Anforderungsanalyse zur Einführung eines Unterstützungssystems bei Software-Entwicklern

Hartmut Wandke, Andreas Dubrowsky und Jens Hüttner

Institut für Psychologie, Humboldt-Universität zu Berlin

Zusammenfassung

Durch die Implementierung eines elektronischen Informations- und Beratungssystems (Hyperbase) sollen Software-Entwickler im Hinblick auf Kooperation und Kommunikation innerhalb der Abteilung unterstützt werden. Vordergründig zur Unterstützung in bezug auf software-ergonomisches Wissen, aber auch anderen damit verbundenen Aktivitäten zum Wissensmanagement, wird das Vorgehen zur Analyse des Ist-Zustandes dargestellt. Diese Anforderungsanalyse diente zur Identifikation des vorhandenen Wissens, seiner Repräsentation sowie der individuellen Verteilung des „Organisationswissens" und sollte darüber hinaus klären, auf welche Art und Weise die Entwickler dieses Wissen erwerben (z. B. mit welchen Hilfsmitteln). Für weitere Entscheidungen wurden diese Ergebnisse ergänzt durch eine prospektive Analyse von Wünschen und Erwartungen der potentiellen Nutzer an ein solches Unterstützungssystem (Inhalte, Darbietung, Motivation zur Nutzung). Die mit Hilfe von Fragebögen, Testaufgaben, Interviews und Gruppendiskussionen erbrachten Ergebnisse zeigen, daß CSCW-Systeme ein geeignetes Medium zur Unterstützung von wissensbasierten Kooperationsprozessen sein können, ihre Effizienz allerdings davon abhängt, inwieweit die Erwartungen und Bedürfnisse der Konsumenten befriedigt werden. Aktive Benutzerbeteiligung ist in dieser Hinsicht unabdingbar.

1 Einleitung

Während in den Frühzeiten der Software-Ergonomie intuitive und empirische Vorgehensweisen beim Entwurf von Oberflächen und Dialogen dominierten, stehen den Software-Entwicklern heute zahlreiche Standards, Regelwerke, Gestaltungsrichtlinien und Gestaltungsempfehlungen zur Verfügung. Neben den immer noch dominierenden papierbasierten Materialien (z. B. [1, 2, 3]) gibt es mittlerweile einige Informations- und Beratungssysteme, die als Hypertexte oder Hypermedia-Anwendungen vorliegen (z. B. [4, 5, 6, 7]). Einige verbinden Werkzeuge zur Oberflächen- und Dialoggestaltung oder zur Evaluation mit Richtlinien und Empfehlungen (z. B. [8, 9, 10]). Alle uns bekannten Systeme dieser Art sind jedoch ausschließlich für die Benutzung durch einzelne Entwickler konzipiert.

Software-Entwicklung ist dagegen meist ein hochgradig arbeitsteiliger Prozeß, der ein erhebliches Ausmaß an Kooperation zwischen allen daran Beteiligten erfordert. Für den Fall der Kooperation zwischen Entwicklern und Benutzern - oft als Benutzerpartizipation bezeichnet - ist dies auch auf den Veranstaltungen dieser Konferenzserie thematisiert worden. Dagegen spielt die Kooperation bei der Bearbeitung software-ergonomischer Fragen so gut wie keine Rolle. Dies zeigt sich auch im Fehlen von kooperativ zu benutzenden Unterstützungssystemen für die software-ergonomische Gestaltung.

Das verwundert um so mehr, da Konzepte wie „Lernende Organisation" oder „experience factory" (z. B.[11]) in der Diskussion um die Wettbewerbsfähigkeit eines Unternehmens eine zunehmende Rolle spielen. Ein strategischer Faktor ist in diesem Zusammenhang das im

Unternehmen vorhandene Wissen, dessen Nutzung und Management im Fokus der fachübergreifenden Diskussion steht. Da dieses Wissen in der Regel verteilt ist, kommt der Kooperation und Kommunikation ein hoher Stellenwert zu. Für den Fall der Software-Entwicklung konnte dies auch empirisch sehr gut nachgewiesen werden: Im IPAS-Projekt [12, 13, 14] wurden durch Mitarbeiterbefragungen in 29 verschiedenen Software-Entwicklungsprojekten 1679 Tätigkeiten analysiert. Davon waren mehr als 30 % direkt oder indirekt kommunikativen Tätigkeiten zuzuordnen (21% Beratungen, Gespräche; 12% Organisation). Weitere 12% waren dem selbständigen Wissenserwerb gewidmet und solchen kommunikativen Tätigkeiten, wie der Konzipierung und Durchführung von Mitarbeiterschulungen sowie Reisetätigkeiten und Teilnahme an Weiterbildungen. Nicht nur die Häufigkeit kooperativer und kommunikativer Tätigkeiten ist beeindruckend, sondern auch ihre Relevanz: Besonders erfolgreiche Software-Entwickler unterscheiden sich von ihren Kollegen neben anderem durch Kommunikations- und Kooperationskompetenz [14, 15]. Trennt man Software-Projekte danach, ob zwischen den Projektmitarbeitern über- oder unterdurchschnittlich viel kommuniziert wird (unter Fortlassung der Projekte mit mittlerem Kommunikationsaufwand), dann ergibt sich, daß in Projekten, in denen mehr kommuniziert wird, die Termin- und Kostenpläne besser eingehalten werden, ein größerer Projekterfolg erreicht wird, die entstandene Software später leichter änderbar ist und insgesamt eine bessere Teameffizienz vorliegt.

Wenn die Kommunikations- und Kooperationsprozesse sich nicht nur spontan entfalten, sondern gezielt durch organisatorische und/oder technische Maßnahmen gefördert werden, sprechen wir von Wissensmanagement.

Die Modellvorstellungen zum Wissensmanagement sind zahlreich; einen interessanten Ansatz bildet dabei das modulare Konzept von Probst et al. [16], da es insbesondere für angewandte Fragestellungen einen brauchbaren Zugang liefert. Die (miteinander vernetzten) Kernprozesse dieses Konzeptes stellt die folgende Tabelle dar:

Baustein	spezifische Fragestellungen
Wissensidentifikation	Wie schaffe ich Transparenz über das im Unternehmen vorhandene Wissen? Wer weiß was?
Wissenserwerb	Welche Wissensquellen lassen sich außerhalb des Unternehmens erschließen? Welche sind relevant?
Wissensentwicklung	Wie bilde ich (auf individueller und kollektiver Ebene) neue Fertigkeiten / Fähigkeiten aus?
Wissensverteilung/ Wissensaustausch	Wie bringe ich das notwendige Wissen an den richtigen Ort?
Wissensnutzung	Wie stelle ich die Nutzung des zur Verfügung stehenden Wissens sicher? Wie beseitige ich bestehende Barrieren?
Wissensbewahrung	Wie bewahrt / pflegt das Unternehmen sein Wissen und schützt sich vor Wissensverlust?

