**Tutorial im Rahmen der Fachtagung Software-Ergonomie '97 in Dresden 3.-6. März 1997**

# Kontextbezogener Entwurf von Benutzungsoberflächen mit MUSE II

Peter Gorny, Universität Oldenburg

## 1. Vorbemerkung

Ein Tutorial zum Entwurf von Benutzungsoberflächen nach dem Stand der Software-Ergonomie (s. Kasten 1) muß sich heute auf die ISO-Norm (ISO 9241 und deren europäische Umsetzung EN-ISO 9241 [DIN 29241]) und die Richtlinie des Europäischen Rates (90/270/EWG - s. Kasten 2) und deren Umsetzung in die Bildschirmarbeitsverordnung des Bundes stützen. Hier werden die Gebrauchstauglichkeit und einige Kriterien für die Oberflächen-Gestaltung festgelegt, obwohl es erst bruchstückhaft Meßmethoden gibt, mit denen man (fertige) Oberflächen bewerten kann, und nur rudimentäre Erkenntnisse, wie die Kriterien bei der Konstruktion von (neuen) Oberflächen zu berücksichtigen sind. Die Anwendung software-ergonomischer Methoden auf Systeme zur Unterstützung kooperativer Arbeit macht die Designentscheidungen noch anspruchsvoller, weil für die Werkzeugunterstützung von Kooperation kaum empirische Forschungsergebnisse vorliegen: die Charakteristika der Aufgaben und die erforderlichen Kenntnisse und Fähigkeiten der Benutzer werden von den Arbeitswissenschaften erst gerade kategorisiert.

Die Software-Ergonomie will die Aufmerksamkeit der Entwickler, Benutzer und Auftraggeber von Software-Systemen auf deren Gebrauchstauglichkeit[1] lenken. Es ist offenbar, daß man Gebrauchstauglichkeit nur messen (oder erzeugen) kann, wenn man den ***Kontext*** betrachtet, in dem ein Programm als Werkzeug benutzt werden soll. Konkret bedeutet das die Berücksichtigung der Aufgabenstellungen, der Eigenschaften der Benutzer und einiger Merkmale der Werkzeugbenutzung. Vor einer Gestaltung oder Bewertung einer Benutzungsoberfläche nach den Gütekriterien der Software-Ergonomie muß man also eine Arbeitsorganisations- und Aufgabenanalyse vornehmen und die Benutzereigenschaften feststellen.

Kasten 1

**Software-Ergonomie bezieht sich auf**
- aufgabengesteuerte, zielorientierte Arbeitssituationen
- Anwendungsbereiche:
  - Büro / Verwaltung
  - Prozeßsteuerung (Leitstände)
  - Konstruktiosnbüro

**Mensch-Computer Interaktion** (engl. HCI) umfaßt zusätzlich Nicht-Arbeitssituationen wie
- "information kiosks", "points of information"
- Informationstechnologie für Heim und Freizeit
- Lernsoftware,

die zugehörige Grundlagenforschung und die Entwicklung der erforderlichen Techniken.

Kasten 2

Aus der **EU-Bildschirmrichtlinie 90/270/EWG:**
...
3. Bei Konzipierung, Auswahl, Erwerb und Änderung von Software sowie bei der Gestaltung von Tätigkeiten, bei denen Bildschirmgeräte zum Einsatz kommen, hat der Arbeitgeber folgenden Faktoren Rechnung zu tragen:

a) Die Software muß der auszuführenden Tätigkeit angepaßt sein.
b) Die Software muß benutzerfreundlich sein und gegebenenfalls dem Kenntnis- und Erfahrungsstand des Benutzrs angepaßt werden können; ohne Wissen des Arbeitnehmers darf keinerlei Vorrichtung zur quantitativen oder qualitativen Kontrolle verwendet werden.

...

c) Die Grundsätze der Ergonomie sind insbesondere auf die Verarbeitung von Informationen durch den Menschen anzuwenden.

## 2. Gütekriterien der Software-Ergonomie und des Software-Engineering

### 2.1 Gebrauchstauglichkeit

Die Europäische Richtlinie 90/270/EWG über die "Mindestvorschriften bezüglich der Sicherheit und des Gesundheitsschutzes bei der Arbeit an Bildschirmgeräten" ist inzwischen auch in Deutschland rechtlich bindend: Ein entsprechendes deutsches Arbeitsschutzgesetz und die

Kasten 3

**ISO 9241 "Ergonomic requirements for office work with visual displays (VDTs)"**

**Part 11 Guidance on specifying and measuring usability (Draft)**

...

**3. Definitions**

3.1 **usability**: The effectiveness, efficiency and satisfaction with which specified users achieve goals in particular environments.

3.1.1 **effectiveness**: The accuracy and completeness with which users achieve specified goals.

3.1.2 **effiency**: The accuracy and completeness of goals achieved in relation to resources expended.

.....

**4. Specifying and measuring usability**

4.1 **Framework** for specifying usability

...

4.1.2 **Information needed**

When specifying or measuring the usability of an overall system, the following information is needed:

- A description of the relevant characteristics of the context of use including users, equipment, environments and users' task goals. ...
- A description of the usability measures consisting of target or actual values of effectiveness, efficiency, and satisfaction for the required contexts.

....

Bildschirmarbeitsverordnung wurden 1996 verabschiedet; die darauf bauenden Richtlinien der Berufsgenossenschaften sind noch in Arbeit. Deshalb nimmt das Tutorial hauptsächlich Bezug auf die EN/ISO-Norm 9241 "Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten". Diese ISO-Norm hat die alte DIN 66234 zum Vorbild. Von den für die Softwaregestaltung wesentlichen Teilen sind erst zwei relativ weit entwickelt: Teil 2 "Anforderungen an die Arbeitsaufgaben - Leitsätze" ist als ISO-Norm verabschiedet und Teil 10 "Grundsätze der Dialoggestaltung" liegt auch als Europäische Norm vor. Die Entwürfe für die übrigen Teile mit Software-Ergonomie-Bezug dürften noch einigen Änderungen in den internationalen Standardiserungsgremien unterworfen werden.

In den Teilen 2 und 11 wird der Zusammenhang hergestellt zwischen der Gebrauchstauglichkeit (usability) des Gesamtsystems und den Benutzern, den Zielen der Aufgaben einschließlich der Aufgabencharakteristika, den zur Verfügung stehenden Werkzeugen und der Arbeitsumgebung. Außerdem wird der Begriff "Gebrauchstauglichkeit" aufgebrochen in die Teilkriterien "Effektivität", "Effizienz" und "Zufriedenheit" (Kasten 3).

Teil 10 detailliert die Grundsätze für die Beurteilung von Dialogen:

- Aufgabenangemessenheit
- Selbstbeschreibungsfähigkeit
- Steuerbarkeit
- Erwartungskonformität
- Fehlertoleranz
- Individualierbarkeit
- Lernförderlichkeit

Im Tutorial sollen diese Grundsätze nicht nur für die Bewertung fertiger Benutzungsoberflächen, sondern für das Design neuer Systeme und ihrer Oberflächen genutzt werden. Dabei müssen sich die Designer bewußt sein, daß mit der Beachtung der Kriterien nicht gewährleistet ist, daß die Oberflächen ästhetisch ansprechend ("schön") oder originell sind. Hier bleibt Intuition und Kreativität gefordert.

