Automatic Ergonomic Evaluation : What are the Limits ?

Christelle Farenc[1], Véronique Liberati[2], and Marie-France Barthet[3]

[1,3]Laboratory for Information Science, Université Toulouse I, Place Anatole France, F-31042 Toulouse Cedex, France
Phone: +33-61.633{588, 603} – Fax: +33-61.63.37.98
E-mail : {farenc, barthet} @cict.fr - http://lis.univ-tlse1.fr/~{farenc, barthet}
[2]Direction Recherche et Développement, Service de Recherche Technique de la Poste, La Poste, 10 rue de l'Ile Mabon, F-44063 Nantes Cedex 02, France
Phone: +33-40.69.96.89 – Fax: +33-40.89.60.00
E-mail: liberati@srt-poste.fr

Abstract

There are currently automatic evaluation tools the purpose of which is to implement the knowledge gained by ergonomics experts, with a view to enabling non-experts and particularly user interface designers themselves to carry out this evaluation. Capturing the description of the User Interface to evaluate automatically this interface is one of the main objectives of these automatic evaluation tools. The question that comes to mind is knowing the limits of these tools and just how far evaluation computerisation can go. This article presents a study whose aim is to precise the qualitative and quantitative aspects of those limits.

Keywords

Evaluation, ergonomic rules, knowledge base, user interface.

Introduction

Current software development methods tend increasingly to take the user into account very early on by integrating, from the design phase on, ergonomic guidelines aimed at ensuring easy use and functional adequation. Indeed, most development tools and UIMS implement a certain number of these guidelines. However, this number remains limited and the tools are rather permissive.

Furthermore, although guides exist which bring together these recommendations, there are many of them and they are complex for developers to implement. Consequently, a posterior ergonomic testing is still necessary to confirm the ergonomic quality of interfaces produced.

There are currently automatic evaluation tools the purpose of which is to implement the knowledge gained by ergonomics experts, with a view to enabling non-experts and particularly computer scientists themselves to carry out this evaluation.

The question that comes to mind is knowing the limits of these tools and just how far evaluation computerisation can go. Indeed, the principle of these tools rests on the comparison between observed values, i.e. the description of the interface, and reference values contained in the ergonomic guidelines. An automatic evaluation tool must therefore be able to obtain the description of the interface without the help of a human operator.

This article presents a study aimed at defining, from rules implemented in the ERGOVAL [Barthet94] an ergonomic evaluation aid tool, those rules that it is possible to computerise, and those that cannot be computerised, because they require information that can only be described by a human operator.

1 The Problem

Evaluation as offered by automatic evaluation tools falls within the framework of analytical methods and is based on formal interface quality models [Senach90].

The principle behind analytical methods is the interlinking of a standard set of attributes of the object under evaluation using a measurement scale which integrates reference values.

Formal interface quality models seek to identify measurable qualities characterising the requirements that a user-friendly interface must satisfy (consistency, readability,...). They elaborate abstract representations of the object under evaluation that make it possible to predict user performances. These models are less interested in what the subject has to do with the device to achieve his task, than in the actual structure of the interface being used.

In contrast with empirical evaluation with testing it out on end users, analytic evaluation is based on HCI models and on ergonomic criteria and recommendations.

In the life cycle of a software program, this evaluation can be done very early on, from the specification phase. It relies on a structured knowledge base implemented in an expert system. The evaluation method is represented in figure 1.

Work carried out within the framework of the ERGOVAL design, in particular the production of a breadboard model to validate the knowledge base, have underscored the importance and complexity of the interaction between the tool user and the knowledge base. Indeed, with this model, many items of information (values of the attributes of graphical objects and contexts where rules come into force) have to be described by the user for evaluation of the interface ; this description is very cumbersome for the user.

Before coming to a precise definition of the modes of tool/user interaction, it therefore seems important to reduce the amount of information to be described by the user.

This is why the study presented in this article has centred on the computerisation of the description of the interface and more precisely, the "Specify" action in the evaluation process. In the rest of the article, we will use the term "computerise" in this precise meaning.

