# Datenhaltung für ein Betriebsleitsystem und dessen Realisierung in PEARL

S. Kühner, Universität Karlsruhe

Kurzfassung

Für ein Betriebsleitsystem für den öffentlichen Personennahverkehr wurde ein Datenhaltungssystem entworfen und in BASIS-PEARL realisiert.
Es ermöglicht die Eingabe der Solldaten des Verkehrsbetriebes über maskengesteuerte Editoren. Die eingegebenen Daten werden auf Konsistenz geprüft und auf Hintergrundspeicher abgelegt. Den Grundsätzen einer mehrschichtigen Datenbank entsprechend, werden Zugriffsfunktionen für den Aufbau lokaler Datensichten für den Prozeß zur Verfügung gestellt.

Schlüsselwörter

Betriebsleitsystem, Datenhaltung, Konsistenzerhaltung, PEARL

Summary

A database system has been designed for a command and control system for public transit and implemented in BASIC-PEARL.
It allows the input of nominal data of the transport operation by editors using a formular driven dialog.
According to the rules of a multilevel database system access functions are established allowing the process to build up private views of data.

## 1. Einleitung

Die gesellschaftlichen Anforderungen an den öffentlichen Personennahverkehr sind in den letzten Jahren enorm gestiegen. Um weitere Kunden für öffentliche Verkehrsmittel gewinnen zu können und die bereits die ÖPNV-Systeme benutzenden Fahrgäste halten zu können, sind die Verkehrsbetriebe bestrebt, insbesondere die Zuverlässigkeit und Bequemlichkeit im öffentlichen Personennahverkehr zu verbessern.
Verspätungen von Fahrzeugen und damit "geplatzte" Anschlüsse sollen ebenso vermieden werden wie beispielsweise die Überfüllung einzelner Fahrzeuge. Die Verkehrsunternehmen sind bestrebt solche Störungen des vorgeplanten Verkehrsablaufes möglichst rasch und umfassend zu erkennen, um geeignete dispositive Maßnahmen zur schnellen Störungsbeseitigung ergreifen zu können.
Nur eine möglichst vollständige Übersicht über das gesamte aktuelle Verkehrsgeschehen in Bezug auf den vorgeplanten Zustand erlauben es einem Verkehrsbetrieb dann auch, über geeignete Einrichtungen die Fahrgäste über Störungen bzw. eingeleitete Ersatzmaßnahmen zu informieren.
Zur Erfassung des aktuellen Verkehrsgeschehens gehört eine exakte Erfassung des aktuellen Aufenthaltsortes sämtlicher sich im Einsatz befindenden Fahrzeuge. Die echtzeitmäßig über verschiedene Meßverfahren gewonnenen Ist-Daten werden über Datenfunk an einen Leitrechner übertragen und ausgewertet. Anhand vorgegebener Sollzustands-Daten kann die Abweichung des aktuellen Verkehrszustandes vom Sollzustand ermittelt werden. Durch tabellarische und graphische Darstellungen wird diese Information in einer Betriebsleitzentrale dargestellt.
Ein Betriebsleitsystem, was diesen Anforderungen genügt (und zudem weitere Funktionen erfüllen soll), wurde im Auftrag des BMFT gemeinsam mit den Hannoverschen Verkehrsbetrieben (ÜSTRA) von mehreren Instituten und Software- und Ingenieurbüros entwickelt.
Bild 1 zeigt die einzelnen Bausteine (Moduln) des Betriebsleitsystems BON

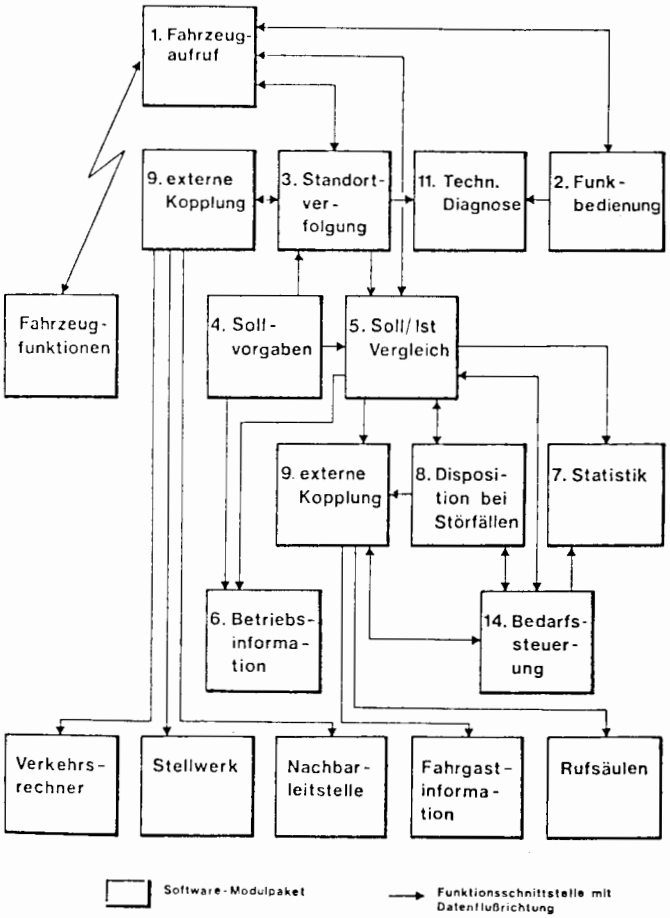


Bild 1 : Modulpaket - Strukturierung von BON
Quelle : [3]

