Proceedings of ICAD2002
Second International Conference on Axiomatic Design
Cambridge, MA – June 10&11, 2002
ICAD 020

COUPLING IN DESIGN OF HUMAN COMPUTER INTERACTION.

Martin G. Helander
mahel@ntu.edu.sg
School of Mechanical and Production Engineering
Nanyang Technological University
Singapore 639790

Jianxin Jiao
mjiao@ntu.edu.sg
School of Mechanical and Production Engineering
Nanyang Technological University
Singapore 639790

ABSTRACT

Axiomatic design procedures may be used to decouple usability analyses in Human-Computer Interaction. Nielsen's ten usability heuristics were analyzed in terms of implications for FRS and DP's. From the results we conclude that heuristic usability analysis leads to a coupled design process. To uncouple the design a cluster analysis was performed on the original design matrix. FR's were then split and recombined in order to reduce the coupling.. The

Keywords: Usability Design, Software Design, Decoupling,

1 INTRODUCTION

In the design of a user interface the designer must consider both the capabilities and limitations of the user. The purpose is to enhance productivity as well as user satisfaction. In this paper we will primarily focus on usability assessment of software.

In typical application software the user interface takes approximately 55-60% of the written code. According to Landauer (1997) there are in average 40 usability bugs in applications software. Removing them is highly profitable. Typically the productivity in handling the application will improve by about 50 %, and the benefit-cost ratio for improving the user interface has been estimated to about 40:1.

Early studies in Human-Computer Interaction focused on the generation of design recommendations. As an example the U.S. Air Force supported for several years research to produce guidelines for user-interface design. The report by Smith and Mosier (1986) was proceeded by several reports by Smith in 1980, 1981, 1982, 1983 and 1984. The 1986 report presented 994 guidelines. These were design recommendations mostly at a low level, such a principles for abbreviation command words. The guidelines were however difficult to use - in may cases because it was difficult to interpret the implications (semantics) for use with a specific application. Guidelines have since then become less favored (Helander 1988; Helander, Landauer and Prabhu, 1997), and usability testing with real users is now often favored.

One popular alternative to usability testing is Usability Heuristics (Molich and Nielsen, 1990; Nielsen, 1994). These are high level functional requirements, which are then interpreted in terms of design requirements. This final paper will give several examples of usability design using heuristics. Below we analyze, as an example the 10 heuristics presented by Nielsen (1994), see Table 1. The context of axiomatic design is used, so that Functional Requirements (FRs) and Design Parameters (DPs) were added to Nielsen's original text.

Table 1. Ten Usability Heuristics (Nielsen, 1994). FRs and DPs below are added to Nielsen's text.

FR 1. Visibility of system status
The system should always keep users informed about what is going on, through appropriate feedback (**DP 1**) within reasonable time.

FR 2. Match between system and the real world
The system should speak the users' language (**DP 2**) with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

FR 3. User control and freedom
Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo (**DP 3**).

FR 4. Consistency and standards
Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions. (**Author's interpretation - Use same words, DP 4).**

FR 5. Error prevention
Even better than good error messages is a careful design which prevents a problem from occurring in the first place. (**Author's Interpretation – Careful Design DP 5).**

FR 6. Recognition rather than recall
Make objects, actions, and options visible (**DP 6).** The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

FR 7. Flexibility and efficiency of use
Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user (**DP 7**) such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

FR 8. Aesthetic and minimalist design
Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility. (**Author's interpretation – Show only relevant information, DP 8).**

FR 9. Help users recognize, diagnose, and recover from errors
Error messages should be expressed in plain language (no codes) **(DP 9),** precisely indicate the problem, and constructively suggest a solution.

FR 10. Help and documentation
Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, **(DP 10)** list concrete steps to be carried out, and not be too large.

2. ANALYSIS OF HEURISTICS

The heuristics were interpreted in terms of FRs and corresponding DPs, see Figure 1.

For the purpose of our calculations we assume that all A_{xx} are unitary, such as in Figure 2. This has the advantage that it is possible to perform a cluster analysis on the matrix.

