

	Informationsverarbeitung Programmiersprache PEARL Mehrrechner-PEARL	DIN 66 253 Teil 3
--	---	--

Information processing; programming language PEARL; PEARL for distributed systems

Inhalt			
	Seite		Seite
1 Anwendungsbereich und Zweck	2	7.2.1.1.1 TRANSMIT-Anweisung	25
2 Allgemeines zu Mehrrechner-PEARL	3	7.2.1.1.2 RECEIVE-Anweisung	25
3 Umfang der Norm	4	7.2.1.2 Das Blocking-Send-Protokoll	26
4 Bezeichnung	4	7.2.1.2.1 TRANSMIT-Anweisung	26
5 Begriffe	5	7.2.1.2.2 RECEIVE-Anweisung	26
6 Beschreibungsteil für Mehrrechnersysteme (ARCHITECTURE description)	6	7.2.1.3 Das Send-Reply-Protokoll	26
6.1 STATION-Teil: Beschreibung von Stationen	6	7.2.1.3.1 TRANSMIT-Anweisung	26
6.2 NET-Teil: Beschreibung der physikalischen Verbindungswege	8	7.2.1.3.2 RECEIVE-Anweisung	27
6.3 SYSTEM-Teil: Beschreibung des Zugangs zur Peripherie	9	7.2.2 Protokolle mit mehreren Empfängern	27
6.3.1 Zuordnung zur Station	9	7.2.2.1 Das No-Wait-Send-Protokoll für 1-- > n-Strukturen	27
6.3.2 Stationsübergreifender Gerätezugriff	10	7.2.2.2 Das Blocking-Send-Protokoll für 1-- > n-Strukturen	28
6.4 CONFIGURATION-Teil: Beschreibung der Software-Konfiguration	11	7.2.3 Protokolle mit mehreren Sendern	29
6.4.1 COLLECTIONS: Konfigurierbare Einheiten	12	7.2.3.1 Protokolle für n-- > 1-Strukturen (Server-Schnittstelle)	29
6.4.2 Beschreibung logischer Verbindungen	13	7.2.3.2 Alternatives Warten mit selektiver Fortsetzung	31
6.4.3 Beschreibung des initialen Ladens und der Rekonfiguration	15	7.2.4 Timeout-Mechanismen	32
7 Botschaftenaustausch	20	8 Abstufung der Sprache	34
7.1 Portspezifikation	20	Anhang A Vordefinierte Attribute	35
7.2 Botschaftsmechanismen	24	Anhang B Vordefinierte Signale	36
7.2.1 Einfache Kommunikationsprotokolle (1-- > 1) ..	25	Anhang C Vordefinierte Prozeduren	37
7.2.1.1 Das No-Wait-Send-Protokoll	25	Anhang D Syntax-Zusammenfassung	38
		Anhang E Syntax des Systemteils	41
		Stichwortverzeichnis	42

Fortsetzung Seite 2 bis 44

Normenausschuß Informationsverarbeitungssysteme (NI) im DIN Deutsches Institut für Normung e.V.

1 Anwendungsbereich und Zweck

Die Norm dient dazu, die Programmiersprache PEARL nach DIN 66253 Teil 2 um Sprachkonstrukte zu erweitern, die für die Programmierung von Mehrrechnersystemen geeignet sind. Für Programme, die in dieser Sprache abgefaßt sind, soll ein weltweiter Austausch zwischen Datenverarbeitungsanlagen ermöglicht werden.

Die Norm stellt damit die Grundlage für das Herstellen von Übersetzern und das Ausarbeiten von Anleitungen (Handbücher, Lehrbücher) zur Programmierung in Mehrrechner-PEARL dar.

2 Allgemeines zu Mehrrechner-PEARL

Die Komplexität heutiger Aufgaben in Automatisierung und Echtzeitdatenverarbeitung hat einen Trend zu verteilten, fehlertoleranten Mehrrechnersystemen hervorgebracht. Um die heute vorliegenden Anforderungen bezüglich Wirtschaftlichkeit und Zuverlässigkeit bei der Programmierung solcher Systeme zu erfüllen, werden vorwiegend höhere Programmiersprachen eingesetzt. Die Sprache PEARL nach DIN 66253 Teil 2 hat sich hierfür als besonders geeignet erwiesen.

