# inforum

INFOrmationsforum des Rechenzentrums der Universität Münster

Jahrgang 11, Nr. 2 – April 1987 ISSN 0931–4008

## Inhalt

**Impressum** **inforum**

ISSN 0931-4008

Redaktion: A. Achilles (Tel. 83-2607)
W. Bosse (Tel. 83-2461)
H. Pudlatz (Tel. 83-2472)
E. Sturm (Tel. 83-2609)

Satz: E. Krause
L. v. Wüllen

Druck: H. Mecke

Universitätsrechenzentrum
Einsteinstraße 60
4400 Münster

Auflage dieser Ausgabe: 1000

Redaktionsschluß der nächsten Ausgabe: 1.7.1987

## Editorial

von

*A. Achilles*

Sind Sie gespannt auf die neue inforum-Ausgabe? Ich kann Ihnen verraten, daß auch ich neugierig auf die ersten Drucke dieser Auflage warte. Dank neuer Hardware – eines Scanners, der die graphischen Ein- und Ausgabegeräte um eine neue Komponente bereichert – sollen Sie angeblich oben auf der Seite ein Foto von mir zu sehen bekommen. Für alle die, die von mir gehört, mich aber noch nicht gesehen haben, wird damit der Schleier gelüftet.

Die Abbildung, die durch den Scanner in diese Auflage übernommen wird, sollte aber nicht das einzige sein, was Sie neugierig stimmt! Mit diesem Hinweis meine ich weder die übliche Übersicht über die Lehrveranstaltungen des Rechenzentrums im Sommersemester noch den Dauerbrenner „Bildschirmsteuerung im Dialog mit ISPF". (Hierzu kann ich Sie trösten: nach diesem Beitrag kommen nicht mehr allzu viele Fortsetzungen.) Ich beziehe mich vielmehr auf das Thema Datensicherung, das keinen EDV-Benutzer kalt läßt! Zwei Beiträge sind diesem Thema gewidmet.

Als nützliches CMS-Kommando wird der elektronische Kalender vorgestellt. Sie müssen nur noch selber *planen*, die Erinnerung erfolgt – bei täglicher Benutzung Ihrer CMS-Maschine – automatisch.

Mit der Welt der PCs beschäftigen sich der Bericht „Computer-Investitionsprogramm" und der Beitrag „Programmverwaltung unter dem Betriebssystem MS-DOS". Erste Informationen über die neuen PCs gibt der gleichnamige Beitrag. Auf den Einsatz von SPSS auf PCs wird in „Neues vom SPSS" unter anderem hingewiesen.

Für diejenigen, die an Graphik interessiert sind, gibt es nun die Möglichkeit, für den Farbrasterplotter erstellte Bilder zu beschriften. Dies wird dargestellt in „Beschriftungen für COLPLOTs".

Wenn Ihnen nach soviel Lektüre der Kopf brummt, sollten Sie sich ein wenig entspannen: denken Sie z. B. über die „Auswahl der richtigen Programmiersprache" nach oder nehmen Sie am inforum-Quiz teil. Vielleicht versuchen Sie es auch mit dem kleinsten Formatierungssystem der Welt?

Und nun viel Spaß bei der Lektüre!

## RUM-Aktuell

# Datensicherung am Rechenzentrum

von

*H. Schalthöfer*

Ein großer Teil sowohl der Arbeitszeit der Benutzer als auch der erbrachten Leistung der Rechner wird darauf verwendet, Informationen in Dateien zu speichern. Diese Dateien sollten deshalb mit vertretbarem Aufwand so gut wie möglich gegen unbeabsichtigtes Löschen geschützt werden.

Im folgenden werden nach einer Klärung der verwendeten Begriffe die vom Rechenzentrum durchgeführten Maßnahmen der Datensicherung vorgestellt und Empfehlungen für die eigene Datensicherung der Benutzer erteilt.

### *Abgrenzung der Begriffe*

Mit **Datensicherung** werden alle Vorkehrungen bezeichnet, die das Wiederherstellen des Zustandes möglichst kurz vor Eintritt eines Datenverlusts ermöglichen. Ein solcher Datenverlust kann verschiedene Ursachen haben, z. B. Fehlfunktionen von Teilen der Hardware oder der Systemsoftware, aber auch falsche Eingaben durch den Benutzer.

Unter **Datenschutz** versteht man Maßnahmen, die den unbefugten Zugriff auf Dateien verhindern. Der Datenschutz fällt in die alleinige Verantwortung der Benutzer, es stehen unter CMS das PASSWORD-Kommando, unter UNIX das CHMOD-Kommando und unter MVS die RUMSERV-Anweisungen PROTECT und PERMIT zur Verfügung.

**Datenarchivierung** soll die Übertragung von Dateien auf ein Speichermedium mit höherer Kapazität und geringerer Zugriffsgeschwindigkeit bezeichnen, in der Regel das Kopieren von Magnetplatte auf Magnetband. Das Archivieren seiner Dateien kann nur der Benutzer selbst durchführen, da allein er die gewünschte Aufbewahrungsdauer kennt. Zum Archivieren von Daten im CMS kann das ARCHIV-Kommando (vgl. „Datei-Archivierung mit ARCHIV" in dieser Ausgabe) bzw. im MVS die RUMSERV-Anweisung UNLOAD verwendet werden.

Die Datensicherung für alle Rechner und Betriebssysteme wird zentral durch das Rechenzentrum durchgeführt. Zielsetzung ist es, die negativen Folgen einer Zerstörung von Daten so gering wie möglich zu halten.

### *Datensicherung unter VM/CMS*

Die auf CMS-Platten gespeicherten Daten werden an jedem Arbeitstag einmal auf Magnetband übertragen. Dies hat den Vorteil, daß ein Benutzer im Falle eines Fehlers die Arbeit maximal eines Tages wiederholen muß. Die Aufbewahrungsfrist dieser Bänder beträgt in der Regel zwei Wochen, einige ausgewählte Sicherungskopien werden länger (z. Z. bis zu acht Monate) aufbewahrt. Von der CMS-Datensicherung ausgenommen sind Kurs- und Übungsnummern, die lediglich in unregelmäßigen Abständen gesichert werden.

Hinweise:

- Während der Datensicherung der A-Maschine (z. Z. morgens von 6.45 Uhr bis ca. 7.30 Uhr) ist kein Dialogbetrieb möglich.
- Die Datensicherung der C-Maschine erfordert keine zwingende Einschränkung des Dialogbetriebs, jedoch wird an die aktiven Benutzer eine Warnung geschickt, einen Augenblick lang ihre Minidisk nicht zu verändern.
- Es erfolgt prinzipiell keine Sicherung der SPOOL-Dateien. Aus diesem Grund kann nur davor gewarnt werden, Dateien, die sich nicht leicht neu erzeugen lassen, auf dem virtuellen Leser zu speichern.
- Da die Magnetbandstationen normalerweise nur vom MVS aus benutzt werden können, erfordert das Wiederherstellen von CMS-Dateien eine Reihe von Operatoreingriffen in die Betriebssysteme der B- und der C-Maschine. Deshalb kann es mehrere Tage dauern, bis verlorene Dateien wieder verfügbar sind.

### *Datensicherung unter UNIX*

Unter UNIX wird wöchentlich eine Sicherung der Daten auf Band durchgeführt, die jeweils 30 Tage

aufbewahrt wird. Zusätzlich werden täglich Änderungen an den Dateien gesichert, diese werden sieben Tage aufbewahrt.

*Datensicherung unter MVS*

Mit Ausnahme der Tagesdateien werden im MVS die mit der RUMSERV-Anweisung ALLOCATE neu angelegten Dateien innerhalb eines Tages gesichert. Danach wird eine neue Sicherungskopie dann angelegt, wenn die Datei verändert wurde und die alte Version der Sicherung ein bestimmtes Mindestalter (z. Z. zwei Tage) hat. Nachdem eine seit 32 Tagen nicht benutzte MVS-Datei gelöscht wurde, existiert die Sicherungskopie noch weitere drei Monate und wird dann ebenfalls gelöscht.

Hinweise:

- Da der Dateiname die einzige Identifikation einer Datei darstellt, wird eine alte Sicherungskopie dann gelöscht, wenn eine neue Datei gleichen Namens angelegt wird. Es besteht im MVS also nicht die Möglichkeit, eine ältere Version einer Datei wiederherzustellen.
- Auch im MVS erfordert das Wiederherstellen von gelöschten Dateien einen recht hohen Zeitaufwand, weshalb das Rechenzentrum dies nur in begründeten Einzelfällen durchführt.

*Zusammenfassung*

Die Datensicherungsmaßnahmen des Rechenzentrums dienen in erster Linie dem Schutz der Benutzer vor von ihnen nicht zu vertretenden Datenverlusten. Es kann jedoch nicht 100-prozentig garantiert werden, daß in jedem Fall eine brauchbare Sicherung der Daten existiert. Deshalb sollten besonders wichtige und schwer zu erzeugende Daten zusätzlich durch den Benutzer selbst gesichert werden.

Ansprechpartner bei Fragen zur Datensicherung sind Reinhard Mersch und Heinz Schalthöfer (Tel. 2488) für die Systeme CMS und MVS, Wilhelm Stockhofe (Tel. 2685) für das Betriebssystem UNIX.

## Datei-Archivierung mit ARCHIV

von

*R. Mersch*

Die Archivierung von Dateien, d.h. das Kopieren von Dateien auf ein Magnetband, empfiehlt sich immer dann, wenn man sich vor Datenverlust schützen will und/oder wenn man vorläufig nicht benötigte Dateien löschen will, um auf der Minidisk Platz zu schaffen.

Da im Universitätsrechenzentrum nur die MVS-Maschine über Bandstationen verfügt, müssen zu archivierende CMS-Dateien mit Hilfe von MVS-Jobs auf ein Magnetband kopiert werden. Sollen mehrere Dateien archiviert werden, so besteht grundsätzlich die Wahl, jede dieser Dateien separat zu bearbeiten oder sie alle zu einer Datei zusammenzufassen und diese zu archivieren. Beide Vorgehensweisen sind umständlich, wobei erstere zusätzlich die Operateure entnervt, weil sie für jede Datei das Magnetband erneut auflegen müssen.

Hier schafft das ARCHIV-Kommando Abhilfe, das am 23.4.1987 zur Verfügung gestellt werden soll. Es faßt die zu archivierenden Dateien (im DISK DUMP-Format) zu einer Datei, der Archiv-Datei, zusammen, bettet diese in den erforderlichen MVS-Job ein und schickt diesen Job an die MVS-Maschine. ARCHIV muß dazu mit den Namen der zu archivierenden Dateien, dem Namen des Magnetbandes, der Position, an der die Archiv-Datei auf dem Magnetband abgespeichert werden soll, und, falls nicht die Voreinstellung benutzt werden soll, dem gewünschten Namen der Archiv-Datei auf dem Magnetband versorgt werden. Zudem kann ARCHIV aufgefordert werden, im Anschluß an die Archivierung eine Liste der auf dem benutzten Magnetband enthaltenen Dateien erstellen zu lassen. Näheres ist den HELP-Informationen zum ARCHIV-Kommando zu entnehmen.

