Kapitel 4 – ISPF Dialog Management Services

Die Servicepalette des Dialog-Managers umfasst mehrere Bausteine, die im Verbund das Erstellen von Dialog-Anwendungen sehr komfortabel gestalten.

Die Services des Dialog Managers

Display Service

wird zur Anzeige von Bildschirmmasken genutzt. Dieser Service wird im Allgemeinen zur Datenerfassung oder Präsentation genutzt.

Selection Service

Auch hier kann eine Bildschirmmaske angezeigt werden. Im Unterschied zum Display Service handelt es sich um so genannte Auswahlmenüs (vgl. Menü 0 aus ISPF/PDF), bei dem durch Eingabe Kürzels nachfolgende Aktivitäten aus dem Panel heraus gesteuert werden. Aktivitäten, die der Selection Service durchführt sind die Aufrufe von

- Clisten oder REXX-Programme
- Lademodulen
- Auswahl-Bildschirmmasken

Table Service

Mit dem Tabellen Service wird ein Werkzeug angeboten, das Tabellenverarbeitung auf einer sehr hohen Komfortstufe ermöglicht. Bei kleineren Anwendungen (bis etwa 20.000 Datensätze) ersetzt dieser Service im Regelfall den Einsatz von Datenbanken.

Die einzelnen Felder der Tabellensätze müssen weder in ihrer Länge, noch in ihrem Datentyp identisch sein. Zudem können im Nachhinein für einzelne oder alle Tabellensätze Felder angefügt werden, die in ihrem Aussehen ebenfalls vollkommen flexibel sein können.

Durch die Definition von Schlüsselfeldern können doppelte Datensätzen ausgeschlossen werden. Die Prüfung auf Vorhandensein übernimmt der Service selbst.

Über Sortierbefehle kann die Tabelle nach beliebigen Feldern sortiert aufbereitet werden.

Die wesentlichen Funktionen des Tabellen Service sind:

- Anlegen und Löschen von Tabellen
- Einfügen und Löschen von Tabellensätzen
- Anzeige
- Sequentielles Lesen
- Suchvorgänge

File Tailoring Service

bietet im Wesentlichen die Möglichkeit, vorbereitete Datengerippe zusammenzuführen und in eine gemeinsame Datei auszugeben. Dabei können sowohl Variable substituiert, als auch Tabellen des Dialog Managers implementiert werden.

Eine klassische Anwendung des File Tailoring wäre beispielsweise die Druckausgabe einer DMS-Tabelle (gegebenenfalls auch nur selektierte Sätze) durch JCL, die durch das File Tailoring vervollständigt wird.

Variablen Service

Der Variablen Service bietet die Möglichkeit, die Inhalte beliebiger Variabler über Programmende oder über LOGOFF hinaus festzuhalten und zu einem beliebigen späteren Zeitpunkt wieder zur Verfügung zu haben. Für das Ablegen von Variablen nutzt der Dialog Manager drei verschiedene Variablen-Pools:

- Function Pool
- Shared Pool
- Profile Pool

Darüber hinaus erlaubt er eine sehr komfortable Kommunikation zwischen Programmen (Variablenaustausch), auch wenn diese nicht innerhalb der gleichen Task im TSO-Adressraum laufen.

Dateilandschaft

Die Dienste des Dialog Managers werden als Member von PO-Dateien erstellt.

Die Aufteilung nach Zugehörigkeit zu einem Service und die Nutzung nachfolgender Dateinamen ist eine Empfehlung:

userid.????????.PANELS

Bildschirmmasken für die Services DISPLAY und SELECT

userid.????????.TABLES.

Tabellenein- und -ausgaben für alle Tabellen-Services (wobei Ausgaben prinzipiell wahlfrei zugeordnet werden können).

userid.????????.MSGS

Nachrichten, die in Bildschirmmasken benutzt werden.

userid.????????.SKELS

Skelette für den FILE-TAILORING-SERVICE.

userid.????????.PROFILE

Bildet mit seinen Membern den applikationsabhängigen Variablenpool ab.

Begriffe

Die einzelnen Elemente eines Dialogs unter Dialog Manager werden unter folgenden Begriffen zusammengefasst, wobei nicht immer alle Elemente vorhanden sein müssen:

Funktionen

sind Programme, die im Dialog aufgerufen werden. Sehr gängig, weil sehr komfortabel zu schreiben, ist hier die Kodierung in

- REXX oder
- CLIST

aber auch die klassischen Programmiersprachen

- PL/1
- COBOL
- FORTRAN
- PASCAL
- APL2 und
- ASSEMBLER

sind möglich. Ein Dialog kann prinzipiell beliebig viele Funktionen enthalten, die in einer der oben angeführten Sprachen geschrieben sind.

Paneldefinitionen

sind die Formatbeschreibungen in Form kodierter Anweisungen über das Aussehen einer Bildschirmmaske. Ebenso liegen dort Anweisungen, die vor Anzeige einer Maske durchlaufen werden und die innerhalb der Maske durchzuführenden Prüfungen.

Message Definitionen

enthalten das Format und die Meldungen, die in Bildschirmmasken erscheinen sollen.

Tabellen

können im Dialog angelegt und manipuliert werden und werden in Dateien abgelegt.

Datenskelette

sind Fragmente beliebigen Inhalts (Daten, JCL-Sätze etc.), die zu einer Ausgabedatei zusammengefügt werden.

Variablen

werden in der Regel als Bindeglied zwischen den einzelnen Elementen des Dialogs genutzt und steuern über ihre Inhalte im Allgemeinen den Verlauf des Dialoges.

Die Dienste des Dialog-Managers sind nur unter ISPF verfügbar. Startet eine ISPF-Anwendung aus TSO-native, muss ISPF nachgestartet werden.

Die Zuordnung kann über den TSO-Befehl ALLOCATE, oder, soweit es sich um erweiterte Zuordnungen über den Standard hinaus handelt, über die LIBDEF-Funktion des Dialog Managers erfolgen.

DD-Namen und ihre Bedeutung

DD-Name	Bedeutung
SYSPROC	Programmbibliothek für CLISTen
SYSEXEC	REXX-Programmbibliothek
ISPLLIB	Lademodulbibliothek
ISPPLIB	Bibliothek für Panels (Bildschirmmasken).
ISPMLIB	Bibliothek für Nachrichten (Messages), die in Bildschirmmasken verarbeitet werden.
ISPPROF	Bibliothek mit den applikationsabhängigen PROFILE-Pools für Variable. Standardgemäß werden von ISPF/PDF folgende Member benutzt: • ISRPROF ISPF-Standard-Profile-Pool • ISREDIT Last-Qualifier-Dateiprofil (maximal 25) • ISREDRT Recovery-Tabelle für EDIT (maximal 5) • ISRCMDS ISPF-Kommandotabelle
ISPTLIB/ISPTABL	Eingabe/Ausgabe für Tabellenverarbeitung
ISPSLIB/ISPFILE	Eingabe/Ausgabe für Skelett-Dateien
ISPILIB	Graphik-Bibliothek für GIF-Dateien (Image-Lib)

Dialoge

Je nach Anforderung können unterschiedliche Arten von Dialogen aufgebaut werden. Grundsätzlich unterscheiden wir zwischen

- **einfachen Dialogen**
  aus einem Programm heraus werden ein oder mehrere Bildschirmmasken aufgerufen

- **komplexen Dialogen**
  hierarchisch geordnet werden mehrere Programme und Bildschirmmasken genutzt, die in der Regel über den SELECT-Service miteinander verbunden sind.

Um mit den Routinen des DMS unter TSO arbeiten zu können, gibt es zwei Möglichkeiten, welche davon abhängen, ob ISPF bereits geladen ist, oder man sich im TSO-READY-Modus (TSO-native) befindet.

- **ISPF aktiv**
  Alle Serviceroutinen sind verfügbar. Meist werden die Zuweisungen aller benötigten Bibliotheken über den LIBDEF-Service veranlasst, der im hierarchisch obersten Programm liegen sollte.

- **TSO-READY**
  Dialog Manager muss über das ISPSTART-Kommando geladen werden (siehe unter ISPSTART in der Syntaxbeschreibung

Ob ISPF aktiv ist oder nicht, kann unter REXX erfragt werden:

```
IF SYSVAR(SYSISPF)="ACTIVE" THEN DO
     [ISPF ist geladen]
END
ELSE DO
     [ISPF ist nicht aktiv]
END
```

oder

```
"SUBCOM ISPEXEC"
IF RC = 0 THEN DO
     [ISPF ist geladen]
END
ELSE DO
     [ISPF ist nicht aktiv]
END
```

Returncodes und Fehlerbehandlung

Endet ein ISPF-Service, reicht er einen Returncode an das aufrufende Programm zurück. Die möglichen Returncodes sind 0, 4, 8, 10, 12, 16 und 20. Der jeweilige Hintergrund ist abhängig vom Service und kann bei der dazugehörenden Syntaxbeschreibung nachgelesen werden.

In einem REXX-Programm liegt der Returncode in der Variablen RC, bei CLIST in &LASTCC vor, sobald die Steuerung über den weiteren Ablauf von DMS wieder an das Programm zurückgegeben wird.

Tritt im Service ein Returncode < 12 auf, wird mit dem nächsten Befehl im Programm die Verarbeitung fortgesetzt.

Bei einem Returncode größer oder gleich 12 ist die weitere Verarbeitung abhängig von der Einstellung des Control-Statements im Dialog Manager:

"CONTROL ERRORS CANCEL"

Es erfolgt ein uneingeschränkter Abbruch des gesamten Dialoges bis hin zu ISPF oder TSO-READY.

"CONTROL ERRORS RETURN"

Es wird in jedem Fall der nächste Befehl nach dem RC-verursachenden Service angesprungen. Im Allgemeinen ist dies die sicher sinnvollere Lösung, da in jedem Fall bei auftretenden Fehlern ein ungeregeltes Dialogende vermieden werden kann.

Edit MODEL und MODEL CLASS

Um bei der Entwicklung eines Dialoges möglichst viel Komfort zu haben, empfiehlt es sich, mit dem EDIT-Befehl MODEL zu arbeiten. Über die Befehlskombination "MODEL-A/B" werden die Befehle der jeweils aktiven Model-Klasse angezeigt. Nach Auswahl eines Befehles wird die Befehlssyntax mit diversen Erläuterungen (Notes) und den jeweiligen Returncodes an der vorgesehenen Stelle eingefügt. Wird im Dateiprofil "NOTES OFF" gesetzt, werden die Erläuterungen nicht angezeigt. Soll eine andere Model-Klasse angesteuert werden, als die zur Zeit aktuelle, kann diese über den Befehl "MODEL CLASS" eingestellt werden.

Dialog Test

Um einen Dialog zu testen, sollte grundsätzlich die PDF-Option 7 genutzt werden, da andernfalls beispielsweise Veränderungen an Panels nicht registriert werden (ein Panel wird beim ersten Aufruf in den Adressraum gelesen und nicht erneut von der Platte geladen, solange es im Speicher vorhanden ist).

