

Benutzungs-orientierte Benchmark-Tests: eine Methode zur Benutzerbeteiligung bei Standardsoftware-Entwicklungen¹

Matthias Rauterberg

Zusammenfassung

Es werden zwei Methoden - induktive und deduktive benutzungsorientierte Benchmark Tests - zur partizipativen Entwicklung von Standardsoftware vorgestellt, mit denen es möglich ist, unter Verwendung von Benchmark-Aufgaben das zu bewertende interaktive System durch ausgewählte Benutzer zu testen. Diese beiden Methoden wurden im Rahmen einer mehrjährigen Standardsoftware-Weiterentwicklung eines relationalen Datenbankprogrammes erfolgreich angewendet.

1. Einleitung

Es lassen sich vier Arten von Software-Entwicklungsprojekten unterscheiden (STROHM 1990; WAEBER 1990): 1. spezifische Anwendungen für firmeninterne Fachabteilungen (*Typ-A*); 2. spezifische Anwendungen für externe Kunden (*Typ-B*); 3. Standardbranchenlösungen für externe Kunden (*Typ-C*); 4. Standardsoftware für einen anonymen Kundenkreis (*Typ-D*). Während bei Typ-A bis C die potentiellen Benutzer weitgehend bekannt sind, müssen bei Typ-D neue Wege der Benutzerbeteiligung beschritten werden. Es wird nun eine Möglichkeit vorgestellt, aus Ergebnissen empirischer Studien generalisierbare Aussagen in Form von benutzer-orientierten Gestaltungsvorschlägen auf der Grundlage des Kriterienkonzeptes von ULICH (1985, 1986, 1991) ableiten zu können (RAUTERBERG 1988, 1990b).

2. Stand der Forschung zur Benutzerbeteiligung

Je nach Form, Grad, Inhalt, etc. der Benutzerbeteiligung (ACKERMANN 1988, SPINAS 1990) werden schon heute schon bei der Entwicklung von Standardsoftware ansatzweise Benutzer beteiligt (STROHM 1990). Die am häufigsten eingesetzten Methoden sind: 'hot-line', alpha-, beta-Tests, 'user group'-Workshops, Kontakt zu speziellen Kunden, etc. Diese Methoden dienen jedoch fast ausschließlich der Überprüfung auf funktionale Vollständigkeit und Korrektheit. Benutzungsprobleme im handlungs- und arbeitspsychologischen Sinne werden dabei kaum berücksichtigt. Hier setzt die Methode der *benutzungs-orientierten Benchmark-Tests*² an. Benutzerbeteiligung ist deshalb notwendig, weil bestimmte Eigenschaften interaktiver Systeme nur in der konkreten Interaktion meßbar sind (DZIDA 1990, S.10).

¹ Die Ergebnisse dieser Studie sind im Rahmen des vom BMFT geförderten Forschungsprojektes "Benutzerorientierte Software-Entwicklung und Schnittstellengestaltung (BOSS)" am Institut für Arbeitspsychologie (IfAP) der ETH-Zürich im Verbund mit der ADI-GmbH aus Karlsruhe gewonnen worden.

² zur Verwendung des Begriffes "Benchmark" im Kontext von Benutzerbeteiligung siehe: WILLIGES, WILLIGES & ELKERTON (1987, S. 1434); WHITESIDE, BENNETT & HOLTZBLATT (1988, S. 796); MACAULAY et al. (1990, S. 109).

3. Benutzerbeteiligung bei Standardsoftware-Entwicklungen

Als ein besonderes Problem im Zusammenhang mit der Beteiligung von Benutzern bei Standardsoftware-Entwicklungen muß die große *Heterogenität des Benutzerkreises* angesehen werden. Um nun benutzungs-orientierte Benchmark-Tests durchführen zu können, sollte das zu testende System bestimmte Voraussetzungen erfüllen (siehe weiter unten bei der Beschreibung der Methode). Dies führt dazu, daß benutzungs-orientierte Benchmark-Tests vorwiegend im Entwicklungs- oder Lebenszyklus des Standardsoftware-Produktes eingesetzt werden (siehe Abb. 1).