## 2.Sollzustandsdaten eines Verkehrsbetriebes

Der geplante Verkehrsablauf in einem Verkehrsbetrieb wird durch eine große Anzahl von Einzeldaten und Vorschriften, wie diese Einzeldaten zu interpretieren sind und wie sie zueinander in Beziehung stehen, ausgedrückt.
Im einzelnen werden folgende Datenkomplexe benötigt: (vergleiche Bild 2)

- Streckennetz
  Das Streckennetz umfaßt die Beschreibung der physikalisch vorhandenen Schienen- und Straßenstücke, auf welchen sich Fahrzeuge des ÖPNV bewegen können.
  Desweiteren enthält das Streckennetz Informationen über Haltepunkte, Wendeanlagen, Betriebshöfe, Hilfseinrichtungen für die Ortung etc. und deren Lage im Streckennetz.

- Linien- und Fahrwege
  Auf dem physikalisch existierenden Netz legt der Verkehrsbetrieb seine Betriebsform in Form von Verkehrsbeziehungen (Linien) fest. Er bestimmt gewisse Punkte in diesem Netz (Endhaltestellen) sowie Fahrwege zwischen ihnen, über welche er eine regelmäßige Verkehrsbedienung anbieten will.

- Fahrpläne und Anschlüsse
  Die zeitliche Abfolge der Fahrten einer Linie sowie die Abstimmung der Fahrten mehrerer Linien aufeinander wird in den Fahrplänen und Anschlußbeziehungen festgelegt.

- Dienstpläne

- Fahrzeug- und Personalverzeichnisse

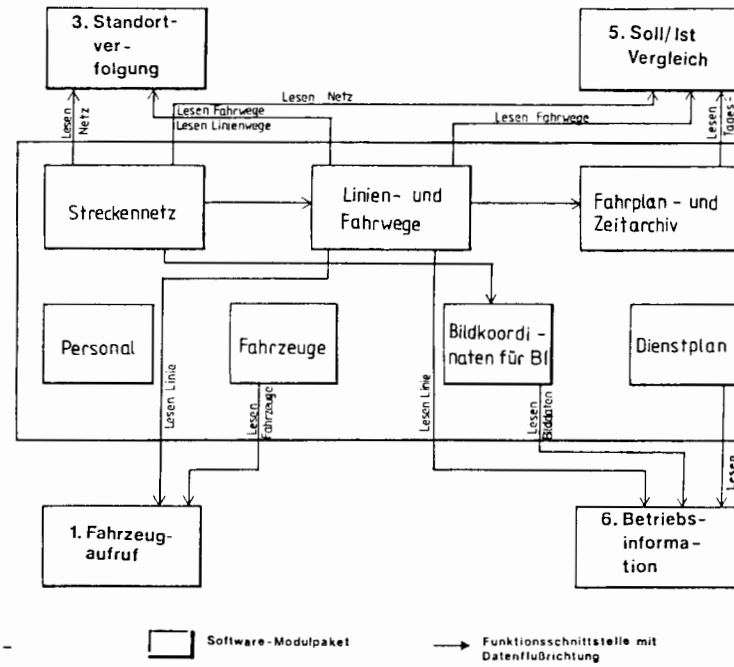


Bild 2 : Einbettung des Modulpaketes Sollvorgab
über Schnittstellen in das Gesamtsyste
Quelle : [4]

## 3.Grundlagen des Datenmodells

Die Gesamtheit dieser Daten sowie die Bedingungen durch welche beschrieben wird, in welcher Beziehung einzelne Datenelemente oder Datengruppen zueinander stehen, ergeben ein Modell eines Verkehrsbetriebes.

Die Festlegung der Objekt [1], die im Verkehrsbetrieb für das Betriebsleitsystem eine Rolle spielen, sowie eine lückenlose Sammlung aller Vorschriften, durch welche im Verkehrsbetrieb die Zusammenhänge zwischen einzelnen Objekten oder Exemplaren von Objekten festlegt sind, ergeben die Grundlage für die Gestaltung der Datenbasis des Betriebsleitsystems. Das so formulierte Modell stellt eine Gesamtsicht des Verkehrsbetriebes dar.
Ohne eine sorgfältige und vollständige Ausformulierung dieses Modells können die Anforderungen an das Datenhaltungssystem in keinem Punkt vollständig spezifiziert werden. Dieses abstrakte Modell ist auch Grundlage für die Konzipierung der Speicherstrukturen in der Datenhaltung.
Entweder muß dann das obige Modell mit den Mitteln eines existierenden Datenbanksystems und der dort angebotenen Datendefinitionssprache formuliert werden oder es sind, wie hier geschehen, Konzepte zu entwickeln, wie Objekte und Beziehungen möglichst einheitlich in Programm- und Speicherstrukturen umgesetzt werden können.
Einen Ausschnitt aus der Modellbeschreibung für BON zeigt Bild 3.