Für das Austesten eines Dialoges bieten sich an:

- 7.1 Select Services (Programme und Panels)
- 7.2 Display Services (Panels und Messages)
- 7.3 Manipulation des Profile-Variablenpools
- 7.4 Arbeiten an DMS-Tabelle

Variablen

Variablen im Dialog Manager können in unterschiedlichen Bereichen definiert und benutzt werden. Anwendung finden sie innerhalb von

Funktionen (zum Beispiel CLISTEN oder REXX)

- Bildschirmmasken
- Skeletons
- Tabellenverarbeitung

DMS unterscheidet dabei unter drei verschiedenen Bereichen, innerhalb derer Variable abgelegt sein können. Diese drei Bereiche werden zu unterschiedlichen Zeitpunkten gebildet und haben unterschiedliche Lebensdauer. Die Namen dieser Bereiche lauten

- Funktion-Pool
- Shared-Pool und
- Profile-Pool
- Function-Pool

Jede Funktion im Dialog (Programm, REXX oder CLIST) hat ihren eigenen Function-Pool. Er wird bei Start der Funktion initialisiert. Die Variablen sind nur innerhalb der Funktion verfügbar, die sie angelegt hat. Unter REXX oder CLIST haben auch externe UDF's keinen Zugriff auf diese Variablen.

Wird innerhalb einer Funktion eine neue aufgerufen, wird der aktuelle Function-Pool abgedeckt und für die neue Funktion ein neuer Function-Pool zugewiesen. Wird eine Funktion beendet, wird der dazugehörende Function-Pool gelöscht.

Shared-Pool

Der Shared-Pool wird gebildet, wenn ISPF gestartet wird und besteht bis zum Ende von ISPF. Funktionen können durch die Befehle VGET/VPUT auf den Shared-Pool zugreifen. Für den Zugriff auf den Shared-Pool ist zu beachten:

Selection-Panels (Auswahl-Bildschirmmasken) greifen unmittelbar auf den Shared-Pool zu, da sie keinen Function-Pool kennen.

Soll eine Variable aus dem Function-Pool auch im Shared-Pool abgebildet werden, ist sie mittels VPUT-Anweisung dorthin zu schreiben.

Muss eine Variable aus dem Shared-Pool im aktuellen Function-Pool verfügbar sein, ist sie über VGET aus dem Shared-Pool zu lesen.

Profile-Pool

Variable, die über die Dauer einer ISPF-Session oder auch über LOGOFF hinaus festgehalten werden sollen müssen im Profile-Pool abgebildet werden. Dieser Bereich ist in Form einer PO-Datei realisiert, innerhalb derer applikationsabhängige Variablen-Pools in Form von Membern eingetragen werden können. Die betreffende PO-Datei muss über das Symbol ISPPROF bekannt gegeben worden sein.

Die im Profile-Pool hinterlegten Variablen werden bei Beginn des Dialoges von Datei gelesen und im virtuellen Speicher abgebildet.

Anders als der Shared- oder Function-Pool wird der Profile-Pool bei Dialogende vom virtuellen Speicher auf Platte zurück geschrieben. Über diesen Weg ist es beispielsweise auch dem ISPF-Editor möglich, sich zu merken, wie der aktuelle Wert für das Blättern am Bildschirm (SCROLL-AMOUNT) gesetzt war.

Die Eintragungen in den Profile-Pool oder das Lesen daraus werden über die DMS-Befehle "VPUT" und "VGET" durchgeführt.

Vor Änderung des Profile-Pools im Editor wird dringend abgeraten, da die Werte teils im Binärformat hinterlegt sind. Explizite Veränderungen im Profile-Pool sind über Menü 7 des ISPF durchzuführen.

VPUT-Service

Soll eine Variable in allen Funktionen des Dialogs verfügbar sein, muss sie durch VPUT in den Shared- oder Profile-Pool übertragen werden. Existiert dort bereits eine gleichnamige Variable, wird ihr Inhalt überschrieben. Um den Service gezielt zum Schreiben in einen bestimmten Variablenpool zu veranlassen, können entsprechende Schlüsselworte mitgegeben werden.

ASIS

Die Variable wird in demjenigen Pool (Shared oder Profile) hinterlegt, in dem bereits eine Variable gleichen Namens existiert. Wird sie in keinem der beiden Pools gefunden, wird sie in den Shared-Pool eingetragen.

SHARED

Die Variable wird in den Shared-Pool eingetragen.

PROFILE

Die Eintragung erfolgt im Profile-Pool. Existiert eine gleichnamige Variable im Shared-Pool, wird sie dort ausgetragen.

VGET-Service

Durch VGET kann eine Variable, die nur im Shared-Pool oder Profile-Pool existiert gelesen und in den Function-Pool kopiert werden.

Ist eine gleichnamige Variable im Function-Pool bereits vorhanden, wird ihr Inhalt überschrieben.

Gewichtung der POOLS

Generell wird von den ISPF-Services die Suchreihenfolge

- Function-Pool
- Shared-Pool
- Profile-Pool

eingehalten. Existiert eine Variable in mehr als einem Variablenpool (prinzipiell wäre dies auch mit unterschiedlichen Inhalten möglich), wird sie mit ihrem Inhalt aus dem Pool genommen, in dem sie zuerst gefunden wurde.

Die Ausnahme bildet hier der SELECT-Service, der keinen Function-Pool kennt. Von ihm werden zwangsläufig nur Shared-Pool und Profile-Pool durchsucht

Panels

Unter einem Panel versteht man das Aussehen eines Bildschirmes (Layout), wie es sich für den Terminalbenutzer darstellt. Die Kodierung eines Panels enthält die Angaben zur Text- und Variablenanordnug auf einer Bildschirmseite, sowie Definitionen über Initialisierungsarbeiten vor Anzeige und Plausibilitätsprüfungen vor Verlassen der Maske.

Der Name der Maske ist gleichbedeutend mit dem Membernamen, unter dem sie in einer Datei unter ISPPLIB abgespeichert wird.

Sections

Das Panel gliedert sich in verschiedene Bereiche (Sections), die bei der Verarbeitung eine unterschiedliche Rolle spielen. Um welchen Abschnitt im Panel es sich handelt, wird mit dem so genannten Sectionnamen bestimmt, der direkt hinter einer schließenden Klammer ab Spalte 1 kodiert werden muss.

Nicht alle Abschnitte sind zwingend notwendig. Die Reihenfolge ist zwingend.

)CCSID

Ist sehr selten anzutreffen. Hier könnten u.U. Zeichenumsetztabellen hinterlegt werden, sofern diese notwendig sind (meist für chinesisch oder japanisch). Wenn vorhanden, wird das Panel im CUA-Modus angezeigt.

)PANEL

definiert beispielsweise die so genannten KeyLists (Panel-abhängige Funktionstastenbelegungen) und legt fest, dass das Panel im CUA-Modus angezeigt werden soll.

)ATTR

ist wahlfrei und wird nur benötigt, wenn von den Standard-Attributen abgewichen wird, oder diese ergänzt werden sollen.

)ABC

ActionBarChoice, die Aktionsleiste, sofern das Panel eine haben soll.

)ABCINIT

Initialisierungsteil für die ActionBar. Wird durchlaufen, bevor die Maske aufgebaut wird.

)ABCPROC

ActionBar-Verarbeitungsteil.

)BODY

ist zwingend und beschreibt das Aussehen der Maske.

)MODEL

wird nur für die Anzeige von Tabellen-Panels benötigt und beschreibt das Aussehen einer Tabellenzeile.

)AREA bereichsname

beschreibt einen Teil der Bildschirmmaske, in dem mit den Scrollfunktionen UP und DOWN geblättert werden kann. Hier könnten Masken erzeugt werden, die die physische Länge des Bildschirmes überschreiten.

)INIT

ist wahlfrei und enthält Instruktionen, die vor der Anzeige der Maske ausgeführt werden.

)REINIT

wird ausgeführt, wenn aufgrund eines erkannten Fehlers im PROC-Teil das Panel erneut angezeigt wird.

)PROC

wird ausgeführt, sobald der Bediener ENTER, oder die Funktionstaste END gedrückt hat.

)FIELD

Scrollable Field Section. Definiert ein Feld als scrollable.

)HELP

hier werden feldbezogene HELP-Panels definiert.

)LIST

List Section für CUA-like Auswahlverfahren. Darauf wird an dieser Stelle nicht weiter eingegangen.

)PNTS

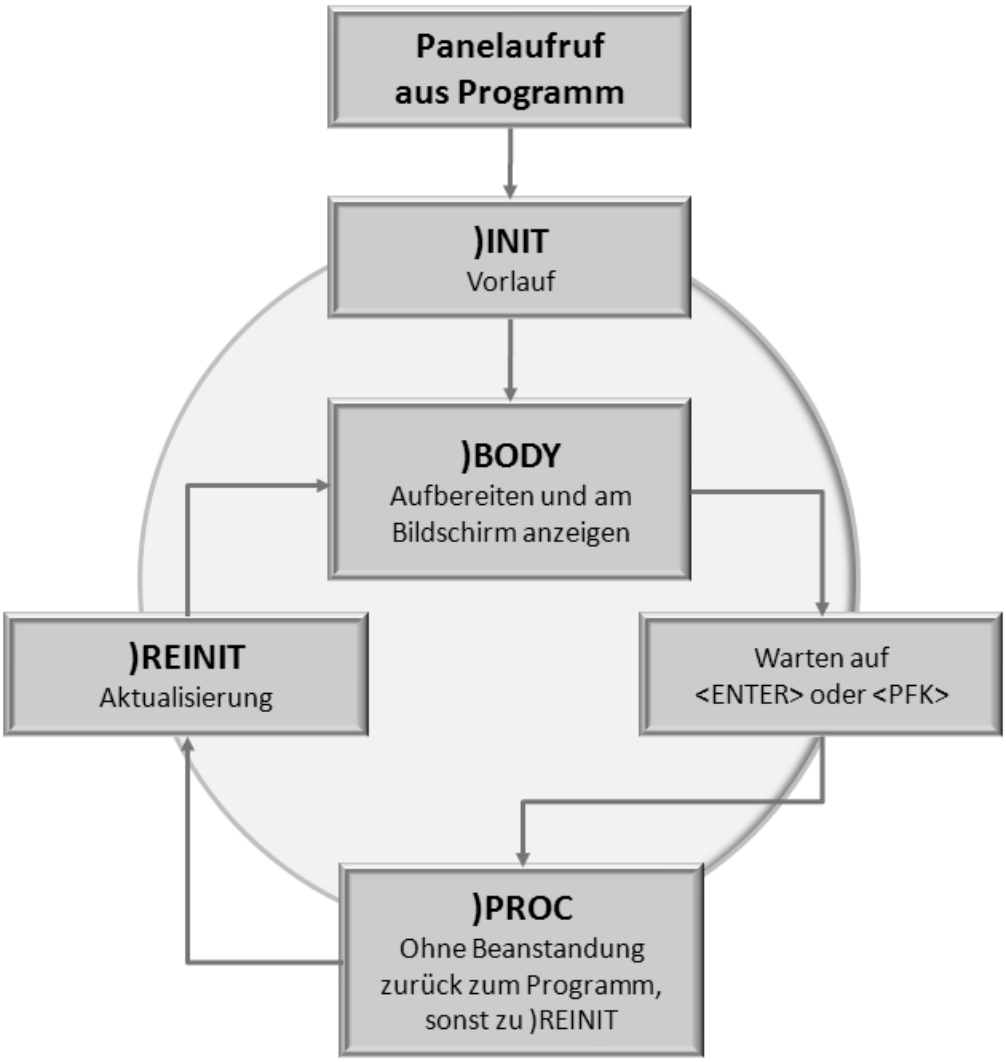
Point-and-shoot Section. Enthält für jedes Point-and-shoot-Feld einen Eintrag.

