Getting Users to Understand Designers via Help Systems

Milene Selbach Silveira[1] **Simone D.J.Barbosa** **Clarisse Sieckenius de Souza**[2]
TeCGraf, PUC-Rio
[1]also FACIN/PUCRS, [2]also DI/PUC-Rio
milene@inf.puc-rio.br, sim@les.inf.puc-rio.br, clarisse@inf.puc-rio.br

PUC-RioInf.MCC36/00 September, 2000

ABSTRACT
Help systems are often underutilized, because they fail to clarify users' doubts. One of the reasons for this shortcoming is the fact that help systems are usually developed at the end of an application development process. This paper presents an approach to the development of help systems throughout applications' design and development processes, supported by both a model and an architecture of online help systems. This work is developed in the context of semiotic engineering, in which interactive applications are one-shot messages from designers to users, about how to exchange messages with the applications in order to achieve a range of goals.

Keywords
online help systems; semiotic engineering; web applications

RESUMO
Sistemas de ajuda (*help*) são, quase sempre, pouco utilizados, e, quando o são, costumam não auxiliar o usuário em esclarecer suas dúvidas. Isto ocorre com freqüência porque estes sistemas são comumente construídos ao final do processo de desenvolvimento da aplicação. Este trabalho apresenta um modelo e uma arquitetura de sistemas de *help online*, no intuito de dar suporte ao desenvolvimento de sistemas de ajuda durante o processo de design e desenvolvimento das aplicações. O mesmo é desenvolvido no contexto da engenharia semiótica, na qual aplicações interativas são mensagens de "mão-única" dos designers para os usuários, sobre como trocar mensagens com as aplicações a fim de alcançar os objetivos desejados.

Palavras-chave
sistemas de *help online*; engenharia semiótica; aplicações *web*

1. INTRODUCTION

Help systems are often underutilized, or accessed only as a last resource, typically because users don't see a good and immediate cost/benefit ratio of using them. Frustrating experiences with help systems of some existing applications may prevent users from even thinking about accessing online help when they need some information.

One of the major reasons for this lies on the difficulty of developing these systems during the application design and development processes. Instead, they are developed only at the end of the process, when there isn't enough time, and when important project decisions and design rationale aren't in the immediate team memory any longer. An important cause of this late construction of help systems is the difficulty to obtain consistent and complete information about the application during the intermediate stages of the development process. Thus, if the help system is developed *a posteriori* (i.e. after finishing implementation and tests), we gain a complete structural vision but we lose part of the project memory, where we find the global vision, the principles of the decisions and the final product's underlying logic and design rationale. On the other hand, if the help system is developed *pari passu* with the application development, it would be subject to all changes to specifications and decisions, and all the frequent adjustments made during the software development process.

This paper presents a model and an architecture of online help systems proposed in [18], and describes implementation cases of this approach in real web applications. The model is based on the concepts of Semiotic Engineering [4], that aims mainly at transmitting, through the interface, the designers' intentions and project decisions that resulted in the implemented interaction possibilities. Our approach is an attempt to facilitate the capture of the fundamental elements necessary for communicating the design logic to users. In Semiotic Engineering, the interface is a meta-message from designer to users, of which the help system is a distinguished component.

In the next sections, we present the main challenges associated to online help systems, and the proposed model and architecture. We illustrate the help systems of two real web applications designed and developed at TeCGraf (our laboratory), with and without an assisting help editing tool. In the final considerations, we emphasize the role of help systems in bridging understanding gaps between users and designers, by providing extensive and multifaceted information about the designers' point-of-view.

2. ONLINE HELP MODEL

According to [8], the major problems users report with respect to existing help systems are:

- help systems don't provide the specific information desired;
- help information is not available when needed;
- help information is not accurate or is incomplete;
- it is difficult to switch between the help system and the application.

When asking for help within an application, users would like to receive answers to their more frequent oubts [1,17], summarized in Table 2.1.

Types of Questions	Sample Questions
Informative	*What kinds of things can I do with this program?*
Descriptive	*What is this? What does this do?*
Procedural	*How do I do this?*
Interpretive	*What is happening now? Why did it happen? What does this mean?*
Navigational	*Where am I? Where have I come from? Where can I go to?*
Choice	*What can I do now?*
Guidance	*What should I do now?*
History	*What have I done?*
Motivational	*Why should I use this program? How will I benefit from using it?*
Investigative	*What else should I know? Did I miss anything?*

Table 2.1: Taxonomy of Users' Frequent Doubts

A large body of research is being developed in an attempt to effectively help users overcome these problems and attain their goals, i.e., acquire the desired information. After doing a survey on the subject, it was possible to verify that most of the research on this topic deals with issues such as: taxonomies for online help systems [16]; context-sensitive help [10,20]; help for the web [2, 13, 15] and user-system dialogues [3, 5, 6, 8, 11, 14], among others.

According to the Semiotic Engineering approach, help systems are a distinguished meta-message from designers to users [9]. In this case, the designer is explicitly saying what he believes are the users' problems or tasks, what he think is the best solution for them, and how he intends to make it available to users' practical use.

This paper proposes that this designer's vision – sent to users through the help system – be captured during the design and development processes.

