Befehl oder Dateiname nicht gefunden

Möglichkeiten und Grenzen der Kommunikation mit dem Computer

Von STEFAN RABANUS

Abstract

Mensch-Maschine-Kommunikation als die Steuerung des Computers durch den Benutzer ist heute auf drei Arten möglich: a) durch die Eingabe des Systemkodes über die Kommandozeile, b) mit Mausoperationen und c) über Systeme der automatischen Spracherkennung. Im vorliegenden Beitrag werden diese Kommunikationsformen analog zur zwischenmenschlichen Kommunikation konzeptualisiert und analysiert. Dabei zeigt sich, dass die sprachlichen Handlungsmöglichkeiten des Menschen mit zunehmender Benutzerfeundlichkeit des Systems abnehmen.

Man-Machine Interaction in controlling the computer today occurs in three different manners: (i) by entering command lines, (ii) doing mouse operations, and (iii) using systems of automatic speech recognition. In my paper these communication forms are conceptualised and analysed analogously with forms of interaction among humans. The analysis reveals that the possibility of man to act linguistically decreases as the user friendlyness of the system increases.

1 *Einführung: Mensch-Maschine-Kommunikation*

Der Computer ist in den vergangenen Jahren für die Mehrheit seiner Benutzer von einer komfortablen Schreibmaschine zu einem multimedialen Werkzeug geworden, zu dessen wichtigsten Anwendungsfeldern die Vermittlung zwischenmenschlicher Kommunikation zählt. E-Mail- und Chat-Kommunikation bilden zusammen mit dem Erstellen von Textdokumenten und dem Betrachten von WWW-Seiten die Gruppe der Anwendungen, deren sich die meisten, auch die nicht weiter an Datenverarbeitung interessierten Benutzer bedienen. Mit der Popularisierung der vor 1995 auf einen relativ engen Kreis von Spezialisten und Computerinteressierten beschränkten computervermittelten Kommunikationsformen[1] wuchs auch die Zahl der sprachwissenschaftlichen Untersuchungen dieser Formen. Dabei erwiesen sich E-Mail, Diskussionsforen und Chat als Experimentierfelder für eine neue Schriftlichkeit, die wesentliche Merkmale mit der spontanen Mündlichkeit teilt.[2] Die verstärkte Untersuchung der computervermittelten Kommunikation darf aber nicht den Eindruck erwecken, dass die Mensch-Maschine-Kommunikation heute an Bedeutung verloren hätte. Mensch-Maschine-Kommunikation fängt direkt nach dem Einschalten des Computers mit den ersten Systemmeldungen und entsprechenden Reaktionsmöglichkeiten des Benutzers (z.B. die BIOS-Einstellungen aufrufen) an. Sie geht jeder computervermittelten zwischenmenschlichen Kommunikation voraus. Bevor ein Chat beginnen oder eine E-Mail verschickt werden kann, muss ein entsprechendes Anwendungsprogramm (E-Mail-Client, WWW-Browser etc.) gestartet werden. Für die Datenverarbeitung im engeren Sinn wie für die Vielzahl der Angebote im WWW ist der Computer ohnehin alleiniger Kommunikationspartner. Kommunikation wird dabei als wechselseitige Übermittlung von Informationen durch Zeichen konzeptualisiert. Der Maschine fehlen zwar wesentliche Eigenschaften des

[1] Zum Stand der Forschung bis 1995 vgl. Rabanus 1996.
[2] Vgl. dazu den ausführlichen Überblick von Runkehl/Schlobinski/Siever 1998.

menschlichen Kommunikationspartners, dennoch kann die Mensch-Maschine-Kommunikation bis zu einem gewissen Punkt analog zur zwischenmenschlichen Kommunikation aufgefasst werden.

Ziel des vorliegenden Beitrags ist die Rekonstruktion der Mensch-Maschine-Kommunikation mit dem Beschreibungsinventar, wie es die Linguistik für die zwischenmenschliche Kommunikation entwickelt hat. Den Ausgangspunkt bildet die Beschreibung der Verhältnisse bei kommandozeilenbasierten Systemen (Abschnitt 2). Der Systemkode erlaubt die unmittelbarste Kommunikation und direkteste Steuerung des Computers. Der nicht-professionellen Anwender beherrscht den Systemkode aber meist nur unvollständig. Systeme mit graphischer Oberfläche (Abschnitt 3) und Systeme mit Spracherkennungsfunktionen (Abschnitt 4) werden deshalb als Entwicklungen mit dem Ziel verstanden, die Kommunikation für den Menschen zu erleichtern.

2 *Tastaturgesteuerte Zeicheneingabe: Kommandozeilenbasierte Systeme*

2.1 *Syntax und Semantik des Systemkodes*

Die Kommunikation mit der Maschine besteht heute vor allem darin, den Mauszeiger über eine graphische Benutzeroberfläche zu führen und Programmikons oder sonstige Symbole anzuklicken. Bis zur Einführung der Maus (erstmals 1983 beim Apple Lisa) erfolgte die Zeicheneingabe jedoch ausschließlich dadurch, über die Tastatur Kommandozeilen zu tippen.[3] Systemadministratoren ziehen die tastaturgesteuerte auch heute vielfach noch der mausgesteuerten Zeicheneingabe vor, die Fernadministration von Rechnern ist oft nur über Kommandozeilen möglich. Jedes Betriebssystem bietet dem Anwender mindestens eine Shell zur Eingabe von Kommandozeilen.[4] Bei Kommandozeilen handelt es sich um ein von einem Prompt (im Beispiel unten visualisiert durch `C:\>`) eröffnetes Feld, in das der Benutzer sprachliche Ausdrücke eines Systemkodes ein- und durch Drücken der Eingabetaste an den Rechner übergibt. Den Ausdrücken liegen in den meisten Fällen englische Wörter zugrunde. Ihre syntaktischen und semantischen Eigenschaften richten sich jedoch nicht nach ihren Ursprungswörtern, sondern sind im Systemkode davon prinzipiell unabhängig festgelegt. Die Ausdrücke sind Ausführungsbefehle, ihre Semantik besteht darin, nichtsprachliche Objekte (logische Systemeinheiten, Programme) zu repräsentieren und zu steuern. Die Befehle werden jedoch nur dann ausgeführt, wenn die syntaktischen Regeln des Systemkodes streng eingehalten werden. So bewirkt im Betriebssystem DOS von Microsoft allein die Eingabe der Kommandozeile

```
C:\> rename a.txt b.txt
```

die Umbenennung von Text `a` zu `b` – unter der Voraussetzung, dass `rename` für ein im lokalen System installiertes Programm zur Umbenennung von Dateien steht und dass die Datei `a.txt` im Arbeitsverzeichnis vorhanden ist. Obwohl der Ausdruck `rename` aus dem Englischen entlehnt ist, kann er im Systemkode nicht durch einen anderen Ausdruck mit gleicher Semantik ersetzt werden, zum Beispiel mit *give a new name*. Auch in der Syntax werden keine Abweichungen von den Regeln geduldet. In der na-

[3] Vgl. zu den Eingabekanälen Giese/Januschek 1990.