)END

definiert das Ende des Panels und kennt keine möglichen Operanden.

Die Verarbeitung der einzelnen Sektionen im Panel geschieht nicht in einer linearen Reihenfolge von oben nach unten und wird in folgender Grafik verdeutlicht (Standardpanel):

Ablaufsteuerung Panel



Vorbereitende Arbeiten

Zunächst werden die Attributmerkmale interpretiert, sind keine Attribute definiert, erfolgt die Umsetzung dem Standard entsprechend.

Variablen werden entsprechend den Anweisungen des INIT-Teiles gefüllt. Darüber hinaus sind alle Variablen des Funktion-, Shared- und Profile-Pools verfügbar.

Das Panel ist zu diesem Zeitpunkt noch nicht am Bildschirm sichtbar.

Panel-Anzeige

Das Panel wird am Terminal abgebildet. Nach Aufruf des Panels tritt die Funktion in einen Wartezustand der entweder durch < ENTER> oder eine PF-Taste beendet wird.

Ausnahme:

Wird in der Kommandozeile des Panels (soweit überhaupt kodiert) ein gültiges ISPF-Kommando eingegeben, wird dies vom Dialog Manager ausgeführt und die steuernde Funktion im Wartezustand belassen.

Panel-Auswertung

Sinnvollerweise (wenn auch nicht zwingend) wird im Panel eine)PROC-Section kodiert. Dieser Bereich wird für die Auswertung der Dateninhalte angesprungen. Hier erfolgt im Regelfall die Plausibilitätsprüfung für alle Eingabefelder des Panels.

Grundsätzlich können diese Prüfungen auch in der aufrufenden Funktion kodiert werden. Aus Performancegründen ist aber eine Überprüfung im Panel soweit als möglich empfehlenswert.

Erneute Panelanzeige

Werden in der Plausibilitätsprüfung Fehler erkannt, wird in den)REINIT-Teil des Panels verzweigt. Hier können fehlerabhängige Modifikationen an der Bildschirmmaske vorgenommen werden. Im Anschluss daran, oder wenn kein)REINIT-Teil kodiert wurde, wird das Panel erneut am Terminal abgebildet (die Steuerung geht wieder zum)BODY-Teil).

Ende der Panelanzeige

Bei Erreichen der)END-Marke wird die Steuerung an die aufrufende Funktion zurückgegeben. Diese Rückgabe der Steuerung über den weiteren Ablauf erfolgt bei END, RETURN oder wenn im)PROC-Teil keine Fehler erkannt wurden.

Panel-Attribute

Im Attribute-Bereich können zunächst die unterschiedlichen Attribute der im Panel angezeigten Felder (Text, Output oder Input) abweichend zum Standard geregelt werden. Fehlt der)ATTR-Teil, werden die Standardzeichen benutzt, die folgende Bedeutung haben:

%	Text intensiv
+	Text normal
_	Inputfeld intensiv

Diese Attribute müssen im)BODY-Teil dem Feld vorangestellt werden, für das sie gelten sollen. Diese Definition für ein Feld gilt so lange, bis das nächste Attributzeichen erkannt wird.

Soll eines oder mehrere der Standardattribute als Text genutzt werden, sind die Default-Werte neu zu setzen. Dies geschieht im Regelfall in der Kopfzeile des)ATTR-Teiles, oder falls dieser fehlt in der Kopfzeile des)BODY-Teils in folgender Form:

```
)ATTR DEFAULT(x1 x2 x3) oder
)BODY DEFAULT(x1 x2 x3)
```

Falls DEFAULT kodiert wird, müssen immer alle drei Standardzeichen angegeben werden, wobei die Reihenfolge strikt einzuhalten ist und die Bedeutung immer der Reihe nach

1. Text intensiv
2. Text normal und
3. Eingabefeld intensiv

erfolgt.

Werden im Panel weitere Feldattribute abweichend zu den standardgemäßen benutzt, müssen diese im)ATTR-Teil definiert werden. Das Attributzeichen kann zweistellig-HEX oder als Characterzeichen dargestellt werden. Die Zeichen
&, blank '40'x, NUL '00'x, ShiftOut '0E'x, ShiftIn'0F'x
dürfen nicht benutzt werden. Die Eigenschaften der Felder werden durch Schlüsselworte bestimmt. Die Wertstellungen der einzelnen Schlüsselworte kann durch Literal oder Variable festgesetzt werden (Ausnahme: TYPE(TEXT) muss kodiert werden wie geschrieben).

Attribut-Defaults

Die Standardwerte bei einem Panelaufruf sind bei

```
TYPE(INPUT) CAPS(ON) JUST(LEFT) PAD(USER)
TYPE(OUTPUT) CAPS(ON) JUST(LEFT) PAD(' ')
```

Attribut-Schlüsselworte

Die gängigen Attribut-Schlüsselworte sind:

Keyword	Bedeutung
AREA	Bereichsdefinitionen innerhalb der Maske
TYPE	bestimmt die Art des Feldes
INTENS	Steuert die Helligkeit
CAPS	Groß-/Kleinschreibung
JUST	Ausrichtung der Daten
PAD	Füllzeichen für Ein-/Ausgabefelder
PADC	Abhängiges Füllzeichen (Conditional)
SKIP	Textfelder werden übersprungen
COLOR	Weist dem Feld eine Farbe zu

Aus nachfolgendem Beispiel kann der Aufbau einer Bildschirmmaske mit diversen Feldattributen ersehen werden:

```
)ATTR DEFAULT(@+_)
     *    TYPE(OUTPUT) INTENS(HIGH) CAPS(OFF)
     ^    TYPE(INPUT) INTENS(LOW) COLOR(BLUE)
     $    TYPE(TEXT) INTENS(LOW)
     #    TYPE(TEXT) INTENS(HIGH) COLOR(YELLOW)
)BODY
#======== Kunden-Abfrage ============================
@
$Datum:*ZDATE                              $Zeit: *ZTIME
@
@
$    Eingabe:   Kundennummer
$    Ausgabe:   Adresse und Rabattsatz
@
@
#    Kundennummer          ==>  ^KDNR  #
#    Name                  ==>  *NAME
#    Strasse               ==>  *STR
#    PLZ/ORT               ==>  *PLZ   *ORT
#    Rabatt                ==>  *RAB@%
#
#
$    ENDE mit   #END
$    Speichern  #ENTER
)END
```

TYPE

Das Schlüsselwort TYPE beschreibt die unterschiedlichen Formen der Darstellung der einzelnen Felder in der Maske.

TYPE(TEXT)

zeigt ein Feld exakt so, wie es im)BODY-Teil geschrieben ist. Werden Variablen (führendes &) im Textteil genutzt, erscheint am Bildschirm an der entsprechenden Stelle der Variableninhalt. Werden Variablen benutzt, ist die Ausgabezeilenlänge flexibel.

TYPE(INPUT)

Eingabefeld. Diese Stelle in der Maske ist beschreibbar. Der hier eingegebene Wert wird bei E in eine Variable übertragen und im Variablenpool abgestellt. Die Länge der Felder ist abhängig vom nächsten nachfolgenden Attributzeichen.

TYPE(OUTPUT)

Ausgabefeld. Der Inhalt wird einer Variablen entnommen und angezeigt. Diese Felder sind nicht beschreibbar. Im Unterschied zu Variablen in TEXT-Feldern wird hier unter Umständen nicht der gesamte Inhalt einer Variablen, oder aber mehr (abhängig von der Längendefinition des Output-Feldes) angezeigt.

Im)BODY-Teil werden die Feldnamen mit Attributzeichen, aber ohne Ampersand geschrieben (Ausnahme ist eine Variable in einem Textfeld).

Gestaltung der Panels

Im)BODY-Teil wird die Bildschirmmaske so abgebildet, wie sie für den Benutzer am Bildschirm erscheinen wird. Dieser Teil darf deshalb in Punkto Spalten- und Zeilenzahl die Werte eines Terminals nicht überschreiten.

In Übereinstimmung zu den PDF-Panels sollte der Aufbau innerhalb gewisser Normen stattfinden:

- Durch den Befehl **PANELID ON** erscheint in Zeile 1, Stelle 1 bis 8 der Panelname
- In den maximal letzten 24 Byte der ersten Zeile erscheint im Bedarfsfall eine kurze Nachricht (Short Message).
- Der verbleibende Rest der ersten Zeile sollte für eine Überschrift genutzt werden.
- In der zweiten Zeile steht links das COMMAND-Feld und rechts das SCROLL-Feld (für Blätterfunktionen bei Tabellenanzeige).
- Die dritte Zeile sollte generell freigelassen werden. Hier erscheint eine Nachricht (Long Message), wenn auf eine kurze Nachricht hin die HELP-Funktion aufgerufen wird.

Der Rest des Bildschirmes steht zur freien Verfügung.

Pfeilmarkierungen (CUA: Leader Dots)

In den Panels des PDF werden Eingabefeldern in aller Regel Pfeile vorangestellt. Diese Vorgehensweise wird auch bei eigenen Dialogen aus zweierlei Gründen empfohlen:

- Eingabefelder werden visuell sofort erkannt, da sie das gewohnte Erscheinungsbild aus PDF aufweisen.
- Der Pfeil als Feldmarkierung erlaubt auch innerhalb des eigenen Dialoges Menüsprünge über das letzte vorgelagerte Primary-Menü (vgl.: Direktsprung von EDIT nach BROWSE durch =1).

Ausnahmen

Manche Bildschirmmasken zwingen aufgrund bestimmter Vorgaben zu einer Abweichung vom Standard-Layout. In diesem Fall sieht die Kodierung etwas anders aus, wie die folgende Seite zeigt.

```
)BODY DEFAULT(#+_)  CMD(variable)
    SMSG(variable) LMSG(variable) ASIS
    EXPAND(//)     WIDTH(integer)
    WINDOW(spalten,zeilen)
+/-/< Überschrift >/-/
```

Die Angaben im)BODY-Teil haben folgende Bedeutung:

- DEFAULT gleichbedeutend wie in)ATTR
- CMD, SMSG und LMSG müssen mit Variablennamen versorgt werden, wenn die Standorte der Felder abweichend vom Standard an beliebiger Stelle der Bildschirmmaske liegen sollen.
- ASIS veranlasst, dass die Standorte des COMMAND-Feldes und der Nachrichtenfelder nicht durch die PDF-Option 0.4 beeinflusst werden.
- WIDTH regelt die Spaltenbreite grafikfähiger Terminals, die die Breite von 80 überschreiten können. Die Angabe ist zwingend, wenn das Terminal über die Spalte 80 hinaus genutzt werden soll.
- EXPAND erlaubt das Festlegen so genannter Begrenzer, zwischen denen ein Füllzeichen gesetzt werden kann. Mit diesem Füllzeichen lassen sich Texte links- oder rechtsseitig bis Zeilenende auffüllen.
- WINDOW definiert die Fenstergröße, in dem das Panel als PopUp dargestellt wird. Die Anzeige des Fensterpanels variiert je nach Aufruf.

