Aufgabenbezogene Dialogstrukturen für Informationssysteme

Jürgen Ziegler, Christian Janssen

Zusammenfassung

Aufgabenangemessene Gestaltung von Dialogstrukturen ist ein wesentliches Ziel der Software-Ergonomie. Die Dialogstruktur soll eine effektive und effiziente Navigation des Benutzers im System ermöglichen. In diesem Beitrag wird dargestellt, daß existierende Ansätze der Aufgabenanalyse auf einzelne Perspektiven von Aufgaben beschränkt sind. Sie sind deshalb nicht für eine Beurteilung von Dialogstrukturen bei unterschiedlich ausgerichteten oder situativ stark variierenden Zielsetzungen des Benutzers geeignet, wie sie bei komplexen, ganzheitlichen Tätigkeiten typisch sind. Es wird ausgeführt, daß Aufgabenanalyse deshalb nicht auf die Untersuchung oder Modellierung einzelner Aufgaben oder Abläufe beschränkt bleiben kann, sondern zu einer abstrahierten Darstellung möglichst aller potentiell entstehenden Ziele des Benutzers kommen muß. Eine solche Darstellung wird als *Aufgabenraum* bezeichnet. Diesem werden eine Reihe von Grundstrukturen von Dialogen gegenübergestellt, die sich mit graphischen Benutzungsschnittstellen realisieren lassen. Die Eignung verschiedener Grundstrukturen von Dialogen für verschiedene Dimensionen des Aufgabenraums wird diskutiert. Die Folgerungen für die Systemgestaltung werden im Hinblick auf ein integriertes Gestaltungsvorgehen dargestellt.

1. Aufgabenangemessenheit und Systemgestaltung

Die Forderung nach einer aufgabenangemessenen Gestaltung von Informationssystemen stellt eines der Kernprinzipien der Software-Ergonomie dar und hat Eingang in alle wesentlichen Richtlinien und Normen gefunden [3, 5, 12, 23]. So fordert z. B. ISO [12] unter dem Aspekt der Aufgabenangemessenheit die effektive und effiziente Unterstützung des Benutzers bei der Aufgabenerfüllung.

Die konkrete Umsetzung dieser Forderung beim Entwurf von interaktiven Informationssystemen ist allerdings nach wie vor eine Problemstellung, für die bestenfalls in Einzelbereichen Lösungen vorhanden sind. Insbesondere zeigt sich, daß Aufgabenangemessenheit auf den unterschiedlichen Abstraktionsebenen eines Systems mit unterschiedlichen Gestaltungsaspekten in Verbindung zu bringen ist (vgl. [27]). Ein integriertes Vorgehen zur Realisierung aufgabenorientierter Systeme auf unterschiedlichen Ebenen liegt nicht vor.

Ein häufig verwendeter Ansatzpunkt besteht darin, Aufgabenstrukturen und -abläufe, die bei einer Tätigkeit vorgefunden werden, im System mehr oder weniger starr abzubilden. Die Leitvorstellung dabei ist die einer Hierarchie von Zielen oder Funktionen, die durch eine Zerlegung übergeordneter Ziele erhalten wird. Die vorhandenen Analyse- und Modellierungsmethoden unterscheiden sich insbesondere nach dem jeweiligen Analyseziel: Psychologisch ausgerichtete

Methoden wie GOMS [2] oder CCT [17] versuchen, mentale Repräsentationen von Aufgaben abzubilden. Software-technische Verfahren wie Strukturierte Analyse (s. z. B. [26]) sind auf die Erfassung benutzerunabhängiger „logischer" Anforderungen an die Systemfunktionalität ausgerichtet. Die mangelnde Verbindung dieser beiden Ansätze wird von verschiedenen Autoren festgestellt [4, 8].

Ansätze zu einer verbesserten Integration dieser Ansätze finden sich zum einen hinsichtlich der Einbeziehung von Benutzungsaspekten in das Entwicklungsvorgehen (s. z. B. [10, 24]), zum anderen im Bezug auf eine stärkere methodische Integration [16 , 22] für eine funktional orientierte Sichtweise. Für die Gestaltung objektorientierter Benutzungsschnittstellen wurden in den letzten Jahren Ansätze zur Ableitung von Dialogstrukturen aus softwaretechnischen Daten- bzw. Objektmodellen entwickelt [1, 7].

Ein gemeinsames Merkmal und zugleich Problem der genannten Methoden ist, daß sie jeweils *einen einzelnen Aspekt* der zu unterstützenden Aufgabe in den Vordergrund stellen, z. B. entweder das funktionale Ziel oder das Objekt der Aufgabe. Entsprechend führen die Ansätze zu funktional orientierten oder zu objektorientierten Grundstrukturen der Benutzungsschnittstelle [28].

