Ören, T.I. and Çetin, S. (1999). "Quality Criteria for User/System Interfaces". RTO Meeting Proceedings 38 – Modelling and Analysis of Command and Control (Papers presented at the Symposium of the RTO Studies, Analysis and Simulation (SAS) Panel held at Issy les Moulineaux, France, 12-14 January, 1999). Published June 1999. pp. 18-1 – 18-8.

Quality Criteria for User/System Interfaces

Tuncer I. Ören, Selim Çetin

Tübitak, Marmara Research Center
Information Technologies Research Institute
Gebze, Kocaeli, Turkey
{tuncer, selim}@ mam.gov.tr

ABSTRACT

User/system interfaces are essential components of any interactive software, including command and control software. As part of overall quality of any interactive software, quality issues of interfaces are very important. A set of 27 interface quality criteria for user/system interfaces are presented in four groups which are convenience (usability), communicativeness, reliability and evolvability. The *convenience* criteria are related with: conveniences of the language, terminology, metaphor and the inputs; and functionality, simplicity, consistency, minimum memory load, navigability and least training. *Communicativeness* criteria cover: informativeness, guidance, perceptiveness, explanation ability, expressiveness, esthetic/cultural acceptance and types of user/system relationship. *Reliability* criteria are concerned with: error prevention, error tolerance, caution, predictability and access reliability. *Evolvability* criteria cover: adaptability, customizability, learning ability, maintainability and portability. The criteria can be used for evaluation and comparison of existing interfaces as well as for the design and implementation of new ones. Four tables with appropriate questions are provided to systematize the evaluations.

1. USER SYSTEM INTERFACES

1.1 INTRODUCTION

User/system interfaces are important components of all software systems (Galitz, 1996). One can consider user/system interfaces from several points of view: Goodwin (1989) presents interface issues for C programmers. Marcus (1992) presents detailed descriptions and comparisons of Macintosh, Nextstep, Open Look, Motif system, Microsoft Windows, and OS/2 Presentation Manager. Moreover, he provides a comparative product-specific terminology used in the systems listed above. Molich and Nielsen (1990) concentrate on the essence of the user/system interface design problem and provide a list of suggestions for good designs. Horton (1990) considers design and implementation of on-line documentation and provides answers to the fundamental questions such as "what makes a good dialog?" Lee (1993) concentrates on object-oriented graphical user interfaces. In the late 1990s, functionalities and appearances of Internet interfaces are of importance. The Internet brought also new dimensions in interfaces such as Internet ethics (Cheong 1996). Sullivan and Tyler (1991) explore intelligent user interfaces. This topic is maturing both in the applications of agent technology as well as interactive voice technology to user/system interfaces (IUI '93, IUI '97, IUI '98, and IUI '99) and will necessitate refinement of the quality criteria for user/system interfaces.

An interactive software can be considered in two parts: a solution engine and a user/system interface (Figure 1). The solution engine part is used to process the input and to produce solutions for given problems.

The user/system interface is used to communicate with a user in interactive systems. The user/system interface can be divided into two sections: A front-end interface and a back-end interface.

The front-end interface is used to enter inputs.

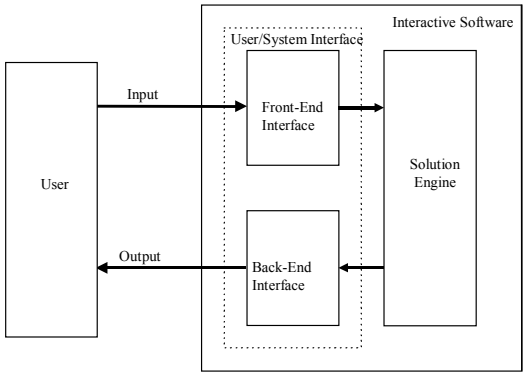

Figure 1. The elements of an interactive software

The back-end interface is used to get, process, and/or display outputs produced in the solution engine. For example, the back-end interface can display the data which is produced by the solution engine in graphic format for virtual or augmented reality applications.

1.2 THE NECESSITY OF USER SYSTEM INTERFACES

The term user-friendliness is overloaded in referring to desirable aspects of user/system interfaces. Implementers, vendors and users of software tools and

environments need a set of well-defined quality criteria for user/system interfaces. The criteria can be used to design and implement good interfaces as well as to evaluate, compare, and/or provide a basis to improve existing ones. Furthermore, a set of well-defined quality criteria allows one to choose explicitly the characteristics of interfaces that one would like to have.