Panelaufruf als Window mit REXX

Die WINDOW-Angabe wird ignoriert, wenn der DISPLAY-Service ohne vorhergehenden ADDPOP aufgerufen wird. Ohne WINDOW() im Panel, werden, sobald ADDPOP einmal genutzt wurde, die folgenden Panels im zuletzt definierten Fenster gezeigt. Wird WINDOW() nicht kodiert und der ADDPOP-Befehl genutzt, wird das Panel ganzseitig mit Rahmen abgebildet. Die Parameter POPLOC(), ROW() und COLUMN() werden ignoriert.

Panelaufruf als Window aus DMS

Ist einmal ein Fenster definiert, erscheinen alle nachfolgenden Panels als Fensterpanel in der Größe der zuletzt aufgerufenen Bildschirmmaske, unabhängig davon, ob das Panel eine WINDOW()-Angabe im)BODY-Teil enthält, oder nicht.

Mit der Systemvariablen ZWINTTL kann ein Text angegeben werden, der in der oberen Leiste des Fensterrahmens als Titel zentriert angezeigt wird.

Scrollable Panels

Ein häufiges Problem ist die physische Größe des Bildschirmes. Der)BODY-Teil darf grundsätzlich nur so viele Zeilen lang sein, als auf dem Terminal Platz haben. Oftmals können aber nicht alle benötigten Felder auf einer Seite abgebildet werden. Generell könnte dieses Problem auf der Funktionsseite über die Bearbeitung mehrerer Panels gelöst werden. Diese Verfahrensweise macht eine Dialoganwendung aber kompliziert und änderungsunfreundlich (auch wenn dies in der Praxis angetroffen wird).

Abhilfe schafft ein Bereich innerhalb des Panels, in dem über UP und DOWN geblättert werden kann. Der Bereich kann von seiner Größe her im Panel fest definiert werden, oder sich situationsabhängig selbstständig bis zum unteren Panelrand ausdehnen. Innerhalb des Bereiches können beliebige Informationen oder Felder liegen.

Prinzipiell können mehrere derartige Bereiche in einem Panel definiert werden. Mit jedem der Bereiche kann im Grunde ein Panel im Panel dargestellt werden, das angesteuert werden kann, ohne die eigentliche Bildschirmmaske verlassen zu müssen.

Ist nur ein solcher Bereich vorhanden, kann innerhalb dessen mit UP oder DOWN geblättert werden, unabhängig davon, wo der Cursor aktuell steht. Bei mehr als einem Bereich erfolgt das Blättern in dem Bereich, in dem der Cursor positioniert ist.

)AREA bereichsname

Ist eine Section, die nur benötigt wird, wenn mit einer SCROLL-AREA gearbeitet werden soll. bereichsname ist der Name des Bereiches, der in der)AREA-Section beschrieben werden soll.

Für jede SCROLL-AREA im)BODY-Teil des Panels muss eine)AREA-Section kodiert werden, die den gleichen Namen trägt, wie er im)BODY-Teil angegeben wurde.

Im nachfolgenden Beispiel wird ein Panel gezeigt, das Hintergrundinformationen (weniger wichtige) in einer SCROLL-AREA führt.

```
)ATTR DEFAULT(@+_)
     § TYPE(INPUT) JUST(RIGHT) PAD(0)
     ^ TYPE(INPUT) JUST(RIGHT) PADC('_') CAPS(OFF)
     # AREA(SCRL) EXTEND(OFF)
)BODY EXPAND($$)
@--------< Kunden-Erfassung >$-$
@COMMAND ==>_ZCMD
@
@
+  Kundennummer ==>§KDNR +
+  Name         ==>^NAME          +
+  Vorname      ==>^VNAME         +
+  Strasse      ==>^STR           +
```

```
+  PLZ/Ort      ==>^PLZ  ^Ort                 +
+
@  Weitere Angaben:
+
#AREA1                                       #
#                                            #
#                                            #
#                                            #
#                                            #
#                                            #
+
@  Ab: PF8   Auf: PF7
+
)AREA AREA1
+Rabatt        ==>§RAB+%
+Telefonnummer ==>^TELNR      +
+Faxnummer     ==>^FAXNR      +
+Anrede        ==>^ANRED+  (Herr, Frau, oder Firma)
+Sprache       ==>^S+      (D - F - I - E)
+Kunde seit    ==>^SEITDAT   +
+Hausbank      ==>^            +
+Kontonummer   ==>^            +
+Vertreter     ==>^VERTRET    +
+Interne Nr.   ==>^INTERN +
+Interner Code ==>^CODE +(numerisch)
+Werbung       ==>^WERB+ (Ja oder Nein)
+Bemerkung 1   ==>^                   +
+Bemerkung 2   ==>^                   +
)PROC
   VER(&NAME,NB,MSG=FAKT001)
   VER(&VNAME,NB,MSG=FAKT002)
   VER(&RAB,NB,NUM,MSG=FAKT012)
   VER(&ANRED,LIST,'Herr','Frau','Firma',MSG=FAKT021)
   VER(&S,LIST,'D','E','F','I',MSG=FAKT022)
   VER(&CODE,NB,NUM,MSG=FAKT023)
   VER(&WERB,LIST,'JA','NEIN',MSG=FAKT024)
)END
```

Feldhilfen -)HELP

Hier wird ein HELP-Panel definiert, das feldbezogen bei Drücken der PF1-Taste erscheint. Je Feld im BODY-Teil kann ein HELP-Panel definiert werden. Im Beispiel hat das INPUT-Feld für Rabatt ein HELP-Panel:

```
:
# PLZ/ORT  ==> *PLZ   *ORT
# Rabatt   ==> *RAB@%
)HELP
   FIELD(RAB) PANEL(hpname)
)END
```

Panel-Aktivitäten -)INIT,)REINIT,)PROC

In allen drei Bereichen können bestimmte Informationen aus einem Panel verarbeitet werden. Da jeweils die gleichen Sprachelemente in diesen Sektionen erlaubt sind wird bei der Beschreibung der Möglichkeiten im Folgenden nicht weiter unterschieden.

Grundsätzlich finden Befehle, Funktionen und Steuervariable Anwendung, wobei nicht in jedem Bereich die Verwendung aller Elemente sinnvoll ist.

Befehle

werden für die Steuerung des Ablaufes genutzt. Folgende sechs Befehle können kodiert werden:

Befehl	Bedeutung
VGET	Lesen Variable aus Variablenpool
VPUT	Schreiben Variable in Variablenpool
IF / ELSE	Entscheidungsfindung
VER	Feldprüfung (Verify)
REFRESH	Feld-Aktualisierung
&var = wert	Variablenzuweisung
*REXX	REXX-Logik im Panel

Funktionen

sind kleine Befehlsprozeduren oder Programme, die den Bedürfnissen des Benutzers entsprechend eine Wertrückgabe bewirken. Funktionen sind beispielsweise:

Funktion	Bedeutung
TRANS	Zeichenfolgen übersetzen
TRUNC	Zeichenfolgen abschneiden
PFK	Definition der Funktionstasten

Steuervariablen

sind Variablen, die (wie ihr Name sagt) steuernde Funktionen in einem Panel ausführen. Beispielsweise kann die Cursor-Positionierung in einer Bildschirmmaske individuell geregelt werden. Einige der Steuervariablen sind:

Steuervariable	Bedeutung
.ALARM	Ausgabe Warnton
.CURSOR	Cursorpositionierung auf Feld
.CSRPOS	Cursorpositionierung innerhalb eines Feldes
.MSG	Verweis auf eine definierte Nachricht
.TRAIL	Restzeichenfolge nach TRUNCate

Einige der gängigsten Elemente soll das nachfolgende Beispiel demonstrieren:

```
)ATTR DEFAULT(@+_)
   * TYPE(OUTPUT) INTENS(HIGH) CAPS(OFF)
   ^ TYPE(INPUT) INTENS(LOW) PADC('.')
)BODY EXPAND(//)
@/*/ Kunden-Erfassung /*/
@COMMAND ==>_ZCMD                                          +
@
+Kundennummer ==>*KDNR  +
+Name         ==>^NAME                    +
+Vorname      ==>^VNAM                    +
+Strasse      ==>^STR                      +
+PLZ/ORT      ==>^PLZ  ^ORT                          +
+Rabatt       ==>^RAB@%
)INIT
   &NAME=&Z
   &VNAM=&Z
   &STR=&Z
   &PLZ=&Z
   &ORT=&Z
   &RAB=0
   .CURSOR=NAME
)REINIT
   .ATTR(.CURSOR)='INTENS(HIGH) COLOR(RED)'
)PROC
   VER(&NAME,NB,MSG=MSG001)
   VER(&VNAM,NB,MSG=MSG001)
   VER(&VNAM,LEN,'<=',2,MSG=MSG001)
   VER(&STR,NB,MSG=MSG001)
   VER(&PLZ,NB,NUM,MSG=MSG002)
   VER(&PLZ,RANGE,1000,8999,MSG=MSG003)
   VER(&ORT,NB,MSG=MSG001)
   VER(&RAB,LIST,0,5,10,20,MSG=MSG004)
)END
```

Wertzuweisungen

Mit den Wertzuordnungen wird einer Variablen ein Wert zugewiesen. Dazu kann eine Zeichenfolge oder eine Variable dienen.

Wird eine Zeichenfolge zugewiesen sollte diese zwischen Apostrophe eingegrenzt sein. Zwingend ist dies, wenn die Zeichenfolge Sonderzeichen (auch Umlaute oder ß) oder Blanks enthalten soll.

Wird für die Zuweisung eine Variable genutzt, wird deren Inhalt übertragen. Bei der Auflösung von Variablen kommt es allerdings nur zu einer einmaligen Substitution (zeigt der Inhalt einer Variablen auf den Namen einer anderen Variablen, kann deren Inhalt nicht ermittelt werden).

Anders als bei Wertzuordnungen in REXX oder CLIST kann im Panel bei der Wertzuweisung keine Arithmetik aufgelöst werden. Erforderliche Berechnungen müssen in die aufrufende Funktion verlagert werden.

Feldprüfungen

Durch VERIFY kann der Inhalt einer Variablen überprüft werden. Die Prüfungen erfolgen in der Reihenfolge der VER-Instruktionen. Endet die Prüfung negativ, wird entweder ein Standardtext, oder eine vorbereitete Meldung ausgegeben. Anschließend wird (sofern vorhanden) der)REINIT-Teil angesprungen, bevor wieder in den)BODY-Teil verzweigt wird. Der CURSOR wird dabei auf das reklamierte Feld positioniert.

Bei positivem Ergebnis ist sichergestellt:

VERIFY	**Variableninhalt**
NB	Ungleich BLANK
RANGE,7,25	Numerisch zwischen 7 und 25
LIST,A,B,Z	Einer der drei angegebenen Werte
LISTV	Die zu prüfenden Werte liegen in einer Variablen
LEN,GT,7	Der Inhalt (Feldlänge) ist mindestens 8 Byte

Re-Initialisierung

Sofern der)REINIT-Teil kodiert ist, wird er durchlaufen, wenn im)PROC-Teil Fehler erkannt wurden, um vor erneuter Präsentation der Bildschirmmaske Veränderungen in Variablen oder Attributen zu erreichen.

In unserem zugrunde liegenden Beispiel wird bei Erkennen eines Fehlers das betreffende Feldattribut geändert, so dass das fehlerhaft ausgefüllte Feld auch optisch unter den restlichen Feldern hervortritt.

