

## **Phasenmodell ist OUT**

### **Benutzerbeteiligung jetzt auch bei Standardsoftware-Entwicklung<sup>1</sup>**

**Matthias Rauterberg**  
**ETH-Zürich**

**Raimund Mollenhauer**  
**ADI-Karlsruhe**

**Philipp Spinas**  
**ETH-Zürich**

#### **Zusammenfassung**

Die wachsende Komplexität der Anwendungen wird von Softwareentwicklern häufig nicht mehr hinreichend durchschaut und bewältigt. Eine enge Zusammenarbeit von Benutzern als Experten für das Anwendungsgebiet und Entwicklern als Experten für Informatikwerkzeuge ist erforderlich. Probleme bereitet allerdings häufig die praktische Realisierung von Benutzerbeteiligung. In dem Verbundprojekt 'Entwicklung und empirische Überprüfung von Kriterien, Methoden und Modellen zur benutzerorientierten Software-Entwicklung und Dialoggestaltung' (Förderkennzeichen 01HK706) wurden deshalb Kriterien, Methoden und Modelle zur benutzerorientierten Softwareentwicklung und Schnittstellengestaltung (weiter-) entwickelt und empirisch überprüft. Nach der Aufarbeitung der Ergebnisse aus den Experimental- und Felduntersuchungen wird ein praxisorientierter Leitfaden für Software-Entwickler und Organisatoren erstellt. Im folgenden werden exemplarisch einzelne Untersuchungen und ihre Ergebnisse dargestellt.

#### **Abstract**

The current state of traditional software development is surveyed and essential problems are investigated on the basis of system theoretical considerations. The concept of user participation is proposed as a solution. The relation of several different methods of user participation to the specifications, the communications, and the optimisation barrier is integrated into a concept of participatory software development. The pros and cons of essential problems known to obstruct optimal software development and possible ways of solving them are considered. The preparation of this paper was supported by the BMFT (AuT programme) grant number 01 HK 706-0 as part of the BOSS "User oriented Software Development and Interface Design" research project.

#### **1. Einleitung**

Bei Vorhaben zur Computerunterstützung von Bürotätigkeiten stellt sich die Grundfrage: wie können Benutzerbedürfnisse bzw. Anforderungen von Fachabteilungen nach Computerunterstützung von Arbeitsabläufen so in Software umgesetzt werden, daß diese den Anforderungen auch wirklich entspricht und dadurch eine echte Hilfe bei der Bewältigung der anfallenden Aufgaben darstellt? Allzuoft sind Entwicklungsprojekte gescheitert oder haben die entwickelten Lösungen den Benutzern nicht die erwartete Unterstützung gebracht, weil ihre Bedürfnisse und Aufgaben im Prozeß der Software-

---

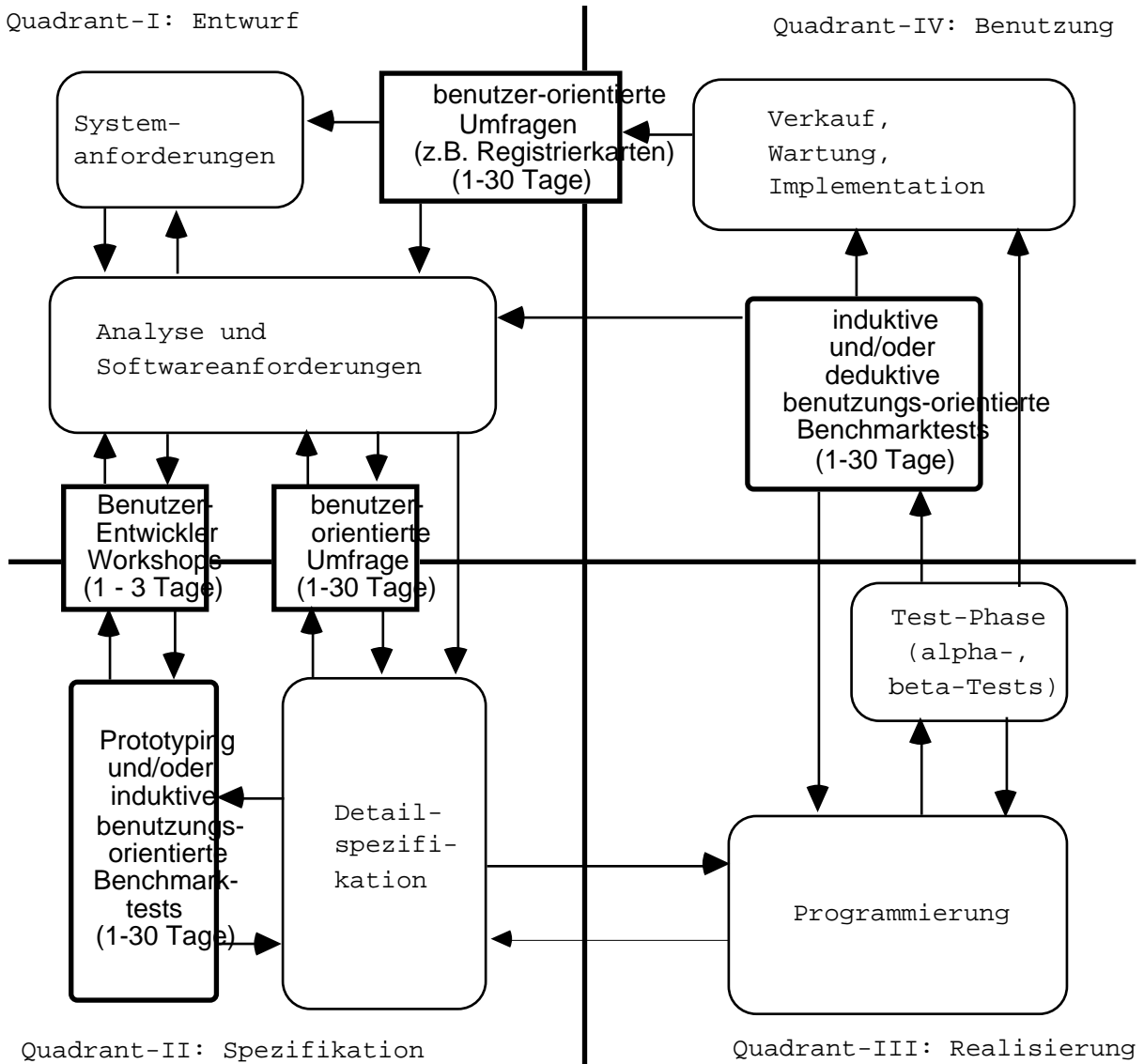
<sup>1</sup> 'Entwicklung und empirische Überprüfung von Kriterien, Methoden und Modellen zur benutzerorientierten Software-Entwicklung und Dialoggestaltung'; ADI Software GmbH, Karlsruhe/ Institut für Arbeitspsychologie der ETH-Zürich; gefördert durch das Bundesministerium für Forschung und Technologie, Projektträger "Arbeit und Technik", Bonn; Förderkennzeichen 01HK706)

Entwicklung zu wenig berücksichtigt wurden. Die Erfahrungen der letzten Jahre haben auch gezeigt, daß die Akzeptanz angebotener Computerunterstützung wesentlich von deren Benutzungsfreundlichkeit abhängt. Zur Entwicklung software-ergonomisch optimierter Anwendungen gibt es bereits eine Reihe von Erkenntnissen, die in dem Forschungsprojekt BOSS erarbeitet wurden. Die Komplexität heutiger Anwendungen verlangt aber darüber hinaus bei der Software-Entwicklung vielfach die Mitarbeit von Benutzern, um optimale, auf die konkrete Situation angepasste Lösungen zu finden. Die Vorteile einer Benutzerorientierung haben sich schon in manchen Projekten gezeigt (Strohm 1991a, 1991b, Strohm & Ulich 1991, Rauterberg & Strohm 1992, Spinass 1992); in der Praxis herrscht jedoch häufig Ungewißheit über die Realisierungsmöglichkeiten für den Einbezug von Benutzern und den Einsatz von Prototyping. Im folgenden werden verschiedene Möglichkeiten und ihre praktische Umsetzung im Prozeß der Software-Entwicklung von Standardsoftware aufgezeigt. Darüberhinaus lassen sich auch Benutzervertreter bei der Erstellung von Individual- oder Branchen-Software sinnvoll in das Projektteam integrieren (Spinass et al. 1993).

