COMPUTER-LAIEN ALS EXPERTEN?

Warum Benutzerpartizipation bei der Entwicklung von Benutzerschnittstellen wichtig ist

Josef Bösze, David Ackermann und H.-J. Lüthi, Zürich

Die vorliegende Arbeit soll aufzeigen, wie mittels Benutzerpartizipation bei der Entwicklung und Evaluation von Programmsystemen arbeitspsychologischen Kriterien angepasste und damit für den Benutzer adäquate Systeme erzeugt werden können. Die verwendete Organisation und die programmiertechnischen Verfahren beim Design, bei der Implementierung und bei der Evaluation werden beschrieben. Anhand von ausgewählten Beispielen wird aufgezeigt, wie die in Experimenten gewonnenen Erkenntnisse über die Denkweise von Benutzern das Systemdesign verändern und verbessern.

Einleitung

Eine zentrale Rolle bei der Entwicklung und Beurteilung von Arbeitssystemen spielt das Prinzip der differentiellen und dynamischen Arbeitsgestaltung (Ulich 1978, S. 568). Dieses Prinzip besagt, dass "eine optimale Entwicklung der Persönlichkeit in der Auseinandersetzung mit der Arbeitstätigkeit auf dem Hintergrund individueller Differenzen" erfolgt. Auch gibt es gesicherte Hinweise darauf, dass an individuelle Bedürfnisse angepasste Dialoge meist effizienter sind als vorgegebene (Ackermann 1983, 1986a, 1986b, 1987).

Wie kann nun dieses Prinzip beim Design von Software berücksichtigt werden? Wie lassen sich interindividuell unterschiedliche mentale Repräsentationen und Informationsbedürfnisse von Benützern im Softwareentwicklungsprozess berücksichtigen? Wie muss eine Benutzerschnittstelle gestaltet sein, damit möglichst viele Benutzer möglichst gut damit umgehen können? Im folgenden soll am Beispiel der Entwicklung eines Decision Support Systems (DSS) gezeigt werden, wie sich oben erwähnte arbeitspsychologischen Prinzipien im Softwareentwicklungsprozess berücksichtigen lassen, wie der Prozess der "adaptiven Systemgestal-

tung" funktioniert und wie das zur Zeit lauffähige DSS sich dem Benutzer präsentiert.

Fragestellung

Die Stadt Zürich besitzt eine zivile Schutzorganisation, kurz Zivilschutz (ZS) genannt. Aufgabe dieser Organisation ist es unter anderem, der Bevölkerung im Falle einer notwendig werdenden Evakuation Schutzplätze zuzuweisen. Um die Disposition zu erleichtern, ist das Gebiet der Stadt Zürich in verschiedene Einheiten wie Abschnitte, Sektoren, Quartiere, Blöcke eingeteilt. In jedem Block (Bl) befinden sich Schutzräume (SR) verschiedener Qualitätskategorien (Kat) mit unterschiedlicher Anzahl Schutzplätze (SP). Die zu evakuierenden Personen werden zuerst zu sogenannten Sammelstellen geführt, und von dort aus auf Blöcke mit Schutzräumen, die noch freie Schutzplätze haben, verteilt. Falls keine freien SP mehr vorhanden sind, können künstlich zusätzliche geschaffen werden, indem der Überbelegungsfaktor der SRe erhöht wird oder man die Leute in weiter entfernte Gebiete (mit noch freien Kapazitäten) verlegt.

Im Rahmen der Arbeit sollte für die beschriebene Evakuationsplanung ein entscheidungsunterstützendes System (Decision Support System, DSS, vgl. Sprague & Carlson 1982) für und gemeinsam mit Angehörigen der Zivilschutzorganisation der Stadt Zürich zu Schulungszwecken entwickelt werden. Als Basis diente ein von H.-J. Lüthi (1982) entwickeltes System mit den entsprechenden Zuweisungs-Algorithmen. Wichtige Anforderungen des Zivilschutzes an das System waren, dass es auf einem Personal Computer (PC) lauffähig sein musste und dass auch Computer-Laien das System bedienen sollten. Vor allem die erste Forderung bereitete anfangs einige Schwierigkeiten, da das oben erwähnte Programm mit grossen Datenmengen über Personen, Schutzräume und die geographische Anordnung der Schutzräume auf dem Stadtgebiet arbeitete. Abklärungen und Berechnungen ergaben dann aber, dass die Speicherkapazität heutiger PC's für eine vernünftige Implementation genügen.