Wie Hartson & Boehm-Davis [9] feststellen, liefern Aufgabeanalysemethoden immer nur einen beschränkten Ausschnitt des realen Aufgabenspektrums. Als Folge werden nur bestimmte Aufgabenstellungen in der realen Nutzungssituation wirklich effektiv unterstützt, andere Aufgaben können gar nicht oder nur mit Zusatzaufwand erfüllt werden. Dies ist besonders bei Tätigkeiten zu erwarten, die sich durch ein umfangreiches Aufgabenspektrum und hohe Variabilität und Flexibilität auszeichnen, Aufgaben also, die heute sowohl aufgrund neuer Organisationsansätze wie auch arbeitswissenschaftlicher Gesichtspunkte häufig gefordert werden.

In dem vorliegenden Beitrag soll ausgeführt werden, daß die Aufgabenanalyse für die Unterstützung solcher komplexer und flexibler Tätigkeiten nicht auf die Untersuchung oder Modellierung einzelner Aufgaben oder Abläufe beschränkt bleiben kann, sondern zu einer abstrahierten Darstellung möglichst aller potentiell entstehenden Ziele des Benutzers kommen muß. Eine solche Darstellung soll hier als *Aufgabenraum* bezeichnet werden. Diesem Aufgabenraum werden eine Reihe von Grundstrukturen von Dialogen gegenübergestellt, die sich mit graphischen Benutzungsschnittstellen realisieren lassen. Die Eignung verschiedener Grundstrukturen von Dialogen für verschiedene Dimensionen des Aufgabenraums wird diskutiert. Zunächst werden hierfür die verschiedenen Gestaltungsbereiche eines interaktiven Systems im Hinblick auf ein integriertes Gestaltungsvorgehen dargestellt.

2. Auf dem Weg zu einer integrierten Entwicklungsmethodik

In einer integrierten Entwicklungsmethodik wird zum einen ein Vorgehensmodell benötigt, das durch Benutzerbeteiligung eine ausreichende Aufgaben- und Benutzerorientierung gewährleistet, wobei in der Analyse und Anforderungsdefinition neben den software-technischen Schritten der Aufgaben- und Datenmodellierung insbesondere Aufgabengestaltungsschritte vorzusehen sind [1]. Ein Gesamtmethodenmodell ist in Abb. 1 dargestellt.[1] In der Analyse entsteht das essentielle Aufgabenmodell des Systems, das noch unabhängig von den verwendeten Technologien ist, also z. B. noch keine Aussage darüber enthält, welche Aufgaben vom Benutzer und welche vom Computersystem durchgeführt werden [18]. Parallel wird ein grobes Objektmodell (Objektabgrenzung und Objektstruktur) entwickelt.

Für die *Anwendungsspezifikation* (d.h. Anforderungsdefinition für die Anwendung) wird das essentielle Aufgabenmodell verfeinert, indem die essentiellen Aufgaben in Teilaufgaben aufgeteilt werden. Die Verfeinerung von Aufgaben und Teilaufgaben ist beendet, wenn alle Teilaufgaben eindeutig dem Benutzer oder dem Computersystem zugeordnet werden können oder Dialogaufgaben sind, deren Verfeinerung erst während des Entwurfs der Dialogabläufe stattfindet. Das bisherige Objektmodell wird verfeinert, indem die Attribute vervollständigt werden. Aufgrund der erfolgten Mensch-Rechner-Funktionsteilung werden die Funktionen der Objekte ermittelt.

Als verbindendes Glied zwischen Analyse und Detail-Spezifikation ist ein konzeptioneller Entwurf erforderlich, der zum Ziel hat, ein zu entwerfendes Informationssystem aus der Sicht der Benutzung zu beschreiben. Das hierbei entstehende konzeptionelle Benutzungsschnittstellenmodell (*Benutzungsmodell*) besteht aus Objekten, Funktionen und Objektstrukturen, mit denen die funktionalen Elemente der Anwendung an der Benutzungsschnittstelle abgebildet werden. Diese benutzungsorientierte „Architektur" der Benutzungsschnittstelle repräsentiert bereits Entwurfsentscheidungen hinsichtlich der Systemstruktur, ist aber noch frei von Annahmen über Realisierungsmittel, also hinsichtlich Darstellungsarten und Interaktionstechniken. Sie bildet einen sinnvollen Ausgangspunkt für systematisches

---

[1] Diese Arbeit basiert auf Ergebnissen des Projektes "TASK - Technik der aufgaben- und benutzerangemessenen Software-Konstruktion", das vom Projektträger "Arbeit und Technik" des BMFT gefördert wurde (Förderkennzeichen 01 HK 849 A2).

