RESPONSIBILITIES IN THE USABILITY REQUIREMENTS ELICITATION PROCESS

Marianella Aveledo
Ana M. Moreno
Facultad de Informática
Universidad Politécnica de Madrid
Madrid, Spain

ABSTRACT

Like any other software system quality attribute, usability places requirements on software components. In particular, it has been demonstrated that certain usability features have a direct impact throughout the software process. This paper details an approach that looks at how to deal with certain usability features in the early software development stages. In particular, we consider usability features as functional usability requirements using patterns that have been termed usability patterns to elicit requirements. Additionally, we clearly establish the responsibilities of all the players at the usability requirements elicitation stage.

Keywords: Usability, Requirements Engineering, Human-Computer Interaction and Elicitation Patterns.

1. INTRODUCTION

Usability is a software quality attribute listed in most classifications. Usability means anything that helps a specific group of users to use a software product to effectively, efficiently and satisfactorily achieve their specific goals in a specific use context [1]. Along these lines, usability goals can include a wide range of system aspects also related to other aspects such as learning. Note that usability is increasingly recognized as one of the most important factors for software system acceptance [2].

Over the last twenty years, usability in software development has been primarily related to how to present information to the user. Taking into account this principle, software engineers have dealt with usability using design strategies that separate the presentation layer from the system functionality. One example is the "Model View Controller" software architecture [3]. This separation makes it possible and much easier to modify the user interface to increase system usability without affecting the remainder of the application. This way usability could be dealt with in the later development stages, particularly as part of testing.

Recently though, it has been demonstrated that usability has implications beyond the user interface and affects the software architecture components [4].

The paper follows on from a broader study looking at how to deal with usability throughout the development process starting with the requirements stage [5]. Approaching usability at the requirements elicitation stage has the same benefits as dealing with any other software quality attribute at the early stages of development, plus the fact that it is much less costly than dealing with it later on when modifications are generally highly complex, impractical and sometimes even out of the question [6].

The aim then is to study the best approach for incorporating usability features with greater implications for software functionality at the requirements elicitation stage.

The seminal research used as a basis for this paper [5] focuses on creating unambiguous usability guidance for developers and other players involved in the requirements elicitation stage by proposing clear and precise artefacts for eliciting usability requirements, as well as establishing the responsibilities of each player at this stage.

The paper is divided into the following sections. Section 1 is the introduction. Section 2 presents the background to this research. Section 3 sets out the contributions of this research. Section 4 outlines the conclusions. Finally Section 5 contains the references.

2. BACKGROUND OF THIS RESEARCH

Patterns are well-known and established formats for exchanging experience and have been used in several disciplines to capture engineering knowledge and also provide support for generating successful engineering solutions [7]. Erich Gamma et al. popularized patterns in software engineering. Their work was founded on research into building patterns by the architect Christopher Alexander [8].

Since then, patterns have moved into several software engineering areas [6]. This way, we now have patterns for the different software development stages. For the requirements engineering stage, in particular, patterns have been developed as general-purpose methods for capturing and exchanging tried and tested practices [4].

The goal is to propose artefacts (patterns) for reusing usability knowledge and supporting developers during the usability requirements elicitation stage. These patterns will then be able to be used to extract all the information required to fully and unambiguously specify the system's usability features [7].

These patterns were published in [5] at http://is.ls.fi.upm.es/research/usability/usability-elicitation-patterns and match the usability features of:

- Feedback
- Undo/cancel
- User input error prevention
- Wizard
- User profile
- Help
- Command aggregation

3. PATTERN ENRICHMENT

3.1 Extension of pattern information

The first noteworthy contribution of this paper is to upgrade the knowledge covered in the above patterns by either identifying new types of usability features within each pattern family or by specifying new information to be discussed during each pattern's requirements elicitation process.

Table 1 is a summary of the effect of this paper's input. Column 1 shows the different usability features examined, called families. Column 2 lists the patterns of each of these families. Column 3 shows the new proposals in human-computer interaction (HCI) literature, which have been used to update each pattern. This column specifies the aliases that the consulted HCI authors use to refer to the pattern about which information has been extended. Column 4 shows the patterns included with their respective aliases as specified in the literature.

It is noteworthy that the upgrades and the additions come from sources both in and outside the area of HCI. Within the UNDO/CANCEL family, for example, the inclusion of the "Multi-user-undo" pattern [9],[10], [11],[12], [13], applicable for multi-user applications, and the "Selective Undo" pattern [14], [15], [16], designed to undo particular actions in a history, leaving work done afterwards unchanged, deserve a special mention.

