Use-centered interface design for an adaptable administration system for chemical process design

Christian Foltz[1], Bernhard Westfechtel[2], Holger Luczak[1]

[1]Chair and Institute of Industrial Engineering and Ergonomics
RWTH Aachen University
Bergdriesch 27, D-52062 Aachen
{c.foltz | h.luczak}@iaw.rwth-aachen.de

[2]Department of Computer Science III
Software Engineering
RWTH Aachen University
Ahornstr. 55, D-52074 Aachen
bernhard@i3.informatik.rwth-aachen.de

Abstract

In this paper, the use-centered interface design for an adaptable administration system for chemical process design is presented. On the basis of the main work activities to be supported with the tool, two different analytical evaluation methods were applied to the prototypical original interface. The derived requirements led to the design and implementation of an alternative interface which was compared with the original one in an experimental study with ten users.

1 Introduction

In general, chemical process design is characterized as a complex, iterative and creative activity typically starting as an ill-defined problem (e.g., Westerberg, Biegler & Grossman, 1997). Within the course of a development project an interdisciplinary team designs and uses different documents or models in various software systems like spreadsheet and flowsheet tools, process simulators, word processors, etc. These data are often stored in document management systems and engineering data management systems, respectively. Additionally, strictly separated software systems address the coordination of the project team supporting classical management functions such as planning, organizing, and controlling by means of project plans.

To permit an integrated view on weakly structured development processes, the above mentioned features of different systems have been combined in one tool called AHEAD (adaptable and human-centered environment for the administration of development processes, Westfechtel, 1999). Within AHEAD three different environments can be distinguished. First, project managers will be supported by the management environment which is concerned with the coordination of teams, i.e. responsibility assignment, monitoring of milestones and deadlines, etc. Second, the developer environment aims to support project team members on the individual level by giving a coherent view on tasks and relevant documents. Third, the modeling environment allows to provide domain specific knowledge, e.g., by defining task types and pre-defined task sequences.

This contribution is concerned with the analysis of the original and the design of an alternative graphical user interface for the developer environment which basically contains two user interfaces. The agenda presents a to-do list showing the developer's tasks with present status and deadline. Choosing one task from the agenda, a new window named work context gives the following detailed information about this task. First, a task net allows to overview related tasks and documents. Second, all documents concerning the task are listed in a document list. Third, with a version graph different document versions can be managed (Figure 1, left side).

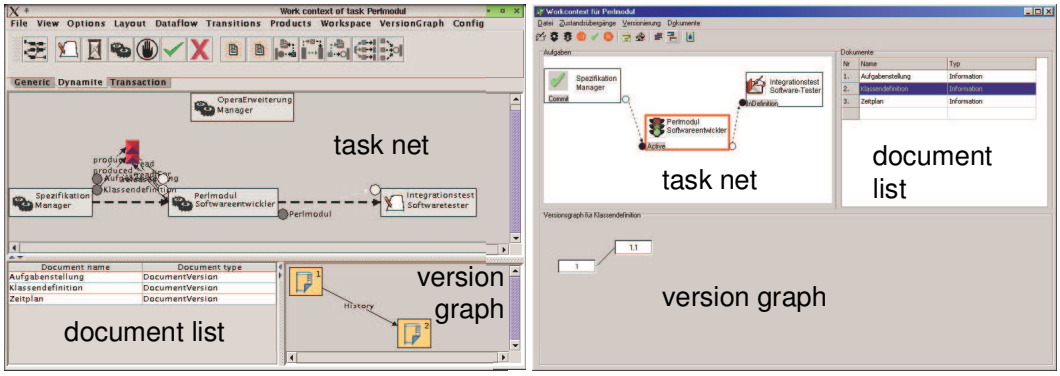

Figure 1: Work context of the AHEAD developer environment (left: original; right: alternative)

2 Analysis and Implementation

The aim of the original, prototypical user interface for the developer environment was mainly to integrate the different software features in one graphical interface. From a computer science perspective this implementation of different conceptual models has been a challenging activity (cf. Westfechtel, 1999).

The design of an alternative interface followed the use-centered design approach. In the use-centered perspective the focus shifts from the interaction between humans and machines to the interaction between humans and work (cf. Flach, Tanabe, Monta, Vicente & Rasmussen, 1998). Consequently, the main work activities which should be supported by the developer environment were identified. In a second step, it was explored which actions and operations have to be performed by an user in order to fulfil these activities using the original interface. For the sake of clarity HTA-diagrams (hierarchical task analysis; Diaper, 1989; Kirwan & Ainsworth, 1992) were used to visualise the necessary interaction steps. Figure 2 reveals that even simple actions like "create a new document" demand a great amount of operations, i.e., browsing through menus and select functions, and mouse movements. To unburden the user from those annoying interactions (Shneiderman, 1998; Johnson, 2000) several single operations were merged into one operation, e.g., when a new document is created an initial version is automatically produced. The grey diagonal bars in Figure 2 denote which user interactions could be saved in the alternative interface without removing functionality.