Die Ovale repräsentieren hier Objekte, die

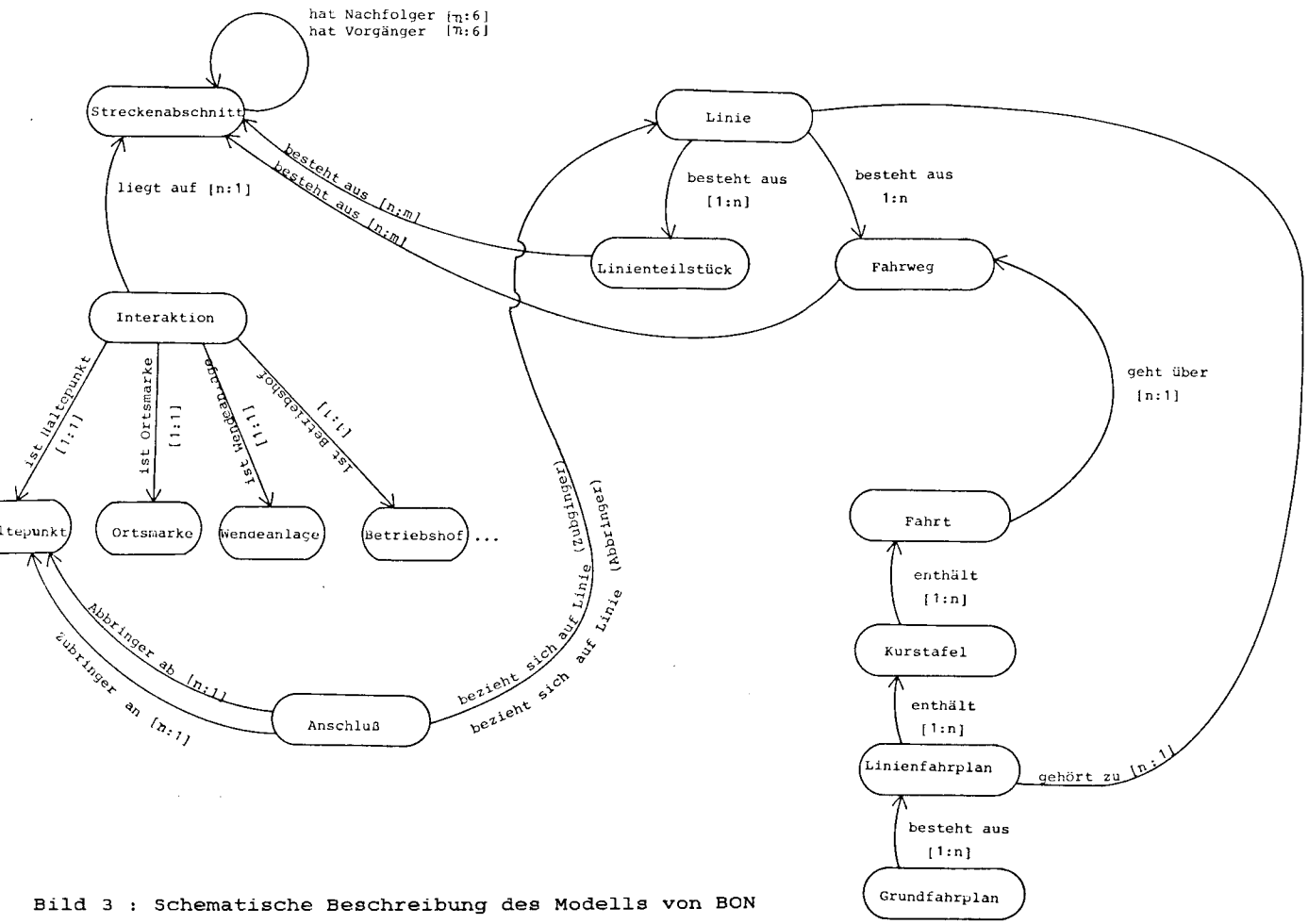


Bild 3 : Schematische Beschreibung des Modells von BON

Pfeile bezeichnen die Beziehungen zwischen den Objekten bzw. Exemplaren von Objekten.
Diese Abbildung zeigt (durch die Pfeile) nur einen Teil der Beziehungen zwischen den Objekten. Weitere Konsistenzbedingungen müssen durch andere formale Hilfsmittel (z.B. prädikatenlogische Ausdrücke) formuliert werden.