This knowledge elicitation (capturing the designers' knowledge about the application designed and developed by themselves) is based upon questions for the designers, classified into three major topics. From the **designers' point-of-view**:

1. What are the users' problems/needs?
2. What is the best solution for these problems? And what are the alternatives?
3. How was this made available for operational use?

These questions summarize our conclusions after relating research about available technical literature and practice in the design and development of online help systems for groupware applications on the web. Each topic can be extended into subtopics, whose answers constitute the semantic dimension of the message from designers to users about the application. These are:

1. What are the users' problems and needs?

What is the application domain?
What is the nature of the work in this domain?
Who are the actors?

What role do they carry out?
What tasks do they do?

2. What are the best solutions for these problems and needs?

What is the application?
How will this technology affect the domain?
What is possible to do with it (goals)?
What is the application useful for?
What are the advantages of the application?

Technology

What computational environment is presumed for the full operation of the application?
What does the user need to know in order to use this application?

Activities

What activities (tasks) can be carried out in the application environment?
What are the available options in the current version?

3. How can all of this be put to operational use?

Computer–Human Interaction Analogy

What is the basic computer–human interaction analogy used?

Tasks

What does each task mean?
How can/must users do that? When?
Where in the application can users do this task?
How can users do and undo (parts of) tasks?
Why is it necessary to do this or that task?
Examples of performing the task (scenarios)
Who is or isn't affected by a task or part of a task?
What do we do after finishing a task? Until when can we do that?

Given an actual context of interaction, the user must be able to answer:

What can I do now?
Where am I?
Where can I go?
Where did I come from?
What happened?

We can analyze these questions from four different perspectives: Domain, Tasks, Agent (inspired by [21]) and Application. These perspectives define the **help model** proposed here (Figure 2.1).

As we can see in this figure, the proposed model is subdivided in: **static model** and **dynamic model**. In the **static model,** we find the answers to almost all of the described questions, except the contextual ones (*What can I do now? Where am I? Where can I go? Where did I come from? What happened?*). The entities' attributes become the answers to the preceding questions, which are listed under each corresponding entity. The

questions of a contextual nature compose the **dynamic model**, because they are generated at execution time, according to the task and the actual application state.