Einige Bemerkungen zur Benutzung:

- Nach Durchführung der Archivierung wird die übliche Job-Ausgabe zum virtuellen Kartenleser geschickt. Hier sollte auf jeden Fall nachgeschaut werden, ob die Archivierung geglückt ist, bevor etwa Dateien gelöscht werden. Die unten angeführte Bemerkung über die Dateigröße sollte beachtet werden!
- Zum Zurückholen der archivierten Dateien muß ein kurzer Job, der in den HELP-Informationen zum ARCHIV-Kommando beschrie-

ben wird, abgeschickt werden. Die gesamte Archiv-Datei wird dann zum virtuellen Kartenleser der anfordernden CMS-Maschine geschickt. Zum Empfang aller in der Archiv-Datei enthaltenen Dateien kann das RECEIVE-Kommando verwendet werden, jedoch ist zu beachten, daß RECEIVE die Dateien in diesem Falle nur auf die A-Platte übertragen kann. Eventuell bereits existierende Dateien gleichen Namens werden dabei überschrieben! Ist dies nicht erwünscht, so empfiehlt es sich, zunächst eine temporäre Minidisk als A-Platte in Zugriff zu nehmen, dann das RECEIVE-Kommando aufzurufen, und anschließend die benötigten Dateien zu kopieren.

- Von der MVS- zu einer CMS-Maschine können z. Z. nicht mehr als 10000 Zeilen übertragen werden. Dies bedeutet, daß einerseits zwar eine Archiv-Datei, die länger als 10000 Zeilen ist, erstellt und auf ein Magnetband übertragen werden kann, andererseits aber diese Datei nicht mehr zur CMS-Maschine zurückgeholt werden kann. Leider kann erst nach dem Verschicken der Archiv-Datei ermittelt werden, wie groß sie ist. Deshalb sollte der Benutzer selbst darauf achten, daß sie nicht zu groß wird. Als Daumenregel mag dienen, *daß die zu archivierenden Dateien zusammen nicht mehr als 500 Blöcke (zu je 1024 Byte) einnehmen sollten.* Ist die Archiv-Datei zu groß, so wird zwar die Archivierung vorgenommen, das ARCHIV-EXEC gibt aber anschließend eine Warnung aus.

- Bitte beachten Sie bei der Angabe der Magnetband-Position für die Archiv-Datei die Eigenarten der Magnetbandverarbeitung. Diese Position sollte in der Regel um 1 größer sein als die Position der letzten Datei auf dem Band. Die Wahl einer größeren Position führt zu einem Fehler. Die Wahl einer kleineren Position bewirkt, daß alle Dateien, die zuvor an dieser *und an allen folgenden* Positionen standen (also der gesamte Bandinhalt ab dieser Position), gelöscht werden!

- Wenn Sie ein Magnetband, das mittels ARCHIV erstellte Archiv-Dateien enthält, an einem anderen Rechenzentrum, das nur über eine CMS-Maschine, nicht aber über eine MVS-Maschine verfügt, lesen wollen, so ist dies grundsätzlich möglich. Wenden Sie sich in diesem Fall an den Autor (URZ57 at DMSWWU1A).

## Computer-Investitions-Programm (CIP)

von

*W. Held*

Nachdem 1985 im Rahmen des Computer-Investitions-Programms (CIP) sechs vernetzte Rechner-Pools eingerichtet wurden, konnte Ende 1986 mit der Installation neun weiterer Pools begonnen werden. Eine Restabwicklung des Antrags aus 1986 steht noch aus.

Damit sich die für die Pool-Betreuung zuständigen Wissenschaftler kennenlernen und Erfahrungen austauschen konnten, fand am 11.3.87 ein erstes Treffen statt. Der Erfahrungsaustausch stieß auf großes Interesse und soll vor Beginn des Wintersemesters wiederholt werden.

Einige Hinweise aus dieser Veranstaltung sollen hier zusammengefaßt werden:

- Reparatur der Geräte und Netze:
  Dem Rechenzentrum ist aus Landesmitteln Geld zur Verfügung gestellt worden, um anfallende Hardware-Reparaturkosten finanzieren zu können. Defekte Geräte, die 1985 beschafft wurden, können bis auf weiteres im Universitätsrechenzentrum bei Herrn Kisker (Tel. 2467) oder Herrn Goorkotte (Tel. 2672) abgegeben werden. Soweit der Vorrat reicht, stellen wir vorübergehend Ersatzgeräte zur Verfügung. Defekte Geräte, die 1986 beschafft wurden, unterliegen noch der Garantie. In diesem Fall sind die Lieferfirmen unmittelbar zur Reparatur aufzufordern. In beiden Fällen sollte die Reparatur über den für den Pool Verantwortlichen oder dessen Beauftragten abgewickelt werden.

- Die Universitätsverwaltung wird in den Technischen Diensten Stellen für Hardware- und LAN-Technik einrichten. Die Mitarbeiter sollen in Zukunft in enger Zusammenarbeit mit dem Universitätsrechenzentrum für die Reparatur zuständig sein.

- Probleme bei der Nutzung der Pools.
  Probleme und Schwierigkeiten, die mit der Nutzung von Arbeitsplatzrechnern, LAN (Local Area Network), Peripherie und Software auf-

treten, sollten an Herrn Kisker oder Herrn Goorkotte gemeldet werden. In vielen Fällen können wir Hilfe anbieten. Verlassen Sie sich bitte nicht nur auf Ratschläge anderer Pool-Nutzer!

- Überarbeitete CIP-Empfehlungen der DFG:
  Die DFG hat ihre ursprünglichen CIP-Empfehlungen überarbeitet. Es gelten jetzt die Empfehlungen vom 16.12.85. Eine wichtige Änderung betrifft die Aufstellung eines Pools in *einem* Raum. Sie lautet:
  *Von einer Aufstellung der Arbeitsplätze eines Pools in einem Raum kann abgewichen werden, wenn Mikrorechner beispielsweise für die Meßdatenerfassung, Experimentsteuerung, Test- und Prüfaufgaben und zum Entwerfen und Konstruieren in eine Laboreinrichtung eingebettet werden. Die Auflage der Poolbildung und der Vernetzung (vgl. Punkt 3c) bleibt davon unberührt. Der Einsatz von Geräten für derartige fortgeschritten fachspezifische Ausbildungsaufgaben setzt voraus, daß für die DV-technische Grundausbildung der Studenten im betreffenden Bereich bereits gesorgt ist.*
- CIP-Software:
  Den einzelnen Pool-Betreuern liegt eine Liste über alle (1985 und 1986) beschafften Software-Produkte vor. Sie kann dort eingesehen werden.

Weitere Anregungen zum CIP-Betrieb (z. B. Hardware, Software, Vernetzung, Lehre, Beratung und Organisation) nehmen wir gern entgegen.

## infoRUM-Quiz

von

*E. Sturm*

Nach langer Zeit soll jetzt die Tradition wieder aufgenommen werden, pro infoRUM-Ausgabe ein Quiz zu veröffentlichen, in dem unbekannte Annehmlichkeiten oder Verrücktheiten einer Programmiersprache, meist PL/I, vorgestellt werden.

Aus Anfängervorlesungen sollte bekannt sein, was der folgende Programmabschnitt in PL/I bewirkt:

```
DECLARE A (8,8) BIN FIXED
   INIT ((4)((4)(0,1),(4)(1,0)));
A = 1 - A;
```

Stellt man sich A als ein Schachbrett vor (das sollte die Initialisierung nahelegen), so bedeutet die Zuweisung, daß jede 1 der Matrix in eine 0, jede 0 in eine 1 geändert wird, beim Schachbrett also die weißen und schwarzen Felder vertauscht werden. Dies kann man nach einiger Überlegung auch einsehen: die 1 wird sozusagen vervielfältigt zu einer Matrix, von der man wieder eine Matrix abziehen kann.

Was aber bedeutet der folgende Programmabschnitt?

```
DECLARE A (8,8) BIN FIXED
   INIT ((21)(1,2,3),1);
A = (A < 3) * (A + 1) + (A = 3);
```

Die Auflösung erfolgt im nächsten infoRUM. Lösungen können an mich geschickt werden (URZ31 at DMSWWU1A), falls jemand nicht bis dann warten kann.

Wer sich den Effekt visuell veranschaulichen möchte, kann sich (vorzugsweise auf der C-Maschine mit Hilfe der IBM 5080) des folgenden Programms bedienen:

```
QUIZ: PROC OPTIONS (MAIN);
CALL START_TERMINAL (DUMMY, DUMMY);
DO UNTIL (REPLY = 'N');
   CALL CELL_ARRAY (10,10,60,60,A);
   CALL UPDATE_TERMINAL;
   A = (A < 3) * (A + 1) + (A = 3);
   DISPLAY ('AGAIN (Y/N)?')
      REPLY (REPLY);
   CALL CLEAR_TERMINAL;
   END;
CALL END_TERMINAL;
DECLARE A (8,8) BIN FIXED
   INIT ((21)(1,2,3),1),
   DUMMY BIN FIXED,
   REPLY CHAR VAR;
%INCLUDE GKS, GKSTERM;
END;
```

## Neue Entwicklungen in der PC-Welt

von

*Hans-Werner Kisker*

Anfang April wurden einige neue Arbeitsplatzrechner in den Markt eingeführt, die nur teilweise an die bisher verbreiteten Linien anknüpfen. Dieser Artikel soll in kurzer Form die wesentlichen Neuerungen zusammenfassen.

*Zwei neue Systeme*

Vorgestellt wurden zwei neue Systeme, die sich spezifizieren lassen durch die jeweils eingesetzten Prozessoren. Eine Linie verwendet den 8086- Prozessor, die andere den 80286- oder auch 80386-Prozessor. Die zweite Linie deckt natürlich ein weites Leistungsspektrum mit durchaus unterschiedlichen Prozessorarchitekturen ab, muß aber aus konzeptioneller Sicht zumindest zum jetzigen Zeitpunkt als eine Linie gesehen werden. Einige Änderungen treffen für beide Linien gleichermaßen zu:

- Die Systemplatinen sind standardgemäß mit den erforderlichen Schnittstellen ausgerüstet. Hierzu gehören Speicher, serielle und bidirektionale parallele Schnittstellen, Disketten und Festplatten-Controller, Mausanschluß, Uhr und Bildschirm-Controller.
- An die Bildschirm-Controller können nur noch Monitore mit analoger Ansteuerung angeschlossen werden. Dadurch wird eine größere Anzahl von Farben bzw. Graustufen möglich. Die alten Monchrom-, Color- oder EGA-Monitore sind nicht mehr verwendbar. Für die beiden Linien werden von den eingebauten Bildschirm-Controllern unterschiedliche Leistungen angeboten.
- Es werden standardgemäß 3.5"-Diskettenlaufwerke verwendet. Allerdings mit unterschiedlicher Kapazität für die beiden Linien: 720 KB für die 8086- und 1.44 MB für die 80286-/80386- Rechner. Ein Datenaustausch mit 5 1/4"Disketten ist nur über ein extern anschließbares Laufwerk möglich.
- Die Grundversionen der beiden Linien sind nur mit 20 MB Festplatten mit einer Zugriffszeit von 80 ms ausgestattet. Dies ist ein sehr zu bedauernder Rückschritt gegenüber den schnellen Platten der bisherigen AT-Versionen. In der 8086 Linie gibt es auch ein reines Diskettengerät.

*Die 8086-Linie*

Die 8086-Linie kann als Nachfolger der bisherigen PC-, XT- und AT-Rechner angesehen werden. Die meisten der durchaus sinnvollen Erweiterungen gehen in die Richtung, die von anderen Rechnern auch vorher schon realisiert wurde.