Prototyping, das begleitend zum Entwurf als Explorations- und Validierungsmittel eingesetzt werden sollte (vgl. [6].

Hinsichtlich einer einfachen Realisierung von Benutzungsschnittstellen sind - sozusagen am unteren Ende eines Gesamtvorgehens - geeignete Spezifikations- und Generierungsmethoden erforderlich, die ausführbare Beschreibungen für existierende Benutzungsschnittstellenwerkzeuge erzeugen. Hierbei können die Einzelaspekte der Erzeugung statischer *Sichten* (d.h. Bildschirmdarstellungen von konzeptionellen Objekten) und der Definition der Dialogdynamik unterschieden werden, wofür bereits Methoden beschrieben wurden. (Siehe [25] zur Maskengenerierung, [13] zur Dialogspezifikation, [14, 15] zur Beschreibung des Gesamtzusammenhangs der Realisierungsmethode).

In den folgenden Abschnitten werden zunächst die für den konzeptionellen Benutzungsschnittstellenentwurf relevanten Aufgabendimensionen aufgezeigt. Danach werden Klassen von Dialogstrukturen in Benutzungsschnittstellen, speziell graphischen Benutzungsschnittstellen definiert und mögliche Einsatzfälle diskutiert.

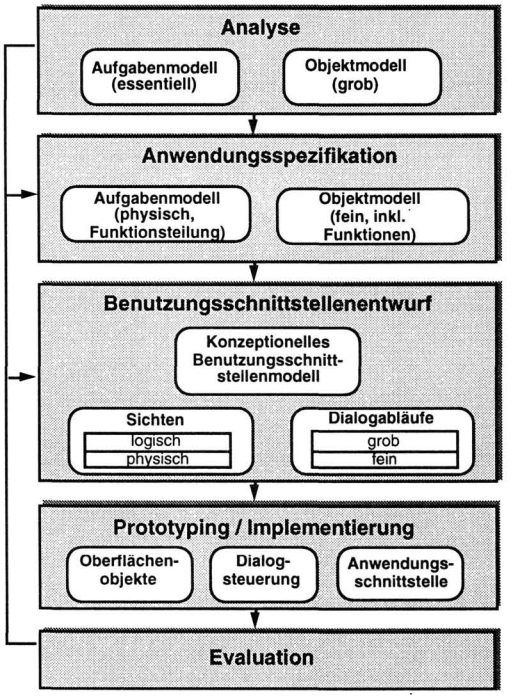

Abb. 1: Methodenmodell für die Entwicklung interaktiver Systeme

3. Das Konzept des Aufgabenraums als Basis für die Definition von Dialogstrukturen

Aufgabenanalysemethoden wie GOMS liefern an den funktionalen Zielen des Benutzers orientierte Beschreibungen. Diese Ziele sind zwar hierarchisch zerlegbar, in sich aber in der Regel nicht weiter strukturiert. Der Ansatz der Task-Action Grammar [19] bietet demgegenüber die Möglichkeit, Aufgabenparameter für eine weitergehende Abstraktion von den Einzelaufgaben zu nutzen und kann Aufgaben durch eine Kombination von Parametern beschreiben.

In den integrierten softwaretechnischen Vorgehensweisen werden mehrere Perspektiven auf das System nebeneinander modelliert, wobei die unterschiedlichen Modelle durch Regeln oder Werkzeuge konsistent gehalten werden müssen. Typisch ist eine Daten-, Funktions- und Dynamiksicht (Verhalten) auf das zu entwickelnde System, die allerdings getrennt voneinander vorliegen. Die Zusammenführung dieser unterschiedlichen Sichten an der Benutzungsschnittstelle stellt ein ungelöstes Problem dar. Dies gilt im wesentlichen auch für objektorientierte Analyseverfahren (vgl. [21]), wobei diese allerdings eine stärkere Integration des Objekt- und Funktionsaspekts liefern.

Es soll deshalb nun näher untersucht werden, durch welche unterschiedlichen Grundkomponenten Aufgaben definiert werden können. Diese bilden jeweils den Hauptausgangspunkt für die Zielsetzung des Benutzers. Bereits angesprochen wurde die mit einer Aufgabe verbundene *Funktion* im Sinne eines funktionalen Ziels. Eine typische Zielsetzung des Benutzers wäre z. B.: 'Ich möchte einen Brief schreiben'.

Eine andere Sichtweise ist der Ausgangspunkt vom Gegenstand bzw. *Objekt* der Aufgabe: 'Ich möchte die Akte der Firma XY sehen, um evtl. weitere Kontakte aufzunehmen'. Hier bildet ein Objekt die Invariante der Aufgabe, während das funktionale Ziel nicht oder nur unscharf definiert ist. Dies ist die typische Form bei objektorientierten Benutzungsschnittstellen, bei der Interaktionen mit der Identifikation eines Objekts beginnen.