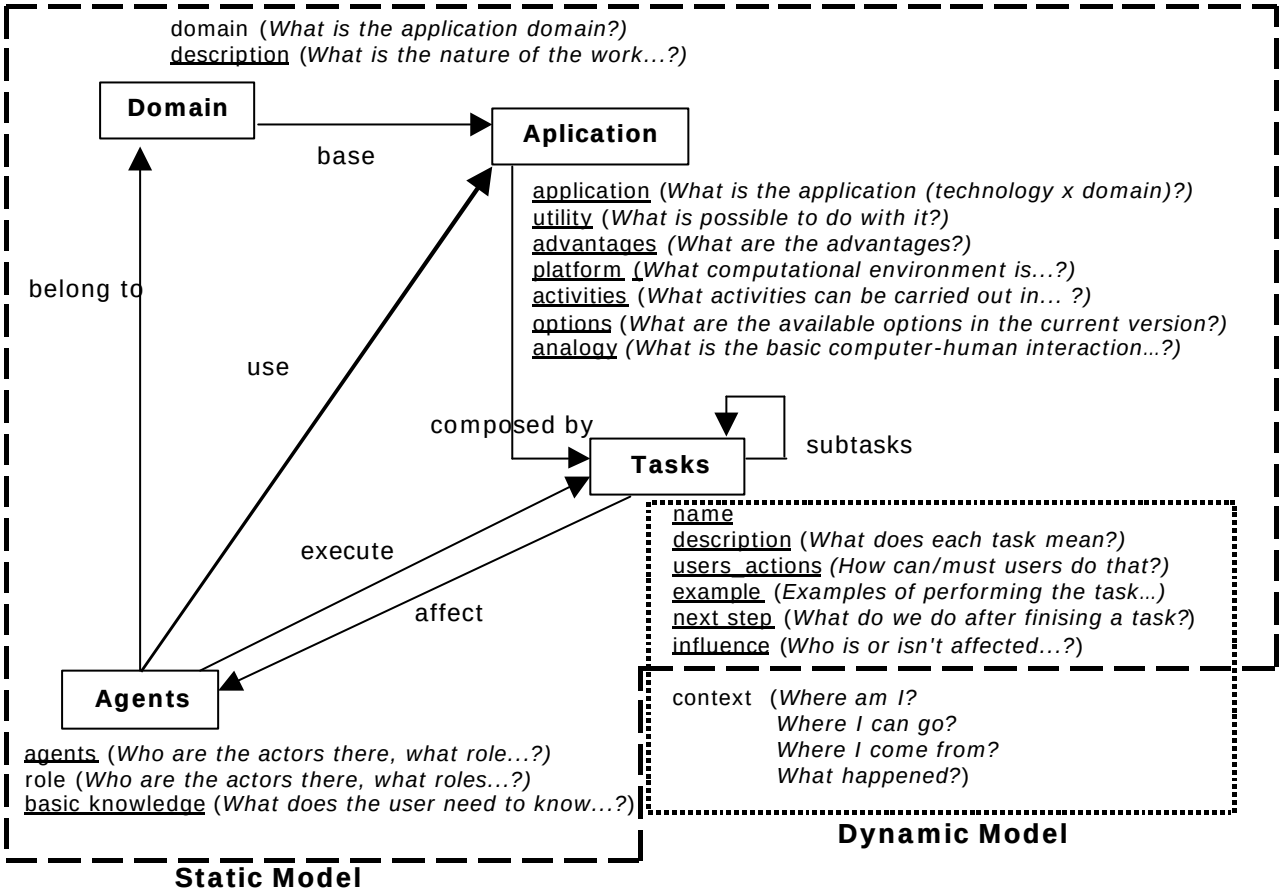

Figure 2.1: Proposed Help Static and Dynamic Models

3. COMPONENTS OF ONLINE HELP ARCHITECTURE

Based on the model above, we propose that the knowledge extracted from the designer be distributed in an architecture [19], where help information is provided in different ways and whose main components are:

- Main Help Module: this module comprises the traditional help module found in most applications, and answers most of the questions contained in the static model. This module is separate from the application (for example, an independent window or application), and contains a description of it. This description includes from a general conceptualization of the domain, to the tasks supported by the application and their means of interaction, and on to the application's functions. Besides this description, it presents scenarios with examples of application use in three distinct ways: positive scenarios, negative scenarios and tragic scenarios. The tragic scenarios provide important information for users, and can sometimes be found in traditional help systems under the title of Troubleshooting. In addition, the scenarios may appear in three distinct formats: linguistic (for example, text), static images (for example, a picture or a cartoon) and dynamic images (for example, a video or an animation of an actual interaction);

- On-demand Hints on Featured Interactive Elements: these are local help messages, with summarized explanations about some interface elements. When activated (for example, through a shortcut key or a link in the element), it presents explanations about the corresponding element, in terms of its syntax (for example, expected format for the entry) or semantics (explanation of the element in terms of the application domain);
- Direct Instructions: these messages are activated by the application, presenting warnings and instructions from the designer to users. These messages inform users about how they must proceed in a certain moment (for example, an information about what the user must do to proceed with the interaction or about what is going on in the application, so he can understand the current context);
- Error Messages: this kind of help is generated when an error occurs or when the user executes some inappropriate action. This message reveals the information contained in the hints components and explains how to perform a correct action within the current context.

In the figure below (figure 3.1) we see the content source of each architectural component described above in terms of the information contained in the model.