Der Entwicklungsprozess

Die Planungsphase begann mit intensiven Diskussionen mit den Verantwortlichen des Zivilschutzes, wobei es darum ging, die

Begriffe zu standardisieren, ein vertieftes Problemverständnis zu erlangen und ein (meta-mathematisches) Modell der Evakuationsplanung zu entwerfen, bzw. das Modell des bestehenden Grosscomputer Prototyps neu zu überdenken. Da die Beteiligten mit der im Operations Research gebräuchlichen mathematischen Notation nicht vertraut waren, mussten die abstrakten Formeln jeweils verbalisiert und in klare, einfach einzusehende, aussagekräftige graphische Darstellungen abgebildet werden. In einem nächsten Schritt, wiederum zusammen mit Leuten des Zivilschutzes, wurde die Funktionalität des Programmes und die Bedienungsphilosophie (Menü-Auswahl, mehrere nichtüberlappende Fenster, Help-Funktion) definiert. Ein wichtiger Punkt dabei war die Suche nach einer guten, dem Benutzer möglichst vertrauten Repräsentation des Problem- und Lösungsraumes auf dem Bildschirm und die Definition von Menustrukturen, welche den Problemlöseprozess in seiner logischen Abfolge unterstützen (Ackermann 1987). Eine gute Darstellung für die meisten Daten wurde in Form einer vom Zivilschutz erstellten Karte der Stadt gefunden, da diese Darstellung den Benützern vertraut war und praktisch alle problemorientierten Informationen sich in dieser Kartendarstellung visualisieren liessen.

Um den detaillierten Entwurfsprozess zu unterstützen, wurde von einer Vielzahl von Techniken Gebrauch gemacht. In einer ersten Phase wurden anhand von Papierskizzen Vorschläge zur Dialog- und Bildschirmgestaltung, insbesondere die Aufteilung und Anordnung der Fenster, diskutiert. In einem nächsten Schritt wurde, um ein Gefühl für den Dialog und den ganzen Problemlösungsprozess zu bekommen, mit 'Comics-Serien' (von Hand skizzierte kommentierte Bildschirmzustände) gearbeitet. In einer weiteren Phase wurde in einem iterativen Prozess mittels eines interaktiven Graphikprogrammes der Bildschirmlayout definiert. Bedingt durch die direkte Arbeit auf dem Bildschirm war es möglich, wahrnehmungspsychologische Kriterien bei der Gestaltung der Bildschirmdarstellung auf einfachste Art und Weise zu berücksichtigen und auszutesten (wie z.B. bei der Darstellung von Kartenelementen (Stricharten, Füllmuster), Plazierung der verschiedenen Arbeitsfenster und Fehler-Informationen, Cursortypen). Besonders beim Design der oben genannten Kartendarstellung erwies sich die Verwendung des Graphik-Programmes als äusserst nützlich. Mit Hilfe dieser Techniken war es möglich, viele Eigenschaften des

Systems zu definieren, zu testen und wertvolle Erkenntnisse für die Gestaltung des Handlungsspielraums zu gewinnen, ohne aber auch nur eine Zeile programmiert zu haben, dies im Gegensatz zum "Rapid Prototyping".

Anschliessend an die Abklärungen wurde das System programmiert, wobei bewusst ein etwas höherer Implementationsaufwand in Kauf genommen wurde, um ein leicht adaptierbares System zu erhalten, welches in weiten Grenzen modular entwicklungs- und änderungsfähig ist, sowohl bezüglich der Funktionalität als auch der Dialogführung. Die Entwicklungs- und Änderungsfähigkeit des Systems bildet eine der wichtigsten Voraussetzungen, um eine iterative Systementwicklung (und Anpassung) mit Benutzerbeteiligung durchführen zu können.

