

INTERNSCHRIFT Nr. 42

THEMA: Das Datenmanagement im BSM

VERFASSER: Heupel, Jammel,
Plickert, Stiegler

DATUM: 21.5.1970

FORM DER ABFASSUNG

ENTWURF

AUSARBEITUNG

ENDFORM

SACHLICHE VERBINDLICHKEIT

ALLGEMEINE INFORMATION
DISKUSSIONSGRUNDLAGE

ERARBEITETER VORSCHLAG

VERBINDLICHE MITTEILUNG
VERALTET

ÄNDERUNGSZUSTAND

BEZUG AUF BISHERIGE INTERNSCHRIFTEN

Vorkenntnisse aus:

Erweiterung von:

Ersatz für:

BEZUG AUF KÜNFTIGE INTERNSCHRIFTEN

Vorkenntnisse zu:

Erweiterung in:

Ersetzt durch:

ANDERWEITIGE LITERATUR

Korrekturen zu IS 42

Seite 4

Zeile 5

"Datenbasen oder Dateien $\alpha_0 \dots \beta_1$ "
statt "Datenbasen $\alpha_0 \dots \beta_1$ "

8. Zeile von unten

"Datenbasen und Dateien"

statt "Datenbasen"

Seite 7

Zeile 5

"Benutzerverzeichnis (BVZ)"
statt "Benutzerverzeichnis (BV)"
(auch an allen anderen relevanten
Stellen)

Seite 18

Zeile 3 und 4

"oder Datei" streichen

3. Zeile von unten

"im Behälter nicht aus
2 Teilen" statt "nicht
aus 2 Teilen"

Seite 29

7. Zeile von unten

" SR_j " statt " SR_i "
" R_j " statt " S_j "

	<u>Inhaltsverzeichnis</u>	Seite
1.	Die Datenbasis	1
1.1	Sprachmittel	1
1.1.1	Namen (von Datenbasen und Dateien)	1
1.1.1.1	Syntax	1
1.1.1.2	Semantik	3
1.1.2	Einrichtung von Datenbasen	3
1.1.2.1	Syntax	3
1.1.2.2	Semantik	3
1.1.3	Aufgabe von Datenbasen	4
1.1.3.1	Syntax	4
1.1.3.2	Semantik	4
1.2	Listen	5
1.2.1	Listenkopf einer Datenbasis	5
1.2.2	Beschreibungsteil einer Datei	6
1.2.3	Listenrumpf einer Datenbasis	6
1.3	Spezielle residente Datenbasen	7
2.	Die Datei	8
2.1	Dateitypen bezüglich des satzweisen Zugriffs	8
3.	Das Gebiet	9
3.1	Motivation	11
3.2	Inhalt des Gebiets	11
3.3	Dynamisches Verhalten	11
3.3.1	Einrichtung eines Gebiets	11
3.3.2	Verlängerung und Verkürzung eines Gebiets	12
3.3.3	Aufgeben eines Gebiets	12

	Seite
4. Der Behälter	13
4.1 Motivation	13
4.1.1 Hardwareadressenverzeichnis	13
4.1.2 Transportgröße	14
4.1.3 Vermeidung von Querverweisen	15
4.2 Auslagerung	15
4.2.1 Algorithmus	15
4.2.2 Plattenkanalbelastung	15
4.3 Behältertypen	16
4.4 Verwaltung von Behältern	17
4.4.1 Behälterliste	17
4.4.5 Verteilung der Gebiete auf die Behälter	18
4.5 Inhalt des Behälters	18
4.5.1 Listen des Behälters	20
4.5.2 Längenänderung von Gebieten	21
4.5.2.1 Verlängerung an einer Bereichsgrenze	21
4.5.2.2 Verlängerung am Gebietsende	22
4.6 Schlußbemerkung	22
5. Die Abbildung von Datenbasen auf Gebiete	23
6. Die Abbildung von Dateien auf Gebiete	23
6.1 Motivation	23
6.2 Stellvertreterliste	26
6.3 Indexhierarchie	27
6.4 Liste der Überlaufbereiche (LDR)	29
6.5 Suchalgorithmen	30
6.6 Ablage der Sätze	31
6.6.1 Puffer	31
6.6.2 Ablage bei speichersequentieller Anlieferung	31

6.6.3	Ablage bei wahlfreier Anlieferung	32
6.6.4	Überlauf in R	32
6.6.5	Pufferpool für R	32
6.6.6	Überlauf in I	33

Sachverzeichnis	34
-----------------	----

1. Die Datenbasis

Der Benutzer legt Information in Dateien in Form von Sätzen ab. Für Zwecke höherer Gliederung bedient er sich der Datenbasen, indem er von einer Datenbasis auf andere Datenbasen und/oder Dateien verwiesen wird. Die Einrichtung (Kreation) der benötigten Datenbasen und Dateien obliegt dem Benutzer. Jedoch ist sie ihm nur in der Weise möglich, daß die Gesamtheit aller innerhalb des Datenmanagements existierenden Datenbasen und Dateien bzgl. der zwischen ihnen bestehenden Verweise einen Baum bilden. Dieser Baum heißt Dateienbaum, seine Wurzel Zentrale Datenbasis.

Datenbasen sind Listen von Verweisen.

Es können temporäre und permanente Datenbasen und Dateien eingerichtet und aufgegeben werden. Temporäre Datenbasen und Dateien existieren nur während des Abschnitts (Jobs) in dem sie eingerichtet werden, permanente hängen in ihrer Lebensdauer nicht von Abschnittsgrenzen ab. Daneben existieren noch residente Datenbasen und Dateien, die ausschließlich für Systemzwecke zur Verfügung stehen. Auch sie können eingerichtet und aufgegeben werden.

1.1 Sprachmittel

Dem Benutzer stehen mindestens folgende Sprachmittel zur Verfügung.

1.1.1 Namen (von Datenbasen und Dateien)

1.1.1.1 Syntax

$$\langle \text{qualifizierter Name} \rangle \rightarrow \langle \text{einfacher Name} \rangle \{ . \langle \text{einfacher Name} \rangle \}_0^\infty$$

$$\langle \text{einfacher Name} \rangle \rightarrow \langle \text{Anfangszeichen} \rangle \{ \langle \text{Zeichen} \rangle \}_0^{11}$$

Als Zeichenvorrat kommt zunächst die Menge der Buchstaben und Dezimalziffern in Betracht, als Anfangszeichenvorrat Buchstaben.

Jedoch ist für bestimmte Anwendungen ein um einige Sonderzeichen (etwa / , - $\frac{1}{2}$ $\frac{1}{4}$) erweiterter Zeichenvorrat wünschenswert. Außerdem ist zu überlegen, ob einem ausgezeichneten Benutzerkreis (etwa S-Bearbeitern, Datenmanagement) ein erweiterter Anfangszeichenvorrat zur Verfügung gestellt wird (aus Gründen der Zugriffssicherung etwa) ¹⁾ .

1) Namenskonventionen in anderen Systemen:

(i) MTS: es gibt drei Filetypen

userfile: Namen bestehen aus maximal 12 alpha/numerischen Zeichen, beginnend mit einem Buchstaben, werden mehr als 12 Zeichen angeboten, so werden die letzten ignoriert.

scratchfile: Namen beginnen mit -, es folgen maximal 8 alpha/numerische Zeichen, mindestens eines.

libraryfile: Namen beginnen mit *, es folgen maximal 15 alphanumerische Zeichen, mindestens eins.

MTS empfiehlt dringend, nur alpha/numerische Zeichen zuzulassen, wegen der uneinheitlichen Codierung der Sonderzeichen. Kleinbuchstaben sind mit Großbuchstaben äquivalent.

(ii) Telefunken

Datenbasis: Maximal 6 alphanumerische Zeichen, beginnend mit einem Buchstaben.

Datei: Maximal 12 Zeichen, alphanumerisch und &, beginnend mit einem Buchstaben oder &.

(iii) IBM OS/360

einfache Namen: Maximal 8 alphanumerische Zeichen, beginnend mit einem Buchstaben;

qualifizierte Namen: einfache Namen, getrennt durch Punkte, jedoch nicht mehr als 44 Zeichen.

1.1.1.2 Semantik

Sind $\alpha_0, \dots, \alpha_n$, $n \geq 1$ Zeichenfolgen mit der Syntax einfacher Namen und existiert eine Datenbasis oder eine Datei mit dem Namen $\alpha_0 * \dots * \alpha_n$, so existieren n Datenbasen mit den Namen $\alpha_0 * \dots * \alpha_k$, $k = 0, \dots, n - 1$, und die Datenbasis $\alpha_0 * \dots * \alpha_k$ enthält einen Verweis auf die Datenbasis oder Datei $\alpha_0 * \dots * \alpha_{k+1}$, $k = 0, \dots, n - 1$, und dies ist auch der einzige Verweis auf diese Datenbasis oder Datei. $\alpha_0 * \dots * \alpha_k$ heißt die $\alpha_0 * \dots * \alpha_{k+1}$ übergeordnete Datenbasis und α_{k+1} einfacher Name von $\alpha_0 * \dots * \alpha_{k+1}$ in $\alpha_0 * \dots * \alpha_k$.