"Phasenmodell ist OUT"

Quadrant-I: Entwurf

Quadrant-IV: Benutzung



Quadrant-II: Spezifikation

Quadrant-III: Realisierung

**Abbildung 1:** Übersicht über verschiedene Methoden der Benutzerbeteiligung bei Standardsoftwareentwicklung im Rahmen des zyklischen Quadrantenmodells (Rauterberg 1991f, 1992b, 1992g, 1992h).

Mittels Fallstudien und einer schriftlichen Umfrage in mehr als 100 Betrieben wurden Entwicklungsprozesse für Applikationssoftware untersucht (Strohm 1990a). Die Ergebnisse zeigen, daß eine stark arbeitsteilige Entwicklungsorganisation und die konsequente Einhaltung eines Phasenmodells im Projektablauf nicht nur eine Beteiligung von Benutzern erschweren, sondern auch - neben Kosten- und Terminüberschreitungen wegen notwendiger Rückschritte in frühere Phasen - die Zielerreichung im Projekt insgesamt gefährden können (Rauterberg & Strohm 1992). Die Grundannahme dieses Modells - daß nämlich alle Anforderungen in der Anfangsphase vollständig bekannt sind, exakt beschrieben werden können und sich nicht verändern - kann nicht länger aufrecht

erhalten werden (Rauterberg 1991f). Das Endprodukt entfernt sich, z.B. durch Modifikationen oder Restriktionen bei der Programmierung, von den ursprünglichen Benutzeranforderungen. Ferner wurde deutlich, daß schriftliche Systembeschreibungen, Pflichtenhefte und Diagramme von den Benutzern schwer verstanden und Konsequenzen für die spätere Systembenutzung kaum erkannt werden können (Waeber 1991).

Innovative, erfolgreiche Entwicklungsprojekte hingegen weisen folgende Merkmale auf: ganzheitliche Aufgaben für die ProjektmitarbeiterInnen, aktive Benutzerbeteiligung, einkalkulierter Mehraufwand für analytische und konzeptionelle Arbeiten in den Anfangsphasen, Durchführung von Aufgabenanalysen, iterativ-zyklische Vorgehensweise im Projektablauf sowie Einsatz von Prototyping zur Anforderungspräzisierung und zur Gestaltung der Benutzungsoberfläche. Prototypen eignen sich nicht nur als anschauliche 'Verhandlungsobjekte' für Diskussionen zwischen Benutzern und Entwicklern, sondern tragen auch zur besseren Strukturierung von Ideen der Entwickler bei. Auf diesen Erkenntnissen aufbauend wurden die Organisation benutzerorientierter Softwareentwicklungsprozesse in konkreten Entwicklungsprojekten untersucht und mitgestaltet. In einem Softwareprojekt zur Entwicklung und Einführung eines Bürokommunikationssystem war das mit traditionellem Vorgehen (Phasenmodell/keine Partizipation) entwickelte Produkt dem mit einem innovativen Vorgehen (Arbeitsanalysen, Prototyping, Benutzerbeteiligung) entwickelten System hinsichtlich Benutzerfreundlichkeit und Aufgabenangemessenheit deutlich unterlegen.

## **2. Verbesserung von Standardsoftware**

Die Zusammenarbeit zwischen dem Institut für Arbeitspsychologie (IfAP) an der ETH-Zürich und der ADI-Karlsruhe im Verbundprojekt "BOSS" zeigte, daß auch bei der (Weiter-) Entwicklung von Standardsoftware – das relationale Datenbanksystem ADIMENS (Mollenhauer 1989, 1990, 1991a, 1991b) – Benutzerbeteiligung erfolgreich praktiziert werden kann: Benutzer-Entwickler Workshops (Waeber 1990), verschiedene benutzungsorientierte Benchmarktests (mit echten Benutzern), eine Umfrage in einer Fachzeitschrift und Benutzertreffen trugen wesentlich zur software-ergonomischen Optimierung des Systems bei! Die Verbesserung verschiedener Veränderungen der grafischen Benutzungsoberfläche von ADIMENS konnte in *benutzungsorientierten Benchmarktests* aufgezeigt werden (Rauterberg 1989b, 1989c, 1990a, 1990b). Insbesondere durch den Umstieg auf grafische Benutzungsoberflächen und die Entwicklung von "aktiven Joins" konnte ein Großteil der Schwächen traditioneller Datenbanksysteme nachweislich überwunden werden (Rauterberg 1991a, 1991d, 1992d).

Einen weiteren Schwerpunkt des Verbundprojektes bildeten die empirischen Untersuchungen zu den verschiedenen Kriterien eines benutzungsorientierten Dialogdesigns. Die Transparenz von Benutzungsoberflächen läßt sich durch den Einsatz von Farbe unter bestimmten Bedingungen erhöhen (Rauterberg 1992c). Je höher die Flexibilität der Dialogstruktur ist, desto mehr wird das Programm den verschiedenen Aufgabenstellungen, als auch den unterschiedlichen individuellen Bedürfnissen und Anforderungen der Benutzer gerecht (Rauterberg 1992a). Um dem Softwareentwickler eine anwendungsnahe Umsetzung der Gestaltungskriterien zu ermöglichen, wurde das Beschreibungskonzept der "*Interaktionspunkte*" entwickelt (Rauterberg 1991e, 1993). Mit Hilfe dieses Konzeptes lassen sich verschiedene Gestaltungskriterien anwendungsnah beschreiben. Feldstudien zeigten, daß durch individuell angepaßte Oberflächen die Lösungsgüte von Textverarbeitungsaufgaben signifikant ansteigt (Spinas &

### "Phasenmodell ist OUT"

Rauterberg 1992). Dennoch ergab eine Umfrage unter 2000 BenutzerInnen eines Textverarbeitungsprogrammes, daß Angebote zur individuellen Gestaltung nur zu einem geringen Anteil genutzt werden.

### **3. Benutzungs-orientierte Benchmark-Tests (bBTs)<sup>2</sup>**

Benutzungs-orientierte Benchmark-Tests (bBTs) lassen sich in zwei Typen unterteilen: *induktive* und *deduktive bBTs* (Rauterberg 1990c, 1991b, 1991c). Die induktiven bBTs sind bei der Evaluation eines (z.B. vertikalen) Prototypen, oder einer (Vor)-Version zur Gewinnung von Gestaltungs- und Verbesserungsvorschlägen, bzw. zur Analyse von Schwachstellen in der Benutzbarkeit einsetzbar. Induktive bBTs können immer dann zum Einsatz kommen, wenn nur *ein* Prototyp, bzw. *eine* Version der zu testenden Software vorliegt. Demgegenüber verfolgen deduktive bBTs primär den Zweck, zwischen mehreren Alternativen (mindestens zwei Prototypen, bzw. Versionen) zu entscheiden. Zusätzlich lassen sich jedoch auch mit deduktiven bBTs Gestaltungs- und Verbesserungsvorschläge gewinnen (Rauterberg 1989a, 1990b, 1991a).

---

<sup>2</sup> bBT = benutzungs-orientierter Benchmark-Test; mit dem Adjektiv "benutzungs-orientiert" soll eine Verwechslung mit den sonst üblichen, rein system-technischen Benchmark-Tests vermieden werden.

PRODUKT	JAHR	MASSNAHME	ERGEBNIS
ADIMENS-ascii	1987	Kriterien-geleitete Softwareentwicklung	Entwicklung der Desktop-Oberfläche GT
ADIMENS-GT	1988	induktiver Benchmarktest "GT" (N=8)	Gestaltungsvorschläge -> GT+
		1. Entwickler Workshop	konsensfähiger Entwurf
		deduktiver Benchmarktest "GT <-> ASCII" (N=24)	Entscheidung zugunsten der Desktop-Oberflächen GT, GT+
ADIMENS-GT+ Vers. 3.0	1989	europaweite Umfrage im ST-Magazin (220 Rücksendungen)	Gestaltungsvorschläge -> GT+
		Benutzer-Entwickler Workshop	Gestaltungsvorschläge -> Desk
ADIMENS-GT+ Vers. 3.1	1990	Methodenvergleich für Ikon-Konstruktionen (N=194)	Vorgehensmodell
		2. Entwickler Workshop	konsensfähiger Entwurf -> Desk
ADIMENS-Desk	1991	deduktiver Benchmarktest "GT <-> GT+" (N=30)	empirische Prüfung der Gestaltungsvorschläge

**Abbildung 2:** Übersicht über die durchgeführten Workshops, Benchmark-Tests und benutzer-orientierte Umfragen (fett umrandete Maßnahmen) im Rahmen der Standardsoftware-Entwicklung von ADIMENS in den Jahren 1988 bis 1991.

