computer-supported informal social processes. Interacting With Computers, Amsterdam, 16, 4 (2004), 635-656.

[9] Denzin, N. K. and Lincoln, Y. S. The landscape of qualitative research: Theories and issues. Thousand Oaks, Ca. Sage Publications, Inc, 2003.

[10] Eco, U. A theory of semiotics. Bloomington, In. Indiana University Press, 1976.

[11] Goguen, J. An Introduction to Algebraic Semiotics, with Application to User Interface Design. In Nehaniv, C. L. (Ed.): Computation for Metaphors, Analogy, and Agents Lecture Notes in Computer Science (Springer Berlin / Heidelberg, 1999), 1562, 242.

[12] Microsoft Corporation. Microsoft® Word® 2002. All rights reserved.

[13] Nielsen, J. Heuristic Evaluation. In Nielsen, J. & Mack, R. L. Usability inspection Methods. Eds. John Wiley & Sons, New York, NY, 1994, 25-64.

[14] Peirce, C. S. The essential Peirce (Vols. I and II) - Edited by Nathan Houser and Christian Kloesel. Bloomington, In. Indiana University Press, 1992,1998.

[15] Prates, R. O., de Souza, C. S., and Barbosa, S. D. J. A method for Evaluating the Communicability of User Interfaces. Interactions, 7, 1 (Jan.-Feb. 2000), 31-38.

[16] Seixas, M.L.A.; de Souza, C.S. Um método de avaliação para interfaces baseadas em mapas, VI Simpósio sobre Fatores Humanos em Sistemas Computacionais (IHC 2004), (Curitiba, Brazil, Oct. 17-20, 2004), 159-170.

[17] Wharton, C., Rieman, J., Lewis, C., and Polson, P. The cognitive walkthrough method: a practitioner's guide. In: Nielsen, J. & Mack, R. L. Usability inspection Methods. John Wiley & Sons, New York, NY, 1994, 105-140.