1.1.2 Einrichtung von Datenbasen1.1.2.1 Syntax

Der Einrichte-Befehl enthält (u.a.) einen qualifizierten Namen, optional eine Anzahl einfacher Namen und die Beschreibung der Zugriffsberechtigungen.

Es ist noch nicht entschieden, ob die Spezifikation "Temporär" bzw. "Permanent" über einen Parameter des Einrichte-Befehls angegeben wird oder ob es zwei entsprechende Einrichte-Befehle geben wird.

1.1.2.2 Semantik

Der Dateienbaum wird erweitert. Die einfachen Namen werden benötigt, wenn eine Datenbasis im Dateienbaum "eingeschoben" und nicht am "Ende angefügt" werden soll. Sei der spezifizierte Name $\alpha_0 * \dots * \alpha_n$. Eine Datenbasis dieses Namens wird eingerichtet, unter folgenden zwei Bedingungen:

- (i) Für den Benutzer, für den die Datenbasis eingerichtet werden soll, existiert noch keine Datenbasis mit diesem Namen oder α_n wurde unter dem einfachen Namen aufgeführt.

Jener Benutzer wird Eigentümer der Datenbasis. Der Eigentümer einer Datenbasis ist also auch Eigentümer aller in ihr eingetragenen Datenbasen/Dateien.

- (ii) Ist $n > 0$, so muß jener Benutzer bereits Eigentümer einer Datenbasis mit dem Namen $\alpha_0 \dots \alpha_{n-1}$ sein, in welche ein Verweis auf die Datenbasis $\alpha_0 \dots \alpha_n$ eingetragen wird.

Wurden einfache Namen angegeben, etwa β_1, \dots, β_r , so müssen Datenbasen ^{oder Dateien} $\alpha_0 \dots \alpha_{n-1} \beta_1, \dots, \alpha_0 \dots \alpha_{n-1} \beta_r$ (für den Eigentümer von $\alpha_0 \dots \alpha_n$) existieren, sie erhalten die Namen $\alpha_0 \dots \alpha_n \beta_1, \dots, \alpha_0 \dots \alpha_n \beta_r$, die Verweise auf sie im $\alpha_0 \dots \alpha_{n-1}$ werden im $\alpha_0 \dots \alpha_n$ umgetragen. Anschließend sind sie nur noch unter ihrem neuen Namen erreichbar.

Eine Datenbasis, auf die aus einer permanenten (temporären) Datenbasis verwiesen wird, ist stets permanent (temporär).

1.1.3 Aufgabe von Datenbasen

1.1.3.1 Syntax

Die Syntax des Aufgabe-Befehls ist ähnlich der des Einrichte-Befehls.

1.1.3.2 Semantik

Der Verweis auf die aufzugebende Datenbasis in der übergeordneten Datenbasis wird gelöscht. Mit einer Datenbasis $\alpha_0 \dots \alpha_n$ werden auch alle Datenbasen, ^{und Dateien} die ihr untergeordnet sind, aufgegeben, mit folgender Ausnahme: Wurden einfache Namen angegeben, etwa β_1, \dots, β_r , so müssen Datenbasen $\alpha_0 \dots \alpha_n \beta_1, \dots, \alpha_0 \dots \alpha_n \beta_r$ vorhanden sein, ihr Name wird in $\alpha_0 \dots \alpha_{n-1} \beta_1, \dots, \alpha_0 \dots \alpha_{n-1} \beta_r$ geändert und ein Verweis auf sie in $\alpha_0 \dots \alpha_{n-1}$ eingetragen. Falls durch die Namensänderungen gleiche Namen auftreten würden, wird die Aufgabe der Datenbasis abgelehnt.

Eine nicht vorhandene Datenbasis kann nicht aufgegeben werden, d.h. ein solcher Versuch endet mit einer Fehlermeldung. Die Aufgabe einer Datenbasis wird abgelehnt, falls über sie auf eine gerade in Bearbeitung befindliche Datenbasis/Datei verwiesen wird.

1.1.4 Für die Einrichtung und Aufgabe von Dateien stehen ähnliche Sprachmittel zur Verfügung.

1.1.5 Die (teilweise) beschriebenen Sprachmittel werden auf Kommando- und Assemblerebene vorhanden sein. Entsprechende Fortranerweiterungen sind von Telefunken zum Teil schon definiert.

1.2 Listen

1.2.1 Listenkopf einer Datenbasis

Die Länge eines Listenkopfs ist veränderlich, er enthält ein Interpretationsmerkmal, das besagt, daß es sich um den Listenkopf einer Datenbasis handelt (wichtig zur Selbstidentifizierung der Behandlung von Fehlerfällen), einen Zähler, in dem festgehalten wird, wieviele Benutzer diese Datenbasis gerade bearbeiten und wieviele Benutzer über diese Datenbasis auf andere Datenbasen/Dateien, die sie gerade bearbeiten, verweisen, einen Verweis auf dem Anfang des Listenrumpfs, die Länge des Listenrumpfs, Koordinierungsinformation für Mehrfachzugriff, Zugriffssicherungsinformation, Lebensdauerspezifikationen, die über die Kennzeichnung temporär bzw. permanent hinausgehen, statistische Meßgrößen, vielleicht das Benutzerkennzeichen des Eigentümers (vgl. jedoch 1.3.1 und 1.1.2.2).

Es ist noch nicht entschieden, ob dem Benutzer gestattet werden soll, eigene Information in einer Datenbasis ablegen zu können (die vom Datenmanagement nicht interpretiert, sondern nur verwaltet wird), etwa in Form von Kennsätzen, wie sie z.B. im Datenhaltungssystem von Telefunken für Dateien optional sind.

1.2.2 Beschreibungsteil einer Datei

Das Äquivalent zum Listenkopf einer Datenbasis heißt bei einer Datei Beschreibungsteil. Er enthält bei demselben Aufbau die gleichen Daten wie der Listenkopf einer Datenbasis, jedoch an Stelle des Verweises auf den Listenrumpf und der Länge desselben einen Verweis auf die Verwaltungsinformation für satzweisen Zugriff auf die Datei.

1.2.3 Listenrumpf einer Datenbasis

Die Listenelemente einer Datenbasis enthalten die Verweise auf die der Datenbasis untergeordneten Datenbasen und Dateien. Für jede untergeordnete Datenbasis oder Datei gibt es genau ein Listenelement, das ihren einfachen Namen bzgl. ihrer übergeordneten Datenbasis, eine Interpretation, ob es sich um eine Datenbasis oder eine Datei handelt, ihr Lebensdauerzeichen (permanent/temporär) und einen Verweis auf ihren Listenkopf bzw. ihren Beschreibungsteil enthält.

Die Frage der Zugriffssicherung ist noch nicht durchdiskutiert, erfolgt sie über ein Paßwort, so wird dieses in den Verweis auf die Datenbasis/Datei aufgenommen, entschließt man sich für ein Modell, in dem die Sicherungsinformation keine festgelegte, konstante Länge hat, so wird das "Schloß" für eine Datenbasis/Datei in ihrem Listenkopf untergebracht. 1)

Die Listenelemente sind nach einfachen Namen sortiert. Neu hinzukommende Elemente werden einsortiert, bei Aufgabe einer Datenbasis/Datei wird das zugehörige Element entfernt. Leere Elemente gibt es nicht, wohl kann der Listenrumpf leer sein. Die Elemente sind formatisiert, das Format ist in allen Datenbasen gleich.

1) Sicherungsinformation in OS/360

Genau ein Paßwort ist optional, bietet ein Benutzer zweimal hintereinander das falsche Paßwort an, wird sein Job abgebrochen.

1.3 Spezielle residente Datenbasen

Es existieren folgende residente Datenbasen, die für das Datenmanagement von Bedeutung sind:

- 1.3.1 (i) die Zentrale Datenbasis (ZDB)
- (ii) das Benutzerverzeichnis (BV)
- (iii) die Benutzerknoten (<bkz>)

Die Zentrale Datenbasis enthält den Verweis auf das Benutzerverzeichnis. (Über das Benutzerverzeichnis hinaus sind z.Zt. keine Datenbasen bekannt, auf die natürlicherweise vor der Zentralen Datenbasis aus verwiesen werden sollte, so daß die Absicht besteht, die Datenbasis ZDB.BV zu fallen zu lassen und ihre Funktion der Zentralen Datenbasis zu übertragen.)