### 3.1. Induktive benutzungs-orientierte Benchmark-Tests (ibBTs)

Bei der Durchführung eines induktiven bBTs sind die im folgenden aufgeführten Bedingungen zu beachten. Da die meisten Bedingungen zur Durchführung eines induktiven bBTs mit denen eines deduktiven bBTs übereinstimmen, werden dann später bei der Darstellung deduktiver bBTs nur noch die Änderungen und Ergänzungen erwähnt.

Um Artefakte bei der Benutzung - hervorgerufen durch unvollständige Systemfunktionalität - zu vermeiden, sollte im Bereich der zu testenden Systemfunktionen ein möglichst realitätsgerechtes Systemverhalten gegeben sein. Um die einzelnen Aktionen der Benutzer später auswerten zu können, empfiehlt es sich, eine automatische Aufzeichnung der Tastendrucke in das Test-System einzubauen.

Als erstes sollte man eine oder mehrere Benchmark-Aufgaben abgestimmt auf die zu testenden Systemteile festlegen. Diese Benchmark-Aufgaben sind dem typischen Aufgabenkontext des zukünftigen End-Benutzers zu entnehmen. Sofern dieser Aufgabenkontext nicht oder nur recht vage gegeben ist, sollten die Benchmark-Aufgaben zumindest jedoch typische Teil-Aufgaben enthalten. Eine einzelne Benchmark-Aufgabe sollte nicht zu komplex, aber auch nicht zu einfach sein; die Bearbeitungszeiten sollten

## "Phasenmodell ist OUT"

ungefähr zwischen fünf und fünfzehn Minuten liegen. Die Formulierung der Aufgaben sollte als schriftliche Fassung den Benutzern während der Aufgabenbearbeitung zugänglich sein. Je nach fach-spezifischem Vorwissen der Benutzer über den Aufgabenkontext ist es sinnvoll, die Aufgabenbeschreibung unterschiedlich abzufassen; bei hohem fach-spezifischem Vorwissen ist die Beschreibung möglichst *problem-orientiert* abzufassen, bei geringem, bzw. keinem fach-spezifischen Vorwissen ist die Beschreibung *handlungs-orientiert* zu halten (Rauterberg 1991b, 1991c). Die handlungs-orientierte Aufgabenbeschreibung soll verhindern helfen, daß die beobachteten Benutzungsprobleme überwiegend aufgrund fehlenden Fachwissens zustande gekommen sind.

Die Benutzergruppe sollte möglichst *repräsentativ* für die Population der End-Benutzer sein. Die Auswahl der Benutzer muß daher *zufällig* aus dem potentiellen oder aktuellen Benutzerkreis erfolgen. Die ausgewählten Benutzer sollten das zu testende System nicht kennen. Da sich diese Bedingung oftmals bei größeren Entwicklungsphasen, bzw. Weiterentwicklungen nicht einhalten läßt, muß die Vorerfahrung der Benutzer mit dem System, bzw. ähnlichen Systemen kontrolliert werden. Die Gruppengröße sollte aus statistischen Gründen zwischen sechs und zwanzig Personen liegen. Wenn kein oder nur sehr wenig Wissen über die Population der End-Benutzer vorliegt, lohnt es sich, die Benutzergruppe hinsichtlich der folgenden Parameter möglichst *heterogen* zusammenzusetzen: EDV-Vorerfahrung, Alter, Geschlecht, Ausbildung, Beruf, sowie der zu unterstützenden Aufgaben (Spinas et al. 1993).

Anzustreben ist eine möglichst den realen Einsatzbedingungen entsprechende Beobachtungs-Umgebung. Da es für die spätere Auswertung jedoch oftmals sehr wichtig ist, das Verhalten der einzelnen Benutzer sowie das entsprechende Systemverhalten auf Video, Tonband, "Screen-recording", etc. aufzuzeichnen, empfiehlt es sich, einen ruhigen, abgeschlossenen Raum mit entsprechendem Mobiliar zu benutzen. Stehen keine speziellen Aufzeichnungsgeräte zur Verfügung, sind für den Beobachter einfach auszufüllende Protokollbögen (z.B. mittels Strichlisten für vorgegebene Problem-, Fehler-Kategorien, etc.) schon sehr nützlich.

Der Beobachter sollte sich stets darum bemühen, jedem Benutzer das Gefühl der Wichtigkeit und Wertschätzung zu vermitteln. Die folgenden drei Aspekte sind für die Schaffung eines vertrauensvollen und für Kritik offenen Klimas hilfreich. Mit möglichst allgemein verständlichen Worten wird dem Benutzer Ziel und Zweck des bBTs erläutert (z.B. Test eines Prototypen in einem frühen Entwicklungsstadium, etc.). Es ist besonders wichtig, dem Benutzer verständlich zu machen, daß das System und *nicht* der Benutzer getestet werden soll. Jede Art von Schwierigkeiten seitens des Benutzers bei der Aufgabenbearbeitung sind von besonderem Interesse! Jedem Benutzer muß zugesichert werden, daß er die Durchführung des bBTs jederzeit unter- und sogar abbrechen kann, ohne daß irgendwelche negativen Konsequenzen (z.B. Wegfall einer zugesagten Bezahlung, etc.) erfolgen. Die vollständige Freiwilligkeit der Teilnahme und Zusicherung der Anonymität ist unabdingbar für die Gewinnung von brauchbaren Beobachtungen.

Jedem Benutzer werden alle für die Durchführung des Benchmark-Tests in dem Beobachtungsraum vorhandenen Geräte (Computer, Video-Kamera, Mikrophone, etc.) hinsichtlich Einsatzzweck und Funktionsweise erläutert. Als besonders wichtig erweist sich eine möglichst standardisierte Einführung in die Handhabungs- und Bedienungsweise der zu testenden Software (Rauterberg 1991c). Je nach Vorerfahrung des Benutzers können unterschiedliche Schwerpunkte gesetzt werden. Um die Motivation der Benutzer bei einer längeren Einführungsphase (in die Benutzung der Software) aufrechtzuerhalten, hat es sich als sinnvoll gezeigt, den Benutzer zur Generierung von

Frage- und Problemstellungen über die "Welt der Anwendungsobjekte" zu stimulieren und zur selbstständigen Exploration anzuregen. Dennoch muß unbedingt darauf geachtet werden, daß jeder Benutzer das gleiche Vorwissen erhält. Da diese Bedingung nur schwer zu kontrollieren ist, begnügt man sich oftmals mit Personen *ohne* jegliches EDV-Vorwissen.

Viele Benutzer haben Schwierigkeiten, ihre Gedanken während der Bearbeitung einer Aufgabe laut zu äußern. Es empfiehlt sich daher, dem Benutzer zu verdeutlichen, was mit "lautem Denken" gemeint ist, und mit ihm einige Übungsbeispiele gemeinsam durchzugehen. Trotzdem neigen Benutzer einerseits bei hoch routinisierten Handlungen, andererseits bei komplexen Problemlösungsprozessen dazu, das "laute Denken" einzustellen. Dies kann man durch folgende Maßnahmen umgehen: a) der Beobachter fordert den Benutzer in solchen Situationen zum "lauten Denken" auf; b) es werden *zwei* Benutzer als Paar mit der Aufgabebearbeitung betraut, wodurch die verbale Kommunikation zwischen beiden Partnern das "laute Denken" ersetzt; c) man führt nach Beendigung des bBTs dem Benutzer problematische Ausschnitte per Videoaufzeichnungen vor und bespricht mit ihm seine jeweiligen Ziele und Handlungsabsichten (Moll 1989).

Es empfiehlt sich, vor Beginn einer Beobachtungsserie, ein bis zwei Probe-Durchläufe zur Optimierung der gesamten Beobachtungsprozedur durchzuführen. Als bedeutsame Einflußgrößen auf die Art und Weise der Aufgabebearbeitung haben sich die EDV-Vorerfahrung der Benutzer und die Hilfestellungen durch den Beobachter herausgestellt (Rauterberg 1990b). Beide Variablen sollten gemessen werden, um sie später bei der statistischen Auswertung berücksichtigen zu können. Die Vorerfahrung läßt sich mittels Fragebogen erfassen. Die Art und die Anzahl der gegebenen Hilfestellungen des Beobachters wird von diesem während des bBTs auf einem Protokollbogen vermerkt.