Um die Möglichkeiten der Ablaufsteuerung und Feldauffrischung zu demonstrieren wird unser Beispiel entsprechend erweitert:

```
:
+Rabatt       ==>^RAB@%
)INIT
   &NAME=&Z
   &STR=&Z
   &PLZ=&Z
   &ORT=&Z
   &RAB=&Z
   .CURSOR=&NAME
)REINIT
   .ATTR(.CURSOR)='INTENS(HIGH) COLOR(RED)'
)PROC
   VER(&NAME,NB,MSG=MSG001)
   VER(&VNAM,NB,MSG=MSG001)
   VER(&VNAM,LEN,'<=',2,MSG=MSG001)
   VER(&STR,NB,MSG=MSG001)
   VER(&PLZ,NB,NUM,MSG=MSG002)
   VER(&PLZ,RANGE,0,99999,MSG=MSG003)
   IF (&PLZ = '85774')
      &ORT = 'Unterföhring'
      REFRESH(ORT)
   VER(&ORT,NB,MSG=MSG001)
   VER(&RAB,LIST,0,5,10,20,MSG=MSG004)
)END
```

Ablaufsteuerung

Durch die Fragestellung mit IF kann der Inhalt einer Variablen überprüft und ein vom Prüfergebnis abhängiger JA- oder NEIN-Zweig durchlaufen werden.

Die Kodierung des NEIN-Zweiges ist nicht zwingend, sofern von der Ablauflogik her im NEIN-Fall keine gesonderte Verarbeitung durchgeführt werden soll.

Die Kodierung der IF-Klausel ist spaltensensitiv. Befehle, die an die erfüllte Bedingung des IF geknüpft sind müssen eingerückt und spaltengleich geschrieben werden. Sobald eine Zeile spaltengleich zum IF oder links davon kodiert wird, wird der IF-Zweig als beendet betrachtet. Wird ein NEIN-Zweig kodiert , ist er in der gleichen Spalte wie IF mit ELSE einzuleiten. Die abhängigen Befehle sind wiederum einzurücken und spaltengleich zu schreiben.

Als Vergleichsoperanden in der Abfrage stehen zur Verfügung:

Operator	Aussage
=, EQ	Gleich
^=, NE	Ungleich
>, GT	Größer als
>=, GE	Größer oder gleich als
^>, NG	Nicht größer als
<, LT	Kleiner als
<=, LE	Kleiner oder gleich zu
^<, NL	Nicht kleiner als

Feld-Aktualisierung

Ändert sich der Wert einer Variablen innerhalb des)PROC-Teils oder im)REINIT-Bereich des Panels, ist ein REFRESH nötig, damit der geänderte Wert der Variablen bei erneuter Panelanzeige registriert wird. Die Aktualisierung (REFRESHING) von Variablen kann gezielt auf eine oder mehrere Variable hin, oder pauschal für alle Variable durchgeführt werden:

```
REFRESH(var-1 [,var-n])
REFRESH(*)
```

Angenommen, ein Feld &MWST soll geprüft werden. Abhängig vom Ergebnis wird ein Feld auf "INLAND" oder "EXPORT" gesetzt.

```
)PROC
   VER(&MWST,LIST,0,19,MSG=MSG005)
   IF (&MWST = 0)
      &UMSART = 'EXPORT'
   ELSE
      &UMSART = 'INLAND'
   REFRESH(UMSART)
```

Zeichenfolgen abschneiden

Über TRUNCATE kann in Verbindung mit einer Wertzuweisung eine Zeichenfolge nach bestimmten Kriterien zerlegt werden. Das Zerlegen erfolgt, indem die Zeichenfolge auf ein bestimmtes Zeichen durchsucht wird. Wenn gefunden, wird alles bis hin vor das Trennzeichen einer Variablen zugeordnet. Der Rest wird in .TRAIL abgelegt. Das Trennzeichen geht verloren. Anderenfalls wird der gesamte Inhalt der Variablen übernommen.

Die zweite Möglichkeit besteht in der Angabe einer Byteposition. Hier wird von Byte 1 bis zur angegebenen Stelle eine Variable gefüllt, der Rest steht auch hier in der Steuervariablen .TRAIL zur Verfügung. Es geht nichts verloren.

Das Beispiel zeigt das Zerlegen eines Datumsfeldes anhand der enthaltenen Punkte:

```
)PROC
   VER(&DATUM,NB,PICT,'NN.NN.NNNN')
   &TT=TRUNC(&DATUM,'.')
   &REST=.TRAIL
   &MM=TRUNC(&REST,'.')
   &JJ=.TRAIL
```

Soll die im vorangegangenen Beispiel erzeugte Variable &MM inhaltlich überprüft und aufgrund des Ergebnisses eine Variable &MAXTAG mit den maximal möglichen Tagen abhängig zum Monat gefüllt werden, wäre im Grunde eine Mehrfachabfrage (vergleichbar zu SELECT in REXX) erforderlich. Da diese Mehrfachabfrage nicht existiert, muss anstelle dessen ein in sich geschachteltes IF/ELSE-Konstrukt treten. Das nachfolgende Beispiel zeigt ein solches Gebilde:

```
VER(&MM,RANGE,01,12)
IF (&MM = 02)
   &MAXTAG = 29
ELSE
   IF (&MM = 04,06,09,11)
      &MAXTAG = 30
   ELSE
      &MAXTAG = 31
```

Für eine sinnvolle Prüfung dieser Art, die auch ein mögliches Schaltjahr berücksichtigen kann, wäre allerdings die Einbettung von REXX-Funktionalität im Panel nötig, weil innerhalb des Panels keine Arithmetik stattfinden kann (wird später erklärt).

Zeichenfolgen übersetzen

Über die Funktion TRANSLATE kann in einer Wertzuweisung bei Erfüllung einer bestimmten Bedingung der zuzuweisende Wert bestimmt werden.

Auch hier soll vom Beispiel ausgehend eine Übertragung der Kundengruppe in die Variable &KUGRU abhängig vom Rabattsatz erfolgen. Wurde kein Rabatt gewährt, bleibt die Kundengruppe unverändert.

```
)PROC
   VER(&RAB,LIST,0,5,10,20,MSG=MSG003)
   &KUGRU = TRANS(&RAB,
          5,'Einzelhändler'
         10,'Großhändler-A'
         20,'Großhändler-B'
           *,&KUGRU)
)END
```

Steuervariablen

Panel-Steuervariablen weisen als Erkennungsmerkmal im Unterschied zu den Programm- oder Systemvariablen einen Punkt auf und werden genutzt, um bestimmte generelle Aussagen zu treffen.

Die wichtigsten Panel-Steuervariable in alphabetischer Reihenfolge und ihre mögliche Schreibweise sind:

Steuervariable	Bedeutung
.ALARM	Ausgabe Warnton
.ATTR	Temporäre Änderung eines Feldattributes
.ATTRCHAR	Änderung eines eingestellten Panel-Attributzeichens
.CURSOR	Cursorposition auf ein Feld
.CRSPOS	Cursorpositionierung innerhalb eines Feldes
.HELP	Zuweisung eines HELP-Panels
.MSG	Zeigen auf eine definierte Nachricht
.RESP	Ende der)BODY-Section durch ENTER, END oder RETURN
.TRAIL	Restinhalt einer Variablen nach TRUNCate-Funktion
.ZVARS	Reihenfolge der Platzhalter &Z im)BODY

Die folgenden Beispiele verdeutlichen den Umgang mit den Panel-Steuervariablen:

.ZVARS

Werden im)BODY-Teil Variable benötigt, deren inhaltliche Länge kürzer als ihre Variablennamen sein soll, kann als Platzhalter die Variable &Z für die betroffenen Felder benutzt werden.

```
)BODY
+Name        ==>_NAME                    +
+Vorname     ==>_VNAM                     +
+Alter       ==>_Z +
+Geschlecht  ==>_Z+(M/W)
+Beruf       ==>_Beruf           +
+Studium     ==>_Z+(J/N)
)INIT
   .ZVARS='(ALTER,MANNWEIB,STUDIUM)'
```

Im)INIT-Teil wird über die Steuervariable .ZVARS festgelegt, wie die Verbindung zwischen den Platzhaltern und den jeweiligen Variablen herzustellen ist. Die Platzhalter werden dabei im)BODY-Teil von links nach rechts und von oben nach unten gelesen und ihre Inhalte in dieser Reihenfolge den Variablen zugewiesen.

.ALARM .ATTR .ATTRCHAR .CSRPOS .MSG .RESP

Diese Steuervariablen werden häufig im Verbund in Verbindung mit Fehlerbehandlung genutzt.

```
)REINIT
   .ATTRCHAR(%)='INTENS(LOW)'
   .ATTRCHARS(_)='INTENS(LOW)'
   .ATTR(.CURSOR)='INTENS(HIGH)
)PROC
   IF (.RESP ^= 'ENTER')
      .MSG=MSG003
      .ALARM=YES
   VER(&DATUM,NB)
   &TT=TRUNC(&DATUM,'.')
   &REST=.TRAIL
   &MM=TRUNC(&REST,'.')
   &JJ=.TRAIL
   VER(&TT,NB,RANGE,1,31) .CSRPOS = 1
   VER(&MM,NB,RANGE,1,12) .CSRPOS = 4
   VER(&JJ,NB,RANGE,1950,2030) .CSRPOS = 7
)END
```

Werden im)PROC-Teil Fehler erkannt, können auf dem Rückweg zum)BODY im)REINIT-Teil Attribute temporär geändert werden, um das fehlerhafte Feld bei neuer Präsentation hervorzuheben.

Action Bars und Pull-Down-Menüs

Eine Variante, Auswahlmöglichkeiten in einem Panel zu treffen, ist der Einsatz von Action Bars und Pull-Down-Menüs. Oben am Panelrand wird ein "Balken" eingeblendet, in dem verschiedene ausführbare Aktionen stehen. Wird der Cursor auf ein Feld des Aktionsblocks gesetzt und ENTER gedrückt, erscheinen die jeweiligen Wahlmöglichkeiten in Form eines nach unten gezogenen Menüs (Pull-Down), in dem durch Cursorpositionierung oder durch Zahleneingabe selektiert werden kann.

Hier soll nicht auf die Details der ActionBar/Pulldown-Panel-Programmierung eingegangen werden, da dies den Rahmen sprengen würde. Dennoch werden die wichtigsten Elemente im Anschluss beschrieben. Ich muss mich an dieser Stelle auch outen. Ich habe eine tiefe Abneigung gegen Actionbars und PullDown Menüs – sowohl in der Kodierung als auch in der Bedienung, sofern ISPF nicht als Client/Server-Anwendung läuft und vollumfänglich maussensitiv ist. Nur unter diesen Umständen ist der Mehraufwand in der Kodierung durch den erhöhten Bedienerkomfort gerechtfertigt. Leider ist dies nur sehr selten der Fall (ich kenne derzeit kein Unternehmen, das ISPF als Client/Server fährt).

Action Bars

Action Bar ist die "Wortleiste", die am oberen Panelrand, welche die Namen von Auswahlgruppen enthält.

Pull-Downs

Pull-Downs erscheinen nach anwählen einer Auswahlgruppe der Action Bar. Die sichtbaren Möglichkeiten beziehen sich auf die angewählte Gruppe. Nach Auswahl (Cursor oder Tastatureingabe) kann bereits im Panel eine Aktion erfolgen, oder die Funktion im Hintergrund entscheidet, was zu tun ist.