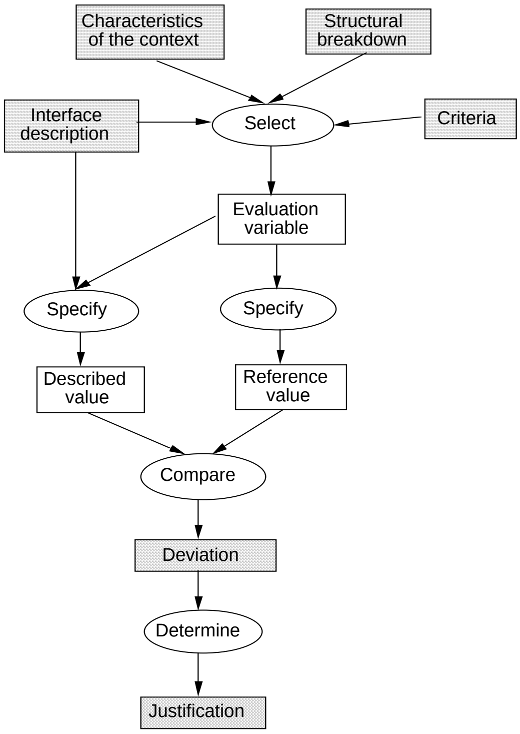

Figure 1. Diagram of the evaluation method

2 Automatic Evaluation Tools

The computerisation of the interface description has been envisaged in a variety of ways in different automatic evaluation tools: KRI/AG (Knowledge-based Review of user Interfaces) [Löwgren92], SYNOP [Kolski91], CHIMES (Computer Human Interaction Models) [Jiang92]. The use of these three tools requires at least two of the three following principles:

- a specific format for interface description files;
- the interface developed in the MOTIF environment;
- restricting oneself to presentation aspects.

Concerning this study, the automatic evaluation aims to be applied to any specific software or software package developed in the Windows environment, regardless of the development tool used. Moreover, the considered ergonomic rules are not strictly limited to the presentation aspects.

In view of this aim, the problem with automatic retrieval of the interface description is more complex than for the three tools. Indeed, unlike tools that have restricted their evaluation to a particular file format (KRI/AG, SYNOP, CHIMES), and perhaps even to a development tool (SYNOP), ERGOVAL has to be capable of recovering the interface description for any software written under Windows.

In contrast with the MOTIF environment, each Windows development tool uses a specific format for its source files. Recovering data from these files means that ERGOVAL must be capable of decoding every type of Windows source file in existence, and not just for the tools quoted here. In addition, when evaluating a software package, the source files are not available, which means that the evaluation can only be carried out from one of the application's run files.

Finally, unlike the three tools mentioned, ERGOVAL includes in its knowledge base rules on the semantic and pragmatic levels that require information which a priori does not appear in the interface description files. In this way, whereas the method followed in designing the KRI/AG tool involved integrating computational rules only, the method followed for ERGOVAL is a completely different one. The aim of ERGOVAL has been to achieve as exhaustive an evaluation as possible, which involved integrating into the tool a large number of ergonomic rules that were representative of ergonomic knowledge as regards interfaces. The next step of the ERGOVAL design then involved seeking as far as possible to computerise this evaluation, in such a way as to avoid it becoming too unwieldy to use in proportion to the tool's value in terms of software design.

Technical requirements for utilisation of ERGOVAL and the scope of its field of application mean that problems posed by automatic retrieval of interface descriptions cannot be solved as easily as for the tools described. Before we even start searching for the specific technical means of recovering data on the interface, we may ask ourselves just how far such computerisation is going to be possible, and with regard to the set of ergonomic rules, what can we hope to evaluate with no user intervention.

The study presented below determines in the case of ERGOVAL the very lowest automatic evaluation level that it is possible to provide and in what conditions such a level of automation can be improved.

3 The Limits of Automatic Evaluation

The ergonomic recommendations integrated in the knowledge base come from literature [Smith86, Scapin93, Vanderdonckt94e]. These recommendations were selected in function of two principal criteria:

- a good level of accuracy;
- taking into account, in as representative a way as possible, the various elements involved in ergonomic expertise, namely : the diversity of objects involved ; lexical, syntactic, pragmatic, semantics levels ; and ergonomic design principles.

Moreover, the knowledge base was structured by organising the graphic objects: these objects are grouped into classes of objects all concerned by the same set of recommendations.

These recommendations are, for the most part, in a style guide written for Post Office designers : the graphic interface design guide MICE/D [MICE93]. As regards the pragmatic level, only those guidelines that did not require in-depth analysis of the task were incorporated.

The capability of computerising the recovery of information required to check the rules was not a decisive criterion in the first instance ; it should be remembered that the prime objective was rather evaluation quality.

