

5 Interaktionsformen

Ein- und Ausgabevorgänge sind in einem Mensch-Computer-Dialog nicht unabhängig voneinander. Bestimmte *Eingaben* bewirken in einem bestimmten Systemzustand bestimmte *Ausgaben*. Häufig verwendete Ein-/Ausgabemuster werden durch sogenannte *Interaktionsformen* beschrieben. Eine solche Interaktionsform besteht aus festgelegten Ein- und Ausgabevorgängen in bestimmten Abfolgen in einer typischen Präsentationsform.

Alle im Folgenden beschriebenen Interaktionsformen basieren auf Informationsdarstellungen, wie sie in Kapitel 6 beschrieben werden. Neben diesen Formen der Informationsdarstellung (Ausgabetechniken) werden für Interaktionsformen auch Techniken zur Informations-eingabe benötigt. Die wichtigsten Eingabetechniken werden in Kapitel 7 beschrieben. Interaktionsformen sind in ihren Ausprägungen stark von der verwendeten Computerperipherie zur Ein- und Ausgabe abhängig.

In diesem Kapitel werden Interaktionsformen definiert, klassifiziert und die wichtigsten davon beschrieben. Sie dienen dem Interaktionsdesigner als *Bausteine* oder *Muster* für die Entwicklung vielfältigster, interaktiver Systeme. Im Folgenden soll jedoch von einer technischen oder gar produktorientierten Beschreibung der Interaktionsformen abgesehen werden. Stattdessen werden Interaktionsformen so beschrieben, dass sie auf die unterschiedlichsten Realisierungen abgebildet werden können. In Beispielen werden dazu konkrete Ausprägungen gezeigt, die nicht definierenden oder einschränkenden, sondern ausschließlich illustrativen Charakter haben.

5.1 Definitionen

Für die weiteren Betrachtungen im Interaktionsdesign sollen Interaktionsformen folgendermaßen definiert werden:

Interaktionsformen sind stereotype Abläufe zur Ein- und Ausgabe von Information, die Eingabe- mit Ausgabetechniken in definierten Abfolgebedingungen verknüpfen.

Zwei grundlegende Klassen von Interaktionsformen lassen sich hierbei unterscheiden:

1. *Deskriptive Interaktionsformen* auf der Grundlage *sprachlicher Beschreibungen*; sie korrespondieren in hohem Maße mit Systemparadigmen, die den Computer als Kommunikationspartner präsentieren.
2. *Deiktische Interaktionsformen* auf der Grundlage von *Zeige- und Auswahlhandlungen*; sie bilden die Grundlage für die Verwendung des Computers als Handlungsraum oder Medium.

Aus diesen beiden grundlegenden Klassen von Interaktionsformen lassen sich vielfältige, gemischte Interaktionsformen zusammensetzen. Wir wollen sie *hybride Interaktionsformen* nennen.

Höherwertige Interaktionsformen, wie zum Beispiel die genannten hybriden Interaktionsformen, lassen sich durch Aggregation aus einfacheren Interaktionsformen bilden. Auf diese Weise bilden Interaktionsformen oft baum- oder netzartige Strukturen, deren Blätter aus Elementen der *Informationsausgabe (Ausgabezeichen)* und *Informationseingabe (Eingabezeichen)* gebildet werden und deren Knoten selbst wieder Interaktionsformen sind.

Bei der Nutzung von Interaktionsformen können Eingabezeichen durch *Selektion* aus Ausgabezeichen gebildet werden. Auf diese Weise ergeben sich enge Verknüpfungen zwischen Systemausgaben und Benutzereingaben, die geradezu bestimmend für effiziente Interaktionsformen sind. Das Angebot an Ausgabezeichen für Eingaben kann für den Benutzer *handlungsleitend* und damit wesentlich *unterstützend* sein.

5.2 Deskriptive Interaktionsformen

Für kommunikationsorientierte Dialoge benötigen wir Interaktionsformen, die eine explizite Sprache verwenden. Dafür eignen sich

- *Symbole (Zeichen)*,
- *formale Sprachen* und
- *natürliche Sprachen*.

Das Hauptproblem der Verwendung deskriptiver Interaktionsformen ist die notwendige erhebliche Gedächtnisleistung bei der Konstruktion der *Eingaben*. Benötigt wird hier vor allem das sogenannte *Recall-Memory*, das *freie Erinnern*. Dieses Problem kann durch diverse Unterstützungsfunktionen gemildert, jedoch nie völlig beseitigt werden.

5.2.1 Symbole (Zeichen)

Symbole repräsentieren beliebige Objekte, Operatoren (Funktionen) oder auch ganze Operationen (Kommandos, Befehle). Symbole verbergen dabei jegliche *Komplexität* der repräsentierten Elemente. Dies ist einerseits ein Vorteil, andererseits fehlt oft der sichtbare Zusammenhang zwischen dem Symbol und dem, was es repräsentiert. Dadurch fällt es meist schwer, von einem bestimmten Objekt oder Operator auf dessen vereinbartes Symbol und umgekehrt zu schließen.

Semiotisch gesehen sind solche Symbole Zeichen, die einen meist sichtbaren Zeichenträger (z.B. Zeichen auf dem Bildschirm) und ein Referenzobjekt (das repräsentierte Objekt oder den Operator) besitzen sowie eine Bedeutung, die sich aus dem Nutzungskontext ergibt und das Wissen über das Symbol darstellt.

Die Verwendung von Symbolen ist eine sehr effiziente Interaktionsform, die sich für häufig benötigte Interaktionen anbietet.

Die *Vorteile* von Symbolen sind:

- Symbole sind Grundlage aller Sprachen und stellen damit ein natürliches Grundkonzept dar;
- effiziente Repräsentationen beliebig komplexer Objekte, Operatoren oder kompletter Operationen sind möglich;
- einfache Ein- und Ausgabe.

