An HCI quality attributes taxonomy for an impact analysis to interactive systems design and improvement

Fernando Pinciroli

Research Institute, Faculty of Informatics and Design, Champagnat University.
Godoy Cruz, Argentina
pincirolifernando@uch.edu.ar

Abstract. In the interaction between users and systems, software quality attributes are mainly involved. When designing interfaces for human-computer interaction different alternatives can be considered in order to obtain the highest quality in an interactive system. However, quality attributes have positive and negative contribution relationships among each other, so that a change in one of them can cause a higher improvement than expected or an unwanted degradation of the system. This is the reason why in this paper we propose a taxonomy of non-functional requirements that can be assigned quality properties susceptible to be measured to propose alternatives that achieve a better quality for the systems. Quality that can be obtained by taking into account the contribution relationships among quality attributes, in order to select those alternatives that provide the biggest gain of system quality for the design and improvement of systems and software interfaces.

Keywords: HCI, quality attributes, quality attributes taxonomy, contribution relationships among quality attributes, quality metrics.

1 Introduction

The design and improvement of human-computer interaction (HCI) is a delicate task and requires a lot of effort. In both cases, for the design and the improvement, alternative solutions are analyzed and one of them is chosen to be finally implemented. The overall quality of the interface will have a value that can be measured by different techniques offered in the literature. However, the greatest difficulty is not in the measurement techniques, but on what should be measured, and we believe that this is due to two main factors: the lack of a taxonomy for quality attributes involved in HCI and the lack of analysis of the positive and negative contribution relationships existing among them.

In this work, we offer a taxonomy of quality attributes for HCI and a perspective of how to analyze the impact of the positive and negative contribution relationships among quality attributes that could enhance an alternative solution or could prevent the degradation in the overall quality of an interactive system after the efforts to improve it.

In section 2, we will present how to measure the quality properties of a set of non-functional requirements that are structured as a taxonomic classification. In section 3, we will review how to analyze the positive and negative contributions that occur among non-functional requirements. Finally, in section 4 we will offer a taxonomy of requirements for the design and improvement of interactive systems that can be used to determine its quality measurements and to analyze the cross-impact among them.

2 Taxonomy of quality attributes to measure software systems quality

Taxonomy is the "science that deals with the principles, methods and purposes of classification"[1], normally used to present information classified in a hierarchical and systematic way. Villegas et al. [1] present a list of the main taxonomies used in the disciplines of Software Engineering and HCI. Other authors offer a good overview of the major classifications over time of quality attributes, also called non-functional requirements [2] [3]. In the particular case of quality attributes for quality in use of software products, which are of particular interest in the area of HCI, one of the most widespread classifications is the proposal by ISO/IEC 25010 [4] standard:

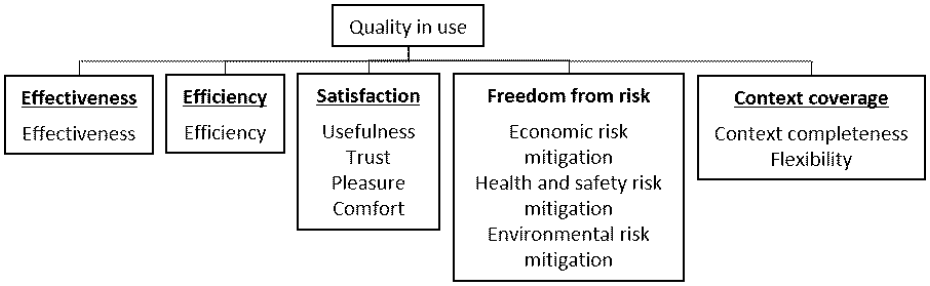
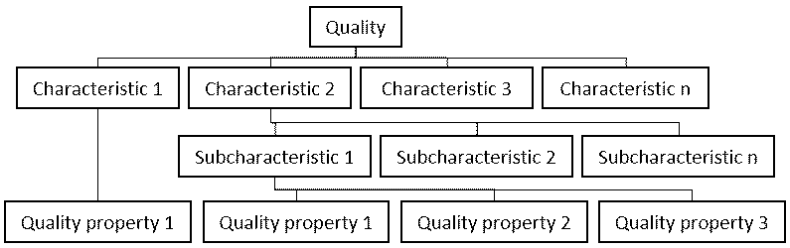


Fig. 1. Quality in use model in the ISO/IEC 25010 standard.