- Der Prozessor 8086 läuft mit 8 Mhz. Waitstates werden nicht eingeschoben.
- Die Systemplatine ist mit 640 KB Speicher voll ausgebaut. Die Datenwege zum Speicher sind 16 Bit breit.
- Es stehen drei PC/XT-kompatible Steckplätze zur Verfügung. Der externe Daten-Bus ist also 8 Bit breit.
- Der integrierte Bildschirm-Controller bietet einen zum bisherigen Color-Grafik-Adapter kompatiblen Modus. Es werden zusätzliche Auflösungen bis hin zu 640x480 Punkten geboten. Der EGA-Standard wird leider nicht unterstützt.
- Als Betriebssystem steht DOS 3.3 zur Verfügung, eine unwesentliche Erweiterung von DOS 3.1/3.2.

Von der Einschätzung der Verarbeitungsgeschwindigkeit her ist dieser Rechner wohl bei den bisherigen AT-Rechnern einzuordnen. Leider trübt die langsame und in der Kapazität zu schwach ausgelegte Festplatte das Bild. Die Auflösung des Bildschirmadapters ist nur dann akzeptabel, wenn man die neuen Modi benutzt. Für bereits vorhandene Software muß man sich mit der groben Color-Grafik-Auflösung begnügen, auch wenn das Programm die EGA-Modi unterstützen würde.

*Die 80286/80386-Linie*

Bei diesen Modellen handelt es sich um Rechner mit einer weitgehend neuen Architektur, die für Multi-Tasking- und Mehrprozessoranwendungen optimiert sein soll.

- Die Prozessoren laufen mit unterschiedlichen Taktraten; dies reicht vom 10-MHz-80286 mit einem Waitstate bis zum 20-MHz-80386.
- Die Systemplatine ist standardgemäß mit 1 oder 2 MB Hauptspeicher ausgestattet. Der Speicher kann bis zur Größe von 22 MB ausgebaut werden.
- Die Rechner verfügen über ein neu ausgelegtes, zum bisherigen Standard inkompatibles Bus-System mit 16 Bit Daten-/24 Bit Adressen-Bus beim 80286 und 32 Bit Daten und Adressen beim 80386. Erweiterungskarten für die bisherigen Rechner können nicht genutzt werden.

- Der integrierte Bildschirm-Controller bietet einen zum bisherigen EGA-Standard kompatiblen Modus. Es werden zusätzliche Auflösungen bis hin zu 640x480 Punkten geboten.
- Die Nutzung der Möglichkeiten der neuen Rechner ist nur von dem neu eingeführten Multi-Tasking-Betriebssystem mit Namen Betriebssystem/2 möglich. DOS-Applikationen können unter diesem Betriebssystem als eigene Task ablaufen. Bei Bedarf ist auch DOS 3.3 direkt ablauffähig.

*Zusammenfassung*

Die 8086-Linie scheint weitgehend kompatibel zu der jetzigen Rechnergeneration zu sein. Der Markt für diese Rechner hat sich also nicht wesentlich geändert. Die für uns wünschenswerte Vielfalt von Herstellern ist erhalten geblieben. Ob sich die neuen Bildschirme durchsetzen werden, bleibt abzuwarten. Im Moment ist wohl der vorhandene EGA-Standard vorzuziehen.

Die 80286/80386-Linie ist dagegen neu und von ihrer Leistungsfähigkeit her auch durchaus interessant. Ihre Rechner sind aber wegen ihrer mangelnder Kompatibilität im Moment nicht in die von uns empfohlene Rechnerumgebung einzubinden. Ob sie sich auf Dauer durchsetzen werden oder ob sie eine Außenseiterrolle wie der 3270 PC, der 370 PC oder der PC RT spielen werden, hängt davon ab, wie viele Mitbewerber diese Linie mittragen werden. Für uns werden diese Rechner erst interessant, wenn diese Marktvielfalt gegeben ist. Für den Augenblick erscheint mir eine abwartende, beobachtende Haltung geboten.

## Ein elektronischer Kalender im CMS

von

*F. Budde*

Ist es Ihnen auch schon passiert, daß Sie einen wichtigen Termin verpaßt haben? Haben auch Sie EXECs oder Befehle, die regelmäßig an bestimmten Tagen ausgeführt werden sollen? Bei diesen und ähnlichen Problemen kann Ihnen die neue Funktion des elektronischen Kalenders weiterhelfen.

Diese Kalenderfunktion besteht im wesentlichen aus zwei Befehlen und einer Datei mit der Bezeichnung `userid DATES`. Diese Datei enthält Informationen über Ihre Termine. Sie wird mit dem Befehl DATES ediert. Das Kommando CALENDAR durchsucht dann die Datei und gibt Meldungen über Ihre Termine und/oder führt von Ihnen angegebene Befehle aus. Wie geht dies nun vonstatten?

Nachdem Sie den Befehl DATES eingegeben haben, gelangen Sie in ein Menü, das dem des NAMES-Kommandos sehr ähnlich ist. Sie müssen nun mehrere Felder ausfüllen, um einen Termin zu beschreiben. Am wichtigsten ist die Angabe des Datums, es wird hinter „`Date:`" eingetragen. Sie können nicht nur ein Datum, sondern auch wöchentliche, monatliche oder jährliche Termine angeben. Selbst Termine an bestimmten Wochentagen können dort spezifiziert werden.

Im Feld darunter geben Sie an, ab wieviel Tagen vor dem Termin Sie erinnert werden möchten. Wenn Sie nun noch in die Zeilen auf dem unteren Teil des Bildschirms, die mit einem „:" beginnen, eine Nachricht schreiben, die dann ausgedruckt wird, um Sie an den Termin zu erinnern, so haben Sie schon einen Termin vollständig beschrieben und können ihn nun mit der Funktionstaste PF2 in Ihrer Datei `userid DATES` abspeichern.

Die übrigen Felder auf dem Bildschirm dienen der weiteren Beschreibung Ihres Termins. Man kann z. B. die Nachricht hellerleuchtet ausdrucken lassen, falls es sich um einen besonders wichtigen Termin handelt, oder man kann zusätzlich oder alternativ zum Ausdrucken der Nachricht auch ein EXEC oder CMS-Kommando ausführen lassen, indem man das Feld „`Command or Exec:`" ausfüllt. Weitere Informationen darüber liefert das Kommando HELP DATES.

Wenn Sie nun den Befehl CALENDAR ausführen, so wird Ihre Datei nach Terminen durchsucht, und es wird geprüft, ob auf einen Termin reagiert werden muß. Soll eine Nachricht ausgegeben werden, so erscheint sie zusammen mit dem Termindatum auf dem Bildschirm. Soll ein Kommando ausgeführt werden, so wird der hinter „`Command or Exec:`" eingetragene Befehl an das CMS übergeben. Einträge für Termine, die schon vergangen sind, werden dabei vom Programm automatisch gelöscht.

Es ist sinnvoll, den Befehl CALENDAR im PROFILE EXEC aufzurufen. So ist man immer schon bei Beginn der Sitzung über seine Termine informiert.

Probieren Sie doch einmal folgendes kleine Beispiel aus. Tragen Sie unter „Date:" einfach „*/*/*" ein. Dies ist eine Datumsspezifikation, die stellvertretend für jeden Tag steht. Nun tragen Sie im Feld darunter „0" ein. Dies bedeutet, daß Sie nur am Tag des Termins erinnert werden wollen. Zum Schluß schreiben Sie noch eine kurze Nachricht in die Felder, die mit „:" beginnen, speichern Ihren Termin-Eintrag und verlassen das Menü.

Wenn Sie nun CALENDAR aufrufen, sollte Ihre Nachricht zusammen mit dem aktuellen Datum auf dem Bildschirm erscheinen.

Ich hoffe, daß Ihnen dieses Programm ein wenig hilft, Ihre Termine zu planen und einzuhalten. Wenn Sie Anregungenen oder Kritik haben, senden Sie diese bitte an mich (URZ66 at DMSWWU1A).

## Neues vom NAG und IMSL

von

*B. Süselbeck*

### *NAG*

NAG hat die MARK 12 FORTRAN LIBRARY für dieses Jahr angekündigt, die die im Moment im Rechenzentrum im Einsatz befindliche Version MARK 11 ablösen wird.

Gegenüber MARK 11 wird es eine ganze Reihe von Änderungen geben, u. a.:

- Übergang auf FORTRAN 77,
- modifizierte Fehlerbehandlung,
- verändertes Kapitel X02 (Maschinenkonstanten),
- ein neues Kapitel F06 mit 97 Hilfsroutinen aus dem Bereich der linearen Algebra,
- Revision aller Beispielprogramme,
- Unterstützung von Vektorrechnern und
- vereinfachte Struktur in der Dokumentation.

Wie bei jeder Umstellung auf eine neuen Version, werden auch beim Übergang von MARK 11 nach Mark 12

- einige alte Routinen wegfallen,
- neue Routinen hinzugefügt,
- der geplante Wegfall von Routinen im nächsten Release angekündigt und
- eine Reihe von Verbesserungen in bestehenden Routinen durchgeführt.

Da an dieser Stelle nicht alle 175 neuen Routinen besprochen werden können, sei auf einen Aushang im Terminalraum, der auf die geplanten Änderungen ausführlich eingeht, hingewiesen.

### *IMSL*

IMSL hat ebenfalls eine neue Version der FORTRAN-Library angekündigt. Es handelt sich um die EDITION 10, als Nachfolger der zur Zeit benutzten EDITION 9.

Die Verbesserungen umfassen:

- 150 neue Unterprogramme,
- Erweiterungen in den Kapiteln über Optimierung, Integration und Interpolation,
- neue Möglichkeiten in den Statistik-Routinen,
- intensive Nutzung von BLAS (Basic Linear Algebra Subroutines),
- verbesserte Fehlerbehandlung,
- automatisches Anlegen von Speicherplatz und
- eine neue, sieben Bände umfassende Dokumentation.

Nach Abschluß der Implementationen und Tests für die neuen Versionen (voraussichtlich zweite Jahreshälfte 1987), wird darüber in NEWS und infoRUM berichtet.

## Neues vom SAS

von

*S. Zörkendörfer*

Sowohl im Stapelbetrieb MVS wie im Dialogbetrieb CMS ist SAS 5.16 die aktuelle Version. Bereits mit der Version 5.15P (vgl. infoRUM Nr. 4/1986) konnte ich die Farbgrafiken der Produkte SAS/GRAPH und SAS/IML außer auf der IBM 3279 (im Hörsaal M4 mit Großbildprojektion) auch auf der IBM 3179G vorführen. Aktuell ist zu berichten, daß das Rechenzentrum mit dem Color Jetprinter IBM 3852-2 nun über eine Hardcopy-Ausgabe für den Farbgrafik-Bildschirm IBM 3179G verfügt. Es dauert ca. fünf Minuten, den Bildschirminhalt auf Papier oder Folie zu kopieren; die Bildgröße ist etwa DIN A5. Das Rechenzentrum bemüht sich

derzeit um eine betriebliche Regelung, wie diese Kombination Farbgrafikbildschirm-Rasterplotter den Benutzern zugänglich gemacht werden kann. Die Grafik-Prozeduren des SAS werden in die SAS-Lehrveranstaltungen des Sommersemesters einbezogen.