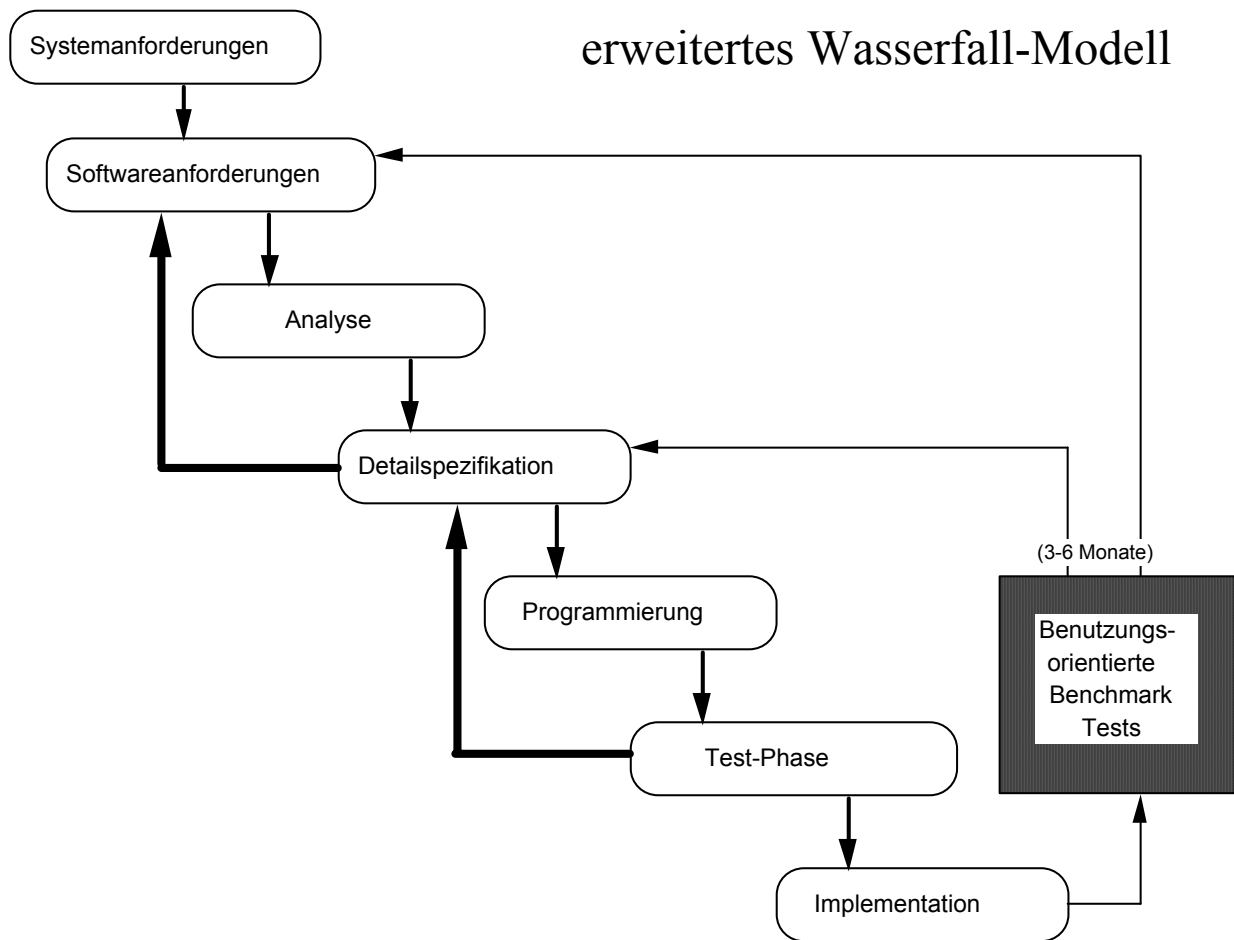


Abbildung 1: Das traditionelle "Wasserfall"-Modell der Software-Entwicklung (nach BOEHM 1981) erweitert um Rückkopplungszyklen über benutzungs-orientierte Benchmark-Tests (siehe auch WILLIGES, WILLIGES & ELKERTON 1986, S. 1418).