Die *Nachteile* von Symbolen sind:

- oft existiert kein ersichtlicher Zusammenhang zwischen Symbol und repräsentiertem Objekt wodurch eine hohe Gedächtnisbelastung entsteht;
- Eigenschaften, Zustand und Struktur von repräsentierten Objekten oder Operatoren sind nicht sichtbar;
- Symbole basieren oft auf kulturspezifischen Konventionen und sind dadurch benutzerabhängig.

Zur Eingabe von Symbolen dient bei Computersystemen im Allgemeinen die Tastatur. Beliebige Tasten oder Tastenfolgen können als Symbole vereinbart werden. Häufig werden dabei auch neutral beschriftete *Funktionstasten* (z.B. F1 bis F12 oder PF1 bis PF15) verwendet (siehe Abbildung 43).



Abbildung 43: Neutral beschriftete Funktionstasten auf einer Standardtastatur

5.2.2 Formale Sprachen

Formale Sprachen erweitern die Verwendung von Symbolen zu Sprachen. Mittels einer *Grammatik* wird auf Grundlage der Symbole eine Sprache definiert. Die sogenannte *Semantik* legt in Form von Konventionen die Bedeutung der sprachlichen Konstrukte fest.

Formale Sprachen können sehr unterschiedlich aufgebaut sein. Eine bekannte Form wird bei *Kommandosprachen* eingesetzt, bei denen meist fast natürlichsprachige Textstücke die Verwendung der sonst abstrakten Sprache vereinfachen. Der Aufbau formaler Sprachen orientiert sich oft an natürlichen Sprachen und erleichtert so deren Verwendung. Die Überinterpretation dieses Bezugs zur natürlichen Sprache führt allerdings leicht zu Problemen. *Unerfahrene Benutzer* verlassen oft den meist eng festgelegten Bereich der formalen Sprache und erwarten ein natürlichsprachiges System. Eine besondere Form von Kommandosprachen sind *Job-Control-Sprachen* (*Betriebssystemanfragsprachen*) wie DOS-Kommandos für Windows oder die verschiedenen Shells für UNIX.

Die wichtigsten formalen Sprachen für Computersysteme sind *Programmiersprachen*, die es in unterschiedlichster Ausprägung und Mächtigkeit gibt. Programmiersprachen erlauben die Beschreibung komplexer sequentieller, iterativer, bedingter und paralleler Abfolgen von *Aktionen* und sind dadurch sehr mächtige *Interaktionsformen* für praktisch alle Problemstellungen, die heute mit Computern zu lösen sind. Für unerfahrene Benutzer sind sie jedoch nicht einsetzbar und entziehen sich daher der Mehrzahl der Computernutzer. Als Alternative zu textuellen Programmiersprachen wird daher seit vielen Jahren versucht, auf dem Wege *graphischer oder visueller Programmiersprachen* diese Lücke zu schließen. Leider weisen die existierenden Beispiele solcher Sprachen noch viele ungelöste Schwachstellen auf. *Visuelle Programmierung* ist vor allem dann gut praktikabel, wenn die Möglichkeiten der graphischen Programmiersprache stark eingeschränkt sind. Dies gilt beispielsweise für Datenbank-anfragesprachen. Hier gibt es schon seit vielen Jahren einfache und nützliche formularbasierte Sprachen wie Query-by-Example (Zloof, 1975) sowie neuere Abwandlungen davon.

Die *Vorteile* formaler Sprachen sind:

- effiziente Ein- und Ausgabe,
- Formulierbarkeit komplexer Abläufe,

- Formulierbarkeit komplexer Datenstrukturen und Sachverhalte,
- einfache, maschinelle Spracherkennung möglich.

Die *Nachteile* formaler Sprachen sind:

- beträchtlicher Lernaufwand,
- geringfügige, syntaktische Unterschiede können große semantische Unterschiede bewirken,
- formale Sprachen sind für unerfahrene Benutzer und Gelegenheitsbenutzer und damit für die Mehrheit der Computernutzer ungünstig.

Formale Sprachen sind für die Ausnutzung des großen Potenzials, das mit Computersystemen verbunden ist, außerordentlich wichtige Hilfsmittel. Oft lohnt sich der Aufwand zum Erlernen einer anwendungsgerecht konzipierten, formalen Sprache durch die später langfristige, intensive und effiziente Nutzung eines Systems.

5.2.3 Natürliche Sprache

Seit Beginn der Computerentwicklung versucht man Systeme zu entwickeln, mit denen sich Benutzer in ihrer gewohnten *Umgangs- oder Fachsprache* unterhalten können. Natürliche Sprache wird oft als die beste Interaktionsform für benutzerfreundliche Systeme angesehen.

Hilfe- und Auskunftssysteme für *Gelegenheitsbenutzer* bilden einen besonders interessanten Anwendungsbereich für natürliche Sprache. Die Verwendung natürlicher Sprache erspart den Benutzern in einer Situation, in der sie Hilfe benötigen, die Verwendung einer fehlerträchtigen anderen Interaktionsform.

Da die Realisierung natürlichsprachiger Systeme sehr aufwendig und in vielen Aspekten begrenzt ist, sollte die Notwendigkeit einer solchen Interaktionsform und der entstehende Entwicklungsaufwand sehr genau untersucht werden.

Die *Vorteile* natürlicher Sprachen sind:

- natürliche Sprachen sind für *unerfahrene Benutzer* einfach anwendbar;
- natürliche Sprachen sind mächtig genug, um praktisch jede Information oder Anfrage zu kommunizieren;
- Benutzer haben keine Angst vor der Benutzung und fühlen sich dem System überlegen;
- es muss keine spezielle formale Sprache für die Anwendung erlernt werden.

Die *Nachteile* natürlicher Sprachen sind:

- bis heute ist nur die Verarbeitung von Teilmengen natürlicher Sprache möglich;

- nur klar abgegrenzte, kleine Gegenstandsbereiche sind als Problembereiche möglich;
- meist überschätzen die Benutzer die Leistungsfähigkeit der natürlichsprachigen Systeme;
- schwierige Sachverhalte lassen sich natürlichsprachig oft nur umständlich ausdrücken;
- es ergeben sich oft Mehrdeutigkeiten auf der Syntaktischen und Semantischen Ebene.