The purpose of the survey shown hereinafter is to specify the automation limits of these ergonomic rules, whatever the automation methods implemented in a tool are.

In order to do this, the first step of this survey is to determine the minimum limit of the automation, i.e., the percentage of rules that can be easily automated.

It is considered that a rule can be easily automated, when all the information required to verify it is included in the source files.

These source files must include the interface description as data and not code, so that the automation can still be considered 'easy'. Example : « Any dialogue box or window should have a title ». This rule can easily automated, since any dialogue boxes or windows as well as their title appear in the source files. This rule is therefore easy to verify.

After this, rules are classified into two classes: rules that can be automated with source files and rules that can not be automated with source files.

The second step is to determine the maximum limit of the automation, i.e. the percentage of ergonomic rules that can be automated, even if the methods to be implemented for retrieving the information required for running these rules, are complex.

It is considered that a rule can be automated, whatever the implemented methods are, when all of the information required to verify it, can be found in the system. Example : « Any non-accessible action must be greyed ». At a « t » moment, it is virtually possible to know all of the actions that can not be accessed. It also possible to know whether the object of this action is greyed or not. All of the information is in the system, therefore the rule can be automated.

After this, rules are classified into two classes: Rules that require information automatically retrievable whatever the implemented methods are, and rules that require information not automatically retrievable whatever the implemented methods are.

In order to determine why certain rules can not be automated, it is necessary to separate each of these classes in two sub-classes: Rules that require information related to items included in the application and rules that require information related to items not included in the application. For both classes, rules are also classified based on the type of information required for running them. A summary of these various classifications is shown in figure 2.

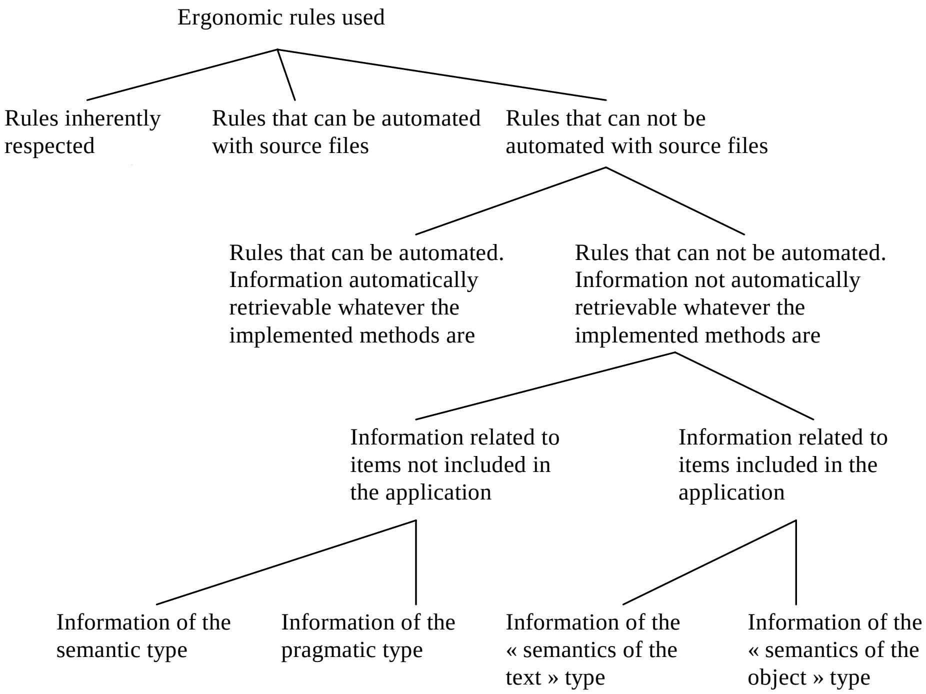

Figure 2. Classification of ergonomic rules.

3.1 Minimal Automatic Evaluation

For software programs developed under Windows, it is sometimes possible to recover information about the interface in text format from "rc" resource files. Such information is mainly to do with the static description of the interface and is very simple to recover by means of tools such as the [Borland91].

Ergonomic rules integrated into ERGOVAL have been analysed in such a way as to count the percentage of rules that are concerned by the automatic recovery of data from these resource files.