Um ein möglichst repräsentatives Abbild der verschiedenen Benutzungskontexte von Standardsoftware-Produkten zu erhalten, lassen sich Befragungen mittels *Fragebögen* unter dem registrierten Benutzerkreis durchführen (KLAS, KUICH & LAES 1990; HUNZIGER & HÄSSIG 1990; STUTZ et al. 1990). In einer verkürzten Form können auch wichtige Informationen über eine entsprechend erweiterte Version der *Registrierkarten* fortlaufend erhalten werden (MOLLENHAUER 1991). Um End-Benutzer möglichst direkt und unmittelbar einzubeziehen, können *Be-*

nutzer-Entwickler-Treffen durchgeführt werden (WAEBER 1990). Bei reinen Neuentwicklungen lassen sich im Rahmen von Marktanalysen auch stichprobenartig Befragungen (mittels Fragebogen oder Interview) unter dem potentiellen Benutzerkreis durchführen.

4. Benutzungs-orientierte Benchmark-Tests (bBTs)¹

Benutzungs-orientierte Benchmark-Tests (bBTs) lassen sich in zwei Typen unterteilen: *induktive* und *deduktive bBTs* (RAUTERBERG 1990a). 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 (KARAT 1988, S. 894). Zusätzlich lassen sich jedoch auch mit deduktiven bBTs Gestaltungs- und Verbesserungsvorschläge gewinnen.

4.1. Induktive benutzungs-orientierte Benchmark-Tests (bBTs)

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.

4.1.1. Vorgehensweise bei induktiven bBTs

4.1.1.1. Anforderungen an die zu testende Software

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 (MOLL 1987, MÜLLER-HOLZ et al. 1991).

4.1.1.2. Gestaltung der Beobachtungssituation

Konstruktion der Benchmark-Aufgaben:

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

¹ bBT = benutzungs-orientierter Benchmark-Test; mit dem Adjektiv "benutzungs-orientiert" soll eine Verwechslung mit den sonst üblichen, rein system-technischen Benchmark-Tests vermeiden werden.

einfach sein; die Bearbeitungszeiten sollten ungefähr zwischen fünf und fünfzehn Minuten liegen. Die Formulierung der Aufgaben sollte als schriftliche Fassung den Benutzern während der Aufgabebearbeitung 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*¹ abzufassen, bei geringem, bzw. keinem fach-spezifischen Vorwissen ist die Beschreibung *handlungs-orientiert*² zu halten. Die handlungs-orientierte Aufgabenbeschreibung soll verhindern helfen, daß die beobachteten Benutzungsprobleme überwiegend aufgrund fehlenden Fachwissens zustande gekommen sind.

Auswahl der Benutzer:

Die Benutzergruppe sollte möglichst *repräsentativ* für die Population der End-Benutzer sein (siehe CAMPBELL & STANLEY 1970). 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 (siehe dazu mehr im Abschnitt zur "Messung von intervenierenden Variablen"). 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.

Umgebungsbedingungen während der Beobachtung:

Anzustreben ist eine möglichst den realen Einsatzbedingungen entsprechende Beobachtungs-Umgebung (siehe zum Thema "kontext-sensitive Beobachtung" WHITESIDE, BENNETT & HOLTZBLATT 1988). 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 (siehe VOSSEN 1991), 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.

¹ z.B. "Erstellen Sie bitte einen Brief mit folgendem Inhalt: ... und bereiten Sie ihn zum Eintüten und Versenden an folgende Adressen: ... vor. Benutzen Sie dabei bitte als Briefvorlage das Dokument mit dem Namen: ..."

² z.B. "Laden Sie bitte das Textdokument mit folgendem Namen: ... und ergänzen es um den folgenden Inhalt: ... Versehen Sie dieses Textdokument mit allen für die Serienbriefferstellung notwendigen Steueranweisungen. ... usw."

4.1.1.3. Vorbereitung der Benutzer

Der Beobachter sollte sich stets darum bemühen, jedem Benutzer das Gefühl der Wichtigkeit und Wertschätzung zu vermitteln (WEISER & SHNEIDERMAN 1987, S. 1401). Die folgenden drei Aspekte sind für die Schaffung eines vertrauensvollen und für Kritik offenen Klimas hilfreich.

Beschreibung von Ziel und Zweck:

Mit möglichst allgemein verständlichen Worten wird dem Benutzer Ziel und Zweck des bBTs erläutert (z.B. Test eines Prototypen in einem frühen Entwicklungsstadium, etc.)¹. 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!

