DYNAFORM
Ein interaktives Formularsystem zum Aufbau und zur Bearbeitung von Datenbasen

Michael Herczeg
Projekt INFORM
Abteilung Dialogsysteme / Mensch-Maschine-Kommunikation
Institut für Informatik - Universität Stuttgart

Zusammenfassung

Anhand des Beispiels der Spezifikation von Anwendungsprogrammen zur Dateneingabe und Datenausgabe soll gezeigt werden, wie sich die Anforderungen an die Benutzerschnittstelle ändern, wenn der Anwender selbst die Applikation gestaltet und modifiziert. Es werden einige Eigenschaften diskutiert, die eine solche Benutzerschnittstelle haben sollte, um eine adäquate Kommunikation mit einem Benutzer ohne EDV-Erfahrung zu erlauben. Dies sind die Funktionsaufrufmechanismen, das Stornieren von Funktionen, das Sprachkonzept, sowie Fragen des Bildschirmlayouts und Codierungsmethoden, wie z.B. Farbe. Durch die Implementierung eines Prototypen mit einer derartigen Benutzerschnittstelle wurden Erfahrungen gesammelt, um die theoretischen Überlegungen zu stützen.

Stichwörter:

benutzergestaltbare Schnittstellen, Benutzermodell, Farbe, Formulardialoge, graphische Hilfsmittel, Interaktionstechniken, Layout, Maskengeneratoren, Sprachkonzepte, UNDO, Visualisierungshilfen

1. Einleitung

Ein großer Teil der existierenden Programme im Bereich der Büroautomatisierung bestehen aus der Abwicklung von Dateneingabe und Datenausgabe über den Bildschirm. Die Programmierung dieser Anwendungsschnittstelle wurde immer mehr automatisiert durch Maskengeneratoren. Man spezifiziert, welche Daten in welcher Reihenfolge eingelesen werden sollen, wo und wie sie auf dem Bildschirm darzustellen sind. Es ist nicht mehr erforderlich, dies in einzelnen Programmstatements explizit zu programmieren. **Es hat eine Entwicklung vom "WIE?" zum "WAS?" stattgefunden.**

Durch diese Entwicklung wird zunehmend der Anwender in die Lage versetzt, die Spezifikation und Modifikation seiner Bildschirmmasken selbst vorzunehmen. Diese Möglichkeit wurde durch die genannten Maskengeneratoren vorbereitet, die bei den meisten Softwareerstellern Anwendung finden. Es existiert aber noch eine große Kluft zwischen derartigen durchaus bewährten Generatoren und zukünftigen Systemen, die es auch dem Anwender erlauben sollen, seine Anwendungssysteme in einer für ihn angemessenen Weise durch Spezifikation seiner sich ändernden Bedürfnisse zu gestalten.

Das hier vorgestellte System DYNAFORM [7] ist ein prototypisches System, mit dem eine Weiterentwicklung der üblichen Maskengeneratoren im Hinblick auf eine anwendergerechte Benutzerschnittstelle untersucht wird.

DYNAFORM wurde auf einer VAX 11/780 in FRANZ LISP [6] implementiert und entstand im Rahmen der Forschungsarbeiten des Projekts INFORM [1]. Als Terminal wurde das Farbgraphiksystem GIGI von DEC verwendet. Das Rasterterminal mit einer Matrix aus 768 x 480 Pixeln, 8 Farben und Blinking besitzt einen eigenen Prozessor, auf dem ein Interpreter für eine Farbgraphiksprache läuft. Das Lisp-Programm steuert diesen Interpreter mittels Printsequenzen an.