This model depicts a first level of five characteristics and a second level of its main subcharacteristics, which is usually extended with a new level of quality properties, which are the attributes that quality measurements are assigned to when the quality of a software product is objectively measured.



[1] Translated from the Real Academia Espanola's definition.

Fig. 2. Structure used to measure the quality.

Mairiza et al. [5] offer a broad classification of quality attributes with the same hierarchical structure, in which they call "non-functional requirement", "definition" and "attribute" to ISO's "characteristic", "subcharacteristic" and "quality property" respectively.

Taking all things considered, a breakdown structure of quality attributes is performed, where the last level has detailed the properties that will be measured to determine the level of the quality attribute and therefore go on aggregating the results in the higher nodes up to the root node, wherein the overall quality of the system is obtained.

There are different techniques for measuring the properties of the quality attributes of a system that can provide a quantitative assessment for each of them. One of these proposals is GOCAME [6], which stands for Goal-Oriented Context-Aware Measurement and Evaluation. As it can be seen in figure 3, GOCAME is applied to measure the quality of two consecutive versions of a software product. There, the values obtained for the properties of subcharacteristics are combined, then these results are combined among them to get the quality of each characteristic, and finally the same thing is repeated to obtain the overall quality of the software product.

	JIRA v.1		JIRA v.1.1	
Characterisitcs and Attributes	EI	P/G I	EI	P/G I
1. Actual Usability		53.3%		67.0%
1.1. Effectiveness		73.2%		86.7%
1.1.1. *Sub-Task Correctness*	86.4%		91.9%	
1.1.2. *Sub-Task Completeness*	87.9%		95.5%	
1.1.3. *Task Successfulness*	45.5%		72.7%	
1.2. Efficiency		29.3%		42.8%
1.2.1. *Sub-Task Correctness Efficiency*	37.4%		44.3%	
1.2.2. *Sub-Task Completeness Efficiency*	37.5%		47.3%	
1.2.3. *Task Successfulness Efficiency*	13.1%		36.8%	
1.3. Learnability in use		57.3%		71.6%
1.3.1. *Sub-Task Correctness Learnability*	78.8%		75.1%	
1.3.2. *Sub-Task Completeness Learnability*	26.4%		77.3%	
1.3.3. *Task Successfulness Learnability*	66.7%		62.5%	

Fig. 3. Measurement of Jira's usability in versions 1.0 and 1.1 (taken from [6]).

However, as shown in figure 3, although the usability of the system improved from 53.3% to 67.0% of a Jira's[2] version to the following one, some subcharacteristics suffered degradation, such as those related to system learnability. Even though it is difficult to ensure, we could think that the increment of the functionality from one version to the next one could have resulted in a greater difficulty to learn them. This is well known as contribution relationships among quality attributes contribution, which can be positive or negative, occurring the latter in the mentioned example.

[2] Jira is one of the most popular applications in the world for collaborative work, developed by the company Atlassian; http://www.atlassian.com

3 Contribution relationships among quality attributes

Quality attributes have positive and negative contribution relationships among each other. This fact, taken into account by the lead authors on requirements, is of enormous importance in establishing solutions to improve the quality of software products [7] [8] [9] [10] [11] [12].

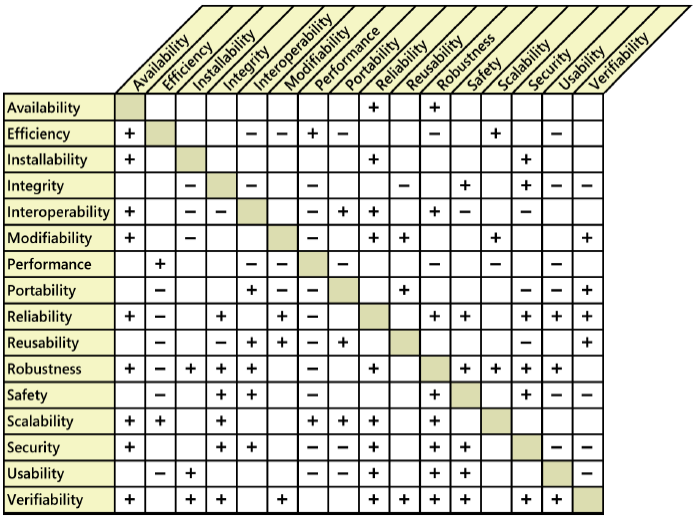


Fig. 4. Contribution relationships among quality attributes (taken from [7]).