## 2.2 Regeln für die Gestaltung von Benutzungsoberflächen

Neben diesen sehr allgemeinen Grundsätzen zur Beurteilung der Gebrauchstauglichkeit gibt es eine Vielzahl von einzelnen "Regeln", die aufgrund von qualitativen oder quantitativen Untersuchungen von Kognitions- oder Arbeitspsychologen aufgestellt wurden. Die größte Sammlung dazu ist von Smith und Mosier[2] (HyperSAM[3]) aufgestellt worden, allerdings ohne direkte Bezüge zum jeweiligen Anwendungskontext herzustellen.

### 2.3 Style Guides

Viele Softwarehäuser oder Hersteller haben ihren Mitarbeitern Richtlinien für die Gestaltung von Benutzungsoberflächen an die Hand gegeben. Die bekanntesten sind der IBM-CUA und die Style Guides von Apple (für den Macintosh), Microsoft (für Windows und NT) und von der OSF (für Motif). In Deutschland haben die umfangreichsten Style Guides die Firmen SNI und SAP entwickelt.

Die Style Guides sollen hauptsächlich ein spezifisches "look and feel" für einen Benutzungsoberflächen-Typ oder für die Software-Produkte eines Unternehmens (corporate image) sicherstellen. Damit ist das ISO-Kriterium "Erwartungskonformität" wenigstens teilweise berücksichtigt. Andere ergonomische Aspekte wie die Merkmale der Benutzer, der Aufgaben oder der Werkzeugbenutzung spielen kaum eine Rolle. Ebenso geben Style Guides keine Hinweise zu ästhetischen Eigenschaften der Oberfläche, auch wenn sie einige "Gesetze" der Gestaltpsychologie aufnehmen.

## 3. Systematisches Vorgehen beim Design von Benutzungsoberflächen

### 3.1 Einordnung des Oberfächen-Designs in den Software-Entwicklungsprozess

Wir gehen davon aus, daß die Gestaltung der Benutzungsoberfläche prinzipiell in eine der frühen Software-Entwicklungsphasen gehört, bei denen die Anforderungen an das Gesamtsystem zusammengestellt werden. Der hier verwendete Begriff der Phase soll nicht implizieren, daß das das klassische Phasenmodell des Software-Engineering bevorzugt wird. Im Gegenteil will ich betonen, daß die hier hilfreichsten Vorgehensmodelle den Benutzer frühzeitig beteiligen und das Prototyping erfordern, z.B. die evolutionären Vorgehensmodelle. Ein *Rapid* Prototyping von Oberflächen eignet sich dazu nicht, weil hierbei die Gefahr besteht, daß bloß schnell einige Masken herunterprogrammiert und die Ergebnisse sorgfältiger Analysen des Anwendungskotexts nicht berücksichtigt werden. Stattdessen ist ein *Slow and Principled* Prototyping zu fordern[4]. Die Zuordnung des Oberflächendesigns zur Phase der Anforderungsdefinition würde bedeuten, daß damit der Prototyp der Oberfläche Bestandteil dieser Definition würde, die den darauffolgenden Entwicklungsprozeß bstimmt.

Damit wird nicht ausgeschlossen, daß auch in späteren Phasen oder Iterationen der Software-Entwicklung noch Entscheidungen zur Oberfläche getroffen werden können

und müssen; es wird jedoch klargestellt, daß diese Entscheidungen an den Kriterien der ISO gemessen -- also konkret auf die Merkmale der Benutzer, der Aufgaben und der Werkzeugbenutzung bezogen werden müssen.

## 3.2 MUSE II - Eine Methode zum Entwurf von Benutzungsoberflächen

Um das systematische Design von Benutzungsoberflächen zu ermöglichen, wurde in Oldenburg ein Vorgehensmodell entwickelt, das im Folgenden in seiner zweiten Fassung skizziert werden soll. Wie immer ist auch mit diesem Vorgehensmodell beabsichtigt, die Entscheidungskomplexität zu reduzieren und Einzelkriterien zu geeigneten Augenblicken während des Designprozesses bewußt zu machen. Als Gütemaß werden dabei die ISO-Dialogprinzipien herangezogen, allerdings jetzt *als Maß für den Aufwand* , der bei der Entwicklung der Software zu der Erfüllung eines der Prinzipien betrieben werden muß.

Kasten 4

**Ziele von MUSE II**

- Entscheidungskomplexität reduzieren
- Ergonomische Prinzipien systematisch und dem Kontext entsprechend gewichtet anwenden

Kasten 5

**KABA** (Konstrastive Aufgabenanalyse)

| | |
|---|---|
| Zielgerichtetheit | Entscheidungsspielraum<br>Zeitspielraum<br>Strukturiertheit<br>Belastungen |
| Gegenständlichkeit | Körperliche Aktivität<br>Kontakt zu matriellen und sozialen Bedingungen des Arbeitshandelns |
| Soziale Eingebundenheit | Kooperation und unmittelbare zwischenmenschliche Kommunikation |

Kasten 6

**Beispiel:**
**Entwicklungsaufwand für die ISO-Prinzipien "Erwartungskonformität" und "Erlernbarkeit"**

Wenn z.B. ein "fachlich kompetenter" Benutzer eine Software-Funktion "häufig" benutzt, so muß ein relativ hoher Aufwand betrieben werden, um das Kriterium "Erwartungskonformität" zu erfüllen, während für die "Erlernbarkeit" ein geringerer Aufwand erforderlich ist. Dagegen müßte an Arbeitsplätzen mit häufig wechselndem "ungeschultem" Personal die "Erlernbarkeit" sehr viel höher gewichtet werden.

Der methodische Hintergrund stammt aus den Arbeitswissenschaften und stützt sich auf die Handlungstheorie (auch Handlungsregulationstheorie) und das Verfahren KABA (Konstrastive Aufgabenanalyse)[5]. In KABA werden die Grundmerkmale menschlichen Handelns "Zielgerichtheit", Gegenständlichkeit" und Soziale Eingebundenheit", die bei einer Aufgabengestaltung zu beachten sind, in Verbindung gesetzt zu den Humankriterien der Arbeit.

MUSE II (Method for User Interface Engineering) gliedert die Design-Entscheidungen zur Erfüllung der Prinzipien der ISO 9241 bei der Software-Gestaltung in drei Sichten, die beliebig oft eingenommen werden können.

Für jede der drei im Folgenden beschriebenen Sichten, die die Designer bei Anwendung von MUSE einnehmen, werden fünf Designschritte empfohlen (Merkwort: AWARE):

- Anforderungen definieren
- Wahlmöglichkeiten benennen
- Argumente zusammenstellen
- Richtlinien anwenden
- Entscheidungen notieren.

### *1. Zweckgebundene Sicht*

Bei der Zweckgebundenen Sicht kommt es darauf an, daß die Analyseergebnisse des Arbeitssystems umgesetzt werden in Designentscheidungen über die erforderlichen Werkzeugfunktionen, die dabei in vier Kategorien (Anwendungs-, Steuer-, Adaptier- und Metafunktionen) geordnet werden. Die Entscheidungen werden sich dabei einerseits auf die Erfahrungen mit ähnlichen Systemen (appli-

Kasten 7

**Werkzeug/Material-Leitbild zur Gestaltung der Benutzungsoberfläche eines Anwendungssystems**

In der Realität werden Aufgaben von Menschen erledigt, indem sie mit Hilfe von Werkzeugen Objekte (Material) bearbeiten.