2. Modifizierbarkeit von Systemen

Die Erstellung von Anwendungssystemen im Bereich der Büroautomatisierung ist ein langwieriger Spezifikationsprozeß zwischen Anwendern, Systemanalytikern und Programmierern. Dieser Prozeß wird gewöhnlich mehrfach durchlaufen, bis das Produkt durch wiederholte Modifikation und Erweiterung einen akzeptablen Zustand erreicht. Daß dies nicht in einem einzigen Schritt erreicht werden kann, hat verschiedene Gründe:

- Komplexität der Anwendung
- Unklarheit über den Sollzustand
- Kommunikationsprobleme zwischen Anwender, Systemdesigner und Programmierer
- Berücksichtigung von Sonderfällen
- sich ändernde Anforderungen während der Entwicklung
- Unzulänglichkeiten der Maschine

Selbst wenn eine akzeptable Lösung erreicht ist, kann diese durch sich ändernde Anforderungen nach der Fertigstellung des Systems teilweise unbrauchbar geworden sein.

Der Anwender ist also ständig auf den Systementwickler angewiesen, wenn es darum geht, Änderungen an der Software durchzuführen. Das erneute Durchlaufen eines Modifikationszyklus ist zeitraubend, teuer und häufig wieder mit neuen Fehlern behaftet. Daher wäre es günstig, wenn der Anwender selbst an dem Anwendungssystem Änderungen vornehmen könnte. Dem Anwender kommt hierbei eine aktive Rolle bei der Gestaltung des Anwendungssystems zu. Der Computer wird zum konvivialen Werkzeug [4].

Im Fall der Systeme zur Ein- und Ausgabe von Daten über Bildschirmmasken (Formulare) treten genau diese Probleme häufig auf. Verschiedene Änderungsgründe können auftreten, z.B.:

- es müssen zusätzliche Daten erfasst werden
- es entfallen zu erfassende Daten
- die Daten unterliegen anderen Beschränkungen
- die günstigste Erfassungsreihenfolge ändert sich
- das Layout der Bildschirmmaske muß umgestaltet werden
- die Bezeichnungen von Feldern ändern sich

DYNAFORM zeigt prototypisch Methoden auf, mit denen der Anwender diese Modifikationen vornehmen kann. Insbesondere die Probleme, die an der Benutzerschnittstelle auftauchen, werden dabei untersucht. Abbildung 2 zeigt das veränderte Formularlayout, das aus dem Formular in Abbildung 1 durch einfach durchzuführende Modifikationen entstanden ist.

Abbildung 1:

Ausgangsformular, eingebettet in die Benutzerschnittstelle

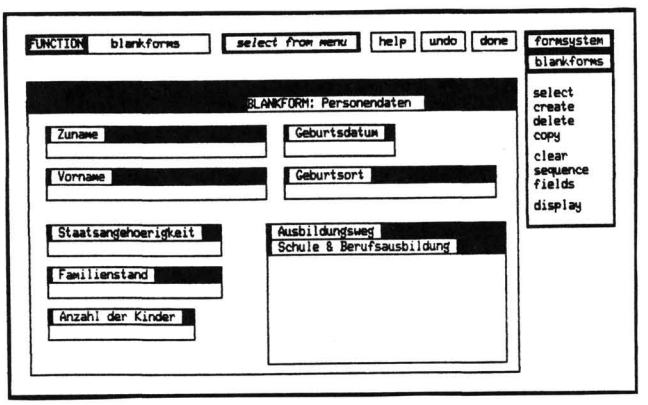
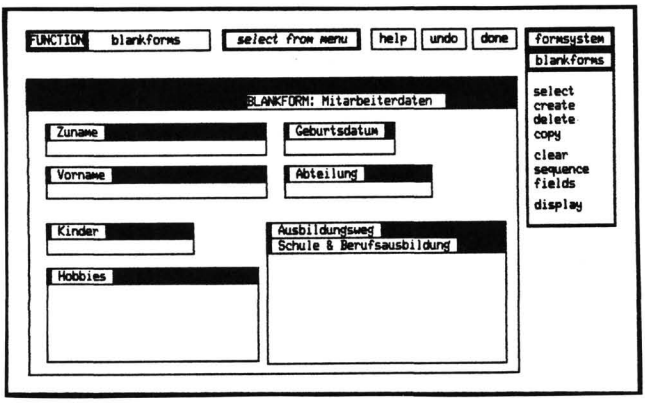