Bei vielen Aufgaben bildet nicht die Identität, sondern der *Zustand* von Objekten den Ausgangspunkt für die Bearbeitung. Eine typische Fragestellung wäre: 'Ich will alle Rechnungen im Zustand ‹überfällig› bearbeiten' oder 'Liste alle Kunden im Raum Stuttgart auf'. Diese Konzepte werden seit langem bei Datenbankanwendungen verwendet, allerdings ist zu beachten, daß die interessierenden Zustände sehr variabel sein können, so daß ein flexibilisierter Zugriff in vielen Fällen erforderlich ist. Zustandsorientierte Zugriffe sind hilfreich, um den Ablauf

von Aufgaben zu unterstützen, da alle Objekte eines gleichen Zustands gleichartige Folgebearbeitungen erlauben können.

Weitere aufgabenbestimmende Merkmale können die *Zeit* ('Ich will alle Vorgänge sehen, die heute zu bearbeiten sind') oder der *Ort* sein. Ortsbezogene Aufgaben treten insbesondere bei räumlich verteilten Systemen auf und spielen in der herkömmlichen DV keine Rolle. Bei einem integrierten elektronischen Haussystem etwa können aber durchaus Fragen auftreten wie 'Ich will den Zustand aller Geräte in der Küche überprüfen'.

Schließlich ist zu beachten, daß es beliebige, komplexe Zusammenfassungen von Objekten zu *Vorgängen* gibt, die häufig über längere Zeiträume hinweg bearbeitet werden und vorgangsorientierte Zielsetzungen des Benutzers auslösen können ('alle Objekte, die mit dem Lebensversicherungsantrag von X zusammenhängen'). Abbildungen solcher Konzepte finden sich z. B. in Workflow-Management-Systemen.

Führt man diese unterschiedlichen Ausgangspunkte für Aufgaben zusammen, so lassen sich Aufgaben als eine Kombination der Grunddimensionen

<Funktion, Objekt, Zustand, Zeit, Ort>

auffassen. Diese Dimensionen spannen folglich einen *Aufgabenraum* auf, der die einzelnen Aufgabeninstanzen beinhaltet. In jeder konkreten Aufgabeninstanz sind die Dimensionen mit spezifischen Ausprägungen belegt, wobei Dimensionen auch unbelegt bleiben können. Eine Aufgabe ‹Dokument Y löschen› z. B. wird durch eine Kombination aus einer Funktions- und einer Objektkomponente gebildet. Der Benutzer wird nun situationsabhängig die einzelnen Komponenten mit einer unterschiedlichen Gewichtung belegen, so wird im Zusammenhang mit dem Löschen alter Dokumentversionen die Funktion das größere Gewicht besitzen. Ein Funktionsmodus 'Löschen' wäre hier eine effektive Unterstützung, wobei natürlich sorgfältig gegenüber anderen Kriterien wie Konsistenz oder Erlernbarkeit abgewogen werden muß.

Die zielbestimmenden Aufgabendimensionen können also vom Benutzer situativ nicht nur mit verschiedenen Werten sondern auch mit unterschiedlichen Gewichten belegt werden. Entsprechend der Dimension, die jeweils das stärkste Gewicht hat, sind verschiedene Dialogstrukturen der Benutzungsschnittstelle geeignet, um das Aufgabenziel am effektivsten erreichen zu können. Solche Dialogstrukturen werden im nächsten Abschnitt beschrieben. Neben der variierenden Gewichtung der Dimensionen müssen für den Entwurf noch weitere, aus dem Arbeitskontext abzuleitende Merkmale berücksichtigt werden. Hierzu zählt insbesondere die er-

wartete Häufigkeit einer bestimmten Zielausprägung. Für selten auftretende Ausprägungen wird man häufig spezifische Unterstützungsfunktionen zugunsten einer geringeren Systemkomplexität weglassen.

Das Konzept des Aufgabenraumes erlaubt es, potentielle Zielsetzungen des Benutzers in systematischer Weise zu synthetisieren und nicht nur existierende Konzepte oder Abläufe zu analysieren und ggf. zu variieren. Somit kann eine beliebige Vielfalt an Szenarien generiert werden, für die jeweils die Eignung des zu entwerfenden Systems überprüft werden kann.