Tab. 2: Kernprozesse des Wissenmanagements nach Probst, Raub & Romhardt [16]

In bezug auf **Wissensverteilung / Wissensaustausch** bieten sich neue Medien geradezu an. Durch die Verwendung geeigneter Informations- und Kommunikationssysteme der im Rahmen der CSCW-Forschung entwickelten Groupware (zur Klassifikation siehe [17]) läßt

sich breit verteiltes Erfahrungswissen der Organisationsmitglieder bündeln und bereitstellen, wovon insbesondere Teamarbeitsprozesse profitieren [18, 19].

Die Grundidee des hier vorzustellenden Ansatzes besteht darin, Kommunikation und Kooperation zwischen Entwicklern zu fördern, indem ihnen ein kooperativ zu benutzendes hypermediales Informations- und Beratungssystem (im folgenden kurz Hyperbase genannt) im Rahmen eines Intranets zur Verfügung gestellt wird. Diese Hyperbase soll ihnen zunächst eine Menge von Basis-Informationen zur Software-Ergonomie anbieten, so wie es auch die oben genannten, für die individuelle Nutzung konzipierten Informations- und Beratungssysteme tun.

Zugleich bietet die Hyperbase aber auch die Möglichkeit, die Erfahrungen, die im Prozeß der Gestaltung von Benutzungsschnittstellen von einzelnen Entwicklern oder kleinen Teams von 2-3 Personen gemacht wurden, aufzubereiten, abzuspeichern und sie so allen Mitgliedern einer größeren Organisationseinheit zur Verfügung zu stellen. Dabei können und sollen Erfahrungen, die sich auf Prozesse beziehen, die nicht zur Gestaltung von Benutzungsschnittstellen gehören, ausdrücklich miteinbezogen werden.

Mit der Einführung einer solcher Hyperbase ist die Erwartung verbunden, daß sich der Anteil von Kommunikation- und Kooperationsprozessen bei der software-ergonomischen Gestaltung von Systemen erhöht und daß Entwicklerteams letztlich mit weniger Aufwand bessere Benutzungsschnittstellen produzieren. Natürlich ist auch bei der Entwicklung und Einführung der Hyperbase in allen Phasen eine partizipative Beteiligung aller Mitarbeiter / Gruppen sicherzustellen [20, 21].

2 Fragestellung

Wie bei allen Vorhaben dieser Art empfiehlt es sich, vor der Entwicklung eines solchen Systems durch eine Anforderungsanalyse den Unterstützungsbedarf und die Erwartungen der zukünftigen Benutzer an solch eine Hyperbase zu erfassen. Im einzelnen fragten wir:

4. Benötigen die von uns untersuchten Software-Entwickler überhaupt Unterstützung auf dem Gebiet der Software-Ergonomie? Wie ausgeprägt ist ihr Wissen auf diesem Gebiet? Wie gut können sie es anwenden?
5. Wie ist das Wissen auf verschiedene Personen verteilt? Lohnt sich ein Austausch von Wissen?
6. Spielen Kenntnisse zur Software-Ergonomie aus der Sicht der Entwickler überhaupt eine wichtige Rolle in ihrer Arbeit?
7. Welche Rolle wird dem Erfahrungswissen auf dem Gebiet der Software-Ergonomie zugeschrieben?
8. Wird Erfahrungswissen jetzt schon dokumentiert? Welche Techniken werden dabei eingesetzt?
9. Auf welchen Wegen wird jetzt Wissen erworben?
10. Welche Unterstützungsformen wünschen sich die Software-Entwickler?
11. Welche Inhalte und Funktionen sollte eine Hyperbase zum Thema Software-Ergonomie und Wissensaustausch zur Verfügung stellen?

Software-Entwickler befinden sich generell in einer Doppelrolle. Einerseits sind sie Entwickler von Software, die von anderen benutzt wird, andererseits sind sie selbst Benutzer

z. B. von Betriebssystemen, Entwicklungswerkzeugen und Standardsoftware. In unserer Untersuchung ging es allerdings nur um ihre Rolle als zukünftige Benutzer der konzipierten Hyperbase zur Software-Ergonomie.

3 Methodik und Durchführung

3.1 Stichprobe

Alle befragten Personen sind diplomierte Informatiker und arbeiten als Entwickler/Programmierer in einem Unternehmen, das komplexe elektronische Anlagen mit einem hohen Anteil von Bedien-Software erstellt. Die Arbeitsaufgabe erfordert im wesentlichen die Spezifikation und Entwicklung datentechnischer Systemkonzepte unter Berücksichtigung ihrer konkreten Nutzungsbedingungen, die Integration der Anwendungen in vorhandene Umgebungen sowie die Pflege der Produkte. Die Abteilung ist dabei in mehrere Teams aufgeteilt, die jeweils eigenständige Projekte bearbeiten.

In die Untersuchung einbezogen wurden insgesamt 24 Mitarbeiter, davon waren 6 Teamleiter, 18 Entwickler.

Testaufgabe, Fragebogen & Interview	18 Entwickler	6 Teamleiter
Durchschnittsalter	34	41
Beschäftigungsdauer im Rahmen der Software-Entwicklung allgemein	11	17
Beschäftigungsdauer im Rahmen der Entwicklung unternehmensspezifischer Software	7	12

Tab 2: Angaben zu befragten Personen (Jahre; gerundete Mittelwerte)

Die Teamleiter und Entwickler unterscheiden sich hinsichtlich aller drei Merkmale signifikant (a=0,05) voneinander; sie sind älter und beruflich länger mit der Entwicklung von Software befaßt.