## Neues vom SPSS

von

*S. Zörkendörfer*

Die Version 2.2 des SPSS$^X$ hat die Vorgängerversion 2.1 abgelöst, mit dem Aufruf „EXEC SPSSX" wird nun diese neue Version aufgerufen. Die Neuerungen lassen sich über das INFO-Kommando des SPSS$^X$ erfragen; als wesentliche Bereicherung im Umfeld der Variablen-Transformation wurde eine Prozedur AUTORECODE zugefügt, die vorkommende Variablenwerte in eine Folge natürlicher Zahlen transformiert (und dabei sinnvolle Etiketten für Variablen und Werte erzeugt). Die Benutzung von AUTORECODE mag also angezeigt sein, bevor eine Variable im „integer mode" in eine Prozedur eingeht.

Mit der Version 2.2 wurde – wie bereits im letzten inforum vorangekündigt – das Programmprodukt LISREL VI (Linear Structural Relationships, Pfadanalyse) angemietet und als Userproc zum SPSS$^X$ angebunden. Hinweise auf Handbücher zu dieser Prozedur sind durch das SPSS$^X$-Kommando INFO LISREL abzurufen.

Kurzfristig stand vom SPSS die Version SPSS/PC + für IBM-PCs und Kompatible zur Erprobung zur Verfügung. Vor allem Fachbereiche mit CIP-Pools bekundeten Interesse. Zur Koordinierung entsprechender Anschaffungen gebe ich gerne Auskünfte, das Rechenzentrum selbst bietet SPSS derzeit nur am Großrechner an.

## Personalia

Als studentische Mitarbeiter sind zum 1.3.1987 die Herren St. Möller, St. Althoff und St. Arntzen eingestellt worden.

Ausgeschieden als studentische Mitarbeiter sind zum 28.2.1987 Herr M. Kasubke und zum 31.3.1987 Herr H.-G. Wittkemper.

## SLASH – das kleinste Formatierungssystem der Welt

von

*E. Sturm*

Für den, der auch bei der Textaufbereitung nicht mit Kanonen auf Spatzen schießen will, z. B. bei einer nur für den Bildschirm gedachten Dokumentation die Hilfe von DCF zu bemühen, sei hier eine Kleinigkeit vorgestellt, die man eher als Zwille bezeichnen könnte. Sie leistet gute Dienste beim Erstellen von Notes, sofern diese eine gewisse äußere Form haben sollen, oder z. B. bei Programm-Dokumentationen.

Eine wesentliche Aufgabe fällt im Laufe der Erstellung einer einfachen Textdatei immer wieder an: nach irgendwelchen mit dem SPLTJOIN-Kommando bewerkstelligten Einfügungen sieht der Text ziemlich zerrisssen aus. Sollten gar ganze Absätze eingerückt werden, hätte man doch lieber DCF genommen.

Hier empfiehlt sich das SLASH-Kommando (in REXX für XEDIT geschrieben), das mit einem einzigen Steuerzeichen auskommt, dem umgekehrten Schrägstrich (\), und auch der ist nicht unbedingt notwendig. Will man z. B. 72-spaltigen Text anfertigen, so sollte man im XEDIT das Kommando

```
set pf1 slash 72
```

absetzen. Ab jetzt kann man einfach durch Drücken der PF1-Taste einen Absatz formatieren, genauer gesagt, ab der Zeile, in der der Cursor steht bis zur nächsten Leerzeile. Der Text erscheint linksbündig so, daß möglichst viele Wörter in jede Zeile gepackt werden. Trennungen muß man ggf. selbst machen.

Der umgekehrte Schrägstrich kann jetzt noch dazu verhelfen, Absätze oder Teile von ihnen einzurücken. Erscheint er in einer Zeile, so wird der Text links davon unverändert gelassen ("as-is"). Der Text rechts davon gilt als Fließtext, kann also ggf auf die nächste Zeile geschoben werden. Außerdem wird ab dieser nächsten Zeile an der Position hinter der \-Spalte eingerückt ("hanging indent")

Man sollte also folgendermaßen vorgehen (Back-Slashes wurden absichtlich stehengelassen):

```
1)\eine PF-Taste auf das
   SLASH-Kommando mit der
   gewünschten Länge setzen,
2)\ggf. umgekehrte Schrägstriche
   an den betreffenden Stellen
   einsetzen,
3)\den Cursor in die Zeile bewegen,
   in der mit der Formatierung
   begonnen werden soll und
4)\die oben gewählte PF-Taste
   drücken.
```

Es gibt noch ein paar selbstverständliche Regeln:

- Das erste Wort eines Absatzes bleibt dort, wo es steht, damit evtl. gewünschtes Einrücken zu Beginn eines Absatzes erhalten bleibt.
- Steht der Cursor in einer Zeile ohne Slash, wird in der Vorzeile nachgeschaut, wie verfahren werden soll. Ist dort auch kein Slash zu finden, wird wie dort eingerückt.
- Eine Zeile, in der ein Slash am Ende steht, gilt als As-Is-Zeile und unterbricht die Absatzformatierung. Danach beginnt per definitionem ein neuer Absatz.

Als Abschluß noch das allgemeine Format des Slash-Kommandos:

```
SLASH laenge [([Justify] [Null] ]
```

`laenge` bezeichnet die Anzahl der bei der Formatierung zu verwendenden Spalten. Gibt man `Justify` an, so wird auch noch rechtsbündig formatiert. Bei `Null` nimmt das Programm an, daß bei den Zeilen, die in Spalte 1 anfangen, deren Vorzeile aber nicht in Spalte 1 beginnt, ein Slash in „Spalte 0/" angenommen werden soll. Diese schreckliche Definition soll ermöglichen, daß man mitten in einem eingerückten Absatz wieder in Spalte 1 beginnen kann.

Sollten Sie nicht wissen, wie Sie am Ende alle Schrägstriche wieder los werden sollen, so hilft das Kommando SLASH OFF.

Lob bitte an URZ31 at DMSWWU1A.

## Auswahl der richtigen Programmiersprache

von

*A. Achilles*

Wer kennt sie nicht, die Qual der Wahl der richtigen Programmiersprache?! Die Informatiker D Solomon und D. Rosenblueth haben sich darüber tiefe Gedanken gemacht, die keinem Leser von inforum vorenthalten werden sollen. Natürlich sind diese Gedanken – in bester Manual-Tradition – in amerikanischem Englisch niedergelegt. Den folgenden Beitrag der beiden Autoren fand ich in SIGPLAN Notices, 21.9. Ich wünsche viel Vergnügen bei der Lektüre und hoffe, daß der Artikel Ihnen, lieber Leser, bei der Entscheidung hilft, die richtige Auswahl zu treffen:

With such a large selection of programming languages, it can be difficult to choose one for a particular project. Reading the manuals to evaluate the languages is a time-consuming process. On the other hand, most people already have a fairly good idea of how various automobiles compare So in order to assist those trying to choose a language, we have prepared a chart that matches programming languages with comparable automobiles:

**Assembler** A Formula I race car. Very fast, but difficult to drive and expensive to maintain

**FORTRAN II** A Model T Ford. Once it was king of the road.

**FORTRAN IV** A Model A Ford.

**FORTRAN 77** A six-cylinder Ford Fairlane with standard transmission and no seat belts

**COBOL** A delivery van. It's bulky and ugly, but it does the work.

**BASIC** A second-hand Rambler with a rebuilt engine and patched upholstry. Your dad bought it for you to learn to drive. You'll ditch the car as soon as you can afford a new one.

**PL/I** A Cadillac convertible with automatic transmission, a two-tone paint job, whitewall tires, chrome exhaust pipes, and fuzzy dice hanging in the windshield.

**C** A black Firebird, the all-macho car. Comes with optional seat belts (lint) and optional fuzz buster (escape to assembler).

**ALGOL 60** An Austin Mini. Boy, that's a small car.

**Pascal** A Volkswagon Beetle. It's small but sturdy Was once popular with intellectuals

**Modula-2** A Volkswagon Rabbit with a trailer hitch

**ALGOL 68** An Aston Martin. An impressive car, but not just anyone can drive it.

**LISP** An electric car. It's simple but slow. Seat belts are not available.

**PROLOG/LUCID** Prototype concept-cars.

**Maple/MACSYMA** All-terrain vehicles.

**FORTH** A go-cart.

**LOGO** A kiddie's replica of a Rolls Royce. Comes with a real engine and a working horn.

**APL** A double-decker bus. It takes rows and columns of passengers to the same place all at the same time. But, it drives only in reverse gear, and is instrumented in Greek.

**Ada** An army-green Mercedes-Benz staff car. Power steering, power brakes and automatic transmission are all standard. No other colors or options are available. If it's good enough for the generals, it's good enough for you. Manufacturing delays due to difficulties reading the design specification are starting to clear up.

RUM-Graphik

## Beschriftungen für COLPLOTs

von

*E. Sturm*

Benutzer des Farbrasterplotters hatten bisher – abgesehen von Ausfällen nach längeren Standzeiten – vor allen Dingen ein Problem: Die Bilder sehen ja toll aus, aber wenn man doch bloß die Beschriftungen nicht hinterher mit der Schreibmaschine anbringen müßte! All denen kann jetzt geholfen werden.

Zusätzlich zu den in infoRUM Nr. 2/1986 und Nr. 4/1986 vorgesehenen Dateien MAT und COL (im CMS Filetype, im MVS DD-Name) kann bei Bedarf eine TXT-Datei bereitgestellt werden. Diese ist wie die COL-Datei eine mit XEDIT bearbeitbare Klartext-Datei, in der nähere Angaben zu Lage und Art von Beschriftungen vorzusehen sind.

In jeder Zeile der TXT-Datei steht nur eine Angabe in der Form

```
Schlüsselwort Wert Kommentar,
```

wobei die drei Felder durch mindestens ein Blank voneinander zu trennen sind. Steht in Spalte 1 ein Sternchen (*), so wird die gesamte Zeile als Kommentar aufgefaßt; gleiches gilt für völlig leere Zeilen. Alle Angaben gelten so lange, bis sie durch eine Angabe mit gleichem Schlüsselwort geändert werden.

Hier nun die Liste der möglichen Schlüsselworter mit ihren impliziten Werten:

```
LINE       20
COLUMN     20
HEIGHT     10
TEXT       'HALLO!'
COLOUR     1
SHIELD     -1
FONT       1
DIRECTION  0
```

(Die Setzung für TEXT ist natürlich nicht implizit!) Die Zeilen- und Spaltenangaben (LINE, COLUMN) sowie die Zeichenhöhe (HEIGHT) beziehen sich auf das bei COLPLOT übliche Raster von 528 Spalten und entweder 400 Zeilen (DIN A4) oder 800 Zeilen (DIN A3). Mittels der TEXT-Angabe kann man die Beschriftung spezifizieren, als Position gilt die linke untere Ecke. Die Schrift selbst ist übrigens in einem feineren Raster als die Farbmatrix, so daß für kleine Schrift wegen der dann kaum möglichen additiven Farbmischung möglichst eine der acht Grundfarben weiß, schwarz, rot, grün, blau, magenta, cyan oder gelb gewählt werden sollte. Dies geschieht über die Angaben COLOUR und SHIELD, jeweils als Farbindex der COL-Datei. Während COLOUR die Farbe der Zeichen angibt, definiert SHIELD die Farbe eines eventuell gewünschten Freistellungsraumes um den Text herum. Ein negativer Wert für SHIELD bedeutet: kein Freistellungsraum. Mit FONT ist einer der vom GKS her bekannten Zeichensätze gemeint (siehe infoRUM Nr. 3/1983). Umlaute kann man auch dadurch spezifizieren, daß man ein # vor den „verwandten" Buchstaben stellt (z. B. #s für ß). Die Schreibrichtung (DIRECTION) ist einfacher als im GKS gehalten: es gibt nur 0 (waagerecht), 1 (senkrecht, zu lesen, wenn man den Kopf nach links neigt), 2 (auf dem Kopf) und 3 (senkrecht, Kopf nach rechts geneigt).