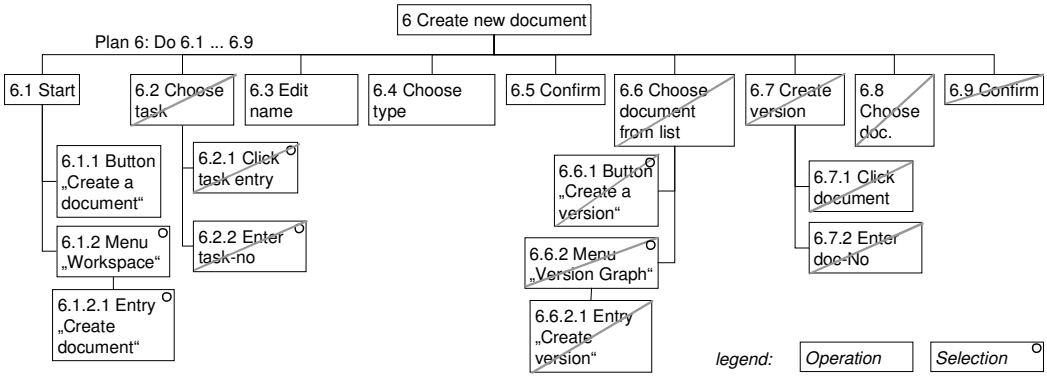

Figure 2: HTA-diagram for the activity "Create new document"

Before prototyping an alternative interface a heuristic evaluation (Nielsen, 1993) was used to uncover further usability problems. Here, different labelling of functions in menu and in button tool tip can be mentioned as well as the arrangement of buttons following neither the importance principle nor the frequency-of-use principle nor the sequence-of-use principle (Sanders & McCormick, 1993).
Based on these insights an alternative interface was designed and implemented with Borland's Delphi. This interface is a mock-up with which all relevant activities can be performed. However, the mock-up cannot be connected and used with the full-functional AHEAD system.

3 Empirical Study

In an empirical study the interfaces have been tested for at least two reasons. First, it should be compared how efficient and effective users can work with both interfaces. Second, it was expected that the empirical study will reveal further usability problems which have not been detected using the analytical evaluation methods.

3.1 Methodology

3.1.1 Independent variables

The two independent variables were the graphical user interfaces (original vs. alternative) and the sequence of use (original-alternative vs. alternative-original).

3.1.2 Dependent variables

The five dependent variables can be distinguished in measures for effectiveness (a. labeling of objects (correct/ false), b. producing states (solved/ unsolved), c. editing tasks (solved/ unsolved)) and efficiency (d. time to produce states, e. time to edit tasks)).

3.1.3 Apparatus

The experiment was performed on a notebook (Pentium III processor with 600 MHz, 128MB RAM) with dual boot option (Windows 98, SuSE Linux 7.3). Due to additional requirements the original prototype could only be used running Linux, although the interface was written with Java. The alternative graphical user interface was developed running Windows 98 and Borland's Delphi. The keyboard, a mouse, and the notebook's touchpad could be used to interact with both software prototypes. The complete experiment was video taped and advised by two investigators. A stopwatch was used to detect the time consumed.

3.1.4 Participants

Ten male participants aged between 25 and 34 years with a mean of 28.8 years (SD = 3.1 years) were recruited for the empirical study. Because it was not possible to recruit experienced chemical engineers, people with experiences in weakly structured engineering processes like software development were chosen. Eight participants worked in the IT sector in positions like system administrator or software engineer. The scenario for testing was build according to this fact. The subject's work experience differed between 0 and 8 years (mean = 3.0 years, SD = 2.4 years). No participant had experience with document and workflow management systems, respectively.

However, some test persons had experiences with project and version management systems. All participants were familiar with at least one version of the Windows operating system and one half is familiar with one of the Unix operating systems including Linux. Participation was voluntary, however, a fee of 50 € each was paid to the participants who were free to abort the experiment at any time.

3.1.5 Procedure

In a preliminary survey the test persons were introduced in the aims of the AHEAD system. Afterwards a questionnaire was used to survey demographic data and information about education and experiences. Each participant used both interfaces whereas one half started with the original interface and proceeded with the alternative one. The other group used the interfaces vice versa. The experiment itself was divided into three parts. First, to test self-descriptiveness and conformity with user expectations (ISO 9241, Part 10, 1996) the participants were asked to label different buttons and screen areas of the graphical user interface. Second, three screenshots representing different states of the software system were presented. The subjects tried to produce this states within 3 minutes. Third, brief written tasks were given to the test persons. These tasks should be solved by the participant within 3 minutes. Finally, parts of the IsoMetrics usability inventory (Gediga, Hamborg & Düntsch, 1999) were used to collect data about the subjective usability estimation of each participant. However, the results of this test will not be presented here. Except for the preliminary survey this procedure was repeated for the second user interface. All in all the experimental study took between 75 and 120 minutes.