When a need for improvement is detected in any of the quality attributes, it is a mistake to seek improvement solutions only for one attribute regardless of its relationship with the remaining ones. This is due to the fact that the solution implemented to improve one attribute could negatively affect other attributes and degrade them in such a way that a lower overall quality of the product would be obtained.

Thus, when we look for an improvement in the quality of a software product, the combination of positive and negative contributions of the different solutions must be measured in order to choose the one offering the greatest gain in overall quality. We have recently published an article presenting a technique to analyze and to measure the impact of solutions on the overall software quality, to guide the selection of the best alternative in this regard [13].

Our goal is to develop a matrix of contribution relationships to the quality attributes involved in interactive systems, but we found some difficulties on this that we will try to resolve.

4 Taxonomy of quality attributes for HCI

It is difficult to set a taxonomy for quality attributes related to HCI. Different authors, many of them recognized in the field, have different opinions and sometimes even contradictory. One of the difficulties is to set a classification of quality attributes. In the research conducted by Mairiza et al. [5] along the existing literature on quality attributes, they obtained an exhaustive list of them. But at the same time, they found that some authors present as quality attributes what other authors consider properties of quality attributes.

Another difficulty is the diversity of criteria in establishing the quality attributes that influence the acceptability of interactive systems. For example, the ISO/IEC 25010 standard [4] details the quality attributes and their respective subcharacteristic to a system's quality in use. It also indicates which quality properties of software products and computer systems influence on quality in use for primary users of the system. As shown in figure 5, ISO considers that *"functional suitability"*, *"performance efficiency"*, *"usability"*, *"reliability"* and *"security"* of the product have influence on quality in use, while according to that standard *"compatibility"* (besides *"maintainability"* and *"portability"*) does not.

Software product properties	Computer system properties	Product quality characteristic	Influence on quality in use for primary users	Influence on quality in use for maintenance tasks	Information system quality concerns of other stakeholders
↳	↳	Functional suitability	*		
↳	↳	Performance efficiency	*		*
↳	↳	Compatibility		*	
↳	↳	Usability	*		
↳	↳	Reliability	*		*
↳	↳	Security	*		*
↳	↳	Maintainability		*	
↳	↳	Portability		*	

Fig. 5. Product and system quality that influence on quality in use (taken from [4]).

However, if we look at the known Nielsen's classification of acceptability [14], we will find a different opinion.

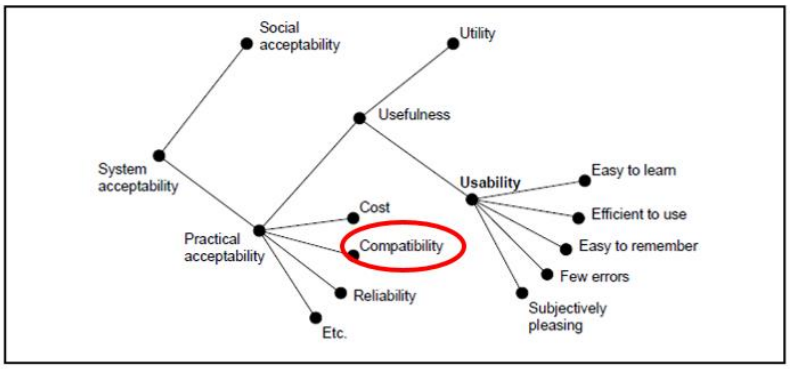


Fig. 6. Nielsen's system acceptability model.

In the classification presented in figure 6, it can be seen that *"compatibility"* is part of the system acceptability, contrary what is indicated above for quality in use by ISO 25010 standard. For all the aspects previously considered is that we dare to propose a taxonomy of quality attributes for this area. To develop this taxonomy we start from Nielsen's model due to its widespread use and acceptance, but we extend it with a new level of quality subcharacteristics obtained from the following sources:

1. Compilations of the most comprehensive lists of quality attributes presented by different authors, from which we have taken only those attributes that influence the HCI [2] [3] [5] [15].
2. Within the Nielsen's category of "Few errors" we incorporate the *"antifragility"* quality attribute which is a new concept that considers that the systems must not only be robust, but even strengthened and improved from the impacts they receive [16] [17] [18].
3. We incorporate other HCI specific quality attributes from the material of the PhD course "Design of interactive systems from a user-centered approach", taught by Dr. César Collazos [19].