3.2 Methoden und Ablauf der Untersuchung

Die Untersuchung vollzog sich in mehreren Schritten:

Zu Beginn gab es eine ca. zweistündige Zusammenkunft aller Beteiligten (Software-Entwickler, Management, Mitglieder des Untersuchungsteams). In dieser Zusammenkunft wurde das Anliegen der Untersuchung vorgestellt und mit den Entwicklern diskutiert. Ferner wurde der weitere Ablauf erläutert und die verschiedenen Methoden erklärt. Dann wurden Fragebögen ausgeteilt, die von den Entwicklern individuell am Arbeitsplatz oder zu Hause ausgefüllt werden sollten. Die Fragebögen enthielten Fragen zu biographischen Daten, zur Bedeutung der Software-Ergonomie in den Projekten der Abteilung, zu den Kenntnissen auf dem Gebiet der Software-Ergonomie und zur Kenntnis und Relevanz von Informationsquellen. Bei der Auswahl und Formulierung der Fragen orientierten wir uns an früheren Studien [22, 23, 24, 25, 26] und berücksichtigten neuere Entwicklungen am Lehrstuhl. Zum Ende der Zusammenkunft bearbeiteten die Entwickler eine heuristische Evaluationsaufgabe,

die von Nielsen [27] unter der Bezeichnung *Travel Weather* entwickelt und als Übungsaufgabe publiziert wurde. Wir wählten diese Aufgabe, weil sie vom Inhalt her recht ähnlich zu den realen Projekten der Untersuchungsteilnehmer war. Die Aufgabe war überschaubar und in gut 45 Minuten zu bearbeiten. Nielsen bietet Lösungen als Ergebnis einer Evaluation durch vier Usability-Experten zum Vergleich an.

Am Nachmittag nach der Startsitzung begannen die Einzelinterviews mit den Entwicklern. Die Teilnehmer brachten zu den Interviews die ausgefüllten Fragebögen mit. Die Interviews bezogen sich zum Teil auf die Antworten im Fragebogen (insbesondere wurden Begründungen zu einzelnen Antworten erfragt), zum Teil waren es auch freie Fragen, die den Entwicklern Gelegenheit gaben, über ihre Arbeitsverfahren, Erfahrungen, Wünsche und Erwartungen zu berichten. Die Interviews wurden auf Tonband aufgezeichnet.

Nach den Interviews trat eine längere Auswertungsphase ein, während der die Lösungsvorschläge für die Testaufgabe und die Antworten in den Fragebögen ausgewertet wurden. Außerdem wurden die Interview-Daten transkribiert und inhaltsanalytisch ausgewertet.

Alle Ergebnisse wurden ca. zwei Monate später in einem gemeinsamen Workshop mit den Entwicklern vorgestellt. Sie dienten dazu, den Entwicklern eine Rückmeldung über das software-ergonomische Wissen in der Abteilung, über die Erfahrungen und Bedarfsanmeldungen der Untersuchungsteilnehmer zu geben und sie zur Mitarbeit zu motivieren. Zugleich legten die Ergebnisse die Grundlage für die dreistündige Diskussion in zwei Teilgruppen, die sich zum einen mit den Inhalten und Funktionen einer zukünftigen Hyperbase und zum anderen mit Organisationsfragen rund um die Entwicklung, Nutzung und Pflege der Hyperbase beschäftigten.

4 Ergebnisse

4.1 Aktueller Wissensstand zur Software-Ergonomie

Von den 31 Usability-Problemen, die im Original von Nielsen [27] beschrieben wurden, sind von den Probanden insgesamt 24 erkannt worden. Dabei wurden im Mittel von jedem Teilnehmer sieben Probleme benannt. Die Spanne reicht von vier bis zu zehn Problemen. Die Probanden haben zusätzlich insgesamt 26 neue Probleme aufgelistet, die die bei Nielsen (1993) einbezogenen Experten nicht angesprochen hatten. Im Mittel hat jeder Teilnehmer 3 neue Gestaltungsmängel oder Verbesserungsmöglichkeiten genannt (Spanne von eins bis vier). Die nebenstehende Abbildung faßt die Ergebnisse zusammen.

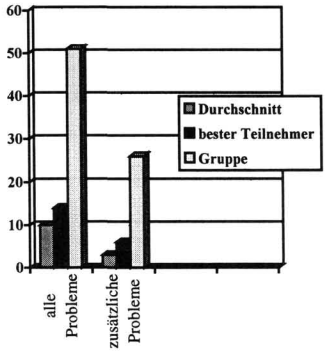


Abb. 1: Die linke Säulengruppe zeigt die Häufigkeiten für alle 50 software-ergonomischen Mängel bzw. Verbesserungsvorschläge, die von den Teilnehmern insgesamt benannt wurden. Die rechte Säulengruppe stellt diese Häufigkeiten für die Gestaltungsaspekte dar, die in der Beschreibung von Nielsen nicht erwähnt wurden. Der Vorteil der Nominalgruppe wird in beiden Kategorien sehr deutlich.

Zur Zeit existieren keine Vergleichsdaten, so daß keine definitive Aussage über das Ausmaß des individuell verfügbaren und anwendbaren Wissens getroffen werden, obwohl der Mittelwert von 22% individuell erkannter Probleme durchaus akzeptabel erscheint. Wichtiger aber als der Durchschnittswert ist die Relation zwischen individueller Leistung und Leistung der Gruppe.
Unterschiede zwischen Teamleitern und Entwicklern sind hinsichtlich genannter Mängel nicht ausweisbar. Dieser Befund wird durch den Fragebogen (Selbsteinschätzung zum software-ergonomischen Wissen) bestätigt, da Teamleiter und Entwickler hier gleichermaßen über ihren eigenen Wissensstand urteilen.

4.2 Subjektive Selbsteinschätzung des Wissensstandes

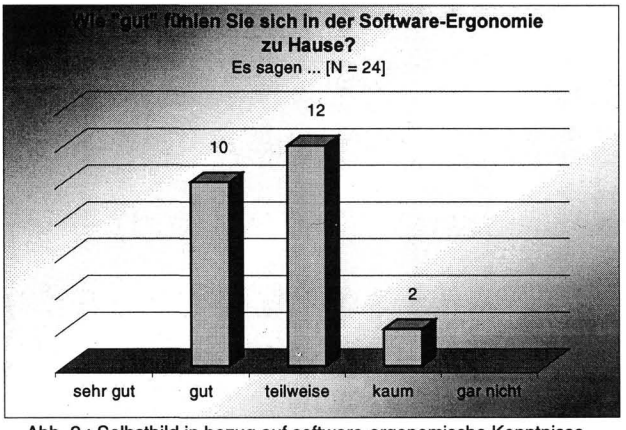


Abb. 2 : Selbstbild in bezug auf software-ergonomische Kenntnisse

Wie spiegelt sich die Situation aus der Sicht der Entwickler wider? In welchem Maße fühlen sich die befragten Entwickler auf dem Gebiet der Software-Ergonomie zu Hause?