Im Computer sind die zu bearbeitenden Objekte (Arbeitsgegenstände) repräsentiert in Form von Datenstrukturen des Anwendungssystems.

Im Computer sind die Funktionen zur Verarbeitung der Daten als Programm repräsentiert.

Das Werkzeug/Material-Leitbild trennt bewußt die Computerfunktion in

- Software-Werkzeug (Werkzeug-Funktion)
- Software-Material (Arbeitsgegenstand)

Das Software-Werkzeug ermöglicht Materialänderungen Œ *Anwendungsfunktionen*

Das Software-Werkzeug stellt den sachgerechten Umgang mit dem Material sicher Œ *Steuerfunktionen*

Der Benutzer stellt sich das Software-Werkzeug ein Œ *Adaptierfunktionen*

Das Software-Werkzeug gibt Auskunft über seine Benutzung Œ *Metafunktionen*

cation domain knowledge) stützen und andererseits die Auswahl der Funktionen abhängig machen von den konkreten Merkmalen der zu erledigenden Aufgaben, der Benutzer (von ihren Individualeigenschaften in "Rollen" abstrahiert) und der Werkzeugbenutzung. Dieses Tripel {Aufgabe, Rolle, Werkzeugbenutzung} bestimmt die software-ergonomischen Anforderungen an die Oberfläche.

Es hat sich hier als hilfreich erwiesen, für die Analyse des Arbeitssystems das Verfahren der Objektorientierten Analyse (OOA)[6,7,8] zu verwenden. Dabei stehen *Arbeitsgegenstände* ("objects") im Zentrum der Aufmerksamkeit, denen jeweils *Bearbeitungsverfahren* ("methods") zugeordent werden (Beispiel: Arbeitsgegenstand "Fahrauftragsformular", Bearbeitungsverfahren: "Formular ausfüllen"). Wenn nun das "Formular" im Rechnersystem als Maske realisiert wird, so benötigt man statt Bleistift und Radiergummi eine Reihe von *Werkzeugfunktionen* im Rechner.

Gerade bei kooperativen Organisationsformen bietet dieser oo-Zugang Vorteile gegenüber der im Software Engineering häufig verwendeten funktionalen Beschreibungsmethode. Wenn zum Beispiel in einem "virtuellen Unternehmen" die Just-in-Time-Logistik kooperativ zwischen den Disponenten der verschiedenen realen Unternehmen betrieben wird, so fällt es leicht, den Zugriff auf die gemeinsamen Objekte "Touren-Plan" und "Fahrauftrag" zu entwerfen. Wegen der in diesem Beispiel angenommenen Kooperation wird die Erfüllung der Kriterien "Steuerbarkeit", "Aufgabenangemessenheit" und "Erwartungskonformität" besondere Bedeutung haben. Habe die Benutzeranalyse ergeben, daß die Rolle des Disponenten von selten wechselnden, fachlich erfahrenen Benutzern ausgeübt wird, ist "Erlernbarkeit" und "Selbstbeschreibungsfähigkeit" von geringerem Gewicht.

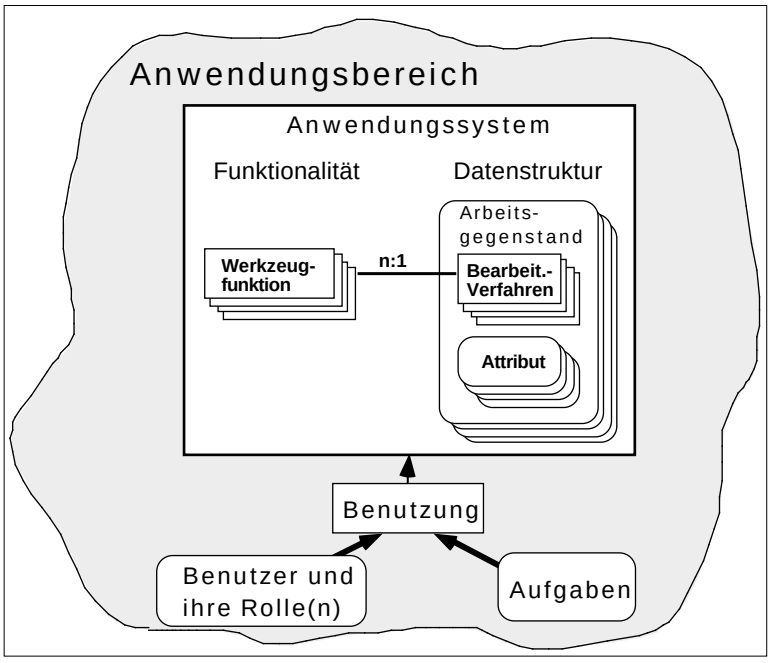



**Bild 1: Einflußfaktoren aus der Organisations-, Arbeits- und Aufgabenanalyse auf die Software-Gestaltung**

Als Ergebnis der Zweckgebundenen Sicht liegt eine Liste von Werkzeugfunktionen vor, die bei dem genannten Beispiel für die Bearbeitung eines "Fahrauftrags" erforderlich sind.

Im Folgenden soll das Vorgehen an dem Szenario aus einer Bauunternehmung verdeutlich werden, in der die Aufgabe der Angebotserstellung durch den Kalkulator gestaltet werden soll. (Kasten 8 - 15)

Kasten 8

**Szenario: Angebotserstellung** **Organisation: Bauunternehmung**

**Aufgabe: Erstellung eines Angebots aufgrund einer Ausschreibung**

**Aufgabenbestandteile:**

- Technische Informationen für die geplante Konstruktion sammeln
- Kostenfaktoren ermitteln
- Verfügbarkeit von Personal und Geräten prüfen
- Daten in Angebotsformular eintragen
- Angebots-Politik des Unternehmens klären (Faktoren: Kreditwürdigkeit des Auftraggebers, Konjunkturdaten, Lage des eigenen Unternehmens, zu erwartende Konkurrenz usw.

**Rollen:**

- Kalkulator
- Assistenzkraft
- Geschäftsführer etc.

Kasten 9

**Designschritt 1.1 :** (Anforderungen definieren)
**Organisation:** Bauunternehmung

**Aufgabe:** Erstellung eines Angebots aufgrund einer Ausschreibung

**Teilaufgaben:**

- Technische Informationen für die geplante Konstruktion sammeln
- Kostenfaktoren ermitteln
- Verfügbarkeit von Personal und Geräten prüfen
- Daten in Angebotsformular eintragen
- Angebots-Politik des Unternehmens klären Faktoren: Kreditwürdigkeit des Auftraggebers, Konjunkturdaten, Lage des eigenen Unternehmens, zu erwartende Konkurrenz etc.

**Rollen**

- Kalkulator
- Assistenzkraft
- Geschäftsführer etc.

Kasten 10

**Designschritt 1.2** (Wahlmöglichkeiten benennen)

**Bestimmen der Gestaltungsinhalte** durch Auswahl:

**Rolle:** Kalkulator

**Aufgabe:** Angebot erstellen

**Arbeitsgegenstand:** Angebotsformular

**Bearbeitungsverfahren**

- NewTenderForm
- DeleteTender
- StoreTender
- RetrieveTender
- Fill_inTender etc.