Normalerweise wird die Farbmatrix der MAT-Datei in die Mitte des Bildes gerückt, was bei DIN A4 an der IBM 5080 heißt, in der oberen Hälfte des Schirms. Möchte man die Matrix selbst positionieren, so gibt es noch ein weiteres Schlüsselwort, das man am besten in der folgenden Weise in die TXT-Datei eingibt:

```
LINE   150
COLUMN 10
MATRIX
```

(MATRIX ist ausnahmsweise ein Schlüsselwort ohne Wertangabe.) Durch diese Anweisungen wird die linke obere Ecke der Farbmatrix in Zeile 150, Spalte 10 gebracht.

Anregungen und weitere Wünsche bitte an mich (URZ31 at DMSWWU1A).

## RUM-Lehre

### Lehrveranstaltungen im 1. Halbjahr 1987

#### 1. Einführende Veranstaltungen im Sommersemester 1987

| | | |
|---|---|---|
| 320065 | Einführung in die EDV<br>Di 15-17<br>Hörsaal: M5, Beginn: 28.4.1987 | *Held, W.* |
| 320070 | Programmieren in FORTRAN<br>Mi, Fr 13-15<br>Hörsaal: M5, M4, Beginn: 29.4.1987 im M5 | *Mertz, K.-B.* |
| 320084 | Programmieren in FORTRAN<br>Mi 13-15<br>Hörsaal: M3, Beginn: 22.4.1987 | *Neukäter, B.* |
| 320099 | Programmieren in Pascal<br>Mo 8-10<br>Hörsaal: M2, Beginn: 27.4.1987 | *Benduhn-Mertz, A.* |
| 320103 | Programmieren in Pascal<br>Mo 15-17<br>Hörsaal: M3, Beginn: 27.4.1987 | *Lange, W.* |
| 320118 | Programmieren in PL/I<br>Mi 9-11<br>Hörsaal: M5, Beginn: 22.4.1987 | *Mersch, R.* |
| 320122 | Das Statistical Analysis System SAS<br>Mo 15-17<br>Hörsaal: M4, Beginn: 27.4.1987 | *Steinhausen, D./<br>Zörkendörfer, S.* |
| 320137 | *) Textverarbeitung mit Mikrorechnern<br>Do 13-15<br>Hörsaal: M5, Beginn: 23.4.1987 | *Kamp, H.* |
| (122042) | *) English for Computer Science<br>Di 16-18<br>Hörsaal: M6, Beginn: 28.4.1987 | *Gress, E.* |

*) Wegen der Begrenzung der Teilnehmerzahl ist für diese Lehrveranstaltung eine frühzeitige Anmeldung im Dispatch des Rechenzentrums erforderlich.

## 2. Weiterführende Veranstaltungen im Sommersemester 1987

| | | |
|---|---|---|
| 320141 | Programmieren in PL/I für Fortgeschrittene<br>Mi 15-17<br>Hörsaal: M5, Beginn: 29.4.1987 | *Sturm, E.* |
| 320156 | Datenstrukturen und Programmierverfahren in Pascal<br>Di 13.30-15<br>Hörsaal: M4, Beginn: 28.4.1987 | *Bosse, W.* |
| 320160 | Programmieren in C<br>Mi 11-13<br>Hörsaal: M5, Beginn: 6.5.1987 | *Richter, G.* |
| 320175 | Programmieren in Modula-2<br>Mi 15-17<br>Hörsaal: M4, Beginn: 29.4.1987 | *Pudlatz, H.* |
| 320180 | Höhere Programmiersprachen: Prolog<br>Di 13-15<br>Hörsaal: M5, Beginn: 28.4.1987 | *Ost, St./<br>Süselbeck, B.* |
| (057695) | Einführung in LISP<br>Mi 13-15<br>Hörsaal: M6, Beginn: 22.4.1987<br>und 2 Std. Übungen n.V. | *Göttsche, H* |
| 320194 | Unix<br>Mi 13-15<br>Hörsaal: M4, Beginn: 29.4.1987 | *Stockhofe, W.* |
| 320209 | Datenbanksysteme: Theorie und Anwendungen<br>Di 15-17<br>Hörsaal: M4, Beginn: 28.4.1987 | *Achilles, A.* |
| 320213 | Moderne Textverarbeitungssysteme: TEX<br>Mi 9-11<br>Hörsaal: M4, Beginn: 22.4.1987 | *Kaspar, W.* |
| 320228 | Anwendungen in SAS<br>Do 15-16<br>Hörsaal: M4, Beginn: 30.4.1987 | *Zörkendörfer, S.* |
| 320232 | Systemprogrammierung von Mikrorechnern<br>Mi 13-15<br>Hörsaal: M3, Beginn: 29.4.1987 | *Kisker, H.-W.* |
| 320247 | Rechnernetze und ihre Anwendungen<br>Do 13-15<br>Hörsaal: M4, Beginn: 7.5.1987 | *Richter, G./<br>Schachtner, G.* |
| 320251 | Kolloquium über Themen der Informatik<br>Fr 15-17<br>Hörsaal: M5 | *Held, W./<br>die wiss. Mitarbeiter<br>des Rechenzentrums* |
| 320266 | Anleitung zum Einsatz der EDV bei<br>wissenschaftlichen Arbeiten | *die wiss. Mitarbeiter<br>des Rechenzentrums* |

*RUM-Tutorial*

## Bildschirmsteuerung im Dialog mit ISPF

von

*A. Achilles*

In den letzten beiden Ausgaben des Infoyum wurde dargelegt, wie ISPF eingesetzt werden kann, um Bildschirmmasken zu benutzen. Im ersten Artikel (Infoyum Nr. 4/1986) wurde besprochen, wie einfache Bildschirmmasken aufgebaut sind und wie sie von EXEC-Kommandos bzw. von kompilierten Programmen aus aufgerufen werden können. Der letzte Beitrag beschrieb den Aufbau der Bildschirmmasken detailierter und erläuterte die Benutzung von eingebauten Funktionen, die zur Prüfung der eingegebenen Daten und zur Steuerung des Ablaufs eingesetzt werden können. In diesem Beitrag soll zunächst der Zugriff auf Variablen dargestellt werden.

### *Benutzung von Dialogvariablen*

Dialogvariablen dienen zur Kommunikation zwischen den verschiedenen Komponenten eines Dialogs. Module, EXEC-Kommandos, Bildschirmmasken, ISPF-Meldungen und -Tabellen usw. können vermittels Dialogvariablen auf die gleichen Daten zugreifen. Der Wert einer Dialogvariablen ist eine Zeichenkette, deren Länge zwischen 0 und 32 KByte variieren kann (manche Komponenten beschränken jedoch die Länge). Der Zugriff erfolgt über den Namen der Variablen. Der Name besteht aus mindestens einem, höchstens acht Zeichen (in FORTRAN nur sechs), dabei sind die Buchstaben A-Z, die Ziffern 0-9 und die Sonderzeichen #, $ bzw. @ erlaubt. Das erste Zeichen darf allerdings keine Ziffer sein.

Die Dialogvariablen sind in Gruppen organisiert, abhängig vom Dialog und der Anwendung. Eine Anwendung besteht aus einem oder mehreren Dialogen, die mit der gleichen Anwendungskennung gestartet werden. Dies kann beim ersten Aufruf gesteuert werden durch den Zusatz **NEWAPPL(anwendung)**, wobei anwendung eine Zeichenkette ist, die aus maximal vier Zeichen besteht:

```
                [ PANEL(name) ]
ISPSTART        [ CMD(name)   ]
                [ PGM(name)   ]

                      NEWAPPL(anwendung)
```

oder beim Aufruf einer neuen Komponente des Dialogs:

```
                      [ PANEL(name) ]
ISPEXEC SELECT        [ CMD(name)   ]
                      [ PGM(name)   ]

                      NEWAPPL(anwendung)
```

Wird auf den Zusatz NEWAPPL verzichtet, so wird automatisch **NEWAPPL(ISP)** eingesetzt. Die Aufrufe wurden in ihrer einfachsten Form bereits beschrieben (Infoyum Nr. 4/1986).

Eine Gruppe (s. o.) kann als Liste von Variablennamen angesehen werden, mittels derer ISPF auf die zugehörigen Werte zugreifen kann. Wenn bei der Bearbeitung auf eine Dialogvariable zugegriffen werden soll, durchsucht ISPF die folgenden drei Gruppen (Pools) in der angegebenen Reihenfolge:

1. **Funktionsgruppe:** diese Gruppe enthält Variablen, die nur innerhalb der aktiven Funktion bekannt sind.
2. **Gemeinsame Gruppe:** hierin werden die Variablen geführt, die von Dialogen, die zur gleichen Anwendung gehören, erreicht werden können.
3. **Anwendungs-Profile:** die hierin geführten Variablen, die zur genannten Anwendung gehören, werden automatisch für den Benutzer für die nächste Sitzung aufbewahrt.

Auswahl-Menüs sind eigenständig und nicht als Bestandteil einer Dialogfunktion anzusehen. Werden Dialogvariablen in Menüs angesprochen, so werden sie nur in der gemeinsamen Gruppe bzw. im Anwendungs-Profile gesucht. Wird jedoch eine Dialogfunktion aufgerufen, z. B. ein EXEC-Kommando oder ein kompiliertes Programm, so wird beim Aufruf eine dieser Funktion zugeordnete Funktionsgruppe angelegt. Soll eine andere Dialogfunktion oder ein Menü auf die Werte dieser Variablen zugreifen können, so müssen diese erst

mittels eines Aufrufs von VPUT in die gemeinsame Gruppe oder das Anwendungs-Profile kopiert werden (auch bei gleicher Namensgebung sind Variablen zweier verschiedener Dialogfunktionen unterschiedliche Variablen!):

```
                        [ ASIS    ]
ISPEXEC VPUT liste      [ SHARED  ]
                        [ PROFILE ]
```

In `liste` können die zu kopierenden Variablen aufgezählt werden. ASIS ist als Option voreingestellt. Diese Option bewirkt, daß die Variablen in diejenige Gruppe (gemeinsame Gruppe oder Anwendungs-Profile) kopiert werden, in der sie bereits angelegt sind. Wird eine Variable neu erzeugt oder ist sie in beiden Gruppen bekannt, so wird sie nur in die gemeinsame Gruppe kopiert. SHARED bewirkt, daß die Variablen in die gemeinsame Gruppe kopiert werden. Entsprechend bewirkt PROFILE ein Kopieren in das Anwendungs-Profile, zugleich werden dabei aber Variablen gleichen Namens in der gemeinsamen Gruppe gelöscht.

Andere Funktionen können dann mit Hilfe eines VGET-Aufrufs Variablen aus der gemeinsamen Gruppe bzw. dem Anwendungs-Profile in die Funktionsgruppe kopieren:

```
                        [ ASIS    ]
ISPEXEC VGET liste      [ SHARED  ]
                        [ PROFILE ]
```

Die Verwendung der Optionen entspricht dabei der Benutzung im VPUT-Aufruf. Es ist jedoch zu bemerken, daß bei Benutzung von PROFILE Variablen gleichen Namens in der gemeinsamen Gruppe auch dann gelöscht werden, wenn sie im Anwendungs-Profile nicht gefunden werden.