Selbstbild und Leistung in den Testaufgaben korrelieren allerdings nicht signifikant, was auch an der geringen Streuung der Fragebogendaten liegt.

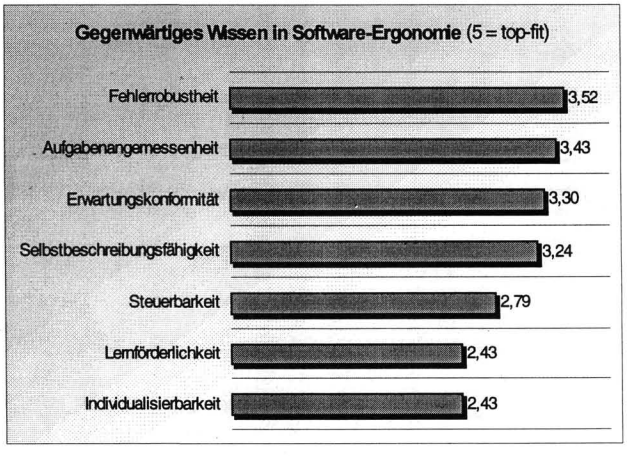


Abb. 3: Selbsteinschätzung der Entwickler hinsichtlich ihres Wissens zu den verschiedenen Kriterien der Software-Ergonomie (Rating 1 - 5). Die Unterschiede zwischen den einzelnen Kriterien sind nicht signifikant.

Fragt man bei der Selbsteinschätzung nach dem Wissen um die Kriterien der ISO 9341/Teil 10 so zeigt sich, daß die Entwickler meinen, relativ viel über Aufgabenangemessenheit und Fehlerrobustheit zu wissen. Ihr Wissen über Erwartungskonformität schätzen sie mittelmäßig ein, während das Wissen über Steuerbarkeit, Individualisierbarkeit und Lernförderlichkeit eher als gering eingeschätzt wird.

Das schlägt sich sowohl in den Ergebnissen der Testaufgabe als auch in den Antworten im Rahmen der Interviews nieder.
In bezug auf die Selbsteinschätzung zum software-ergonomischen Wissen sind keine Unterschiede zwischen Teamleitern und Entwicklern feststellbar. Dieser Befund wird durch die Ergebnisse der Testaufgabe bestätigt, da Teamleiter und Entwicklern hier gleichermaßen gut abschneiden.

4.3 Wie ist das software-ergonomische Wissen verteilt?

Das ist die zentrale Frage, wenn es um den Austausch von Wissen geht. Ein Austausch macht z. B. wenig Sinn, wenn alle Mitarbeiter in etwa über das gleiche Wissen verfügen. Das dies nicht so ist, läßt sich am Beispiel der Lösungen der Testaufgabe zeigen.

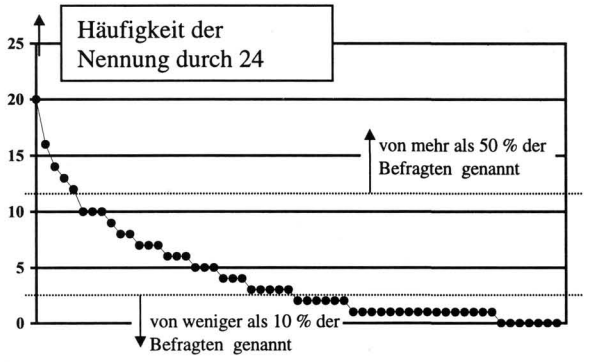


Abb. 4: Absolute Häufigkeit, mit der die einzelnen Probleme (insgesamt 57) genannt wurden. Jeder Punkt steht für die Häufigkeit, mit der die einzelnen Probleme jeweils individuell entdeckt wurden. Das Bild zeigt, daß nur fünf Probleme übereinstimmend von mehr als 50% der Entwickler genannt wurden, während 29 von weniger als 10% entdeckt wurden.

Wenn es um Wissensaustausch geht, spielt nicht nur die Verteiltheit des Wissens eine Rolle, sondern auch die Symmetrie oder Asymmetrie der Verteilung. Einige Entwickler wissen weniger als andere. Sie könnten sicher von einem Wissensaustausch profitieren. Lohnt es sich jedoch für diejenigen, die jetzt schon über mehr Wissen verfügen, andere daran partizipieren zu lassen? Um diese Frage zu beantworten haben wir gegenübergestellt, welche Probleme bei der Testaufgabe von den leistungsstarken Testpersonen (dadurch definiert, daß sie mehr als elf Probleme benannt haben) und den leistungsschwachen Testpersonen (dadurch definiert, daß sie weniger als neun Probleme benannt haben) benannt worden. Folgendes Bild ergibt sich:

1 ●●●	2 ●●●	3 ●●●●	4 ●●●●	5	6	7 ●●	8
9 ●●●●	10 ●●●●	11 ●●●●	12 ●●	13 ●●●	14 ●●	15 ●●	16 ●●
17 ●●	18 ●●	19 ●●	20	21 ●●●●●	22	23	24 ●
25 ●●	26 ●●	27	28 ●●	29 ●●●●●	30 ●●	31 ●●	32 ●●
33 ●●	34 ●●●●	35 ●●●	36 ●●	37 ●●●●	38 ●●●	39 ●●	40 ●●●●
41 ●●●●	42 ●●●●	43 ●●●●	44 ●●●	45 ●●●●	46 ●●●●	47 ●●●●	48 ●
49 ●	50 ●	51 ●	52 ●	53 ●	54 ●●●●	55 ●●●●	56 ●●●●
57 ●●●●							

Abb. 5: Die Nummern kennzeichnen die software-ergonomischen Probleme. Nr. 1 bis 31 entsprechen den Angaben von Nielsen (1993). Die Nummern 32 bis 57 kennzeichnen zusätzlich gefundene Probleme.
Je mehr Punkte auf ein Problem entfallen, desto weniger verteilt sich das Wissen auf verschiedene Personen. Felder ohne Punkte kennzeichnen dabei die Probleme, die überhaupt nicht angesprochen wurden (siehe auch Tab. 3).
Die Probleme 21 und 29 werden von jeweils nur einer leistungsschwachen Testperson angesprochen.

