9 Eine Fallstudie zu Benutzerbeteiligung und Prototyping

Michael Frese, Jochen Prümper & Ferdinand Solzbacher

9.1 Zusammenfassung

In diesem Beitrag werden Erfahrungen aus einem SE-Projekt berichtet, in dem Benutzerbeteiligung und Prototyping verwirklicht wurden. Diese Kombination behinderte den SE-Prozeß nachhaltig. Auf Basis einer retrospektiven Analyse der Vorkommnisse und ihrer besonderen Dynamik werden alternative Vorgehensweisen vorgeschlagen: (1) Ein Versioning- Ansatz mit Ausarbeitung der Funktionalität und (2) eine Trennung zwischen einem Software-Entwickler-Team auf der einen Seite und einem Spezialisten-Team für das Fachgebiet und Human Factors auf der anderen Seite.

9.2 Einleitung

In diesem Beitrag wird ein Fallbeispiel geschildert, an dem sich charakteristische Probleme der Kombination von Prototyping und Benutzerbeteiligung beschreiben lassen.

Kaum ein Aspekt der Software-Entwicklung erfährt in letzter Zeit so hohe Akzeptanz wie die Benutzerbeteiligung (vgl. Mambrey, Oppermann, & Tepper, 1986; Oppermann, 1983; Ulich, 1991, 1993). Man erhofft sich von der Benutzerbeteiligung, daß Software-Entwickler sich mehr mit der Sicht der Benutzer auseinandersetzen, die Programme sich mehr an den Bedürfnissen der Benutzer orientieren und dadurch die Benutzer auch eine höhere Akzeptanz gegenüber dem Programm entwickeln.

In dem hier zu berichtenden Fallbeispiel wurde der Versuch unternommen, Benutzerbeteiligung mit Prototyping zu koppeln, wobei zunächst das Konzept einer möglichst schnellen Abfolge von Prototypen im Vordergrund stand (Floyd, 1984)[21]. Eine Reihe von Autoren haben argumentiert, daß solche Projekte effizienter arbeiten als herkömmliche Projekte (z.B. Mantei & Teorey, 1988). Allerdings gibt es für diese Behauptung kaum empirische Beweise. Ein wichtiger Beleg ist die Untersuchung von Strohm (1991). Hier

[21] Es gibt natürlich eine Reihe von Unterschieden zwischen den einzelnen Formen des Prototyping. Darüber hinaus ist Prototyping von Benutzerbeteiligung zu unterscheiden. Beispielsweise wird Benutzerbeteiligung in den von IPAS untersuchten Projekten (siehe Kapitel 7 und 8) nicht mit Prototyping verbunden. Allerdings findet Prototyping häufig unter Einbezug von Benutzervertretern statt.

ERSCHIENEN IN:

F.C. Brodbeck & M. Frese (Hrsg.) (1994). *Produktivität und Qualität in Software-Projekten. Psychologische Analyse und Optimierung von Arbeitsprozessen in der Software-Entwicklung* (S. 135-143). München: Oldenbourg.

wird im wesentlichen berichtet, daß Benutzerbeteiligung bzw. Prototyping die Prozeß- und Produktqualität von SE-Projekten positiv beeinflußt. Seine Daten beruhen jedoch ausschließlich auf Auskünften von Managern. Diese Mitarbeitergruppe hat allerdings ein vitales Interesse daran, vor allem die positiven Effekte eines von ihnen zu verantwortenden, neuen Verfahrens der Software-Entwicklung zu betonen, sonst würden sie innerhalb ihrer Organisation nicht als erfolgreich angesehen. Darüber hinaus verwenden Manager von SE-Projekten den Begriff Prototyping in einem ausgesprochen breiten Sinn und damit häufig unklar und zum Teil sogar falsch (Beck, 1993). Im Kapitel 7 werden Befunde berichtet, die denen von Strohm (1991), zumindest was Benutzerbeteiligung angeht, widersprechen, die sich allerdings auf den Konsens der Auskünfte von verschiedenen Mitarbeitergruppen stützen. Von dieser Untersuchung werden fast ausschließlich negative Auswirkungen von Benutzerbeteiligung auf den Prozeß der Software-Entwicklung zu Tage gefördert.