For all ergonomic rules contained in the ERGOVAL rule base, rules have been placed in the following categories:

- **rules directly obeyed** by construction, knowing that this figure may vary depending on the development tool used to design the interface. For the purposes of this study, it has been taken as read that the development tool used was the Resource Workshop;

- **rules requiring automatically recoverable information** because they are contained in the resource files;
- **rules requiring not automatically recoverable information** because it does not appear in the resource files.

The rules were also divided into two main categories, rules that focus on static interface presentation and those that focus on interface or system behaviour (dynamic presentation, data flow,...). The purpose of this division was to check that most of the information contained in the resource files does indeed concern static interface presentation, but also to determine whether the recovery of resource files is sufficient to evaluate static interface presentation as a whole. Table 1 shows how the rules are distributed for the above two categories in function of the aforementioned classes.

	Rules /presentation	Rules /behaviour	Total
Rules inherently respected (1)	28	64	93 (22.9%)
Rules that can be automated with source files (2)	82	2	84 (20.6%)
Rules that can not be automated with source files (3)	161	69	230 (56.5%)
Total	271	135	406

Table 1. Summary chart of the computerisation of ergonomic rules from resource files

Examples of class (1) rules:

- as regards static presentation: "Labels for push buttons must be centred";
- as regards behaviour: "In a menu bar, a drop-down menu, a cascading menu, and a system menu, the cursor must run automatically from the last option to the first".

Examples of class (2) rules:

- as regards static presentation: "All boxes and windows must have a title";
- as regards behaviour: "All boxes and windows must be movable".

Examples of class (3) rules:

- as regards static presentation: "For any input field, if there are any acceptable values, such values must be displayed";
- as regards behaviour: "If the system's response time is of between two and five seconds, a wait pointer must be displayed".

Thus, it emerges from this table that the majority of recoverable rules at resource file level are indeed rules of static presentation: 82 out of 84. On the other hand, recoverable information in these files is not sufficient to evaluate static presentation as a whole, for whilst this recovery ensures the checking of 82 static presentation rules, it does not do so for 161 others. These files do not therefore provide a precise enough description of the interface to ensure that a fair percentage of ergonomic rules governing the static presentation of the interface are checked.

Lastly, if we consider the 22.9% of rules that are obeyed automatically and the 20.6% of rules that can be executed automatically on the basis of data from resource files, this leaves 56.5% of rules that cannot be evaluated on the basis of the content of these files. If we are to be able to claim to have a valid automatic method of evaluation, it seems we have to reduce this percentage of outstanding rules and therefore achieve automatic retrieval of further data on the interface.

The following paragraph presents an analysis of information that does not figure in resource files and that is required for the execution of these remaining rules.

The aim of this analysis is to define:

- whether it is possible to increase the number of executable rules through automatic retrieval of the data required to run them;
- what the limits are to the computerisation of an evaluation process based on ergonomic rules by determining the number of rules dependent on data that is not automatically recoverable.

3.2 Maximum Automatic Evaluation

The aim is to determine the percentage of ergonomic rules that it will be possible to evaluate automatically, whatever the technical resources used, as opposed to an « easy » automation.

The rules considered are the 230 rules (56,5 %) that can not be automated with source files. One distinction can be established depending on whether the rules:

- **require information that focuses on elements that are in the application.**

For example, for the rule "If there are acceptable values within the system, then they must be displayed", the information "there are acceptable values within the system" refers to the element "acceptable value", which is within the application, and to be more precise in its functional core.

- **require information on elements that are not contained in the application.**

For example, for the rule "When a selection is displayed by default by the application, this selection must be relevant for the user", the information "this selection is relevant for the user" refers to the element "relevance for the user", which is not contained in the application.

As regards the former category of rules, it is clear that if an element does not exist in the application, no information that needs to be attached to this element can be found in it either. Therefore since this information is not contained in the application, it cannot be automatically retrieved from it.

On the other hand, for the former category of rules, information on these elements may be found in the application, but it must be noted that this is not systematic.

The following table presents the distribution of these rules with respect to the information source, in order to distinguish between those which are potentially computational and those which, although having a bearing on elements contained in the application, can never be automated.

Rules	Information automatically retrievable	Information not automatically retrievable	Total
Elements in the application	140	62 (1)	164
Elements not in the application	0	66 (2)	66
Total	140 (60.87%)	90 (39.13%)	230

Table 2. Summary chart of the grading of rules by source of the information required for their verification.