Table 2 shows a sample of the above-mentioned additions for the UNDO/CANCEL family.

3.2 Identification of responsibilities

Finally, with the aim of stipulating exhaustive guidance for developers in the elicitation process, the responsibilities of each of the players in this elicitation process were established. To do this, players were divided into three different groups. The first group, stakeholders (S), covers all those people and/or organizations that have some sort of stake in the system [17]. This group includes users. Remember, in this respect, that users are not a homogeneous group. The second group is developers without HCI knowledge (D), and the third is analysts and/or developers with some HCI knowledge (DHCI). These two groups can provide support and offer help and/or suggestions to stakeholders on how to deal with the different features mentioned above. Table 3.2.a shows the responsibilities for the pattern illustrated in Table 3.

To summarize, Table 4 shows the role of each of the three types of players in the requirements process. Using this guidance it is possible to schedule the requirements elicitation sessions to be held. This is a further aspect to be taken into account to calculate the costs associated with the usability requirements elicitation process. In order do it, Table 5 shows the percentage of participation of each player involved in the elicitation process of usability requirements. This percentage was calculated based on the number of questions to be held by each player.

3.3 Proposed Process

We suggest then that these usability requirements elicitation patterns be added to the mechanisms and techniques to be used in the process of eliciting the requirements of the new software system to be developed and be used to specify all the usability requirements that have a direct impact on system architecture. Figure 1 is an illustration of this process.

4. CONCLUSIONS

Adding usability features to a software system requires a great many situations to be taken into account that very often not even the stakeholders are able to formulate. Therefore, a lot of information is required.

This information is a product of lengthy discussions between system users and developers to be able to properly specify the different usability features.

The patterns developed in this research help to define these features, as they clearly set out the different scenarios to be taken into account, as well as providing guidance as to exactly how the discussions should be staged. Through these patterns, system developers that are not necessarily familiar with the different usability mechanisms can advise stakeholders about the usability solutions for the system under construction.
Similarly, the responsibilities tables establish exactly what responsibilities each of the different players have in the specification of usability requirements.

Pattern Family	**Pattern**	**Action**	
		Upgrade	**Addition**
FEEDBACK	System status feedback	-	-
	Interaction feedback	-	-
	Progress feedback	Progress bar[18], Determinate and Indeterminate Progress bar[19] Progress Bar[20], Feedback[21], Progress Indicator[22], Progress indicator [23].	
	Warning	Give a warning[24] y [25], Warning message[22], Forgive the user[21], Warning or Error Message[21].	
UNDO/CANCEL	Global Undo	Linear Multi-Level Undo[9].	
	Multi-User Undo		Multi-user undo[9], [10] and [11], Undo in Multi-user Applications[12], Group Undo[13].
	Object Specific undo	-	-
	Selective Undo		Selective Undo[14], Direct Selective Undo[15] and [16]
	Abort Operation	Cancelability [26]	
HELP	Multilevel Help	Multilevel Help [26]	
USER INPUT ERRORS PREVENTION/ CORRECTION	Structured Text entry	Format Required[27], Structured Format[26], Input Hints[26], Input Prompt[26].	
STEP BY STEP	Wizard	Wizard[26]	
COMMANDS AGGREGATION	Macros	Macros[26]	

Table 1: Enrichment of usability patterns.

IDENTIFICATION	
Name: Selective UNDO **NON HCI AUTHORS**	
Family: UNDO/CANCEL	
Alias : Selective Undo [14], Direct Selective Undo [15], Direct Selective Undo [16]	
PROBLEM	
Which information needs to be elicited and specified in order to provide users with selective undo information	
USABILITY CONTEXT	
Situation: When building a highly interactive system with multiple and complex functionalities on specific objects of the system	
USABILITY FEATURE CONFIGURATION GUIDE	
NON HCI AUTHORS RECOMMENDATION	Issues to be discussed with stakeholders
In some cases, it can be meaningful to allow single actions from the history to be deleted. This is the case when a certain 'episode' of work must be deleted or undone while keeping work that has been done later on [16]. Selective Undo is conceptually more difficult than linear undo since there is a notion of 'dependency between actions' that determines the consequences of undoing a particular action. For example, if a 'create circle' action is undone at some point in the history, subsequent actions in the history working on that object lose their meaning and must be deleted [14].	*1.1 Is it useful to provide selective undo?*
A simple and general way to present the commands is by describing them by a text string, similar to the example above. The string should contain —the name of the command, —denotations for the affected objects, and —the values of the actual parameters. The user can select the desired command from this list of strings. When the user selects a command, it is immediately undone. Selection of another command undoes the previous selective undo and instead undoes the new command. In this way the user can quickly locate the desired command by observing the effects of selective undo. Only by clicking on "Undo This" is the undo really accepted (and the dialogue dismissed). "Cancel" undoes the last selective undo and also dismisses the dialogue box, restoring the situation before calling the selective undo dialogue [15]. The basic meaning of selective undo is that the affected values of the objects are returned to the state just before the command was executed [16]	*1.2 If so, how should it be presented?*

Table 2: "Selective Undo" Pattern.