9.3 Das Projekt: Verlag 2000

Der Fallbericht behandelt das Projekt Verlag 2000[22], in dem von einem mittelständischen Betrieb mit Sitz in Ost-Berlin eine integrierte Software für klein- und mittelständische Betriebe aus dem Druck- und Verlagsbereich entwickelt wird. Im Rahmen einer Anwendung von Rechnern mittlerer Größenordnung (IBM Datenbankcomputer AS/400) wird eine integrierte Software-Lösung entwickelt, die solche Bereiche miteinander verbinden soll, die in der mittelständischen Verlagsbranche oftmals nur nebeneinander stehen (Adressenverwaltung, Anzeigen, Abo- und Buch-Bereich, Buchhaltung und Kostenrechnung). Weitere Ziele sind, eine Software zu erstellen, die den neuesten Software-ergonomischen Erkenntnissen entspricht, die auf Basis moderner Software-Entwicklungsmethoden entwickelt wird und die darüber hinaus auch Chancen anbietet, die Arbeitsbedingungen der Betroffenen nach arbeitswissenschaftlichen Kriterien zu verbessern[23].

Eine wesentliche Aufgabe des Projekts besteht darin, im Rahmen der Software-Entwicklung eines mittelständischen Unternehmens zu überprüfen, ob Prototyping, Benutzerpartizipation und intensive arbeitswissenschaftliche Ausrichtung zu einer Verbesserung des Produkts, einer besseren Einführung der Software in den Betrieb sowie zu einer Verbesserung der

[22] Das Projekt der Firma Data Train GmbH „Verlag 2000: Eine benutzerfreundliche integrierte Lösung für die mittelständische Verlags- und Druckereibranche unter Berücksichtigung von zu verbessernden Arbeitsbedingungen für die Beschäftigten" wird vom Bundesminister für Forschung und Technologie, Förderschwerpunkt „Arbeit und Technik" (Förderkennzeichen 01 HK 601 8) gefördert.

[23] Ob die neue Software-Lösung insgesamt zu einer besseren Benutzbarkeit und zu verbesserten Arbeitsbedingungen führt, wird in dieser Fallstudie bisher noch nicht thematisiert, weil die neue Software noch nicht fertiggestellt und noch nicht eingeführt wurde. Eine solche Analyse wird z.Z. geplant.

Arbeitsbedingungen führen. Eine weitere Frage ist, ob sich ein solches Vorgehen auch in anderen Projekten dieser Art lohnen würde. Um dies zu organisieren und zu dokumentieren sind an diesem Projekt auch Arbeitswissenschaftler beteiligt (als Mitarbeiter und als Berater).

Die Rahmenbedingungen waren folgende: Die Software-Entwickler wurden intensiv geschult und immer wieder in arbeitswissenschaftlichen Fragen beraten (vgl. Prümper, 1993). Um die Benutzerpartizipation sicherzustellen, wurden zwei Gremien eingerichtet: Eine Benutzergruppe und eine Kundengruppe. In der Benutzergruppe waren Angestellte, die die alte Software in ihrer täglichen Arbeit verwendeten und in der Kundengruppe Manager und EDV-Leiter, die die Software in den Verlagen betreuten. Mit der Zweiteilung der Gruppen sollte sichergestellt werden, daß auch wirklich die End-Benutzer in den Software-Entwicklungs-Prozeß einbezogen wurden. Gleichzeitig war es sinnvoll und auch notwendig, die Manager in die Partizipation einzubinden. Alle Teilnehmer dieser Gruppen kamen aus kleineren und mittelgroßen Betrieben der Verlagsbranche. Beide Gruppen trafen sich für jeweils 1 – 2 Tage und tagten etwa alle 6 Wochen.