Als "abhängige" Variablen werden alle erhobenen Meßwerte bezeichnet, welche bei der Auswertung Aufschluß über die Güte der Benutzbarkeit des zu testenden Systems Auskunft geben können. Die Menge der abhängigen Variablen teilt sich auf in die Menge der *qualitativen* Variablen (problematische Benutzungssituationen, handlungs-psychologische Bedingungskomplexe, etc.) und die Menge der *quantitativen* Variablen auf (Bearbeitungsdauer, Einlernzeit, Überlegungszeit, Anzahl Tastendrucke, Anzahl Fehler) (Rauterberg 1992e, 1992f).

Es gibt grundsätzlich zwei verschiedene Vorgehensweisen für die Gestaltung des zeitlichen Rahmens: 1. der Benutzer wird aufgefordert, die gestellte Aufgabe solange zu bearbeiten, bis er sie vollständig gelöst hat oder die Bearbeitung auf eigenen Wunsch abbricht; 2. dem Benutzer wird eine maximale Zeitspanne für die Aufgabebearbeitung zur Verfügung gestellt. Falls die Aufgabebearbeitungszeit als ein relevanter Indikator für Benutzungseigenschaften des zu testenden Systems herangezogen werden soll, empfiehlt sich die erste Variante. Bei der zweiten Variante muß man für die Auswertung den erreichten Lösungsgrad der Aufgabe bewerten. Diese Bewertung ist oft nicht einfach möglich.

Der Beobachter hält sich während der Aufgabebearbeitung ruhig im Hintergrund. Für ihn ist es sehr wichtig, seinen natürlichen Impuls, dem Benutzer in problematischen Situationen sofort zu helfen, "im Zaume zu halten". Hier kann eine klare Absprache zwischen Beobachter und Benutzer hilfreich sein: nur wenn sich der Benutzer explizit an den Beobachter wendet und um Hilfe nachfragt, wird der Beobachter aktiv. Ein zu frühes Eingreifen des Beobachters hindert den Benutzer, eine eigene Lösung zu finden. Als Beobachter sollten daher keine an der Entwicklung des zu testenden Systems unmittelbar

## "Phasenmodell ist OUT"

beteiligten Personen eingesetzt werden. Es wird sogar empfohlen, um eine möglichst realistische Beobachtungssituation herzustellen, daß der Beobachter sich völlig passiv verhält und dies dem Benutzer vorher deutlich macht.

Jedem Benutzer muß die Gelegenheit gegeben werden, sofern dies nicht schon im Rahmen der Video-Konfrontation eingeplant ist, für ihn wichtige und noch offene Fragen zu Zwecken und Zielen des bBTs, problematische Situationen während der Aufgabenbearbeitung, etc. in einem Gespräch nach Beendigung der Aufgabenbearbeitung mit dem Beobachter klären zu können. Meistens erhält man sehr zutreffende Problem-sichten aus diesen Gesprächen (Moll 1989). In dieser Nachbereitung lassen sich auch Fragebögen mit standardisierten oder offenen Fragen zur Beantwortung spezieller Benutzungsaspekte einsetzen.

Während der Durchführung eines bBTs wird man immer wieder überraschende und sehr informative Benutzungsweisen beobachten können. Es lohnt sich, diese auf Video aufzuzeichnen, um sie dann mit den Entwicklern später detailliert diskutieren zu können. Wichtig ist dabei, daß die beobachtbaren Schwierigkeiten niemals dem Benutzer, sondern ausschließlich der Benutzbarkeit des zu testenden Systems angelastet werden.

Die aufgezeichneten Beobachtungsdaten (Video, Tonband, Logfile, 'screenrecording') müssen hinsichtlich der design-relevanten Informationen ausgewertet werden. Dazu empfiehlt es sich, auf der Grundlage der durchgeführten Beobachtungen ein Auswertungsraster aufzustellen und dieses Raster systematisch auf die Beobachtungsdaten aller Benutzer anzuwenden. Aufwendig und schwierig sind die qualitativen Auswertungen von Logfile-Daten, weil es sich hierbei oftmals um komplexe Mustererkennungsprozesse handelt, welche bisher nur begrenzt automatisch ausführbar sind.

Als eine besonders informative Quelle für design-relevante Entscheidungen hat sich die Analyse von Videoaufzeichnungen ergeben. Hierbei werden Erklärungsmodelle für bestimmtes Benutzungsverhalten aufgestellt, um das beobachtbare Verhalten in problematischen Situationen handlungspsychologisch begründen zu können. Die eingehendere Beschäftigung mit diesen Situationen führt dann unter Berücksichtigung des Kriterienkonzeptes von Ulich (1985, 1986, 1991) zu konstruktiven, tragfähigen Gestaltungsvorschlägen (Rauterberg 1991d, 1992d).

Um die in der qualitativen Auswertung gewonnenen Erklärungsmodelle für einzelne Benutzungsprobleme auf eine möglichst breite Basis zu stellen (Problem der 'Generalisierbarkeit'), sind statistische Auswertungen unumgänglich. Diese können von einfachen Häufigkeitszählungen bestimmter Problemklassen, bis hin zu komplexen inferenz-statistischen Zusammenhangsanalysen gehen. Dazu ist es notwendig, daß die verschiedenen Benutzungseigenschaften (Dauer der Aufgabenbearbeitung, mittlere Überlegungszeit, Fehlerart, Ausmaß an Beanspruchung, etc.) pro Benutzer in quantifizierter Form vorliegen (Rauterberg 1992e, 1992f).

### **3.2. Deduktiver benutzungsorientierter Benchmark-Test (dbBT)**

Deduktive bBTs dienen primär der Entscheidungsfindung zwischen verschiedenen System-Alternativen, bzw. zur Kontrolle der erreichten Verbesserung (siehe zum Stichwort "Oberflächen-Effekt" bei Rauterberg 1991c) und erst sekundär der Gewinnung von

Gestaltungsvorschlägen. Es ergeben sich daher einige unterschiedliche Anforderungen an die Durchführung von deduktiven bBTs.

Im Gegensatz zu induktiven bBTs müssen die verschiedenen zu testenden, alternativen Systemversionen beim deduktiven bBT verstärkt der Forderung nach einem - an realen Einsatzbedingungen gemessenen - realistischen Systemverhalten (d.h. möglichst funktionale Vollständigkeit, adäquates Systemantwortzeitverhalten, etc.) genügen. Diese Anforderungen sind deshalb wichtig, weil die Entscheidung zwischen den System-Alternativen primär mittels quantitativer Meßgrößen gefällt wird. Je nachdem, ob viele Benutzer wenig Zeit haben (Fall-I), oder, ob wenige Benutzer viel Zeit haben (Fall-II), an einem bBT teilzunehmen, wird die Gruppenbildung unterschiedlich vorgenommen. Im Fall-I wird pro System-Alternative eine eigene Gruppe gebildet. Im Fall-II hat lediglich eine Gruppe alle System-Alternativen zu testen. Um Lerneffekte zu kontrollieren, muß dann die Reihenfolge der zu testenden Systeme innerhalb dieser einen Gruppe ausbalanciert werden. Im Fall-II müssen für jede System-Alternative *parallele* Benchmark-Aufgaben entwickelt werden (siehe als Beispiel Rauterberg 1991a, 1991c).

Die Gruppengröße sollte aus statistischen Gründen mindestens sechs Personen pro Gruppe betragen. Wenn man plausible Annahmen über den zu erwartenden Oberflächeneffekt hinsichtlich einer quantitativ zu messenden Benutzungseigenschaft (z.B. Dauer der Aufgabenbearbeitung, Anzahl Fehler, etc.) für die Alternativen hat, kann man die genau benötigte Gruppengröße auch ausrechnen. Die einzelnen Gruppen müssen hinsichtlich verschiedener Parameter möglichst homogen sein: EDV-Vorerfahrung, Geschlecht, Alter, Beruf. Mittels inferenzstatistischer Auswertungsverfahren können die gewonnenen Beobachtungsergebnisse auf ihre *Generalisierbarkeit* hin getestet werden (Rauterberg 1989b, 1989c, 1991a, 1991d, 1992a, 1992c, 1992d).