nur von einer leistungsschwachen Testperson entdeckt	●●●●●
nur von einer leistungsstarken Testperson entdeckt	●●●●
Von mehreren leistungsstarken Testpersonen entdeckt	●●●
Sowohl von leistungsstarken als auch von leistungsschwachen Testpersonen entdeckt	●●
Weder von leistungsstarken noch von leistungsschwachen Testpersonen entdeckt	●
Von keiner der 24 Testpersonen entdeckt	

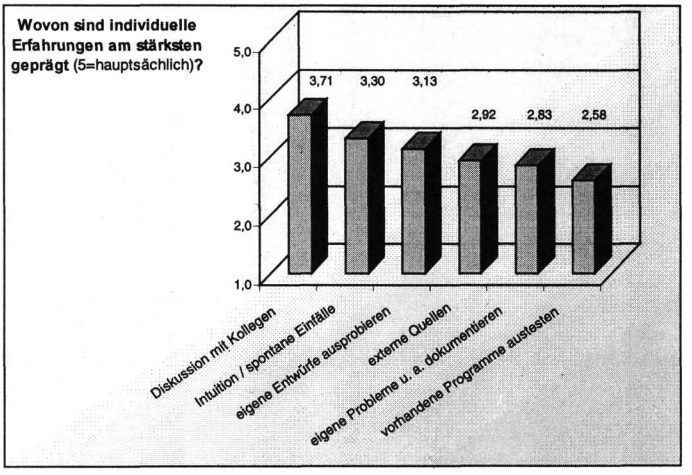
Tab. 3: Verteiltheit des Wissens

4.4 Wo kommen Wissen und Erfahrung her?

In den Interviews wurde sehr deutlich, daß viele Entwickler auf individuell archivierte Problem- oder Fallsammlungen ihrer Karriere zurückgreifen. Von den im Interview befragten 18 Entwicklern dokumentieren insgesamt 15 auftretende Probleme, etwaige Fehler oder Lösungen einzelner Projektabschnitte individuell für sich. Etwa 1/3 der Entwickler dokumentiert neben allgemeinen Problemen auch solche, die direkt im Zusammenhang mit software-ergonomischen Fragen stehen.

Die Art und Weise der individuellen Dokumentation von Erfahrung ist dabei sehr verschieden: die Spanne reicht von einfachen papierbasierten Sammlungen wie Notizen (sog. „schlaue Bücher") und Papierskizzen (mock-ups) bis hin zu elektronisch archivierten Beispieloberflächen, „digitalen" Tagebüchern (die Vorgehensweisen, Probleme und Fehler für einzelne Schritte enthalten) und eingescannter Literatur.

Darüber hinaus interessierte uns, wovon die individuelle Erfahrung der Mitarbeiter in bezug auf software-ergonomisches Wissen besonders geprägt ist und aus welchen anderen Quellen sie ihr Wissen schöpfen.



Im Fragebogen geben die Entwickler an, daß „individuelle Erfahrung" neben externen Quellen am stärksten durch die Diskussion mit Kollegen geprägt sei. Diese Wertung schlägt sich sowohl in den Fragebogendaten (siehe Abb. 6) als auch in den Interviews und den Diskussionen im Rahmen der Workshops nieder.

Abb. 6 : Wovon individuelle Erfahrung geprägt ist

Deutlich wurde sehr schnell, daß elektronische Unterstützungssysteme den direkten Kontakt der Mitarbeiter nicht ersetzen können, sondern immer eine Ergänzung sein sollen. Entwickler wünschen sich bspw. in einer Hyperbase Informationen darüber, wer von den Kollegen was macht, woran einzelne Teams arbeiten, wofür es Lösungen gibt; sie möchten vorab Ansprechpartner und Anregungen finden, dann zielorientiert mit diesen Kollegen diskutieren und so die Rückkopplung zu eigenen Erfahrungen herstellen.

Ebenfalls im Fragebogen erfaßt wurde, welche externen Quellen wie häufig von den Entwicklern benutzt werden, um das Wissen über die Benutzerfreundlichkeit von Software zu erweitern (siehe Abb. 7). Dabei dominieren Software-Werkzeuge (z.B. Starview), Fachliteratur, sog. Vorbildlösungen der Abteilung, dokumentierte Probleme / Fehler sowie der Austausch mit Ergonomie-Spezialisten (bspw. im Rahmen von Seminaren). Standards bzw. Guidelines verlieren demgegenüber an Bedeutung. Gleiches gilt für selbst archivierte Sammlungen von Positiv- bzw. Negativbeispielen, die jedoch im bisher vorliegenden Umfang auch nicht sehr zahlreich sind (lediglich zwei Kollegen archivieren Positiv-/Negativbeispiele).

Netzbasierte elektronische Informationssysteme (WWW, Newsgroup) werden kaum benutzt; dies liegt daran, daß an einigen Arbeitsplätzen kein Internetanschluß verfügbar oder die Netzkapazität unzureichend ist. Klassische Einzelplatzsysteme zur Unterstützung (Beratung und Information, Expertensysteme, User Interface Management Systeme) stehen nur eingeschränkt zur Verfügung und werden so gut wie nicht benutzt. Die Möglichkeit zur Teilnahme an Kongressen besteht kaum, diese Form der Wissensgewinnung schneidet am schlechtesten ab.

4.5 Anforderungen der Entwickler an ein Unterstützungssystem

Für die Erfassung der Anforderungen und Wünsche, die Entwickler an ein Unterstützungssystem haben, wurden im wesentlichen drei Quellen herangezogen: Fragebogen, Interview und Workshop-Diskussion.

Im Fragebogen wurden die Entwickler einerseits befragt, wie häufig sie welche Informationsquellen nutzen (siehe 4.4). Darüber hinaus wollten wir wissen, wie hilfreich sie die vorgegebenen Informationsquellen für sich einschätzen. Zwischen beiden Fragen zeigen sich klare Unterschiede in den Rangfolgen (siehe Abb. 7).

Es zeigt sich deutlich, daß die derzeitige Rezeption externer Quellen nicht den tatsächlichen Anforderungen der Entwickler entspricht. Sie schreiben den einzelnen Quellen generell mehr Potential zu, als ihnen die gegenwärtige Nutzung ermöglicht. Sie wünschen sich durchweg eine Intensivierung der Informations- und Weiterbildungsangebote; dies sowohl zu Themen der Software-Ergonomie als auch zu anderen Themen (siehe unten).