Die Verwendung *gesprochener Sprache* anstatt *geschriebener Sprache* vertieft die genannten Probleme noch erheblich. Der Wortschatz muss weiter eingeschränkt werden und Mehrdeutigkeiten treten bereits auf der Sensorischen und der Lexikalischen Ebene auf. Darüber hinaus ist das Arbeitsumfeld, in dem gesprochene Sprache verwendet werden soll, sehr sorgfältig zu untersuchen, da Störeinflüsse, wie beispielsweise Hintergrundgeräusche und Lärm, eine solche Lösung ausschließen können. Außerdem sind erfahrungsgemäß viele Benutzer gehemmt, wenn sie mit einem technischen System laut und deutlich sprechen sollen.

5.3 Deiktische Interaktionsformen

Bei diesen, auch *selektionsorientiert* genannten Interaktionsformen wird dem Benutzer die Möglichkeit geboten, aus einem Angebot auszuwählen. Bei diesem Auswahlvorgang wird vor allem das *Recognition-Memory (Wiedererkennungsgedächtnis)* gefordert. Der Benutzer wird durch die angebotene Auswahl davon befreit, sich an mögliche Objekte oder Operatoren erinnern zu müssen.

Wichtige deiktische Interaktionsformen sind:

- *Icons* als bildhafte, symbolische Repräsentationen,
- *Menüs* als Auswahllisten,
- *beschriftete Funktionstasten* als Tastaturmenüs,
- *metaphorische Interaktionsformen* als Abbilder physischer Aktionen und
- *Netze* als Repräsentationen relationaler Informationen.

Weitere deiktische Interaktionsformen sind vorstellbar und werden auch eingesetzt. Meist lassen sie sich als Verfeinerungen der genannten Basistypen einordnen.

5.3.1 Icons

Icons sind kleine, abstrahierte, bildliche Darstellungen mit denen Objekte, Operatoren und Operationen selektiert und aktiviert werden können (Brami, 1997; Marcus, 2003). Die abstrakten Bilder selbst nennen wir *Piktogramme*. Piktogramme sind passive, graphische Darstellungen, während Icons aktive Interaktionsformen mit einem Interaktionsverhalten sind.

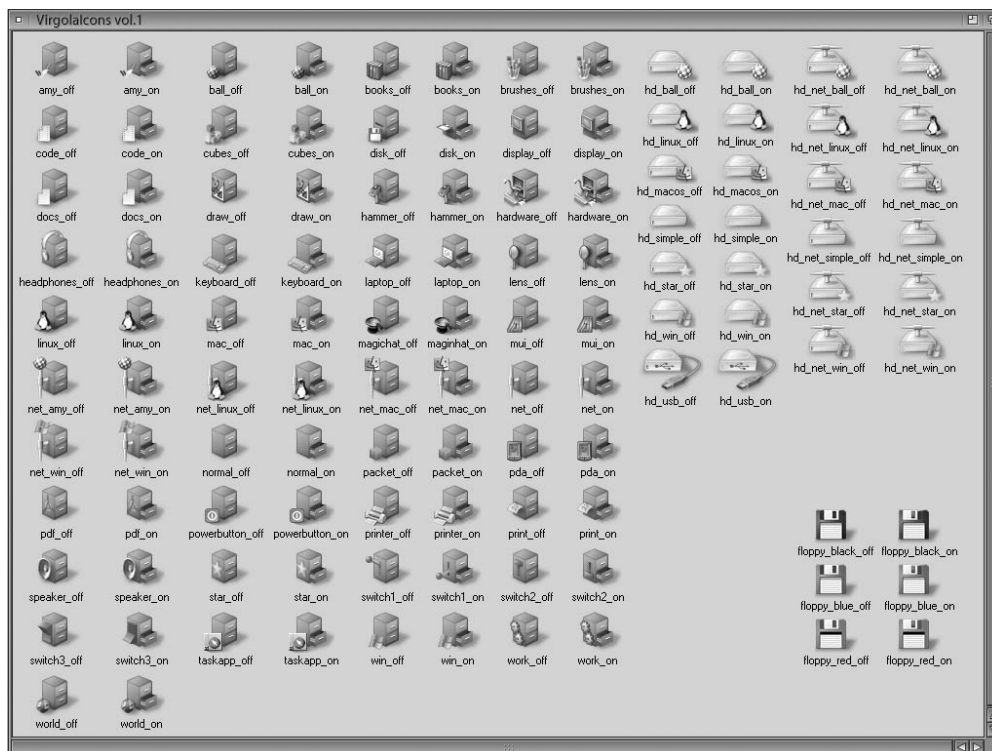


Abbildung 44: Systematisches Piktogrammsystem zur Verwendung in Icons

Die *Vorteile* von Icons sind:

- natürliches Grundkonzept;
- sichtbarer Zusammenhang zwischen Piktogramm und repräsentiertem Objekt;
- einfache Ausgabe.

Die *Nachteile* von Icons sind:

- begrenzte Dekodierbarkeit der Piktogramme durch die Benutzer;
- großer Platzbedarf auf dem Bildschirm.

Icons werden heutzutage in fast allen Computeranwendungen reichhaltig verwendet. Die verwendeten Piktogramme müssen systematisch konzipiert und gestaltet werden (siehe Abbildung 44). Dabei wird allerdings oft vergessen, dass Piktogramme selten selbsterklärend sind, und so erst einmal durch intensive Nutzung eines Systems sicher erlernt werden müssen. Dies steht der Intention einer unmittelbaren Verständlichkeit eines Dialogs direkt entgegen. Oft werden die Piktogramme deshalb durch zusätzliche Beschriftungen erklärt (siehe Abbildung 46).