[4] Bei den Windows-Betriebssystemen sind das die *MS-DOS-Eingabeaufforderung* und die Befehlszeile hinter der Schaltfläche *Ausführen*.

türlichen Gesprächssituation sind syntaktische Irregularitäten häufig. Anakoluthe, fehlende Kongruenzen, Ellipsen und dergleichen sind typische Erscheinungen der spontan gesprochenen Sprache und werden im Gespräch nur selten als Fehler kontextualisiert:[5] Das gegenseitige Verstehen der Gesprächspartner leidet unter solchen Erscheinungen kaum, so genannte Ellipsen sind im Gegenteil oft erforderlich, um überflüssige Redundanzen zu vermeiden und die Äußerung im Sinne des Prinzips der Sprachökonomie optimal zu gestalten.[6] Das Prinzip der Sprachökonomie gilt für die kommandozeilenbasierte Kommunikation jedoch nicht. Kommandozeilenbasierte Systeme inferieren nichtkodierte Informationen nicht aus dem Kontext und sind auch sonst nicht kooperativ.[7] In der natürlichen Gesprächssituation wird jede anfangs noch so defizitär oder unsinnig anmutende bzw. syntaktisch inkorrekte Äußerung so lange interpretiert, bis ein Sinn zum Vorschein kommt. Das schließt zwar Missverständnisse und Nichtverstehen nicht prinzipiell aus, Kooperationsbereitschaft hängt aber nicht vom guten Willen der Gesprächspartner ab, sondern ist ein Grundmerkmal menschlicher Kommunikation. Wenn der Anwender in der Kommunikation mit der Maschine dagegen die in der obenstehenden Kommandozeile aus natürlichsprachlicher Sicht überflüssige (weil redundante) Endung `.txt` hinter `a` auslässt, findet der Computer die gemeinte Datei nicht und reagiert mit einer Fehlermeldung:

```
C:\> rename a b.txt
Datei nicht gefunden – a
```

Umgekehrt werden auch zusätzliche sprachliche Ausdrücke nicht toleriert. Unter Windows NT verlangt der Befehl `move`, mit dem Dateien verschoben werden können, genau zwei Ergänzungen in der folgenden syntaktischen Form: `move [Quelle] [Ziel]`. Zusätzliche Elemente wie die untenstehenden Beispiel verwendete Präposition *to* werden also nicht einfach als überflüssig 'überlesen', sondern verhindern die korrekte Ausführung des Befehls und verursachen eine Fehlermeldung:

```
C:\> move a.txt to Texte\a.txt
Syntaxfehler
```

2.2 *Pragmatik des Systemkodes*

Die pragmatischen Verhältnisse sind bei der Eingabe von Kommandozeilen wesentlich unkomplizierter als in natürlichen Gesprächen. Kommandozeilen sind immer Befehle. Es liegt nahe, sie in der Sprechaktklassifikation von Searle[8] zur direktiven Klasse zu zählen. Im Unterschied zu direktiven Sprechakten in der zwischenmenschlichen Kommunikation hat der Computer als Adressat aber nicht die Möglichkeit, einen korrekt formulierten Befehl nicht zu befolgen. Wenn die syntaktischen Regeln eingehalten werden und die Ausdrücke für vorhandene Systemeinheiten (Programme, Dateien) stehen, führt die Eingabe einer Kommandozeile unmittelbar zur Ausführung der technischen Operation. Es gibt keine Differenz zwischen der semantischen Bedeutung eines Ausdrucks (als Lexikoneintrag) und seiner Handlungsbedeutung im spezifischen Kon-

[5] Zu Kontext und Kontextualisierung vgl. Auer 1996.
[6] Vgl. dazu die Konversationsmaximen von Grice 1975, S. 45 ff.
[7] Vgl. Grice 1975, S. 45.
[8] Vgl. dazu Searle 1976, S. 10–16 und Searle/Vanderveken 1985, S. 37 f.

text,[9] denn zur Kommandozeile gibt es keinen Kontext. Zwar wird die »Kommunikationsgeschichte« von den meisten Betriebssystemen in so genannte *Log-Files* protokolliert. Kommdandozeilenbasierte Programme greifen aber nicht auf die dort gespeicherten Informationen über vorhergehende Operationen zurück. Auch die Unterscheidung von Illokution und Perlokution ist in der Mensch-Maschine-Kommunikation nicht sinnvoll. Ein durch angemessene Illokutionsindikatoren ausgedrückter direktiver Sprechakt kann nicht nicht glücken, und der Erfolg des perlokutionären Akts ist damit garantiert. Deshalb ist der Klassifikation der Kommandozeile als direktiv diejenige als deklarativ vorzuziehen. Für deklarative Sprechakte gilt nach Searle, dass unter der Voraussetzung bestimmter institutioneller und sozialer Bedingungen »successful performance guarantees that the propositional content corresponds to the world«.[10] Mit dem Vollzug deklarativer Sprechakte werden also unmittelbar außersprachliche Fakten geschaffen. Weil das Glücken des in der Kommandozeile ausgedrückten Sprechaktes aber unabhängig von der für deklarative Sprechakte notwendigen Erfüllung institutioneller oder sozialer Bedingungen ist, schlägt Schmitz vor, Befehle an den Computer als »Computive« zu bezeichnen:

> Wir wollen diese Klasse von Sprechakten Computive nennen, weil sie [...] berechneten und (unter normalen technischen Umständen) berechenbaren Routinen folgen. Deshalb sind sie sowohl in ihrer illokutionären Rolle als auch in ihrem propositionalen Gehalt eindeutig festgelegt.[11]

Auf eine Kommandozeile, die keinen ausführbaren Befehl ausdrückt, reagiert das System prinzipiell nicht bzw. mit einer Fehlermeldung. Eine Selbstverpflichtung des Anwenders wie `I will be back soon`, die nach Searle als kommissiver Sprechakt eingeordnet wird, quittiert das System mit ›Unverständnis‹:

```
[rabanus@pc0897 /]$ I will be back soon
bash: I: command not found
[rabanus@pc0897 /]$
```

Der Satz `I will be back soon` wird vom Computer (hier einer Shell mit dem Namen *bash*) als Aufruf des Programms `I`, gefolgt von den Parametern `will`, `be`, `back` und `soon` interpretiert. Ein Programm mit dem Namen `I` ist aber auf dem hier von mir verwendeten Linux-System mit dem Namen *pc0897* nicht installiert. Dem Ausdruck `I` entspricht keine logische Systemeinheit, er ist kein Element des Systemkodes und somit bedeutungsleer.