Bestreben des Projekts war, die Gruppen dazu zu motivieren, sich aktiv in den Entwicklungsprozeß einzuschalten. Dies ist gut gelungen. Im Gegensatz zu manchen Berichten über Benutzerpartizipation waren sowohl Benutzer- als auch Kundengruppe ausgesprochen aktiv. Sie haben eine Reihe von Vorschlägen eingebracht und Ausarbeitungen für einzelne Sitzungen übernommen. Die Benutzer waren stark motiviert, aktiv an der Erstellung der Software mitzuwirken. Dies mag an der Art der Einführung der Arbeit gelegen haben. Sie bestand darin, aus den Fehlern des alten Systems für das neue System zu lernen. Um die Fehler in Erinnerung zu rufen, wurden auf Grundlage von Arbeitsanalysen konkrete Aufgaben konstruiert und während der Sitzung bearbeitet. Die bei der Bearbeitung auftretenden Probleme wurden gesammelt und geordnet (Prümper, 1993). Darüber hinaus wurden empirische Untersuchungsergebnisse präsentiert und als Anregung für die Diskussion verwendet (Prümper & Anft, 1993).

Der Ansatz, konkrete Arbeitsaufgaben zu lösen und die dabei auftretenden Schwierigkeiten zu diskutieren, um daraus Vorschläge zu entwickeln, wurde immer wieder verwendet. Das heißt, die ersten implementierten Software-Teile wurden präsentiert und mit Hilfe solcher Aufgaben beurteilt. Dabei wurde von Seiten des verantwortlichen Arbeitspsychologen kein Versuch unternommen, Schwierigkeiten in der Software-Lösung zu glätten oder zu übergehen, so daß auch die Probleme und Fehler der neuen Software schnell deutlich wurden.

Eine konkrete Beschreibung einer Sitzung mag das Vorgehen verdeutlichen. Es handelt sich um die 3. Benutzergruppensitzung zum Problembereich „Adresse". In dieser Sitzung bestand das Ziel darin, anhand der Bearbeitung von Standardaufgaben durch die Benutzer Schwachstellen bezüglich des Adreßteils der alten Lösung aufzudecken, Verbesserungsmöglichkeiten zu diskutieren sowie einen gemeinsamen Standard für die Anforderungen der

einzelnen Verlage zu finden. In drei Kleingruppen zu je drei bis vier Benutzern wurden vorbereitete Aufgaben bearbeitet. Nach Durchführung der 13 Aufgaben wurden die Erfahrungen jeder Gruppe dem Plenum präsentiert und gemeinsame Problembereiche erarbeitet. Das Ergebnis sollte in Verbesserungsvorschlägen für die zukünftige Lösung münden. Deshalb wurden die bei der Bearbeitung aufgetretenen Probleme zusammengefaßt, in einer Übersichtstabelle schriftlich niedergelegt und am zweiten Tag der Benutzergruppensitzung kritisch diskutiert. Offene Fragen waren z.B.:

1. Soll Adreß-Art, Branche, Gebietsnummer in den Standard?
2. Aufgliederung Firma – Privatperson?
3. Anzahl der Adreßzeilen – 3 oder 4?
4. Gestaltung von Telefon und Telefaxfeldern?
5. Etikettendarstellung, was kommt rein?
6. Soll Fachgruppe/Berufsbezeichnung in den Standard?

Zum Abschluß des Tages wurde die Gestaltung der Eingabemaske „Adresse" besprochen. Dazu wurde direkt am Rechner in Zusammenarbeit mit den Benutzern, den Entwicklern und den Arbeitswissenschaftlern die Anordnung und Länge der Standardeingabefelder erarbeitet. Man gelangte so zu einem ersten, allgemein akzeptierten Maskenentwurf. Bis zur nächsten Benutzergruppensitzung sollte dieser von den Entwicklern vervollständigt werden.

9.4 Probleme

Aus dieser Vorgehensweise ergab sich eine interessante soziale Projektdynamik mit zwei Phasen: Die erste Phase war durch hohen Enthusiasmus charakterisiert, die zweite durch zunehmende Frustrationen.

In der ersten Phase ist es ausgesprochen gut gelungen, die Benutzer und Kunden zu einer aktiven und produktiven Mitarbeit zu bewegen. Dies geschah durch die oben dargestellte Moderationstechnik mit Hilfe der Fehleranalysen.