Im Folgenden sind die dafür nötigen Bereiche beschrieben. Das Panel muss um die)ABC-Sektionen erweitert werden.

)ABC (Action Bar Choice Section)

Im)ABC-Teil wird ein Eintrag der Action Bar und die Untergruppen (Pull-Downs) definiert. Alle Einträge bilden die Action Bar. Sie wird von links nach rechts gezeigt, wie von oben nach unten kodiert.

)ABCINIT (Action Bar Initialization Section)

Der)ABCINIT-Teil wird durchlaufen, wenn ein Feld der Action Bar angesteuert wird. Danach wird das Pull-Down-Menü gezeigt.

)ABCPROC (Action Bar Processing Section)

Der)ABC-PROC-Teil ist wahlfrei. Wird er kodiert, muss er aber mindestens ein Statement enthalten. Der)ABCPROC-Teil wird durchlaufen, wenn eine Auswahl in einem Pull-Down-Menü getroffen wurde.

Die Reihenfolge

1.)ABC
2.)ABCINIT
3.)ABCPROC

muss eingehalten werden.

Panelzusätze für Action Bars

Eine Action Bar benötigt zusätzliche Attributtypen, die im)ATTR-Teil des Panels hinterlegt werden. Die wichtigsten Attribute sind:

TYPE(AB) Action Bar (Auswahlmöglichkeit)

Markiert eine Auswahlgruppe innerhalb der Auswahlleiste. Jeder Auswahlblock muss je aufs Neue mit dem Attribut TYPE(AB) gekennzeichnet werden. Zwischen Atrributszeichen und Text wird zwingend ein BLANK erwartet.

TYPE(NT) Normal Text

Erlaubt vergleichbar zu den herkömmlichen Textattributen die Definition von Farben und anderen Charakteristiken.

TYPE(ABSL) Action Bar Separator Line

Erzeugt eine Trennzeile zwischen Auswahlleiste und der restlichen Bildschirmmaske.

TYPE(PT) Panel Title

Liefert eine Panel-Überschrift.

Hinweis

Zwingend ist ausschließlich die Kodierung des Typs(AB). Die restlichen Attribute können prinzipiell herkömmlich kodiert werden (TYPE(TEXT) COLOR(GREEN) INTENS(LOW)....).

Beispiel

Das folgende Beispiel zeigt ein Action Bar Panel. Zur Erläuterung wird die Auswahlgruppe SORT beschrieben.

```
)ATTR FORMAT(MIX)
   + TYPE(TEXT)
   ! TYPE(AB)
   # TYPE(NT)
   [ TYPE(ABSL)
)ABC
   desc('Sort')
   pdc desc('Abteilung')
   pdc desc('Name')
   pdc desc('Ort')
)ABCINIT
   .zvars=pdw
   &pdw=&z
)ABCPROC
   ver(&pdw,LIST,1,2,3,4,5)
   &rxprog='SORT'
)BODY expand($$)
+#! Wartung     ! Sort     ! Druck    ! Ende#
```

```
[$-$ Trennzeile AB - Panel $-$
+$-$< Telefonbuch Grundmenu >$-$
+Command ===>_zcmd
+
)INIT
   &rxprog=&z
   &pdw   =&z
)PROC
)END
```

Der Funktion sieht man nicht an, dass die Entscheidungen mit Hilfe von Action Bar und Pull-Downs getroffen wurden:

```
/* REXX ***************************************/
do forever
   "display panel(actbar)"
   if rc=8 ! translate(rxprog)="ENDE" then leave
   if length(rxprog) > 0 then do
      interpret "call" rxprog
      iterate
   end
   /* Verarbeitung ohne Auswahl über Action-Bar */
end
exit
sort:
   select
      when pdw=1 then sk="....."
      when pdw=2 then sk="....."
      when pdw=3 then sk="....."
      otherwise nop
   end
   "tbsort telverz fields("sk")"
return
```

Auswahl-Bildschirme (Selection-Panels)

Der Unterschied zu Datenpanels besteht darin, dass hier normalerweise nicht mehrere Felder zur Dateneingabe präsentiert werden, sondern unter mehreren angebotenen Funktionen durch eine Eingabe entschieden wird, welche Aktion im Anschluss stattfinden soll (siehe Grundmenü des PDF).

Als mögliche Aktionen können Aufrufe für

- PANELS, die direkt aufgerufen werden, ohne die Steuerung an die Funktion zurückgeben zu müssen
- Loadmodules (Ablauffähige Programme), die innerhalb der ISPLLIB-Zuordnung zu finden sein müssen
- Prozeduren (CLIST oder REXX), die unter der Zuordnung von SYSPROC/SYSEXEC gesucht werden

stattfinden.

Im Auswahlpanel können die gleichen Sections wie in einem Datenpanel vorkommen.)BODY und)PROC sind zwingend. Durch die Angabe "&ZPRIM=YES" kann eine Auswahlmaske als so genanntes Primary-Option-Menü definiert werden. Die Folge daraus ist

- bei Benutzung der RETURN-Funktion wird in einem hierarchischen Dialog hier Halt gemacht
- Sprungbefehle wie "=3.2" können nur unterhalb dieser Hierarchiestufe benutzt werden.
- Das direkte Verlassen eines Dialoges durch "=X" bewirkt einen Rücksprung in die nächste hierarchisch übergeordnete Ebene.

Die)BODY-Section im Selection-Panel

Für die Kodierung gelten folgende Regeln:

- Es muss mindestens ein Eingabefeld für die Aufnahme der ausgewählten Option vorhanden sein. Im Allgemeinen wird hierfür das ZCMD-Feld benutzt, es kann aber auch das erste Feld nach dem ZCMD-Feld sein.
- Weitere Eingabefelder können vorhanden sein. In der Regel ist aber im Hinblick auf eine übersichtliche Bedienerführung möglichst darauf zu verzichten.
- Das Selection-Panel hat keinen Zugriff auf den Funktion-Pool. Es können nur Variable aus dem Shared-Pool oder Profile-Pool abgebildet werden.

Die)PROC-Section im Selection-Panel

Besonderheiten sind:

- Der &ZSEL-Service muss im)PROC-Teil kodiert sein. Hier erfolgt über eine eingebettete SELECT-Funktion die Entscheidung, welche Aktivität aufgrund der Eingabe eingeleitet werden soll.
- Wird eine gegliederte Auswahl getroffen wie beispielsweise 3.5, sorgt der Select Service für die entsprechende Weitergabe. Hierbei werden zu überspringende Panels verarbeitet, aber nicht angezeigt.

Als Beispiel für ein Auswahlpanel betrachten wir ein mögliches Primary-Option-Menü des ISPF/PDF:

```
)BODY EXPAND(//)
%/*/ ISPF/PDF PRIMARY OPTION MENU /*/
%OPTION ==>_ZCMD                                          +
%
% 0 + ISPF PARMS- Specify terminal
% 1 + BROWSE    - Display source data .........
% 2 + EDIT      - Create or change ...........
% 3 + UTILITIES - Perform utility functions ...
:
% T + TUTORIAL
%
% PH+ PHONE     - INTERNAL PHONEBOOK
)INIT
   .HELP=ISP00003
   &ZPRIM=YES
   &ZHTOP=ISP00003
)PROC
   &ZSEL=TRANS(TRUNC(&ZCMD,'.')
         0,'PANEL(ISPOPTA)'
         1,'PGM(ISRBRO) PARM(ISRBRO01)'
         2,'PGM(ISREDIT) PARM(P,ISREDM01)'
         3,'PANEL(ISRUTIL)'
         :
         T,PGM(ISPTUTOR) PARM(ISP00000)'
         :
         PH,'CMD(PHONBOOK)'
        :
```

Hierarchische Dialogstrukturen

Um die hierarchischen Strukturen eines Dialoges aufzuzeigen, betrachten wir uns PDF in graphischer Darstellung. Das hierarchisch oberste Panel ist das PRIMARY-OPTION-MENU. Es wird durch einen SELECT PANEL Service beim Start des PDF aufgerufen.

Untergeordnete Panels, die aus einem Programm aufgerufen werden, sind Datenpanels. Sie stellen das Ende einer Hierarchie dar und werden über den DISPLAY PANEL Service aktiviert.

Messages

Messages sind Nachrichten die situationsabhängig in den vorgesehenen Feldern SMSG (Short Message) oder LMSG (Long Message) präsentiert werden können.

Grundsätzlich unterscheiden wir zwischen Nachrichten, die innerhalb eines Panels aufgerufen werden und solchen, die über die steuernde Funktion (z.B.: REXX) abgehandelt werden. Die Verarbeitung in beiden Fällen ist jedoch die gleiche.

Alle im Dialog Manager zu verarbeitenden Meldungen müssen innerhalb einer unter ISPMLIB zugeordneten PO-Datei unter Berücksichtigung bestimmter Formalismen in beliebigen Membern untergebracht sein.

Meldungsdefinition

Beim Verweis auf eine Meldung geht aus der Meldungsdefinition der Membername innerhalb der ISPMLIB-Datei und die darin liegende gewünschte Meldung hervor.

```
a[aaaa]123[s]

  a   Meldungsvorspann. 1-5 Alphazeichen
123   Dreistellige Nummer
  s   Suffixzeichen
```

Die maximale Länge ist 8 Byte.

Das Suffixzeichen darf nur benutzt werden, sofern nicht bereits 5 Prefixzeichen benutzt wurden.

Der Membername, in welcher die Meldung innerhalb ISPMLIB gesucht wird, bildet sich aus dem Alpha-Prefix und den ersten beiden numerischen Werten. Eine Meldung AKUV001A muss also zwingend im Member AKUV00 in ISPMLIB gefunden werden.

Meldungsformat

Die Datei, in die die Meldungsmember eingetragen werden ist mit einer festen Satzlänge von 80 Byte zu erstellen. Das Dateiprofil muss auf "NUMBER OFF" gesetzt sein.

Jede Meldung besteht aus einem zweizeiligen Eintrag. Ab Spalte 1 der ersten Zeile steht die Message-ID. Anschließend folgt der kurze Meldungstext zwischen Hochkommata. Die kurze Nachricht darf maximal 24 Byte (Ohne Hochkommata) lang sein.

Der lange Meldungstext steht in der nächsten Zeile und ist ebenfalls zwischen Apostrophe eingeschlossen. Die maximale Länge ist 78 Byte.

In beiden Meldungstextzeilen können Variable innerhalb der Hochkommata genutzt werden. Die Variablen werden vor Anzeige der Meldung substituiert.

Sinnvoll sind auch hier sprechende Namen. So ist empfehlenswert, die Message-ID und damit den Membernamen innerhalb von ISPMLIB mit der Applikation in Verbindung zu bringen, für die das Meldungspaket genutzt werden soll. Neutrale Namen können für Meldungen genutzt werden, die nicht applikationsspezifisch sind.

Schlüsselwortparameter

Die Aussagekraft der Messages kann mit verschiedenen Parametern verstärkt werden. Die Schlüsselwortparameter werden im Anschluss an die kurze Meldung kodiert.

.HELP=hpname

zeigt auf ein HELP-Panel, das präsentiert wird, wenn nach der langen Meldung die HELP-Taste gedrückt wird.

.ALARM=YES|NO

steuert, ob ein akustisches Signal bei Erscheinen der Meldung ausgelöst werden soll.