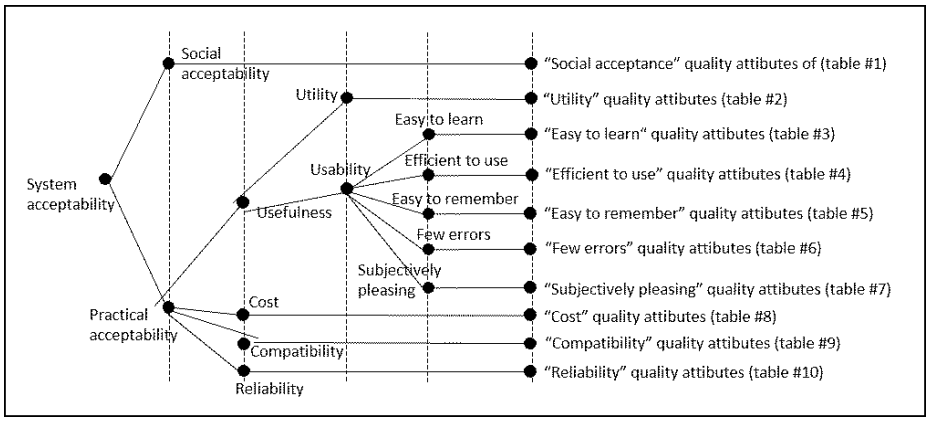


Fig. 7. Extended Nielsen's acceptability model.

We have extended the Nielsen's model with a new level of quality attributes, as shown in figure 7. The quality attributes of the new level mentioned in the figure above are detailed in the following explanatory tables.

Table 1. "Social acceptability" quality attributes.

Attributes	
Diffusion	Status

Table 2. "Utility" quality attributes.

Attributes	
Completeness	Functionality
Comprehensiveness	Maturity
Coverage	Service quality
Effectiveness	Suitability

Table 3. "Easy to learn" quality attributes.

Attributes	
Affordability	Natural relationship between controls and their functions
Auto-description	Observability
Available helps	Perception
Communicativeness	Predictability
Comprehensibility	Presentability
Conciseness	Readability
Consistency	Simple design
Content and interaction metaphor	Structuredness
Cultural level	Trainability
Decreased cognitive load	Universality
Degree of technology knowledge	Use of analogies
Easy to use	Use of icons
Expressiveness	Use of standards
Language and communication	Visibility

Table 4. "Easy to use" quality attributes.

Attributes	
Adaptation to different types of environments	Latency
Agility	Layout and organization of controls
Alternatives of use	Layout of the most important information
Availability	Manageability
Comparability	Multiuser architecture
Complexity of interaction	Navigation among windows

Configurability	Navigation inside the windows
Conformance	Operability
Controllability	Physical size of the equipment
Customizability	Reconfigurability
Design for users with cognitive decreases	Remote operation
Design for users with hearing decreases	Repeatability
Design for users with motion decreases	Replicability
Design for users with visual decreases	Response time
Dialogue techniques	Simplicity
Flexibility	Use of system resources
Handling of attention	Use of user resources
Installability	WYSIWYG

Table 5. "Easy to remember" quality attributes.

Attributes	
Experience gain	Mechanisms to help remember where you are
Mechanisms to help remember actions	Memory of use

Table 6. "Few errors" quality attributes.

Attributes	
Access control	Possibility of correcting an error
Accuracy	Possibility of reversing an error
Antifragility	Presentation of correct messages
Correctness	Recoverability
Demonstrability	Restriction indication
Distinction of colors	Stability
Error highlighting	Traceability
Error protection	Uniformity
Fault tolerance	Use of codes besides colors
Feedback of results of actions	Verifiability
Feedback of the actions taken	Visibility of system status
Immunity	Visual structure of information
Integrity	

Table 7. "Subjectively pleasing" quality attributes.

Attributes	
Actualization of technology	Esthetic
Appropriate combination of colors	Fatigue and health
Attractiveness	Formality
Comfort	Gamification
Emotionality	Motivation
Ergonomics	Social context

Table 8. "Cost" quality attributes.

Attributes	
Acquisition cost	Training cost
Cost of consumables	Update cost
Maintenance cost	Upgrade cost

Table 9. "Compatibility" quality attributes.

Attributes	
Coexistence	Mobility
Standardizability	Portability
Generality	Replaceability
Integratability	Transferability
Interoperability	

Table 10. "Reliability" quality attributes.

Attributes	
Anonymity	Privacy
Auditability	Protection
Certainty	Responsibility
Confidentiality	Security
Dependability	Supportability
Non-repudiation	Trustability

Quality attributes presented in the tables above are, then, the basis of quality measurement of interactive systems and of confrontation of their respective contribution relationships in order to determine the best alternative for the design and improvement of these systems.