Als eine der wenigen Programmiersprachen besitzt die Sprache PEARL nach DIN 66253 Teil 2 neben Beschreibungsmitteln für algorithmische Abläufe (Problemteil) auch Beschreibungsmittel für Strukturmerkmale zentraler Prozeßautomatisierungssysteme (Systemteil).

Wie bei allen anderen Sprachen fehlen jedoch auch bei PEARL gewisse Sprach- und Benutzerhilfsmittel für den Einsatz auf Mehrrechnersystemen. Dies hat in der Vergangenheit dazu geführt, daß für verschiedene Anwendungen auf Mehrrechnersystemen von den jeweiligen Firmen interne Spracherweiterungen für Mehrrechnersysteme vorgenommen wurden.

Mit der vorliegenden Norm erhält die Sprache PEARL mit dem deskriptiven Teil (Architekturbeschreibung) eine neue, beschreibende Programmkomponente. Als weitere Hauptkomponenten stehen weiterhin die Moduln zur Verfügung. Der Beschreibungsteil ist als übergeordnete Liste aufzufassen, deren Anweisungen nicht direkt in Maschinencode umgesetzt werden, sondern lediglich als Steueranweisungen für Systemprogramme dienen, das heißt zur Parametrierung von Code.

Der Beschreibungsteil umfaßt die Beschreibung von Stationen, die Beschreibung der physikalischen Verbindungswege zwischen Stationen, die - gegebenenfalls stationsübergreifenden - Anschlüsse zur Peripherie und die Beschreibung der Software-Konfiguration, das heißt die Zuordnung von Programtteilen zu Stationen.

Ein Hauptaspekt bei der Sprachentwicklung war die Berücksichtigung des zunehmenden Einsatzes fehlertoleranter Systeme. Bei Mehrrechnersystemen kann die "natürliche Redundanz" zum Erreichen von Fehlertoleranz herangezogen werden. Aus diesem Grunde umfaßt die Beschreibung der Software-Konfiguration auch die dynamische Rekonfiguration, das heißt die automatische Verlagerung von Programtteilen einer fehlerhaften Station auf eine andere Station. Als verlagerbare Einheiten dienen sogenannte Collections, das sind zu Modulgruppen zusammengefaßte Moduln.

Ein weiterer Gesichtspunkt bei der Programmierung von Mehrrechnersystemen ist die Kommunikation zwischen den Teilsystemen, die in Mehrrechner-PEARL ausschließlich durch Botschaftenaustausch auf Basis eines Portkonzepts möglich ist. Als netzglobale Größen dürfen lediglich Systemteilobjekte (Unterbrechungen, Datenstationen) eingesetzt werden.

Ports bilden - neben den Systemteilobjekten - die einzige Schnittstelle zwischen Collections. Im Beschreibungsteil können logische Verbindungen zwischen Collections sowie eine Abbildung der logischen auf physikalische Verbindungen vorgenommen werden. Der Botschaftenaustausch über logische Verbindungen kann auf Basis asynchroner und synchroner Mechanismen erfolgen.

Die in dieser Norm beschriebenen Botschaftsmechanismen sind oberhalb der Ebene 7 des ISO-Referenzmodells für offene Systeme nach DIN ISO 7498 *) angesiedelt und können auf den dort festgelegten Kommunikationsprotokollen aufsetzen.

*) Z. Z. Entwurf

3 Umfang der Norm

Mit der vorliegenden Norm wurde eine aufwärtskompatible Spracherweiterung zu DIN 66253 Teil 2 geschaffen. Eine Implementierung kann so vorgenommen werden, daß eine Neuübersetzung existierender Programmsysteme ohne vorherige Korrektur möglich ist.

Mit DIN 66253 Teil 2 wird sowohl der Gesamtumfang (Full-PEARL), als auch eine Untermenge (Basic-PEARL) festgelegt. Diese Abstufung wurde ebenfalls bei Mehrrechner-PEARL vorgenommen.