Das Recht des Benutzers auf Beendigung:

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 Zusage der Anonymität ist unabdingbar für die Gewinnung von brauchbaren Beobachtungen.

Vertrautmachen mit der Untersuchungssituation:

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. 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.

Die Technik des "lauten Denkens":

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

¹ für weitere Formulierungsvorschläge siehe GOMOLL (1990, S. 87).

"lauten Denken" auf; b) es werden *zwei* Benutzer als Paar mit der Aufgabenbearbeitung 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 1987).

4.1.1.4. Durchführung der Beobachtung

Es empfiehlt sich, vor Beginn einer Beobachtungsserie, ein bis zwei Probe-Durchläufe zur Optimierung der gesamten Beobachtungsprozedur durchzuführen.

Messung von intervenierenden Variablen:

Als bedeutsame Einflußgrößen auf die Art und Weise der Aufgabenbearbeitung haben sich die EDV-Vorerfahrung der Benutzer und die Hilfestellungen durch den Beobachter herausgestellt (RAUTERBERG 1988). Beide Variablen sollten gemessen werden, um sie später bei der statistischen Auswertung berücksichtigen zu können. Die Vorerfahrung läßt sich mittels Fragebogen erfassen (YAUERBAUM & CULPAN 1990). Die Art und die Anzahl der gegebenen Hilfestellungen des Beobachters wird von diesem während des bBTs auf einem Protokollbogen vermerkt.

Messung der abhängigen Variablen:

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 (BOOTH 1990, S. 123). 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, Überlegungszeit¹, Anzahl Tastendrucke, Anzahl Fehler²; siehe MÜLLER-HOLZ et al. 1991).

Aufgabenbearbeitung durch den Benutzer:

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 Aufgabenbearbeitung zur Verfügung gestellt. Falls die Aufgabenbearbeitungszeit 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.

¹ siehe hierzu ACKERMANN & GREUTMANN (1987), bzw. GREUTMANN & ACKERMANN (1987).

² zum Thema "Fehler"-Analyse siehe ZAPF & FRESE (1989).

Verhalten des Beobachters:

Der Beobachter hält sich während der Aufgabenbearbeitung 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 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 (GOMOLL 1990, S.89).

4.1.1.5. Nachbereitung mit dem Benutzer

Jedem Benutzer muß die Gelegenheit gegeben werden, sofern dies nicht schon im Rahmen der Video-Konfrontation¹ 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 Problemsichten aus diesen Gesprächen. In dieser Nachbereitung lassen sich auch Fragebögen mit standardisierten oder offenen Fragen zur Beantwortung spezieller Benutzungsaspekte einsetzen.

4.1.1.6. Auswertung der Beobachtungen

Während der Durchführung eines bBTs wird man immer wieder überraschende und sehr informative Benutzungsweisen beobachten können (KARAT 1988, S. 895). 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.

Aufbereitung der Daten:

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 (ACKERMANN & GREUTMANN 1987, MÜLLER-HOLZ et al. 1991).

¹ siehe mehr zur Methode der Videokonfrontation bei MOLL 1987, S. 183f.

Qualitative Auswertungen:

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.

Quantitative Auswertungen:

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 inferenzstatistischen Zusammenhangsanalysen gehen (siehe mehr bei LANDAUER 1988). Dazu ist es notwendig, daß die verschiedenen Benutzungseigenschaften (Dauer der Aufgabenbearbeitung, mittlere Überlegungszeit, Fehlerart, Ausmaß an Beanspruchung, etc.) pro Benutzer in quantifizierter Form vorliegen.

4.2. Deduktive benutzungs-orientierte Benchmark-Tests (bBTs)

Deduktive bBTs dienen primär der Entscheidungsfindung zwischen verschiedenen System-Alternativen (z.B. WANDKE 1990), bzw. zur Kontrolle der erreichten Verbesserung (siehe zum Stichwort "Oberflächen-Effekt" bei RAUTERBERG 1991) und erst sekundär der Gewinnung von Gestaltungsvorschlägen (KARAT 1988). Es ergeben sich daher einige unterschiedliche Anforderungen an die Durchführung von deduktiven bBTs.

4.2.1. Vorgehensweise bei deduktiven bBTs

4.2.1.1. Voraussetzungen an die zu testende Software

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.