Im Benutzerverzeichnis sind alle Benutzerkennzeichen aufgelistet, die Benutzerknoten sind Datenbasen, mit dem Benutzerkennzeichen als einfachem Namen bzgl. BV. Richtet ein Benutzer eine Datenbasis/Datei mit einem einfachen Namen ein, so wird ein Listenkopf/Beschreibungsteil für diese Datenbasis/Datei aufgebaut und der Verweis auf sie in "seinem" Benutzerknoten (d.h. dem Knoten, der sein Kennzeichen als einfachen Namen in BV hat) eingetragen. Ein Benutzer ist Eigentümer der Datenbasen/Dateien, die in "seinem" Benutzerknoten eingetragen sind. Zur Erinnerung: Eine Datei mit einfachem Namen ist eine temporäre Datei.

1.3.2 Alle Zugriffe auf Datenbasen und Dateien erfolgen mit Hilfe des Datenmanagements. Der von einem Benutzer mit dem Benutzerkennzeichen β spezifizierte Datenbasis/Datei-Name α wird dabei zu ZDB.BV. β . α ergänzt. Greift ein Benutzer auf Datenbasen/Dateien eines anderen Benutzers zu, muß er neben dem Datenbasis/Datei-Namen das Benutzerkennzeichen des Eigentümers angeben.

Das Datenmanagement hat (allein) Zugang zur Zentralen Datenbasis.

2. Die Datei

Auf Hintergrundspeichern (d.h. nicht wortweise adressierbaren Speichern) abgelegte Informationsmengen, die durch Namen identifiziert werden, nennt man Dateien. Dateien sind aufgebaut aus Sätzen. Sätze sind Folgen von Oktaden, Viertel- oder Ganzwörtern. Ihre Zusammenfassung zu Sätzen bestimmt der Benutzer beim Aufbau der Datei. Jeder Satz ist durch einen Satznamen gekennzeichnet. Zwischen den Satznamen einer Datei besteht eine Ordnungsrelation.

2.1 Dateitypen bezügl. des satzweisen Zugriffs

Für die Bearbeitung von Dateien unterscheiden wir gemäß den Zugriffsformen sequentielle Dateien und random Dateien.

Auf sequentielle Dateien kann nur sequentiell zugegriffen werden, d.h. es stehen dem Benutzer folgende Dienste zur Verfügung:

- (i) lies nächsten/vorangehenden Satz
- (ii) schreibe nächsten Satz
- (iii) lösche nächsten/vorangehenden Satz 1)
- iv) positioniere auf n-ten Satz

Der Zugriff auf random Dateien ist sequentiell oder wahlfrei.

Der wahlfreie Zugriff umfaßt folgende Dienste:

- (i) lies Satz mit Namen a
- (ii) schreibe Satz mit Namen a
- (iii) ersetze Satz mit Namen a
- (iv) lösche Satz mit Namen a

1) mit dem gelöschten Satz werden auch alle folgenden Sätze gelöscht.

Als Satznamen gibt es Nummern (natürl. Zahlen), Marken (Gleitpunktzahlen oder alphanumerische Zeichenreihen fester Länge) oder Schlüssel (Zeichenreihen beliebiger Länge).¹⁾ Gemäß diesen Satznamen unterscheiden wir RAN-Dateien (random access über Nummern), RAM-Dateien (random access über Marken) und RAS-Dateien (random access über Schlüssel). Der Name eines Satzes wird beim Schreiben des Satzes vom Benutzer mitgeliefert.

3. Das Gebiet

Gebiete sind virtuelle Adressenräume auf der logischen Platte innerhalb von Behältern (S. 4.)

Die Identifizierung von Gebieten geschieht über behälterrelative Nummern.

Gebiete sind von Gebietsanfang an in Stücke von Bereichsgröße unterteilt.²⁾

1) In der Literatur versteht man unter "Schlüssel" einschränkend eine Zeichenreihe, die Teilfolge der Zeichenreihe ist, die den Satz bildet, so daß eine Datei mit Schlüsseln als Satznamen ein "partionally inverted file" ist.

2) Die Bereichsgröße ist auf 1 K festgelegt.

Motivation:

- i) Bereichsgrößen unter einem K sind indiskutabel, da die Pagenierung des Kernspeichers hardwaremäßig in Seiten von 1 K Größe realisiert ist.
- ii) Bereichsgrößen über 4 K kommen nicht in Frage, wegen a) zu großen Verschnitts innerhalb der Behälter: Bei 1000 Behältern auf der logischen Platte gibt es 2000 K Verschnitt, das sind 6 % der gesamten log. Platte, b) Schwierigkeiten bei der Pufferbehandlung

(Fortsetzung der Anmerkung nächste Seite)

Bei der Bearbeitung eines Gebiets wird eine Bereichstabelle im Adreßraum der DBV angelegt, in der die Hardwareadresse auf der logischen Platte und eventuell die Adresse einer Kopie im Segment-speicher verzeichnet steht.

Unter der Paginierung eines Gebiets verstehen wir im Folgenden die Unterteilung des Gebiets in die Bereiche, die in der Bereichstabelle aufgeführt sind.

Bereiche zerfallen in Bereichsfragmente konstanter Größe.²⁾

Die aktuelle Länge eines Gebiets wächst in Quanten von Bereichsfragmentgrößen und richtet sich nach der Schreibgrenze, indem jeweils auf volle Bereichsfragmente aufgerundet wird.

Jedes Gebietszerfällt bezgl. der Paginierung in zwei Teile, den Gebietshauptteil, der aus allen ganzen Bereichen des Gebiets besteht, und dem Gebietsrest, der die Bereichsfragmente zwischen letzter Bereichsgrenze und Gebietsende umfaßt.

-
- iii) Bei Auswahl unter den Möglichkeiten Bereichsgröße 1 oder 2 K zu wählen, haben wir uns für 1 K entschieden, wegen a) Kompatibilität zur Seitengröße im Kernspeicher (z.B. in bezug auf Speicherschutzfragen), b) vorgegebener Standardgebiete (wie Druckgebiete, Lochkartenlesegebiete), die in 1 K Stücken vergeben und transportiert werden, c) der Übereinstimmung zur Puffergröße von 1 K.

- 2) Die Bereichsfragmentgröße ist auf Achtelseitengröße festgelegt.

Motivation:

- i) Achtelseiten sind auf der Platte die kleinsten adressierbaren Einheiten.
- ii) Achtelseiten können einzeln transportiert werden.

Große Gebiete sind Gebiete, deren Länge Bereichsgröße überschreitet, deren Hauptteil also nicht leer ist.

Kleine Gebiete haben kleinere Länge als Bereiche und bestehen nur aus dem Gebietsrest. Ihr Gebietshauptteil ist leer.

3.1 Motivation

Gebiete linearisieren die im BSM hierarchisch strukturierten Daten des Datenbasissystems. Sie realisieren dynamische Speicheranforderungen unabhängig voneinander. Durch sich am aktuellen Bedarf orientierenden Speicherzuteilung wird ein "innerer Verschnitt" von im Mittel nur einem halben Bereichsfragment pro Gebiet erreicht.

3.2 Inhalt des Gebiets

Auf die Information in Gebieten kann über die Datenbasisverwaltung zugegriffen werden.

3.2.1 Gebiete enthalten zweierlei Information.

- (i) solche, die nur von der DEV, BHV, GV gelesen und geschrieben wird, wie Datenbasisverzeichnisse und evtl. Bereichs- und Gebietslisten, und
- (ii) solche, die der Abwickler eines Benutzers pufferweise selbst bearbeitet oder Operatorläufen zur Bearbeitung zur Verfügung stellt, wie Stellvertreterliste und Sätze einer Datei.

3.3 Dynamisches Verhalten

Gebiete können eingerichtet, aufgegeben, verlängert und verkürzt werden.

3.3.1 Einrichtung eines Gebiets

Es wird eine Nummer vergeben, über die das Gebiet identifizierbar ist, es hat aber noch keine Anfangsadresse und seine Länge ist Null.

3.3.2 Verlängerung und Verkürzung eines Gebiets

Ein Gebiet ist an allen Paginierungsgrenzen des Hauptteils, einschließlich Anfang und Ende, um Stücke von Bereichsgrößen verlänger- und verkürzbar. Am Ende des Gebietsrests ist es nur um Stücke von Bereichsfragmentgröße verlängerbar.

3.3.3 Aufgeben eines Gebiets

Ein gelöschttes Gebiet ist nicht mehr identifizierbar, die Gebietsnummer kann neu vergeben werden.