Einzelne Abteilungen des Verkehrsbetriebes und einzelne Moduln des Betriebsleitsystemes brauchen in der Regel ausschließlich einen Teil der Daten des Gesamtsystems. Sie benötigen außerdem die Daten in unterschied-

1)
Unter OBJEKT verstehe ich hier einen Repräsentanten für eine Menge gleichartiger Gegenstände (z.B. Haltepunkte). Objekte werden durch Attribute (z.B. Haltestellennummer) beschrieben.
Ein einzelnes konkretes Element aus einer solchen Menge bezeichne ich als EXEMPLAR (z.B. der spezielle Haltepunkt mit der Nummer 123).

licher Zusammenstellung: d.h. sie besitzen ein eigenes Teilmodell des Gesamtmodells.
Diese Anforderung ist keineswegs charakteristisch für Verkehrsbetriebe sondern typisch für sehr viele Anwendungsbereiche von Datenbank- oder Datenhaltungssystemen. Deshalb bietet sich eine mehrschichtige Vorgehensweise an. Solche Vorgehensweisen werden in der Literatur als mehrschichtigen Datenbanken bezeichnet. [1]
Welche Konsequenzen sich daraus für die Schnittstellen zwischen Betreiber und Datenhaltungssystem einerseits sowie Datenhaltungssystem und Anwenderprogrammoduln andererseits ergeben, wird in den nachfolgenden Abschnitten noch eingehender behandelt.

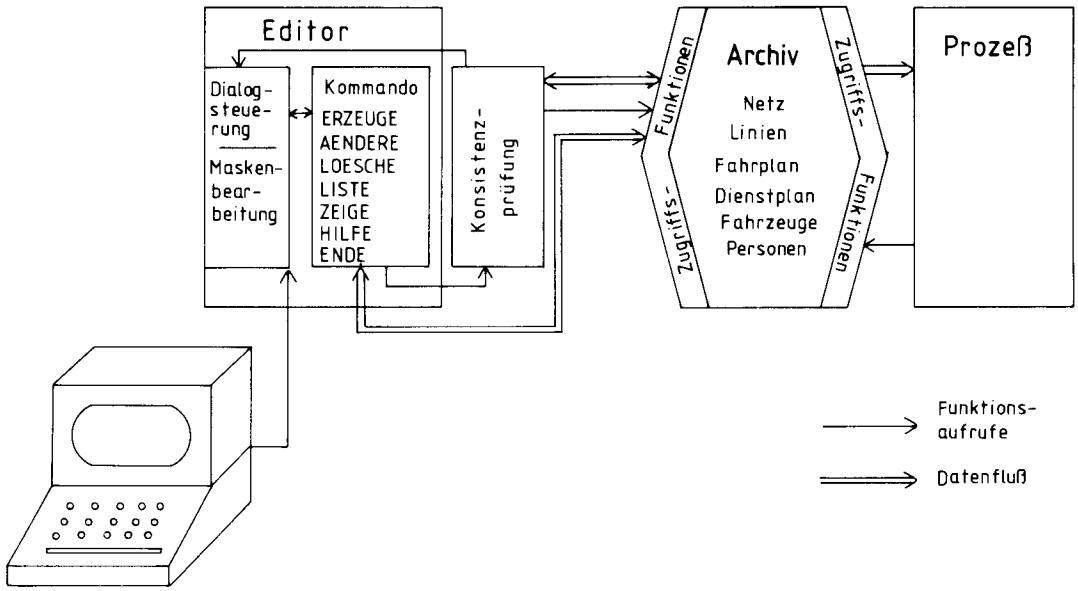


Bild 4 : Aufgaben der Datenhaltung

## 4.Aufgaben des Datenhaltungssystems

Sämtliche Daten des Verkehrsbetriebes weisen eine hohe Dynamik auf. Dies gilt auch für alle Daten, durch welche der Sollzustand beschrieben wird. So werden Fahrpläne zumindest zweimal im Jahr geändert. Erweiterungen und Änderungen des Streckennetzes oder der Linien treten, beispielsweise auf Grund baulicher Maßnahmen, ebenfalls häufig auf.
Daraus erwächst die Notwendigkeit, den verschiedenen Planungsabteilungen des Verkehrsbetriebes ein einfach zu bedienendes aber effektives Instrument zur Pflege der Soll-Daten in die Hand zu geben.
Die mittels der Editoren in das Rechnersystem eingegebenen Daten müssen auf Konsistenz geprüft und auf Hintergrundspeicher abgelegt werden. Schließlich sind für die einzelnen Moduln des Betriebsleitsystems, Zugriffspfade auf die Daten zur Verfügung zu stellen.