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \\ FR_4 \\ FR_5 \\ FR_6 \\ FR_7 \\ FR_8 \\ FR_9 \\ FR_{10} \end{bmatrix} = \begin{bmatrix} A_{11} & & A_{13} & & A_{15} & A_{16} & & A_{18} & A_{19} & \\ & A_{22} & & A_{24} & A_{25} & & & A_{28} & A_{29} & A_{210} \\ A_{31} & & A_{33} & A_{34} & A_{35} & A_{36} & A_{37} & & & \\ & A_{42} & & A_{44} & & & & A_{48} & & \\ & A_{52} & & & A_{55} & & & A_{58} & A_{59} & \\ & A_{62} & A_{63} & A_{64} & & A_{66} & & A_{68} & & \\ & A_{72} & & & & & A_{77} & & & \\ & A_{82} & & & & & & A_{88} & & \\ A_{91} & A_{92} & A_{93} & & & A_{96} & & A_{98} & A_{99} & \\ & A_{102} & & & & & & A_{108} & & A_{1010} \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \\ DP_4 \\ DP_5 \\ DP_6 \\ DP_7 \\ DP_8 \\ DP_9 \\ DP_{10} \end{bmatrix}$$

Figure 1. FRs are influenced by several DPs.

	DP1	DP2	DP3	DP4	DP5	DP6	DP7	DP8	DP9	DP10
FR1	1	0	1	0	1	1	0	1	1	0
FR2	0	1	0	1	1	0	0	1	1	1
FR3	1	0	1	1	1	1	1	0	0	0
FR4	0	1	0	1	0	0	0	1	0	0
FR5	0	1	0	0	1	0	0	1	1	0
FR6	0	1	1	1	0	1	0	1	0	0
FR7	0	1	0	0	0	0	1	0	0	0
FR8	0	1	0	0	0	0	0	1	0	0
FR9	1	1	1	0	0	1	0	1	1	0
FR10	0	1	0	0	0	0	0	1	0	1

Figure 2. Design matrix assuming unitary relationships

The main effect of the cluster analysis is that it will push the design matrix to be diagonal. The result indicates the maximal extent to which the original matrix can be diagonal. In some sense, this represents a measure of axiom 1 - the extent of interdependency (Harutunian et al. , 1996). In essence the rows and columns of the equation are used to form clusters, such as in Figure 3.

	DP7	DP1	DP3	DP6	DP4	DP5	DP9	DP10	DP8	DP2
FR3	1	1	1	1	1	1	0	0	0	0
FR1	0	1	1	1	0	1	1	0	1	0
FR7	1	0	0	0	0	0	0	0	0	1
FR8	0	0	0	0	0	0	0	0	1	1
FR4	0	0	0	0	1	0	0	0	1	1
FR6	0	0	1	1	1	0	0	0	1	1
FR9	0	1	1	1	0	0	1	0	1	1
FR5	0	0	0	0	0	1	1	0	1	1
FR10	0	0	0	0	0	0	0	1	1	1
FR2	0	0	0	0	1	1	1	1	1	1

Figure 3. Clustered Matrix

The clustered design matrix can now be further analyzed according to several rules.

- The numbers of FRs and DPs can be unequal. This suggests possible ways to elaborate FRs of DPs so as to manipulate the matrix for various purposes.
- In this paper, FRs represent usability heuristics and DP represents design parameters. The usual approach is to select a set of DPs that would make the FRs independent - and hence produce a diagonal matrix. In our case the DPs have little room to change. This seems a "reverse engineering" would be a better approach – given DPs, elaborate/determine FRs through matrix analysis.

- While clusters/cells indicate strong coupling, the inter-cluster/cell elements suggest tradeoffs between two groups (cluster/cell).

- Based on the clustered matrix, FRs may be elaborated according to possible improvement of "clusters". In figure 4 six FRs were split up into two component parts: FR 2 into FR2-1 and FR 2-2, FR 4 into FR41 and FR42 , FR5 into FR51 and FR 52, FR6 into FR61 and FR62, FR9 into FR91 and FR92 and FR 10 into FR10-1 anf FR10-2. This created sixteen FRs. Through this manipulation it was possible tpo create four clusters, which are shaded in Figure 4. The seven FR listed at the bottom have two DPs in common: DP8 and DP2. It eould be possible to combine the seven FRs into two Frs one for each DP. Thereby the nubmer of FRs would again be identical to the number of DPs. Simalarly it may ber possiblre to "redesign" some of the other FR's which are conbimed in the other clusters. In this way, we may define alternative FRs or heuristics, which are identified primarily through their DPs.
- The rational of this type of analysis lies in that usability heuristics are developed with reference to the DPs (the way users are supposed to follow the heuristics). This alleviates the "jargon" heuristics that are formulated in isolation from the DPs.