4.2.1.2. Gestaltung der Beobachtungssituation

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 (BORTZ 1984, S. 422ff).

Konstruktion der Benchmark-Aufgaben:

Im Fall-II müssen für jede System-Alternative *parallele* Benchmark-Aufgaben entwickelt werden (siehe als ein Beispiel RAUTERBERG 1991).

Auswahl der Benutzer:

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 (BORTZ 1984, S. 504ff; LANDAUER 1988, S. 918ff). Die einzelnen Gruppen müssen hinsichtlich verschiedener Parameter möglichst homogen sein: EDV-Vorerfahrung, Geschlecht, Alter, Beruf.

4.2.1.3. Auswertung der Beobachtungen

Mittels inferenzstatistischer Auswertungsverfahren können die gewonnenen Beobachtungsergebnisse auf ihre *Generalisierbarkeit* hin getestet werden (LANDAUER 1988; BORTZ 1989).

4.3. Aufwandabschätzung der Methode

Der Durchführungsaufwand von bBTs hängt im wesentlichen von der Anzahl der zu beteiligenden Benutzer ab. Je mehr Benutzer beteiligt sind, desto repräsentativer und umfassender sind die Ergebnisse. Dennoch ist es ratsam, den Aufwand auf ein notwendiges Minimum zu beschränken. Am einfachsten lassen sich induktive bBTs durchführen. Für die Planung, Durchführung und Auswertung sind in diesem Fall einige Tage bis Wochen zu veranschlagen. Dagegen ist bei deduktiven bBTs in der Regel mit einigen Wochen bis Monaten zu rechnen.

5. Das Fallbeispiel ADIMENS

Im Rahmen der Benutzerbeteiligung bei der Standardsoftware-Entwicklung des relationalen Datenbank-Programmes ADIMENS wurden Evaluationsstudien (induktive bBTs) und experimentelle Vergleichsstudien mit zwei verschiedenen Oberflächenvarianten (deduktive bBTs) durchgeführt (ADIMENS-ST, ADIMENS-ascii vs. ADIMENS-GT; siehe RAUTERBERG 1989, 1990b). Die konkreten Analyseergebnisse flossen in die Oberflächengestaltung von ADIMENS-GT+ (Version 3.0 und 3.1) und in die Windows-Portierung GX ein (MOLLENHAUER 1991; siehe auch Abb. 2). Die Veränderungen zwischen GT und GT+ wurden bereits experimentell auf ihre ergonomische Relevanz anhand von vier Benchmark-Aufgaben getestet (RAUTERBERG 1991).

In den empirischen Untersuchungen mit ADIMENS-GT wurde eine Reihe von verbesserungswürdigen Oberflächeneigenschaften gefunden (RAUTERBERG 1988): 1. die Aufteilung der Pull-down Menüs erwies sich als mögliche Quelle von Fehlbedienungen; 2. die Funktionalität der Desktop-Ikonen konnte von Anfängern nicht "entdeckt" werden, weil sie diese in den Pull-down Menü-Optionen vergeblich suchten; 3. die Wirkung einer Reihe von wichtigen Schaltern wurde oft deshalb falsch berücksichtigt, weil ihre Schalterstellung nicht permanent sichtbar waren; 4. die enorme Bedeutung der "Wahl" (permanente Selektion von Datensätzen mittels 'Filter') war durch die sehr häufige Verwendung der Ausgabeform "Anzeigen als Liste" begründet; hier soll die Möglichkeit zur temporären Selektion vorab Entlastung bringen; dies wird dem Benutzer durch die Implementation des Bearbeitungsmodus "Bearbeiten" ermöglicht; 5. die wichtigste Eigenschaft eines relationalen DBMS - die *Relationalität* - sollte dem Benutzer in einer direkt-manipulativ interaktiven Form durch Verbund-Dateien bei ADIMENS-GT+ zugänglich sein (RAUTERBERG 1991).