2. QUALITY CRITERIA FOR USER/SYSTEM INTERFACES

Twenty seven quality criteria are identified for user/system interfaces. They are grouped in four areas, namely, convenience (or usability), communicativeness, reliability and evolvability.

2.1 CONVENIENCE (USABILITY) CRITERIA

Some advantages of satisfying convenience or usability criteria are as follows:

- Users can use the computer (or more specifically, the software) without needing additional documentation.
- Necessary information can be displayed on the screen when needed. In this way the memory load of the user can be minimized.
- Definitions of problems and evaluations the results can be done easily.
- Users can use the terminology of the application area.

The convenience criteria are listed in the sequel:

1. Convenience of the language (1.1)
2. Convenience of the terminology (1.2)
3. Convenience of the metaphor (1.3)
4. Convenience of the inputs (1.4)
5. Functionality (1.5)
6. Simplicity (1.6)
7. Consistency (1.7)
8. Minimum memory load (1.8)
9. Navigability (1.9)
10. Least training (1.10)

2.1.1 Convenience of the language (1.1)

The natural language used in an interface should ideally be the native language of a user or at least it should not hinder the proper use of the software.

2.1.2 Convenience of the Terminology (1.2)

An interface should be based on the application domain's terminology. The terms should not be confusing.

2.1.3 Convenience of the Metaphor (1.3)

The interface metaphor should be most appropriate (i.e., natural) to the application domain. Examples: desktop, book, index, card, form, calendar, agenda, instrument panel, warning or traffic lights, map, office, supermarket and layout (factory, theater, airplane). A door is an example of a 3-D metaphor in virtual reality.

2.1.4 Convenience of the Inputs (1.4)

An interface should to able to accept the types of inputs most appropriate (i.e., natural) for the application. Examples (conventional): keyboards, pointing devices (mouse, lightpen, trackball, joystick), touch screens, touch pens. Examples (relatively new types): handwriting, dataglove, deictic input (gestures), haptic inputs (touch, pressure), eye gaze tracking, speech or voice, and multimodal input.

Deictic inputs provide flexibility in virtual and/or augmented reality. Eye gaze tracking is important in civilian as well as defense applications. Interactive voice technology, once more mature, can be the basis for applications and/or operating systems based on speech; thus allowing voice commands.

2.1.5 Functionality (1.5)

An interface should offer complete set of abilities to specify problems and to process, analyze, and present results. Therefore, the input functionalities can be as advanced as the computer-aided problem solving environments. The output functionalities can be graphically oriented as it is the case of virtual or augmented realities; they can also include statistical or reasoning abilities.

2.1.6 Simplicity (1.6)

An interface should not have unnecessary and distracting information. The displays should be as uniform as possible.

2.1.7 Consistency (1.7)

There should be no ambiguity to initiate an action in different parts of the interface.

2.1.8 Minimum Memory Load (1.8)

Users should not be obliged to remember information from one part of the interface to another. Users should not be obliged to memorize the instructions. Instructions to use the system should be visible (for example, through icons or pull down or pop up menus). If there is a sequence of activities to perform a task, they should be performed for the user or at least the sequence should be made clear to the user. For complex and/or routine tasks, software agents can and should be used to alleviate the workload of the user (Bradshaw, 1997).

2.1.9 Navigability (1.9)

Activities should be initiated as directly as possible. Navigation should be done with least movements. At

every state of the system, the user should know: how to cancel the current activity and how to exit the system as well as how to initiate necessary activities.

2.1.10 Least Training (1.10)

An interface should require least amount of training. Any needed training should be available as just-in-time learning facility. It is highly desirable to have a self-pace demo on the utilization of the system.

2.2 COMMUNICATIVENESS CRITERIA

Some advantages of satisfying communicativeness criteria are as follows:

- The functions of programs can be visualized.
- Users can obtain information about the software system directly from the system.
- The software systems can support different types of users.