In der zweiten Phase führte allerdings gerade die aktive Mitarbeit zu einer letztlich unproduktiven Dynamik: Die Teilnehmer beider Gruppen wurden zunehmend ungeduldiger mit der Geschwindigkeit der Software-Entwicklung, besonders mit der Entwicklung der Funktionalität. Das wirkte sich in den Sitzungen besonders dann drastisch aus, wenn aufgrund des oben beschriebenen aufgabenorientierten Ansatzes festgestellt wurde, daß Teile des Programms nicht in der gewünschten Art und Weise funktionierten.

Diese Ungeduld und Frustationen sind leicht verständlich, wird doch in der Arbeitspsychologie immer wieder das Primat der Aufgabe für die Arbeitenden betont (z.B. Frese, 1987; Hacker, 1986). Für die Arbeitenden steht die Aufgabe im Vordergrund ihrer Bemühungen und auch die Selbstdefinition bezieht sich auf die Aufgaben. Aus diesem Grund wird bei einem Werkzeug

– wie z.B. einem Computerprogramm – immer zuerst die Frage gestellt, ob dieses die Erledigung der Aufgaben unterstützt oder behindert. Alle anderen Fragen sind demgegenüber zweitrangig, wie zum Beispiel die Benutzergruppensitzung zeigt, in der das Modul „Abonnement" diskutiert wurde. Ziel dieser Veranstaltung war es, basierend auf den Erkenntnissen aus der Bearbeitung der Standardaufgaben die Anforderungen an das neue Modul zu ermitteln. Auf der Tagesordnung standen damit die Bearbeitung von Standardaufgaben zur Fehleranalyse und eine erste Definition des Standards für die Teilanwendung „Abonnement". Der zweite Tag war von wesentlicher Kritik an der Vorgehensweise geprägt. So wurde z.B. die nicht lauffähige Demo-Version kritisiert – die verwendete Verlagslösung paßte nicht mit einigen hinterlegten Dateien zusammen. Ebenso wurde die fehlende Dokumentation der Funktionalität bemängelt. Notwendig ist es z.B., einen Überblick darüber zu haben, was mit Eingaben „hinter dem Feld, hinter der Maske" passiert (Zitat: „Wohin gehen die 30 DM Gutschrift?"). Des weiteren kritisierten die Benutzer, daß das Thema „Privatadresse" zugunsten der „Geschäftsadresse" bislang vernachlässigt wurde.

Diese Ungeduld und Frustration der Benutzergruppe verursachte auch auf seiten der Software-Entwickler eine bestimmte Dynamik, die mit drei Konsequenzen beschrieben werden kann: Erstens wurde versucht, die Prototypen immer elaborierter zu gestalten und mit mehr Funktionalität zu verbinden. Damit wurden sie natürlich aufwendiger und der Prototyping-Ansatz wurde mehr und mehr zum Versioning-Ansatz (vgl. Rauterberg, 1991)[24]. Zweitens wurden die Software-Entwickler immer weniger mutig und bevorzugten konservative Lösungen, weil sie zunehmend unter Druck standen, Experimente zu unterlassen, um sofort funktionierende und damit „vorzeigbare" Software zu produzieren. Drittens wurde es immer wichtiger, daß die jeweilige Sitzung auch ein Erfolg wurde. Damit stand nicht mehr das Interesse, aus Fehlern zu lernen, im Vordergrund, sondern es sollte vielmehr dokumentiert werden, daß der Fortschritt der Software schnell genug voranging.

Allerdings war der Prozeß der ständig steigenden Erwartungen in den Benutzer- und Kundengruppen schneller als der eben dargelegte Lernprozeß der Software-Entwickler. Als die Software-Entwickler unmittelbar vor einer Benutzergruppensitzung noch eine weitere Funktion programmierten, um die Lösung noch etwas zu verbessern, ruinierten sie dabei die bereits geprüften Abläufe. Daraufhin fand die Benutzergruppe anstelle der erwarteten