It should be noted that the sum of (1) and (2) does not correspond to the total number of the rules requiring non recoverable information, because 38 rules require information on elements from both within and outside the application. These 38 rules are therefore posted twice at the level of Table 2. One example of this type of rule : "if there are codes in an input field literal, then these codes must be known to the user". The context "if there are codes" is linked to the meaning of the displayed text, and the displayed text is « written » in the application. On the other hand, the action part "then these codes must be known to the user" relates to the user and is not in the application.

It is therefore potentially possible to recover from the system the data required in order to verify 60.87% of rules remaining after utilisation of the resource files, that is to say 34,49% of the total number of rules included in the knowledge base. If we refer to the Seeheim model [Pfaff85], data to be found within the application is either at functional core level or at dialogue controller level, or again at presentation component level. Whilst it is relatively easy to recover data contained in the presentation component, on the other hand recovering information contained in the functional core or the dialogue controller can prove extremely complex, and all the more so in the case of applications developed with different tools.

As regards the 39.13% of remaining rules (table 2), that is to say 22 % of the total number of rules included in the knowledge base, these can only be run if the data is supplied to the system by a human operator.

These results demonstrate that ergonomic rule verification cannot be made totally automatic without the intervention of a human operator.

The following paragraph presents an analysis of the nature of all the data that cannot be recovered automatically in order to explain why this is so, and to determine the type of information and knowledge that will have to be supplied by a human operator.

3.3 Data that is not Automatically Recoverable

Qualitative analysis of this data has shown that it is of two kinds:

- **pragmatic,**

For example: for the rule "When the data input, selected or restored consists of units of measure, then the unit displayed must be the one commonly used by the user". The information "commonly used by the user" is information of the pragmatic type, one that varies according to the task for which the software is being used.

- **semantic.**

For example: for the rule "If a message signals an error, it must contain the cause of the error", the information "must contain the cause of the error" is information of the semantic type.

In the case where the necessary information focuses on elements from outside the application (cf. preceding paragraph), rule distribution is as follows:

Semantic	21
Pragmatic	45
Total	66

Table 3. Summary chart indicating the nature of information about elements outside the system.

- An example of a rule requiring semantic information: "If a literal or title contains an abbreviation, such an abbreviation must comply with abbreviation norms."
- An example of a rule requiring pragmatic information: "If a literal or title contains a code, the meaning of this code must be known to the user".

In the case where the necessary information focuses on elements to be found within the application, (cf. previous paragraph), it is interesting to distinguish two kinds of information of the semantic type:

- **information linked to the semantics of the text displayed**, for example: "if there are any codes in a literal";
- **information linked to the semantics of the graphical objects** of the interface, for example: "If there are any acceptable values in the system, are they displayed ?". The context corresponds to an item of information that is present within the application, but to test if the acceptable values are displayed, we need to know the "meaning" of the graphical objects displayed on the screen.

The rules are distributed as reported in table 4.

Semantics of displayed text	39
Semantics of graphical objects	23
Total	62

Table 4. Summary chart of the source of information within the system.

- An example of a rule requiring information about the semantics of the text displayed: "If a text message signals an error, it must contain an explanation of the cause of that error".
- An example of a rule requiring information on the semantics of the graphical objects: "A list box literal must be presented above the object that it designates". In this case, at interface level, there is a text restore field that is located "near" the list box, but the system does not know whether or not the purpose of this restore field is to label the list box.

Note that in this second instance, no rule requiring information of a pragmatic nature was found.

Although the results presented in this paragraph are not unexpected, it is nonetheless important to underscore the fact that in addition to the task-linked information, any human operator will also have to be able to provide the tool with information as to the meaning of objects within the interface.

Conclusion

This article attempts to indicate the limits to the computerisation of ergonomic rules in a graphic interface evaluation tool, that is to say, just how far it is possible to recover automatically the data required to execute ergonomic rules without the need for intervention by a human operator.

In other words, it seemed interesting to define more precisely what such limits in terms of quantity and quality involved.

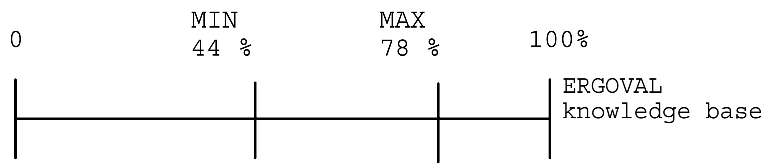

Figure 2. Limits to computerisation of ergonomic rules

Knowing that the rules contained in ERGOVAL represent 100%, 44% of these rules are automatically verifiable using the resource files. This percentage represents the minimum number of rules that can be automated inasmuch as recovering resource files is the easiest thing to do.