The communicativeness criteria are listed in the sequel:

1. Informativeness (2.1)
2. Guidance (2.2)
3. Perceptiveness (2.3)
4. Explanation ability (2.4)
5. Expressiveness (2.5)
6. Esthetic/cultural acceptance (2.6)
7. Types of relationship (2.7)

2.2.1 Informativeness (2.1)

An advanced interface can and should be able to prompt several types of knowledge which may (or should) exist in the system:

- Knowledge that the interface is incrementally receiving from the user and/or other knowledge which exist in the system, (including a user profile that the system should, maintain.
- Knowledge that the interface (should be able to), deduce from the knowledge provided by the user.
- Knowledge about the methodology on which the system is based on (e.g., simulation methodology).
- Fundamental scientific and engineering knowledge.

This may necessitate agents or mobile agents directly accessing to the appropriate sources of knowledge (by on-line payment of a fee, if necessary).

- Knowledge about the application domain (defense, business, etc.).

Comments similar to the one given for the previous topic is also applicable here.

- Knowledge about the software system and how to use it (e.g., annotation of icons upon focus).

2.2.2 Guidance (2.2)

An interface should be able to guide the user in solving problems by providing:

- Alternatives,
- Examples (demonstrations), and
- Sample data (with the possibility to modify and save them).

2.2.3 Perceptiveness (2.3)

An interface should be able to observe the user:

- To perceive the intentions of the user
- To decide when to initiate an advice.

This features are implementable by software agents (Bradshaw, 1997).

2.2.4 Explanation Ability (2.4)

A back-end interface should be able:

- To provide explanations/justifications of the decisions taken by the system, and
- To explain the results or the solutions recommended by the system

This features are basically implementable by artificial intelligence techniques.

2.2.5 Expressiveness (2.5)

An interface should be able to provide necessary output modes warranted by an application. Examples: direct feeding of actuator devices, voice annotation, on-line video help, multimedia outputs where text, picture (from files or rendered), animations, and video may co-exist.

2.2.6 Esthetic/Cultural Acceptance (2.6)

Shape, size, location, color, and movement of displayed objects; sound of audio signals and messages; and their relations to other objects should be consistent with universal (as well as local) cultural and esthetic norms**.**

2.2.7 Types of Relationship (2.7)

Patronizing, informal, and insulting tone should not be used. Human-like entities (including avatars) should be used when warranted and not just as technological curiosities.

2.3 RELIABILITY CRITERIA

Some advantages of satisfying reliability criteria are as follows:

- Possibility of preventing some types of errors in new programs and
- The avoidance of some types of errors of legacy (existing) programs.

The reliability criteria are listed in the sequel:

1. Error prevention (3.1)
2. Error tolerance (3.2)
3. Caution (3.3)
4. Predictability (3.4)
5. Access reliability (3.5)

2.3.1 Error Prevention (3.1)

A front-end interface should screen the inputs to prevent errors.

A back-end interface should filter the outputs to intercept unacceptable (and possible dangerous) outputs. For example, regardless how the software is implemented in a radiation therapy device (e.g., Therac), lethal radiation dosage can easily be avoided by an appropriate feature of the back-end interface.

2.3.2 Error Tolerance (3.2)

A front-end interface should tolerate errors (with confirmation):

- In case of erroneous activation of a task, the user should be able to exit without any side effect.
- An interface should encourage trial-and-error learning without causing frustration.

2.3.3 Caution (3.3)

An interface should:

- Confirm irreversible actions and
- Include an undo command (preferably several levels)

2.3.4 Predictability (3.4)

An interface should do what its users would expect it to do.

2.3.5 Access Reliability (3.5)

An interface should be able to monitor access to the system and report it.

This feature should be in addition to appropriate detection tools of viruses and trojans and fire walls (Scientific American, 1998).

2.4 EVOLVABILITY CRITERIA

Some advantages of satisfying evolvability criteria are as follows:

- A good interface can be changed easily; hence its maintenance is easy.
- A good interface can be adapted to the needs of a user.

The evolvability criteria are listed in the sequel:

1. Adaptability (4.1)
2. Customizability (4.2)
3. Learning ability (4.3)
4. Maintainability (4.4)
5. Portability (4.5)

2.4.1 Adaptability (4.1)

An interface should provide information needed by different categories of users such as: experts, transfer users, occasional users, and novices.

2.4.2 Customizability (4.2)

One should be able to easily tailor an interface to suit different:

- Nationalities and/or
- Preferences (for example, tailoring toolbars).

The natural language used in the interface should be easily and correctly understood by users. This may require multilingual abilities in an interface.

2.4.3 Learning Ability (4.3)

An interface should be able to remember the usage of the system by a user and should provide the relevant knowledge to enhance problem solving abilities of the user.