2.3 *Dialogstrukturen*

Ungeachtet dieser Einschränkungen erfolgt auch die kommandozeilenbasierte Kommunikation mit dem Computer in dialogischen Strukturen. Auf die Übergabe einer korrekten und vollständigen Kommandozeile reagiert das System unmittelbar, wenn auch oft nichtsprachlich: Die angewiesene technische Operation wird ausgeführt. Den Benutzereingaben können aber auch sprachliche Reaktionen folgen, woraus in Abhängigkeit von der Funktionsweise der aufgerufenen Programme unterschiedliche Dialogstrukturen entstehen. Auf die Frage nach der Version des laufenden Betriebssystems (Befehl `ver`

[9] Zur Unterscheidung bzw. Interaktion von abstrakter semantischer Struktur und konkreter Handlungsbedeutung vgl. z. B. Bierwisch 1980.

[10] Searle 1976, S. 13.

[11] Schmitz o. J., Abschn. 2.

unter Windows NT) gibt der Computer die gewünschte Antwort und meldet dann durch die Wiederherstellung des Prompts, dass er bereit für eine neue Eingabe ist:

```
C:\> ver
Windows NT Version 4.0
C:\>
```

Wilhelm Franke (1990) hat im Rahmen der sprechakttheoretischen Dialoggrammatik[12] ein Beschreibungsmodell für elementare Dialogstrukturen vorgelegt, das sich mit leichten Modifikationen auch für die Beschreibung von Dialogstrukturen in der kommandozeilenbasierten Mensch-Maschine-Kommunikation eignet. Als wichtigste elementare Dialogstruktur führt Franke den »Minimal-Dialog« ein.[13] Ein Minimaldialog ist eine Sequenz aus zwei oder mehr Redezügen, in denen ein Sprecher das in seinem initialen Sprechakt formulierte Handlungsziel erreicht, was seitens des Adressaten durch einen »positiven Bescheid« manifestiert wird. Im letzten Beispiel hat das System mit der Angabe der Versionsnummer den positiven Bescheid erteilt, der Minimaldialog ist erfolgreich abgeschlossen. Der im vorhergehenden Abschnitt abgedruckte von der Zeile `I will be back soon` ausgehende Dialog ist dagegen eine Sequenz aus einem initialen kommissiven Sprechakt und einem »negativen Bescheid« des Systems: Der Sprecher hat sein Handlungsziel, die in natürlichen Gesprächen erwartbare Ratifizierung seiner Selbstverpflichtung durch den Adressaten, nicht erreicht.[14]

Der positive Bescheid kann auch implizit erteilt werden. Bei der Umbenennung einer Datei mit `mv` unter Linux wird die korrekte Ausführung der Operation nur durch Wiederherstellung des Prompts (hier visualisiert durch `[rabanus@pc0897 /]$`) angezeigt:

```
[rabanus@pc0897 /]$ mv a.txt b.txt
[rabanus@pc0897 /]$
```

Die Dialogstrukturen in der kommandozeilenbasierten Kommunikation sind nicht auf zwei Züge beschränkt. Der Befehl `time` unter Windows 98 gibt dem Anwender die Möglichkeit, eine neue Uhrzeit einzugeben. Auf den initialen direktiven Sprechakt `time` reagiert das System mit einem nicht-spezifischen reaktiven Sprechakt, einer Präzisierungsfrage:[15] mit der Aufforderung, die Uhrzeit einzugeben.

```
C:\> time
Aktuelle Uhrzeit: 11:57:19,43
Neue Uhrzeit:
```

Im dritten Zug hat der Anwender drei Handlungsmöglichkeiten:[16] Er kann durch direktes Drücken der Eingabetaste auf die Zuweisung einer neuen Uhrzeit verzichten. Mit Franke vollzieht er dann einen »retraktiven Sprechakt«. Der Minimaldialog ist abgeschlossen, das System zeigt mit dem Prompt (`C:\>`) seine Bereitschaft für weitere Eingaben bzw. seine generelle Dialogbereitschaft an. Die zweite Möglichkeit besteht darin, eine neue Uhrzeit einzutippen und durch die Eingabetaste zuzuweisen. Der An-

[12] Vgl. dazu Hindelang 1994.
[13] Vgl. Franke 1990, S. 22.
[14] Zur Systematik der Handlungsmöglichkeiten im zweiten Zug vgl. Franke 1990, S. 15–25.
[15] Vgl. Franke 1980, S. 239.
[16] Zur Systematik der Handlungsmöglichkeiten im dritten und vierten Zug vgl. Franke 1990, S. 26–41.

wender führt damit einen »reinitiativen Sprechakt« durch, weil er sein initiales Handlungsziel beibehält, indem er die Präzisierungsfrage beantwortet. In diesem Fall ist die Wiederherstellung des Prompts als Ratifizierung der Uhrzeiteingabe und damit als positiver Bescheid im vierten Zug zu interpretieren, womit der Minimaldialog ebenfalls abgeschlossen ist.[17] Die dritte Möglichkeit besteht in der Eingabe eines Ausdrucks, der kein Uhrzeitformat hat und damit als »revidierender Sprechakt« aufzufassen ist. Das System reagiert darauf im vierten Zug mit einer Fehlermeldung und einem reinitiativen Sprechakt:

```
Ungültige Uhrzeit
Neue Uhrzeit:
```

Die Präzisierungsfrage wird also wiederholt. Jeder weitere revidierende Sprechakt des Anwenders zieht einen reinitiativen Sprechakt des Systems nach sich. Die daraus entstehende Schleife wird nur beendet, wenn der Anwender entweder eine Uhrzeit eingibt oder durch Drücken der Eingabetaste einen retraktiven Sprechakt ausführt.

Abb. 1[18] zeigt die hier betrachteten Minimaldialoge in der Zusammenschau. Mit den Programmen sind unterschiedliche Dialogskripten assoziiert, die nur in Einzelfällen mehr als zwei Redezüge (initialer Sprechakt – positiver bzw. negativer Bescheid) vorsehen. Allen Minimaldialogen ist gemeinsam, dass das Recht zur Eröffnung der Kommunikation mit einem initialen Sprechakt beim Anwender liegt. Unter Umständen lässt sich die Abfrage eines Anfangspasswortes beim Systemstart als eigeninitiativer initialer Sprechakt des Computers interpretieren. Wenn das System aber erst einmal läuft, zeigt ein kommandozeilenbasiertes System keine sprachliche Initiative. Der Prompt signalisiert lediglich, dass das System zur Verarbeitung des nächsten Befehls bereit ist.