### 4.1 Editoren

Für jeden der in Bild 2 gezeigten Datenbereiche wurde ein Editor entwickelt, der es erlaubt, Daten in die Datenbasis einzufügen, sie zu ändern oder zu löschen. Für die Editoren wurden einheitliche Bedienabläufe und Kommandosätze festgelegt. Zu Manipulation und Überprüfung des Inhalts der Datenbasis wurden für jedes Objekt, durch welches der Verkehrsbetrieb beschrieben wird, die Kommandos ERZEUGE, AENDERE, LOESCHE, ZEIGE, LISTE vorgesehen. Weitere Kommandos sind zur Steuerung der Arbeitsweise des Editors vorgesehen. (HILFE, SCHLUSS, ANFANG).
Die Dateneingabe erfolgt interaktiv über Eingabemasken.
Dafür wurde ein eigenständiges Hilfsmittel, eine "allgemeine Dialogschnittstelle" entwickelt und implementiert. [2]
Ein Editor erlaubt dabei nur die Manipulation der Daten des entsprechenden Datenbereiches. Die durch einen Editor, d.h. über seine Mensch-Maschine-Schnittstelle, definierte Datensicht kann als externes Teilmodell der Gesamtsicht der Daten bezeichnet werden.
Bei der Spezifikation der Mensch-Maschine-Schnittstelle wurde nach dem Grundsatz vorgegangen, daß der Betreiber keine redundanten Daten einzugeben hat. Das Datenhaltungssystem selbst beschafft sich im Bedarfsfall alle Daten aus anderen Datenbereichen, um die betreiberorientierte Datensicht eines Editors zu erzeugen. Das bedeutet, daß dem Betreiber in den o. g. Eingabemasken auch solche Einzeldaten angezeigt werden, die sich aus dem Gesamt-

modell ergeben und für das Teilmodell relevant sind.

### 4.2 Konsistenzerhaltung

Für die korrekte Funktion des Betriebsleitsystemes ist eine konsistente Datenbasis unerläßlich. Mehrdeutigkeiten oder Widersprüche in den Daten, auf welche die einzelnen Moduln des Betriebsleitsystemes zugreifen, führen zu unkontrollierbarem Verhalten des Gesamtsystemes.
Dem Datenhaltungssystem wurde deshalb die Aufgabe zugeordnet, die Konsistenzhaltung der Datenbasis sicherzustellen.
Dazu wird zwischen vier Stufen von Konsistenzerhaltungsmaßnahmen unterschieden :

- 1.Stufe (Datentypkonsistenz)
  Bei jedem Attribut eines Exemplars eines Objekts wird geprüft, ob es vom vorgegebenen Typ ist und im vorgegebenen Wertebereich liegt. Konsistenzprüfungen dieser Stufe erfordern lediglich die Kenntnis über einen einzigen Wert (- den aktuellen Wert des betreffenden Attributes).

- 2. Stufe (Objektspezifische Konsistenz)
  Zwischen zwei oder mehreren Attributen eines Exemplars eines Objekts können Bedingungen bestehen, welche deren Abhängigkeit beschreiben. Zur Überprüfung solcher Konsistenzbedingungen genügen die Daten eines Exemplares eines Objekts (repräsentiert beispielsweise durch eine strukturierte PEARL-Variable).

- 3. Stufe (lokale Konsistenz eines Archives)
  Zwischen zwei oder mehreren Attributen verschiedener Exemplare desselben oder unterschiedlicher Objekte bestehen weiter einzuhaltende Abhängigkeiten. Einfache Beispiele für solche Konsistenzbedingungen sind :
  + Eindeutigkeit des Schlüssels
  + Verweise auf andere Objekte (bzw. Exemplare)

  Zur Sicherstellung solcher Konsistenzbedingungen müssen mehrere Exemplare von Objekten miteinander verglichen werden.

- 4.Stufe (globale Konsistenz zwischen Archiven)
  Die starke Vernetzung der Daten des Betriebsleitsystemes bewirkt, daß Abhängigkeiten zwischen verschiedenen Bereichen der Sollvorgaben bestehen, z.B. zwischen Streckennetz- und Linienbeschreibungen oder Linienbeschreibungen und Fahrplänen und umgekehrt. Die vollständige Einhaltung aller derartiger Konsistenzbedingungen erfordert einen großen Aufwand insbesondere dann, wenn gegenseitige Abhängigkeiten bestehen. Hier können nicht mehr allein Prüfmechanismen über direkte Verweisketten erfolgen, sondern es müssen Hilfsmittel wie z.B. Referenzzähler, herangezogen werden.

Konsistenztests der beiden ersten Stufen erfolgen bei der Manipulation jedes Exemplars eines Objekts durch den Editor.
Die Konsistenztests der 3. und 4. Stufe werden abhängig von Effizienzgründen an unterschiedlichen Stellen durchgeführt. Bei jedem Sitzungsende wird eine umfassende Konsistenzprüfung der 3. und 4. Stufe durchgeführt.
Alle Konsistenzverletzungen werden vom Editor durch eine Fehlermeldung angezeigt. Die Dialogabläufe sind so ausgelegt, daß der Betreiber erst nach Beseitigung der Konsistenzverletzung weitere Datenmanipulationen mit dem Editor vornehmen kann.
Dadurch soll sichergestellt werden, daß nur solche Daten archiviert werden, welche als konsistent erkannt worden sind.