In der Rangreihe (hinter der sich die favorisierte Quellennutzung niederschlägt) dominieren der direkte Austausch mit Spezialisten, der Zugriff auf Vorbildlösungen der Abteilung und dokumentierte Fehler / Probleme, die Nutzung von Expertensystemen, Beratungs- und Informationssystemen sowie die Verwendung von Standards und Beispiellösungen.

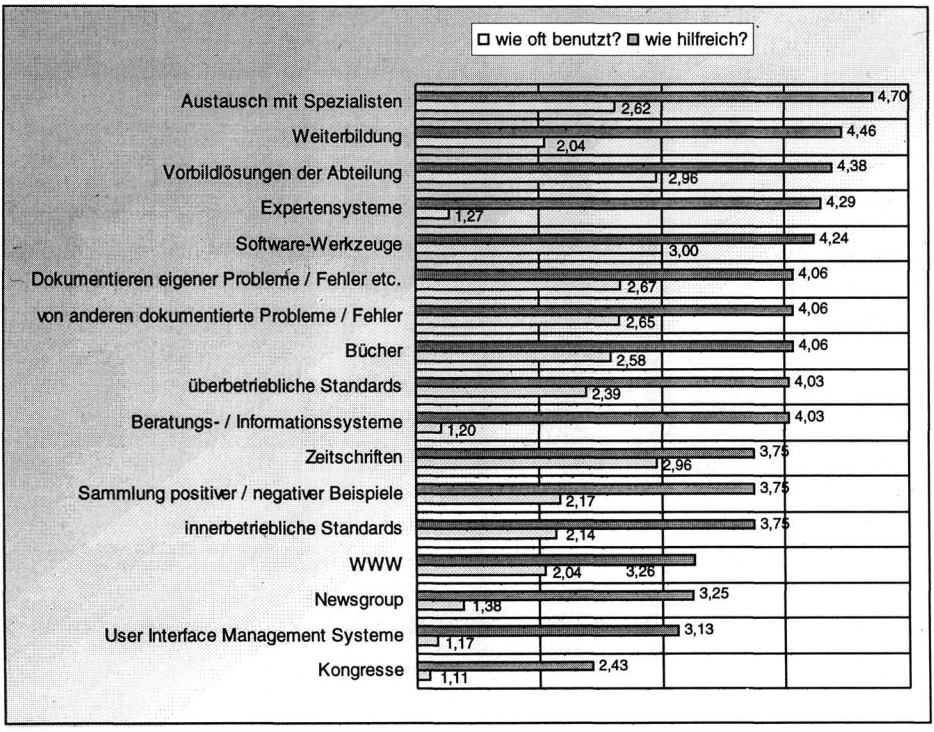


Abb. 7: momentane Häufigkeit der Nutzung bestimmter Quellen und deren Bewertung (5= sehr oft/sehr hilfreich)

Bei den Interviews mit den Entwicklern konnten die individuellen Ideen und Wünsche der Teilnehmer zu den Inhalten und der Realisierung der Hyperbase erfragt werden. Was sollte in einer Unterstützungsumgebung enthalten sein? Wie stellen sich die Entwickler die Gestaltung des Systems vor? Die Antworten lassen sich in drei große Bereiche gliedern, die alle von dem Gedanken der intensiven Kommunikation zusammengehalten werden.

1. Die meisten Ideen beschäftigen sich mit dem Thema Software-Ergonomie.
2. Es gibt Wünsche nach Informationen auf Projektebene - vor allem soll der Austausch zwischen den Projekten durch ein netzbasiertes Informationssystem gefördert werden.
3. Den dritten Bereich bilden die "sonstigen" Anforderungen an eine solche Hyperbase.

Die Schwerpunkte innerhalb dieser drei Bereiche lassen sich wie folgt komprimieren:

Unter der Rubrik **Softwareergonomie** werden Tips/Tricks und Erfahrungen zum Software-Entwurf gefordert. Dies sollte in einer sehr praktischen Sprache und ganz konkret erfolgen. In einer komprimierten Form soll das an guten und schlechten Beispielen orientierte Lehrbuchwissen (Farben, Schriften etc.) aufbereitet vorliegen und durch das Wissen der Kollegen laufend ergänzt werden (Verknüpfungen der Inhaltsteile untereinander). Hierzu gehört der Wunsch nach Verweisen auf entsprechende Verordnungen und Normen (innerbetriebliche und externe). Vorhandene (teilweise unbekannte) Verwendungsvorschriften des Hauses sollen aufbereitet in einen Hypertext o. ä. eingebunden sein. Eine Verbindung der alltäglichen Aufgaben mit den Maßgaben der Software-Ergonomie soll eine Sammlung aller in der Abteilung entwickelten Oberflächen herstellen, die durch 2-3 Personen nach SE-Kriterien beurteilt worden sind.

Um die Konstruktion der Hyperbase aus inhaltlicher und zeitlicher Perspektive zu optimieren, mußten die aus Sicht der Entwickler zunächst wichtigsten Inhalte zu Themen der Software-Ergonomie identifiziert werden. Dazu wurde im Rahmen des Workshops die jeweils individuelle Gewichtung potentieller Inhalte und Funktionen einer Hyperbase ermittelt; 7 Teilnehmer konnten in der von uns vorgegebenen Matrix individuell pro Zelle einen Punktwert von 0 bis 4 vergeben. Die folgende Abbildung stellt die über alle Teilnehmer gebildete Matrix dar. Graue Zellen kennzeichnen die zunächst wichtigsten Inhalte und Funktionen:

Inhalte / Funktionen	Generische Schnittstellen-objekte	Generische Funktionen und Interaktions-techniken	Aufgabenbezogene Gestaltung der Funktionalität	Aufgabenbezogene Gestaltung der Semantik (Objekte und Funktionen)	Strukturierung der Oberfläche (Gruppierung etc.)	Kodierung der Interaktions-elemente (Form, Farbe, Text, Icon)	Geräte & Medien für die Interaktion (bspw. Spezial-geräte, VR, Spracheingabe etc.)	Punktzahl
Planen			12	12	4		4	**32**
Beraten	20	20	22	28	23	15	12	**140**
Vorschlagen	12	13	16	18	17	10	13	**99**
Entscheiden	4	4	5	5	8	5	8	**39**
Informieren	17	14	25	25	22	20	13	**136**
Ausführen	5	3		5		3	1	**17**
Überprüfen	1	5					1	**7**
Σ	**59**	**59**	**80**	**93**	**74**	**53**	**52**	**470**

Tab. 4: Häufigkeitsverteilung gewünschter Inhalte und Funktionen

Die **projektbezogenen übergreifenden Informationen** sollen eine Kommunikationslücke schließen. Wie in vielen vergleichbaren Unternehmen verbleiben Informationen in kleinen Gruppen, obwohl sie durchaus für alle Mitarbeiter von Interesse wären. Als Basisanforderung

wurde mehrmals eine Endauswertung zu jedem einzelnen Projekt genannt, die mit einer Personenliste gekoppelt ist, auf der vermerkt ist: Wer hat was gemacht? Arbeitet jemand z. Z. an ähnlichen Problemen wie ich? Welche Klassenbibliotheken gibt es, welche sind von wem entwickelt worden? usw.