Exkurs: »Computer Talk«

Die Tatsache, dass erfolgreiche Kommunikation mit der Maschine von der genauen Kenntnis des Systemkodes abhängig ist, steht im Gegensatz zur Tendenz, den Computer als einen »verstehenden« Kommunikationspartner zu begreifen. In den 60er Jahren führten die Experimente des Computerkritikers Joseph Weizenbaum mit einem Psychotherapieprogramm namens »Eliza« eine Anzahl an praktizierenden Psychiatern – gegen Weizenbaums Absichten – dazu zu glauben, die (damals noch sehr primitiven) Rechnern schnell zu »automatisierten Psychotherapeuten« weiterentwickeln zu können.[19] In der Tat wurde Eliza trotz kommunikativer Defizite von vielen Versuchspersonen als Kommunikationspartner akzeptiert, manche Versuchspersonen bauten sogar ein emotionales Verhältnis zur Maschine auf. Dieser Erfahrung aus einer Zeit, in der Computer den meisten Menschen noch vollkommen unbekannt waren, steht die so genannte Computer-Talk-Hypothese gegenüber, nach der sich Menschen in der Interaktion mit maschinellen Dialogsystemen sprachlich systematisch anders verhalten als in der natür-

[17] Im Gegensatz zu Franke, der einen positiven Bescheid zum Abschluss eines Minimaldialogs nur im zweiten Zug zulässt, muss m.E. hier auch der vierte Zug als positiver Bescheid interpretiert werden. Eine Präzisierungsfrage impliziert auch in der natürlichen Gesprächssituation nicht notwendigerweise eine Ablehnung des initialen Sprechaktes, die nur mittels eines retraktiven Sprechaktes zurückgenommen werden könnte.

[18] Die Abbildungen befinden sich im Anhang.

[19] Vgl. Weizenbaum 1990, 14 ff.

lichen Gesprächssituation.[20] Huberta Kritzenberger (1997) hat in einer Studie das Sprachverhalten von Benutzern maschineller natürlichsprachlicher Auskunftssysteme (Bahnauskunft, Bibliotheksauskunft) mit dem Verhalten kontrastiert, das dieselben Versuchspersonen in Gesprächen mit menschlichen Informanten zeigen. Dabei hat sich gezeigt, dass

> die Benutzer sich in der natürlichsprachlichen MCI [Mensch-Computer-Interaktion, StR] in systematischer Weise nicht an die Konventionen der zwischenmenschlichen Kommunikation halten"[21].

Das gilt auch für eine in *Wizard of Oz*-Experimenten untersuchte Interaktion mit einem simulierten maschinellen Informationssystem, das über uneingeschränkte Sprachkompetenz und über kooperatives Verhalten wie ein menschlicher Informant verfügt.[22] Ein typischer Anwender produziert im Dialog mit einem Bahnauskunftssystem also in der Regel keine kommissiven oder expressiven Sprechakte, auch die Form seiner Äußerungen unterscheidet sich signifikant von derjenigen in der natürlichen Gesprächssituation: Die Zahl der ungrammatischen Sätze sowie von Partikeln, Ellipsen und vergleichbaren Phänomenen ist in der Kommunikation mit der Maschine niedriger als in der zwischenmenschlichen Kommunikation.[23] Weizenbaums und Kritzenbergers Ergebnisse lassen übereinstimmend folgendermaßen interpretieren: Für die Erwartungen an die Sprachkompetenz der Maschine ist weniger die beobachtbare Performanz als vielmehr das mentale Modell des Menschen vom Computer verantwortlich.[24] Dieses Modell bildet die tatsächliche Systemleistung selten zutreffend ab und wandelt sich im Laufe der Zeiten. Die derzeitige Multimedia-Euphorie deutet darauf hin, dass die Erwartungen an die Leistungsfähigkeit des Computers heute wieder ähnlich überzogen sind wie in den 60er Jahren.

3 *Mausgesteuerte Zeicheneingabe: Systeme mit graphischen Oberflächen*

Der Systemkode ist eine formale Sprache, deren Syntax und Semantik wie eine Fremdsprache gelernt werden muss. Dazu sind die meisten heutigen Anwender nicht willens oder in der Lage. Seit Computer nicht mehr vor allem von Technikern bedient werden, bemühen sich deshalb die Systementwickler darum, die Kommunikation von der Beherrschung des Systemkodes unabhängig zu machen. Kommandozeilenbasierte Systeme wurden weitgehend durch Systeme mit graphischer Oberfläche ersetzt, die mit der Maus bedient werden. Trotz der Erfindung dieser Systeme durch die Firma Apple gelten heute Mircosoft Windows-Systeme als die typischen Vertreter dieses Rechnertyps. Aber auch bei UNIX- oder Linux-Systemen gewinnen graphische Oberflächen eine immer größere Bedeutung. Zwar verfügen auch moderne Windows-Systeme noch über mindestens eine Shell für die tastaturgesteuerte Zeicheneingabe (wie in den Beispielen in Abschnitt 2 exemplifiziert). Der Aufruf von Programmen erfolgt aber üblicherweise durch Mausoperationen.

---

[20] Vgl. Krause 1992.

[21] Kritzenberger 1997, S. 152. Zum maschinellen Sprachverstehen vgl. Abschn. 4.

[22] Vgl. Kritzenberger 1997, S. 25 ff. *Wizard of Oz*-Experimente eröffnen die Möglichkeit, »hypothetische Systeme unter annähernd realen Bedingungen in Benutzertests auszuprobieren, ohne daß die Systemkomponenten tatsächlich realisiert sind« (Kritzenberger 1997, S. 33). Programme mit uneingeschränkter menschlicher Sprachkompetenz sind weder heute noch in absehbarer Zukunft realisierbar.

[23] Vgl. Kritzenberger 1997, S. 11.

[24] Vgl. Kritzenberger 1997, S. 39 ff.

Die Veränderungen des Kommunikationsprozesses beim Schritt von der tastatur- zur mausgesteuerten Zeicheneingabe lassen sich als Umsetzung von drei Prozessen begreifen. Erstens werden sprachliche Ausdrücke durch bildliche Zeichen (Programmikons) ersetzt. Zweitens werden komplexe Kommandozeilen in Folgen einfacher Anweisungen aufgelöst. Drittens wird die Formulierung bestimmter Befehle dadurch vermieden, dass die gewünschte Operation durch eine Bewegung mit dem Mauszeiger simuliert wird.