3.2 Results and Discussion

In a t-test highly significant differences between the alternative and the original user interface were observed in the dependent variables. These differences were independent of the sequence of using the graphical interfaces. Figure 3 displays exemplary the box plots for the time consumed to solve states and tasks on the left side ($t = -11.485$; $p < 0.01$). On the right side the box plots for the number of solved states and solved tasks are presented ($t = 8.333$; $p < 0.01$). There are great differences in the values for both effectiveness and efficiency measures. Even the best value for the original interface is far away from the worst value for the alternative interface. Interestingly, neither the underlying conceptual model nor the work context's general appearance and arrangement of interface parts has been changed essentially (Figure 1). Instead, the visual appearance of buttons and their arrangement has been changed as well as the sensomotoric effort

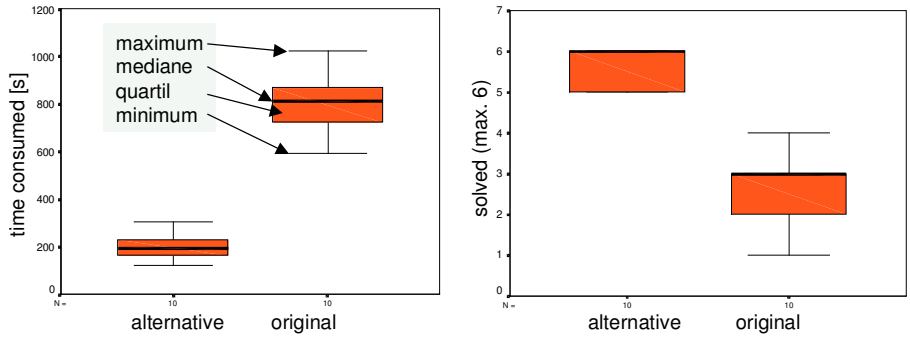

Figure 3: Time consumed (solved states and tasks only) and amount of solved states and tasks using the original and the alternative user interface of AHEAD's developer environment

for using the interface. So far, it cannot be ascertained to what extent the reduction of time consumed is caused by a shortened information processing or by reduced mouse movements. However, the increased number of solved states and tasks using the alternative interface indicates that the information processing has a considerable influence on the time consumed.
Moreover, the empirical study revealed additional usability information. For example, some users remarked that in the document list no hint is given if a document was provided by a third party or created by the user himself. Other participants interpreted the task net view as life cycle of the actual task rather than identifying the actual task and its predecessor(s) and successor(s) and, therefore, had problems in allocating documents. Thus, a further survey should deal with those issues on the conceptual level. In addition, another study should compare the developer environment with "traditional" support tools by measuring the results of realistic work scenarios.

4 Concluding Remarks

Based on an use-centered approach two analytical evaluation methods and ergonomic basics have been applied to the interface design of a novel administration system for chemical process design. This has led to significant differences in both effectiveness and efficiency of use. However, some important shortcomings have not been identified until a test with "real" users, i.e. some questions concerning the conceptual ideas of the tool have not been tackled. Three important conclusions can be drawn from this. First, analysis should be based on the user's work task, rather than on a technology-centered human-computer interaction approach. Second, ergonomics should be integrated into the software development process as early as possible. Therewith, important conceptual questions can be addressed at once. Third, a variety of both analytical and empirical usability methods should be used in all phases of the software design process.

5 Acknowledgement

The authors gratefully acknowledge the financial support of the Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center 476 IMPROVE.

6 References

Diaper, D. (Ed.). (1989). Task Analysis for Human-Computer Interaction, Chichester: Ellis Horwood.
Flach, J.M., Tanabe, F., Monta, K., Vicente, K.J., Rasmussen, J. (1998) An ecological approach to interface design. In HFES (Ed.) *Proceedings of the Human Factors and Ergonomics Society 42nd Annual Meeting* (pp. 295-299). Santa Monica: HFES.
Gediga, G., Hamborg, K.-C., Düntsch, I. (1999) The IsoMetrics usability inventory: an operationalization of ISO 9241-10 supporting summative and formative evaluation of software systems. *Behaviour & Information Technology*, 18 (3), 154-164.
Johnson, J. (2000) GUI Bloopers. San Diego: Academic Press.
Kirwan, B., Ainsworth, L.K. (1992) A guide to task analysis. London: Taylor & Francis.
Sanders, M.S., McCormick, E.J. (1993) Human Factors in Engineering Design (7th ed.) New York: McGraw-Hill.
Shneiderman, B. (1998) Designing the User Interface (3rd ed.). Reading: Addison-Wesley.
Nielsen, J. (1993) Usability Engineering. San Diego: Morgan Kaufman.
Biegler, L.T., Grossmann I.E., & Westerberg, A.W. (1997) Systematic Methods of Chemical Process Design. Upper Saddle River: Prentice Hall PTR.
Westfechtel, B. (1999) Models and Tools for Managing Development Processes. Berlin: Springer.