Abbildung 2:

im Layout modifiziertes Formular

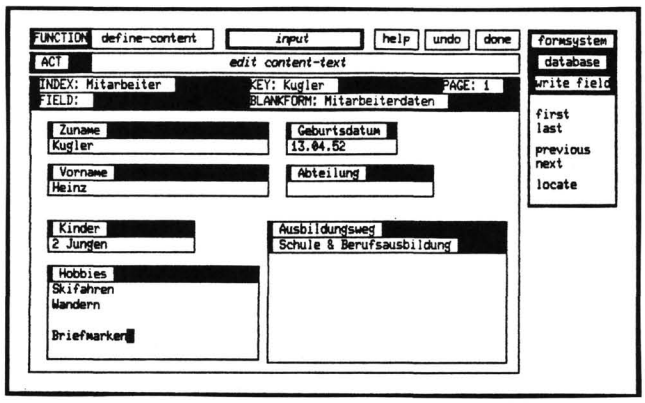


3. Konzeption der Anwendungschnittstelle von DYNAFORM

Über die **Metapher vom Formular** stellen wir dem Benutzer eine Schnittstelle zur Verfügung, mit der er Formulare generieren und spezifizieren kann, die ihm dann als Bildschirmmasken zur Dateneingabe und Datenausgabe dienen können.

Abbildung 3:

Dateneingabe über ein Formular



Innerhalb von Formularen werden Felder kreiert, die die Repräsentanten der sonst üblichen Maskenfelder darstellen. Die Definition und Anordnung der Felder findet mittels verschiedener Kreations- und Manipulationsfunktionen statt, deren Eignung und Benutzerfreundlichkeit einen Teil der Untersuchungen darstellen. Die spätere Abarbeitungsreihenfolge der Felder bei der Datenerfassung läßt sich über eine Verkettung der Felder festlegen.

Eine der zahlreichen Feldspezifikationsfunktionen ist die Definition von Abhängigkeitsstrukturen ("Constraints"). Mit Hilfe von verschiedenen Constraints lassen sich Inhalt und Reihenfolge der Abarbeitung

der Felder einschränken oder auch der Inhalt von Feldern automatisch deduzieren. Die Realisierung einer geeigneten Benutzerschnittstelle zur Definition von Constraints steht zum gegenwärtigen Zeitpunkt noch am Anfang. Dem Benutzer muß eine Art formaler Spezifikationssprache angeboten werden, ohne die Kommunikation wesentlich komplizierter zu gestalten.

Durch die Definition eines "Defaults" kann der Feldinhalt durch eine Art Voreinstellung einen Standardwert erhalten.

Zur Verwendung der Formulare als Datenerfassungs- und Datenmanipulationsschnittstelle gibt es Methoden, um die definierten Formulare in Karteien, Schlüssel, mehrseitige Formulare und Zusatzformulare zu organisieren. Felder können statt eines Inhalts einen Verweis auf ein anderes Formular enthalten.

4. Konzeption der Benutzerschnittstelle von DYNAFORM

Der Benutzer kommuniziert mit dem System mittels einer gewöhnlichen Schreibtastatur, einer Funktionstastatur inclusive Cursortasten, sowie einem Graphiktablett als **Zeigeinstrument**. Mit diesem Tablett kann ein Fadenkreuz auf dem Bildschirm positioniert werden. Da der Benutzer einige Layout- und Auswahloperationen durchzuführen hat, hat er durch dieses Zeigeinstrument eine schnelles und leicht zu bedienendes Instrument zur Verfügung. In persönlichen Computern der oberen Leistungsklasse (z.B. LISP-Maschine, Xerox-STAR [10], usw.), die zum Teil bereits zur Serienproduktion gereift sind, hat sich das Vorhandensein einer mit dem Graphiktablett eng verwandten Zeigevorrichtung, der "Maus", als wertvolles Kommunikationswerkzeug erwiesen. Die Notwendigkeit dazu erwächst aus der Konzeption, den Bildschirminhalt als manipulierbares Objekt zu behandeln.