Kasten 11

**Designschritt 1.3** (Argumente zusammenstellen)

**Bestimmung der Rollenmerkmale** für Kalkulator

- solides Ingenieurfachwissen, kaufmännische Kenntnisse
- geringe Dialogerfahrung
- geringe EDV-Kenntnisse

**Bestimmung der Aufgabenmerkmale**

- Große Verantwortung für das Arbeitsergebnis
- Großer Kommunikationsbedarf über div. Medien mit Kunden, Lieferanten, and. Unternehmen sowie Ingenieuren und Kaufleuten im eigenen Haus
- Häufiger Zugriff auf div. Datenbanken
- wohl strukturiert: große Ablaufvarianz möglich

**Bestimmung der Benutzungsmerkmale**

Benutzungsweise:

- häufig
- unregelmäßig
- intensiv
- häufige Unterbrechungen

Kasten 12

**Designschritt 1.4** (Richtlinien anwenden)

**Bestimmung der Werkzeugfunktionen aus erfahrungs-gestützten Ergonomie-Regeln**

**Zusätzliche Werkzeugfunktionen**
(neben den üblichen WF)

- zur Ablaufsteuerung => Steuerfunktionen
- Wechsel zwischen verschiedenen Angebotserstellungen => Steuerfunktionen
- Zugriff auf Datenbanken => Steuerfunktionen
- Verschieben von Fenstern, Ändern von Layout usw. => Adaptierfunktion
- Tutorielle Hinweise, auch nach längerem Nichtgebrauch => Metafunktion
- Fehlerbehandlung => Metafunktionen
- Unterbrechungen => Steuer- and Adaptierfunktionen

Kasten 13

**Designschritt 1.5 (Entscheidungen notieren)**

**Notieren der Design-Entscheidungen 1. Sicht**

**Datenobjekte:** Angebotsformular

- Formularkopf (Absenderangaben, Projektangaben, Kalenderangaben etc.)
- Formularrumpf (Textfeld_Position, Zahlenfeld_Betrag, Zahlenfeld_Maße etc.)
- Formularfuß (Zahlenfeld_Betrag , Textfeld_Bedingungen etc.)

**Werkzeugfunktionen für Objekt:** Angebotsformular
(z.B. für **Bearbeitungsverfahren:** Fill_inTender)

- Posit_Cursor_inOtherField
- Browse_Directory
- Change_FieldSize
- Resize_Window
- Interrupt_FormFilling
- Resume_FormFilling etc.

Kasten 14

**Designschritt 2.1** (Anforderungen definieren)

Hier werden Anforderungen definiert, die aus der Benutzungsweise der Werkzeuge zur Bearbeitung der Arbeitsgegenstände abgeleitet werden können.

*Bereits festgelegt:*
*Aufgabe: Angebotsformular ausfüllen*
*Rolle: Kalkulator*
*Arbeitsgegenstand: Angebotsformular*
*Bearbeitungsverfahren: Fill_inTender*
*sowie zugehörige Werkzeugfunktionen*

**Anforderungen**

- flexible Formularbearbeitung
- leichter Zugriff auf Datenbestände
- Unterbrechbarkeit
- Besondere Unterstützung von oft wiederholten Arbeitsschritten
- Angebot der Werkzeugfunktionen nach Dialogtypen und Häufigkeit geordnet

Kasten 15

**Designschritt 2.2** (Wahlmöglichkeiten benennen)
Mögliche Dialogtypen benennen für jede Werkzeugfunktion

- Kommando-Dialog
- Dateneingabe-Dialog
- Auswahl-Dialog
- Direkt-Manipulations-Dialog

## *2. Interaktionsbezogene Sicht*

In der Interaktionsbezogenen Sicht werden Entscheidungen über Interaktionsformen und die Dialogstruktur (mit ihren temporalen Bedingungen) getroffen. Als Interaktionsformen stehen die Grundformen Dateneingabedialog, Kommandodialog, Auswahldialog und Direkte Manipulation sowie Kombinationsformen wie z.B. Maskendialog zur Verfügung. Man könnte die Werkzeugfunktionen zum Eintragen von Text in ein bestimmtes Feld statt über einen Kommandodialog durch einen Auswahldialog aktivieren. Es könnte jetzt zum Beispiel auch festgelegt werden, daß sie bei simultaner Arbeitsweise (auf der gleichen Datenbank) jeweils nur einem der aktiven Benutzer zur Verfügung stehen, das heißt, daß sie bei den anderen Benutzern dann inaktiv geschaltet werden.

Die oben genannten temporalen Bedingungen in der Dialogstruktur erlauben auch Entscheidungen über die erlaubten Reihenfolgen von Arbeitsschritten während der Aufgabenerledigung, der Anwendung von Bearbeitungsverfahren und des Aufrufs von Werkzeugfunktionen. (Es ist z.B. unsinnig, einen Text in einem noch nicht gefüllten Textfeld markieren zu wollen, um ihn anschließend zu kopieren: durch eine sichtbar gemachte Deaktivierung der Werkzeugfunktionen "cut" and "copy" würde ein Benutzer einen entsprechenden Hinweis erhalten können.)

Ergebnis der interaktionsbezogenen Sicht sind die Interaktionsformen für die Werkzeugfunktionen und jeweils die Vorbedingungen für ihre Anwendbarkeit auf ein Objekt bzw. auf einen Teil eines Objekts, die gegebenenfalls auch grafisch dargestellt werden können (vgl. Bilder 2 und 3).

Kasten 16

**Designschritt 2.3** (Argumente zusammenstellen)

- Vertraute Bearbeitungsreihenfolgen zwischen Zuständen erhalten, damit dem Benutzer die Arbeitsschritt-Reihenfolge nicht aufgezwungen wird.
- Veränderbarkeit der Arbeitsschritt-Reihenfolge zulassen, damit der Benutzer bei wechselnden Anforderungen die Software flexibel einsetzen kann.
- Interaktion entsprechend der Gruppierung der Werkzeugfunktionen nach Anwendungs-F. , Steuer-F., Adaptier-F. und Meta-F. zusammenfassen, damit dem Benutzer der unterschiedliche Werkzeug-Charakter vermittelt wird.
- Werkzeugfunktionen nach Häufigkeit ordnen, damit kurze Interaktionsdistanzen möglich sind.

**Bild 2:**
Beispiel einer Dialogstruktur in grafischer Darstellung – hier für das Angebotsformular mit den Vor- und Nachbedingungen für die Bearbeitungsverfahren "Save" und "Fill–In".