2.4.4 Maintainability (4.4)

The maintenance of the interface should be easy.

2.4.5 Portability (4.5)

A good interface should be portable to different platforms.

3. A SYSTEMATIC APPROACH FOR EVALUATION AND DESIGN

To ease evaluation of a current interface a systematic approach is used; for this purpose, one needs a table of questions (i.e., an assessment table) for each of the four areas, namely for convenience (usability), communicativeness, reliability and evolvability. Tables 1-4 are supplied as the assessment tables with referances to these four groups of criteria.

For each criterion, there is at least one question. For each question, the answer may be either yes (a desirable feature) or no. In the case of lack of a desirable feature, there are three possibilities according to the severity of the feature: (1) the interface is still acceptable, (2) the interface should be improved and (3) the interface should be rejected. Additional comment area can be used for specific information.

4. CONCLUSION

As important components of any software system, interfaces require particular care. Therefore, quality of user/system interfaces is of paramount importance. A set of criteria and a systematic approach is offered for user/system interfaces. The same criteria can be used as

a basis for proper design of new interfaces or to evaluate and hence to improve existing ones.

This study can be enhanced in the following ways:

1. Enhance (add, delete, modify, annotate) the quality criteria.
2. Suggest better grouping for the quality criteria.
3. Enhance (add, delete, modify, annotate) the questions in the evaluation tables.
4. Develop a software for the quality assessment of user/system interfaces.
5. Take into account any feedback that may come from the readers.

5. ACKNOWLEDMENT

The current version of the study is part of a project developed for the Turkish Armed Forces. Two preliminary versions were presented in Turkey, one in English (Ören, 1997) and one in Turkish (Ören et al., 1998).

REFERENCES

Bradshaw, J. (ed.) (1997). Software Agents: AAAI Press, Los Altos, CA.

Cheong, F.C. (1996). Internet Agents: Spiders, Wanderers, Brokers, and Bots. New Riders, Indianapolis, Indiana, USA.

Galitz, W.O. (1996). Essential Guide to User Interface Design. Wiley, New York, NY.

Goodwin, M. (1989). User Interfaces In C - Programmer's Guide to State-of-the-Art Interfaces. Management Information Source Press, Portland, Oregon.

Horton, W.K. (1990). Designing and Writing Online Documentation - Help Files to Hypertext. Wiley, New York, NY.

IUI (1993). Proceedings of the International Workshop on Intelligent User Interfaces. Jan. 4-7, 1993, Orlando, Florida. ISBN: 0-89791-556-9.

IUI (1997). Proceedings of the International Workshop on Intelligent User Interfaces. Jan. 6-9, 1997, Orlando, Florida. ISBN: 0-89791-839-8.

IUI (1998). Proceedings of the International Workshop on Intelligent User Interfaces. Jan. 6-9, 1998, San Francisco, California. ISBN: 0-89791-955-6.

IUI (1999 – In Press). Proceedings of the International Workshop on Intelligent User Interfaces. Jan. 1999.

Lee, G. (1993). Object-Oriented GUI Application Development, Prentice-Hall, Englewood Cliffs, NJ.

Marcus, A. (1992). Graphic Design for Electronic Documents and User Interfaces. ACM Press and Addison-Wesley, Reading, MA.

Molich, R. and J. Nielsen, (1990). Improving a Human Computer Dialogue. CACM, 33:3 (March), 338-348.

Ören, T.I. (1997). Criteria for User/System Interface Design and Evaluation. In: Proc. of AFCEA Turkey Seminar on Battlefield Visualization, Sept. 25-26, 1997, Ankara, Turkey.

Ören, T.I., S. Çetin and F. Hocaoglu, (1998). Quality Criteria for User System Interface (In Turkish: Kullanıcı Makine Arayüzü için Nitelik Ölcütleri). In: Proceedings of Informatics '98, Sept. 2-6, 1998, Istanbul, Turkey, pp. 76-83.

Scientific American (1998). Special Report on: "How Hackers Break In." Volume 279, No. 4, October 1998 issue.

Sullivan, J.W. and S.W. Tyler, (1991). Intelligent User Interfaces. ACM Press and Addison-Wesley, Reading, MA.