Die Verwendung von Bildschirmmasken waren bei der Entwicklung von Anwendungssystemen ein Schritt, um von "teletype-artigen" Kommunikationstechniken wegzukommen. Ein anderer Schritt ist die **Verwendung von Fenstersystemen**. Dieser Methode liegt die Idee zugrunde, mehrere logische Bildschirme zur Verfügung zu haben. Auf diese Weise kann gleichzeitig Information verschiedener Informationskategorien dargestellt und in mehreren Kontexten parallel gearbeitet werden. Auch DYNAFORM bedient sich einer einfachen Fenstertechnik. So werden z.B. Menüs, Statusinformationen, Fehlermeldungen und die eigentliche Anwendung jeweils in eigenen Fenstern untergebracht. Fenster können sich zeitweise gegenseitig überlappen, da der zur Verfügung stehende physische Bildschirm zu klein ist, um allen Fenstern einen eigenen Platz zuordnen zu können.

Diese Menge von Informationskategorien begründet sich darauf, dem Benutzer eine weitergehende Kommunikation mit dem System zu ermöglichen, als es die meisten herkömmlichen Benutzerschnittstellen erlauben. **Die Benutzerschnittstelle bedarf einer breiteren Funktionalität und Transparenz.**

Auf Abbildung 1, 2 und 3 ist der Zustand des Bildschirms während einer Sitzung mit DYNAFORM zu sehen. Es befinden sich mehrere Fenster auf dem Bildschirm.

Das größte Fenster enthält die eigentliche Anwendung, in diesem Fall das zu bearbeitende Formular. Auch die einzelnen Formularfelder können wieder als eigene Fenster betrachtet werden.

Auf der rechten Seite des Bildschirms erscheinen die **Menüs**. Mit Ausnahme der Überschriftszeile überlappen sie sich nacheinander, um ihre hierarchische Ordnung widerzuspiegeln. So bleibt für den Benutzer der Pfad ersichtlich, über den er seine momentane Umgebung erreicht hat. Dies mindert die Gefahr, daß der Benutzer im System "verlorengeht" [9]. Alle Systemfunktionen, außer einigen globalen Funktionen, werden über Menüs aufgerufen. Der Benutzer bekommt so alle jeweils verfügbaren Funktionen im Menü angezeigt. Besonders Benutzer mit geringer Systemerfahrung - diese Zielgruppe wird in Zukunft verstärkt durch Anwendungssysteme angesprochen sein - empfinden dies als angenehm. Dies ist eine empirische Erfahrung. Theoretisch ist sie in dem Unterschied zwischen "recognition memory" und "recall memory" begründet. Die Menüauswahl findet mit Hilfe der Cursortasten statt.

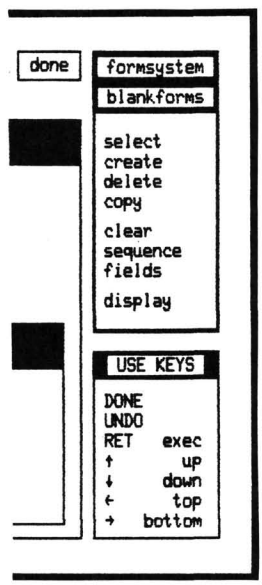


Abbildung 4:

2 Menüs (eines teilweise überdeckt) mit zugehörigem lexikalischem Help, dem "USE KEYS"-Menü, das die momentan aktiven Tasten erklärt

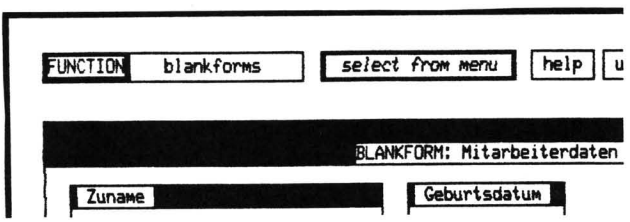
Das jeweils ausgewählte Menüelement wird mit einer anderen Farbe markiert und blinkt, solange nicht mit RETURN die entsprechende