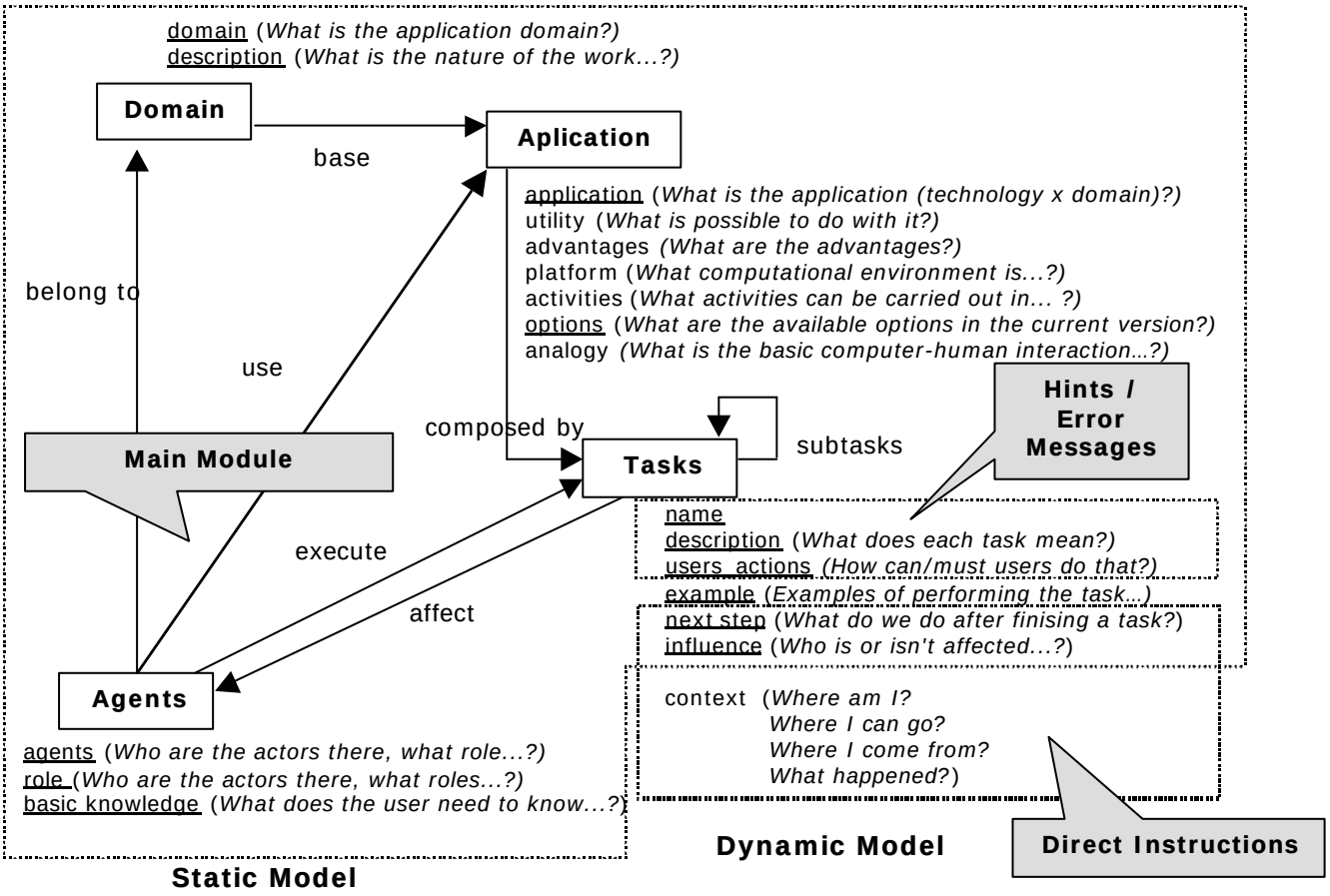

Figure 3.1: Components of Help Architecture x Proposed Model

4. USING THE PROPOSED ARCHITECTURE

The first help system created using the proposed architecture was associated to a workflow system developed for monitoring and supporting group activities certified by ISO 9000 standards. The application manages services rendered by a company and its final goal is to assure conformity to the certified norms and, thus, maintain the desired quality standards.

In this application, the proposed architecture was implemented as follows:

- Main Help Module: this component (figure 4.1) is activated when users select the <help> option from the application main menu. It is presented in a new window – separate from the application – and contains the application's user manual (available functions, technical specifications, form-filling instructions, etc.), as well as documentation corresponding to the institutional policies and internal procedures, and mechanisms for communication via e-mail with the support team. These functions are accessible via menus and hypertext links.

Figure 4.1: Main Help Module

- On-demand Hints on Featured Interactive Elements: these components (figure 4.2) are activated when the user clicks on some field label afforded as a hypertext link. This action opens a small pop-up window on top of the current application window, with an explanation about how to fill in the corresponding field in terms of syntax (for instance, *'the field must be filled with alphanumeric characters in the format XA234."*) and/or semantics (for instance, *"This date must be later than or equal to the creation date."*);

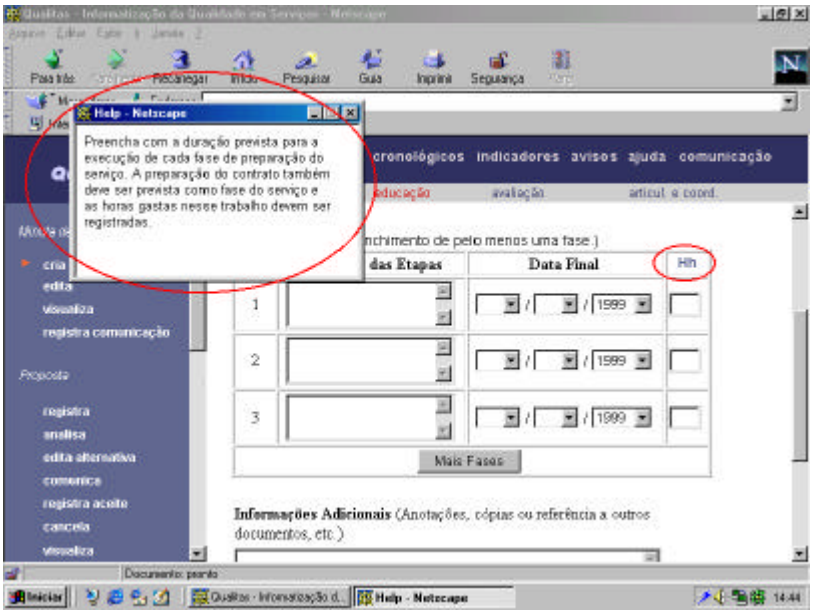

Figure 4.2: Field containing a hint, and the corresponding local help window

- Direct Instructions: this component (figure 4.3) is not activated by the user, but is instead made available by the application itself. It presents messages or comments from designer to user, about how the latter should proceed with the interaction in a given situation. Such comments are presented in the bottom part of the screen. Sample comments would be like *"To go on, click on the **analyze** item, within **proposal**, on the left frame."* or *"Please wait for the endorsement of the supervisor responsible for the area where the non-conformity has occurred. If you are authorized to endorse it and wish to do so, click on the **endorse** item, on the left frame."*). This module helps users to keep track of the current context and the sequence in which tasks should be performed.