4. Dialogstrukturtypen bei graphischen Benutzungsschnittstellen

Im folgenden wird gezeigt, wie aus dem Aufgabenmodell in systematischer Weise angemessene Dialogstrukturen abgeleitet werden können. Das Benutzungsmodell (vgl. Abschnitt 2) zeigt durch die Definition von Zugriffspfaden die wesentlichen Navigationsmöglichkeiten des Benutzers im System im Sinne der Erreichbarkeit auf, ohne die Dynamik der Dialogabläufe im einzelnen festzulegen. Die möglichen Navigationspfade zu einer zu bearbeitenden Sicht, z. B. einem Formular zur Bearbeitung der Attribute eines Objektes, können hinsichtlich der unterschiedlichen Arten von „Einstiegspunkten" differenziert werden. Dabei können bei graphischen Benutzungsschnittstellen analog zu den im vorigen genannten Aufgabendimensionen folgende wesentliche Fälle auftreten:

- *Objektorientierte Navigation* beginnt mit der Auswahl einer bestimmten Objektklasse und identifiziert dann in einem oder mehreren Eingrenzungsschritten das interessierende Einzelobjekt oder die Objektmenge. Im Einstiegsfenster werden dabei die von der Aufgabenstellung her primär relevanten Objekttypen als Sammelobjekte angeboten. Erst nach der Identifikation der Einzelobjekte oder Objektmenge wird die beabsichtigte Bearbeitungsfunktion im Kontext einer Objektsicht ausgelöst. Weitere Dialogpfade ergeben sich, sofern aufgabenabhängig benötigt, entlang der Relationen im Objektmodell.
- Ein *zustandsorientierter Zugriff* ist dadurch gekennzeichnet, daß eine Sicht alle Objekte eines Typs repräsentiert, die bestimmte Merkmale erfüllen bzw. sich in einem bestimmten Bearbeitungszustand befinden. Diese Möglichkeit wird bei gegenwärtigen graphischen Benutzungsschnittstellen noch selten eingesetzt; sie bietet aber die Möglichkeit für die Realisierung aufgabengerechter Abläufe.
- Bei *funktionsorientierter Navigation* wird im ersten Schritt eine Bearbeitungsfunktion bestimmt, ohne die Objektmenge oder das Objekt vorher näher zu

spezifizieren. Hier bildet jeweils eine bestimmte Aufgabe den Einstiegspunkt für einen Dialog.

- Eine *vorgangsorientierte Navigation* verwendet als Startpunkt ein Strukturierungsobjekt, das alle Objekte beinhaltet, die für einen bestimmten Geschäftsprozeß verwendet werden. Dabei können alle drei bereits aufgeführten Zugriffsarten innerhalb dieses Vorgangsobjektes für den weiteren Bearbeitungsweg eingesetzt werden. Wird ein Vorgang selbst wieder als Objekt gesehen, ergibt sich hier eine Spezialform der objektorientierten Navigation.

Für graphische Benutzungsschnittstellen ist die *objektorientierte Dialogstrukturierung* als Grundmuster typisch, die dem Prinzip der Direkten Manipulation [20]) entspricht. Diese Form eignet sich besonders für benutzergesteuerte Aufgabenstellungen, z. B. das gezielte Bearbeiten einer bestehenden Adresse eines Versicherten (Abb. 2), und gewährleistet eine hohe Flexibilität, was die Aufgabenabläufe betrifft. Der einzelne Datensatz eines Versicherten wird dabei über einen Selektionsdialog (z. B. „Versicherten Öffnen") aufgerufen (Abb. 3) und erst sekundär die Bearbeitungsfunktionen aufgerufen.

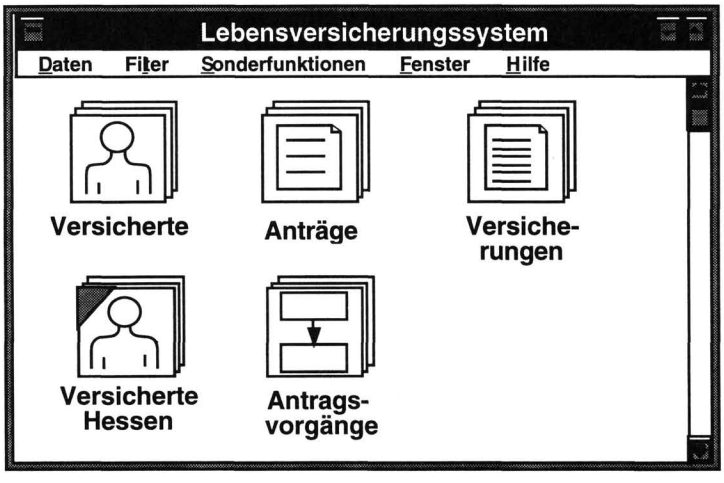

Abb. 2: Beispiel für ein Einstiegsfenster

Für eine zustands- oder merkmalsabhängige Strukturierung des Objektraumes eignen sich sogenannte *Filterobjekte*, die - fest eingestellt oder benutzerdefiniert - eine Gesamtobjektmenge nach einem bestimmten Kriterium einschränken. Ein Beispiel in Abb. 2 ist das Objekt „Versicherte Hessen", das z. B. zuständigkeitsabhängig definiert sein kann, und die Suche in der Gesamtmenge der Versicherten (über einen Suchdialog, siehe Abb. 3) erleichtert.