### 4.3 Datenverwaltung

Die Erstellung des Modells des Verkehrsbetriebes beinhaltet wie oben ausgeführt, die Bestimmung der vorhandenen Objekte und die Beziehungen zwischen Objekten.
Das für BON realisierte Datenhaltungssystem geht davon aus, daß in der Regel für jedes Objekt eine Datei angelegt wird, auf welcher sämtliche Exemplare des Objekts abgelegt werden. Nur im Falle des Fahrplanes werden hierarchisch abhängige Objekte in einer Datei zusammengefaßt, wenn keine Erfordernis besteht, auf das abhängige Objekt via Editor direkt zugreifen zu müssen.
Zur Freispeicherorganisation für die Objektdateien wurde ein einheitliches Verfahren entwickelt und angewendet.
Aus Effizienzgründen wurden Beziehungen zwischen den Exemplaren von Objekten durch direkte Verweise auf die das Exemplar enthaltende Sätze von PEARL-Dations realisiert. Dies erfordert, daß Sätze innerhalb einer Datei nicht verschoben werden dürfen.
Um den Arbeitsspeicherbedarf gering zu halten, werden alle Daten eines Exemplares eines Objekts sofort nach jeder Manipulation auf den Hintergrundspeicher (Plattenspeicher) ausgelagert. Um durch Such- und Sortiervorgänge (z.B. bei Konsistenzprüfungen) verursachte lange Antwortzeiten beim Editieren und beim Zugriff von Anwendermoduln auf die Daten gering zu halten, sind schnelle Zugriffslisten (Indexlisten) vorgesehen worden, die permanent im Arbeitsspeicher gehalten werden können.

### 3.4 Zugriffsfunktionen auf die Datenbasis

Eines der gravierendsten Probleme im Zusammenhang der Datenhaltung in Echtzeitsystemen besteht darin, daß allen Moduln des Prozeßrechnersystems schneller Zugriff auf die Daten zu gewähren ist. Da in der Regel äußerst unterschiedliche Anforderungen von Moduln an die Datenbasis existieren, ergeben sich bei der Strukturierung der Daten und die Ausbildung der Datenpfade Schwierigkeiten. Können Daten zudem vom Prozeß selbst oder durch Betreibereingriff geändert werden, muß durch aufwendige Synchronisationsmaßnahmen sichergestellt werden, daß Zugriffskonflikte beim Lesen und Schreiben der Daten vermieden werden.
Derartige Probleme wurden durch die Verwendung der Grundidee einer mehrschichtigen Datenbankarchitektur und eine starke Abgrenzung der Aufgabenzuweisung an die Datenhaltung weitgehend gelöst.
Einzelne Moduln des Betriebsleitsystems bauen für ihre verschiedenen Aufgaben eigene Datensichten auf, welche gemäß der Aufgabenstellung optimiert werden. (siehe Bild 5)
Dazu lesen sie zu wenigen Zeitpunkten (z.B. einmal pro Tag) die für sie relevanten Daten aus der Datenbasis aus.
Dazu werden wenige einheitliche Schnittstellen vom Datenhaltungssystem zur Verfügung gestellt, die erlauben, die Exemplare eines Objekts sequentiell zu lesen. Im Bedarfsfall werden vom Datenhaltungssystem Dienstleistungen, wie z.B. die Sortierung der Exemplare nach einem oder unterschiedlichen Schlüsselattributen, erbracht.

Die Moduln des Betriebsleitsystems haben lediglich das Recht zu lesen. Änderungen an den Daten sind nur durch die Editoren erlaubt. Diese Änderungen werden auf einer Kopie der aktuellen Laufzeitdatenbasis im "single-user-modus" durchgeführt. Nach Abschluss der Änderung der Daten durch den Editor kann eine neu erzeugte konsistente Datenbasis das alte Laufzeitarchiv (d.h. die alte Datenbasis) überschreiben.
Ergebnisdaten des Prozesses, die vom Rechner oder dem Verkehrsbetrieb weiterverarbeitet werden sollen und dazu langfristig zu speichern sind, werden in eigenen Datenbeständen abgelegt (z.B. Statistik - Dateien).

## 5. Erfahrungen mit BASIS-PEARL bei der Implementierung des Datenhaltungssystems

Eine erste Version eines solchen Datenhaltungssystems wurde für das Projekt BON entwickelt.
Für die Implementierung stand ein EPR 1400 System der Firma KRUPP ATLAS zur Verfügung.
Auf Anforderung der Programmentwickler war der Sprachumfang von BASIS-PEARL um einige features von FULL-PEARL erweitert worden.