3.1 *Ersetzung einfacher sprachlichen Ausdrücke durch Programmikons*

Bei der Ersetzung einfacher sprachlicher Ausdrücke (Programmaufrufe als Wörter oder Abkürzungen wie `explorer`, `coreldrw`, `nc` etc.) durch Programmikons handelt es sich um eine Umwandlung von symbolischen in ikonische Zeichen.[25] Man müsste eigentlich von »Ikons zweiten Grades« sprechen, denn der unmittelbare Bezug zur Lebenswirklichkeit, den beispielsweise die Abbildung von spielenden Kindern auf dem Straßenschild für Spielstraße hat, fehlt den meisten Programmikons. Sie müssen wie symbolische Zeichen gelernt werden, sind aber anschließend als Bildinformation sicherer im Gedächtnis gespeichert und schneller mental verfügbar als die sprachlichen Ausdrücke, für die sie stehen. Denn hinter jedem Programmikon steht eine Kommandozeile, wie ein Blick in die Registerkarte mit den Eigenschaften des Ikons für das Graphikprogramm CorelDraw in Abb. 2 zeigt. Die Ikons für häufig verwendete Programme sind üblicherweise auf dem Bildschirm wie auf einem Schreibtisch[26] angeordnet (in Abb. 2 sind die Ikons für CorelDraw und den Datei-Manager Winfile zu sehen) und mit dem Mauszeiger ansteuerbar. Die mausklicksensitive Schreibtischansicht, der in Windows-Systemen das Programm Explorer als Shell zugrundeliegt, erfüllt die Funktion des Prompts: Das System ist zur Verarbeitung von Befehlen bereit. Das einfache Anklicken eines Programmikons ist bedeutungsgleich mit der Eingabe der im Ziel-Feld aufgeführten Kommandozeile. Der Doppelklick verbindet die Kommadozeilen mit dem Drücken der Eingabetaste und führt zur Übergabe des Befehls an das System.

3.2 *Zerlegung komplexer Ausdrücke in Folgen einfacher Anweisungen*

Programmikons vertreten einfache sprachliche Ausdrücke, also Befehle, bei denen keine Parameter an die aufgerufenen Programme übergeben werden. Von den in Abschnitt 2 angeführten Beispielen wäre nur `time` mit einem einfachen Programmikon zu ersetzen. Der Ausdruck `rename a.txt b.txt` ist, obwohl mit ihm ein sehr einfaches Programm aufgerufen wird, zu komplex, um von einem einfachen Programmikon ersetzt werden zu können. Deshalb wird hier zur Vermeidung der Spracheingabe ein zweites Verfahren angewendet, das auf die (mentalen) Entscheidungsprozesse zurückgreift, die zur Formulierung des Handlungsziels erfolgen müssen: Die Eingabe des komplexen Ausdruck wird in eine Reihe von Einzelschritten aufgelöst. Diese Operation erfolgt bei Windows-Systemen üblicherweise nach Aufruf der graphischen Oberfläche des Explorers.[27] Den Ausgangspunkt der Operation bildet dort nicht das logische Prädi-

[25] Zur Zeichentheorie vgl. Peirce 1993, zur hier verwendeten Terminologie bes. S. 64–67.

[26] Auch die Schreibtisch-Metapher (*Desktop*) wurde erstmals beim Apple Lisa 1983 verwendet.

[27] Wie im vorhergehenden Kapitel ausgeführt, wird der Explorer beim Systemstart automatisch aufgerufen, ohne den Explorer lässt sich kein Programmikon anklicken. Umbenennungen von Objekten der Schreibtischansicht sind folglich auch ohne den Aufruf der graphischen Oberfläche des Explorers möglich.

kat `rename`, sondern das Argument `a.txt`: Die umzubenennende Datei ist der Bezugspunkt der technischen Operation. Nach der Auswahl der Datei wird die Operation gewählt, also `rename`, die Wahl der Prädikatsspezifikation `b.txt` bleibt ihr nachgeordnet. Wie in Abb. 3 dargestellt, wird dem Anwender nach Markierung der Bezugsdatei `a.txt` durch Anklicken der Schaltfläche `Datei` ein Ausklappmenü mit verschiedenen auf Dateien anwendbaren Operationen angeboten, von denen der Anwender durch Mausklick die Funktion `Umbenennen` auswählt. Danach öffnet sich ein Dialogfeld, in das der neue Dateiname `b.txt` eingetippt wird. Bei komplexen Programmpaketen (beispielsweise für Graphik oder Textverarbeitung) ist es keine Seltenheit, dass der Anwender sukzessive zwei oder drei Menüfenster nacheinander bearbeitet, bevor sich dann ein Dialogfeld zur Eingabe eines sprachlichen Ausdrucks öffnet. Oft kommt dieser Dialog auch völlig ohne Spracheingabe aus: Statt eines Textfeldes schlägt der Computer im letzten Menüfenster mehrere Handlungsmöglichkeiten vor, von denen der Anwender eine per Mausklick auswählt.

Auf den ersten Blick erscheint das sukzessive Abarbeiten der Ausklappmenüs in der in Abb. 1 für den Befehl `time` entwickelten Dialogstruktur zu verlaufen. Ein substantieller Unterschied liegt allerdings in der Besetzung der Kommunikationsrollen. Bei kommadozeilenbasierten Systemen bleibt dem Anwender auch nach der Eröffnung der Kommunikation die Möglichkeit zur Spracheingabe. Nach dem Aufruf eines Programms mit Ausklappmenüs (in der Regel durch Anklicken des entsprechenden Programmikons) geht die sprachliche Initiative dagegen an den Computer über. Er bietet sprachliche Ausdrücke an, auf die der Anwender, vermittelt durch Mausoperationen, mit Ja/Nein-Antworten reagiert und deren Summe die Ausführung der Operation bewirkt. Eigeninitiative Spracheingabe findet nicht statt.

3.3 *Ersetzung komplexer Ausdrücke durch Bewegungsoperationen mit der Maus*

Das dritte Verfahren zur Vermeidung der Spracheingabe besteht in der Simulation der durch den Befehl ausgelösten »Bewegung«. So wird der »Standort« einer Datei im Explorer unter Windows 98 in einem Verzeichnisbaum visualisiert.[28] Die Eingabe des Befehls `copy f:\a.txt a:\a.txt` kann deshalb im Explorer durch »Ergreifen« der Datei mit dem Mauszeiger und anschließendes »Ziehen« aus dem Quellverzeichnis `f:\` (einer Festplattenpartition) ins Zielverzeichnis `a:\` (die Diskette) ersetzt werden, wie in Abb. 4 dargestellt. Statt der eigeninitiativen Spracheingabe oder der Auswahl vorgegebener sprachlicher Ausdrücke (über Menüeinträge) wird hier eine nichtsprachliche, körperliche Handlung simuliert.