Table 1. Convenience (usability) criteria for user/system interfaces

	Criteria		Questions	Yes	No			Comments
					Accept	Improve	Reject	
1.1	Convenience of the language	**1**	Is the natural language used in the interface, easy to understand for the users of the system? (e.g., is it the native language of the users?)					
1.2	Convenience of the terminology	**1**	Does the interface use the terminology of the application area?					
		2	Is the terminology used in the interface clear?					
1.3	Convenience of the metaphor	**1**	Is the metaphor appropriate for the application?					
1.4	Convenience of the inputs	**1**	Are the inputs appropriate for the application?					
1.5	Functionality	**1**	Does the interface offer necessary abilities to specify problems?					
		2	Does the interface have capabilities to process, analyze and present results in a manner required by the problem?					
1.6	Simplicity	**1**	Is the interface free from unnecessary or redundant information?					
1.7	Consistency	**1**	Is it easy to initiate/terminate an action in different parts of the interface?					
		2	Are these initiations/terminations specified in different places of the interface in a consistent way?					
1.8	Minimum memory load	**1**	Does the interface avoid referral of information between screens?					
		2	Can users solve problems without memorizing the sequence of steps?					
1.9	Navigability	**1**	Can one activate the actions as directly as possible?					
		2	Can one navigate with minimum movements?					
		3	Is it clear to the users how to exit from the current operations?					
1.10	Least training	**1**	Can one learn how to use the system with a minimal training?					
		2	Does the interface offer just-in-time learning facilities?					
		3	Does the interface offer a self-paced demo on how to use the system facilities?					

Table 2. Communicativeness criteria for user/system interfaces

	Criteria		Questions	Yes	No			Comments
					Accept	Improve	Reject	
2.1	Informativeness	**1**	Can the interface display (when needed) knowledge incrementally provided by a user?					
		2	Can the interface display knowledge deduced from the knowledge provided by a user?					
		3	Can the interface display knowledge about the methodology used in solving a problem?					
		4	Can the interface display fundamental scientific and engineering knowledge?					
		5	Can the interface give knowledge about application domain?					
		6	Can the interface display knowledge about the software system and how to use it?					
2.2	Guidance	**1**	Is the interface able to guide the user for solving problems?					
		2	Can it give examples when solving any problem?					
		3	Can the interface provide sample data (with the possibility to modify and save them)?					
2.3	Perceptiveness	**1**	Is the interface perceptive what users want to do?					
		2	Can the interface determine whether users need help or not?					
2.4	Explanation ability	**1**	Can the interface explain the decisions taken by the system?					
		2	Can the interface explain the results or the solutions generated by the system?					
2.5	Expressiveness	**1**	Is the interface able to provide necessary output modes that are warranted by an application?					
2.6	Esthetic/cultural acceptance	**1**	Do the elements of the interface consistent with universal (as well as local) cultural and esthetic norms?					
2.7	Types of relationship	**1**	Is the type of relationship with users free of patronizing, informal or insulting tone?					
		2	If a human-like entity (including avatar(s)) is used, is it usage warranted (as opposed to technological curiosity) ?					

Table 3. Reliability criteria for user/system interfaces

	Criteria		Questions	Yes	No			Comments
					Accept	Improve	Reject	
3.1	Error prevention	**1**	Does the interface screen user inputs to prevent errors?					
		2	Can the interface filter the outputs to intercept unacceptable (and possibly dangerous) ones?					
3.2	Error tolerance	**1**	In case of user error, does the interface allow return to the previous state without side effects?					
3.3	Caution	**1**	Does the interface require confirmation of users for any irreversible action?					
		2	Does the interface support "undo" operation at any desirable level?					
3.4	Predictability	**1**	Does the interface do what its users would expect it to do?					
3.5	Access reliability	**1**	Does the interface allow control of access to the system?					
		2	Is the interface capable of monitoring access to the system and report it (off-line, on-line)?					

Table 4. Evolvability criteria for user/system interfaces

	Criteria		Questions	Yes	No			Comments
					Accept	Improve	Reject	
4.1	Adaptability	**1**	Can the interface provide information needed by different categories of users such as experts, transfer users, occasional users and novices? (e.g., are there shortcuts for expert users?)					
4.2	Customizability	**1**	Does the interface support preferences?					
		2	Is it possible to change the natural language that is used in the interface?					
4.3	Learning ability	**1**	Can the interface remember the usage and habits of users?					
4.4	Maintainability	**1**	Can one easily maintain the interface?					
4.5	Portability	**1**	Can one use the same interface on different platforms?					