[24] Es gibt natürlich unterschiedliche Konzeptualisierungen darüber, was Prototyping bedeutet (vgl. Züllighoven, Altmann und Doberkat, 1993). Ein Prototyp kann von einem „mock-up" bis zu einer fertigen Version reichen. Dennoch ist den meisten Definitionen von Prototypen eigen, daß die Funktionalität nicht völlig ausgearbeitet ist. Einer Version hingegen liegt eine weitgehend ausgearbeitete Funktionalität zugrunde. Dennoch ist der Übergang von einem Prototypen zu einer Version sicher fließend. Unsere Bemerkungen sind deshalb nicht als allgemeine Kritik an dem Prototypen Ansatz zu verstehen, sondern als notwendige Ergänzung der bisherigen Diskussion über die Dynamik der Verwendung des Prototypen-Ansatzes bei gleichzeitiger Benutzerpartizipation – ein Ansatz, der nach Floyd (1993) besonders in Europa anzutreffen ist.

ausgetesteten Lösung eine mit funktionalen Fehlern vor. An diesem Punkt kulminierten die beschriebenen Prozesse und es kam zum Eklat: Die Teilnehmer der Benutzergruppe waren so frustriert, daß sie sich entschlossen, nicht mehr weiterzuarbeiten. Erst nach entsprechender Überzeugungsarbeit durch den Projektleiter wurde die Bereitschaft wieder hergestellt, weiter an der Software-Entwicklung teilzunehmen. Allerdings wurde der bis dahin bestehende (wenn auch schon zunehmend aufgeweichte) Ansatz des Prototypings zugunsten eines Versioning-Ansatzes aufgegeben.

Wie jede überwundene Krise setzte natürlich auch diese neue Energien frei. Da jede Software-Entwicklung immer krisenhaft verläuft, sind kritische Situationen dieser Art wahrscheinlich ein notwendiger Bestandteil jedes kreativen Prozesses. Dennoch erscheint uns die beschriebene Problematik, die sich aus der Dynamik der beteiligten Gruppen ergibt, symptomatisch für den Verlauf von Benutzerbeteiligung zu sein. Wir vermuten, daß sich aus solchen Situationen eine negative Einstellung der Software-Entwickler gegenüber Benutzerbeteiligung ergibt.

9.5 Schlußfolgerung

Wie bei jeder Fallstudie ist es notwendig zu differenzieren, welche Lehre man daraus verallgemeinern darf und welche nicht. Es gab mit Sicherheit Besonderheiten in diesem Projekt – die meisten Mitarbeiter hatten bisher noch nicht im Verlagsbereich gearbeitet (es handelte sich um eine neue Firma in Ost-Berlin) und es wurden sicherlich auch Managementfehler gemacht. All dies würde für eine spezifische Interpretation sprechen. Trotzdem meinen wir, daß sich einige Ergebnisse verallgemeinern lassen. Der Prototyping-Ansatz mit hoher Benutzerbeteiligung kann zur Frustration sowohl bei den beteiligten Benutzern als auch bei den Software-Entwicklern führen. Es entsteht fast zwangsläufig eine Dynamik, in der die Benutzergruppen immer höhere Anforderungen aufstellen und frustriert sind, wenn diese nicht schnell genug in die Entwicklung aufgenommen werden. Daraus entsteht dann ein Konservatismus auf seiten der Software-Entwickler, der nicht unbedingt zu einer fruchtbaren Arbeit beiträgt.

In der Literatur gibt es durchaus Beispiele für eine erfolgreiche Verwendung des Prototypen Ansatzes mit Benutzerbeteiligung. Es handelte sich allerdings oft um relativ einfache Systeme (wie z.B. bei Boehm, Gray & Seewaldt, 1984), die auch nicht der Normalsituation eines kommerziell arbeitenden Unternehmens unterworfen waren (wie z.B. bei Ortlieb & Holz auf der Heide, 1993).

Die Ergebnisse dieser Fallstudie sollten allerdings nicht mißverstanden werden. Sie sprechen weder gegen eine Benutzerbeteiligung *per se,* noch gegen ein iteratives Design. Denn die am Anfang benannten Vorteile der Benutzerbeteiligung haben sich auch in dieser Fallstudie deutlich bestätigt. Benutzerbeteiligung führt zu einer höheren Benutzerorientierung; die Lösun-

gen werden besser, sie nehmen auf die konkreten Arbeitsbedingungen der Benutzer Rücksicht. Alle diese Effekte lassen sich auch in dieser Fallstudie gut belegen.