4 *Akustische Zeicheneingabe: Maschinelles Sprachverstehen*

Systeme zum maschinellen Sprechverstehen haben im Unterschied zu den oben behandelten Kommunikationsformen heute noch keinen Entwicklungsstand erreicht, der einen zuverlässigen Einsatz auf dem Personalcomputer des privaten Anwenders erlauben würde. Eine Ausnahme bilden lediglich Diktiersysteme und damit kombinierte Zusatzapplikationen. Systeme, die natürliche Sprache sprecherunabhängig in Echtzeit de-

[28] Diese Visualisierung bildet die tatsächlichen technischen Verhältnisse nur sehr bedingt ab und ist als Metapher zu verstehen.

kodieren und bearbeiten können, erfordern überdurchschnittlich leistungsfähige Rechner. Zwar betreiben die im kommerziellen Bereich führenden Unternehmen (v.a. Philips und IBM) intensive Forschung zur Weiterentwicklung der Systeme zur Spracherkennung. Die Entwicklung zielt aber eher auf Anwendungen in der Telephonie und der Sprachsteuerung einzelner elektronischer Bauteile beispielsweise in Autos ab als auf die Bedienung kompletter PC-Systeme. Systeme zum maschinellen Sprachverstehen sollen deshalb im Folgenden nur kurz skizziert werden.

4.1 *Spracherkennung und Sprachverstehen*

Der wesentliche Unterschied zwischen Systemen zum maschinellen Sprachverstehen und den oben behandelten Kommunikationsformen besteht im Übertragungskanal: Sowohl bei der tastatur- als auch bei der mausgesteuerten Zeicheneingabe wird ein visuell kontrollierter taktiler Kanal verwendet. Bei den jetzt behandelten Systemen wird dagegen der akustische Kanal genutzt. Die Zeicheneingabe wird damit für den Anwender unmittelbarer[29], schneller[30] und »natürlicher«[31]. Grundsätzlich besteht das maschinelle Sprachverstehen aus zwei eng miteinander verknüpften Komponenten.[32] Die eigentliche Spracherkennung besteht in der Extraktion physikalischer Merkmale aus dem Sprachsignal und ihrer statistischen Dekodierung. Dieser Vorgang wird durch artikulationsphonetisch bedingte Assimilationen, Reduktionen und Tilgungen von Lauten, Silben und ganzen Wörtern (um nicht von dialektalen oder gar idiolektalen Merkmalen zu sprechen) im kontinuierlichen Redestrom behindert. Eine verlässliche Ableitung von Wörtern aus physikalischen Merkmalen ist nicht möglich. Zur Feststellung der akustisch bestpassenden Repräsentation der Äußerung des Anwenders wird deshalb meist auf statistische Markov-Modelle zurückgegriffen.[33] Akzeptable Erkennungsquoten werden erreicht, wenn schon in der Komponente der Spracherkennung ein Abgleich der möglichen Lautketten mit den Einträgen eines Aussprachelexikons stattfindet. In manchen Systemen findet außerdem schon bei der Spracherkennung eine Bewertung der Wahrscheinlichkeit des Vorkommens einer Sequenz von Wörtern in einer bestimmten Sprache statt.[34] Danach geht der Prozess von der Spracherkennung zum eigentlichen Sprachverstehen über. Im System von Philips werden die wahrscheinlichsten Wortsequenzen in einer Wortgraphen-Repräsentation an die Sprachverstehenskomponente übergeben, wo die Wortsequenzen im Hinblick auf bestimmte Informationen geparst und semantisch ausgewertet werden.[35]

---

[29] In der Telephonie und der Sprachsteuerung in Autos ist die Spracherkennung besonders interessant, weil der Anwender in diesen Situationen keine Tastatur zur Verfügung hat bzw. sie (im Auto) nicht bedienen könnte.

[30] Ein nicht geschulter Anwender tippt pro Minute ca. 10–25 Wörter, er spricht dagegen 120–250. Vgl. Schukat-Talamazzini 1995, S. 1.

[31] Wobei man die eingeschränkten Erwartungen des Menschen an Natürlichkeit im Computer Talk berücksichtigen muss, vgl. Exkurs.

[32] Vgl. Abb. 1.5 bei Schukat-Talamazzini 1995, S. 15.

[33] Vgl. Schukat-Talamazzini 1995, S. 121–163.

[34] Z. B. durch N-gramm- oder Wortgraphen-Modelle, vgl. Kritzenberger 1997, S. 49 ff.

[35] Zum System von Philips vgl. http://www.speech.philips.com [Stand: 30. 08. 2002]. Zu Fahrplanauskunftssystemen generell vgl. neben Kritzenberger 1997 z. B. Gerbino/Danieli 1993.

4.2 *Natürlichsprachliche Dialogsysteme*

Die derzeit leistungsfähigsten Dialogsysteme sind für telefonische Auskunftssysteme entwickelt worden. Im Fahrplanauskunftssystem von Philips[36] durchsucht die Sprachverstehenskomponente die Wortgraphen-Repräsentation jeder Benutzeräußerung nach Informationen über den Ausgangs- und Zielpunkt der Reise sowie nach Zeitangaben. Sie kontrolliert, inwieweit die Benutzeräußerung Werte für Variablen wie *$Ausgangspunkt*, *$Zielpunkt*, *$Monat*, *$Tag*, *$Zeit* und dergleichen enthält. Das System bestätigt das Sprachverstehen, d. h. die Übermittlung solcher Werte, durch ihre Reproduktion. Die Reihenfolge, in der die Werte übergeben werden, ist frei. Bleiben nach der Benutzeräußerung relevante Variablen leer, fordert das System diese Informationen durch Präzisierungsfragen ein. Der Dialog dauert so lange, bis allen Variablen Werte zugewiesen sind: Danach wird die gewünschte Auskunft erteilt. Bei meinem Test des Systems ist die folgende Dialogstruktur entstanden (in eckigen Klammern sind die Werte angegeben, die das System der entsprechend Benutzeräußerung zuweist, also das, was es an der Benutzeräußerung »versteht«):

System: Von wo nach wo möchten Sie fahren?

Benutzer: Ich will nach Leipzig fahren. [$Zielpunkt = Leipzig Hbf]

System: Von wo aus möchte Sie nach Leipzig Hbf fahren?

Benutzer: Ich will von Dessau nach Leipzig fahren. [$Ausgangspunkt = Dessau Hbf]

System: Wann möchten Sie von Dessau Hbf nach Leipzig Hbf fahren?

Benutzer: Morgen Nachmittag. [$Monat = Juli; $Tag = 8; $Zeit = 12.00-18.00][37]

System: Sie möchten also morgen zwischen 12.00 und 18.00 fahren?

Benutzer: Ja.

System: Die früheste Verbindung lautet wie folgt...