Die auditive Entsprechung von Icons wurde auch *Earcons* genannt (Brewster et al., 1993). Diese liefern akustische Muster, die leicht wiedererkannt werden können und vielfältige Funktionen in multimedialen interaktiven Systemen spielen können. Die Unterstützung von Navigation (Brewster, 1998), die Lenkung von Aufmerksamkeit sowie die Bestätigung von Aktivitäten bilden dabei besonders wichtige Anwendungsmöglichkeiten.

5.3.2 Menüs

Die am häufigsten angewandte Interaktionsform ist das auf einem Bildschirm dargestellte Menü (Bildschirmmenü). Ein solches Menü ist eine gruppierte Darbietung von Objekten oder Operatoren auf dem Bildschirm, aus der ein Benutzer mittels einer Auswahl- oder Zeigehandlung auswählen kann. Es sind sehr unterschiedliche *Gestaltungsformen* von Menüs möglich (siehe dazu auch ISO 9241-14):

- Kodierung der Menüelemente mit den allgemeinen Mitteln der Informationskodierung (z.B. Text, Graphik, Farbe, Größe);
- Anordnung der Menüelemente nach den Regeln der Gestaltungsgesetze (z.B. vertikal/horizontal, mehrspaltig, mehrzeilig, gruppiert);
- Darbietung großer Auswahlmengen durch hierarchische Verkettung von Menüs (z.B. Menübäume, kaskadierte Menüs, scrollende Menüs), um keine Überladung der Übersicht und des Gedächtnisses zu verursachen²²;
- Bestimmung der Reihenfolge der Menüelemente (z.B. nach Wichtigkeit, Konventionen, anwendungsbedingter Reihenfolge, Häufigkeit der Auswahl, alphabetischer Reihenfolge);
- Art des Erscheinungszeitpunkts und des Erscheinungsorts (z.B. stationäre Menüs, Pop-Up-Menüs, Pull-Down-Menüs, Tear-Off-Menüs);
- Auswahlart als Einfachauswahl (single-choice) oder Mehrfachauswahl (multiple-choice);

²² Mit Bezug auf die bedeutende Arbeit des Psychologen Miller *The Magical Number Seven, Plus or Minus Two* (1956) wird immer wieder darauf hingewiesen, dass Menüs nicht mehr als etwa 7 Einträge besitzen sollen.

- Art der Zeigehandlung in Abhängigkeit von der verwendeten Ein- und Ausgabehardware (z.B. Auswahl mittels Finger, Maus oder Cursortasten).

Als Beispiel für die Vielgestaltigkeit von Menüs und Menüelementen siehe Abbildung 45.

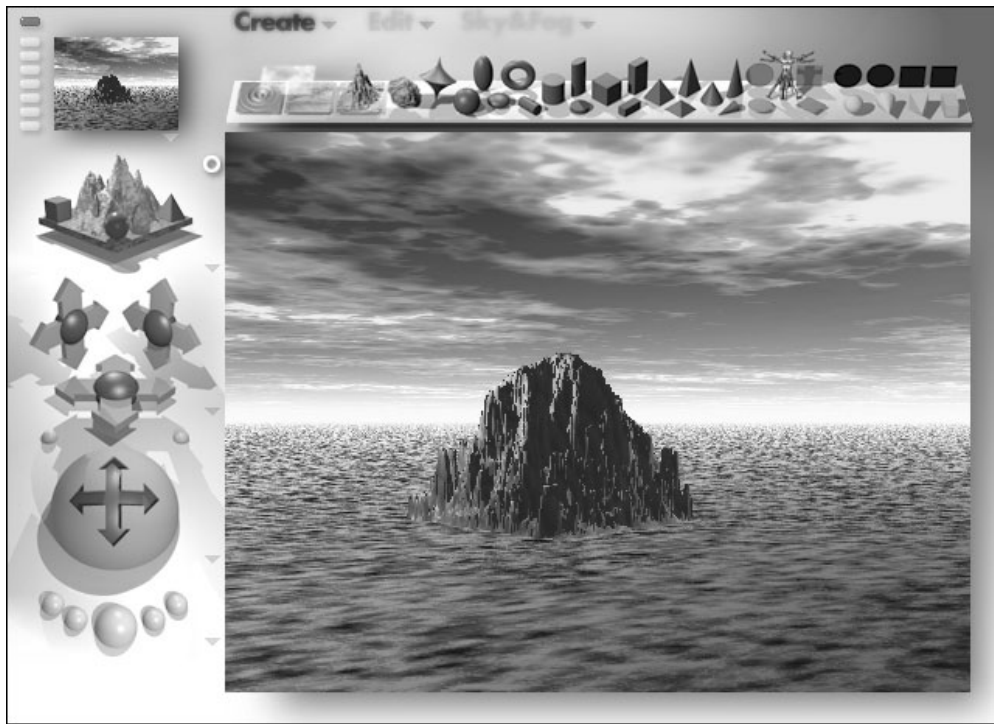


Abbildung 45: Vielgestaltige Menüs beim Graphiksystem Bryce von Metacore²³

Menüs stellen eine natürliche und intuitive Form der Interaktion dar, die schnell verstanden und automatisiert wird und daher vor allem gut für *unerfahrene Benutzer* oder für *Gelegenheitsbenutzer* verwendbar ist. Durch die richtige Ausnutzung der vielfältigen Gestaltungsdimensionen lassen sich anwendungsgerechte Lösungen schaffen. Bedingt durch den recht hohen Interaktions- und Zeitaufwand für die Darstellung und die nachfolgende Auswahlhandlung ist diese Interaktionsform für zeitkritische Anwendungen oft ungünstig.

Die *Vorteile* von Menüs sind:

- sie sind leicht erlernbar („natürliche“ Interaktionsform);
- Menüelemente wirken als Erinnerungshilfe;

²³ Inzwischen Produkt von Corel.

- es sind kaum syntaktische Eingabefehler möglich;
- sie sind für eine Vielzahl von Anwendungen verwendbar (hohe Anwendungsneutralität).