Da das System vor allem hinsichtlich der Benutzungseigenschaften (Darstellungen von Daten, Übereinstimmung der Menustruktur mit der Handlungsstruktur des Benutzers) optimiert werden sollte, wurden die Änderungsmöglichkeiten vor allem im Dialogbereich des Systems implementiert. Andere Teile des Systems, wie z.B. die Datenverwaltung, der verwendete Transhipmentalgorithmus für die Optimierung) wurden fest programmiert.

Das System musste aber nicht nur anpassungsfähig sein, die Änderungen mussten auch schnell, sicher und ohne Seiteneffekte durchgeführt werden können. Dies wurde durch eine weitgehende Modularisierung des Programmsystems, wie sie in der Programmiersprache Modula-2 unterstützt wird, auf elegante Art und Weise erreicht. Die gerätenahen Ein/Ausgabe Funktionen wurden in seperaten Modulen zusammengefasst und werden durch ein einfaches, der GKS-Norm entsprechendes Graphiksystem angesprochen. Durch einfaches Auswechseln des Gerätetreibermoduls ist das Programm nun auf verschiedenen PC's mit verschiedenen Bildschirmen lauffähig. Um einen einfachen und konsistenten Dialog gewährleisten und auch die Logfile-Technik einfach implementieren zu können, wurde für die Dialogsteuerung ein eigenes Modul entwikkelt. Dieses Dialog-Modul verwaltet die verschiedenen Bildschirmfenster und stellt dem Hauptprogramm nur einige wenige Funktionen zur Interaktionssteuerung zur Verfügung. Es werden auch nur einige wenige Interaktionstypen unterstützt, wie z.B. die Eingabe ganzer Zahlen, von Text, von JA/NEIN Antworten, von Punktkoordi-

naten im Graphikfenster, wobei verschiedene Cursortypen verwendet werden können, und die Auswahl von Menuitems.

Der letzte Punkt, dass die Menuauswahl - und dadurch auch die Aktivierung der Funktionen (bzw. Prozeduren) - direkt durch das Dialog-Modul kontrolliert wird, erlaubte es erst, auf einfachste Art und Weise Menustrukturen zu verändern und gar unterschiedliche Menustrukturen wahlweise aktivieren zu können. Die Informationen über die Menustrukturen sind zur Zeit fest aber leicht änderbar im Programm eingebaut; im Prinzip kann das System aber so abgeändert werden, dass diese Informationen in einem Textfile vorliegen und erst beim Starten des Systems geladen werden. Dies würde zu noch grösserer Flexibilität führen, ohne dass jedesmal das Programm neu übersetzt werden müsste.

Für die Datenorganisation wurden je nach Datentyp (geographische Daten, Schutzraum- und Personendaten, Daten über die aktive Lösungsvariante) verschiedene Module entwickelt, sodass die eigentlichen Problemlösungsmodule sich weder um Darstellungen, um Interaktionen noch um Datenverwaltung zu kümmern brauchen.

Diese zusätzliche "Investition" in die klare Modularisierung und relativ starke Verallgemeinerung von Funktionen hat sich während der diversen Iterationszyklen der Systementwicklung bestens "ausgezahlt" gemacht. Sobald nämlich ein Teil der Software lauffähig war, wurde sie mit Benutzern ausgetestet. Design- und Implementationsfehler wurden schnell erkannt und konnten sogleich korrigiert werden. Gleichzeitig wurden aber auch neue Wünsche und Ideen von der Benutzerseite aufgenommen und soweit als möglich bei der weiteren Programmierung berücksichtigt. Der Schlüssel zu dieser hohen Adaptierbarkeit lag im oben beschriebenen Aufbau des Programmsystems.