Im Gegensatz zu der freien Navigation im Objektraum sind für manche Aufgabenstellungen stärker geführte, *funktionale Dialoge* wünschenswert, etwa wenn formalisierte Abläufe vorliegen (vgl. Greutmann [7]). Ein Beispiel bei Finanzdienstleistern ist die Durchführung des Jahresabschlusses. Dieser sollte direkt als Funktionsaufruf (z. B. im Menü des Hauptfensters) ausgeführt werden und ggf. mit Parametern versorgt werden können.

Vorgangsobjekte können dazu dienen, bei komplexeren Aufgaben die Überwachung und Prüfung von Konsistenzbedingungen gegenüber der rein objektorientierten Navigation zu vereinfachen. Beispielsweise werden bei der Beantragung einer Lebensversicherung zunächst die Daten für den Versicherten und die Antragsdaten erfaßt und anschließend eine Gesundheitsprüfung für die zu

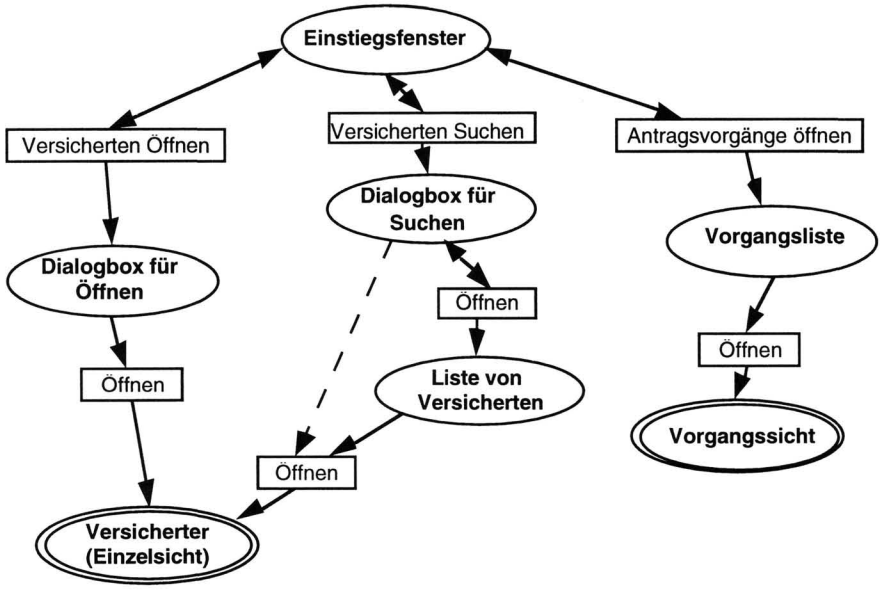

Abb. 3: Grundmuster für objektorientierte Dialogabläufe unter Einschluß von Vorgangsobjekten (Dialognetz nach [13])

versichernde Person angefordert, bevor schließlich der Vertrag ausgestellt wird. Die Überwachung offener Vorgänge kann in einem entsprechenden Vorgangsobjekt erfolgen, das über eine Liste zugänglich ist. Beispielsweise kann für „Antragsvorgänge“ in Abb. 2 eine solche Vorgangsliste und aus der Liste heraus die einzelnen Vorgänge geöffnet werden (Abb. 3). Die Teilobjekte für Antragsvorgänge (z. B. „Versicherter“, „Vertrag“) sind dann über die Vorgangssicht aufrufbar (der Unterdialog kann aus Platzgründen nicht gezeigt werden).

5. Einsatzerfahrungen und Schlußfolgerungen

Das hier dargestellte Konzept eignet sich für den Entwurf interaktiver Informationssysteme, mit denen vorwiegend Verwaltungsaufgaben bearbeitet werden, beispielsweise im Versicherungs- oder Produktionsplanungsbereich. Bei solchen betrieblichen Informationssystemen sind heute im Gegensatz etwa zu PC-Standardsoftware häufig noch funktionsorientierte Dialogstrukturen mit entsprechenden Einschränkungen der Flexibilität anzutreffen. Beim Übergang auf graphische Benutzungsschnittstellen werden nun auch zunehmend objektorientierte Strukturen eingesetzt, obwohl dieser Übergang für viele Entwickler noch nicht selbstverständlich ist. Die objektorientierte Strukturierung von Dialogen in graphischen Benutzungsschnittstellen hat sich als leicht erlernbare und flexible Grundstruktur in Anwendungsprojekten bewährt. Die Evaluation eines Prototypen für eine Anwendung im Versicherungsbereich zeigte, daß sich die Benutzer im System sofort zurechtfanden, und Testaufgaben lösen konnten.