PRODUKT	JAHR	MASSNAHMEN	ERGEBNIS
ADIMENS-ascii	1987	Kriterien-geleitete Softwareentwicklung	Entwicklung der Desktop-Oberfläche
ADIMENS-GT	1988	induktiver Benchmark-Test "GT" (N=8)	Gestaltungsvorschläge
		deduktiver Benchmark-Test "GT - ascii" (N=24)	Entscheidung zugunsten der Desktop-Oberflächen
ADIMENS-GT+ Vers. 3.0	1989	Fragebogen (N=220)	Gestaltungsvorschläge
ADIMENS-GT+ Vers. 3.1	1990	deduktiver Benchmark-Test □ "GT - GT+" (N=30)	empirische Überprüfung der Gestaltungsvorschläge
ADIMENS-GX	1991	deduktiver Benchmark-Test □ "GT+ - GX" (geplant)	empirische Überprüfung der Gestaltungsvorschläge

Abbildung 2: Übersicht über die durchgeführten und geplanten Benchmark-Tests (fett umrandete Maßnahmen) im Rahmen der Standardsoftware-Entwicklung von ADIMENS in den Jahren 1988 bis 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.

7. Literatur

- ACKERMANN D. (1988): Empirie des Softwareentwurfes: Richtlinien und Methoden. In: Einführung in die Software-Ergonomie. Mensch-Computer Kommunikation - Grundwissen 1. (BALZERT H. et al.; Hrsg.), Berlin New York: Walter de Gruyter; S. 253-276.
- ACKERMANN D. & GREUTMANN T. (1987): Interaktionsgrammatik und kognitiver Aufwand. In: German Chapter of the ACM Bericht 29: Software-Ergonomie'87; (SCHÖNPFLUG W. & WITTSTOCK M.; Hrsg.); Stuttgart: Teubner; 262-270.
- APENBURG E. (1986): Befindlichkeitsbeschreibung als Methode der Beanspruchungsmessung. Zeitschrift für Arbeits- und Organisationspsychologie, 30 (N.F. 4); 3-14.
- BOEHM B.W. (1981): Software Engineering Economics. Englewood Cliffs, NJ: Prentice-Hall.
- BOOTH P. (1990): An Introduction to Human-Computer Interaction. Hove London Hillsdale: Lawrence Erlbaum.
- BORTZ J. (1984): Lehrbuch der empirischen Forschung für Sozialwissenschaftler. Berlin New York: Springer.
- BORTZ J. (1989): Lehrbuch der Statistik. Berlin Heidelberg New York: Springer.
- CAMPBELL D.T. & STANLEY J.C. (1970): Experimentelle und quasi-experimentelle Anordnungen in der Unterrichtsforschung. In: Handbuch der empirischen Unterrichtsforschung I. (INGENKAMP K.H.; Hrsg.), Weinheim: Beltz.
- DZIDA W. (1990): Ergonomische Normenkonformität. In: 5. Herbstschule 'Software-Ergonomie' in Zürich, 18.-21. Spet. 1990, (GI Deutsche Informatik-Akademie, Ahrstr. 45, D-5300 Bonn 2); S.1-15.
- GOMOLL K. (1990): Some Techniques for Observing Users. in: The Art of Human-Computer Interface Design. (LAUREL B.; ed.) Reading, Mass.: Addison-Wesley; pp.85-90.
- GREUTMANN T. & ACKERMANN D. (1987): Individual differences in human-computer interaction: how can we measure if the dialog grammar fits the user's needs? In: Human-Computer Interaction: INTERACT'87; (BULLINGER H.-J. & SHACKEL B.; eds.); Amsterdam: North-Holland; 145-149.
- HUNZIKER C. & HÄSSIG S. (1990): ADIMENS-Umfrage: Untersuchung von Datenbankstrukturen. unveröffentlichte Gruppensemesterarbeit WS'89/90. Institut für Arbeitspsychologie, Zürich: Eidgenössische Technische Hochschule.
- KARAT J.. (1988): Software Evaluation Methodologies. in: Handbook of Human-Computer Interaction. (HELANDER M.; ed.), Amsterdam: Elsevier Science; pp. 891-903.
- KLAS H., KUICH D. & LAES T. (1990): Auswertung der Fragebögen zur Anwendung und Beurteilung von ADIMENS ST/GT. unveröffentlichte Gruppensemesterarbeit WS'89/90. Institut für Arbeitspsychologie, Zürich: Eidgenössische Technische Hochschule.
- LANDAUER T.K. (1988): Research Methods in Human-Computer Interaction. in: Handbook of Human-Computer Interaction. (HELANDER M.; ed.), Amsterdam: Elsevier Science; pp. 905-928.
- MACAULAY L., FOWLER C., KIRBY M. & HUTT A. (1990): USTM: a new approach to requirements specification. Interacting with Computers. vol 2 no 1, pp. 92-118.
- MOLL T. (1987): Über Methoden zur Analyse und Evaluation interaktiver Computersysteme. In: Software-Ergonomie. State of the Art 5. (FÄHNRIK K.P.; Hrsg.), München Wien: Oldenbourg, S. 179-190.
- MOLLENHAUER R. (1991): Benutzerorientierte Software-Entwicklung des voll-grafischen Datenbanksystems Adimens: Projektreflexionen aus der Perspektive des beteiligten Softwarehauses. in: Software für die Arbeit von morgen. Bilanz und Perspektiven anwendungsorientierter Forschung. Ergänzung zum Tagungsband. Herausgegeben für DLR, Projektträger Arbeit und Technik. Krefeld: Vennekel & Partner.
- MÜLLER-HOLZ auf der Heide B., ASCHERSLEBEN G., HACKER S. & BARTSCH T. (1991): Methoden zur empirischen Bewertung der Benutzerfreundlichkeit von Bürosoftware im Rahmen von Prototyping. In: Software für die Arbeit von morgen. Bilanz und Perspektiven anwendungsorientierter Forschung. (FRESE M.; KARSTEN C.; SKARPELIS C. & ZANG-SCHEUCHER B.; Hrsg.), Berlin Heidelberg New York: Springer.
- RAUTERBERG M. (1988): Untersuchung der Benutzerfreundlichkeit einer desktop-orientierten Benutzungsoberfläche am Beispiel eines relationalen Datenbanksystems. unveröffentlichter Forschungsbericht, Institut für Arbeitspsychologie, Zürich: Eidgenössische Technische Hochschule.