4. Der Behälter

4.1 Motivation

Ein Behälter ist ein virtueller paginierter (unendlicher) Adressraum, der Ganzwörter adressiert. Die "Seiten" dieses Adressraums heißen virtuelle Bereiche. Zum Behälter gehört eine Abbildung des virtuellen Adressraums in den in Bereichen paginierten Adressraum der "logischen" Platte. Bereiche und virtuelle Bereiche haben gleiche Länge.

Behälter werden von der Behälterverwaltung verwaltet. Ein Behälter gehört genau einem Benutzer, jedoch kann ein Benutzer mehrere Behälter besitzen. Die Behälter werden global durchnummeriert. Im Behälter werden Gebiete abgelegt. In diesen Gebieten stehen die Datenbasen und Dateien des Benutzers, der den Behälter besitzt.

Die Abbildung, die zum Behälter gehört, bildet virtuelle Bereiche auf Bereiche ab und zwar so, daß den ersten n virtuellen Bereichen eineindeutig n Bereiche zugeordnet werden; n ist dynamisch veränderlich und bezeichnet die Anzahl der Bereiche, die von der Bereichsverwaltung dem Behälter aktuell zugeteilt sind.

Unter Behältergröße verstehen wir die Anzahl der in diesen Bereichen enthaltenen Ganzwörter.

Der Sinn des Behälters liegt darin, die Datenbasen und Dateien, die einem Benutzer gehören, speichermäßig als Gesamtheit zusammenzufassen. Insbesondere hat man folgende Ziele im Auge:

4.1.1 Hardwareadressenverzeichnis

Die Hardwareadressen der Bereiche, auf die die virtuellen Bereiche abgebildet werden, werden in der Bereichsliste des Behälters aufgehoben, in der auch der Belegungszustand der Bereiche festgehalten wird.

Wenn die Datenbasen und Dateien eines Benutzers verändert, aufgegeben oder ausgelagert werden, so soll damit die Speicherung der Dateien anderer Benutzer auf der logischen Platte nicht berührt werden. Das wird durch das Behälterkonzept dadurch erreicht, daß alle Hardwareadressen der Bereiche, die einem Benutzer zugewiesen sind, an einem Ort, nämlich der Bereichsliste seines Behälters verzeichnet sind, und andererseits in jedem Bereich Daten nur eines Benutzers abgelegt sind.

4.1.2 Transportgröße

Es wird geschätzt, daß der Platzbedarf eines Benutzers im Mittel 50 K Wörter beträgt. Dies sollte auch die mittlere Behältergröße werden. Es wird mit etwa 1000 Benutzern mit permanenter Datenhaltung gerechnet. Das ergibt einen Platzbedarf von 50 Mill. Wörtern. Da die logische Platte nur etwa 32 Mill. Wörter umfaßt, wovon ein Teil für residente und temporäre Datenbasen und Dateien benötigt wird, muß jeweils ein Teil der permanenten Daten zeitweise auf Magnetband ausgelagert sein.

Die Auslagerung auf Magnetband geschieht nur behälterweise, ebenso das Einfahren.

Man orientiert sich an der Transporteinheit beim paging Kernspeicher - Segmentspeicher, indem man die Gleichung

$$\frac{\text{Größe einer Seite}}{\text{Größe des KSP}} = \frac{\text{Behältergröße}}{\text{Größe der logischen Platte}}$$

betrachtet, also

$$\frac{1 \text{ K}}{256 \text{ K}} = \frac{\text{Behältergröße}}{32000 \text{ K}}$$

Mit einer mittleren Behältergröße von 50 K liegt man also immer noch unter der hieraus ermittelten Größe von etwa 120 K.

4.1.3 Vermeidung von Querverweisen

und Dateien

Beim Benutzer mit nur einem Behälter sind alle seine Datenbasen entweder ausgelagert oder auf der logischen Platte. Dadurch wird vermieden, daß aus einer Datenbasis des Benutzers auf eine Datenbasis oder Datei verwiesen wird, die sich auf einem anderen Speichermedium befindet (2 verschiedene Magnetbänder gelten als verschiedene Medien).

4.2 Auslagerung

4.2.1 Algorithmus

Das Auslagern geschieht nach einer Art "paging" - Algorithmus: Wird auf der logischen Platte Speicherplatz verlangt (dies kann z.B. durch den Wunsch auf Einfahren eines Behälters geschehen), und unterschreitet dabei der noch freie Speicherplatz eine bestimmte Größe, so wird eine Anzahl Behälter ausgewählt und auf Band ausgelagert. Das Auswahlkriterium ist noch nicht festgelegt, desgleichen die Anzahl der auszulagernden Behälter (Soll man etwa genausoviel Speicherplatz freigeben, wie benötigt wird, oder auf Vorrat mehr Platz freigeben?). Mit dem Auslagern kann die Einbettung der Gebiete in den Behälter verändert werden. Einzelheiten sind noch nicht festgelegt.

4.2.2 Plattenkanalbelastung

An die Betrachtung des Behälters als Transportgröße schließt sich die Frage nach der Anzahl der Behälter pro Benutzer an. Wenn man schon nur behälterweise einfährt und auslagert, dann könnte man dafür die Daten des Benutzers auf mehrere (kleinere) Behälter verteilen. Das könnte den Vorteil bringen, daß der Plattenkanal weniger belastet wird, da die einzufahrenden Behälter kleiner sind. Dagegen spricht aber, daß die Behälter eines Benutzers möglicherweise zu verschiedenen Zeiten ausgelagert werden und sich nun auf verschiedenen Magnetbändern befinden.

Werden nun mehrere Behälter eines Benutzers gleichzeitig verlangt, so müssen mehrere Magnetbänder aufgespannt werden, was zu großen Wartezeiten und zu großer Bandgeräthbelastung führt. Die Plattenkanalbelastung bei nur einem Behälter pro Benutzer möge folgende Rechnung veranschaulichen:

Voraussetzung: 20 Konsolbenutzer mit je 20 min. Sitzungsdauer; d.h. im Mittel beginnt ein Benutzer pro Minute eine Sitzung. Bei 25% der Benutzer wird ein Behälter eingefahren und damit gleichzeitig irgend ein anderer Behälter ausgelagert. Die Größe jedes Behälters sei 50 K, seine Bereiche mögen auf 30 Spuren verteilt sein.

Die Zugriffs- und Transportzeit für die Bereiche auf einer Spur beträgt im Mittel etwa 200 ms. Dies ergibt $2 \times 30 \times 200 \text{ ms} = 12 \text{ sec}$ für beide Behälter. Alle 4 min. wird nun im Mittel dieser Vorgang ausgeführt, das ergibt $12 \text{ sec} / 240 \text{ sec} = 5\%$ Plattenkanalbelastung.

4.3 Behältertypen

Es gibt 3 Behältertypen.

4.3.1 Ein permanenter Behälter nimmt Datenbasen und Dateien der permanenten Datenhaltung auf. Ein solcher Behälter wird bei Bedarf auf Magnetband ausgelagert.

4.3.2 Ein temporärer Behälter nimmt temporäre Datenbasen und Dateien auf. Temporäre Behälter werden nicht auf Magnetband ausgelagert.

4.3.3 Der zentrale Behälter enthält die residenten Datenbasen und Dateien. Es gibt nur einen zentralen Behälter. Es ist ständig auf der logischen Platte vorhanden und wird nicht auf Magnetband ausgelagert.

Der zentrale Behälter gehört der Datenbasisverwaltung. Er ist der einzige Behälter, aus dem in einen anderen Behälter verwiesen werden darf.

Jeder Benutzer besitzt nur einen temporären Behälter, er kann mehrere permanente Behälter besitzen.

4.4 Verwaltung von Behältern

Unter Behälter verstehen wir in diesem Kapitel nur permanente und temporäre Behälter.

4.4.1 Behälterliste

Es gibt die Liste der temporären und die der permanenten Behälter. Es ist noch nicht entschieden, ob sie im Adressraum der Behälterverwaltung eingebettet sind, oder ob sie in residenten Dateien abgelegt sind. Das Listenelement enthält die Behälteradresse (s. 4.5.1), und die für die Auslagerung nötige Information.

4.4.2 Die Einrichtung eines Behälters geschieht durch Eintrag des Behälters in die jeweilige Behälterliste. Es wird mindestens ein Bereich für diesen Behälter angefordert.

4.4.3 Die Aufgabe eines Behälters geschieht durch Austrag des Behälters aus der Behälterliste und durch Freigabe aller Bereiche, die ihm zugeteilt waren.