.WINDOW=RESP|NORESP

bewirkt, dass die Meldung in einem Fenster erscheint. RESP erzwingt das Drücken der ENTER-Taste nachdem die Meldung erschienen ist, NORESP nicht.

.TYPE=NOTIFY|WARNING|ACTION|CRITICAL

verändert Farbe und Darstellung der Meldung je nach Wichtigkeit, in der folgenden Tabelle die Übersicht:

Type	Farbe	Erscheint	Enter zwingend	Alarm
Notify	Weiß	Normal oder als Fenster	Nein	Nein
Warning	Gelb	Normal oder als Fenster	Nein	Nein
Action	Rot	Normal oder als Fenster	Nein	Ja
Critical	Rot	Immer als Fenster	Nein	Ja

Beispiele

```
FAKT002A 'Datumsformat falsch' .HELP=hpname
'Das Datumsformat muss TT.MM.JJJJ sein!'
```

Messagebeispiel mit HELP-Panel. Wird nach dem Erscheinen der langen Meldung die HELP-Taste gedrückt, wird das angegebene HELP-Panel nachgerufen.

```
FAKT003A 'Wert falsch!' .WINDOW=NORESP .ALARM=YES
'Der Wert muss zwischen 1 und 5 liegen.'
```

Lässt die Nachricht in einem Fenster erscheinen (ohne Bestätigung mit ENTER) und den Warnton erklingen.

Anzeige der Meldungen

Sind keine benutzereigenen Meldungen definiert, werden vom Dialog Manager Standardmeldungen benutzt.

Anzeige von HELP-Panels

Generell können an HELP-Panels weitere HELP-Panels angeschlossen werden. Ist ein HELP-Panel sichtbar, kann der Benutzer des Dialoges durch drücken der ENTER-Taste von einem HELP-Panel zum Nächsten blättern.

Ablauf der Meldungsanzeige

Es ist ein Fehler aufgetreten.

1. Sofern eine Meldung definiert ist, wird die Kurznachricht ausgegeben.
2. Wird HELP gedrückt, wird die Langnachricht ausgegeben.
3. Wird HELP gedrückt, wird das Help-Panel angezeigt

Ausgabeort der Meldungen

Soll vom Standard abgewichen werden (kurze Meldung in Zeile 1 rechts, lange Meldung in Zeile 3) sind entsprechende Definitionen im)BODY-Teil nötig.

Im)ATTR-Teil muss für die Meldungsausgabe ein Attribut mit Type Output definiert sein, da es ein solches standardgemäß nicht gibt.

```
)ATTR
   $ TYPE(OUTPUT) INTENS(HIGH)
)BODY EXPAND(//) SMSG(KMESS) LMSG(LMESS)
+/-/ Überschrift /-/
+COMMAND ==>_ZCMD                                +
:
:   Text-, Eingabe- und Ausgabefelder
:
$KMESS   +
$LMESS                                           +
```

Die kurze Nachricht erscheint in der vorletzten Zeile, die lange Nachricht in der letzten Zeile jeweils linksbündig.

Steuerung der Meldungs-Anzeige

Die Anzeige der Meldungen kann entweder über Angaben in der)PROC-Section oder durch eine Dialog Manager Routine in einer Funktion (REXX) gesteuert werden. Wir sehen uns zu beiden Möglichkeiten jeweils ein Beispiel an:

```
)PROC
   IF (&RAB='')
      .MSG = FAKT001A
   VER (&ORT,NB,MSG=FAKT002A)
   :
)END


:
DO FOREVER
   "DISPLAY PANEL(pname)"
   IF DATOK(datum)=1 THEN LEAVE
   "SETMSG MSG(FAKT003A)"
END
:
```

Tabellen

Eine Tabelle ist ein im Allgemeinen zweidimensionales Gebilde aus Spalten und Zeilen, wobei spaltengleich sinngemäß gleiche Inhalte abgebildet sind.

Alle zu einer Ebene (Zeile) zusammengefassten Daten ergeben einen Datensatz. Die einzelnen Spalteneinträge innerhalb des Satzes bezeichnet man als Glied oder Feld oder Tabellenelement.

Tabellen werden als Member von PO-Dateien geführt, die unabhängig von der Größe der Tabelle mit einer festen Satzlänge von 80 Byte angelegt werden. Die symbolischen Namen für das Lesen und Schreiben von Tabellen sind unterschiedlich. Das Lesen wird über den symbolischen Namen "ISPTLIB", das Schreiben über "ISPTABL" durchgeführt, wobei durchaus beide symbolischen Namen auf die gleiche PO-Datei zugewiesen sein können.

Bei Verarbeitung einer Tabelle wird diese zunächst komplett in den virtuellen Speicher gelesen. Sämtliche Manipulationen an der Tabelle finden hier statt. Nach Veränderungen wird die gesamte Tabelle gegebenenfalls wieder auf Platte zurück geschrieben.

Die Größe einer Tabelle ist theoretisch beschränkt. In der Praxis stellen diese Grenzen aber im Allgemeinen keine Einschränkung dar. Die maximale Feldgröße (Variablengröße) einer Dialog Manager Tabelle liegt bei 32 Kb. Die maximale Gesamtgröße ist abhängig von der Größe des verfügbaren virtuellen Speichers.

Zugriff auf Tabellensätze

Der Zugriff erfolgt immer auf eine komplette Zeile aus der Tabelle, unabhängig, ob eine sequentielle Verarbeitung durchgeführt wird, oder ob über SCAN-Vorgänge nach bestimmten Kriterien gefiltert werden soll.

Der aktuelle Tabellensatz wird über einen so genannten CRP (Current Row Pointer oder Satzzeiger) markiert. Der CRP kann vorwärts und rückwärts positioniert werden. Nach dem Öffnen einer Tabelle steht der CRP auf "TOP" (am Anfang der Tabelle noch vor dem ersten Datensatz).

Die Inhalte aller Tabellenfelder des Satzes, auf den der CRP zeigt sind unmittelbar über gleichnamige Variable im Function-Pool verfügbar.

Sollen Tabelleneinträge verändert werden muss bei Tabellen ohne Schlüssel zunächst der CRP auf den zu ändernden Satz positioniert werden. Anschließend kann der Inhalt der feldnamensgleichen Variablen entsprechend verändert werden, bevor eine Aktualisierung der Tabelle mittels DMS-Befehl erfolgt. Alle Änderungen sind zunächst nur temporär, da sie nur im virtuellen Speicher erfolgen.

Tabellen erzeugen

Um eine Tabelle erstellen zu können, muss man sich zunächst mit zwei Begriffen auseinandersetzen:

KEYS

Keys benennt die Schlüsselfelder. Prinzipiell kann eine Tabelle ohne Schlüsselfelder definiert werden. Es können aber auch beliebig viele Felder zum Schlüssel erhoben werden.

Sinn des Schlüssels ist, die Eintragung zweier Sätze, die in allen Schlüsselfeldern identisch sind, zu verhindern.

NAMES

beschreibt alle Tabellenfelder über die Schlüsselfelder hinaus. Soll eine Tabelle keine Schlüssel enthalten, müssen alle Felder innerhalb NAMES angegeben werden.

Feldformate und -längen

Angaben über Feldlängen und Feldformate sind nicht erforderlich. Die Felder werden in Bezug auf Länge und Format so in die Tabelle eingearbeitet, wie sie als Variable im Function-Pool vorbereitet werden.

Sortierung

Häufig wird nach dem Erstellen einer Tabelle (sofern sie sortiert vorliegen soll) mit der Instruktion "TBSORT FIELDS(f1,x,y[...])" die Sort-Order innerhalb der Tabelle festgelegt. Über einen Operanden "ORDER" kann später auf die festgesetzte Sortierfolge Bezug genommen werden.

Tabellen einlesen

Über die Instruktion "TBOPEN tabname" wird ein Abbild der Tabelle im virtuellen Speicher erstellt. Durch entsprechende Parameter kann gesteuert werden:

- Die Tabelle wird nur temporär geführt. Beim schließen findet kein Rückschreiben auf Platte statt.
- Die Tabelle wird beim Schließen auf Platte zurück geschrieben.
- Die Tabelle wird beim Rückschreiben unter einem anderen als dem vorgesehenen symbolischen Namen abgestellt (nicht empfehlenswert ohne Zwang).
- Die Tabelle kann am Terminal im SPLIT-SCREEN-Modus verarbeitet werden.

Tabelle schreiben

Zunächst sind zwei unterschiedliche Befehle für das Rückschreiben einer Tabelle verfügbar. Mit TBSAVE kann ein Rückschreiben der Tabelle erfolgen, ohne diese schließen zu müssen. Mit "TBCLOSE tabname" erfolgt das Rückschreiben in Verbindung mit dem Schließen der Tabelle. Die Tabelle wird im virtuellen Speicher gelöscht.

Die Schreibvorgänge finden über den symbolischen Namen "ISPTABL" statt. Lesevorgänge werden über "ISPTLIB" durchgeführt. In der Praxis zeigen beide Namen auf die gleiche Datei.

Nachdem DMS-Tabellen als Member von PO-Dateien geführt werden, muss (speziell bei größeren Tabellen) die Problematik der Mitgliedsleichen beachtet werden. Um nicht permanente Verarbeitungsabbrüche wegen zu kleiner Dateien hinnehmen zu müssen, kann beim Schreiben zweierlei versucht werden:

Mit REPLCOPY wird versucht, die Tabelle an die Stelle zurück zu schreiben, von der sie gelesen wurde. Ist dies aus Platzgründen nicht möglich, wird die Tabelle am Ende angefügt (die Mitgliedsleiche ist erzeugt).

Über "PAD(prozentwert)" kann beim ersten Schreiben, und immer dann, wenn sich der Standort der Tabelle verändert, ein bestimmter Prozentualwert im Verhältnis zur aktuell benötigten Größe als Vorhalt angegeben werden. Findet keine Ortsveränderung der Tabelle statt, wird die Angabe ignoriert.

Temporäre Tabellen

existieren ausschließlich im virtuellen Speicher. bei Ende der Tabellenverarbeitung werden sie nicht unter "ISPTABL" auf Platte geschrieben. Wie bei permanenten Tabellen wird bei Ende der Verarbeitung die Tabelle aus dem virtuellen Speicher gelöscht.

Temporäre Tabellen können über die DMS-Befehle "TBOPEN tabname" oder "TBCREATE tabname" mit dem Attribut "NOWRITE" erzeugt und durch "TBEND tabname" im virtuellen Speicher gelöscht werden.

Tabellenverarbeitende Befehle

Über eine Vielzahl von Befehlen kann die Bearbeitung eines Tabellensatzes erfolgen. Für einige Befehle ist Bedingung, dass die einzutragenden Feldinhalte in gleichnamigen Variablen im Function-Pool vorbereitet sind. Die Basisbefehle sind:

TBTOP

Positioniert den CRP vor den ersten Datensatz. Dies ist immer sinnvoll, wenn ein Suchvorgang in einer Tabelle gestartet werden soll und nicht sicher ist, ob der Satzzeiger im Moment auf Tabellenanfang steht.

TBBOTTOM

setzt den Satzzeiger auf den letzten Datensatz und überführt dessen Feldinhalte in Variable in den Function-Pool, sofern nicht NOREAD angegeben ist.