- RAUTERBERG M. (1989): MAUS versus FUNKTIONSTASTE: ein empirischer Vergleich einer desktop- mit einer ascii-orientierten Benutzungsoberfläche. In: German Chapter of the ACM Bericht 32: Software-Ergonomie'89. (MAASS S. & OBERQUELLE H.; Hrsg.). Stuttgart: Teubner; 313-323.
- RAUTERBERG M. (1990a): Induktive versus deduktive Untersuchungsmethoden: Vor- und Nachteile für die Gestaltung von Benutzungsoberflächen in der Mensch-Computer-Interaktion. in: 32. Tagung experimentell arbeitender Psychologen Regensburg, April 9-12, 1990. (IRTEL H. & DRÖSLER J.; Hrsg.), Fachbereich Psychologie, Regensburg: Universität Regensburg.
- RAUTERBERG M. (1990b): Experimentelle Untersuchungen zur Gestaltung der Benutzungsoberfläche eines relationalen Datenbanksystems. In: Projektberichte zum Forschungsprojekt Benutzer-orientierte Softwareentwicklung und Schnittstellengestaltung (BOSS), Nr. 3. (SPINAS P.; RAUTERBERG M.; STROHM O.; WAEBER D. & ULICH E. ; Hrsg.), ETH-Zürich: Institut für Arbeitspsychologie.
- RAUTERBERG M. (1991): Benutzerorientierte Benchmarktests: eine Methode zur Benutzerbeteiligung bei der Entwicklung von Standardsoftware. In: Projektberichte zum Forschungsprojekt Benutzer-orientierte Softwareentwicklung und Schnittstellengestaltung (BOSS), Nr. 6. (SPINAS P.; RAUTERBERG M.; STROHM O.; WAEBER D. & ULICH E. ; Hrsg.), ETH-Zürich: Institut für Arbeitspsychologie.
- SPINAS P. (1990): Benutzerfreundlichkeit von Dialogsystemen und Benutzerbeteiligung bei der Software-Entwicklung. In: Das Bild der Arbeit. (FREI F. & UDRIS I.; Hrsg.), Bern: Hans Huber; S. 158-171.
- SPINAS P., WAEBER D. & STROHM O. (1990): Kriterien benutzerorientierter Dialoggestaltung und partizipative Softwareentwicklung: eine Literaturlaufarbeitung. In: Projektberichte zum Forschungsprojekt Benutzer-orientierte Softwareentwicklung und Schnittstellengestaltung (BOSS), Nr. 1. (SPINAS P.; RAUTERBERG M.; STROHM O.; WAEBER D. & ULICH E. ;Hrsg.), ETH-Zürich: Institut für Arbeitspsychologie.
- STROHM O. (1990): Arbeitsorganisation, Methodik und Benutzerorientierung bei der Softwareentwicklung. In: Projektberichte zum Forschungsprojekt Benutzer-orientierte Softwareentwicklung und Schnittstellengestaltung (BOSS), Nr. 2. (SPINAS P.; RAUTERBERG M.; STROHM O.; WAEBER D. & ULICH E. ; Hrsg.), ETH-Zürich: Institut für Arbeitspsychologie.
- STUTZ R., STAUBL, A., ROHRBACH C. & KRIESL K. (1990): Auswertung der Umfrage über die ADIMENS-Datenbank. unveröffentlichte Gruppensemesterarbeit SS'90. Institut für Arbeitspsychologie, Zürich: Eidgenössische Technische Hochschule.