### 5.1 Erweiterungen von BASIS-PEARL

1. TYP-Deklarationen zur Definition von Strukturen und zur Re-Definition einfacher Datentypen.

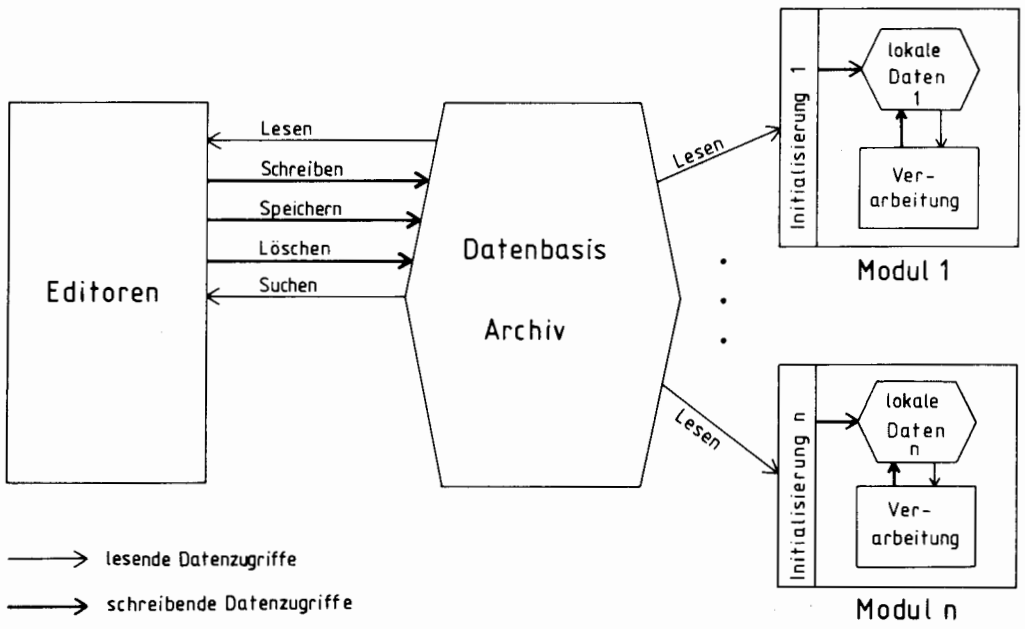


Bild 5 : Zugriffsfunktionen in der BON - Datenhaltung

2. BIT-Slice-Zugriffe aus BIT-Strings
3. Für interne Dations wird für die Datenübertragung der «trf-item-typ» ALL implementiert. Damit können unterschiedliche Strukturen auf eine Dation geschrieben werden.
4. Es wurde eine BOLT-Synchronisation eingeführt.

## 5.2 Erfahrungen bei der Auslagerung von Daten auf Hintergrundspeicher

Für die Grundanforderungen an die Datenauslagerung auf Dateien (DATIONs) und deren Wiederauffindung reichen die BASIS-PEARL Eigenschaften und die in 5.1 genannten Erweiterungen aus. Insbesondere die Fähigkeit der direkten Positionierung bei BASIC-DATIONs wurde intensiv genutzt. Sie stellt ein Standbein der gewählten Implementierungsform dar. Die Verwendung des «trf-item-typ» ALL begünstigte die Freispeicherverwaltung der Objektdateien, die auf dem DATIONs selbst organisiert wurde.
Weitergehende Fähigkeiten des Dateiverwaltungssystemes wie z.B. schlüsselorientierte Dateizugriffe, indizierte Dateiorganisation oder Unterstützung bei der Synchronisation paralleler Dateizugriffe hätten sicherlich elegantere Formen der Datenhaltung erlaubt, hätten aber die Kompatibilität des Systems stark beeinträchtigt.

## 5.3 Daten- und Ablaufstrukturen in BASIS-PEARL

Neben den E/A-Eigenschaften von PEARL wurden für die Datenhaltung in erster Linie die Möglichkeiten zur Daten- und Ablaufstrukturierung benötigt.
Hier mußten einige Schwächen verzeichnet werden, welche Ersatz- bzw. Hilfskonstruktionen notwendig machten.
Die in Abschnitt 5.1 genannten Erweiterungen von BASIS-PEARL stellen eine für die Datenstrukturierung wesentliche Verbesserung dar, die aber für die komplexen Datenstrukturen des Modells eines Verkehrsbetriebes noch unzureichend sind : Es besteht weiterhin keine Möglichkeit, den Wertebereich eines Datentyps selbst zu definieren oder den Wertebereich von Grunddatentypen einzuschränken. Es existiert ebenfalls keine Möglichkeit, Felder als Strukturelemente zu verwenden. Die aus diesem Mangel erwachsenden Schwierigkeiten werden an Beispielen erläutert.

### 5.3.1 Wertebereichsdefinition für Datentypen

Das Datenmodell des Verkehrsbetriebes (vergleiche Bild 3) enthält Objekte unter dem Sammelbegriff "Interaktionen". Dazu gehören die Objekte :

| Objekt | Objektidentifikator | |
|---|---|---|
| Haltepunkt | HPKT | 1 |
| Betriebshof | BHF | 2 |
| Anforderungspunkt für Lichtsignale | LSA | 3 |
| Wendeanlage | WA | 4 |
| Ortscodesender | ORMA | 5 |

Die Objekte werden in PEARL durch strukturierte Datentypen beschrieben. Dabei sind objektspezifische Attribute vorhanden, sowie Attribute, die bei jedem der obigen Objekte existieren. Ein solches Attribut ist die "Interaktionsart" (siehe Objektidentifikator).
Abhängig von der "Interaktionsart" müssen im Datenhaltungssystem gewisse Funktionen ausgeführt werden.
Zur Kennzeichnung der Interaktionsart wäre eine Wertebereichsdefinition wünschenswert gewesen :

```
TYPE INTERAKTIONSART
     ONEOF <HPKT, BHF, LSA, WA, ORMA>;
```