4.4.4 Einrichtung und Aufgabe eines Behälters werden angestoßen durch Aufträge der Datenbasisverwaltung an die Behälterverwaltung. Der Auftrag zur Einrichtung eines Behälters wird in folgenden Fällen erteilt:

- (i) Der Benutzer richtet eine permanente/temporäre Datenbasis ~~oder Datei~~ mit einfachem Namen ein und besitzt im Augenblick keine permanente/temporäre Datenbasis ~~oder Datei~~.
- (ii) Der Benutzer richtet eine permanente Datenbasis mit einfachem Namen ein und besitzt bereits Behälter, die alle größer als 50 K sind, und seine ihm noch verbliebene Speicherberechtigung ist größer als 50 K.

Die Gebiete, die diese Datenbasis oder Datei aufnehmen, werden im neuen Behälter eingebettet.

Weiter vorgesehen sind Teilung, Zusammenfügen und Mischen von Behältern, jedoch ist darüber noch nichts festgelegt.

4.4.5 Verteilung der Gebiete auf die Behälter

Richtet der Benutzer eine Datenbasis oder Datei mit qualifiziertem Namen ein, so werden die Gebiete, die diese Datenbasis oder Datei aufnehmen, in den Behälter eingebettet, in dem sich die übergeordnete Datenbasis befindet. Dadurch werden Querverweise zwischen Benutzerbehältern vermieden.

4.5 Inhalt des Behälters

Der Adressraum eines Behälters wird in folgender Weise ausgenutzt. Es gibt 3 Teile ¹⁾:

- (i) Gebiet Nr. 1 und Nr. 2, die Listen enthalten (s. 4.5.1).
Diese Gebiete bestehen, auch wenn sie größer als ein Bereich ^{im Behälter} sind, nicht aus 2 Teilen, sondern werden hintereinander in die ersten Bereiche des Behälters gelegt. Die Bereichsfragmente bis zur nächsten Bereichsgrenze bleiben frei.

1) Bemerkung zur Einbettung der Gebiete in den Behälter

Die Unterscheidung zwischen großen und kleinen Gebieten könnte vermieden werden, wenn man Gebiete nur als Vielfaches eines

- (ii) Daran schließt sich der Behälterteil, in dem sich die Hauptteile aller übrigen Gebiete befinden. Gebietshauptteile beginnen stets auf Bereichsgrenzen des Behälters.
- (iii) Der letzte Teil enthält kleine Gebiete und Gebietsreste, die möglichst dicht gepackt, auch über Bereichsgrenzen hinweg, gespeichert werden.

Bereichs zuläßt. Rechnet man mit 10 Gebieten pro Benutzer, so hat man dann einen mittleren Verschnitt von $10 \times 1/2 \times$ Bereichsgröße = 5 K pro Benutzer. Dies ist 10% der mittleren Behältergröße. Das Ergebnis wird aber noch ungünstiger, wenn man annimmt, daß die kleinen Gebiete im Mittel kleiner als eine halbe Bereichsgröße sind. Wenn man nun schon kleine Gebiete als Vielfache von Bereichsfragmenten wählt, und sie dann im Behälter möglichst lückenlos hintereinander schreibt, dann liegt es nahe, die großen Gebiete zu zerteilen und die Gebietsreste den kleinen Gebieten gleichzustellen. Man gewinnt dadurch, daß die Gebietshauptteile immer auf Bereichsgrenzen anfangen, was sich bei der dynamischen Größenänderung von Gebieten vorteilhaft bemerkbar macht. Ob die Bereichsfragmentgröße von $1/8$ K zu klein gewählt ist, muß die Praxis zeigen.

4.5.1 Listen des Behälters

Das Gebiet Nr. 1 enthält die Bereichsliste, d.h. die Zuordnung: virtueller Bereich - (realer) Bereich. Jedes Listenelement ist ein Ganzwort. Das i -te Listenelement sieht folgendermaßen aus:

Im 1. Halbwort zeigt Bit $16 + k$, $k = 1, \dots, 8$ an, ob das k -te Bereichsfragment dieses Bereichs durch ein Gebiet belegt ist. Im 2. Halbwort steht die Hardwareadresse des Bereichs, der dem i -ten virtuellen Bereich zugeordnet ist. Dadurch kann die Adresse durch Modifizieren erhalten werden.

Das Gebiet Nr. 2 enthält das Verzeichnis aller Gebiete des Behälters, die Gebietsliste. Das i -te Element der Gebietsliste beschreibt das i -te Gebiet. Ein Element besteht aus 2 Ganzwörtern: Bei großen Gebieten enthält das 1. Ganzwort die Nummer k des virtuellen Bereichs, an dessen Anfang der Gebietshauptteil beginnt und die Anzahl der Bereiche des Gebietshauptteils. Der j -te Bereich des Gebietshauptteils liegt auf dem $(k+j-1)$ -ten virtuellen Bereich des Behälters. Bei kleinen Gebieten ist das 1. Ganzwort gelöscht.

Das 2. Ganzwort enthält die Nummer des virtuellen Bereichs und die Nummer des Bereichsfragments, an dessen Anfang der Gebietsrest oder das kleine Gebiet anfängt, außerdem die Anzahl der Bereichsfragmente des Gebietsrests bzw. kleinen Gebiets. Auch diese Bereichsfragmente liegen im Behälter konsekutiv hintereinander.

Fraglich ist, ob die Bereichsliste und die Gebietsliste in Gebieten abgelegt werden. Fest steht, daß die Listen im Behälter liegen sollen, damit sie bei der Auslagerung des Behälters mit ausgelagert werden. Dies ist möglich und wünschenswert, weil sie nur die innere Struktur des Behälters beschreiben. Bei Unterbringung in Gebieten kann ihre dynamische Speicheranforderung durch die Gebietsverwaltung erledigt werden. Bei einer mittleren Behältergröße von 50 K und bis zu 64 Gebieten genügt für die

Unterbringung von Bereichs- und Gebietsliste je $1/8$ K. Selbst bei extremer Behältergröße werden im allgemeinen Bereichs- und Gebietsliste in einem Bereich Platz haben. Dies ist auch wünschenswert, da beim Zugriff auf ein Gebiet im Behälter beide Listen benötigt werden und ein Bereich durch einen Plattenzugriff in den Kernspeicher transportiert werden kann. Deshalb müßte die Gebietsverwaltung veranlaßt werden, beide Gebiete in einen Bereich des Behälters zu legen, was eine Sonderbehandlung dieser Gebiete bezüglich der Einbettung in den Behälter bedeutet. Die Anfangsadresse dieses Bereichs heißt die Behälteradresse.

4.5.2 Längenänderung von Gebieten

Durch Verlängerung oder Verkürzung eines Gebiets wird im allgemeinen die Einbettung mehrerer Gebiete im Behälter verändert. Ein Gebiet kann nur an einer Bereichsgrenze oder am Ende verlängert oder verkürzt werden. Im folgenden wird die Verlängerung beschrieben.

4.5.2.1 Verlängerung an einer Bereichsgrenze

Zwischen 2 Bereichen des Gebiets - dies seien der i -te und der $(i+1)$ -te virtuelle Bereich dieses Behälters - soll ein Bereich eingefügt werden. Zunächst wird ein Bereich von der Bereichsverwaltung angefordert, die dessen Hardwareadresse liefert. Nun wird in der Bereichsliste die Abbildung:

Inhalt (k -tes Ganzwort) \longrightarrow Inhalt ($(k+1)$ -tes Ganzwort)
 $k = i + 1, \dots, n$ (n = Länge der ursprünglichen Liste)
ausgeführt und die Hardwareadresse des neuen Bereichs in das $(i+1)$ -te Ganzwort geschrieben. Außerdem werden in den Elementen der Gebietsliste alle virtuellen Bereichsnummern, die größer als i sind, um 1 erhöht. Schließlich wird im Gebietslistenelement des verlängerten Gebiets die Anzahl der Bereiche um 1 vergrößert.

4.5.2.2 Verlängerung am Gebietsende

Am Ende eines Gebiets soll ein Vielfaches eines Bereichsfragments angefügt werden.

- (i) Bleibt der Gebietsrest einschließlich der Verlängerung unterhalb Bereichsgröße, so wird geprüft, ob sich hinter dem Gebietsrest für die Verlängerung noch genügend viele freie Bereichsfragmente des Behälters befinden. Ist dies der Fall, so wird das Gebiet um die benötigten Bereichsfragmente verlängert und im Gebietslistenelement die Anzahl der Bereichsfragmente entsprechend erhöht. Ist dies nicht der Fall, so wird der neue Gebietsrest dorthin geschrieben, wo genügend viele freie konsekutive Bereichsfragmente vorhanden sind, wobei möglicherweise ein neuer Bereich angefordert werden muß. Das Gebietslistenelement wird sinngemäß geändert.
- (ii) Überschreitet der Gebietsrest einschließlich der Verlängerung k mal die Bereichsgröße, so wird der Gebietshauptteil um k Bereiche verlängert (s. 4.5.2.1). Der neu entstandene Gebietsrest wird gespeichert gemäß 4.5.2.2 (i). Das Gebietslistenelement wird sinngemäß verändert.