Funktion ausgewählt wird. Das bei Eintritt in ein Menü aktive Menüelement kann vom Programm vorgegeben werden. Geschieht dies nicht, so wird das erste Menüelement aktiviert. Diese **Menüvorauswahl** beschleunigt die Auswahl über Menüs ungemein. Das ist dann der Fall, wenn das Programm weiß, welche Funktion der Benutzer als nächste auswählen will. Der Benutzer muß dann nur noch mit der RETURN-Taste bestätigen und spart sich die Auswahl des Menüelements mit den Cursortasten. In vielen Fällen kann eine solche Vorauswahl durch das Programm durchaus mit hoher Wahrscheinlichkeit getroffen werden. Wenn der Benutzer z.B. die DELETE-Funktion auswählt, ohne vorher ein zu löschendes Objekt selektiert zu haben, so kann das Programm automatisch das Menüelement für die SELECT-Funktion aktivieren, nachdem es den Benutzer auf den Fehler aufmerksam gemacht hat. Dies setzt voraus, daß das Programm Wissen über Anwendung, Kommunikationsprozesse, eventuell sogar über den Benutzer hat [5].

Die beiden Fenster in der linken oberen Ecke enthalten **Feedbacks**, wovon das erste den Namen der ausgewählten Funktion und das zweite den Systemstatus enthält, also z.B. ob das System gerade eine Eingabe erwartet oder ob eine zeitaufwendige Operation durchgeführt wird. Diese Informationen dienen dem Benutzer als Orientierung sowohl über seinen "Aufenthaltsort" im System wie auch über den momentanen Synchronisationsstatus der Kommunikation. Dadurch lassen sich solche Unklarheiten des Benutzers vermeiden, wie: "Welche Systemfunktion habe ich jetzt aufgerufen?", oder: "Erwartet das System jetzt eine Eingabe von mir, oder arbeitet es gerade?".

Abbildung 5:

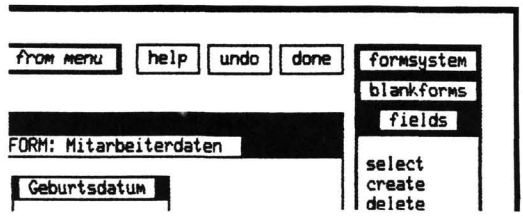
Statusinformationen



Rechts von der Statusinformation befinden sich drei "Softkeys", Repräsentanten der drei **globalen Funktionen** HELP (Aufruf des Hilfesystems), UNDO (Stornieren ausgeführter Funktionen) und DONE (Terminieren von Funktionen), die über Funktionstasten jederzeit aktivierbar sind. Da diese Funktionen sonst in jedem Menü auftauchen würden, ist es besser, sie auf Funktionstasten zu legen. Das Aktivieren einer dieser Funktionen wird durch Einfärben des entsprechenden Repräsentanten auf dem Bildschirm angezeigt. Dies ist insbesondere wichtig, wenn z.B. die HELP-Funktion vom System selbst aktiviert wird, nachdem der Benutzer Fehler gemacht hat.

Abbildung 6:

Repräsentanten der drei globalen Funktionen HELP, UNDO und DONE



Unter dem Statusfenster liegt ein Bereich, den sich mehrere Fenster durch vollständige Überlappung teilen. Dies sind Fehlermeldung, die Aufforderung zu einer Aktion und ein Eingabefenster.

Abbildung 7:

Fehler-meldung

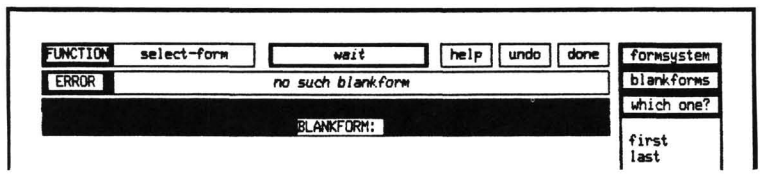


Abbildung 8:

Aktions-aufforderung

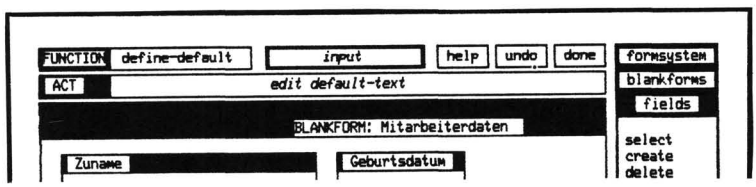
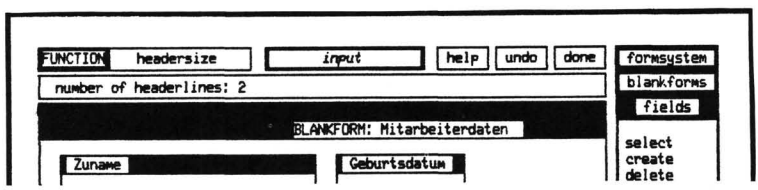


Abbildung 9:

Erfassung einer Eingabe



Der Bildschirm enthält zu einem Zeitpunkt Informationen mehrerer Kategorien. Da er gelegentlich sehr voll wird, besteht die Notwendigkeit, diese Informationen angemessen gegeneinander abzugrenzen, sowie wichtige Information (z.B. Fehlermeldungen) gegenüber weniger wichtigen (z.B. Statusmeldungen) hervorzuheben. Dies muß mittels geeigneter **Codierungsmethoden** geleistet werden.

Eine Codierungsmethode ist die Verwendung von Farbe. Da über den Einsatz von Farbe noch recht wenig aufschlußreiche Untersuchungen existieren, ist man hier in erster Linie auf Experimente angewiesen. So wurde auch bei DYNAFORM zum Teil auf Grund von Versuchen der Einsatz verschiedener Farben für verschiedene Informationskategorien

festgelegt. Das System ist allerdings so konzipiert, daß diese Farbzuordnungen einfach und schnell zu ändern sind. Durch mehrfache Änderungen wurde inzwischen ein Zustand erreicht, der sich in vielen Details mit den Richtlinien des DIN-Entwurfs zum Einsatz von Farbe bei Bildschirmarbeitsplätzen [2] deckt. Ohne hier ins Detail zu gehen, kann die Feststellung getroffen werden, daß Farbe in der Lage ist, eine komplexe Kommunikationsschnittstelle wirksam zu unterstützen.

Eine **HELP-Funktion** erlaubt es, sich in der Benutzung des Systems unterstützen zu lassen.

Die Stornierung ausgeführter Applikationsfunktionen ist über eine **UNDO-Funktion** möglich. Dies ist ein Schutz gegen Fehler und gibt dem Benutzer die Sicherheit, die Wirkung ausgeführter Funktionen wieder rückgängig machen zu können (siehe Abschnitt 6).

Die Sprache, mit der der Benutzer mit dem System kommuniziert, ist objektorientiert. Durch eine uniforme Syntax wird dem Benutzer die Bildung eines Modells vom System erleichtert (siehe Abschnitt 5).

Diese Systemeigenschaften gehören zu einer Klasse von Werkzeugen, die zur Verbesserung der Mensch-Maschine-Kommunikation dringend benötigt werden und die in ihrer Gesamtheit einen Forschungsschwerpunkt des Projekts INFORM darstellen.

5. Uniformität in der Sprache

Durch die verbreiterte Funktionalität der Benutzerschnittstelle wird die Sprache, mit der der Benutzer mit dem System kommuniziert, ebenfalls an Umfang und an Komplexität zunehmen.

Es ist ein Anliegen, dem Benutzer die Interaktion mit dem System so einfach und verständlich wie möglich zu machen. Durch die Realisierung einer weitreichenden Uniformität der Sprache wird diese funktionale Komlexität überschaubar und transparent.

Eine solche Uniformität kann auf mehreren Stufen angestrebt und errreicht werden. Eine Stufe ist die Syntax der Sprache. Eine **einheitliche Syntax**, die vom Benutzer verstanden wird, gibt ihm eine klare Vorgehensweise bei der Interaktion mit dem System vor. Dem Benutzer wird dadurch die Bildung eines Modells vom System erleichtert. Der Nutzen eines derartigen Modells vom System wurde bereits mehrfach untersucht [11] [5].