In PEARL mußte eine Ersatzlösung vorgenommen werden :

```
TYPE INTERAKTIONSART  FIXED;

DCL HPKT INTERAKTIONSART     INIT (1);
DCL BHF  INTERAKTIONSART     INIT (2);
DCL LSA  INTERAKTIONSART     INIT (3);
DCL WA   INTERAKTIONSART     INIT (4);
DCL ORMA INTERAKTIONSART     INIT (5);
```

In diesem Zusammenhang kommt noch eine Schwäche bei den PEARL-Eigenschaften in Bezug auf Fallunterscheidungen zum Tragen

Die CASE-Anweisung erlaubt nicht, mit beliebigen Werten in den CASE-Kopf einzutreten. Es sind lediglich Ausdrücke erlaubt, welche einen FIXED-Wert («int-7-exp») als Ergebnis liefern.
Im CASE-Körper sind zu Kennzeichnung der Fälle (Alternativen) ausschließlich Ganzzahlwerte in ununterbrochener Reihenfolge, beginnend mit dem Wert 1, sowie eine "Fluchtalternative" (OUT) zulässig.
Alternativen im CASE-Körper dürfen weder benamt werden noch können mehrere Alternativen zusammengefaßt werden.

Eine Fallunterscheidung, bei der Lichtsignalanlagen gar nicht, Betriebshöfe und

Wendeanlagen gleich und Haltepunkte sowie Ortsmarken getrennt zu behandeln sind, erfordert in PEARL die nachfolgende fehleranfällige und wartungsunfreundliche Konstruktion :

```
CASE   INTERAKTION

ALT  /* HALTEPUNKT */
     ANWEISUNGSFOLGE 1

ALT  /* BETRIEBSHOF */
     ANWEISUNGSFOLGE 2

ALT  /* LICHTSIGNALANLAGE */
     LEERE ANWEISUNGSFOLGE

ALT  /* WENDEANLAGE */
     ANWEISUNGSFOLGE 2

ALT  /* ORTSCODESENDER */
     ANWEISUNGSFOLGE 3

OUT

FIN;
```

5.3.2 Felder als Strukturelemente

Das Objekt "Streckenabschnitt" wird u.a. durch die Attribute
1-6 Nachfolger
1-6 Vorgänger
8 verschiedene Fahrzeiten
beschrieben.
Da Felder nicht als Strukturelemente verwendet werden dürfen, standen 2 PEARL-Ersatzlösungen zur Auswahl.

Lösung 1 : Wiederholen von Strukturelementen

```
TYPE STRECKENABSCHNITT  STRUCT
  [ STRANUMMER      FIXED,
       :
       :
    NF1             FIXED,
    NF2             FIXED,
    NF3             FIXED,
    NF4             FIXED,
    NF5             FIXED,
    NF6             FIXED,
    VG1             FIXED,
     :
    VG6             FIXED ];
```

Diese Struktur kann mit einem WRITE-Statement auf eine Dation geschrieben werden. Für die effektive Bearbeitung von Nachfolgern und Vorgängern müssen diese jedoch in lokale Felder umkopiert werden.

2. Lösung : Aufspaltung in Teilstrukturen

Das Objekt "Streckenabschnitt" wird durch 3 Strukturen beschrieben :

```
TYPE STRECKENABSCHNITT   STRUCT
   [  STRANUMMER   FIXED,
         :
         :
   ];

TYPE VORGAENGER          (6) FIXED;
TYPE NACHFOLGER          (6) FIXED;
```

wobei um die Gesamtstruktur auf eine Dation schreiben zu können, 3 WRITE-Statements auszuführen sind. Dabei muß beim Schreiben vom Programmentwickler besondere Sorgfalt auf die Positionierung gelegt werden.

Beide Ansätze sind offensichtlich fehleranfällig und beeinträchtigen eine einfache Programmwartung.
Gute Möglichkeiten für die Datenstrukturierung erscheinen mir eine der wichtigsten Anforderungen an eine moderne Programmiersprache zu sein.
Geringe Fähigkeiten bei der Datenstrukturierung ziehen zudem häufig Mängel bei den Ablaufstrukturen nach sich.
Einige hier genannten Kritikpunkte beziehen sich naturgemäß ausschließlich auf BASIS-PEARL und würden bei der Verwendung von FULL-PEARL entfallen.

Literatur

1. Lockemann,P.C. ; Mayr,H.C.
   Rechnergestützte Informationssysteme
   Berlin, Heidelberg, New York 1978

2. Stöhr,G.
   Eine Allgemeine Dialogschnittstelle
   in PEARL für die Kommunikation mit
   PEARL-Prozessen über Bildschirm und
   Tastatur.
   in PEARL-Rundschau Bd. 3 Nr. 5  1982

3. Budig,M; et. al.
   Forschungsvorhaben BON
   Systemkonzept Grobentwurf  (AP 2700)
   Interner Projektbericht (August 1980)

4. Kühner,S.
   Forschungsvorhaben BON
   Streckennetz - Sollvorgaben (AP 3300)
   Interner Projektbericht (Mai 1982)

Anschrift des Autor

Stefan Kühner
Institut für Informatik III
Rechnergestützte Automatisierungssysteme
Universität Karlsruhe
7500 Karlsruhe