Weiterhin sollten Infos zu allen aktuell laufenden Projekten (wer macht aktuell was, Prozeßverlauf, welche Tools werden verwendet, wie sehen die Oberflächen aus, ist eine Datenbankanbindung realisiert usw.) bereitstehen und eine mit Beispielen unterlegte Sammlung von Problemen oder Fehlern, die in den verschiedenen Projekten aufgetreten sind (veranschaulicht an Graphiken, Quellcodes, Texten, Screendumps etc.).

Unter der Rubrik **sonstige Anforderungen** wurden allgemeine Informationen summiert, die sich auf Schwierigkeiten mit bestimmten Betriebssystemen, Eigenheiten einzelner Programmiersprachen und Administratorproblemen beziehen. Gewünscht werden aber auch Informationen zu aktuellen Entwicklungstools, neuer Hardware und Eigenschaften der technischen Geräte, die von der Software angesteuert werden. Einen eigenständigen Schwerpunkt nehmen Informationen zu Datenbanken (Schnittstellen, Anlegen von DB, Anwendungsfälle) ein.

Auch außerhalb einer elektronischen Unterstützungsumgebung - oder parallel zu dieser - wurden immer wieder eher traditionelle Möglichkeiten der abteilungsinternen Kommunikation und Weiterbildung eingefordert. Genannt wurde die stärkere Förderung des Austauschs von und der Diskussion über Zeitschriftenartikel, eine regelmäßige interne Schulung durch vorbereitete Vorträge von Mitarbeitern aus der Abteilung mit hin und wieder auch externen Referenten. Ganz wichtig scheint eine institutionalisierte, regelmäßige Diskussionsrunde über aktuelle Probleme / Methoden / Entwicklungsfragen über die Grenzen der Projektteams hinweg. Dazu gehören auch Feedbackrunden mit mehreren Kollegen zu aktuellen Gestaltungslösungen bei denen dann aufgetretene Fehler/Irrwege gemeinsam diskutiert und Lösungen gefunden werden können.

5 Diskussion und Ausblick

Wissensmanagement erfordert zunächst die Identifikation des vorhandenen Wissens und die Analyse der Kommunikations- und Kooperationsprozesse innerhalb einer definierten Organisationseinheit. Mit dem beschriebenen Vorgehen war es im konkreten Fall möglich zu entscheiden, ob und in welcher Form Entwickler in bezug auf software-ergonomische Kenntnisse Unterstützung benötigen. Es wurde eine Methodenkombination aus Testaufgaben, Fragebögen, Interview- und Moderationstechniken zusammengestellt, mit der neben Aussagen zum aktuellen Wissensstand (Wer weiß was, entspricht die Selbsteinschätzung der tatsächlichen Leistung, wovon hängt die jeweils individuelle Erfahrung ab, welche Quellen werden als hilfreich beurteilt?) erneut nachgewiesen werden konnte, daß die einzelnen Gruppenmitglieder über jeweils anderes Wissen verfügen. Dieser Beleg der sog. *Verteiltheit des Wissens* war ein ausschlaggebendes Argument zur Differenzierung weiterer Maßnahmen.

Da Benutzerbeteiligung ein notwendiges Merkmal zur erfolgreichen Gestaltung von Unterstützungsmitteln ist, wurden im Rahmen von Interviews und Workshops Bedürfnisse und Erwartungen der potentiellen Nutzer an ein Unterstützungssystem ermittelt sowie Ideen zur möglichen Umsetzung generiert und auf Machbarkeit geprüft. Die anschließenden

Schritte sind nur bedingt parallel möglich; deshalb kam es darauf an, die Implementierung notwendiger Inhalte und Funktionen des elektronischen Unterstützungssystems auch zeitlich zu strukturieren.

Die Einführung der Hyperbase soll ebenfalls begleitet und evaluiert werden. Hier ist wichtig, die Interaktion der Benutzer mit dem System bzw. seine Nutzung (Ort, Zeit, Motive, Erfolge usw.) zu analysieren. Darüber hinaus ist zu prüfen, inwieweit sich durch die Nutzung der Hyperbase Kooperation und Kommunikation in der Abteilung verändern und sich die Qualität (prozeß- und produktbezogen) der Software-Entwicklung verbessert.

Nicht unerwähnt bleiben soll die Möglichkeit, daß die hier vorgestellte Methodenkombination leicht auf ähnliche Fragestellungen in anderen Unternehmen adaptiert werden kann. Es bietet sich darüber hinaus an, die geschilderte Vorgehensweise auf andere Probleme und Aufgaben des Wissensmanagements zu übertragen bzw. zu verallgemeinern.

6 Literatur

Smith, L.S. & J. N. Mosier (1986): Guidelines for designing user interface software. Bedford: The Mitre Corporation.

IBM (1991): System Application Architecture: Common User Access. Cary: IBM.

ISO 9241 (1992): Visual Display Terminals (VDT's) used for office tasks, Ergonomic Requirements, Part 10: Dialogue Principles.

Pearl, G. & Moorhead, A.J. (1989): NaviText™ SAM. Westford: Northern Lights Software.

Iannella, R. (1991): BRUITΔSAM™. Gold Coast: Bond University.

Vanderdonckt, J. & F. Bodart: (1996): The "Corpus Ergonomicus": a Comprehensive and Unique Source for Human-Machine Interface Guidelines, In: Ozog, A. F. & G. Salvendy (Hrsg): Advances in Applied Ergonomics. Proceedings of 1st International Conference on Applied Ergonomics ICAE'96 (Istanbul, 21-24 May 1996). Istanbul: West Lafayette, (S. 162-169).