5 Conclusions and future work

We consider this work as a starting point for the design and improvement of HCI from a taxonomy of the involved quality attributes (tables 1 to 10) that can have applied measurement techniques to obtain a comparable measure of the resulting overall system quality (figure 3) and considering the positive and negative contribution relationships among quality attributes (figure 4) in order to select the design or improvement alternatives providing the best measure of the system quality.

To achieve this there is still much work to be done. The first is to achieve an adequate taxonomy for HCI quality attributes. Different taxonomies and classifications of quality attributes that can be found in the literature are not uniform. In many cases the quality attributes are presented clearly, with sufficient definitions, but other times it is difficult to ensure the concept that aims to convey the author. Moreover, these taxonomies can present different quality attributes groupings, where

sometimes some of them are grouped into each other and in other opportunities they are presented at the same level. This is why we presented our proposal in figure 7.

The second step consists on proposing a matrix of contribution relationships among quality attributes brought from taxonomy obtained in the previous point.

Finally, the definition of the form of measurement and the metrics associated with each quality attribute as the proposal presented in section 2 of this work [6] is pending.

References

1. Villegas, M. et al. "Activity theory as a framework for accommodating cultural factors in HCI studies," IEEE Lat. Am. Trans., vol. 14, no. 2, pp. 844–857, 2016.
2. Afreen, N. et al. "A Taxonomy of Software's Non-functional Requirements". Proceedings of the 2nd Intl. Conference on Computer and Communication Technologies, 2015, pp. 47–54.
3. Odeh, Y. and M. Odeh, "A New Classification of Non-Functional Requirements for Service-Oriented Software Enineering," Nauss.Edu.Sa, pp. 1–7, 2009.
4. ISO/IEC, "Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models", vol. 2011, 2011.
5. Mairiza, D. et al. "An Investigation into the Notion of Non-Functional Requirements". Proceedings of the ACM Symp. on Applied Computing, 2010, pp. 311–317.
6. Olsina, L., P. Lew, A. Dieser, and B. Rivera, "Using web quality models and a strategy for purpose-oriented evaluations," J. Web Eng., vol. 10, no. 4, pp. 316–352, 2011.
7. Wiegers, K. and J. Beatty, Software Requirements. Microsoft Press, 2013.
8. Chung, L., B. Nixon, and E. Yu, "Using Non-Functional Requirements to Systematically Select Among Alternatives in Architectural Design". 1st Intl. Work. Archit. Softw. Syst. - Coop. with 17th Int. Conf. Softw. Eng. ICSE 1995, pp. 31–43, 1995.
9. Chung, L., B. Nixon, E. Yu, and J. Mylopoulos, "The NFR Framework in Action," Non-Functional Requirements, pp. 15–45, 2000.
10. Mairiza, D. and D. Zowghi, "Constructing a Catalogue of Conflicts among Non-functional Requirements," Commun. Comput. Inf. Sci., vol. 230, pp. 31–44, 2011.
11. Barbacci, M., et al. "Quality Attributes" CMU/SEI-95-TR-021. 1995.
12. Hines, M. and A. Goerner, "Software quality: attributes and modalities". Trans. Inf. Commun. Technol., vol. 11, 1995.
13. Pinciroli, F. "Improving software applications quality by considering the contribution relationship among quality attributes". 3rd Int. Work. Comput. Antifragility Antifragile Eng. (ANTIFRAGILE 2016). Elsevier, Procedia Computer Science, vol. 83, pp. 970-975, 2016.
14. Nielsen, J. Usability engineering. San Francisco: Morgan Kaufmann Publishers Inc., 1993.
15. Masip, L. et al. "User experience specification through quality attributes," Lect. Notes Comput. Sci., vol. 6949 LNCS, no. PART 4, pp. 656–660, 2011.
16. Taleb, N. Antifragile. Things that gain from disorder. New York: Random House, 2012.
17. De Florio, V. "Antifragility=Elasticity+Resilience+Machine learning: Models and algorithms for open system fidelity," Procedia Comput. Sci., vol. 32, pp. 834–841, 2014.
18. Jones, K. "Engineering antifragile systems: A change in design philosophy," Procedia Comput. Sci., vol. 32, pp. 870–875, 2014.
19. Collazos, C. "Diseño de sistemas interactivos desde un enfoque centrado en el usuario". Material of the PhD on Computer Sciences course, Univ. Nacional de San Juan, May 2016.