4.5.2.3 Das Verkürzen eines Gebiets geschieht auf analoge Weise. Die bei Gebietsänderung entstehenden Lücken, d.h. Bereiche oder Bereichsfragmente, die keinem Gebiet mehr zugeordnet sind, werden in der Bereichsliste als frei geführt und in ihrer Gesamtheit in der Lückensumme des Behälters erfaßt und können wieder anderweitig belegt werden. Was im Einzelnen mit ihnen geschieht, ist noch nicht festgelegt.

4.6 Schlußbemerkung

Die Begriffe Bereich, Gebiet und Behälter bleiben dem Benutzer verborgen. Die Leistungen der zugehörigen Verwaltungen werden durch die Datenbasisverwaltung in Anspruch genommen.

5. Die Abbildung von Datenbasen auf Gebiete

Für die Abbildung von Datenbasen auf Gebiete werden folgende Alternativen diskutiert.

5.1 Jede Datenbasis wird in genau ein Gebiet gelegt, so daß die dynamische Speicheranforderung für eine Datenbasis durch die Gebietsverwaltung erledigt werden kann.

5.2 Mehrere Datenbasen werden in ein Gebiet gelegt, weil

- (i) angenommen wird, daß die meisten Datenbasen weniger als $1/8$ K Platz beanspruchen,
- (ii) die Datenbasen eines Behälters in möglichst wenigen Bereichen liegen sollen, um die Zahl der Plattenzugriffe bei der Abarbeitung der Verweisstruktur klein zu halten.

6. Die Abbildung von Dateien auf Gebiete

6.1 Motivation

Die Umsetzung von Satznamen in Satzadresse geschieht bei Dateien mit wahlfreiem Zugriff praktischerweise mit Hilfe einer Stellvertreterliste (es sei denn, es sind Hardwareeinrichtungen vorhanden, die diese Umsetzung vornehmen, wie etwa beim Plattenspeicher IBM 2302). Ein Listenelement wird durch einen Satznamen identifiziert, es enthält die Adresse des Satzes mit diesem Namen und (in unserem Entwurf) die Länge des Satzes. Um das Suchen in der Stellvertreterliste zu erleichtern, sind die Listenelemente formatisiert und untereinander nach Satznamen geordnet.

Bei sequentiellen Dateien könnte man, ohne die (Wieder)Auffindung der Sätze zu verlangsamen, mit jedem Satz den (die) Verweis(e) auf den Nachfolger (und den Vorgänger) koppeln. Wir werden jedoch auch bei sequentiellen Dateien die Satzadressen in einer Stellvertreterliste halten, um eine einheitliche innere Verwaltungsorganisation für Dateien zu haben. Ebenfalls aus Gründen der Ein-

heitlichkeit werden wir nur einen Typ von Stellvertreterliste haben, d.h. genauer: für alle Dateien hat das Listenelement gleichen Inhalt und gleiches Format und es gibt jeweils genau einen Algorithmus für die Einordnung, Streichung(und vielleicht sogar für die Auffindung) der Elemente in der Stellvertreterliste. Die Listenelemente nennen wir Stellvertreter oder genauer Satzstellvertreter, die mit ihnen verbundenen Satznamen Marken. Bei der Festlegung des Typs der Stellvertreterliste hat man darauf zu achten, daß der Aufwand für den Stellvertreter gegenüber dem Aufwand für den eigentlichen Satz möglichst klein ist, sowohl was den Platzbedarf wie, bei Zugriff auf Sätze, den Suchaufwand für den Stellvertreter einerseits und den Transportaufwand für den Satz andererseits betrifft. Wie gut diese Forderung erfüllt werden kann, hängt zunächst stark davon ab, wie die vom Benutzer ausgewählten Satznamen normalerweise in dem zur Verfügung stehenden (geordneten) Satznamenvorrat verteilt sind und in welcher Reihenfolge sie ausgewählt werden. Über diese Verteilung, die mindestens vom Vorrat selbst und vom Zweck, für den diese Datei gebraucht wird, abhängt, können wir keine begründeten Annahmen machen, so daß wir uns für einen Typ der Stellvertreterliste entschieden haben, der im uns bekannten allgemeinsten Fall ("inverted files" bei der Betrachtung ausgeschlossen) die gestellte Forderung vielleicht optimal erfüllt. Diese Entscheidung scheint deswegen gerechtfertigt, weil dafür gesorgt werden konnte, daß der Mehraufwand gegenüber einer für den sehr speziellen Fall einer sequentiellen Datei optimalen Lösung gemessen am Gesamtaufwand für die Datei nicht erheblich erscheint. Bei der Definition einer Strategie für die Anordnung der Sätze einer Datei im Speicher ist zu beachten, daß auch Dateien mit wahlfreiem Zugriff häufig sequentiell gelesen werden. Dabei verbietet sich ein eigener Transport für jeden Satz von der Platte in den Kernspeicher der Plattenkanalbelastung wegen ebenso wie das Vorhandensein der gesamten Datei im Kernspeicher des Platzbedarfs wegen. Die Mittelung zwischen diesen beiden Extremen wird wie

üblich durch Puffertechnik vorgenommen. Die Frage, ob diese Mitteilung in brauchbarer Weise, etwa mit Hilfe eines Pufferpools dynamisch gestaltet werden kann und ob dies einen relevanten Ausgleich zwischen Plattenkanalbelastung und Kernspeicherbelastung bewirkt, ist noch nicht beantwortet.

Unabhängig davon besteht die Ansicht, daß die Puffertechnik erst ihre volle Wirksamkeit entfaltet, wenn die physikalische Reihenfolge der Sätze mit ihrer logischen insoweit übereinstimmt, daß in einem Puffer stets ein (vollständiger) Ausschnitt der logischen Satzfolge der Datei vorhanden ist, bis auf Satzbruchstücke am Anfang und Ende des Puffers. Bekanntermaßen kann dies im Falle, daß die Datei durch wahlfreie Anlieferung der Sätze aufgebaut wird, nur erreicht werden, wenn beim Schreiben eines neuen Satzes im Mittel die Hälfte der bereits geschriebenen Sätze an einen anderen Speicherplatz umkopiert wird. Bedient man sich einer "Überlaufspurtechnik", so kann dieses dauernde Kopieren ebenso vermieden werden, wie gewährleistet werden kann, daß sich in zwei Puffern jeweils ein vollständiger Ausschnitt der Satzfolge befindet. Wir haben uns daher (zunächst) entschlossen, für die Ablage der Sätze zwei Gebiete pro Datei vorzusehen, das Hauptinformationsgebiet kurz I-Gebiet und das Reservegebiet kurz R-Gebiet. Im R-Gebiet sind dem I-Gebiet zugeordnete Überlaufbereiche zusammengefaßt. Um die Forderungen und Probleme im Zusammenhang mit dem Einfügen von Sätzen nicht lediglich vom I-Gebiet auf das R-Gebiet zu verschieben, hat das R-Gebiet die Aufgabe, Einordnungen ins I-Gebiet zu sammeln und zu puffern.

Der Inhalt beider Gebiete wird in einer gemeinsamen Stellvertreterliste beschrieben, damit beim Zugriff auf die Sätze nicht zwei Listen durchsucht werden müssen. Die Einordnung eines neuen Elements wirkt bei der Stellvertreterliste nicht die praktischen Probleme auf wie die Einordnung des zugehörigen Satzes in das I-Gebiet, weil man wegen der anderen Größenordnung sowohl das Kopieren vieler Listenelemente vertreten kann wie auch gezieltes Lückenlassen bzw. Erzeugen.

In Diskussion befindet sich ein Modell, in dem die Überlaufbereiche innerhalb des I-Gebiets selbst untergebracht werden, und ein Modell, in dem die Sätze in einem Gebiet in der Reihenfolge ihrer Anlieferung abgelegt werden und die Probleme der sequentiellen Bearbeitung durch Pufferpooltechnik gelöst werden.