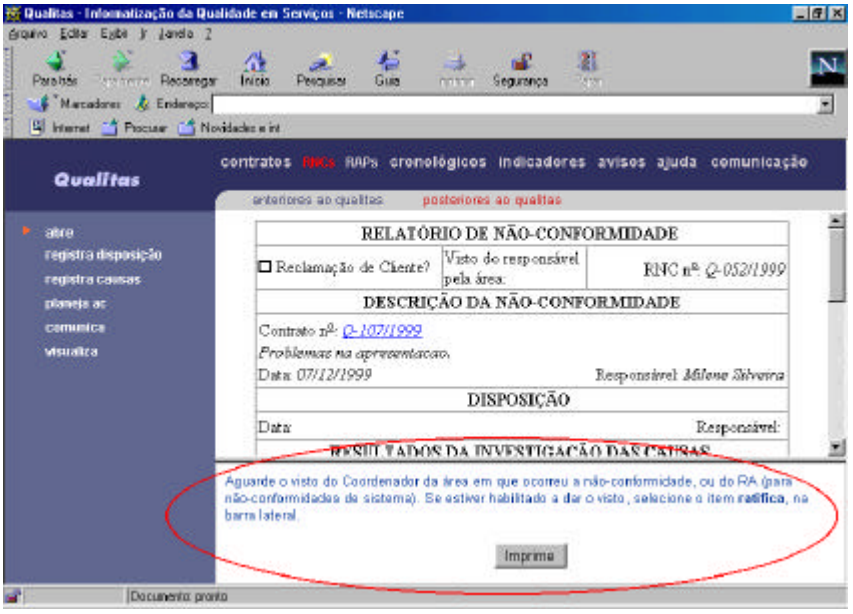

Figure 4.3: Direct Instructions

- Error Messages: this component (figure 4.4) presents messages from designers to users. They appear when some user's action is mistaken or incomplete. They describe what the user must do to perform the action correctly (for instance, *"The form was not filled correctly. Next to the incorrect fields are the instructions about how to correct their values."* or *"Type in a date later than or equal to the creation date."* or *"Type in an integer value."*).

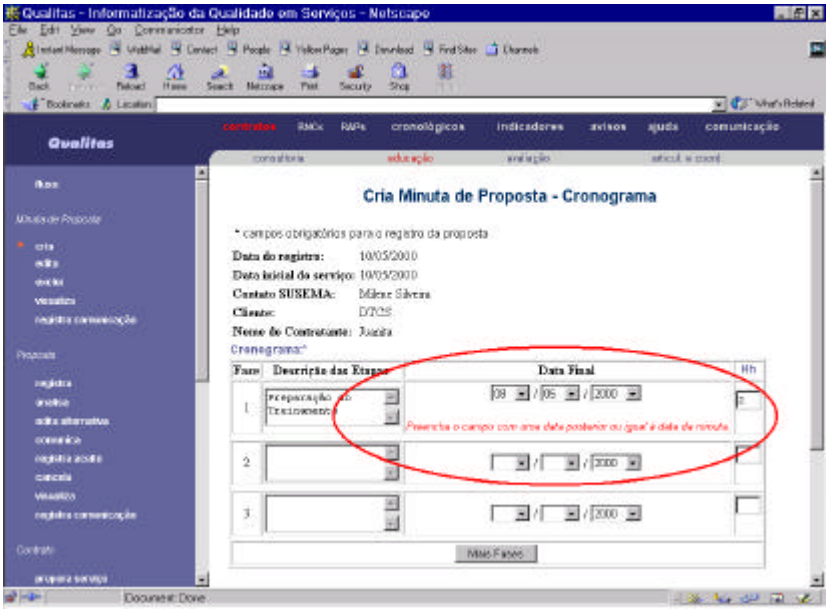

Figure 4.4: Error Message

5. FACILITATING THE GENERATION OF HELP MODULES

With the questions presented in section 2, we have developed a tool to support the design of help systems (namely a Help Editor). Our goal is to support the development of these systems throughout the application development process. While designing and developing an application, the designer should answer the aforementioned questions, and this knowledge will be the foundation for the final construction of the application's help components[1]. Firstly, the editor user (in this case, the application designer), answers a questionnaire for knowledge elicitation (figure 5.1).

[1] Answers to some of these questions may be extracted from the representations used by the designers during the development process, or imported directly from the tools used to support design.