Dennoch spricht die Fallstudie gegen einen einfachen Prototyping-Gedanken. Gerade wenn sich die Benutzer besonders aktiv in den Diskussionprozeß einbringen, wird die Debatte über die Prototypen mit eingeschränkter Funktionalität sehr schnell uninteressant und jeder Fehler erscheint dann als Problem der Software-Entwickler und verringert somit das Vertrauen in deren Kompetenz. Hier ist von besonderer Bedeutung, daß die Benutzer weniger an der reinen Oberflächengestaltung interessiert waren, sondern vor allem an der funktionalen Abbildung ihrer Arbeitsabläufe. Das heißt, eine Software-Ergonomie, die sich fälschlicherweise als reine Oberflächenergonomie versteht, geht an den wirklichen Bedürfnissen der Benutzer vorbei. Unsere Erfahrungen sprechen für zwei Vorschläge, die sich miteinander verknüpfen lassen.

1) Statt eines Prototypings scheint uns eine versionsorientierte Vorgehensweise angemessener.
2) Wir schlagen eine Trennung vor zwischen einem Software-Entwickler-Team auf der einen Seite und einem Spezialisten-Team für das Fachgebiet und für Human Factors auf der anderen Seite.

Zunächst zum Unterschied von Prototyping und Versioning: Man kann die beiden Ansätze akzentuierend unterscheiden. Prototyping ist stärker auf die Oberfläche und schwächer auf die Funktionalität hin orientiert, während dies beim Versioning-Ansatz umgekehrt ist (Floyd, Mehl, Reisin & Wolf, 1990). Beide Ansätze betonen die Abkehr vom reinen Wasserfallmodell und die Hinwendung zu einem iterativen und evolutionären Design (das Wasserfallmodell eignet sich sowieso kaum zur Beschreibung der realen Designprozesse, vgl. Weltz & Ortmann, 1992).

Der wesentliche Unterschied für unsere Zwecke besteht darin, daß im Versioning-Ansatz von vornherein die Entwicklung der Funktionalität im Vordergrund steht. Erst wenn diese gesichert ist, wird die Version den Benutzern vorgestellt. Das ist natürlich eine Abkehr von einem *„rapid-prototyping"*-Ansatz. Die Benutzerbeteiligung wird beim Versioning-Ansatz zweigeteilt. Zum einen werden die Benutzer bei der Spezifikation einbezogen. In dieser Phase sprechen sie aber nicht direkt mit den Software-Entwicklern, sondern mit dem Spezialisten für das Fachgebiet (in unserem Fall also dem Spezialisten für das Verlagswesen).

Anschließend werden die Benutzer zunächst für eine längere Periode nicht mehr aktiv involviert. Erst wenn die Funktionalität bis zu einem gewissen Grade entwickelt wurde, wird eine erste Version in der Benutzergruppe vorgestellt (moderiert durch den Human Factors Spezialisten). Hier werden zwei Punkte schwerpunktmäßig diskutiert: Zum einen, ob die in der Software realisierte Funktionalität auch für die Arbeitssituation brauchbar ist

und zum zweiten, wie sich die Oberfläche verbessern läßt. Dieser Vorgang wird iterativ zweimal wiederholt. Nur bei Spezialfragen, z.B. ob eine bestimmte Menüstruktur besser ist als eine andere, wird eine „rapid prototyping"-Methode verwendet.