In DYNAFORM wurde daher eine einheitliche Syntax der Sprache zugrunde gelegt (Parameter 1 bis n können auch fehlen):

< Objekt > < Operator > < Parameter 1 > ... < Parameter n >

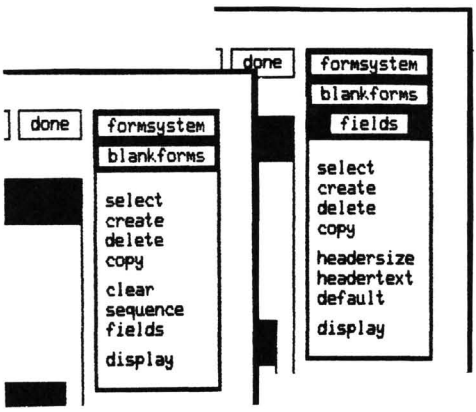
Das zu bearbeitende Objekt steht an erster Stelle. Man spricht von "**prefix-selection**" [8]. Die Auswahl des Objekts erfolgt also vor Auswahl des Operators (objektorientierter Ansatz). Diese Auswahloperation ist die SELECT-Operation. Um Objekte zu erzeugen und gleichzeitig zu selektieren, dient die CREATE-Operation. Zu bearbeitende Objekte können z.B. Formulare, Formularfelder oder Karteien sein.

Eine weitere Uniformität in der Kommunikation ist die **Einführung generischer Funktionen**, d.h. Funktionen, die eine einheitliche Semantik besitzen, auch wenn sie auf verschiedene Objekte angewendet werden. Diese Funktionen sind so grundlegend, daß sie, wenn sie einmal verstanden wurden, leicht auf verschiedene Objekte im System anwendbar sind [10]. Auch dies erleichtert dem Benutzer die Bildung eines Modells vom System wesentlich.

In DYNAFORM wurden zunächst vier generische Funktionen definiert, die z.B. sowohl auf Formulare wie auch auf Formularfelder anwendbar sind. Es sind dies die Funktionen SELECT (zur Objektauswahl), CREATE (zur Objektkreation), DELETE (zum Löschen von Objekten) und COPY (zum Kopieren von Objekten):

Abbildung 10:

Verschiedene Objektbereiche umfassen dieselben generischen Funktionen



Eine wichtige Uniformität im System stellt die Verfügbarkeit der drei **globalen Funktionen HELP, UNDO und DONE** dar. Diese Funktionen sind jederzeit in jeder Umgebung des Systems anwendbar. Sie bilden eine Art Orthogonalität zum restlichen System. Um den einfachsten Zugang zu ihnen herzustellen, wurden sie auf Funktionstasten gelegt.

6. Stornierung ausgeführter Funktionen

Es gibt eine Fülle von Gründen, warum in der Benutzung von Anwendungssystemen Aktionen initiiert werden, die zu unerwünschten Systemzuständen führen:

- der Benutzer hat ein unzulängliches Modell vom System
- von ihrer Natur her inkrementelle Designprozesse führen gelegentlich zu unbefriedigenden Zuständen
- der Benutzer arbeitet unkonzentriert
- Fehler zeigen sich oft erst nach mehreren Aktionen
- der Benutzer weiß nicht genau, was er will

Diese Umstände - und sicher lassen sich noch weitere finden - sind nicht völlig auszuschalten. In manchen Fällen sind sie sogar erwünscht (z.B. Designprozesse). Dort stellen sie einen wichtigen Teil der Arbeitsmethodik[3]dar. Es ist also von großem Nutzen, eine ausgeführte Systemaktion wieder rückgängig machen zu können.