Question	S	D	DHCI	Comments
1.1 Is it useful to provide selective undo?	X	X		If the developers think the selective undo is useful for the user, they will present this option to the stakeholders. Stakeholders will decide whether or not to offer the selective undo to the users.
1.2 If so, how should it be presented?	X		X	Developers will suggest the best way to present this option to the user. Stakeholders will decide how to present this option to the user.

Table 3: Responsibilities for the "Selective Undo" Pattern.

Pattern Family	Pattern	S	D	DHCI
Feedback	System status feedback	X	X	X
	Interaction feedback	X		X
	Progress feedback	X	X	X
	Warning	X	X	X
Undo/Cancel	Global undo	X	X	X
	Multi-user undo	X	X	
	Object specific undo	X	X	X
	Selective undo	X	X	X
	Abort operation	X	X	X
Help	Multilevel help	X	X	X
User input errors preventions/cor-rection	Structured text entry	X	X	X
Step by Step	Wizard	X	X	X
Commands Aggregation	Macros	X	X	

Table 4: Player Involvement.

Pattern Family	Pattern	Number of Questions	S	D	DHCI
Feedback	System status feedback	10	100%	40%	30%
	Interaction feedback	1	100%	0%	100%
	Progress feedback	5	100%	20%	80%
	Warning	2	50%	50%	50%
Undo/Cancel	Global undo	6	60%	80%	20%
	Multi-user undo	2	100%	100%	0%
	Object specific undo	3	100%	6,6%	33,3%
	Selective undo	2	100%	50%	50%
	Abort operation	6	80%	40%	60%
Help	Multilevel help	4	100%	50%	25%
User input errors preventions/cor-rection	Structured text entry	3	66,6%	33,3%	66,6%
Step by Step	Wizard	5	80%	80%	40%
Commands Aggregation	Macros	4	100%	75%	0%

Table 5: Degree of Player Involvement.

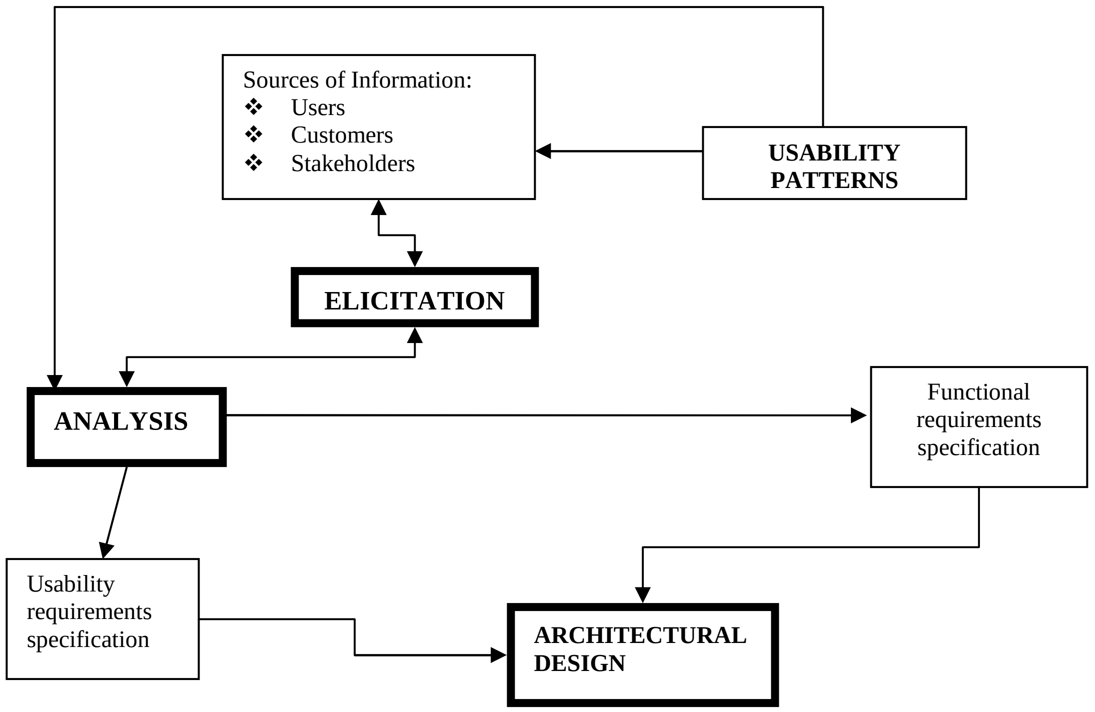

Figure 1: Process

5. REFERENCES

[1] "ISO Std 9241-11: Ergonomic Requirements for office Work with Visual Display Terminals. Part11: Guidance on Usability"ISO,1998.