	DP 7. Use Accelerators	DP 1. User Feedback	DP 3. Support Undo and Redo	DP 6. Make Objects and Actions Visible	DP 4. Use same words	DP 5. Design to avoid errors	DP 9. Informative error messages	DP 10. Concrete help suggestions	DP 8. Show only relevant information	DP 2. Speak users language
FR 3. User Control and Freedom	A37	A31	A33	A36	A34	A35				
FR 1. Visibility of System Status		A11	A13	A16		A15	A19		A18	
FR 4-1.Consistency and Standard					A44					
FR 6-1. Recognition not recall			A63	A66	A64					
FR 9-1. Diagnose & recover from errors		A91	A93	A96			A99			
FR 7. Flexibility and efficiency of use	A77									A72
FR 5-1. Error Prevention						A55	A59			
FR 10-1. Help and Documentation								A1010		
FR 2-1. Match system to real world					A24	A25	A29	A210		
FR 8. Minimalist design									A88	A82
FR 4-2.Consistency and Standard									A48	A42
FR 6-2. Recognition not recall									A68	A62
FR 9-2. Diagnose & recover from errors									A98	A92
FR 5-2. Error Prevention									A58	A52
FR 10-2. Help and Documentation									A108	A102
FR 2-2. Match system to real world									A28	A22

Figure 4. Several FR's are split to to create new FRs.

3. DISCUSSION

As demonstrated in Figure 1, the design matrix is coupled. This makes it difficult to find design solutions that are appropriate to all FRs. Some of the DPs turn out to be relevant to most of the FRs. This

is particularly the case with DP 2 – Speak the user's language and DP 8 – Show only relevant information. Both of these design parameters
were (in the author's judgment) relevant to most of the FRs. This indicates that it is difficult to find a satisfactory solution that satisfies all FRs. Ideally the FRs should be reformulated so that the design matrix becomes decoupled.

Let us take DP 8 as an example – Show only relevant information. This DP must then – through the choice of relevant information – satisfy the requirements of eight different FRs. Since the FRs are different, it is unlikely that they will be satisfied by the same information. Therefore it will be impossible to produce a satisfactory design solution – and the design matrix remains coupled.

In some sense the results are not surprising. Cognitive problems are complex. As illustrated above with DP 2 and DP 8, some DPs are bound to affect most cognitive problems. This is unfortunate, since it makes it difficult to grapple with cognitive ergonomics, and it makes it difficult to propose clear design solutions that can be understood by non-professionals. It should not come as a surprise that Usability testing has become the most popular method for interface evaluation. This method focuses on the discovery and removal of user errors as they are provoked by design deficiencies in the interface. The coupled design solution will not be so obvious, since the interface is improved incrementally without any thought to coupling.

Design problems in physical ergonomics (anthropometrics and biomechanics design) are more straightforward (Helander, 2000). In these cases it is feasible to find decoupled solutions.

4. REFERENCES

Helander, M.G. (Ed.) (1988). Handbook of Human-Computer Interaction. Amsterdam, The Netherlands: North Holland.

Helander, M.G., Landauer, T.K. and Prabhu, P. (Eds.) (1997). Handbook of Human-Computer Interaction, 2nd Edition. Amsterdam, The Netherlands: North Holland.

Helander, M.G. (2000). Anthropometrics Design of Workstations. In Tate, D. (Ed.). Proceedings of First International Conference on Axiomatic Design. pp 130 -139. Cambridge, MA: Institute for Axiomatic Design.

Landauer, T.K. (1995). The trouble with computers. Cambridge, MA: MIT Press

Molich, R, and Nielsen J. (1990). Improving a human-computer dialogue. Communications of the ACM, 33, 3 (March), 338-348.