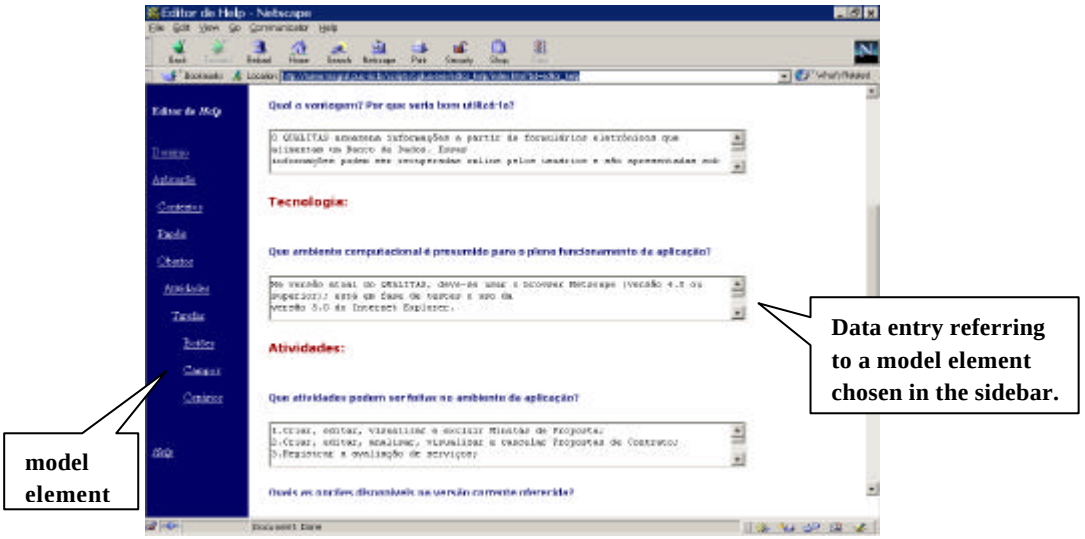

Figure 5.1: Help Editor

After having answered the questions, a preliminary version of the help system is generated (figure 5.2), and the user (the application designer) may change this information, improving it until the desired version is finally generated.

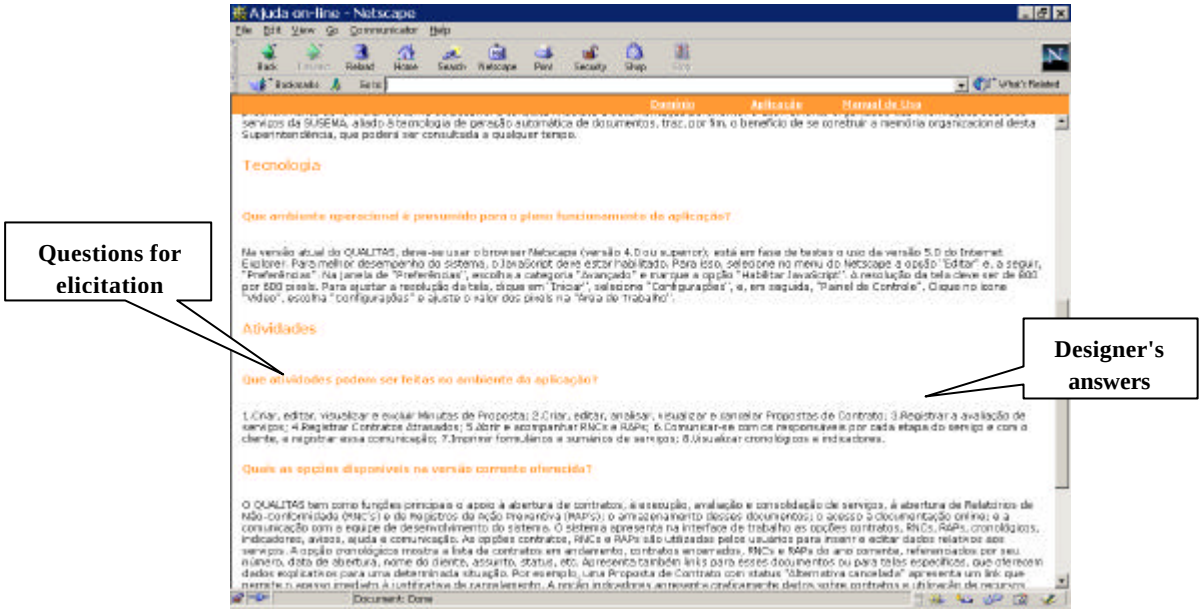

Figure 5.2: Preliminary Version of the Main Module

The purpose of this tool is to suggest, based on the designers' answers to the predefined questions, items that may compose the structure of the main module, the hints, the direct instructions and the error messages.

For example, for each field in the task (form), the following items are asked:

1. field type (text, numeric, date, list, checkbox, radio button or other);
2. mandatory or optional nature of the field;
3. description of the field, in terms of its syntax, semantics and/or additional information;
4. for each item of the description (syntax, semantics or additional info), there is a checkbox for the designer to inform whether he suggests that this field has a hint in the application or not.

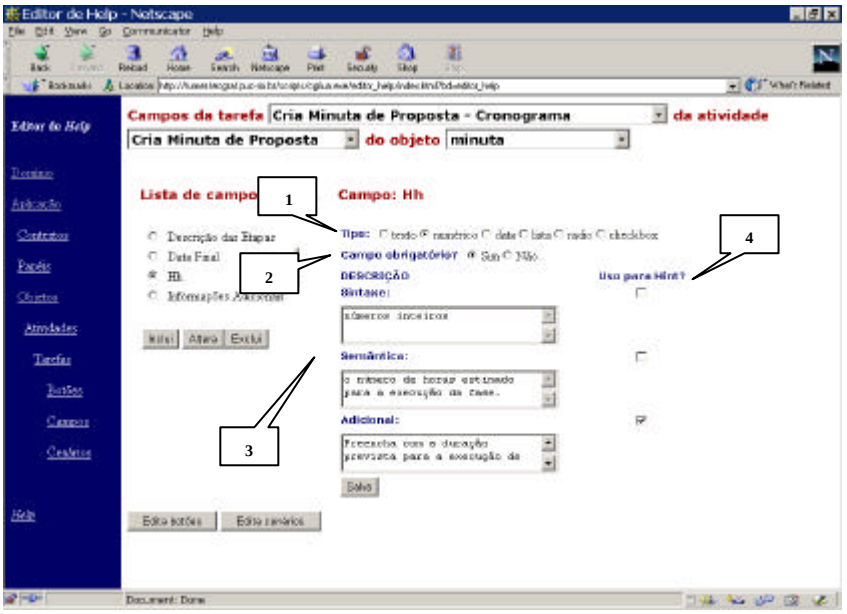

Figure 5.3: Editor form for typing in information about a task's fields

For example, in the figure above (figure 5.3) the field **Hh** is filled with (according to the options above):