Ist die Dialogfunktion ein EXEC-Kommando, so sind die vom Kommando-Interpreter und von ISPF benutzten Listen der Funktionsvariablen identisch. Jede Variable ist somit automatisch eine Dialogvariable, jede in der Funktionsgruppe eingetragene Variable kann direkt im EXEC-Kommando angesprochen werden (allerdings dürfen Variablennamen im ISPF acht Zeichen nicht überschreiten).

Da Programme im Gegensatz zu EXEC-Kommandos kompiliert werden, können sie keine gemeinsame Variablenliste mit ISPF teilen. Hier muß VDEFINE benutzt werden, um die Dialogvariablen überhaupt in die Funktionsgruppe einzutragen und ISPF zugänglich zu machen:

```
CALL ISPLINK('VDEFINE',name,
             variable,format,laenge
             [,optionen] [,userform])
```

Dabei gibt `name` den symbolischen Namen an, unter dem ISPF auf die Variable zugreift, `variable` gibt die Variable innerhalb des Programms an, auf deren Speicherplatz zugegriffen werden soll. `format` bezeichnet das Datenformat der Variablen. Mögliche Werte sind:

**CHAR** Zeichenkette (in PL/I muß eine Zeichenkette für die Verwendung in VDEFINE mit fester Länge deklariert sein). Die Daten werden nicht umgewandelt, es wird im Gegensatz zu den anderen Formaten auch keine Prüfung auf Gültigkeit der Zeichen vorgenommen, wenn ISPF auf den Speicherplatz zugreift.

**FIXED** ganze Zahlen, repräsentiert durch die Ziffern 0-9. Haben solche Variablen einen Speicherplatz von vier Byte, so werden sie als Zahlen mit Vorzeichen behandelt. NULL-Werte (z. B. es ist keine Eingabe erfolgt) werden durch die kleinste negative Zahl (X'80000000') dargestellt. Ist die Länge kleiner als vier Byte, so werden die Variablen als vorzeichenfreie Zahlen interpretiert. NULL-Werte können dann nicht mehr vom Wert 0 unterschieden werden.

**BIT** ist eine Bit-Kette, die durch eine Zeichenkette bestehend aus 0 und 1 repräsentiert wird. In der Variablen werden die Daten linksbündig gespeichert und rechts mit binären Nullen aufgefüllt.

**HEX** ist eine Bit-Kette, die durch eine Zeichenkette, bestehend aus den Ziffern 0-9 und den Buchstaben A-F, repräsentiert wird. In der Variablen werden die Daten linksbündig gespeichert und rechts mit hexadezimalen Nullen aufgefüllt.

**USER** dient dazu, eine selbstdefinierte Konversionsroutine zu vereinbaren.

Der Parameter `laenge` gibt die Länge des Speicherplatzes der Variablen (in Byte) an. (Dieser Parameter muß eine vier Byte lange ganze Zahl sein.)

Als Optionen können einzeln oder gemeinsam

**COPY** zum Initialisieren der Dialogvariablen, wobei die Gruppen in der üblichen Reihenfolge durchsucht werden, und

**NOBSCAN** zum Aufbewahren von abschließenden Leerzeichen

angegeben werden.

Der Parameter `userform` gibt die Eingangsadresse zu einer selbstdefinierten Konversionsroutine an. Dieser Parameter muß immer angegeben werden, wenn als `format` „USER" eingetragen ist.

*Systemvariablen*

Bei der Namensgebung von Dialogvariablen ist zu beachten, daß einige Namen für Systemvariablen reserviert sind. Systemvariablen dienen dazu, spezielle Informationen zwischen dem Dialog und dem den Dialog überwachenden System ISPF auszutauschen. Die Variablen haben folgende Aufgaben:

- Ausgabe von Zeit und Datum,
- Steuerung des Bildschirms und der Funktionstasten,
- „Scrolling" des Bildschirms,
- Tabellen-Dienste (hierüber wird später berichtet),
- Informationen über Dialogfehler,
- Aufruf von Hilfsmenüs,
- Steuerung der Aufrufe und
- allgemeine Funktionen.

Alle Systemvariablen beginnen mit dem Buchstaben Z, aus diesem Grunde sollten Namen von Dialogvariablen, die mit Z beginnen, vermieden werden.

Der Name der jeweiligen Systemvariablen bestimmt nicht nur die Funktion, sondern auch die Eigenschaften. Als Eigenschaften gelten die jeweilige Gruppe, in der die Variable gespeichert werden kann, die Länge des Inhalts der Variablen und die Tatsache, ob sie zur Eingabe, zur Ausgabe, zu beidem oder aber nur zur unveränderbaren Ausgabe von Systemzuständen dient. Systemvariablen, die in der Funktionsgruppe gespeichert sind, können durch EXEC-Kommandos direkt angesprochen werden, bei kompilierten Programmen muß zunächst ein VDEFINE-Aufruf benutzt werden, um die Namen dem Programm bekannt zu machen. Sind die Variablen in einer der anderen Gruppen gespeichert, so müssen von einem EXEC-Kommando aus die Aufrufe VGET bzw. VPUT benutzt werden, um die gewünschte Variable zu erhalten bzw. zu speichern. Für Programme muß durch einen Aufruf von VCOPY zunächst eine Kopie in der Funktionsgruppe erstellt werden, durch einen VDEFINE-Aufruf muß der Name bekannt gemacht werden, das Speichern geschieht mittels VPUT.

**Zeit- und Datums-Variablen:** Alle diese Variablen sind unveränderbare Ausgabevariablen, die in der gemeinsamen Gruppe gespeichert sind. Die Variablen ZDAY, ZMONTH und ZYEAR haben je die Länge von zwei Zeichen und geben den Tag im Monat, den Monat bzw. das Jahr an. ZJDATE ist sechs Zeichen lang und gibt Jahr und Tag in der Form `jj-ttt` an. ZDATE gibt Jahr, Monat und Tag in der Form `jj/mm/tt` an, die Länge beträgt 8 Zeichen. Zur Angabe der Zeit steht ZTIME zur Verfügung. ZTIME zeigt die Stunde und die Minuten an und ist fünf Zeichen lang: `hh:mm`.

**Allgemeine Funktionen:** Bis auf die Variablen ZPLACE und ZPROFAPP gehören alle Variablen dieser Klasse der gemeinsamen Gruppe an. ZPLACE und ZPROFAPP können im Anwendungs-Profile gespeichert werden. Die zunächst erläuterten Variablen sind unveränderbare Ausgabevariablen: die Variable Z steht für die leere Zeichenkette und hat die Länge Null, ZAPPLID gibt in acht Zeichen die Anwendungskennung aus, ZENVIR beschreibt in 32 Zeichen die Umgebung, ZUSER enthält die Benutzerkennung und ist acht Zeichen lang. Bei der Verwendung von APL zählt ZAPLCNT (Länge vier), wie häufig Bildschirmmasken aufgerufen wurden. Die Variablen ZTEMPF und ZTEMPN werden benötigt, um Informationen über den „File-Tailoring-Dienst" bereit zu halten. Die Variable ZVERB ist eine Ausgabevariable, die in acht Zeichen das eingegebene Kommando enthält. ZPLACE ist eine Ein-/Ausgabevariable (sieben Zeichen), in der angegeben werden kann, wo die Kommandozeile angelegt werden soll (ASIS oder BOTTOM), ZPROFAPP ist eine Eingabevariable (acht Zeichen), die den Namen einer Erweiterung des Anwendungs-Profiles enthält.

**Bidlschirm- und Funktionstasten:** Die Variablen ZCOLORS, ZHILITE, ZSCREEN, ZSCREEND, ZSCREENW und ZSPLIT werden in der gemeinsamen Gruppe gespeichert, es handelt sich um unveränderbare Ausgabevariablen. Die übrigen Variablen werden im Anwendungs-Profile gespeichert, bis auf ZKEYS und ZTERM sind diese Variablen Ein-/Ausgabe-Variablen, ZKEYS und ZTERM sind reine Ausgabevariablen. Die einzelnen Variablen und ihre Bedeutung: ZCOLORS

gibt die Zahl der durch das Terminal unterstützten Farben an (1 oder 7, Länge: vier Zeichen), ZHILITE (YES oder NO, drei Zeichen) gibt an, ob „extended highlighting" unterstützt wird, ZSCREEN enthält die logische Bildschirmnummer (1, 2, 3 oder 4, Länge: ein Zeichen), ZSCREEND und ZSCREENW geben in je vier Zeichen die Höhe bzw. Weite des für den Dialog nutzbaren Bildschirms an. ZSPLIT gibt in drei Zeichen an, ob der Bildschirm in mehrere Dialoge geteilt ist (YES oder NO). ZTERM enthält in acht Zeichen den Bildschirmtyp.

ZKEYS gibt die Anzahl der Funktionstasten aus (vier Zeichen), ZPFnn enthält die Setzung der Funktionstaste nn ($1 \leq nn \leq 24$), dazu werden 255 Zeichen benötigt. Die Variable ZPFCTL beschreibt, ob der Dialogbenutzer die Funktionstasten mit dem PFSHOW-Kommando verändern darf (fünf Zeichen), ZPFSHOW beschreibt mit vier Zeichen den Status des PFSHOW-Kommandos. ZPFMT enthält in vier Zeichen, wie viele Funktionstastenbeschreibungen pro Hilfszeile ausgegeben werden, ZPFSET enthält in vier Zeichen, ob die Belegung der Funktionstasten auf dem Bildschirm dargestellt wird.

**Scrolling:** Die Variablen ZSCBR, ZSCED und ZSCML werden im Anwendungs-Profile gespeichert und sind Ein-/Ausgabevariablen. Alle Variablen dieser Klasse sind vier Zeichen lang. Die Variablen ZSCBR, ZSCED bzw. ZSCML bestimmen den Umfang des Scrollens im Browse, Edit bzw. für Member-Listen. Die Eingabevariable ZSCROLLD, die in jeder Gruppe gespeichert werden kann, gibt an, um welchen Betrag „scrollbare" dynamische Bereiche verschoben werden. Die Ausgabevariablen ZSCROLLA und ZSCROLLN, die in der gemeinsamen Gruppe gespeichert werden, beschreiben, um welchen Wert (PAGE, MAX, Zahl usw.) bzw. welche Anzahl von Zeilen (ZSCROLLN wird aus ZSCROLLA berechnet) verschoben werden soll.

**Tabellendienste:** Die Variablen ZTDMAKR, ZTDMSG, ZTDROWS, ZTDSELS bzw. ZTDTOP werden an späterer Stelle beschrieben.

**Dialogfehler:** Alle Variablen dieser Klasse sind Ausgabevariablen, die in der Funktionsgruppe gespeichert sind: ZERRHM gibt in acht Zeichen den Namen der Hilfsmaske an, die mit der Fehlermeldung verbunden ist, ZERRMSG enthält in acht Zeichen die Fehlerkennung, ZERRSM ist 24 Zeichen lang und enthält eine Kurzbeschreibung des Fehlers, ZERRLM enthält eine ausführliche Beschreibung in 78 Zeichen und ZERRALRM gibt in drei Zeichen an, ob der Alarm-Indikator gesetzt ist (YES oder NO).