Wandke, H. & J. Hüttner (1995): Unterstützung bei der Gestaltung von Benutzungsschnittstellen durch die Bereitstellung von Software-Ergonomie-Wissen in einem Informations- und Beratungssystem. In: H.-D. Böcker (Hrsg.): Software-Ergonomie 95, Mensch-Computer-Interaktion: Anwendungsbereiche lernen voneinander. Stuttgart: Teubner, (S.383-393).

Forbrig, P.; Gorny, P. & A. Viereck (1993): Unterstützung des Software-Design-Prozesses durch EXPOSE. In: W. Coy, P. Gorny, I. Kopp und C. Skarpelis (Hrsg.): Menschengerechte Software als Wettbewerbsfaktor. Stuttgart: B. G. Teubner, (S. 463-479).

Balzert, H., (1994): Das JANUS-System: Automatisierte, wissensbasierte Generierung von Mensch-Computer-Schnittstellen. Heidelberg: Springer-Verlag, Issue 9, (S. 22-35).

Reiterer, H. (1995): IDA - A Design Environment for Ergonomic User. In: K. Nordby et al. (1995): Human-Computer Interaction, Interact '95, IFIP Conference. London: Chapman & Hall, (S. 305-310).

Basili, V. R.; Caldiera, G. & H. D. Rombach (1994): The Experience Factory. In: J. J. Marciniak (Hrsg.): Encyclopedia of Software Engineering. Vol. 1. New York: John Wiley & Sons.

Frese, M. & Brodbeck, F. C. (1992): Psychologische Aspekte der Software-Entwicklung (Ergebnisse des IPAS-Projekts). In: IBM Nachrichten (42) 1992, H. 309, (S. 14-19).

Brodbeck, F. C. & M. Frese (1994): Produktivität und Qualität in Software-Projekten - Psychologische Analyse und Optimierung von Arbeitsprozessen in der Software-Entwicklung. München: R. Oldenburg Verlag.

Brodbeck, F. C. (1996): Kommunikation und Leistung in Projektarbeitsgruppen. Eine empirische Untersuchung an Software-Entwicklungsprojekten. Aachen: Shaker Verlag.

Sonnentag, S. (1995): Excellent software professionals: Experience, work activities, and perceptions by peers. Behaviour & Information Technology, 14, (S. 289-299).

Probst, G.; Raub, S. & K. Romhardt (1998): Wissen managen: Wie Unternehmen ihre wertvollste Ressource optimal nutzen. Wiesbaden: Gabler (2. Auflage).

Warnecke, G.; Stammwitz, G. & F. Hallfell (1998): Intranets als Plattform für Groupware-Anwendungen. In: Industrie Management 1/98, (S. 24-28).

Oberquelle, H. (1991 a): CSCW- und Groupware-Kritik. In: Oberquelle, H. (Hrsg.): Kooperative Arbeit und Computerunterstützung. Göttingen: Verlag für Angewandte Psychologie.

Oberquelle, H. (1991 b): Kooperative Arbeit und menschengerechte Groupware als Herausforderung für die Software-Ergonomie. In: Oberquelle, H. (Hrsg.): Kooperative Arbeit und Computerunterstützung. Göttingen: Verlag für Angewandte Psychologie.

Herrmann, T. & T. Walter (1997): Participatory Design and Cyclic Improvement of Business Processes with Workflow Management Systems. In: Proceedings of the Association Information Systems 97 (AIS Indianapolis), (S. 925 - 928).

Hoffmann, M. & K.-U. Loser (1997): Mitarbeiter-orientierte Modellierung und Planung von Geschäftsprozessen bei der Einführung von Workflow-Management. In: Ortner, E. (Hrsg.): Proceedings des EMISA-Fachgruppentreffens 1997: Workflow-Management-Systeme im Spannungsfeld einer Organisation (Darmstadt).

Beimel, J., Hüttner, J. & H. Wandke (1993): Kenntnisse von Programmierern auf dem Gebiet der Software-Ergonomie: Stand und Möglichkeiten zur Verbesserung. In: Arbeits-, Betriebs- und Organisationspsychologie vor Ort. Bonn: Deutscher Psychologen Verlag, (S. 72-82).

Hüttner, J. & H. Wandke (1993): What do system designers know about software ergonmics and how to improve their knowledge? In: H. Luczak, A. Cakir and G. Cakir (Hrsg.): Work with display units 92. Amsterdam: Elsevier, (S. 304-308).

Wandke, H. (1995): Individualisierung als zentrales Problem der Unterstützung und wie man sie erreichen kann. In: Hacker,W., Rothe, H.-J., Wandke, H. und Ziegler, J. (Hrsg.): Entwicklung und Einsatz wissensorientierter Unterstützungsysteme. Bremerhaven : Wissenschaftsverlag NW Verlag für neue Wissenschaft, (S.109-132).

Wandke. H. (1995): Struktur begrifflichen Wissens im menschlichen Gedächtnis - Anregungen für den Software-Entwurf ? In: Dzida, W. und Konradt, U. (Hrsg.): Psychologie des Software-Entwurfs. Göttingen und Stuttgart : Verlag für Angewandte Psychologie, (S. 60-83).

Hüttner, J. (1997): Software-Ergonomie in mittelständischen Softwarehäusern - eine Befragung auf der CeBIT96. Poster in: Lischkowsky, R.; Velichkowsky, B. M. & Wünschmann, W. (Hrsg.): Software-Ergonomie ' 97. Berichte des German Chapter of ACM Nr. 49. Stuttgart : B. G. Teubner, (S. 362 ff.).

Nielsen, J. (1993): Usability Engineering. Cambridge, MA: Academic Press, (S. 273 ff.).

Adresse der Autoren

Prof. Dr. Hartmut Wandke
Humboldt-Universität zu Berlin
Institut für Psychologie
Oranienburger Str. 18
D – 10178 Berlin

Hartmut.Wandke@rz.hu-berlin.de

Dipl.-Psych. Jens Hüttner
Humboldt-Universität zu Berlin
Institut für Psychologie
Oranienburger Str. 18
D – 10178 Berlin

Huettner@rz.hu-berlin.de

Dipl.-Psych. Andreas Dubrowsky
Humboldt-Universität zu Berlin
Institut für Psychologie
Oranienburger Str. 18
D – 10178 Berlin

Dubrowsky@rz.hu-berlin.de