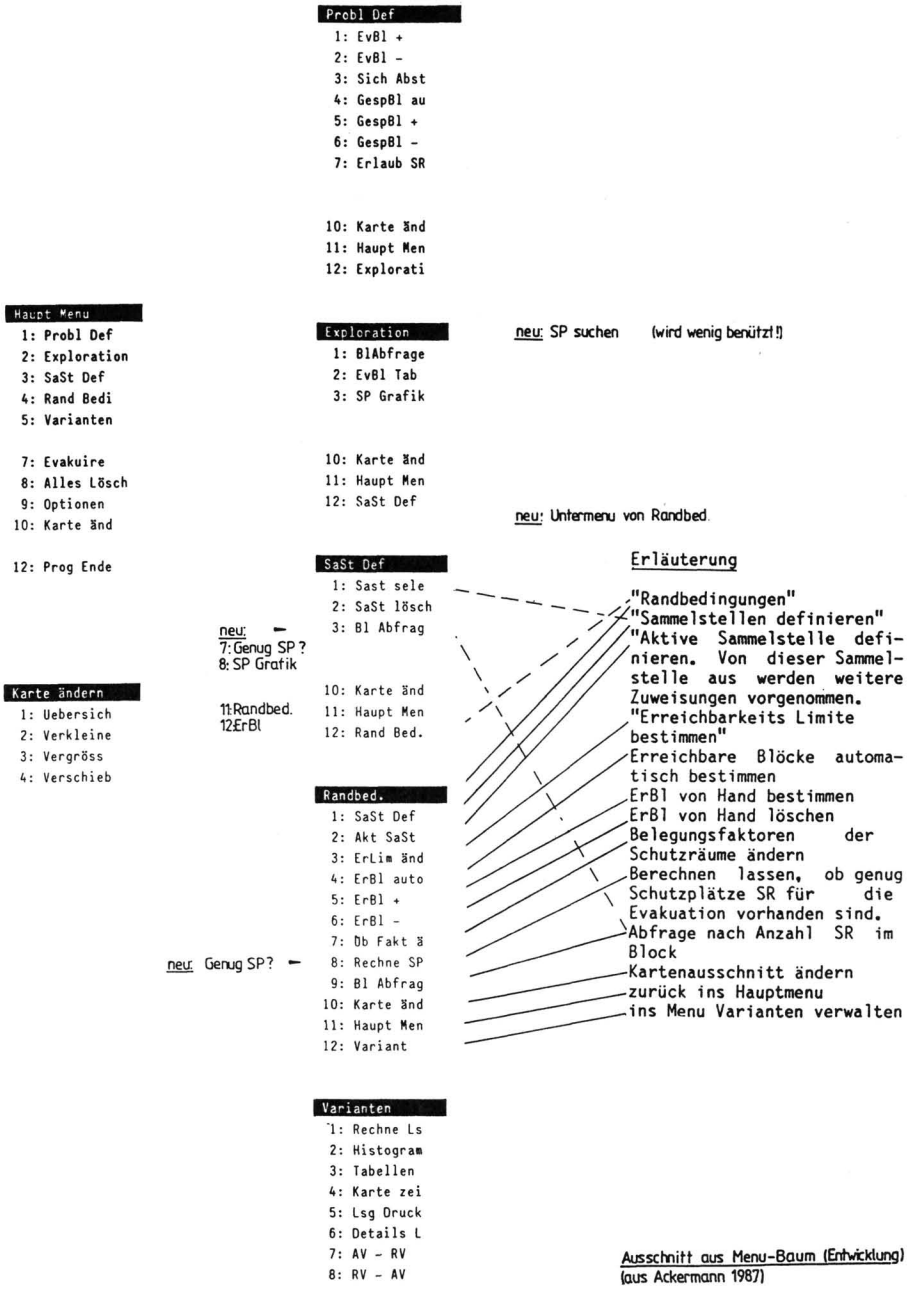
Systemevalution und Logfiles

Weil das System sehr gründlich mit Benutzern getestet werden sollte, wurde die Möglichkeit ins Programm eingebaut, auf einem 'Logfile' jede Benutzer- und Programmaktion genau zu protokollieren. Zu diesem Zweck wurden Daten auf dem Keystroke-, Dialog- und Programmfunktionenlevel erfasst. Die Analyse dieser Daten ermöglichte u.a. typische Fehlersituationen und Handlungsmuster zu erkennen und das Programm dementsprechend abzuändern.