Die Zahl der für einen solchen Dialog relevanten Wörter ist beschränkt. Es ist möglich, einen semantischen Frame[38] zu definieren, der alle für die Fahrplanauskunft relevanten Wörter und Strukturen enthält. Das sind für den Reiseverlauf neben einigen hundert Ortsnamen lediglich Präpositionen wie *von*, *nach* oder *über*, für die Zeitangaben die Zahlen für Datums-, Stunden- und Minutenangaben, die Monats- und Wochentagsnamen sowie Temporaladverbien. Der verbale Rahmen von Auskunftsfragen wird durch Modalverben wie *wollen* oder *müssen* und Bewegungsverben wie *fahren*, *reisen* oder *ankommen* gebildet. Wörter und Strukturen, die nicht im Frame vorgesehen sind, werden vom System ignoriert. Äußerungen, deren Elemente innerhalb des Frames liegen, werden dagegen weitgehend zuverlässig erkannt: Bei meinen Tests des Systems ist es in keinem Fall zu Missverständnissen gekommen.

[36] Telephonisch testbar unter +49 241 604020 [Stand: 30. 08. 2002].
[37] Das Bezugsdatum des Test ist der 07. 07. 2001.
[38] Zur Frametheorie vgl. Minsky 1980.

5 *Fazit*

Weil der Personalcomputer bei der Entwicklung von Systemen für automatisches Sprachverstehen nur eine untergeordnete Rolle spielt, wird es auf absehbare Zeit keine vollständig durch akustische Spracheingabe gesteuerten PC-Systeme geben. Mensch-Maschine-Kommunikation am PC erfolgt bis auf Weiteres über den taktilen und den visuellen Kanal: taktil (durch Tastatur und Maus) mit visueller Kontrolle (über den Bildschirm) bei der Zeicheneingabe des Benutzers und visuell (auf dem Bildschirm, ggf. akustisch unterstützt durch so genannte »Systemklänge«) durch das System. Der unmittelbarste Zugriff auf die Systemressourcen ist mit tastaturgesteuerter und kommandozeilenbasierter Zeicheneingabe möglich. Dafür ist allerdings die genaue Kenntnis von Syntax und Semantik des Systemkodes erforderlich. Um dem privaten Anwender die Erlernung dieses Kodes zu ersparen, wurden Systeme mit graphischer Oberfläche entwickelt, die eine weitgehend intuitive, mausgesteuerte Zeicheneingabe ermöglichen. Die drei oben beschriebenen Verfahren ermöglichen es auch dem technisch nicht versierten Anwender, erfolgreich mit der Maschine zu kommunizieren. Mit Programmikons, Ausklappmenüs und der Simulation von Bewegungsoperationen mit der Maus sind viele Routineaufgaben schnell und unkompliziert zu erledigen. Und die technische Entwicklung wird zweifellos zu immer einfacher mit der Maus (oder einem funktional äquivalenten Touchscreen) zu bedienenden Systemen führen.[39] Der Preis für die einfache Bedienbarkeit ist jedoch in manchen Fällen eine Reduzierung der Handlungsmöglichkeiten, weil bestimmte Programmfunktionen nur über die Kommandozeile aufzurufen sind. In jedem Fall wird die einfache Handhabbarkeit mit dem Verlust der Möglichkeit zur selbstbestimmten Spracheingabe bezahlt. Statt dem Computer direkt Befehle zu erteilen, beschränkt sich der Anwender darauf, vom System vorgegebene Befehlsoptionen mit mausklickvermittelten Ja/Nein-Antworten zu quittieren.

Literatur

Auer, Peter: *From Context to Contextualization*; in: *Link & Letters* 3 (1996), S. 11-28.

Bierwisch, Manfred: *Semantic Structure and Illocutionary Force*; in: Searle, John R./Kiefer, Ferenc/Bierwisch, Manfred (Hgg.): *Speech Act Theory and Pragmatics*; Dordrecht 1980 (*Synthese Language Library*; 10), S. 1-35.

Franke, Wilhelm: *Über nichtspezifische reaktive Sprechakte*; in: Hindelang, Götz/Zillig, Werner (Hgg.): *Sprache: Verstehen und Handeln. Akten des 15. Linguistischen Kolloquiums*; Münster 1980. Bd. 2. Tübingen 1981 (*Linguistische Arbeiten*; 99), S. 237–247.

Franke, Wilhelm: *Elementare Dialogstrukturen. Darstellung, Analyse, Diskussion*; Tübingen 1990 (*Reihe Germanistische Linguistik*; 101).

Gerbino, Elisabetta/Danieli, Morena: *Managing Dialogue in a Continuous Speech Understanding System*;. in: *EUROSPEECH 1993. Proceedings of the 3rd European Conference on Speech Communication and Technology. 21-23 September*; Berlin, S. 1661–1664.

Giese, Heinz W./Januschek, Franz: *Das Sprechen, das Schreiben und die Eingabe. Spekulationen über Entwicklungstendenzen von Kommunikationskultur*; in: Weingarten, Rüdiger (Hg.): *Information ohne Kommunikation? Die Loslösung der Sprache vom Sprecher*; Frankfurt/Main 1990, S. 54–74.

[39] Die einfache Handhabbarkeit wird auch von der deutschen Bildschirmarbeitsplatzverordnung verlangt, in der es heißt: »Die Systeme müssen den Benutzern die Beeinflussung der jeweiligen Dialogabläufe ermöglichen sowie eventuelle Fehler bei der Handhabung beschreiben und deren Beseitigung mit begrenztem Arbeitsaufwand erlauben. [...] Die Software muss entsprechend den Kenntnissen und Erfahrungen der Benutzer im Hinblick auf die auszuführende Aufgabe angepasst werden können.« (http://dhv-cgb.de/gesetze/bildschirm.htm [Stand: 30. 08. 2002])

Grice, H. Paul: *Logic and Conversation*; in: Cole, Peter/Morgan, Jerry L. (Hgg.): *Syntax and Semantics*, Bd. 3. New York 1975, S. 41–58.

Hindelang, Götz: *Sprechakttheoretische Dialoganalyse*; in: Fritz, Gerd/Hundsnurscher, Franz (Hgg.): *Handbuch der Dialoganalyse*; Tübingen 1994, S. 95–112.

Krause, Jürgen: *Natürlichsprachliche Mensch-Computer-Interaktion als technisierte Kommunikation: die computer talk-Hypothese*; in: Krause, Jürgen/Hitzenberger, Ludwig (Hgg.): *Computer Talk*; Hildesheim 1992, S. 1–29.

Kritzenberger, Huberta: *Dialoge mit Computern in natürlicher Sprache*; Diss. Regensburg 1997.

Minsky, M.: *A Framework for Representing Knowledge*; in: Metzing, Dieter (Hg.): *Frame Conceptions and Text Understanding*; Berlin/New York 1980 (*Research in Text Theory*, 5), S. 1–25.

Peirce, Charles Sanders: *Phänomen und Logik der Zeichen*. Hg. und übers. von Helmut Pape. 2. Aufl. Frankfurt/Main 1993.

Rabanus, Stefan: *Die Sprache der Internet-Kommunikation*; Mainz 1996.