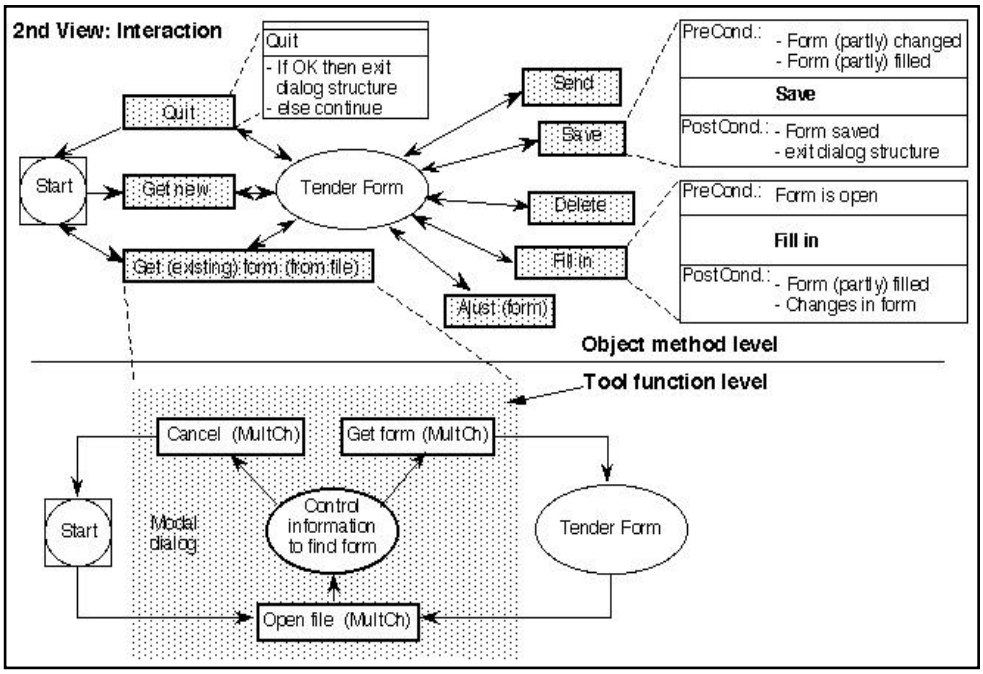
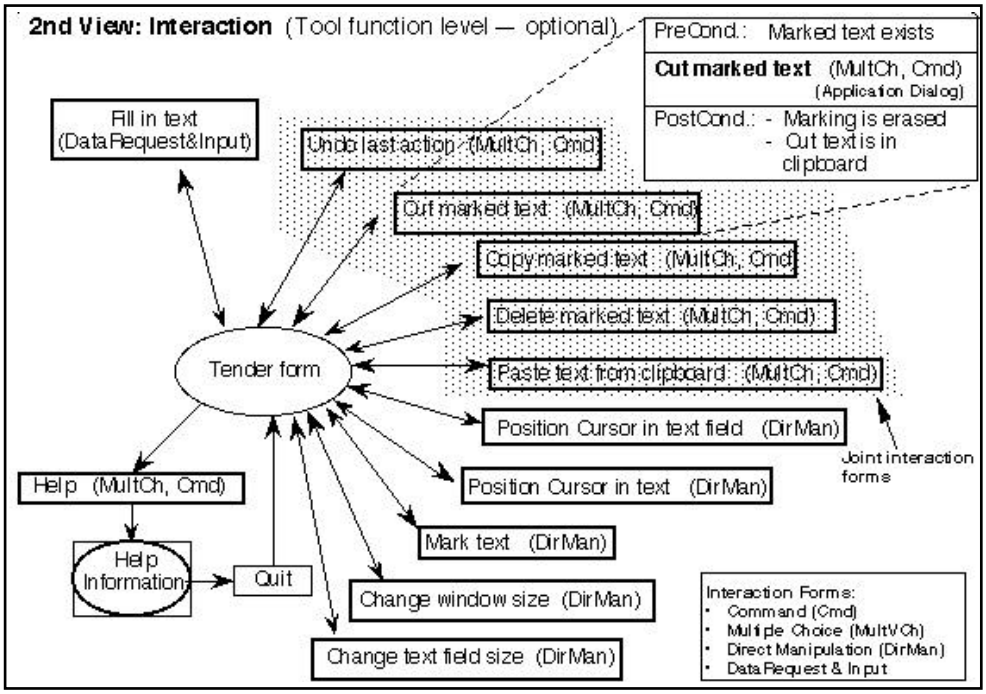



**Bild 3:**
Die oben gezeigte Dialogstruktur für das Angebotsformular, hier für die Werkzeugfunktion "Fill–In" und einige zu ergänzenden Funktionen – mit den Vor- und Nachbedingungen für die Werkzeugfunktion "Cut marked Text".




Kasten 17

**Designschritt 2.4** (Richtlinien anwenden)

**Regeln zur Auswahl des Dialogtyps** (Beispiele) [Daldrup [4]]

**Regel:** Bei geringer Interaktionserfahrung des Benutzers sollte die Kontrolle und Initiative für die Interaktion beim System liegen und der Designer für eine Werkzeugfunktion (oder WF-Gruppe) den Dialogtyp „Datenabfrage" wählen.

**Regel:** Bei mittlerer Interaktionserfahrung sollte die Kontrolle über die Interaktion beim System und die Initiative für die Interaktion beim Benutzer liegen, und der Designer für eine Werkzeugfunktion oder eine WF-Gruppe vden Dialogtyp „Auswahlangebot" wählen.

**Regel:** Bei großer Interaktionserfahrung sollte die Kontrolle und Initiative für die Interaktion beim Benutzer liegen und der Designer sollte für eine Werkzeugfunktion (oder WF-Gruppe) den Dialogtyp „Kommando" wählen.

**Regel:** Häufige, intensive und regelmäßige Benutzung einer Werkzeugfunktion oder einer Gruppe von Werkzeugfunktionen erfordert den Dialogtyp „Kommando" (ggfs. zusätzlich).

Kasten 18

**Designschritt 2.4** (Richtlinien anwenden ) – fortgesetzt

**Regeln zur Gestaltungskompetenz des Benutzers** (Beispiele) [Daldrup [4]]

**Allgemeines:** Die Interaktion soll für jede Benutzercharakteristik so gestaltet werden, daß sie den Benutzer von seiner eigentlichen Aufgabenbearbeitung nicht ablenkt.
Die Reaktionen des Rechners müssen für den Benutzer selbstbeschreibend sein und seinen Erwartungen entsprechen.

**Regel:** Erfordert die Aufgabenbearbeitung die Bearbeitung verschie dener Arbeitsgegenstände, die mit einer Vielzahl von Aktionen bearbeitet werden sollen, so sollte vom Designer ein Dialogtyp gewählt werden, der dem Benutzer verschiedene Handlungsmöglichkeiten offenläßt.

**Regel:** Der Benutzer kann dann während der Bearbeitung einen ihm angenehmen und der Aufgabenbearbeitung angemessenen Umgang mit dem Werkzeug verfolgen.
Zum Beispiel bietet das Auslösen einer Werkzeugfunktion in einem „Auswahl-Dialog" diese Möglichkeit (mit mehreren Werkzeugfunktionen als Alternativen).

Kasten 19

**Designschritt 2.5** (Entscheidungen notieren)

**Ebene Bearbeitungsverfahren**

Für jedes Bearbeitungsverfahren werden die "Pre-Conditions" und "Post-Conditions" notiert.

**Bearbeitungsverfahren:** Formular ausfüllen (Fill_inTender)
Pre-Conditions: Formular ist geöffnet
Post-Conditions: Formular ist teilweise ausgefüllt
Formular wurde verändert

**Ebene Werkzeugfunktionen**

Für jede Werkzeugfunktion werden die Dialogtypen und die Zustandsübergänge festgelegt.

**Bearbeitungsverfahren:** Formular ausfüllen (Fill_inTender)
Werkzeugfunktion: Text ausschneiden (Cut marked text)
Zugeordnete Dialogtyp(en): Auswahldialog
Kommando-Dialog

**Gruppierung der Werkzeugfunktionen**
(mit Cmd, MultCh)
Undo, Cut marked Text, Copy marked text
Delete marked text, Paste marked text

Kasten 20

**Designschritt 3.1** (Anforderungen definieren)

Für jede **Werkzeugfunktion** (bzw. Gruppe von Werkzeugfunktionen) wird die **Präsentation der** vorher ausgewählten **Dialogtypen** festgelegt aufgrund der Anforderungen aus dem Anwendungs-Kontext

1. durch **Festlegen der physikalischen Geräte** für Input/Output
   Visuelle Präsentation (Display-Technik: Farbe, Auflösung usw.)
   Auditive Präsentation (Diplay-Technik: Generierte / Sampled Sprache usw.)
   Haptische Präsentation (Diplay-Technik: Auflösung usw.)
2. durch **Festlegen der Darstellung** auf diesen Geräten
   textuelle / symbolische Darstellung
   graphische Darstellung usw.