1. field type: numeric
2. mandatory filling
3. syntax: integer numbers

   semantics: the number of hours estimated for the execution of the corresponding phase.

   additional info: type in the anticipated duration for the execution of the preparation of each service phase. The contract preparation also needs to be anticipated as a service phase and the time spent in this job may also be recorded.
4. The "additional information" is selected for providing a hint.

According to this information, the help about this field, provided when generating the items of the main module of the architecture proposed, is composed of the three items of the description (syntax, semantics and additional info):

Hh: Type in the number of hours estimated for the execution of this phase as an integer value.

Type in the anticipate duration for the execution of the preparation of each service phase. The contract preparation also needs to be anticipated as a service phase and the time spent in this job may also be recorded.

As for the hint, the suggested information is contained only in the additional item (the one that was checked for Hints Use):

Type in the anticipate duration for the execution of the preparation of each service phase. The contract preparation also needs to be anticipated as a service phase and the time spent in this job may also be recorded (this item was shown in figure 4.2). In the current version of the tool, hints are not placed directly in the application. Their corresponding explanations are generated in a format that can be read by the help mechanism, and their locations within the application are suggested, according to the proposed architecture.

5.1 The Designers' Help Editing Tool

The help system described in section 4 was created manually, before the development of our help editing tool. Having developed a preliminary version of the tool, we have used it for recreating the help system which inspired it, in order to validate the tool and acquire new insights for refining both the tool and the generated help components.

The second version of our tool was used to create the help system of an application during its late development stages (but not during its earlier design stages). Although some of the project's memory had been lost, some of it was recovered by means of the provided questions. Some questions even motivated new discussions among designers and developers about the desired and implemented interactions. Figure 5.4 presents a screen dump of the main help component.

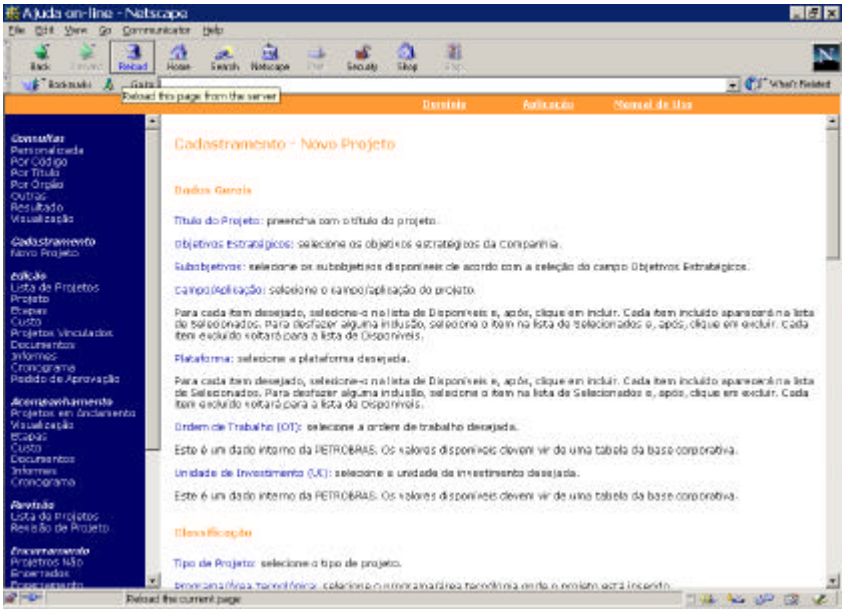

Figure 5.4: Main Help Module generated by the Help Editing Tool

The help editing tool considerably aided the team responsible for the development of the help systems within our web design and development team. The revisions and changes in the application were easily reflected on the help system. Moreover, the automatic generation of the help system from a consistent database eliminated errors and inconsistencies that were previously (before the tool development and use) found within the help components.

6. FINAL CONSIDERATIONS

The help systems shouldn't be seen either as useless or as a remedy for all intrinsic design problems [17]. In order to try to supply users' needs and promote their understanding of applications, we propose a model based on Semiotic Engineering, exploring the direct and indirect messages from designers to users.

Users access help when there is a breakdown in their understanding of an application. Since the designers have only an indirect participation in the interaction scenario, we try to maximize users' comprehension of the application from the designers' point-of-view. This point-of-view is precisely what we are trying to capture within a broader spectrum in the proposed model.

With this model, we intend to help users clarify their most frequent doubts (presented in section 1). The **main module** include the informative, procedural, motivational and investigative questions; the **hints** components, the interpretative and procedural ones; the **direct instructions**, navigational, choice and guidance.