Die *Nachteile* von Menüs sind:

- sie benötigen viel Bildschirmplatz (siehe Abbildung 46);
- die Auswahl ist zeitaufwendig;
- sie sind nur für kleine Auswahlmengen gut geeignet;
- die Parametereingabe für Operationen ist schwierig.



Abbildung 46: Icons im Menü der Systemfunktionen eines Apple Macintosh

Menüs bilden wahrscheinlich die am häufigsten verwendete Interaktionsform. Sie tritt in allen genannten Ausprägungen auf. Insofern muss darauf geachtet werden, Menüs auch in Varianten zu erkennen und zu konzipieren, die nicht als einfache, vertikale Auswahllisten erscheinen. Gerade durch die Verwendung multimedialer Computersysteme können Menüs sehr vielfältig erscheinen und in die unterschiedlichsten Anwendungen unauffällig integriert werden.

5.3.3 Beschriftete Funktionstasten

Eine weitere Art deiktischer Interaktionsform sind *Funktionstasten*, die eine eindeutig anwendungsbezogene Beschriftung besitzen. Diese effiziente Methode wird bei Standardtastaturen sowie bei aufwendigen Spezialesystemen (z.B. CAD²⁴-Systeme, Videoschnittsysteme) angewandt, bei denen eine große Zahl von Operatoren (Funktionen) an spezielle Tastaturen gebunden werden.

Im Unterschied zu symbolisch beschrifteten Funktionstasten (z.B. nur mit Buchstaben- und Zahlenkombinationen beschriftet) bilden diese Funktionstasten ein typisches Menü. Der Benutzer muss sich nicht an die Belegung erinnern (siehe Abbildung 47).



Abbildung 47: Beschriftete Funktionstasten auf einer Standardtastatur

Die *Vorteile* beschrifteter Funktionstasten sind:

- sie sind leicht erlernbar („natürliche“ Dialogform);
- die Beschriftungen der Tasten wirken als Erinnerungshilfen;
- es sind praktisch keine syntaktischen Eingabefehler möglich;
- unabhängig vom Dialog- und Bildschirmzustand ist immer eine Auswahl möglich;
- es ist eine sehr schnelle Auswahlmöglichkeit gegeben;
- der Benutzer automatisiert die Aktion schnell;
- sie sind für eine Vielzahl von Anwendungen verwendbar (hohe Anwendungsneutralität).

²⁴ CAD = Computer-Aided Design

Die *Nachteile* beschrifteter Funktionstasten sind:

- sie sind wegen der Begrenzung der Tastatur nur für kleine Auswahlmengen geeignet;
- es ist keine Parametereingabe für komplexere Operationen möglich;
- die Funktionstasten erhöhen die Komplexität der Tastatur;
- es können sich gefährliche Automatismen entwickeln;
- sie sind tastaturabhängig und damit oft schlecht auf verschiedenen Rechnersystemen realisierbar.

5.3.4 Metaphorische Interaktionsformen

Grundsätzlich kann es sehr hilfreich sein, Benutzungsschnittstellen in Anlehnung an die mentalen Modelle (Carroll & Olson, 1988; Herczeg, 2005), also die Erfahrungswelt der Benutzer, zu gestalten (Carroll et al., 1988). Mit sogenannten *metaphorischen Interaktionsformen* wird auf dem Bildschirm die physikalische Arbeitsumgebung eines Anwendungsbereichs nachgebildet, man könnte auch sagen, verbildlicht²⁵. Die Arbeitsobjekte und die Operatoren (Funktionen, Werkzeuge) werden meist als aktive *Icons* oder passive *Piktogramme* dargestellt. Im Falle von sogenannten Desktop-Systemen werden Objekte einer Büroumgebung, wie beispielsweise Dokumente, Ordner, Schränke oder ein Papierkorb, dargestellt. Mit Hilfe eines Zeigeinstruments (meist Maus) können diese Objekte dann ausgewählt, bewegt, gruppiert oder mit speziellen Editoren bearbeitet werden. Die Auswirkungen auf diese Objekte sind sofort sichtbar.

Für die typischen Aktivitäten in einem solchen handlungsorientierten System benötigt man je nach Anwendung bestimmte Grundbausteine metaphorischer Interaktion. Wichtige Basisinteraktionsformen hierbei sind:

- Selektieren von Objekten und Objektmengen (Selektionen),
- Verschieben von Selektionen (*Drag&Drop*),
- Verschieben von Selektionen in den Abfalleimer (Löschen),
- Ausschneiden von Selektionen,
- Kopieren von Selektionen,
- Ausschneiden und Einfügen von Selektionen (*Cut&Paste*) sowie
- Kopieren und Einfügen von Selektionen (*Copy&Paste*).

²⁵ Metaphorik: Verbildlichung.

Derartige Interaktionsformen sind die Grundlage *direkt manipulativer Systeme* (siehe Kapitel 3.3.3). Die ständige Sichtbarkeit des Systemzustandes in Form seiner Arbeitsobjekte nennt man *WYSIWYG-Prinzip* (*What You See Is What You Get*). Die auf dem Bildschirm dargestellten Objekte müssen für den Benutzer unmittelbar verständlich sein und aktuelle Bezüge zur Anwendung herstellen. Werden sie auf dem Bildschirm manipuliert, müssen die dadurch repräsentierten Arbeitsobjekte mitverändert werden. Das Kopieren eines Dokuments auf dem Bildschirm muss also das Kopieren der entsprechenden, internen Informationsstruktur bewirken.

Das erste kommerzielle Desktop-System dieser Art war der *XEROX Star* (Seybold, 1981; Smith et al., 1982; siehe Abbildung 39). Es war der Vorgänger vieler Bedienoberflächen wie wir sie heute vor allem bei PCs²⁶ verwenden.

Neben der Nachbildung von Büros mittels der *Desktop-Metapher* werden auch technische Labors nachgebildet (Herczeg, 1988; siehe Abbildung 40). Dabei wird ausgiebig von graphischen Darstellungs- und Interaktionsformen, z.B. in Form von *Diagrammen* und *Anzeigeeinstrumenten*, Gebrauch gemacht.