### *3. Präsentationsbezogene Sicht*

Ziel dieser Sicht ist die Festlegung konkreter Erscheinungsformen für die Interaktionsformen. So stehen zum Beispiel für die Interaktionsform "Auswahldialog" die Interaktionsarten (und Widgets) "Menü" (in einer Reihe von Spezialformen), "Listbox" (mit und ohne Scrollbars), "Radio-Buttons", "Checkboxes" und die Funktionstasten zur Verfügung. Um hier für das Oberflächendesign ergonomische Grundlagen zu finden, sind etwa arbeits- oder kognitionspsychologische Forschungsergebnisse wie in den Regeln von Smith und Mosier[3] heranzuziehen. (Beispiel: ein Menü sollte nicht mehr als 10 Items haben. Stattdessen könnte man eine hierarchisch gestufte Menüfolge oder eine Liste verwenden oder - falls jeweils nur einige wenige Items in einem Arbeitskontext ausgewählt werden - die häufig verwendeten Items zusätzlich an den Kopf des Menüs kopieren.) (Kasten 20 - 23)

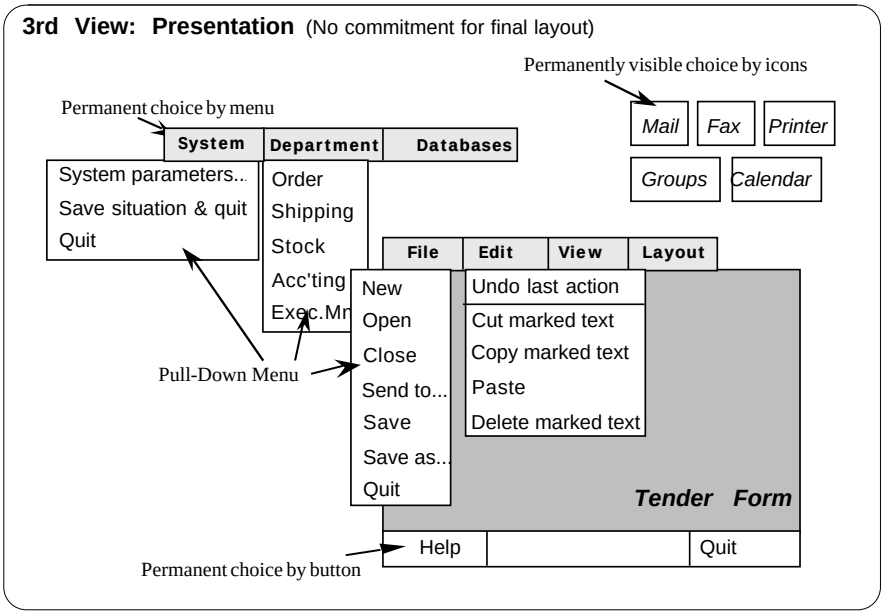



**Bild 4: Erste Grob-Plazierung von Widgets auf der Arbeitsfläche**
Anschließend erfolgt die genaue Plazierung, Größen-, Schrift- und Farbwahl

Jetzt muß auch eine Entscheidung über die anzuwendende Input/Output-Hardware getroffen werden, da viele Interaktionenformen nur auf speziellen Geräten realisierbar sind, z.B. ASCII- oder Graphik-Bildschirm (Farbe?), Zeigegerät (Maus, Trackball, Joystick, Touchpad), Voice-Input, Voice-Output, Funktionstasten usw.

Für kooperative, u.U. simultane Arbeitsprozesse tritt die Frage auf, wie man den Benutzern verdeutlicht, daß einer von ihnen gerade bestimmte Werkzeugfunktionen oder Objekte "blockiert". Um dieses auf der Oberfläche zu präsentieren, könnten die Funktionen entweder nicht mehr im Auswahlangebot erscheinen oder "graugeschaltet" sein. Das software-ergonomische Wissen muß dazu ausgewertet werden, um festzustellen, welche der beiden Präsentationsformen für die gegebenen Benutzer und Aufgaben geeigneter ist. Die gleiche Präsentationsform könnte auch für die wegen anderer temporaler Bedingungen nicht aktivierbaren Werkzeugfunktionen verwendet werden, falls nicht auch der Grund der Nichtaktivierbarkeit durch die Präsentationsform signalisiert werden soll. (bild 4)

Ergebnis der dritten Sicht ist somit die Festlegung konkreter Erscheinungsformen für die in der vorigen Sicht festgelegten Interaktionsformen.

Mit den bisherigen Designentscheidungen kann ein Prototyp der Benutzungsoberfläche erstellt werden. Dazu müssen die bisher ausgewählten Elemente der Benutzungsschnittstelle vollständig beschrieben und positioniert werden. Dies wäre dann Designschritt 3.5. Ein User Interface Prototyping Tool oder ein User Interface Management System ist dazu gut geeignet.

Eine Anbindung der Schnittstelle an das Anwendungssystem kann in einer frühen Phase des Entwicklungsprozes-

Kasten 21

**Designschritt 3.2** (Wahlmöglichkeiten benennen)

Zur **Präsentation von Werkzeugfunktionen** können z.B. folgende Abstrakte Interaktionsobjekte verwendet werden:

| **Dialogtyp** | **Abstraktes Interaktionsobjekt** |
|---|---|
| Kommando-Dial. | Texteingabe-Feld |
| Auswahl-Dialog | Menüs (verschiedener Art), Buttons |
| Direkt-Manip.-Dialog | Piktogramm (z.B. in Palette,Tool Bar) |

Zur **Präsentation von Parametereinstellungen** können z.B. folgende Abstrakte Interaktionsobjekte verwendet werden (oft in Dialog-Panels / Dialog Boxes zusammengefaßt):

| **Dialogtyp** | **Abstraktes Interaktionsobjekt** |
|---|---|
| Auswahl-Dialog | Menü (verschiedene Arten), Listbox (ggfs mit Scroll Bar), Radio Buttons, Check Boxes |
| Direkt-Manip.-Dialog | Piktogramm |

Zur **Präsentation von Dateneingaben** können z.B. folgende Abstrakte Interaktionsobjekte verwendet werden (oft in Dialog Panels / Dialog Boxes zusammengefaßt):

| **Dialogtyp** | **Abstraktes Interaktionsobjekt** |
|---|---|
| Dateneingabe-Dialog | Textfeld, Listbox mit Textfeld |
| Auswahl-Dialog | Listbox (ggfs mit Scroll Bar) |

**Varianten des Auswahldialogs**

**Auswahldialog** (1 : n oder m : n)

- Scroll lists
- Check boxes
- Textmenüs
- Radio buttons
- Piktogramm-Menüs
- (Labelled or commented) function keys etc.

**Menü–Präsentationsformen:**

- Pop-up
- Pull-down
- Statisch
- Rechteckig
- Kreisförmig
- Matrix
- Nicht-hierarchisch
- Hierarchisch

etc.