Zur Funktion eines Human Factors Spezialisten als Mediator zwischen Software-Entwicklern und Benutzern: Der Erstautor hat früher zu verschiedenen Zeitpunkten vorgeschlagen, alle in einem Team involvierten Entwickler in den Prozeß der Benutzerbeteiligung aktiv einzuschalten (z.B. Frese & Brodbeck, 1989). Nach den Erfahrungen mit dieser Fallstudie muß hinterfragt werden, in welchem Maße Entwickler in den Benutzerbeteiligungsprozeß integriert werden sollten. Eine mögliche Alternative lehnt sich an einen Vorschlag von Norman (1986) an, nämlich daß es in jedem Software-Team einen Human Factors Spezialisten geben sollte. Nach diesem Modell gibt es keinen direkten Bezug zwischen Software-Entwicklern und Benutzern – dieser wird vermittelt durch die Human Factors Spezialisten. Solche Spezialisten müssen über ein umfangreiches Software-ergonomisches Wissen verfügen, gute psychologische Kenntnisse von Moderations- und Interviewtechniken besitzen (vgl. Bieger, Feltes, Schmalzriedt & Sieber, 1981; Klebert, Schrader & Straub, 1985), in der Anwendung ergonomischer Normen zur Dialoggestaltung erfahren sein, Kenntnisse über Evaluationsverfahren (z.B. über Fehleranalysen des menschlichen Handelns in der Mensch-Computer Interaktion; vgl. Frese & Zapf, 1991) und Arbeitsanalyseinstrumente aufweisen. Zudem sollten sie entsprechende Benutzungsoberflächen selbständig entwickeln können. Solche Human Factors Spezialisten können entweder Psychologen mit einer Informatikausbildung oder Informatiker mit einer Ausbildung in Psychologie sein.

Die Aufgabe der Software-Entwickler ist es, die erforderliche Funktionalität und die Schnittstellen zur Oberfläche zu entwickeln. Die Aufgaben des Human Factors Spezialisten bestehen in der Gestaltung der Oberfläche und in der Angabe, welche Schnittstellen für welche Funktionalitäten benötigt werden. Damit würden die Probleme reduziert, die sich aus der direkten Kommunikation zwischen Software-Entwicklern und Benutzern ergeben. Benutzer können im SE-Prozeß durchaus Störungen produzieren (wie Kap. 7 nahelegt) und Frustrationen können sich aufgrund des engen Kontakts zwischen Software-Entwicklern und Benutzern in einem negativen Prozeß hochschaukeln (wie diese Fallstudie zeigt). Der Mediator „Human Factors Specialist" verhindert derartige Prozesse. Seine Aufgabe wäre es, die Oberfläche zu erstellen, die Kritik an der Funktionalität zu vermitteln und auf die Schnittstellengestaltung zwischen Funktionalität und Oberfläche zu achten. Er müßte besonders umfassend qualifiziert werden und hätte innerhalb der Software-Entwicklungsgruppe eine herausragende Position. Er hätte also eine „boundary spanner" Funktion (siehe auch Kapitel 4).

Ein Problem besteht zweifellos darin, daß die Software-Entwickler nicht unbedingt bereit sind, Kritik an ihrer Software zu akzeptieren, wenn sie die Rückmeldungen über ihre Arbeit nicht direkt von den Benutzern erhalten.

Aus diesem Grund empfiehlt es sich im Rahmen des hier skizzierten Vorgehens, die Software-Entwickler mit Videoaufnahmen oder direkten Beobachtungen der Benutzer in der Arbeit mit der neuen Version zu konfrontieren.

Dieser Vorschlag kann natürlich gut mit dem oben ausgeführten Versioning-Ansatz verknüpft werden. Aus diesen beiden Vorschlägen dürfte sich eine brauchbare Entwicklung mit Benutzerbeteiligung unter Vermeidung von hohen Frustrationen ergeben.

Wir hoffen, daß mit diesen Vorschlägen bestimmte Probleme der Benutzerbeteiligung und eines iterativen Designs überwunden werden können. Wir vermuten, daß andere Projekte ähnliche Erfahrungen wie die hier berichteten gemacht haben, und meinen, daß es sich lohnt, nicht nur ein allgemeines Bekenntnis zur Benutzerbeteiligung abzugeben, sondern die Prozesse bei der Software-Entwicklung, so wie sie in der Praxis auch ablaufen, genau zu betrachten. Nur so, durch Kritik und fruchtbare Verarbeitung der Fehler in einem iterativen SE-Prozeß, können wir langfristig auch zu einer adäquaten Methode der Benutzerbeteiligung und der iterativen Software-Entwicklung beitragen.