Interessant war, wie das Programm die Benutzer teilweise richtiggehend in Fehlersituationen hineintrieb, wenn die "implizite Führung" des Programmes versagte. Diese Fehlersituationen entstanden oft dadurch, dass die Anordnung der Befehle in den Menüs von der Abfolge dem Benutzer "unlogisch" erschien oder dass die Namensgebung (Abkürzungen) Verwirrung stiftete. Die Anzahl der Fehlersituationen konnte durch Umstellungen und Umbenennungen der Befehle in den einzelnen Menüs und durch Änderung der globalen Menüstruktur drastisch gesenkt werden. Durch die Analyse der Logfiles konnte auch definiert werden, welche Menüs zusammengelegt bzw. aufgeteilt werden sollten, um ein effizienteres Arbeiten zu ermöglichen.

Die markanteste Änderung zeigte sich in der globalen Menustruktur. In den früheren Versionen des Systems bildete das Menu "SSt Def" (Sammelstellen Definieren) ein eigenständiges Menu auf der obersten Menustufe (Hierarchieebene). Das Menu "Randbed." (Randbedingungen) besass dafür die in Abb. 1 abgebildete Struktur. In dieser Anordnung wird die Definition der Sammelstellen in einem eigenständigen Menu vorgenommen, wie dies der im Entwicklungsprozess definierten semantischen Einteilung der Handlungsstruktur entsprach. Sammelstellen müssen obligatorisch vorgegeben werden, erreichbare Blöcke und Belegungsfaktoren sind nur bei Bedarf zu definieren oder zu ändern. Arbeitet der Benutzer entsprechend dieser logischen Abfolge, so wird ihm durch die Menustruktur nahegelegt, erst die obligatorischen Sammelstellen zu definieren und dann erst die weiteren Randbedingungen anzupassen. Durch diese Dialoggrammatik sollte "folgerichtiges Denken" im Dialog unterstützt und möglichen Fehlern vorgebeugt werden (Ackermann 1987). In den Pilotexperimenten mit den Benutzern zeigte es sich, dass die Lösungssuche auf zwei Menus umständlich und verwirrend ist. Will der Benutzer Belegungsfaktoren der Schutzräume nicht ändern, so will er sie doch meist kontrollieren und ruft dieses Menuitem auf, bevor er beginnt, die notwendigen Sammelstellen zu definieren. Die meisten Benutzer definierten oder kontrollierten erst die Belegungsfaktoren, definierten daraufhin die notwendigen Sammelstellen und anschliessend die für die weitere Verteilung allenfalls notwendigen "Erreichbaren Blöcke". Die Semantik der Dialoggrammatik wurde daher an das "natürliche" Denken des Benutzers angepasst und die Menus

Abb. 1: Primäre Menustruktur vor der Korrektur mit Erläuterungen



Ausschnitt aus Menu-Baum (Entwicklung)
(aus Ackermann 1987)

entsprechend geändert. Andere Anpassungen betrafen die Umstellung von Funktionen in einzelnen Menus und die Umbenennung von Funktionen. Abb. 1 zeigt eine der ersten Varianten des Menubaums. Das Menu, welches die Suche nach freien Schutzplätzen unterstützt, ist mit dem Begriff "Exploration" bezeichnet. Es wurde nicht benutzt, da die Versuchspartner den Begriff von ihrem Sprachverständnis aus nicht richtig interpretieren konnten. Sie wussten nicht, was man in der Evakuationsplanung mit "Exploration" anfangen könnte. Das Menuitem "Rechne SP" drückt konkret aus der Sicht des Informatikers das aus, was das Programm macht: Es werden die für die Evakuation benötigten Schutzplätze berechnet und mit den eingeplanten freien Schutzplätzen verglichen, d.h. es wird lediglich geprüft, ob die Evakuation rechnerisch durchgeführt werden könnte. Die Versuchspartner nahmen aber an, dass mit diesem Menuitem die Lösung des Evakuationsproblems ermittelt werde. Das hiefür notwendige Menuitem hiess aber "Rechne Ls" (Rechne Lösung). Die bei Wahl dieses Menuitems allenfalls folgende Meldung "Genug SP: Evakuation kann durchgeführt werden" verwirrte die Versuchspartner, weil sie nicht verstanden, was sie rechnen sollten. Sie nahmen an, der Computer hätte das doch dank des Befehls "Rechne SP" schon gerechnet. Das Menuitem wurde in "Genug SP" geändert und die Fehlerquelle war beseitigt.