6.2 Stellvertreterlisten

Die Stellvertreterliste wird in Teillisten von 128 Ganzwörtern Länge verwaltet. Diese Länge ist so bemessen, daß die Zahl der Hintergrundtransporte beim Zugriff auf das Informationsgebiet im Mittel gleich der Zahl der Hintergrundtransporte beim Zugriff auf das Stellvertretergebiet ist.¹⁾

Beim Aufbau der Datei werden die Stellvertreter lückenlos in die Stellvertreterliste geschrieben. Beim späteren Einfügen eines Stellvertreters in eine volle Teilliste wird diese auf zwei Teillisten aufgeteilt, die nur zur Hälfte gefüllt sind. Zum schnelleren Auffinden eines Satzes führt man gestufte Indizes ein, das bedeutet, daß es Listen von Listen geben wird (s. 6.3), die der Benutzer aber nicht zur Strukturierung seiner Sätze verwenden kann. Die Stellvertreter dieser Listen heißen Listenstellvertreter (im Gegensatz zu den Satzstellvertretern). Die Stellvertreterlisten für die Satzstellvertreter und die Listenstellvertreter sind gleich aufgebaut und bestehen nur aus der Folge der Stellvertreter.

1) Bei einer mittleren Satzlänge von 8-16 Ganzwörtern (Texthaltungsdateien) ist die Stellvertreterliste (Stellvertreterelement = 2 Ganzwörter) für 1 K Ganzwörter Information eine Viertel- bis eine Achtelseite lang, während sie bei längeren Sätzen (Programmdateien, deren Sätze übersetzte Programme sind) kürzer als eine Achtelseite ist.

(i) Satzstellvertreter

3	MARKE	
R	AA	LNG

MARKE = Satzmarke, die den Satz identifiziert

AA = Anfangsadresse des Satzes relativ
zum Anfang des Informationsgebietes

LNG = Satzlänge in Ganzwörtern

R = Überlaufspurmarke

1 bedeutet: Satz in R

0 bedeutet: Satz in I

Ungültige Sätze haben die Länge Null und Überlaufspurmarke R=1

(ii) Listenstellvertreter

3	MARKE	
	AA	EA

MARKE = Marke des ersten Stellvertreters in der Liste,
auf die verwiesen wird

AA = Anfangsadresse der Liste, auf die verwiesen wird

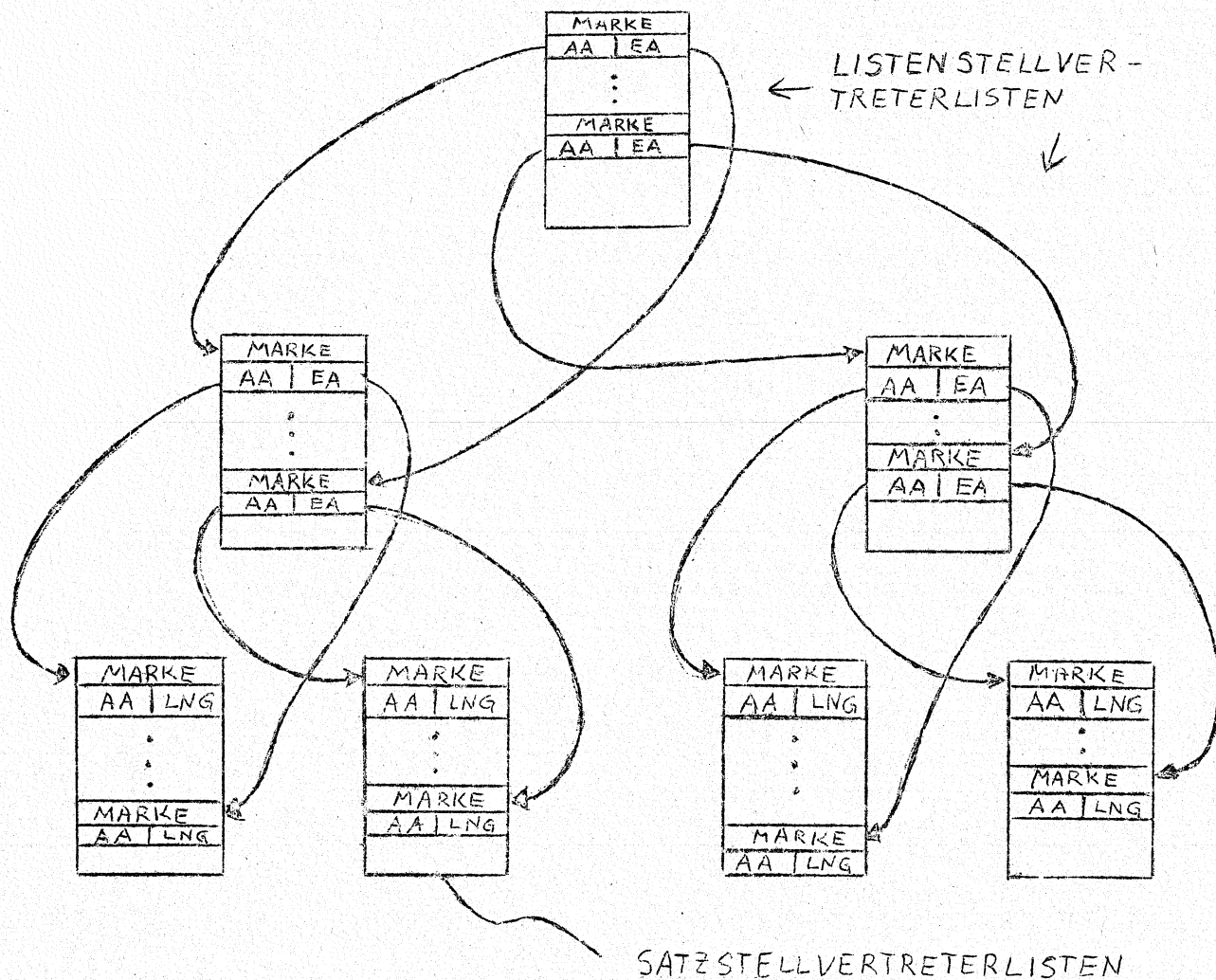
EA = Adresse des letzten, gültigen Stellvertreters in
der Liste, auf die verwiesen wird.

6.3 Indexhierarchie

Die Stellvertreter der Sätze einer Datei werden, nach aufsteigenden Marken geordnet, dicht in die Teillisten geschrieben. Um die Sätze schneller auffinden zu können, wird zu den Satzstellvertreterlisten eine Grob-Stellvertreterliste angelegt, deren Stellvertreter aus der ersten Marke einer Satzstellvertreterliste, der stellvertretergebietsrelativen Anfangsadresse und der Adresse des letzten Stellvertreters in dieser Liste gebildet werden.

Die Grob-Stellvertreterliste wird in Achtelseiten verwaltet. Sobald eine zweite Grob-Stellvertreterteilliste angelegt werden muß, wird auch eine weitere Hierarchiestufe eingerichtet (Grob-Grob-Stellvertreterliste), deren Stellvertreter in analoger Weise aus den ersten Stellvertretern einer jeden Grob-Stellvertreterliste gebildet werden.

Von jeder Hierarchiestufe ist jeweils die Liste im Kernspeicher, die zuletzt benötigt wurde.



6.4 Liste der Überlaufbereiche (LDR)

Ist ein I-Gebiet größer als 5 K, wird es in Gebietsausschnitte I_j ($j = 1, 2, \dots$) eingeteilt. Jedes I_j beginnt mit dem ersten Satz-
anfang nach einer Bereichsgrenze. Die I_j sind disjunkt und über-
decken I vollständig. Die größte Satzmarke in I_j bezeichnen wir
mit m_j , die Anfangsadresse von I_j mit A_j .

Beim Aufbau der Datei werden die Gebietsausschnitte folgendermaßen
festgelegt:

A_1 fällt mit dem Anfang des I-Gebietes zusammen. Sei A_j ($j = 1, 2, \dots$) die Anfangsadresse des letzten Gebietsausschnittes I_j ,
der im n_j -ten Bereich von I beginnt. Wächst I über den Bereich
 n_j+5 ¹⁾ hinaus, so beginnt der Gebietsausschnitt I_{j+1} auf dem
ersten Satzanfang des (n_j+5) -ten Bereiches von I. Jedem Gebiets-
ausschnitt I_j ist ein Überlaufbereich R_j zugeordnet (der leer sein
kann). Aktuell eingerichtet wird R_j bei Bedarf. Die Überlaufbe-
reiche R_j bilden das R-Gebiet. In der Liste der Überlaufbereiche
wird die Zuordnung der Überlaufbereiche zu den Gebietsausschnitten
festgehalten. Das j-te Listenelement hat folgendes Aussehen.

3	m_j		
	SR_j	AA_j	E

m_j = größte Satzmarke in I_j
(wenn diese Marke noch nicht feststeht, wird TK = 0 einge-
tragen)

SR_j = Schreibpegel in R_j

AA_j = Anfangsadresse von R_j relativ zum Anfang des R-Gebiets

E = Einrichtungsbit E = 0 heißt: R_j ist nicht leer
 E = 1 heißt: R_j ist leer

Das j-te Listenelement der LDR wird wie folgt aufgebaut. Sobald
 I_j die erste Bereichsgrenze überschreitet, wird E = 1 gesetzt.
Ist I_j vollständig gefüllt, wird m_j eingetragen. AA_j wird beim

1) Die Anzahl der Bereiche, die ein Gebietsausschnitt umfaßt, wird
möglicherweise parametrisiert sein.