Wichtige *Gestaltungsdimensionen* metaphorischer Interaktionsformen sind:

- abstrahierte (symbolische) oder mehr realitätsnahe (metaphorische) Darstellungen der Arbeitsobjekte und Werkzeuge;
- allgemeine (generische) Operatoren (z.B. Verschieben, Kopieren, Löschen) oder objektspezifische Operatoren (z.B. Dokument Verschieben, Dokument Kopieren, Ordner Löschen).

Metaphorische Interaktionsformen bieten sich vor allem bei *objektorientierten Anwendungen* an. Entsprechende Systeme nennen wir *Metaphorische Systeme* (siehe Abschnitt 4.4). In solchen Systemen werden Objekte bewegt, verändert, verbunden und gelöscht. Diese Aktivitäten lassen sich recht gut als Interaktionsformen auf Bildschirm und Zeigeeinstrument abbilden. Sobald jedoch das Bearbeiten umfangreicher Objektmengen, iterative Prozesse oder Suchvorgänge wichtig wird, muss die Metapher durchbrochen werden. Benutzer haben auch häufig überzogene Erwartungen bezüglich der Abbildungstreue metaphorischer Interaktionen. Zur Vermeidung von Fehleinschätzungen und Enttäuschungen muss dem Benutzer ein klares Modell vom System vermittelt werden.

Die *Vorteile* metaphorischer Interaktionsformen sind:

- einfaches, intuitives Erlernen der Systeme;
- Wissenstransfer aus bekannten Arbeitsbereichen;

²⁶ Die Bedeutung von besonderen Benutzungsschnittstellen für Personal Computer und Medien wurde bereits von Adele Goldberg und Alan Kay in den 70er Jahren diskutiert (Kay & Goldberg, 1976; Kay, 1977).

- hohe Akzeptanz bei Benutzern.

Die *Nachteile* metaphorischer Interaktionsformen sind:

- aufwendig bei der Bearbeitung größerer Objektmengen;
- Beschreibung von Kontrollstrukturen nicht möglich (z.B. bedingte Anweisungen, Iteration, Suche);
- Benutzer überschätzen die Systeme durch überzogenen Wissenstransfer von der physischen Welt in die metaphorischen Interaktionsformen.

5.3.5 Netze

Sollen *Beziehungen (Relationen)* zwischen Arbeitsobjekten dargestellt und manipuliert werden, können diese Beziehungen als Netze dargeboten werden (siehe Abbildung 48). Auch auf diese Weise erhält man eine besondere deiktische Interaktionsform. Insbesondere im Bereich des Entwurfs oder der Analyse komplexer Systeme mit miteinander in Beziehung stehenden Objekten ist diese Interaktionsform hilfreich.

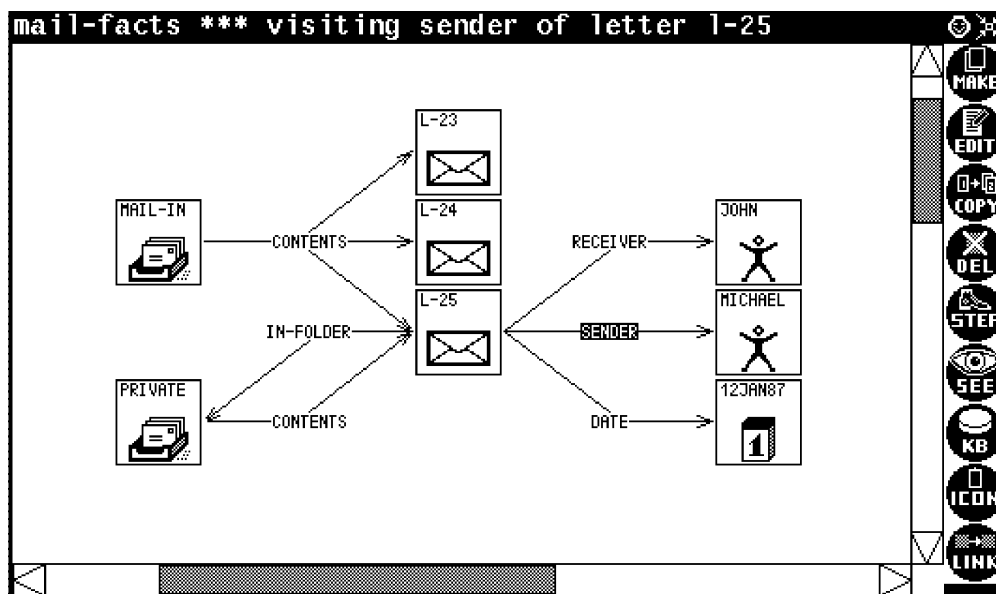


Abbildung 48: Beispiel einer 2-D-Netzdarstellung im System ZOO (Riekert, 1988)

Durch die Verfügbarkeit großer hochauflösender Bildschirme mit Zeigeeinstrumenten ist die Darstellung und Manipulation von Netzen auf Bildschirmen in vielen Anwendungsbereichen in adäquater Art und Weise realisierbar und nutzbar.

Die *Gestaltungsmöglichkeiten* bei Netzdarstellungen sind vielfältig:

- 2-D- oder 3-D-Darstellungen (vgl. Abbildung 48 und Abbildung 49),
- vielfältige Darstellungen von Netzknoten (z.B. Texte, Piktogramme),
- vielfältige Darstellungen von Kanten (z.B. Linientypen, Beschriftungen, Pfeile),
- strukturelle Spezialformen von Netzen (z.B. Bäume),
- generiertes oder manuelles Layout des Netzes (siehe Herman et. al, 2000),
- freie oder gerasterte Positionen der Netzknoten,
- Filter für Objekte und Relationen.