### **3.3. Ergebnisse von benutzungsorientierten Benchmarktests**

Weitere benutzungsorientierte Benchmarktests beschäftigten sich mit der Analyse, Bewertung und Gestaltung rechnergestützter Lern- und Arbeitshilfen. Die Ergebnisse zeigten, daß die Unterstützung der Benutzer durch ein system-orientiertes Hilfesystem nicht ausreichend war; die Entwicklung und Gestaltung *aufgabenorientierter Lern- und Arbeitshilfen* führte zu einer nachweislichen Verbesserung der Unterstützung (Moll 1989; Moll & Fischbacher 1989).

Im Jahre 1991 wurden zur weiteren Erforschung der "Individualisierbarkeit" in der Mensch-Computer Interaktion zwei bBTs und eine Feldstudie durchgeführt. Die Ergebnisse des ersten bBTs zeigten, dass die Resultate einer Datenbankrecherche bei der Benutzungsoberfläche mit einer individuell auswählbaren akustischen Rückmeldung (N=6) genauso gut waren wie bei der voreingestellten Standard-Benutzungsoberfläche ohne akustische Rückmeldung (N=6). Bei dem zweiten bBT wurde der Frage nachgegangen, ob und gegebenenfalls in welchem Ausmass Benutzer die Farbeinstellungen unter MsWindows während einer Textverarbeitungsaufgabe individuell ändern (N=17). Es zeigte sich, dass die individuell erreichte Farbeinstellung umso mehr von einer ergonomisch sinnvollen Konfiguration abwich, je öfter während der Untersuchung geändert wurde (Rauterberg 1992c).

Zu dem Kriterium "*Flexibilität*" wurde ein bBT mit 12 Informatik-Studenten der ETH durchgeführt, bei dem der Effekt der drei folgenden Interaktionsarten getestet wurde: (1) Interaktionssteuerung mit der "Maus", (2) Interaktionssteuerung mit den "Cursor-Tasten" und (3) über die einfach belegten "Funktionstasten". Es zeigte sich, dass die Bedingung

## "Phasenmodell ist OUT"

"Maus" und einfach belegte "Funktionstaste" aufgrund ihres geringeren interaktiven Aufwandes gegenüber der Bedingung "Cursor-Taste" signifikant besser abschnitt. Im Rahmen eines weiteren bBT zur Untersuchung des Oberflächeneffektes zwischen einer traditionellen Menüoberfläche (CUI, "character user interface") und einer grafischen Desktop-Oberfläche (GUI, "graphical user interface") konnte eine signifikante negative Korrelation zwischen dem Verhältnis der Bearbeitungszeit der CUI-Oberfläche zu der Bearbeitungszeit der GUI-Oberfläche und der Anzahl an möglichen Lösungsstrategien bei der GUI-Oberfläche gefunden werden; diese Korrelation bedeutet, dass die Benutzer der GUI-Oberfläche besonders dann gegenüber der CUI-Oberfläche im zeitlichen Vorteil waren, wenn ihnen möglichst verschiedene Lösungsalternativen von dem Dialogsystem zur Verfügung gestellt wurden (Rauterberg, 1992a).

Zu dem Kriterium "*Unterstützung*" wurde in einem Projekt die Analyse, Bewertung und Entwicklung rechnergestützter Lern- und Arbeitshilfen für ein interaktives "Numeric Control"-Programmiersystem, welches für die Programmierung und Steuerung von Werkzeugmaschinen eingesetzt wird, durchgeführt (Moll, 1989). Mit einer Methodenkombination (Standardaufgaben, "Logfile"-Aufzeichnungen, "lautes Denken", Videokonfrontation) wurde zunächst das Benutzerverhalten untersucht. Auswertungen des beobachteten Benutzerverhaltens zeigten, dass die Unterstützung der Benutzer durch das vorhandene systemorientierte Hilfesystem nicht ausreichend war. Deshalb wurde an der Gestaltung und Evaluation einer aufgabenorientierten, rechnergestützten Lernumgebung gearbeitet. Diese Lernumgebung ist im NC-Programmiersystem implementiert worden und wurde auf zwei Schulungskursen eingesetzt und evaluiert (Moll & Fischbacher 1989a, 1989b). Untersuchungen mit insgesamt 15 Benutzern (Werkzeugmacher) zeigten, dass sich die Unterstützung der Benutzer durch das neue aufgabenorientierte Hilfesystem signifikant verbessert hat (Moll 1989).

Zu den Kriterien "*Transparenz*" und "*Feedback*" wurden vier bBTs, eine elektronische Umfrage und eine Methodenvergleichsstudie über den Einsatz und die Verwendung von Farbbildschirmen durchgeführt. Farbige Benutzungsoberflächen wurden im Vergleich zu monochromen Benutzungsoberflächen insgesamt von den Benutzern subjektiv besser bewertet. Die Ergebnisse aus drei der vier dbBTs lassen eine objektive Überlegenheit farbiger Benutzungsoberflächen allerdings nur dann erkennen, wenn die Farbgebung handlungsleitend zur Erhöhung der Transparenz der Dialogstruktur eingesetzt wird (Rauterberg 1992c).

### **3.4. Ergebnisse von benutzungsorientierten Umfragen**

Eine Umfrage bei 61 Entwicklern, 65 Benutzern und 31 Führungskräften zeigte, daß etwa 70% der Befragten den Einbezug von Benutzern befürworteten und den Benutzern vermehrt Entscheidungskompetenzen zugestanden werden sollten. Unsicherheiten und unterschiedliche Vorstellungen zeigten sich allerdings in bezug auf ein adäquates Vorgehen (Spinas & Waeber 1991).

Im Jahre 1991 wurden zur weiteren Erforschung der "Individualisierbarkeit" in der Mensch-Computer Interaktion zwei Umfrageaktionen durchgeführt. Aus einer "electronic-mail"-Umfrage unter ETH-Angestellten (N=181; erste Umfrageaktion) erfuhren wir, dass ca. 90% aller Benutzer mit einem Farbbildschirm aus den verschiedensten Gründen die vorgegebene Farbeinstellung individuell anpassen. Aus den Ergebnissen des induktiven bBTs unter Berücksichtigung des "electronic-mail"-Umfrageergebnisses kann gefolgert werden, daß die individuelle Konfigurierbarkeit von farbigen Oberflächen auf eine Menge von vorgegebenen, ergonomisch vertretbaren Farbkombinationen ("configu-

ration sets") beschränkt und nicht der völlig freien Gestaltung durch den Benutzer überlassen bleiben sollte (Rauterberg 1992c).

Die oben schon erwähnte "electronic-mail"- Umfrage ergab, daß Monochrombildschirme primär bei Textbearbeitungssystemen (75% bei monochromen Anwendungen) und Farbbildschirme primär im Grafik/CAD-Bereich (84% bei farbigen Anwendungen) Verwendung finden.

Für die zweite Umfrageaktion wurde ein Fragebogen konstruiert, welcher die drei Dimensionen "*individuelle Anpassung*", "*individuelle Auswahl*" und "*Flexibilität*" bei der Nutzung des Textverarbeitungsprogrammes MsWord abfragt. Von den 2000 angeschriebenen MsWord Benutzern im süddeutschen Raum (Bayern und Baden-Württemberg) erhielten wir 559 beantwortete Fragebögen zurück. Die Analyse dieser Fragebogendaten ergab eine signifikante positive Korrelation zwischen dem Ausmass an individueller Nutzungsweise und der EDV-Vorerfahrung; d.h., je länger die Benutzer mit EDV Unterstützung ihre Aufgaben erledigen, desto stärker neigen sie dazu, sich die Benutzungsoberflächen an ihre individuellen Bedürfnisse anzupassen (Spinas & Rauterberg 1992). Aus allen 559 beantworteten Fragebögen dieser zweiten Umfrageaktion wurden acht "Hoch-Individualisierer" und acht "Standard"-Benutzer ausgewählt, welche den Hoch-Individualisierern ansonsten möglichst ähnlich waren. Diese 16 Benutzer wurden dann in einer quasi-experimentellen Feldstudie bei der Bearbeitung von drei Benchmarkaufgaben (Erstellung und Formatierung von Texten) in ihrer normalen Arbeitsumgebung (zu Hause mit ihrem eigenen PC) untersucht. Die Hoch-Individualisierer waren hinsichtlich der Bearbeitungszeit zwar genauso gut wie die Standardbenutzer, aber bei der Vollständigkeit der Aufgabenbearbeitung (Produktgüte) der zu erstellenden Textdokumente schnitten die Hoch-Individualisierer signifikant besser ab.