TBEXIST

erlaubt die Überprüfung, ob ein Eintrag in einer Tabelle mit Schlüssel bereits existiert. Ist dies der Fall, ist im Anschluss an TBEXIST die Variable, die den Returncode enthält auf 0 gesetzt.

TBDELETE

führt zum Löschen des Tabellensatzes auf den der CRP zeigt.

TBADD

erlaubt das Hinzufügen eines Tabellensatzes. Mit der Angabe "ORDER" wird der Satz bei einer sortierten Tabelle an der entsprechenden Position eingearbeitet.

TBPUT

führt zur Veränderung des Tabellensatzes, auf den der CRP zeigt.

Bei Tabellen mit Schlüssel wird die Veränderung nur durchgeführt, wenn die Schlüsselfelder des aktuellen Tabellensatzes inhaltlich mit den gleichnamigen Variablen im Function-Pool übereinstimmen.

Bei Tabellen ohne Schlüssel wird der Satz geändert, auf den der CRP zeigt.

Es ist also generell dafür zu sorgen, dass der CRP auf den zu ändernden Tabellensatz positioniert wird.

TBMOD

Vereint die Funktionen aus TBADD und TBPUT.

Zunächst wird geprüft, ob die gesuchte Zeile in der Schlüsselfeldtabelle enthalten ist. Ist dies der Fall, wird der betreffende Tabellensatz aktualisiert. Anderenfalls wird ein neuer Satz in die Tabelle eingefügt.

TBMOD ist bei Tabellen ohne Schlüssel gleichbedeutend mit TBADD.

TBSKIP

verändert den Standort des CRP nach oben oder unten innerhalb der Tabelle und überstellt die Inhalte aller Felder in gleichnamige Variable im Function-Pool.

TBGET

liefert beispielsweise die Namen so genannter SAVE-Variabler, die nur bei einzelnen Sätzen angehängt wurden.

TBERASE

löscht eine Tabelle aus ISPTABL, sofern sie nicht innerhalb der gleichen Datei mit dem Attribut WRITE in einem TBOPEN-Befehl eröffnet wurde

Suche in Tabellen

Um eine Tabelle nach bestimmten Kriterien zu verarbeiten, ohne dabei zu einem sequentielles Lesen in einer Programmschleife gezwungen zu sein, werden zwei Befehle angeboten, die es erlauben eine Tabelle so zu betrachten, als wären nur Sätze mit bestimmten Feldinhalten vorhanden. Alle Tabellenzeilen, auf die bestimmte Angaben nicht zutreffen werden als nicht vorhanden betrachtet. Diese Möglichkeit erleichtert Suchvorgänge innerhalb der Tabelle ganz wesentlich. Die beiden möglichen Verfahren sind:

Zunächst werden durch "TBVCLEAR" alle Variablen im Function-Pool für die es innerhalb der geöffneten Tabelle gleichnamige Felder gibt, gelöscht.

Anschließend werden die Felder und deren Inhalte, nach denen gesucht werden soll in Form von Variablen in den Function-Pool eingetragen. Mit der Instruktion "TBSARG" (TaBle SearchARGument) wird dem nachfolgenden "TBSCAN" mitgeteilt, dass das Scannen der Tabelle über ein vorbereitetes Suchargument erfolgen muss.

"TBSCAN" kann nun in einer Verarbeitungsschleife abgesetzt werden. Vorausgesetzt, der CRP wird vor Beginn dieser Verarbeitung auf den Tabellenanfang gesetzt, kann die Verarbeitung über den Returncode des "TBSCAN" gesteuert werden.

Das folgende Beispiel zeigt ein REXX-Programm zur Abarbeitung einer Tabelle bei der nach bestimmten Sätzen gesucht werden soll:

```
/* REXX * TABUPD
*/
address ispexec
"DISPLAY PANEL(panelname)"
/* Angenommen: Im Panel wird das Feld "GRUPPE" gefüllt,
   das mit ARTGRU aus der Tabelle gleichzusetzen ist */
"TBOPEN ARTIKEL"
"TBVCLEAR ARTIKEL"
ARTGRU=GRUPPE
"TBSARG ARTIKEL"
"TBSCAN ARTIKEL"
DO WHILE RC = 0
   /* Hier liegt die Verarbeitung */
   "TBSCAN ARTIKEL"
END
"TBCLOSE ARTIKEL"
EXIT
```

Ergänzend zu diesem standardisierten Verfahren gibt es zwei Besonderheiten, über die die Suchvorgänge in der Tabelle gesteuert werden können:

Generic Search: Sollen alle Sätze mit Postleitzahlen, die mit "80" beginnen gesucht werden ist ein füllen der entsprechenden Variablen in der Form "PLZ='80*'" möglich.

Bedingtes Suchargument: Durch "TBSARG tabname NAMECOND(PLZ,GE)" werden alle Sätze zur Verarbeitung ausgewählt, bei denen die gesetzte Bedingung erfüllt ist.

Als Vergleichsoperationen für die bedingte Suche können die üblichen Operanden eingesetzt werden: EQ, NE, LE, LT, GE und GT

TBSCAN suchargument

Werden im "TBSCAN" die Suchargumente direkt angegeben, können sowohl "TBVCLEAR" als auch "TBSARG" entfallen. Sollten zum Zeitpunkt des Scanvorganges entsprechende Variable im Function-Pool eingetragen sein, werden sie nicht beachtet.

Betrachten wir unter diesen Voraussetzungen das gleiche Beispiel

```
/* REXX * TABUPD ************************/
address ispexec
"DISPLAY PANEL(panelname)"
"TBOPEN ARTIKEL"
ARTGRU=GRUPPE
DO FOREVER
     "TBSCAN ARTIKEL ARGLIST(ARTGRU)" ,
     "CONDLIST(EQ)"
     IF RC > 0 THEN LEAVE
     /* Hier liegt die Verarbeitung */
END
"TBCLOSE ARTIKEL"
EXIT
```

Positionierung des Satzzeigers (CRP)

Bei Benutzung diverser Tabellenkommandos sind die Angaben POSITION und ROWID möglich. In beiden Fällen wird der Name einer Variablen angegeben, in die der Dialog Manager den Wert des CRP hinterlegt. Die Variablen enthalten bei

- ROWID den Wert des CRP vor Ausführung des Befehles
- POSITION den Wert des CRP nach Befehlsausführung

Beide Operanden sind verfügbar bei den Befehlen

- TBBOTTOM
- TBDISPL
- TBGET
- TBSCAN
- TBSKIP

Wie nach einem "TBSCAN" auf den Tabellensatz rückpositioniert werden kann, auf dem der CRP vor "TBSCAN" stand zeigt das Beispiel

```
"TBSCAN tabname ROWID(OLDCRP)"
:
"TBSKIP tabname ROW(&OLDCRP)"
```

Tabellenanzeige (TBDISPL)

Panels, die über den Service TBDISPL angezeigt werden, weichen in ihrem Aufbau von normalen Datenpanels ab.

Im)BODY wird der konstante Teil des Bildschirmes beschrieben (Bildschirm-Titelzeile, Kommandozeile, SCROLL-Feld und die Spalten-Überschriften)

Im)MODEL wird das Muster für die Abbildung eines Satzes beschrieben. Dieser Satz kann sich über maximal acht Bildschirmzeilen erstrecken. Dieser variable Teil der Maske wird bis zum Bildschirmende hin dynamisch erweitert.

Das 2.Feld im)BODY muss vier Stellen lang sein und enthält den Wert für das Scrolling. Der Feldname ist wahlfrei.

Soll eine Tabelle nur angezeigt werden, müssen alle Felder mit dem Attribut OUTPUT belegt werden.

Sollen nicht alle Sätze über den TBDISPL angezeigt werden, können durch den TBSARG die Suchargumente festgelegt werden. Im Panel des TBDISPL kann durch eine Angabe ROWS(SCANN) in der)MODEL-Zeile erreicht werden, dass nur Sätze gezeigt werden, die dem Suchargument entsprechen.

Werden gültige ISPF-Kommandos oder Scroll-Werte eingegeben, bleibt die Steuerung beim TBDISPL, der die entsprechenden Aktionen durchführt.

Mit der END-Funktion, oder wenn ein nicht gültiges ISPF-Kommando eingegeben wird, geht die Steuerung an die aufrufende Funktion zurück.

Tabellen-Änderungen

Nachdem zur Anzeige einer Tabelle nur eine einzige Musterzeile im)MODEL-Teil definiert wurde, werden alle Tabellensätze über die Feldnamen dieser einen Musterzeile verarbeitet. Diese Situation würde normalerweise bei Full-Screen-Updates ziemliche Probleme aufwerfen. Hier setzt ein weiterer Komfort des Dialog Managers ein:

In der DMS-Systemvariablen &ZTDSELS wird die Anzahl der im Panel gezeigten, geänderten Zeilen abgelegt, wenn die ENTER-Taste gedrückt wurde.

Nach <ENTER> steht der CRP auf dem ersten modifizierten Satz. Die Feldinhalte stehen in gleichnamigen Variablen im Function-Pool.

Durch nachfolgende TBDISPL ohne Panelangabe wird der CRP auf den nächsten modifizierten Satz gesetzt und &ZTDSELS um 1 vermindert. In dieser Form wirkt der TBDISPL wie ein Scanvorgang.

Generelles Tabellenhandling

In der Praxis wird häufig ein Verfahren angewandt, das im ersten Schritt die gewünschten Sätze anzeigt (alle Felder OUTPUT). Vergleichbar zu den EDIT-LineCommands wird vor der Tabellenzeile ein Feld gesetzt, über welches die Verarbeitung gesteuert wird (z.B.: A=Ändern, L=Löschen etc.).

Zu beachten ist, dass es sich bei diesem Feld meist nicht um ein Tabellenfeld handelt und es somit durch den Befehl TBVCLEAR nicht berührt wird.

Auch in dieser Form wird über die Variable &ZTDSELS erkannt, für wie viele Tabellenzeilen das Verarbeitungskennzeichen gesetzt wurde. Die Betreffenden Zeilen können analog zur vorangegangenen Beschreibung in einer Schleife abgearbeitet werden.

Systemvariable bei Tabellenanzeige

Um diverse brauchbare Informationen innerhalb der Tabellenanzeige zur Verfügung zu haben, können einige Systemvariable des Dialog Managers genutzt werden:

Systemrvariable	**Bedeutung**
&ZTDSELS	Enthält die Anzahl der modifizierten Tabellensätze
&ZTDTOP	Übergibt die Nummer des obersten am Bildschirm sichtbaren Tabellensatzes
&ZTDROWS	Liefert die Anzahl der Tabellenzeilen die am Bildschirm abgebildet werden können (wie im)MODEL-Teil beschrieben)
&ZTDMARK	Kann eine andere Schlusszeile als die standardgemäße "***BOTTOM OF DATA***" erzeugen. Diese Variable wird im)INIT-Teil gefüllt. Häufig wird der zuzuordnende Wert aber bereits in REXX in einer Variablen hinterlegt, da hier wesentlich komfortablere Funktionen zur Stringbehandlung zur Verfügung stehen. Diese Variablen stellen lediglich einen Auszug dar. Weitere Systemvariable des Dialog Managers können im Teil SYNTAX nachgelesen werden.

Diese Variablen stellen lediglich einen Auszug dar. Weitere Systemvariable des Dialog Managers können im Teil SYNTAX nachgelesen werden.