Furthermore, the maximum percentage of ergonomic rules that can be incorporated into a totally automated evaluation is 78%.

The effective limit that computerisation of rules is bound to reach falls somewhere between these maximum and minimum figures. This limit depends on the aims set upon the tool: evaluation benchmarks, cost, modes of operator/tool cooperation.

Finally, 32% of these rules of necessity require the intervention of a human operator.

It appears that the information needing to be described by a human operator is mainly of a semantic and pragmatic type. Computerisation will therefore be dealing primarily with lexical and syntactic rules. Automatic evaluation remains however a useful preliminary to tests with users, because it makes it possible to correct design errors that penalise these tests. This is because users are liable to focus their attention on such errors rather than on the functional adequacy that these evaluations are designed to confirm.

Research work has demonstrated that to be reliable, i.e. to identify 80% of design errors, an analytic evaluation must be carried out by 3 experts [Pollier91]. If this result deserves a more thorough study, it is true that an automatic tool has the advantage of systematising the verification of ergonomic rules for all the graphical objects. In fact, a greater number of design errors can be identified.

Moreover, an automatic tool shall help the designer to build the ergonomic recommendations into his product thought successive self-evaluation.

In conclusion, to increase the number of rules evaluated, and hence the effectiveness of the tool itself, requires that a human operator be integrated into the evaluation process. It is therefore necessary to achieve an aid tool capable of both executing some rules automatically and at the same time cooperating with a human operator to execute others.

References

[Barthet94] Barthet, M.-F., Liberati, V., Ponamale, M., *ERGOVAL - A Software User Interface Tool*, in Proceedings of the 12th Triennal Conference of International Ergonomics Association IEA'94 (Toronto, 15-19 August 1994), Vol. 4, Human Factors Association of Canada, Toronto, 1994, pp. 428-431.

[Borland91] Resource Workshop Editor V1.0, *Reference* Manual, Borland International, 1991.

[Jiang92] Jiang, J., Murphy, E., Bailin, S., Truszkowski W., Szczur M., *Prototyping a Knowledge Based Compliance Checker for User-Interface Evaluation in Motif Development Environnements*, in Proceedings of Second Annual International Motif Users Meeting MOTIF'92, Open Systems, Bethesda, 1992, pp. 258-268.

[Kolski91a] Kolski C., Moussa F., *Une approche d'intégration de connaissances ergonomiques dans un atelier logiciel de création d'interfaces pour le contrôle de procédés*, in Proceedings of « Le Génie Logiciel et ses Applications » (Toulouse, 9-13 December 1991), pp 181-194.

[Löwgren92] Löwgren, J., Nordqvist, T., *Knowledge-Based Evaluation as Design Support for Graphical User Interfaces*, in [CHI92], pp. 181-188.

[Mice93] *Guide de conception des interfaces graphiques*, MICE - Sous-groupe D Interfaces Utilisateurs, La Poste, December 1993.

[Pfaff85] Pfaff, G. (Ed.), Proceedings of Eurographics seminar (November 1983), *Tutorial and perspectives in computer graphics; user interface management system*, Springer-Verlag, Berlin, 1985.

[Pollier91] Pollier, A., *Evaluation d'une interface par des ergonomes : Diagnostics et Stratégies*, Research report INRIA No. 1391, February 1991.

[Scapin93] Scapin D.L., Bastien, J.M, *Ergonomics Criteria for the Evaluation of Human-Computer Interfaces*, Report INRIA No. 156, June 1993.

[Senach90] Senach, B., *Evaluation ergonomique des interfaces homme-machine: une revue de la litterature*, Report INRIA No. 1180, March 1990.

[Smith86] Smith, S.L., Mosier, J.N., *Design Guidelines for the User Interface Software,* Technical Report ESD-TR-86-278 (NTIS No. AD A177198), U.S. Air Force Electronic Systems Division, Hanscom Air Force Base, Massachusetts, 1986.

[Vanderdonckt94e] Vanderdonckt, J., *Guide ergonomique des interfaces homme-machine*, Presses Universitaires de Namur, Namur, 1994.