#### **4. Partizipationsmöglichkeiten**

Im Jahre 1990 wurde eine Methodenvergleichsstudie zwischen verschiedenen Verfahren zum Benutzereinbezug ("Partizipationsmethoden") durchgeführt. Wir wollten herausfinden, ob und gegebenenfalls inwieweit das Ergebnis eines neu zu gestaltenden Iconsatzes von der Art und Methode der Gestaltung abhängt. In dieser Vergleichsstudie wurde in einem ersten Schritt ein Icon-Satz für eine vollgrafische Datenbankoberfläche durch vier verschiedene Methoden entwickelt: (1) durch eine Expertengruppe (N=3), (2) durch eine Benutzergruppe (N=6), (3) durch eine Fragebogenaktion mit vorgegebenen Icons, bei der die Benutzer die Bedeutung des jeweils gezeigten Icons anzugeben hatten (N=140), und (4) durch eine Fragebogenaktion mit vorgegebener sprachlich formulierter Funktion, bei der die Benutzer das für die jeweilige Funktion passende Icon selbst malen mussten (N=45). In einem zweiten Schritt wurden dann die durch die vier verschiedenen Methoden gewonnenen Iconsätze einer Bewertung durch 94 Informatikstudenten der ETH, welche an der Erstellung der verschiedenen Iconsätze nicht beteiligt waren, auf ihre Interpretierbarkeit unterzogen. Es zeigte sich, daß diejenigen Icons, welche sowohl im Iconsatz der Expertengruppe, als auch im Iconsatz der Fragebogenaktion mit den vorgegebenen Icons gemeinsam vorkamen, insgesamt als am besten interpretierbar bewertet wurden.

Die Möglichkeiten partizipativer Software-Entwicklung sind vielfältig und reichen von schlichter Information der zukünftigen Benutzer und Rückmeldungen über schriftliche oder mündliche Umfragen, Durchführung von Workshops/Seminaren (Waeber 1990),

## "Phasenmodell ist OUT"

Mitwirkung von Benutzervertretern in Projektgruppen und anderen Gremien (Spinas 1992) bis hin zur Ausführung von Entwicklungsarbeiten durch die Benutzer selbst (z.B. Maskengestaltung mittels Maskengenerator). Die praktische Realisierung in konkreten Fällen ist vom Umfang, vom Inhalt und der Neuartigkeit des Vorhabens, der Anzahl der betroffenen Mitarbeiter sowie von den personellen und infrastrukturellen Voraussetzungen abhängig (Spinas et al. 1993). Die Effizienz des Entwicklungsprozesses und die Qualität des Produktes werden dabei maßgeblich von der Projektorganisation (Rollenverteilung, Kompetenzen, Informationsfluß, Ablaufkonzept usw.), den fachlichen und vor allem sozialen Qualifikationen der Beteiligten sowie den vorhandenen Methoden und Werkzeugen beeinflusst (Strohm 1991b, Strohm & Ulich 1991, Rauterberg & Strohm 1992).

### **5. Praktische Anforderungen**

Die gemachten Erfahrungen können im Hinblick auf eine benutzerorientierte Software-Entwicklung im Bereich von Standardsoftware aus der Sicht des entwickelnden Softwarehauses zu drei miteinander in Wechselwirkung stehenden Anforderungen zusammengefaßt werden (Mollenhauer 1990).

1. In der Aufbauorganisation des Softwarehauses sind Maßnahmen zu treffen, die über die marktnotwendige Kundenorientierung und damit verbundene verkaufsfördernde Strategien zum Absatz der entwickelten Produkte hinausgehen in Bezug auf: (a) die Förderung der Orientierung am Benutzer in *allen* Unternehmensbereichen; (b) die Auffächerung des Kundenkontaktes; sowie (c) die Verfügbarkeit einer Ablauforganisation für die Resultate der Benutzerbeteiligung.

2. Benutzerorientierte Software-Entwicklung benötigt für ihre konsequente Umsetzung ein Qualitätssicherungssystem, das durch hinreichende Systematisierung die fortlaufende Verbesserung der Produkte mit dem Ziel einer menschengerechten Gestaltung von Software fördert.

3. Als wesentliche Instrumente der hier erprobten Software-Entwicklungsmethode, die eine benutzerorientierte Software-Entwicklung im Detail gewährleisten, sind zu nennen: (a) Benutzerprofil und -kontakt; (b) Mitarbeiterbeteiligung; (c) Produktqualität.

Die entstandenen Strukturen und Instrumente können neben Bewertungs- und Gestaltungsschemata ein Software-Engineering für eine menschengerechte Gestaltung von Software unterstützen. Die gemachten Erfahrungen zeigen deutlich, daß Innovationen unter einer ganzheitlichen Sichtweise erforderlich sind. Einer Sichtweise, die die BenutzerInnen im Rahmen aller genannten Maßnahmen als autonomes, sich qualifizierende Subjekte sieht (Ulich 1991).

### **6. Zusammenfassung**

Die Methode der benutzungs-orientierten Benchmark-Tests (bBTs) läßt sich nicht nur zur Gewinnung von Gestaltungsvorschlägen (induktive bBTs), sondern auch zur Entscheidung zwischen Systemalternativen, bzw. zur Überprüfung von getroffenen Designentscheidungen (deduktive bBTs) sinnvoll im Rahmen der partizipativen Entwicklung von Standardsoftware einsetzen. Sehr verkürzt lassen sich so die Ergebnisse des vom BMFT geförderten und jetzt abgeschlossenen Forschungsprojektes

zur "Entwicklung und empirischen Überprüfung von Kriterien, Methoden und Modellen zur benutzerorientierten Software-Entwicklung und Dialoggestaltung", kurz "BOSS" darstellen. Denn die gemeinsam über die letzten 5 Jahre zwischen der Karlsruher ADI Software und dem Institut für Arbeitspsychologie der ETH Zürich durchgeführten Forschungsarbeiten konnten neue Alternativen aufzeigen und erproben, die in einem praxisorientierten Leitfadens für Software-Entwickler und Organisatoren dargestellt werden. In naher Zukunft lautet die Frage nicht mehr, ob Benutzer beteiligt werden sollen, sondern wie durch Benutzerbeteiligung das Fachwissen der Benutzer zur Entwicklung guter, aufgaben- und benutzerorientierter Software genutzt werden kann!

### **Danksagung**

Im Forschungsprojekt "BOSS" zur "Entwicklung und empirischen Überprüfung von Kriterien, Methoden und Modellen zur benutzerorientierten Software-Entwicklung und Dialoggestaltung" haben mitgewirkt: J. Dobrinski, D. Geiss, J. Geiss, C. Heer, M. Lüders, T. Moll, R. Mollenhauer, H. Nibel, M. Rauterberg, C. Rolfsen, K. Schlagenhaut, P. Spinass, O. Strohm, K. Thalmann, E. Ulich, D. Waerber und eine Vielzahl von InformatikstudentInnen der ETH-Zürich. Alle diese Personen haben zum Gelingen des Forschungsprojektes einen wesentlichen Beitrag geleistet. Ohne die Förderung durch das BMFT und die Betreuung durch das AuT-Programm wäre dieses Projekt nicht möglich gewesen. Dafür möchten wir allen Beteiligten an dieser Stelle ganz herzlich danken.