Runkehl, Jens/Schlobinski, Peter/Siever, Torsten: *Sprache und Kommunikation im Internet. Überblick und Analyse*; Opladen/Wiesbaden 1998.

Schmitz, Ulrich: *Zur Sprache im Internet. Skizze einiger Probleme und Eigenschaften*; online im Internet unter http://www.linse.uni-essen.de/papers/sprache_internet.htm [Stand: 30. 08. 2002].

Schukat-Talamazzini, Ernst Günter: *Automatische Spracherkennung. Grundlagen, statistische Modelle und effiziente Algorithmen*; Braunschweig/Wiesbaden 1995.

Searle, John R.: *A Classification of Illocutionary Acts*; in: *Language in Society* 5 (1976), S. 1–24.

Searle, John R./Vanderveken, Daniel: *Foundations of Illocutionary Logic*; Cambridge 1985.

Weizenbaum, Joseph: *Die Macht der Computer und die Ohnmacht der Vernunft*; Frankfurt/Main ^{8}1990.

Dr. Stefan Rabanus
Forschungsinstitut für deutsche Sprache/Deutscher Sprachatlas
Fachbereich 09 der Philipps-Universität Marburg
Hermann-Jacobsohn-Weg 3, 35039 Marburg
E-Mail: rabanus@mailer.uni-marburg.de
Internet: www.deutscher-sprachatlas.de

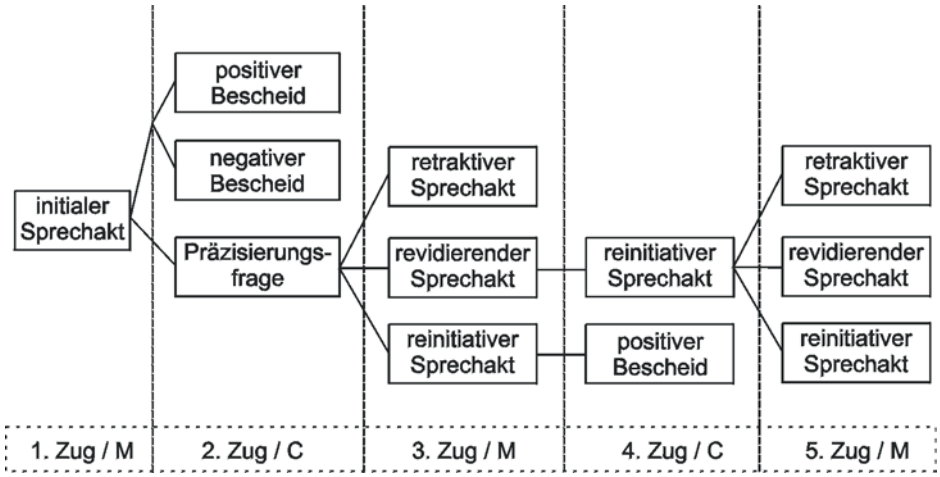

Abb. 1: Minimaldialoge in der Mensch-Maschine-Kommunikation
(M = Mensch, C = Maschine)

Abb. 2: Windows-Desktop mit Registerkarte

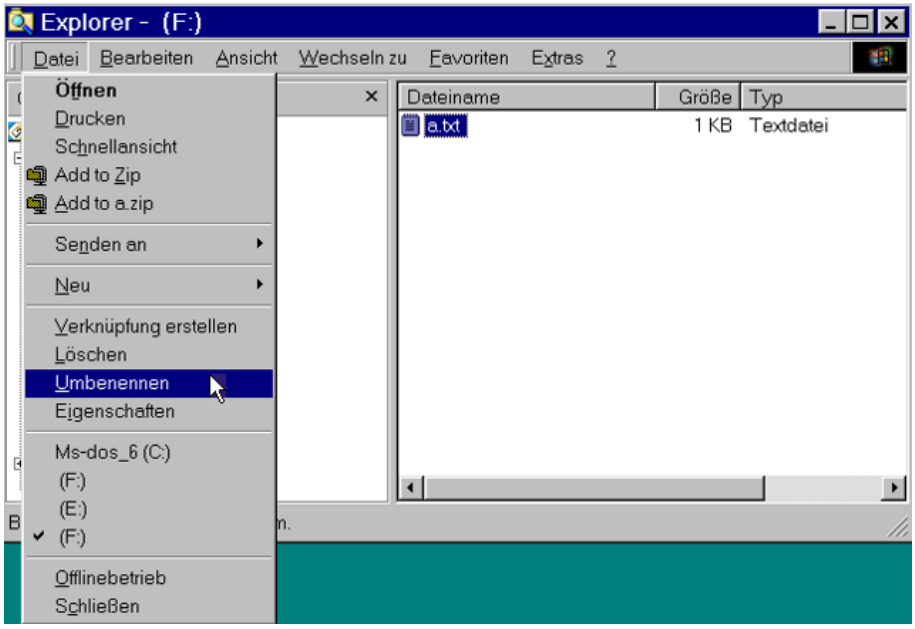

Abb. 3: `Umbennen` im Windows-Explorer

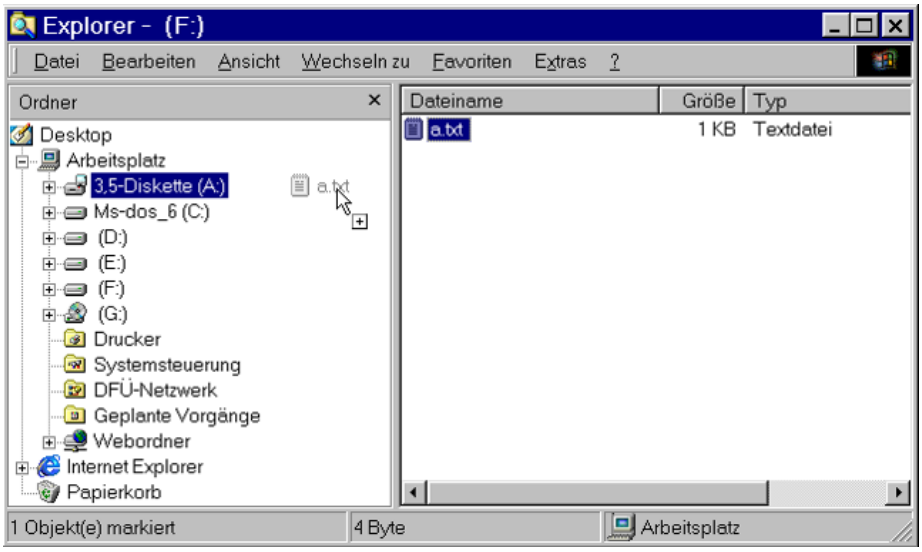

Abb. 4: `Kopieren` im Windows-Explorer