The architecture built to support this model offers users a prompt answer to simple doubts, through the local help, and more detailed explanations in the main help module.

We also intend to promote a high degree of confidence in the process workflow and tasks sequence, through direct messages from designers to users about the next actions to be performed.

At its current version, the help editing tool has already proven to be valuable for designers and developers of web applications. Preliminary assessments with users of the generated help systems were promising, but there is still room for improvement.

At this moment, we are working to address the doubts yet unanswered in the current proposal (*What is happening now? What have I done?*). Further tests with the editor and help generator are necessary to reveal further opportunities for refinement and improvement.

We also intend to make communicability tests [12], asking users to perform complex tasks in which the help system becomes more necessary. These tests would provide us with more accurate information about our model and architecture, and would allow us to capture occasional flaws and point to adjustments.

Finally, we are experimenting with different realizations of the proposed architecture, to build a case base of solutions, with an assessment of their adequacy in different applications and domains, and for different classes of targeted users.

REFERENCES

1. Baecker, R.M. et al. (1995). *Readings in Human-Computer Interaction: toward the year 2000*. San Francisco: Morgan Kaufmann Publishers, Inc.
2. Chamberland, L. (1999). Componentization of HTML-Based Online Help. In *Proceedings of the Seventeenth Annual International Conference on Computer Documentation*, ACM Press, 165-168.
3. Chu-Carrol, J.; Carberry, S. (1998) Collaborative Response Generation in Planning Dialogues. *Computational Linguistics*, 3, 355-400.
4. de Souza, C.S. (1993) The Semiotic Engineering of User Interface Languages. *International Journal of Man-Machine Studies*, 39, 753-773.
5. Hansen, B.; Novick, D.G.; Sutton, S. (1996) Systematic Design of Spoken Prompts. In *Proceedings of CHI´96*, ACM Press, 157-164.
6. Johnson, W.L.; Erdem, A. (1997). A Interactive Explanation of Software Systems. *Automated Software Engineering*, 4, 53-75.
7. Kearsley, G. (1988). *Online Help Systems: design and implementation*. Norwood: Ablex Publishing Corporation.
8. Kedar, S.; Baudin, C.; Birnbaum, L.; Osgood, R.; Bareiss, R. (1993). Ask How it Works: An Interactive Intelligent Manual for Devices. In *Proceedings of the INTERACT'93 and CHI'93*, ACM Press, 171-172.
9. Leite, J.C. (1998). *Modelos e Formalismos para a Engenharia Semiótica de Interfaces de Usuário* (Doctoral Thesis). Rio de Janeiro: DI/PUC-Rio.

10. Marx, M.; Schmandt, C. (1996). MailCall: Message Presentation and a Navigation in a Nonvisual Environment. In *Proceedings of CHI´96,* ACM Press, 165-172.

11. Mittal, V.O.; Moore, J.D. (1995). Dynamic Generation of Follow on Question Menus: Facilitating Interactive Natural Language Dialogues. In *Proceedings of CHI´95*, ACM Press, 90-97.

12. Prates, R.O.; Barbosa, S.D.J.; de Souza, C.S. (2000) A Method for Evaluating the Communicability of User Interfaces. *ACM Interactions*, 31-38.

13. Priestley, M. (1998). Task Oriented or Task Disoriented: *Design*ing a Usable Help Web. In *Proceedings of SIGDOC 98*, ACM Press, 194-199.

14. Raskutti, B.; Zukerman, I. (1997). Generating Queries and Replies during Information-Seeking Interactions. *International Journal of Human-Computer Studies*, 47, 689-734.

15. Rintjema, L.; Warburton, K. (1998). Creating an HTML Help System for Web-based Products. In *Proceedings of the Sixteenth Annual International Conference on Computer Documentation,* ACM Press, 23-28.

16. Roesler, A.W.; McLellan, S.G. (1995). What Help Do Users Need? Taxonomies for On-line Information Needs & Access Methods. In *Proceedings of CHI´95*, ACM Press, 437-441.

17. Sellen, A.; Nicol, A. (1990). Building User-Centered On-line Help. In Laurel, B. *The Art of Human-Computer Interface Design*. Reading: Addison-Wesley.

18. Silveira, M.S.; Barbosa, S.D.J.; de Souza, C.S. (2000). Modelo e Arquitetura de Help Online. In: *Proceedings of IHC2000* (to be published).

19. Silveira, M.S.; Monteiro, C.C.O.; Barbosa, S.D.J.; de Souza, C.S. (2000). The Role of Designer-Generated Scenarios in Developing Web Applications and their Help Systems. *Série Monografias em Ciência da Computação* (MCC09/00). Rio de Janeiro: DI/PUC-Rio.

20. Sleeter, M.E. (1996). OpenDoc - Building Online Help for a Component-Oriented Architecture. In *Proceedings of SIGDOC 96,* 87-94.

21. van der Veer, G.C.; van Welie, M. (1999). Groupware Task Analysis. *Tutorial in CHI'99*. Available at http://www.cs.vu.nl/~martijn/gta/.