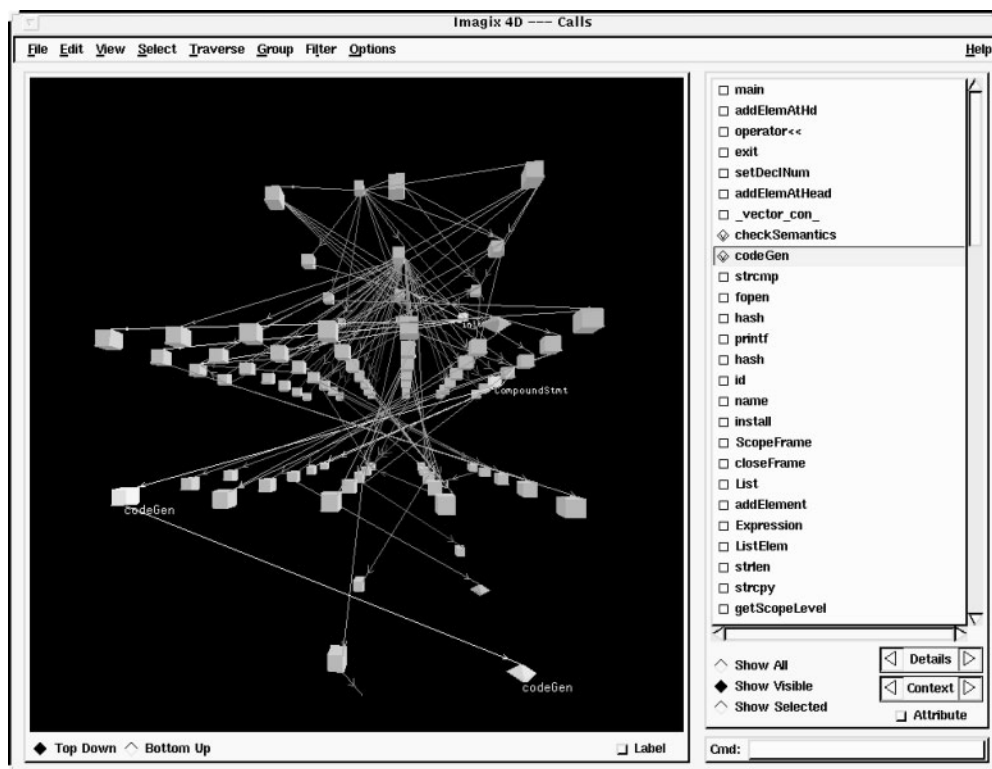


Abbildung 49: 3-D-Netzdarstellung eines komplexen Programmsystems bei Imagix 4D

Die *Vorteile* von Netzdarstellungen sind:

- bei wenigen Objekten und Relationen sind sie übersichtlicher als textuelle Darstellungen;
- die direkte Manipulation der Netze und damit der dahinter liegenden, relationalen Strukturen ist auf natürliche Weise möglich.

Die *Nachteile* von Netzdarstellungen sind:

- sie benötigen viel Bildschirmplatz;
- ein angemessenes, generiertes Layout ist schwierig und zeitaufwendig; Überschneidungen von Kanten sind bei vielen Relationen oft nicht zu vermeiden;
- sie erweisen sich bei vielen Objekten oder Relationen oft als unübersichtlich.

5.4 Hybride Interaktionsformen

Deskriptive und deiktische Interaktionsformen werden oft zu Mischformen, sogenannten *hybriden Interaktionsformen* verbunden. So lassen sich die Vorzüge beider Interaktionsformen ausnutzen.

Im Folgenden werden Beispiele für gemischte Interaktionsformen beschrieben. Es sind beliebig viele Mischformen denkbar, da sie aus einer hierarchischen Zusammensetzung aus anderen Interaktionsformen bestehen. Diese baukastenartige Modellierung findet sich auch in den entsprechenden *User-Interface-Toolkits* zur Realisierung dieser Interaktionsformen wieder.

5.4.1 Formulare

Formulare bieten eine beliebige Anzahl von Eingabefeldern an. Nach Selektion (deiktisch) eines Feldes kann das Füllen (deskriptiv) oder die Bearbeitung des Feldinhaltes stattfinden.

Formulare sind zusammen mit Menüs die heutzutage wahrscheinlich am häufigsten genutzte Interaktionsform. Während Menüs eine natürliche Interaktionsform darstellen, sind Formulare aufgrund ihrer häufigen Nutzung im Geschäftsleben oder bei Organisations- und Verwaltungsprozessen eine gewohnte Interaktionsform.

Die *Gestaltungsmöglichkeiten* für Formulare sind reichhaltig:

- Form und Größe der Felder;
- Schriftarten (Fonts) der Texte in den Feldern;
- Layout bzgl. der Anordnung der Felder;
- Editiermöglichkeiten in den Feldern;

- automatische oder manuelle Feldfortschaltung nach Abschluss der Bearbeitung eines Feldinhaltes;
- Filter zur Reduktion der dargestellten Felder.

Die *Vorteile* von Formularen sind:

- sie sind eine „gewohnte“ (erlernte) Interaktionsform;
- sie sind für eine Vielzahl von Anwendungen verwendbar (hohe Anwendungsneutralität).

Die *Nachteile* von Formularen sind:

- sie benötigen viel Bildschirmplatz;
- ein angemessenes generiertes Layout ist schwierig;
- häufig sind momentan unwichtige Felder sichtbar.

Der Einsatz von Formularen ist ähnlich bedeutungsvoll wie der von Menüs. Dies liegt daran, dass Formulare für alle Aufgaben verwendbar sind, wo vorstrukturierte Information ein- oder auszugeben ist. Dies entspricht auch der Beschreibung und Darstellung von Merkmalen von Arbeitsobjekten.

5.4.2 Property- und Optionsheets

Eine Mischung aus Formularen und Menüs bilden die sogenannten Property- und Optionsheets. Sie dienen der Beschreibung von Objekten (Property sheets) und Funktionen (Optionsheets), siehe Abbildung 50.