Analog zu der Kritik von Hutchins, Hollan und Norman [11] an Direkter Manipulation jedoch wird über die rein objektorientierte Navigation hinaus für manche Aufgaben eine stärkere Unterstützung bis hin zur Automatisierung von Aufgaben benötigt. Beispiele hierfür sind der genannte Jahresabschluß und die Vorgangssteuerung für den Abschluß einer Lebensversicherung. Werden parallel funktionsorientierte und objektorientiert Einstiege vorgesehen, kann in manchen Fällen eine größere Handlungsflexibilität erzielt werden, etwa wenn sich der Benutzer eines PPS-Systems entscheiden kann, ob er eine Auftragsbestätigung sofort ausdrucken möchte (objektorientierte Struktur) oder ob er den Sammeldruck sämtlicher bearbeiteter Aufträge (funktionsorientierte Struktur) zu einem bestimmten Zeitpunkt vorzieht, z. B. weil der Drucker gerade mit anderen Formularen belegt ist. Das Beispiel zeigt, daß es bei der Dialogstrukturierung - und damit bei der Gestaltung des Aufgabenraums - keinen "one-best-way" gibt, sondern unter Umständen mehrere Dialogwege parallel anzubieten sind.

Insgesamt trägt der vorliegende Beitrag zum Schließen der Lücke zwischen Software-Ergonomie und Software-Technik bei, indem der Entwurfsraum für die Dia-

loggestaltung durch die Charakterisierung der wesentlichen Dialogtypen in Abhängigkeit von den Aufgabendimensionen strukturiert wird. Weiterführende anstehende Arbeiten erstrecken sich auf die vollständigere Erprobung des Entwurfskonzeptes in konkreten Projekten. Ferner ist die Übertragung des Ansatzes auf Bereiche außerhalb der betrieblichen Informationssysteme geplant, etwa auf den Bereich von Steuerungssystemen im häuslichen Bereich.

Literatur

[1] Beck, A.; Janssen, Ch. (1993): Vorgehen und Methoden für aufgaben- und benutzerangemessene Gestaltung von graphischen Benutzungsschnittstellen. In: Coy, W. u.a.: Menschengerechte Software als Wettbewerbsfaktor. Forschungsansätze und Anwenderergebnisse aus dem Programm "Arbeit und Technik", Stuttgart: Teubner, 200-221.

[2] Card, S.K.; Moran, T.P. & Newell, A. (1983): The Psychology of Human-Computer Interaction. Hillsdale, NJ: Lawrence Erlbaum.

[3] DIN (1988): DIN 66234, Teil 8: Bildschirmarbeitsplätze - Grundsätze ergonomischer Dialoggestaltung. Berlin: Beuth Verlag.

[4] Eberleh, E., Arend, U., Kasten, C., Strothotte, T., Ziegler, J. (1991): Integration software-ergonomischer Forschungsergebnisse in die betriebliche Software-Entwicklung. Diskussionsunterlagen. In: Ackermann, D., Ulich, E. (Hrsg.) Software-Ergonomie '91. Stuttgart: Teubner, 141-151.

[5] EG (1990): Richtlinie des Rates vom 29. Mai 1990 über die Mindestvorschriften bezüglich der Sicherheit und des Gesundheitsschutzes bei der Arbeit an Bildschirmgeräten (90/270/EWG). Amtsblatt der Europäischen Gemeinschaften, Nr. L 156/14.

[6] Floyd, C. (1984). A systematic look at prototyping. In: Budde, R.; Kulenkamp, K.; Mathiassen, L.; Züllighoven, H. (Eds.). Approaches to Prototyping. Berlin, Heidelberg: Springer.

[7] Greutmann, Th. (1993): Datenmodellierung und aufgabengerechte Dialoge: ein Synchronisierungsproblem. In: Rödiger, K.-H. (Hrsg.): Software-Ergonomie '93. Stuttgart: Teubner, 99-110..

[8] Hartson, H.R., Hix, D. (1989): Toward empirically derived methodologies and tools for human-computer interface development. International Journal of Man-Machine Studies 31, 477-494.

[9] Hartson, Rex H.; Boehm-Davis, Deborah (1993): User-Interface development processes and methodologies. Behaviour &Information Technology 12 (2), 98-114.