[2] L.M. Cysneros, V.M. Wemeck, A. Kushniruk "Reusable Knowledge for Satisficing Usability Requirements". Proceedings of the 2005 13th IEEE International Conference on Requirements Engineering.

[3] http://java.sun.com/blueprints/patterns/MVC.html.

[4] Juristo, N.;Moreno, A.M.; Sanchez-Segura,M. "Analysing the impact of usability on Software Design" Journal of Systems and Software, Vol. 80,Issue 9,2007,pp.1506-1516.

[5] Juristo, N.; Moreno, A.M.; Sanchez-Segura M. "Guidelines for eliciting usability Functionalities". IEEE Transactions on Software Engineering, Vol. 33, N°. 11, November 2007, pp. 744-758.

[6] M. Barbacci, R. Ellison, A. Lattanze, J. A. Stafford, C.B. Weinstock. Quality Attribute Workshop, 3rd ed. CMU/SEI-2003-TR-016, Software Engineering Institute, CMU 2003.

[7] Hagge, L.; Lappe, K." Sharing Requirements Engineering Experience Using Patterns" IEEE Software, January-February 2005, pp. 24-31

[8] Carlsson, D. " A Categorization of HCI patterns" Department of Computing Science, Umea University, Sweden, 2004.

[9] http://eelke.com/wiki/index.php?Multi-Leve-Undo.

[10] Chen, D.; Chengzheng, S. "Undoing any operation in collaborative Graphics editing systems" Copyright 2001 ACM 1-58113-294-8/01/0009

[11] Sun, C. "Undo any operation at any time in group editors" Proceedings of ACM Conference of computer Supported Cooperative work (CSCW2000) ACM Press, Pittsburgh, PA,2000, pp. 191-200

[12] Berlage, T. " A Selective Undo Mechanism for Graphical Use Interfaces Based on Command Objects" ACM Transactions on Computer Interaction, Vol.1,Nº3,September 1994,Pages 269-294.

[13] Ressel, M,; Gunzenhäuser, R. "Reducing the problems of Group Undo" Copyright ACM 1999 1-58113-065-1/99

[14] Folmer, E; Welie, M.;Bosh, J. "Bridging patterns: An approach to bridge gaps between SE and HCI. Information and Software Technology 48 (2006) 69-89.

[15] Berlage, T. " A Selective Undo Mechanism for Graphical Use Interfaces Based on Command Objects" ACM Transactions on Computer Interaction, Vol.1,Nº3,September 1994,Pages 269-294.

[16] Meng, C.; Yasue, M.; Imamiya, A.;Mao,X "Visualizing Histories for Selective Undo and Redo" in the Proceedings of Computer Human Interaction, 1998.

[17] Boutelle, J. "Understanding Organizational Stakeholders for Design Success. http://www.boxesandarrows.com.

[18] http://java.sun.com/docs/books/tutorial/uiswing/components/progress.html

[19] http://developers.sun.com/prodtech/javatools/jscreator/le

arning/tutorials/2/ajaxprogressbar.html
[20] http://www.cs.cf.ac.uk/Dave/HCI/HCI_Handout_CALLER/node126.html#SECTION000111000000000000000
[21] GNOME Human Guidelines (2.0) http://developer.gnome.org/projects/gup/hig/2.0/
[22] http://www.isii.com/style_guide/progress_indicator.html
[23] http://developer.apple.com/documentation/UserExperience/Conceptual/OSXHIGuidelines/XHIGControls/chapter_18_section_5.html
[24] http://www.it.bton.ac.uk/staff/lp22/guidelinesdraft.html
[25] Griffiths, R.; Pemberton, L. “Don’t write Guidelines write patterns” http://www.it.bton.ac.uk/lp22/guidelinesdraft.html
[26] Tidwell, J. “Designing Interfaces” O´Reilly Media, Inc. 2006
[27] http://www.cmis.brighton.ac.uk/Research/patterns/GvWrnng.html