- ULICH E. (1985): Einige Anmerkungen zur Software-Psychologie. sysdata. Nr. 10, S. 53-58.
- ULICH E. (1986): Aspekte der Benutzerfreundlichkeit. In: Arbeitsplätze von morgen. Berichte des German Chapter of the ACM, Band 27. (REMMELE W. & SOMMER M.; Hrsg.), Stuttgart: Teubner, S. 102-121.
- ULICH E. (1991): Arbeitspsychologie. Stuttgart: Poeschel-Verlag.
- VOSSEN P. (1991): EVA II - ein Werkzeug zur interaktiven Protokollerstellung und -analyse. In: Software-Ergonomie'91 - Benutzerorientierte Software-Entwicklung. Poster-Band. (RAUTERBERG M. & ULICH E.; Hrsg.), Institut für Arbeitspsychologie, Zürich: Eidgenössische Technische Hochschule.
- WAEBER D. (1990): Entwicklung und Umsetzung von Modellen partizipativer Softwareentwicklung. In: Projektberichte zum Forschungsprojekt Benutzer-orientierte Softwareentwicklung und Schnittstellengestaltung (BOSS), Nr. 4. (SPINAS P.; RAUTERBERG M.; STROHM O.; WAEBER D. & ULICH E. ; Hrsg.), ETH-Zürich: Institut für Arbeitspsychologie.
- WANDKE H. (1990): Zielkonflikte bei der psychologischen Gestaltung von Mensch-Rechner-Schnittstellen. in: Proceedings of 6th International Symposium on Work Psychology Dresden, March 27-29, 1990. Sektion Arbeitswissenschaften Wissenschaftsbereich Psychologie, Dresden: Technische Universität; S. 132-138.
- WEISER M. & SHNEIDERMAN B. (1987): Human Factors of Computer Programming. in: Handbook of Human-Computer Interaction. (HELANDER, M.; ed.), Amsterdam: Elsevier Science; pp. 1398-1415.
- WHITESIDE J., BENNETT J. & HOLTZBLATT K. (1988): Usability Engineering: Our Experience and Evolution. in: Handbook of Human-Computer Interaction. (HELANDER M.; ed.), Amsterdam: Elsevier Science; pp. 791-817.
- WILLIGES R.C., WILLIGES B. & ELKERTON J. (1987): Software interface design. in: Handbook of Human Factors. (SALVENDY G.; ed.), New York: Wiley & Sons.
- YAUVERBAUM G.J. & CULPAN O. (1990): Exploring the Dynamics of the End-User Environment: The Impact of Education and Task Differences on Change. Human Relations. vol 43 no 5, pp. 439-454.
- ZAPF D. & FRESE M. (1989): Benutzerfehler im Kontext von Arbeitsaufgabe und Arbeitsorganisation. In: Software-Ergonomie'89. Aufgabenorientierte Systemgestaltung. (MAASS S. & OBERQUELLE H.; Hrsg.), Stuttgart: Teubner, S. 213-222.

Matthias Rauterberg
Institut für Arbeitspsychologie
Eidgenössische Technische Hochschule (ETH)
CH-8092 ZÜRICH