**Auswahlmethode:**

- Zeigen auf ein Bildschirmobjekt (= Referenzieren eines Datenobjekts) durch Maus-, Stift-, Finger- oder Tasten-gesteuerte Cursor etc.
- eine Alphataste, als Bezeichnung für ein Menü-Item

**Feedback:**

- Markieren des/der gewählten Item(s),
- Keine Markierung bei "direkter Aktion", z.B. Doppel-Mausklick zum Dateiöffnen

Kasten 22

**Designschritt 3.3 und 3.4** (Argumente zusammenstellen) (Richtlinien anwenden)

**Für Werkzeugfunktionen:**

Abstrakte Interaktionsobjekte für den jeweiligen Dialogtyp hängen von den Kriterien Erwartungskonformität und Selbstbeschreibungsfähigkeit (Konsistenz, Transparenz) ab

=> Prüfen des Kontextes

Zum Beispiel:

- Bei hoher Häufigkeit, Regelmäßigkeit und Intensität => Kommandoeingabe (Shortcut), Buttons, Piktogramme
- Bei geringer Häufigkeit und geringer Dialogerfahrung => Menüs, Buttons

**Für Parameter-Eingabe**

- Wenn jeweils andere Einstellungen ausgeschlossen sind:
  => Radio Buttons
- Wenn jeweils ein Parameter nur binär ist (j/n):
  => Checkboxes
- Wenn größere Zahl von Einstellwerten möglich:
  => Listbox

ses natürlich nicht erfolgen, stattdessen müssen die Zugriffe auf die Daten und Funktionen der Applikation als "Mockup" implementiert werden. Dieser Prototyp wird anschließend mit Anwendern und Benutzern evaluiert, um Grundlage für eine neue Version zu schaffen.

## 3.3 Verallgemeinerung der unter MUSE entstehenden Oberflächen

Theoretisch müßte bei einem Vorgehen nach MUSE für ein Anwendungssystem bei jedem Tripel (Rolle, Aufgabe, Werkzeugnutzung) wegen der Unterschiede in den Merkmalen eine spezielle Benutzungsoberfläche entstehen. Das ist weder ökonomisch zu vertreten noch in einem Unternehmen im praktischen Einsatz der Software durchzuführen. Aus diesem Grunde wird der Designer schon früh im Designprozeß ähnliche Anforderungen zusammenfassen und so die Anzahl unterschiedlicher Oberflächen reduzieren. Die Vereinigungsmenge mehrerer Tripel führt zu Wertebereichen für jedes einzelne Merkmal: wenn zum Beispiel für mehrere Benutzer die Werk-

zeugnutzung *häufig bis selten* auftreten kann, so ist entsprechend die gemeinsame Oberfläche so zu entwerfen, daß beide Benutzungshäufigkeiten entsprechend unterstützt werden (etwa wie üblich für die häufige Nutzung sogenannte Shortcuts durch Kommandokürzel, für die seltene eine Menüführung) (Kasten 23). Der Extremfall "*eine* Benutzungsoberfläche für *alle* Benutzer und Aufgaben" führt dann zur Standardsoftware. Es jedoch angemerkt, daß dieser Extremfall eigentlich nie eintritt, denn auch die Entwickler von Standardsoftware haben - oft unbewußt - ein Bild von den Benutzern ihres Programms und von den Aufgaben, die damit bearbeitet werden sollen, und schließen z.B. Analphabeten aus.

Kasten 23

**Zusammenfassung von Ergebnissen** der Aufgabenanalyse führt zur Möglichkeit, die **Vereinigungsmenge getrennt entworfener Oberflächen** zu bilden.

Bei Software, die in verschiedenen Ländern eingesetzt werden soll, tritt ein weiteres Problem auf - auch bei Software mit sehr spezifischen Anwendungsbereichen: die sogenannte "Internationalierung" der Benutzungsoberfläche; hier spräche man jedoch besser von "Anpassung an andere Kulturkreise". Dazu müssen abgestufte Maßnahmen zur Übertragung eines Programmes in eine andere Sprachwelt geben, beginnend mit einer reinen Übersetzung der Texte und Bezeichnungen auf der Oberfläche über andere Maßsysteme, alphabetische oder ikonische Schriften, Schreibarten (links-rechts/rechts-links), andere Zähl- und Bezeichnungsweisen oder Kalendersystemen bis hin zu in anderen Kulturen üblichen Metaphern und Piktogrammen, gar nicht zu reden von anderen rechtlichen oder technischen Vorschriften im Anwendungsbereich, die sogar eine Änderung der Anwendungsfunktionen erfordern (s. Kasten 24).

Kasten 24

**Internationalisierung** bedeutet Anpassung eines Programms an andere Kulturkreise:

- Übersetzung der Begriffe und Texte auf der Oberfläche in andere Sprachen
- Umstellung auf andere Maßsysteme
- andere Alphabete, Schriften (Buchstaben- versus ikonische Schriften)und Schreibarten (links-rechts, rechts-links, vertikal)
- andere Zähl- und Bezeichnungsweisen
- andere lexikographische und numerische Ordnungssysteme
- andere Kalendersysteme
- andere Bedeutung von Piktogrammen und Metaphern
- andere technische Vorschriften / Normen
- andere Rechtsvorschriften z.B. für Dokumente oder für Zahlungsgeschäfte

Bei einer Analyse des Kontextes, in dem die Software eingesetzt werden soll, müssen also neben der Ergebnissen der Arbeits-, Organisations- und Aufgabenanalyse auch die kulturellen Unterschiede berücksichtigt werden, um daraus Gestaltungsmaßnahmen ableiten zu können.

## 4. Grenzen von MUSE und Ausblick

MUSE II ist nicht der einzige Versuch, durch ein systematisches Vorgehen den Entwurf von Benutzungsoberflächen zu unterstützen. Hier sei deshalb beispielhaft auf die Arbeiten von Zeidler/Zellner[9] und von Ziegler[10] verwiesen.

Die im Tutorial vorgestellte Methode MUSE II soll die Entscheidungskomplexität beim Oberflächendesign reduzieren und die Einflußfaktoren auf die Benutzbarkeit eines Anwendungssystems bewußt machen. Bei dem Vor-

gehen nach MUSE wird vorausgesetzt, daß die an der Gestaltung der Benutzungsoberfläche Beteiligten auch die Kriterien kennen und die Zusammenhänge zwischen den Merkmalswerten und den Designentscheidungen nachvollziehen können. Dieses breite Wissen über die Arbeitswissenschaften, die Arbeits-, Kognitions- und gestaltpsychologie sowie über das Software-Engineering kann weder beim Anwender noch bei den Software-Entwicklern vorausgesetzt werden. Daher ist die Methode praktisch nicht einsatzfähig, es sei denn, die Software-Entwicklung erfolgt in multidisziplinären Teams.

Um dem Problem abzuhelfen, werden in einigen Forschungsgruppen Werkzeuge entwickelt, die eine wissensbasierte Unterstützung des Designprozesses bieten sollen. Die Projekte TASK[11]/TOOLS[12] in Stuttgart, IDA[13] in Birlinghofen, DIADES II[14] in Darmstadt und JANUS[15] in Bochum bauen dabei auf jeweils spezielle Vorgehensmodelle und decken vorwiegend die Bereiche ab, die den zwei letzten Sichten nach MUSE entsprechen, während das Projekt EXPOSE[16] in Oldenburg und Rostock genau das Vorgehensmodell von MUSE I (eine Variante mit vier Sichten) unterstützt.