**Aufrufsteuerung:** In speziellen Bildschirmmasken, den Auswahlmenüs, können die folgenden Eingabevariablen benutzt werden: ZCMD ist für die Eingabe der Kommandozeile vorgesehen, ZSEL enthält die Eingabe des Kommandos bis zum ersten Punkt. In der Kommandozeile (d.h. in ZCMD) kann zum Beispiel mit Hilfe des Schlüsselwortes CMS bzw. CP (beide sind synonym) auch ein CMS-Kommando eingegeben werden, das direkt an das CMS (bzw. CP) weitergereicht wird. ZSEL kann zur Auswertung von Kommandos benutzt werden.

ZPRIM beschreibt, ob es sich um ein Eingangsmenü zu einer Dialoganwendung handelt (YES). Enthält ZPRIM den Wert YES, so bewirkt die Eingabe des RETURN-Kommandos (in der Regel Funktionstaste PF4), daß bis zu diesem Menü zurückgegangen wird. Dieses Menü kann durch das END-Kommando (Funktionstaste PF3) verlassen werden. ZPARENT kann benutzt werden, um die Reihenfolge, in der die Bildschirmmasken bei Aufruf des END-Kommandos durchlaufen werden, zu beeinflussen.

### *Hilfsmenüs*

Auch diese Eingabevariablen können nur in bestimmten Masken, den Hilfsmasken, verwendet werden. Sie dienen dazu, den Benutzer durch mehrere Hilfsmasken, die zu einer Hilfe gehören, zu führen. ZCONT steuert über den eingegebenen Namen, welche Maske als Fortsetzung genommen wird; ZHTOP enthält den Namen der ersten Maske, die zu dieser Hilfsinformation gehört; ZUP enthält den Namen der vorausgehenden Maske, ZIND gibt an, ob es sich um eine Index-Seite handelt (YES) und ZHINDEX enthält den Namen der ersten Index-Seite. Unter Verwendung dieser Variablen können umfangreiche Hilfsinformationen in mehreren zusammenhängenden Bildschirmmasken dargestellt werden.

# Programmverwaltung unter dem Betriebssystem MS-DOS

von

*H.-W. Kisker*

Werden mehrere Programmsysteme auf einer Festplatte vorgehalten, so sind die Files dieser Systeme durch eigene Directories voneinander zu isolieren. Durch die Bereitstellung geeigneter Batch-Files, die die DOS-Kommandos `path` und `append` nutzen, können die Programme dennoch ohne Kenntnis der Directory-Struktur direkt aufgerufen werden.

Im Laufe seiner Entwicklungsgeschichte hat das Betriebssystem MS-DOS immer wieder Erweiterungen erfahren, die es dem Benutzer erleichtern, seine Datenbestände sachgerecht zu verwalten. Insbesondere wurde die Struktur des File-Systems der hierarchischen Ordnung des Betriebssystems UNIX angenähert, und der Vorrat der Systemkommandos wurde um die damit notwendig werdenden Befehle zur Directory-Verwaltung erweitert. Ich möchte mit diesem Artikel einen besonderen Aspekt beleuchten – nämlich die Einbettung von Programmen in die Struktur des File-Systems. Denn obwohl hier Hilfsmittel zur Verfügung stehen, wird diese Einbettung von vielen Anwendern doch nur unvollständig vorgenommen. Eine schwerfällige Handhabung der Programme ist die Folge.

## *Das Problem*

Die weite Verbreitung des preiswerten Massenspeichers „Festplatte" hat die Nutzung auch komplexer Strukturen des File-Systems notwendig gemacht. Denn nur dadurch kann die Übersicht über die dort abgelegten umfangreichen Datenbestände gewährleistet werden. Für die einfache, vom DOS unterstützte Suchstrategie wird es damit immer schwieriger, Programme und Dateien aufzufinden. Als Folge dieser scheinbaren Unzulänglichkeiten ist es inzwischen üblich geworden, daß der resignierende Benutzer dem Betriebssystem sozusagen helfend unter die Arme greift, indem er Files selbst durch explizite Angabe der Directory lokalisiert oder indem er Datenbestände in Directories ablegt, die er an sich programmspezifischen Files vorbehalten möchte. Ziel dieses Artikels ist es, eine Organisationsform aufzuzeigen, die diese lästige Übernahme von Betriebssystemaufgaben durch den Benutzer überflüssig macht.

## *Die Hilfsmittel*

Leitfaden der Darstellung soll die Schilderung der Probleme und Lösungsansätze aus der Sicht des unerfahrenen Umsteigers von einem Diskettengerät auf ein Festplattengerät sein. Dabei möchte ich Grundkenntnisse im Umgang mit MS-DOS und insbesondere der unterstützten Directory-Struktur als vorhanden voraussetzen. Einige andere hier benötigte Hilfsmittel sind vielleicht weniger bekannt und ihre Erläuterung soll deshalb an den Anfang gestellt werden.

**Die Suchstrategie des MS-DOS:** Die Wurzel des Problems liegt in einem Grundverhalten des Betriebssystems. Das DOS nimmt zu jedem Zeitpunkt ein Laufwerk als Standardlaufwerk und zu jedem Laufwerk eine Directory als Standard-Directory an. Ohne weitere Vorkehrungen wird ein auszuführendes Programm oder eine zu bearbeitende Datei nur auf dem Standardlaufwerk in der Standard-Directory gesucht.

Allerdings hat der Benutzer die Möglichkeit, diese Annahmen des Betriebssystems zu beeinflussen.

- Er kann sie dynamisch vorgeben, indem er Standard-Laufwerk und -Directory gemäß seinen momentanen Anforderungen festlegt.
- Er kann sie temporär überschreiben, indem er einen vollen Pfadnamen spezifiziert.
- Er kann sie erweitern, indem er die Kommandos `path` bzw. `append` benutzt (siehe unten).

Darüber hinaus implementieren viele der in neuerer Zeit entwickelten Programme eigene Suchmechanismen für Daten- oder Parameter-Files. Üblich ist es, Environment-Variablen auszuwerten oder es dem Benutzer zu ermöglichen, Directories im Dialog zu spezifizieren. Unser Schwerpunkt liegt aber auf den Programmsystemen, die solche Hilfen nicht anbieten.

**Das Kommando path:** Das Kommando `path` ist wohl vielen Benutzern bekannt. Seine Einführung stellt eine Erweiterung der Suchstrategie des DOS dar, die vom Betriebssystem UNIX übernommen wurde. Es ist ein internes Kommando und hat in der vom DOS her bekannten Notation die allgemeine Form:

```
path d:pathname; ...
```

Mit Hilfe diese Kommandos spezifiziert der Benutzer eine Liste von Directories, die das DOS als potentielle Fundorte für Programme durchsuchen soll. Ausgewertet wird diese Liste immer dann, wenn der Benutzer ein Kommando ohne besondere Pfadangabe aufruft und ein entsprechendes EXE-, COM- oder BAT-File nicht auf dem Standardlaufwerk in der Standard-Directory gefunden wird. Dann werden die Directories der Liste in der angegebenen Reihenfolge nach dem gewünschten Programm durchsucht.

**Das Kommando append:** Die Liste des `path`-Kommandos wird nur bei der Suche nach zu ladenden Programmen ausgewertet. Datenbestände werden hierüber nicht gefunden. Ab der Version 3.10 des Betriebssystems MS-DOS steht allerdings zur Lokalisiereung von Dateien ein eigenes dem `path`-Kommando entprechendes Kommando zur Verfügung – das `append`-Kommando. Es hat die allgemeine Form:

```
append d:pathname; ...
```

Die spezifizierte Liste wird vom DOS immer dann genutzt, wenn eine zu öffnende Datei, für die kein Pfadname angegeben wurde, auf dem Standardlaufwerk in der Standard-Directory nicht gefunden wurde. Seine Zuordnung zum Betriebssystem ist etwas unklar. Bei manchen Firmen wird `append.com` mit MS-DOS ausgeliefert. Andere Firmen rechnen es der Netz-Software zu und stellen es erst mit der Netzwerkerweiterung MS-NET (oder seinen Varianten) zur Verfügung.

### *Die Dateiorganisation auf der Festplatte*

Bei den Nutzern eines Diskettengeräts ist es üblich und auch zu empfehlen, für jedes benutzte Anwendungsprogramm eine eigene Diskette vorzusehen. So gibt es z. B. eine Systemdiskette, eine Textverarbeitungsdiskette oder eine Compilerdiskette. Jede dieser Disketten enthält die zu dem jeweiligem Programmsystem gehörenden Programme und die zur Ausführung benötigten Dateien – wie z. B. Hilfstexte oder Parameterfiles. Will man ein bestimmtes Programm nutzen, so legt man die entsprechende Diskette in das Laufwerk, das man zum Standardlaufwerk gemacht hat, und ruft das gewünschte Kommando auf. Programm und zugehörige Files werden vom DOS auf dem Standardlaufwerk problemlos gefunden.

Mit der Verfügbarkeit preisgünstiger Festplatten für Mikrorechner hat sich die Situation jedoch grundlegend geändert. Zwar ist die oben beschriebene Vorgehensweise auch mit einem Plattengerät noch möglich, normalerweise ist es jedoch vorteilhafter, Programme und Datenbestände auf der schnelleren und stets verfügbaren Festplatte abzulegen. Dementsprechend wird ein Umsteiger von einem Diskettengerät auf ein Festplattengerät es als natürliche Maßnahme ansehen, die Datenbestände seiner Programmdisketten auf die Festplatte zu kopieren. Er kann dann die Festplatte zu seinem Standardlaufwerk machen, und Programme und Daten werden dann vom DOS wie bei der Diskettenlösung problemlos gefunden. In der Tat wird gerade ein Anfänger häufig dazu neigen, eine Festplatte in dem beschriebenen Sinne nur als eine große Diskette anzusehen. Schon bald werden sich jedoch gravierende Beeinträchtigungen bemerkbar machen:

- Die Anzahl der Files, die auf einer Platte untergebracht werden können, ist deutlich größer als bei einer Diskette. Einige hundert Files sind keine Seltenheit. Entsprechend schwierig ist es, den Überblick über die schier endlose Liste, die das `dir`-Kommando auf den Bildschirm ausgibt, zu behalten.
- Die Zuordnung der einzelnen Files zu den Programmsystemen ist nur schwer oder gar nicht mehr zu erkennen. Eine Wartung der Programme wird deutlich erschwert.
- Gehören zu zwei verschiedenen Programmsystemen Files mit gleichem Namen, so bleibt nur die Datei *eines* Programms auf der Platte erhalten. Fehlläufe des anderen Programms sind unvermeidlich.

Die Maßnahmen zur Abhilfe liegen allerdings auf der Hand und sind wohl auch den meisten DOS-Benutzern geläufig. Man wird einfach die Diskettenorganisation durch eine entsprechende Directory-Struktur auf der Festplatte nachbilden. Für jedes eingesetzte Programmsystem wird man also eine eigene Directory anlegen und da hinein die Inhalte der entsprechenden Disketten kopieren. Es gibt dann auf der Platte entsprechend der Diskette neinteilung z. B. eine System-Directory, eine Textverarbeitungs-Directory und eine Compiler-Directory. Dieses Verfahren wird auch durch die Installationsroutinen der Programmsysteme der gehobenen Leistungsklasse unterstützt. Im Verlauf

einer solchen Installation wird nämlich im allgemeinen eine eigene Directory angelegt und in diese die ausführbaren Programme und die zugehörigen programmspezifischen Dateien abgelegt. Zu empfehlen ist es auch, die Datenbestände, die mit den einzelnen Programmsystemen bearbeitet oder verwaltet werden sollen, nicht unter die Files der programmspezifischen Directory zu mischen. Eine Trennung der Benutzer- und Systemdateien wird sonst unnötig erschwert. Die Benutzerdaten werden entweder in einer allgemeinen oder programmspezifischen Arbeits-Directory oder auf einer gesonderten Diskette abgelegt. Darüber hinaus kann es in manchen Fällen sinnvoll sein, auch die ausführbaren Programme und die Systemdateien voneinander durch eigene Directories zu trennen.