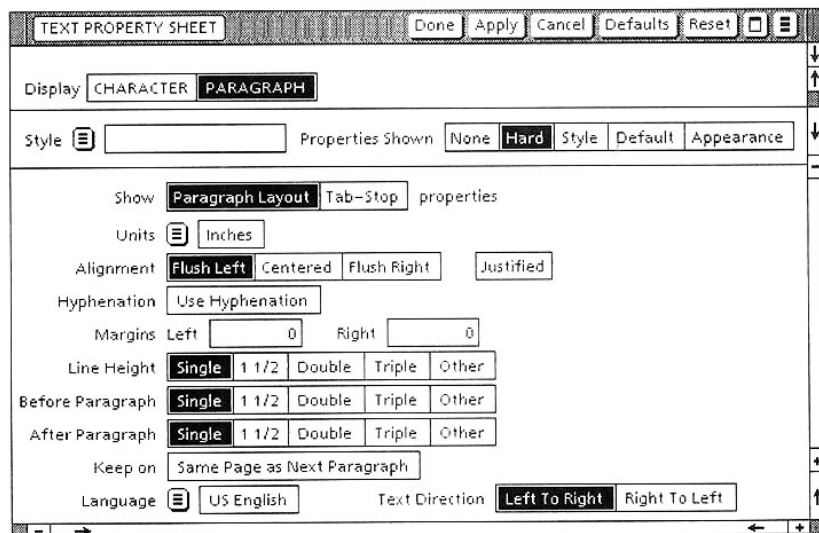


Abbildung 50: Eines der ersten Property sheets im XEROX Star

Oft finden sich auch noch Schaltflächen (Buttons) und Icons zur Ausführung von Funktionen auf Property- und Optionsheets, siehe Abbildung 51.

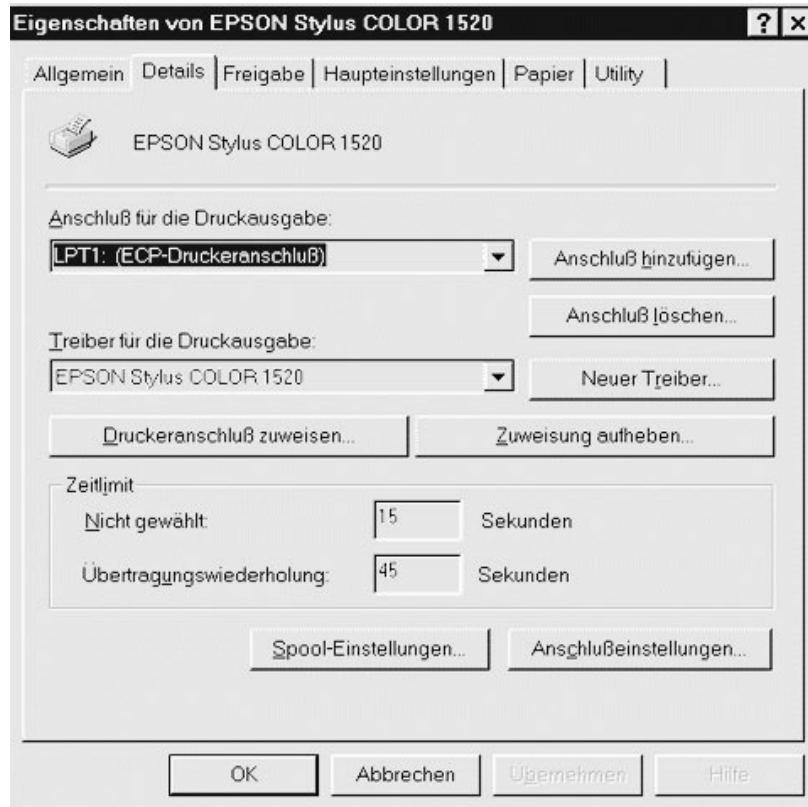


Abbildung 51: Propertysheet mit Schaltflächen (Buttons)

Die Gestaltungsmöglichkeiten und Eigenschaften leiten sich direkt aus denen von Formularen, Menüs und Icons ab.

5.4.3 Menüs mit Texteingabe

Man kann Menüs mit Texteingaben kombinieren, so dass sichn anstatt durch die Zeigehandlung (deiktisch), auch durch eine Tastatureingabe (deskriptiv) aus dem Angebot auswählen lässt. Oft werden die Menüelemente durchnummeriert oder durch Kürzel (*Accelerators*) markiert. Diese schnellen Eingaben führen dann zur Auswahl, ohne dass das Menü mit dem Zeigeelement aktiviert und ein Menüelement ausgewählt werden müsste.

Als Eingabe kann bei Textmenüs auch der eindeutig abgekürzte Text des Menüelements dienen. Diese Methode erlaubt dem Benutzer, mit den Händen ständig auf der alphanumerischen Tastatur zu bleiben. Dies ist dann vorteilhaft, wenn vor allem Text eingegeben wird.

5.5 Zusammenfassung

Verknüpft man bestimmte Formen der *Informationsausgabe* (Ausgabetechniken) mit bestimmten Formen der *Informationseingabe* (Eingabetechniken), so entstehen *Interaktionsformen*. Sie beschreiben zulässige Abläufe von Benutzereingaben und Systemausgaben. Es lassen sich zwei Hauptgruppen von Interaktionsformen unterscheiden:

- *Deskriptive Interaktionsformen*: entstehen auf der Grundlage sprachlicher Beschreibungen; Beispiele sind formale und natürliche Sprachen.
- *Deiktische Interaktionsformen*: entstehen auf der Grundlage von Selektionen mittels Zeigehandlungen; Beispiele sind Menüs, Desktop-Systeme und Netze.

Aus diesen beiden Gruppen von Interaktionsformen lassen sich durch hierarchisches Zusammensetzen nahezu beliebige, *gemischte (hybride) Interaktionsformen*, wie beispielsweise Formulare und Property sheets, bilden.