Es ist zu erwarten, daß einige der in diesen Forschungsprojekten entstehenden Werkzeuge in absehbarer Zeit in stabilen Versionen auf dem Markt sind. Bis dahin sind die Designer gefordert, die software-ergonomische Literatur (siehe Weiterführende Literatur) allein zu erarbeiten.

Kasten 25

**In Entwicklung befindliche Werkzeuge für die Gestaltung von Benutzungsoberflächen**

- TASK[11] / TOOLS[12]
- IDA[13]
- DIADES II[14]
- JANUS[15]
- EXPOSE[16]

Das Verbundprojekt EXPOSE der Universitäten Oldenburg und Rostock wurde gefördert durch den Bundesminister für Forschung und Technologie im Förderschwerpunkt Arbeit und Technik
Förderkennzeichen: 01 HK 190 /4 (Oldenburg) und 01 HK 291 /2 (Rostock)

Das Projekt MUSE II der Universität Oldenburg wurde gefördert durch den Nierdersächsischen Minister für Wissenschft und Kultur im Rahmenprojekt Informatiksysteme.

Das Tutorial wurde während der Fachtagung Software-Ergonomie '97, Dresden 3.-6. März 1997, mit Unterstützung der Deutschen Informatik-Akademie (DIA) durchgeführt.

**Adresse des Autors:** Prof. Dr.-Ing. Peter Gorny, Carl v. Ossietzky Universität Oldenburg
FB Informatik, Abt. Computer Graphics & Software-Ergonomie, 26111 Oldenburg
E-Mail: Gorny@Informatik.Uni-Oldenburg.DE URL: http://www-CG-HCI.Informatik.Uni-Oldenburg.DE/

## Anmerkungen

1 Der englische Begriff "usability" kann im Deutschen sowohl "Gebrauchstauglichkeit" (des ganzen Programms) wie "Benutzbarkeit" (der Benutzungsoberfläche) bedeuten.

2 S. **Smith**, and J. **Mosier**: Guidelines for designing user interface software. The Mitre Corp., Bedford, MA, 1986.

3 **Iannella**, R.: HyperSAM v 2.0 - A hypertext interface to Smith and Mosier. HyperCard Stack Bond University, Gold Coast QLD, Australia, 1993.

4 A. **Viereck**, P. Gorny: Software-Ergonomische Beratung bei der Benutzungsschnittstellen-Entwicklung. In: M. Frese u.a. (Hg.): Software für die Arbeit von morgen. Berlin u.a. 1991: Springer. 309-408.

P. **Gorny**, A. **Viereck**, L. **Qin** und U. **Daldrup**: Slow and Principled Prototyping of Usage Surfaces: a Method for User Interface Engineering. In: H..Züllighoven u.a.(Hg.): Proceedings RE'93 - Prototyping. Bonn April 1993. Stuttgart 1993: B.G. Teubner, 125-133.

U. **Daldrup**: (Un)Ordnung im Gestaltungsprozeß menschengerechter Software. Frankfurt/Main usw. 1996. Peter Lang Verlag.

5 H. **Dunckel** u.a.: Konstrative Aufgabenanalyse. Zürich: 1993.

6 P. **Coad**, E. **Yourdon**: Object-Oriented Analysis. Englewood Cliffs: Yourdon Press, 1991.

7 J. **Rumbaugh** et al: Object-Oriented Modeling and Design. Englewood Cliffs: Prentice Hall, 1991.

8 R. **Budde**, H. **Züllighoven**: Programmierwerkzeuge in einer Programmierwerkstatt. München: Oldenbourg Verlag, 1990.

9 A. **Zeidler**, R. **Zellner**: Software-Ergonomie: Techniken der Dialoggestaltung. Oldenbourg Verlag, München 1994.

10 J. **Ziegler**. R. **Ilg**: Benutzergerechte Software-Gestaltung. Oldenbourg Verlag, München, 1993.

J. **Ziegler**: Eine Vorgehensweise zum objektorientierten Entwurf graphisch-interaktiver Informationssysteme. Dissertation. Universität Stuttgart 1996. (In Druck)

11 A. **Beck**: Methoden und Werkzeuge für die frühen Phasen der Software-Entwicklung. In: D. Ackermann et al: Software-Ergonomie '91: Benutzerorientierte Software-Entwicklung. Stuttgart: Teubner, 1991.

12 C. **Janssen**, A. **Weisbecker**, J. **Ziegler**: "Generierung graphischer Benutzungsschnittstellen aus Datenmodellen und Dialognetz-Spezifikationen". In: H. Züllighoven, A. Altmann, E.-E. Doberkat (Hg): Requirements Engineering'93: Prototyping. German Chapter of the ACM, Band 41, pp. 335-347. Stuttgart: Teubner, 1993.

13 H. **Reiterer**: A Human Factors Based User Interface Design. In: Grechenig, T., and Tschelegi, M. (eds.): Human Comuter Interaction. Proc. Vienna Conference VCHCI'93. Sept. 1993. Lecture Notes in Computer Science. Springer Wien, 1993. 291-302.

14 I. **Dilli**, G. **Vogt** and H.-J. **Hoffmann**: Diades II: Ein objektorientiertes Entwurfsweerkzeug für interaktive Benutzungsschnittstellen. Ergonomie & Informatik, Mitteilungen des GI-Fachauschusses 2.3, 1993, Nr. 19, 14-21.

15 H. **Balzert**: Das JANUS-System: Automatisierte wissensbasierte Generierung von Mensch-Computer-Schnittstellen.. In: Informatik - Forschung und Entwicklung. Heidelberg: Springer, 1994.

16 P. **Gorny**, A. **Viereck**: Ein Software-Ergonomie-Expertensystem. In: D. Ackermann u. E. Ulich (Hg.): Software-Ergonomie '91. Stuttgart 1991: Teubner. 152-161.

P. **Forbrig**, P. **Gorny** und A. **Viereck**: Unterstützung des Software-Design-Prozesses durch EXPOSE. In W. Coy, P. Gorny, I. Kopp und C. Skarpelis (Hg.): Menschengerechte Software als Wettbewerbsfaktor. Forschungsansätze und Anwenderergebnisse aus dem Programm "Arbeit und Technik". Stuttgart 1993, B.G. Teubner. 463-479.

P. **Gorny**: EXPOSE - HCI-Counseling for User Interface Design. In: Knut Nordby et al (eds): Human-Computer Interaction, Proc. Interact '95. 5th Int'l. Conf. on Human Computer Interaction. Lillehammer, Norway, June 1995. London etc. 1995, Chapman&Hall. pp 297-304.

## Weiterführende Literatur

**Eberleh** u.a.(Hg.): Einführung in die Software Ergonomie. Reihe: MCK-Grundwissen. Berlin, New York: de Gruyter, 1993

**Dix**, **Finlay**, **Abowd**, **Beale**: Human-Computer Interaction. New York u.a.: Prentice Hall 1993.

**Koch**, **Reiterer**, **Tjoa**: Software-Ergonomie. Wien, New York: Springer, 1991

**Preece** et al: Human Computer Interaction: Addison-Wesley, 1994

**Richenhagen**: Bildschirmarbeitsplätze - Mehr Schutz für Arbeitnehmer. Neuwied: Luchterhand 1995.

**Shneiderman**: Designing the User Interface: Strategies for Effective Human-Computer Interaction. 2nd Edition, Reading u.a.: Addison Wesley, 1993