Damit sind die einzelnen Programmsysteme und auch die zugehörigen Benutzerdaten sauber voneinander isoliert, die Übersicht ist gewährleistet und auch eine Programmpflege ist leicht durchzuführen. Zusammenfassend seien folgende Punkte für eine sachgerechte Anordnung der Programmsysteme hervorgehoben:

- Für jedes Programmsystem wird eine eigene Directory angelegt.
- In der programmspezifischen Directory werden ausschließlich die jeweiligen ausführbaren Programme und die unmittelbar zum System gehörenden Datenbestände – wie Parameter-Files, Hilfstexte usw. – abgelegt.
- Die mit einem Programmsystem zu bearbeitenden Files werden in einer gesonderten Directory oder auch auf Diskette vorgehalten. Sie werden nicht unter die System-Files gemischt.

### *Die Programmorganisation*

Die oben vorgeschlagene Organisation der Programmsysteme auf der Festplatte bringt für die Verwaltung der Datenbestände offensichtliche Vorteile mit sich. Die Nutzung der Programme wird jedoch auf den ersten Blick erschwert. Wird nämlich ein Programm aufgerufen, so müssen drei Arten von Files vom Betriebssystem lokalisiert werden:

- das auszuführende Programm,
- die systemspezifischen Dateien und
- die Datenbestände des Benutzers

Alle drei können auf verschiedenen Directories, ja auf verschiedenen Laufwerken verteilt sein.

**Ein Beispiel:** Zur Erläuterung sei kurz die File-Organisation des Textverarbeitungssystems TEX, mit dem dieser Artikel entworfen wurde, skizziert. Das ausführbare Programm `tex.exe` zur Dokumentaufbereitung befindet sich auf der Platte C meines Mikrorechners in der Directory `pctex`. Das Programm `tex` benötigt Parameterfiles, die in drei Subdirectories von `pctex` mit Namen `texinput`, `texfmts` und `textfms` untergebracht sind. Den Text meines Artikels habe ich im File `dosfile.tex` abgelegt auf einer eigenen Diskette, die ich normalerweise ins Laufwerk A einlege. Um nun ein Dokument zu erzeugen, schaffe ich mir mit den angegebenen Kommandos die folgende Arbeitsumgebung:

- Das Laufwerk A meines Rechners wird zum Standardlaufwerk gemacht, Standard-Directory ist die `root`-Directory:

```
a:
cd \
```

- Die `path`-Liste setze ich auf die Directory `pctex`:

```
path c:\pctex
```

- In die `append`-Liste werden die Directories `texinput`, `texfmts` und `textfms` aufgenommen:

```
append c:\pctex\texinput;
       c:\pctex\texfmts;
       c:\pctex\textfms
```

(Nach DOS-Konvention muß diese Eingabe jedoch in einer Zeile erfolgen!)

Anschließend kann ich TEX direkt aufrufen durch die Kommandoeingabe:

```
tex dosfile
```

Bei der Abarbeitung dieses Kommandos laufen ineinandergeschachtelt mehrere Suchprozesse ab.

Das Kommando `tex` wird auf dem Laufwerk A nicht gefunden. Deshalb wird die `path`-Liste ausgewertet und das ausführbare Programm `tex.exe` aus der Directory `pctex` des Laufwerks C geladen.

Das Eingabe-File `dosfile.tex` kann vom Standardlaufwerk gelesen werden. Ein Durchsu-

chen der Directories der append-Liste unterbleibt deshalb.

Die Parameter-Files dagegen sind nicht auf der Diskette im Laufwerk A zu finden. Soll eines dieser Files geöffnet werden, muß das DOS deshalb die append-Liste beachten. Von einer der dort angegebenen Directories werden die Files dann gelesen.

Für das Öffnen der Ausgabe-Files spielt die append-Liste keine Rolle. Alle erzeugten Daten werden auf dem Standardlaufwerk in der Standard-Directory abgelegt.

**Grundzüge der Lösung:** Analysiert man die im Beispiel beschriebene Vorgehensweise beim Aufruf eines Programms, so kann man folgenden Grundgedanken festhalten: „Ein Programm, das mit seinen systemspezifischen Files sachgerecht in einer eigenen Directory abgelegt ist, kann unabhängig von der momentanen Wahl des Standardlaufwerks und der Standard-Directory, direkt ohne explixite Pfadangaben aufgerufen werden, wenn die path- und append-Liste auf die programmspezifischen Directories gesetzt ist."

Diesem Grundsatz folgend, ist es ein naheliegender Ansatz, in die path- bzw. append-Liste alle Directories aufzunehmen, die Programme oder programmspezifische Dateien enthalten. Diese simple Lösung schafft aber neue Probleme.

Werden auf einem Rechner sehr viele Programme vorgehalten, so werden die Listen sehr lang. Damit wächst aber die Zeit, die zum Durchsuchen der Directories benötigt wird. Insbesondere bewirkt die Suche nach nicht existierenden Files einen ausgiebigen Suchlauf durch alle potentiellen Fundorte.

Noch schwerwiegendere Nachteile ergeben sich, wenn zu zwei Programmsystemen Files mit gleichem Namen gehören. Dann arbeitet das Programm, dessen Daten-Directory später in der append-Liste steht, mit der falschen Datei. Daraus können sich unabsehbare Folgen ergeben.

Im übrigen ist der für diese Listen zur Verfügung stehende Platz beschränkt. Es ergibt sich also eine natürliche Beschränkung. Als Empfehlung kann man sagen, daß jede der Listen zu keinem Zeitpunkt mehr als vier Directories enthalten sollte.

Es ergibt sich also die zwingende Notwendigkeit, die Directory-Listen nicht statisch festzulegen, sondern jeweils vor dem Aufruf programmspezifisch anzupassen. Dies darf natürlich nicht dem Benutzer zugemutet werden, sondern muß von entsprechend vorbereiteten Batch-Files vorgenommen werden, die für jedes aufzurufende Programm vorzusehen sind. Alle diese Batch-Files werden in einer gemeinsamen Directory abgelegt, die standardgemäß in der path-Liste steht. Die Batch-Files haben dann alle einen identischen aus drei Teilen bestehenden Aufbau:

1. Im ersten Teil werden die Directory-Listen entsprechend den Bedürfnissen des aufzurufenden Programms gesetzt. Die Standardsetzungen können dabei zerstört werden.
2. Danach folgt der eigentliche Programmaufruf. Davor und dahinter können je nach Programm auch noch weitere Kommandos, insbesondere set-Kommandos, vorgesehen werden.
3. Nach der Ausführung des Programms werden die Directory-Listen wieder auf ihren Standardwert zurückgesetzt. Insbesondere muß auch wieder die Directory der Batch-Files enthalten sein.

Wird diese Organisation konsequent implementiert und diszipliniert angewendet, so kann man zu jedem Zeitpunkt von jedem Standardlaufwerk und jeder Standard-Directory aus jedes gewünschte Programm aufrufen. Es wird vom Betriebssystem ebenso wie eventuell benötigte programmspezifische Dateien automatisch lokalisiert. Der Benutzer muß nicht einmal wissen, wo diese Files im File-System abgelegt sind.

### *Ein abschließendes Beispiel*

Zum Abschluß möchte ich exemplarisch die Organisation der Programme auf der Platte N der Arbeitsgeräte im CIP-Pool des Universitätsrechenzentrums vorstellen. Das Laufwerk N ist die vom Netzwerk-Server allen Benutzern zur Verfügung gestellte Platte. Auf ihr sind zentral sämtliche angebotenen Programmsysteme abgelegt. Entsprechend den oben entwickelten Vorstellungen ist jedes Programmsystem in einer eigenen Directory untergebracht, und für jedes Programmsystem existiert ein spezielles Batch-File. Vier Directories spielen dabei eine besondere Rolle:

dos Diese enthält alle direkt zum Betriebssystem gehörenden Programme.

**msnet** Hier sind alle Kommandos und Parameter-Files, die zum Betrieb des Netzwerks erforderlich sind, abgelegt.

**bin** In ihr sind alle Programme zusammengefaßt, die für sich alleine ausgeführt werden können und keine besonderen programmspezifischen Dateien benötigen.

**batch** Dies ist endlich die Directory, in der die Batch-Files der Programmsysteme zusammengefaßt sind.

Standardgemäß enthält die path-Liste diese vier Directories in der Folge `batch`, `bin`, `dos` und `msnet`. Die append-Liste ist leer. Damit könnte die oben skizzierte Batch-Datei `tex.bat` für den Aufruf von TEX folgendermaßen aussehen:

```
append n:\pctex\texinput;
       n:\pctex\texfmts;
       n:\pctex\textfms
path n:\pctex
tex %1 %2 %3 %4 %5 %6 %7 %8 %9
path n:\batch;n:\bin;n:\dos;n:\msnet
append ;
```

(Ebenso wie oben muß die Eingabe des append-Kommandos in einer Zeile erfolgen!)

Wenn Sie sich jedoch die auf dem Laufwerk N installierte Prozedur tatsächlich ansehen, so werden Sie sehen, daß noch einige zusätzliche Kommandos ausgeführt werden. Solche Zusätze werden Sie bei manchen Batch-Files finden. Sie deuten fast immer auf ein nicht netzgerechtes Verhalten der entsprechenden Programme hin. Jedes glatte Konzept wird eben häufig auch durch nicht sofort sichtbare Kanten gestört.

## Die Statistik-Seite

### Programmquerschnitt Januar bis März 1987

von

*A. Ahrens*

Im ersten Quartal 1987 ergab sich im MVS-Batch-Betrieb auf Jobstep-Basis folgende Verteilung:

| Programm | CPU | % | Anzahl | % |
|---|---|---|---|---|
| FORTRAN-66-Compile | 3:44:52 | 0.28 | 5045 | 2.32 |
| FORTRAN-66-Execute | 46:31:13 | 3.44 | 10911 | 5.02 |
| FORTRAN-77-Compile | 42:01:15 | 3.11 | 37018 | 17.04 |
| FORTRAN-77-Execute | 933:36 | 69.12 | 41861 | 19.27 |
| PL/I-Compile | 18:00:55 | 1.33 | 14445 | 6.65 |
| PL/I-Execute | 127:53 | 9.47 | 23858 | 10.98 |
| LINKEDIT | 6:20:10 | 0.47 | 10656 | 4.91 |
| Pascal-Compile | 0:42:53 | 0.05 | 748 | 0.34 |
| Pascal-Execute | 26:46:08 | 1.98 | 756 | 0.35 |
| SAS | 24:14:24 | 1.79 | 4247 | 1.96 |
| SPSS | 30:19:44 | 2.25 | 21054 | 9.69 |
| Utility-Programme | 4:17:09 | 0.32 | 11769 | 5.42 |
| SERVICE | 21:17:36 | 1.58 | 21516 | 9.91 |
| SORT | 3:58:02 | 0.29 | 4867 | 2.24 |
| Sonstiges | 61:00:26 | 4.52 | 8450 | 3.89 |

Insgesamt gab es 217201 Jobsteps mit einer CPU-Zeit von 1350:44 Stunden.