### **Literaturangaben zum Forschungsprojekt BOSS**

- Baitsch, C., Katz, C., Spinass, P. & Ulich, E. (1991) Computerunterstützte Büroarbeit. Ein Leitfaden für Organisation und Gestaltung. 2. Auflage. Reihe Arbeitswelt (Hrsg. A. Alioth), Band 8. Zürich: Verlag der Fachvereine
- Geiss, D. (1987) Ein endbenutzerorientiertes Datenbanksystem auf der grafischen Benutzerschnittstelle GEM. unveröffentlichte Diplomarbeit. Fachbereich Informatik. Universität Karlsruhe.
- Moll, T. (1989) Unterstützung von Softwarebenutzern: Aufgaben- versus systemorientierte Lern- und Arbeitshilfen. Unveröff. Diss. Bern: Universität.
- Moll, T. & Fischbacher, U. (1989a) Über die Verbesserung der Benutzerunterstützung durch ein Online-Tutorial. In S. Maass & H. Oberquelle (Hrsg.), Software-Ergonomie 89: Aufgabenorientierte Systemgestaltung und Funktionalität (S. 223-232). Stuttgart: Teubner.
- Moll, T. & Fischbacher, U. (1989b) Online assistance: The development of a help-system and an online tutorial. In G. Salvendy & M.J. Smith (Eds.), Designing and using human-computer interfaces and knowledge based systems (pp. 97-104). Amsterdam: Elsevier.
- Mollenhauer, R. (1989) Das Adimens Praxis-Buch zum Atari-St. München: Markt & Technik.
- Mollenhauer, R. (1990) Benutzerorientierte Softwareentwicklung im Rahmen der Weiterentwicklung der Standardsoftware Adimens. In: H. Bullinger (Hrsg.), Software-Ergonomie in der Praxis (S. 157-168). Berlin: Springer.
- Mollenhauer, R. (1991a) Benutzerorientierte Softwareentwicklung des vollgrafischen Datenbanksystems Adimens. In: M. Frese, C. Karsten, C. Skarpelis & B. Zang-Scheucher (Hrsg.), Software für die Arbeit von morgen. Bilanz und Perspektiven anwendungsorientierter Forschung. Ergänzung zum Tagungsband (S. 353-360). Krefeld: Vennekel & Partner.
- Mollenhauer, R. (1991b) Benutzerorientierte Softwareentwicklung eines vollgrafischen Datenbanksystems. In: M. Frese, C. Karsten, C. Skarpelis & B. Zang-Scheucher (Hrsg.), Software für die Arbeit von morgen. Bilanz und Perspektiven anwendungsorientierter Forschung. Ergänzung zum Tagungsband (S. 400-401). Krefeld: Vennekel & Partner.
- Rauterberg, M. (1989a) ADIMENS ST und ADIMENS ST Plus: Ein qualitativer Vergleich nach software-ergonomischen Kriterien. Adimens News Letter, Nr. 3, 1-6.
- Rauterberg, M. (1989b) Ein empirischer Vergleich einer desktop- mit einer menüorientierten Benutzungsoberfläche für ein relationales DBMS. In: M. Paul (Hrsg.), Proceedings der GI-Jahrestagung 1989. Informatik Fachberichte Nr. 222, Band 1 (S. 243-258). Berlin: Springer.
- Rauterberg, M. (1989c) Maus versus Funktionstaste: Ein empirischer Vergleich einer desktop- mit einer ascii-orientierten Benutzungsoberfläche. In: S. Maass & H. Oberquelle (Hrsg.), Software-Ergonomie 89: Aufgabenorientierte Systemgestaltung und Funktionalität (S. 313-323). Stuttgart: Teubner.

- Rauterberg, M. (1990a) Ein empirischer Vergleich einer direkt-manipulativen mit einer ascii-orientierten Benutzungsoberfläche für ein relationales Datenbanksystem. In: Gesellschaft für Arbeitswissenschaft (Hrsg.), Bericht zum 36. Arbeitswissenschaftlichen Kongress Zürich (S. 30-31). Köln: Otto Schmidt.
- Rauterberg, M. (1990b) Experimentelle Untersuchungen zur Gestaltung der Benutzungsoberfläche eines relationalen Datenbanksystems. In: P. Spinass, M. Rauterberg, O. Strohm, D. Waeber, & E. Ulich (Hrsg.), Projektbericht Nr. 3 zum Forschungsprojekt BOSS 'Benutzerorientierte Softwareentwicklung und Schnittstellengestaltung'. Zürich: ETH, Institut für Arbeitspsychologie.
- Rauterberg, M. (1990c) Über die Notwendigkeit des Einsatzes psychologischer Methoden zur Lösung von Gestaltungsproblemen in der Informatik. In: D. Frey (Hrsg.), Bericht über den 37. Kongress der Deutschen Gesellschaft für Psychologie in Kiel 1990, Band 1 (S. 111-112). Göttingen: Hogrefe.
- Rauterberg, M. (1991a) Benutzungorientierte Benchmark-Tests: Anwendung auf die Gestaltung der Benutzungsoberfläche von ADIMENS GT hin zu GT+. In: M. Rauterberg & E. Ulich (Hrsg.), Posterband zur Software-Ergonomie'91 (S. 151-162). Zürich: ETH, Institut für Arbeitspsychologie.
- Rauterberg, M. (1991b) Benutzungorientierte Benchmark-Tests: eine Methode zur Benutzerbeteiligung bei Standardsoftware-Entwicklungen. In: D. Ackermann & E. Ulich (Hrsg.), German Chapter of the ACM Berichte Nr. 32 "Software-Ergonomie'91" (S. 96-107). Stuttgart: Teubner.
- Rauterberg, M. (1991c) Benutzungorientierte Benchmarktests: Eine Methode zur Benutzerbeteiligung bei der Entwicklung von Standardsoftware. In: P. Spinass, M. Rauterberg, O. Strohm, D. Waeber & E. Ulich (Hrsg.), Projektberichte zum Forschungsprojekt "Benutzerorientierte Softwareentwicklung und Schnittstellengestaltung" Nr. 6 (S. 1-62). Zürich: ETH, Institut für Arbeitspsychologie.
- Rauterberg, M. (1991d) From passive to active joins: A task oriented desktop interface conception for a relational DataBaseManagement-System. Proceedings of the Conference "Intelligent Text and Information Handling RIAO 91" Barcelona (Spain) - April 2-5,1991. Vol. 2 (pp. 809-821).
- Rauterberg, M. (1991e) Interaktive Aufsetzpunkte: Ein Konzept zur Beschreibung und Klassifizierung von Benutzungsoberflächen interaktiver Software. In: M. Frese, C. Kasten, C. Skarpelis & B. Zang-Scheucher (Hrsg.), Ergänzung zum Tagungsband "Software für die Arbeit von morgen" (S. 21-34). Bonn: BMFT-DLR/ Projekträgerchaft "AuT".
- Rauterberg, M. (1991f) Partizipative Konzepte, Methoden und Techniken zur Optimierung der Softwareentwicklung. In: P. Brödner, G. Simonis & H. Paul (Hrsg.), Arbeitsgestaltung und partizipative Systementwicklung (S. 95-125). Opladen: Leske & Budrich.
- Rauterberg, M. (1992a) An empirical comparison of menu-selection (CUI) and desktop (GUI) computer programs carried out by beginners and experts. Behaviour and Information Technology. 11(4):227-236.
- Rauterberg, M. (1992b) An iterative-cyclic software process model. In: Proceedings of 4th. International Conference on Software Engineering and Knowledge Engineering held in Capri (Italy), June 17-19, 1992; Los Alamitos: IEEE Computer Society Press; pp. 600-607.