Für DYNAFORM wurde ein solches UNDO konzipiert und weitgehend realisiert. Jede ausgeführte Funktion kann wieder rückgängig gemacht werden, allerdings streng rückwärts sequentiell gemäß ihrer Ausführungsreihenfolge. Da die Funktionen in DYNAFORM in einer hierarchischen Ordnung stehen, können auch gleichzeitig ganze Funktionssequenzen niedriger Stufe durch das UNDO der übergeordneten Funktion storniert werden.

Die Existenz eines UNDO erlaubt eine explorative Arbeitsweise, d.h. der Benutzer kann bedenkenlos unbekannte Funktionen oder Funktionen, deren Wirkung nicht sicher abgeschätzt werden kann, ausführen. Entspricht das Ergebnis nicht den Vorstellungen, wird durch ein UNDO wieder ein früherer Systemzustand erreichbar. Anwendungssysteme lassen sich so durch den Benutzer risikolos in ihrer ganzen Tiefe ausloten. Dies ist ein wesentlicher Beitrag zur Konvivialität von Anwendungssystemen.

7. Schlußbemerkungen

Die erhöhte Funktionalität von Anwendungssystemen zwingt dazu, die Benutzerschnittstelle in den Mittelpunkt des Systemdesigns stellen. Es sind leistungsfähigere Techniken der Mensch-Maschine-Kommunikation zu entwickeln. Diese Techniken werden sich nicht allein am grünen Tisch planen lassen. **Man ist auf prototypische Systeme angewiesen, die die Angemessenheit der Interaktionstechniken und Visualisierungsmethoden nachzuweisen haben.**

Wenn seither die Qualität eines Anwendungssystems an seinem Leistungsumfang gemessen wurde, so wird es in Zukunft an seiner **Kommunikationsfähigkeit mit dem Menschen** bewertet werden müssen. Nachdem wir inzwischen einen hohen Grad der maschinellen Effizienz erreicht haben, ist es angebracht, die kognitive Effizienz des Menschen durch eine adäquate Kommunikation mit der Maschine zu steigern.

Literatur

[1] J. Bauer, H.-D. Böcker, F. Fabian, G. Fischer, R. Gunzenhäuser, C. Rathke: "Projekt INFORM 83: Wissensbasierte Systeme zur Verbesserung der Mensch-Maschine-Kommunikation", Projektantrag an das Bundesministerium für Forschung und Technologie, 1982

[2] Deutsches Institut für Normung: Entwurf DIN 66234 Bildschirmarbeitsplätze, Teil 5, Verwendung von Farbe

[3] G. Fischer, H.-D. Böcker: "The nature of design processes and how computer systems can support them", in Proceedings of the European Conference on Intergrated Interactive Computer Systems (ECICS 82), Stresa, Italien, September 1982

[4] G. Fischer: "Computer als konviviale Werkzeuge", Proceedings der Jahrestagung der Gesellschaft für Informatik, München, Oktober 1981, Springer Verlag, pp. 409-417

[5] G. Fischer: "Mensch-Maschine Kommunikation: Theorien und Systeme", MMK-Memo, Institut für Informatik, Universität Stuttgart, 1982

[6] J.K. Foderaro: "The FRANZ LISP Manual", University of California, 1980

[7] M. Herczeg: "DYNAFORM - Ein interaktives Formularsystem zum Aufbau und zur Bearbeitung von Datenbasen", Diplomarbeit Nr. 212, Institut für Informatik, Universität Stuttgart, 1982

[8] U.M. Newman, R.F. Sproull: "Principles of Interactive Computer-Graphics", 2nd Ed., McGraw-Hill Inc., 1981

[9] G. Robertson, D. McCracken, A. Newell: "The ZOG Approach to Man-Machine Communication", Int. J. of Man-Machine Studies, No. 14, pp. 461-488, 1981

[10] D.C. Smith, C. Irby, R. Kimball, B. Verplank: "Designing the Star User Interface", BYTE, April 1982, pp. 242-280

[11] R.M. Young: "Users' Model of Hand-Held Calculators", Int. J. of Man-Machine Studies, No. 15, 1981

Anschrift des Autors: Michael Herczeg
Institut für Informatik
Universität Stuttgart
Herdweg 51
D-7000 Stuttgart 1