Einrichten von R_j eingetragen, und es wird $E = 0$ gesetzt.

Läuft ein Überlaufbereich über, werden seine Sätze in I_j eingeordnet. Dabei muß normalerweise I_j um einen Bereich erweitert werden (Einschieben eines Bereiches in I). Umfaßt I_j 2 x 5 Bereiche, so wird er (s.6.6.6) in zwei Gebietsausschnitte I_{j1} und I_{j2} geteilt. Gleichzeitig muß in der LDR ein Element eingefügt werden.

6.5 Suchalgorithmen

- (i) Gegeben sei M_R , die Satzmarke eines Satzes, der in R geschrieben werden soll. Gesucht wird der Überlaufbereich, in den dieser Satz gehört.

Mit dem TLD (Dehnung = 2) durchsucht man die LDR nach der ersten Marke M' , die größer oder gleich M_R ist.

Wird M' gefunden, so enthält das zu M' gehörende Listenelement die Anfangsadresse des gesuchten Überlaufbereiches.

Wird kein M' gefunden, so gehört der Satz in den letzten Überlaufbereich von R .

(In beiden Fällen muß der Überlaufbereich erst eingerichtet werden, wenn er leer ist.)

- (ii) Gesucht ist die kleinste Satzmarke in der Stellvertreterliste, die größer ~~oder gleich~~ einer gegebenen Satzmarke M ist.

Mit dem TLOG sucht man in der Stellvertreterliste der höchsten Hierarchie (es gibt genau eine solche Liste) nach der kleinsten Satzmarke M'_0 , die größer ~~oder gleich~~ M ist. In dem M'_0 ^{Vorangeschrieben} ~~enthaltenen~~ Listenelement steht der Verweis auf die Anfangsadresse einer Liste der nächstniedrigen Hierarchiestufe. Hier sucht man wieder mit dem TLOG nach der kleinsten Satzmarke M'_1 , die größer ~~oder gleich~~ M ist, usw. In der untersten Hierarchiestufe (Satzstellvertreterlisten) findet man M'_n , die kleinste Satzmarke, die größer als M ~~oder gleich~~ M ist.

6.6 Ablage der Sätze

Die Sätze einer Datei stehen speichersequentiell, wenn ihre Reihenfolge so ist, daß jeweils alle Sätze, die im i -ten Bereich stehen, kleinere Marken haben als alle Sätze im $(i+1)$ -ten Bereich.

(Die Marke eines Satzes, der über Bereichsgrenze steht, ist also größer als alle Marken der Sätze im Bereich, in dem er anfängt, und kleiner als alle Marken der Sätze im Bereich, in dem er aufhört.)

Entsprechend unterscheiden wir bei einer Datei zwei getrennte Teile von Information. Der I-Teil enthält den ersten angelieferten Satz. Alle Sätze in ihm sind speichersequentiell abgelegt.

6.6.1 Puffer

Alle Puffer für Satzinformation haben die Größe eines Bereichs (1 K). Puffer haben im Informationsgebiet der Datei feste Aufsetzpunkte, nämlich die Paginierungsgrenzen des Gebiets.

Bei Bearbeitungseröffnung einer Datei ohne R-Gebiet wird mindestens ein Puffer, mit R-Gebiet werden mindestens zwei Puffer zur Verfügung gestellt.

Im Folgenden wird die Ablage der Sätze über ein bzw. zwei Puffer beschrieben.

6.6.2 Ablage bei speichersequentieller Anlieferung

Es gibt nur einen Puffer P_I . Jeweils, wenn beim Ablegen eines Satzes die Puffergrenze erreicht wird, wird der Inhalt von P_I als letzter Bereich zu I hinzugefügt und dann am Pufferanfang weitergeschrieben.

Bei der Abmeldung einer Datei von der Bearbeitung wird der Inhalt von P_I ebenfalls an I angehängt.

Bei der nächsten Anmeldung für Schreiben wird von I dieser zuletzt abgespeicherte Teil in P_I zurückgeschrieben.

6.6.3 Ablage bei wahlfreier Anlieferung

Beim ersten nicht speichersequentiell angelieferten Satz wird ein Puffer P_R angelegt. Es wird anhand der LDR festgestellt, in welchen Gebietsausschnitt I_j von I der angelieferte Satz gehört, und P_R mit dem bisherigen Inhalt von R_j vorgefüllt, der angelieferte Satz in P_R abgelegt. Sobald ein Satz in einem anderen Überlaufbereich R_k abgelegt werden soll, wird P_R nach R_j kopiert und mit dem Inhalt von R_k vorgefüllt. War R_j noch leer, so wird R um den Bereich R_j verlängert.

6.6.4 Überlauf in R

Im Unterschied zu I bestehen in R Lücken, die nicht durch Streichen oder Ersetzen von Sätzen entstanden sind, und Sätze können in R nie über Bereichsgrenze hinweg abgelegt werden.

Beim ersten angelieferten Satz, der nicht mehr ganz in R_j abgelegt werden könnte, wird R_j als gefüllt betrachtet und eine Einsortierung in I angestoßen. Der zu R_j gehörige Ausschnitt I_j wird pufferweise neu geschrieben, so daß alle Sätze aus R_j und I_j speichersequentiell in I stehen. I_j wird i.Q. dabei um einen Bereich vergrößert. Kurz: R_j wird in I_j einsortiert.

Man sieht: Jeder einmal aktuell in R eingerichtete Bereich R_j enthält immer mindestens einen Satz.

6.6.5 Pufferpool für R

Es ist geplant, daß der Benutzer mehrere Puffer für R haben kann, während ihm natürlich für I immer nur ein Schreibpuffer (der die Sätze des letzten Bereichs von I enthält) zur Verfügung steht.

Ein Pufferpool scheint notwendig, um Simultanarbeit auf großen Dateien zu erleichtern. Es wird die Frage diskutiert, ob die Dienstleistungen, die mit einem Pufferpool verbunden sind, auch einem einzelnen Benutzer zur Verfügung gestellt werden sollen, um den Plattenkanal zu entlasten.

6.6.6 Überlauf in I

Mögliche Kriterien für die Teilung eines Ausschnitts I_j sind:

- (i) I_j überschreitet eine Maximalgröße (10 K)
- (ii) R_j läuft doppelt so schnell über wie ein anderer Überlaufbereich R_k in R.

Die Teilung kann auf folgende Weisen geschehen:

- (i) I_j zerfällt in zwei Ausschnitte mit möglichst gleich vielen Bereichen.
- (ii) I_j zerfällt in zwei Ausschnitte, in die gleich viele Sätze einzusortieren waren, als die Teilung angestoßen wurde.

Sachverzeichnis

	Seite
Aufgabe eines Behälters	17
Auslagerung eines Behälters	14
Behälter	13
- adresse	21
- gröÙe	13
- liste	17
- verwaltung	13
Bereich	13
Bereichsfragment	10
-fragmentgröÙe	10
-gröÙe	9
-liste	20
-tabelle	10
Beschreibungsteil einer Datei	6
Datei	8
Dateienbaum	1
Datenbasis	1
Eigentümer einer Datenbasis/Datei	3
einfacher Name	3
Einfahren eines Behälters	14
Einrichtung einer Datenbasis/Datei	1
- eines Behälters	17
Gebiet	9
Gebietsausschnitt	29
-hauptteil	10
-liste	20
-rest	10
groÙes Gebiet	11

	Seite
Hauptinformationsgebiet	25
Identifizierung von Gebieten	9
I-Gebiet	25
kleines Gebiet	11
Länge eines Gebiets	10
Liste der Überlaufbereiche	29
Listenstellvertreter	26
Lückensumme des Behälters	22
Marke	24
Mischen von Behältern	18
Name einer Datenbasis/Datei	1
Paginierung eines Gebiets	10
permanente Datenbasis/Datei	1
-r Behälter	16
random - Datei	8
Reservegebiet	25
residente Datenbasis/Datei	1
R-Gebiet	25
Satz	1
- name	8
- stellvertreter	24
Schreibpuffer	32
sequentielle Datei	8
speichersequentiell	31
Stellvertreter	24
- liste	23

	Seite
Teilung von Behältern	18
temporäre Datenbasis/Datei	1
-r Behälter	16
übergeordnete Datenbasis	3
virtueller Bereich	13
wahlfreier Zugriff auf eine Datei	8
zentrale Datenbasis	1
-r Behälter	16
Zusammenfügen von Behältern	18