## "Phasenmodell ist OUT"

- Rauterberg, M. (1992c) Der Einsatz von Farbe bei der Gestaltung von Benutzungsoberflächen in der Mensch-Computer Interaktion. Zeitschrift für Arbeitswissenschaft. 46(18 NF/4):233-242.
- Rauterberg, M. (1992d) Designing and testing the usability of a relational data base system with end-user oriented benchmark tasks. In: P. Brödner & W. Karwowski (eds.), Ergonomics of Hybrid Automated System III (pp. 301-306). Amsterdam London Tokyo: Elsevier.
- Rauterberg, M. (1992e) Lässt sich die Gebrauchstauglichkeit interaktiver Software messen? Und wenn ja, wie?. Ergonomie & Informatik. 16:3-18.
- Rauterberg, M. (1992f) Messung der Gebrauchstauglichkeit interaktiver Software. In: W. Görke, H. Rininsland & M. Syrbe (Hrsg.), Information als Produktionsfaktor (S. 211-221). Berlin Heidelberg New York: Springer.
- Rauterberg, M. (1992g) Optimisation Cycle: a Concept for Optimal Software Development. In: R. Trappl (ed.), Cybernetics and System Research, vol.1 (pp.279-286). Singapore London: World Scientific.
- Rauterberg, M. (1992h) Partizipative Modellbildung zur Optimierung der Softwareentwicklung. In: R. Studer (Hrsg.), Informationssysteme und Künstliche Intelligenz: Modellierung (S. 113-128). Berlin : Springer.
- Rauterberg, M. (1992i) Quantitative measures to evaluate human-computer interfaces. In: H. Luczak, A. Cakir & G. Cakir (eds.), Abstract Book of 3rd.International Scientific Conference on "Work With Display Units (WWDU)" (pp. E45-E46). Berlin: Technische Universität - Institut für Arbeitswissenschaften.
- Rauterberg, M. (1993, in press) A product oriented approach to quantify usability attributes and the interactive quality of user interfaces. In: H. Luczak, A. Cakir & G. Cakir (eds.), Work With Display Units (WWDU). Amsterdam: Elsevier.
- Rauterberg, M. & Strohm, O. (1992) Work Organization and Software Development. In: P. Elzer & V. Haase (eds.), Proceedings of 4th IFAC/IFIP Workshop on "Experience with the Management of Software Projects". Annual Review of Automatic Programming. Vol. 16 (2), Pergamon Press.
- Spinas, P. & Rauterberg, M. (1992) Von der Listenverarbeitung zur interaktiven, computergestützten Sachbearbeitung am Bildschirm. In: Institut für Arbeitspsychologie (Hrsg.), Arbeitspsychologie an der ETH Zürich 1972-1992, eine Zwischenbilanz (S. 62-70). ETH-Zürich: Institut für Arbeitspsychologie.
- Spinas, P. & Waeber, D. (1991) Benutzerbeteiligung aus der Sicht von Endbenutzern, Softwareentwicklern und Führungskräften. In: D. Ackermann & E. Ulich (Hrsg.), Software-Ergonomie '91. Benutzerorientierte Software-Entwicklung (S. 36-45). Stuttgart: Teubner.
- Spinas, P. (1989) User oriented software development and dialogue design. In: M.J. Smith & G. Salvendy (Eds.), Work with computers: Organizational, management, stress and health aspects (pp. 200-207). Amsterdam: Elsevier.
- Spinas, P. (1990) Benutzerfreundlichkeit von Dialogsystemen und Benutzerbeteiligung bei der Software-Entwicklung. In: F. Frei & I. Udris (Hrsg.), Das Bild der Arbeit (S. 158-171). Bern: Huber.
- Spinas, P. (1992) Benutzerfreundlichkeit von ETHICS: Ergebnisse einer Untersuchung des Online-Kataloges der ETH-Bibliothek. ABI-Technik, 12, 55-59.

- Spinas, P. (1993, in Druck) Benutzerbeteiligung bei der Software-Entwicklung. In: Brodbeck, F. & Frese, M. (Hrsg.), Software-Entwicklung zwischen Anforderung und Realität. Giessen.
- Spinas, P., Rauterberg, M., Strohm, O., Waeber, D. & Ulich, E. (1993, in Druck). Benutzerorientierte Software-Entwicklung. Konzepte, Methoden und Vorgehen zur Benutzerbeteiligung. Schriftenreihe Mensch, Technik, Organisation (Hrsg. E. Ulich), Band 3. Zürich: Verlag der Fachvereine, Stuttgart: Teubner.
- Spinas, P., Waeber, D. & Strohm, O. (1990) Kriterien benutzerorientierter Dialoggestaltung und partizipative Softwareentwicklung: Eine Literaturlaufarbeitung. In: P. Spinas, M. Rauterberg, O. Strohm, D. Waeber & E. Ulich (Hrsg.), Projektbericht Nr. 1 zum Forschungsprojekt BOSS 'Benutzerorientierte Softwareentwicklung und Schnittstellengestaltung'. Zürich: ETH, Institut für Arbeitspsychologie.
- Strohm, O. (1990a) Arbeitsorganisation, Methodik und Benutzerorientierung bei der Softwareentwicklung. In: P. Spinas, M. Rauterberg, O. Strohm, D. Waeber & E. Ulich (Hrsg.), Projektbericht Nr. 2 zum Forschungsprojekt BOSS 'Benutzerorientierte Softwareentwicklung und Schnittstellengestaltung'. Zürich: ETH, Institut für Arbeitspsychologie.
- Strohm, O. (1990b) Benutzerorientierung und Probleme bei der Softwareentwicklung. Output, 7, 27-31.
- Strohm, O. (1991a) Arbeitsorganisation, Methodik und Benutzerorientierung bei der Software-Entwicklung. In: M. Frese, C. Karsten, C. Skarpelis & B. Zang-Scheucher (Hrsg.), Software für die Arbeit von morgen. Bilanz und Perspektiven anwendungsorientierter Forschung (S. 431-441). Berlin: Springer.
- Strohm, O. (1991b) Projektmanagement bei der Softwareentwicklung. In: D. Ackermann & E. Ulich (Hrsg.), Software-Ergonomie '91. Benutzerorientierte Software-Entwicklung (S. 46-58). Stuttgart: Teubner.
- Strohm, O. (1993, in Druck) Arbeitsorganisation bei der Software-Entwicklung - arbeitspsychologische Konzepte und empirische Befunde. In: F. Brodbeck & M. Frese (Hrsg.), Software-Entwicklung zwischen Anforderung und Realität. Giessen.
- Strohm, O. & Ulich, E. (1991) Arbeitsteilung und Benutzerorientierung bei der Software-Entwicklung. In: P. Elzer (Hrsg.), Multidimensionales Software-Projektmanagement (S. 261-289). Hallbergmoos: AIT Angewandte Informationstechnik.
- Ulich, E. (1985) Einige Anmerkungen zur Software-Psychologie. sysdata. Nr. 10, S. 53-58.
- Ulich, E. (1986) Aspekte der Benutzerfreundlichkeit. In: W. Remmele & M. Sommer (Hrsg.), Arbeitsplätze von morgen. Berichte des German Chapter of the ACM, Band 27. (S. 102-121), Stuttgart: Teubner.
- Ulich, E. (1989a) Arbeitspsychologische Konzepte der Aufgabengestaltung. In: S. Maass & H. Oberquelle (Hrsg.), Software-Ergonomie '89: Aufgabenorientierte Systemgestaltung und Funktionalität (S. 51-65). Stuttgart: Teubner.
- Ulich, E. (1989b) Bürotechnologie zwischen Effizienz und Menschlichkeit. In: M. Rist (Hrsg.), Gesichter des technischen Fortschritts (S. 52-65). Zürich: Verlag Neue Zürcher Zeitung.

## "Phasenmodell ist OUT"

- Ulich, E. (1990) Towards the design of user-oriented dialogue-systems: Experiments. In: Dy, F.J. (Ed.), *New technologies in commerce and offices. A study prepared for the ILO* (pp. 54-56). Sydney: Avebury.
- Ulich, E. (1991) *Arbeitspsychologie*. Stuttgart: Poeschel-Verlag.
- Ulich, E. & Spinass, P. (1990) Arbeitspsychologische Konzepte für die Arbeit mit Bildschirmssystemen. *Die Volkswirtschaft*, 7, 18-21.
- Ulich, E., Rauterberg, M., Moll, T., Greutmann, T. & Strohm, O. (1991) Task orientation and User-Oriented Dialog Design. *International Journal of Human-Computer Interaction*, 3 (2), 117-144.
- Waeber, D. (1989) User oriented software development and dialogue design. *Sammelband der Referate des 1. Kongresses der Schweizerischen Gesellschaft für Psychologie* (S. 286-292). Bern: Schweizerische Gesellschaft für Psychologie.
- Waeber, D. (1990) Entwicklung und Umsetzung von Modellen partizipativer Softwareentwicklung. In: P. Spinass, M. Rauterberg, O. Strohm, D. Waeber & E. Ulich (Hrsg.), *Projektbericht Nr. 4 zum Forschungsprojekt BOSS 'Benutzerorientierte Softwareentwicklung und Schnittstellengestaltung'*. Zürich: ETH, Institut für Arbeitspsychologie.
- Waeber, D. (1991) Aufgabenanalyse und Anforderungsermittlung in der Softwareentwicklung: Zur Konzeption einer integrierten Entwurfsstrategie. In: M. Frese, C. Karsten, C. Skarpelis & B. Zang-Scheucher (Hrsg.), *Software für die Arbeit von morgen. Bilanz und Perspektiven anwendungsorientierter Forschung. Ergänzung zum Tagungsband* (S. 35-45). Krefeld: Vennekel & Partner.
- Waeber, D. & Spinass, P. (1989) Partizipative Softwareentwicklung und benutzerorientierte Dialoggestaltung. *Softwaretechnik-Trends. Mitteilungen der Fachgruppe "Software-Engineering" der Gesellschaft für Informatik*. Band 9, Heft 2, 32-41.