[10] Hoyos, C. Graf; Holz a.d.Heide, B; Ortlieb, S. (1993): Eine iterative Software-Entwicklungsstrategie mit gezielter Benutzerbeteiligung und systematischer Evaluation der Benutzungsfreundlichkeit. In: Frese, M., Kasten, Chr., Skarpelis, C., Zang-Scheucher, B. (Hrsg.). Software für die Arbeit von morgen. Berlin: Springer, 497-525.

[11] Hutchins, E.L., Hollan, J.D. und Norman, D.A. (1986): Direct Manipulation Interfaces. Norman, D.A. und Draper, S. (Hrsg.) User Centered System Design - New Perspectives on Human-Computer Interaction, Hillsdale, London: Lawrence Erlbaum, 87-124.

[12] ISO (1990): ISO Standard 9241: Ergonomic Requirements for Office Work with Visual Display Terminals. Part 10: Dialogue Principles. Draft International Standard. International Standards Organisation.

[13] Janssen, C. (1993): Dialognetze zur Beschreibung von Dialogabläufen in graphisch-interaktiven Systemen. In: Rödiger, K.-H. (Hrsg.): Software-Ergonomie '93. Stuttgart: Teubner, 67-76.

[14] Janssen, C., Weisbecker, A., Ziegler, J. (1993a): Generating User Interfaces from Data Models and Dialogue Net Specifications. INTERCHI'93 Proceedings. New York: ACM, 418-423.

[15] Janssen, Ch.; Weisbecker, A.; Ziegler, J. (1993b): Generierung graphischer Benutzungsschnittstellen aus Datenmodellen und Dialognetz-Spezifikationen. In: Züllighoven, H.; Altmann, W.; Doberkat, E.-E. (Hrsg.) Requirements Engineering '93: Prototyping, Stuttgart: Teubner, 335-347.

[16] Johnson, P.; Drake, K.; Wilson, S. (1990): A Framework for Integrating UIMS and User Task Models in the Design of User Interfaces. in: Duce, D.A.; Gomes, M.R.; Hopgood, F.R.A.; Lee, J.R. (ed.) User Interface Management and Design, Berlin, Heidelberg, New York: Springer, 203-216.

[17] Kieras, D. & Polson, P. (1985): An approach to the formal analysis of user complexity. Int. J. Man-Machine Studies, 22, 365-394.

[18] McMenamin, S.M.; Palmer, J.F. (1988): Strukturierte Systemanalyse. München, Wien: Carl Hanser Verlag.

[19] Payne, S.J. ; Green, T.R.G. (1986): Task-Action grammars: A model for the mental representation of task languages. Human-Computer Interaction, 2, 93-133.

[20] Shneiderman, B. (1983): Direct Manipulation - A Step Beyond Programming Languages. IEEE Computer 16 (8), 57-69.

[21] Stein, W. (1994): Objektorientierte Analysemethoden - Vergleich, Bewertung, Auswahl. Mannheim u.a.: BI-Wissenschaftsverlag.

[22] Sutcliffe, A. G.; McDermott, M. (1991): Integrating methods of human-computer interface design with structured systems develepment. in: Int. J. Man - Machine Studies, Vol. 34, 631-655.

[23] VDI (1988): Software-Ergonomie in der Bürokommunikation. Verein Deutscher Ingenieure, Richtlinie 5005. Berlin: Beuth Verlag.

[24] Wassermann, Antony I.; Pircher, Peter A.; Shewmake, David T.; Kersten, Martin L. (1986): Developing Interactive Information System with the User Software Engineering Methodology. IEEE Transactions on Software Engineering12 (2) Feb. 1986, 326-345.

[25] Weisbecker, A. (1993): Integration von software-ergonomischem Wissen in die Systementwicklung. In: Rödiger, K.-H. (Hrsg.): Software-Ergonomie '93. Stuttgart: Teubner, 299-310.

[26] Yourdon, E. (1989): Modern Structured Analysis. Englewood Cliffs: Prentice Hall.

[27] Ziegler, J. (1988): Software-Ergonomie in der Bürokommunikation. VDI-Berichte 716, Düsseldorf; VDI-Verlag, S. 141-159.

[28] Ziegler, J. (1993): Entwurf graphischer Benutzungsschnittstellen. In: Ziegler, J.; Ilg, R. (Hrsg.): Benutzergerechte Software-Gestaltung - Standards, Methoden und Werkzeuge. München, Wien: Oldenbourg.

Jürgen Ziegler, Christian Janssen

Fraunhofer-Institut für Arbeitswirtschaft und Organisation (IAO)
Nobelstrasse 12c, D-70569 Stuttgart
Tel.: +49 711 970-2334, +49 711 970-2330, Fax: +49 711 970 2300
Email: Juergen.Ziegler@iao.fhg.de, Christian.Janssen@iao.fhg.de