Nach der Entwicklungsphase wurde das Programm mit einer vom Zivilschutz vorgegebenen Aufgabenstellung mit Angehörigen des Zivilschutzes, die bis anhin am Projekt nicht beteiligt gewesen waren, in mehreren längeren Sitzungen getestet. Dabei interessierten neben dem Verhalten im Dialog vor allem Entscheidungsverhalten und -unterstützung sowie die Güte der schliesslich gewählten Lösung.

Zusammenfassung

Das beobachtete Verhalten der Benutzer war äusserst interessant. Als erstes überraschte die überaus grosse Akzeptanz des Systems auch von Personen, die nicht direkt im Entwicklungsprozess miteinbezogen waren und anfangs sehr negativ gegenüber Computern eingestellt waren. Wir vermuten, dass das System nicht so akzeptiert worden wäre, wenn es nicht praktisch von den Benutzern gestaltet worden wäre. Eine weitere

interessante Beobachtung war, dass die durch das System offerierten Handlungsspielräume tatsächlich genutzt wurden. Die Versuchspartner arbeiteten mit sehr unterschiedlichen Strategien und benutzten das Programm auch auf ganz verschiedene Art und Weise. So wurden gute Problemlösungen auf verschiedenste Arten erarbeitet. Diese Feststellung deckt sich mit Ulich's Aussagen über individualisierte Arbeitstätigkeiten.

Die Ergebnisse zeigen, dass Systementwicklungen auf der Basis des Prinzipis der differentiellen und dynamischen Arbeitsgestaltung möglich sind. Erfreulich ist, dass das Resultat die Erwartungen der Beteiligten bei weitem übertroffen hat und das der aufwendige Entwicklungsprozess durch die vorliegenden Resultate gerechtfertigt ist. Zur Zeit werden anhand der gewonnen Erfahrungen neue Applikationen auf der gleichen Basis entwickelt.

Herrn Prof. Dr. E. Ulich danken wir für die Unterstützung dieser Arbeit und die Durchsicht des Manuskriptes.

Literatur

Ackermann, D.(1984): Untersuchungen zum individualisierten Computerdialog: Einfluss des Operativen Abbildsystems uaf Handlungs- und Gestaltspielraum auf die Arbeitseffizienz. In: Dirlich, G., Freksa, C., Schwatlo, U. & Wimmer, K. (Hrsg.): Kognitive Aspekte der Mensch-Computer-Interaktion. Ergebnisse eines Workshops vom 12./13. April 1984 in München. Berlin, Springer, 1986.

Ackermann, D.(1985): A pilot study on the effects of individualization in man-computer-interaction. 2nd IFAC/IFIP/IFORS/IEA Conference on Analysis, Design and Evaluation of man-machine-studies. Varese, September 1985. London: Pergamon Press, 1986.

Ackermann, D.(1987): Handlungsspielraum, mentale Repräsentation und Handlungsregulation am Beispiel der Mensch - Computer Interaktion. Unveröff. Dissertation, Universität Bern.

Betschart, M. & Kalchofner, S.(1985): Evakuationsplanung im Zivilschutz. Unveröffentlichte Semesterarbeit (Betr: H.-J. Lüthi), IFOR ETHZ, 1985.

Lüthi, H.-J.(1982): Computerunterstützte Planung im Zivilschutz - Zuweisungsplanung mit CASA. Output 12, 1982, S. 19-22.

Sprague, R.H. & Carlson, E.D.(1982): Building Effective Decision Support Systems. Prentice-Hall, 1982.

Ulich, E.(1978): Über das Prinzip der differentiellen Arbeitsgestaltung. Industrielle Organisation, 47, 1978.
Ulich, E.(1985): Arbeitspsychologische Konzepte für Computerunterstützte Büroarbeit, Spektrum 14, 1985.

Josef Bösze
NCR (Schweiz)
Hauptsitz
CH-8301 Glattzentrum

David Ackermann
Lehrstuhl für Arbeits- und
Organisationspsychologie
ETH-Zürich
CH-8092 Zürich

Hans-Jakob Lüthi
Institut für Operations Research
ETH-Zürich
CH-8092 Zürich

V PRINZIPIEN DER SCHNITTSTELLENGESTALTUNG