Basic-Mehrrechner-PEARL beschreibt einen minimalen Sprachumfang, der von jedem Übersetzer, der die Norm erfüllt, akzeptiert werden muß. Die Abstufungen bei Full-Mehrrechner-PEARL können frei festgelegt werden. Implementationsabhängige Einzelheiten müssen beschrieben werden und der Sprachumfang deutlich gekennzeichnet sein.

Die Beschreibung ist nach dem Schema Syntax - Semantik - Erläuterungen gegliedert. Die Sprache ist vollständig in den Teilen Syntax und Semantik definiert. Die Erläuterungen dienen zum besseren Verständnis und geben Hinweise auf mögliche Implementierungen.

4 Bezeichnung

Bezeichnung einer Programmiersprache Mehrrechner-PEARL, die den Bedingungen dieser Norm als Basic-Mehrrechner-PEARL (MB) entspricht:

PEARL DIN 66253-MB

Bezeichnung einer Programmiersprache Mehrrechner-PEARL, die den Bedingungen dieser Norm als Full-Mehrrechner-PEARL (MF) entspricht:

PEARL DIN 66253-MF

5 Begriffe

Architekturbeschreibung	Von den Ablaufteilen eines Programmsystems separierte Beschreibungsliste, die ein Abbild der Hard- und Software-Konfiguration, abhängig vom Betriebszustand des Implementierungssystems, beinhaltet.
Betriebszustand	Fehlertolerante Systeme basieren auf der Erfassung des Systemzustandes durch permanente Überwachung von Hard- und Softwarekomponenten. Alle Statusinformationen, die den Überwachern zugänglich gemacht werden (siehe "Statusregister"), werden als "Betriebszustand" bezeichnet. Der Betriebszustand, in dem das Mehrrechnersystem fehlerfrei arbeitet, wird als "initialer Betriebszustand" oder "Initialzustand" bezeichnet.
Botschaften	Botschaften sind Daten, die mit Hilfe expliziter Übertragungsanweisungen zwischen Tasks ausgetauscht werden.
Collection	Eine Menge von PEARL-Moduln kann organisatorisch zu einer Collection zusammengefaßt werden. Ein Mehrrechner-PEARL-Programmsystem besteht im allgemeinen aus mehreren Collections. Mit Hilfe von Collections wird der Gültigkeitsbereich von globalen Objekten (gegenüber DIN 66253 Teil 2) zusätzlichen Beschränkungen unterworfen. Dadurch werden Collections im laufenden Betrieb von einer Station zur anderen verlagerbar. Die Beschreibung von Collections erfolgt in der Architekturbeschreibung.
Fehlertolerantes System	Ein fehlertolerantes System beinhaltet eine Erhöhung der Verfügbarkeit durch die Anwesenheit von redundanten Komponenten. Die Redundanz kann funktionsbeteiligt sein, so daß beim Auftreten von Fehlern ein Leistungsabfall in Kauf genommen werden muß.
Konsistente Verteilung	Eine Verteilung von Collections auf Stationen wird als konsistent bezeichnet, wenn jede Collection in jedem Betriebszustand auf nicht mehr als einer Station in ablaufendem Zustand ist.
Netzwerkbetriebssystem	Zusammenfassung aller Systemprogramme, die die Datenübertragung zwischen Stationen eines Mehrrechnersystems bereitstellen.
Port	Vom Netzwerkbetriebssystem verwaltete Kommunikationsobjekte, die die Ein- und Ausgänge für Botschaften benennen (Sende- beziehungsweise Empfangsports). Ports werden von Tasks mit Hilfe von Übertragungsanweisungen adressiert. Zur Übertragung einer Botschaft muß eine Verbindung zwischen einem Sende- und Empfangsport existieren (siehe Verbindungsstruktur). Ports bilden die einzige Schnittstelle zwischen Collections.
Rekonfiguration	Eine Änderung der Collection-Verteilung beim Wechsel des Betriebszustandes wird als Rekonfiguration bezeichnet. Sie wird durch das Nachladen, Starten und Beenden von Collections bewerkstelligt. Die Rückkehr in den initialen Zustand heißt Rück-Rekonfiguration.
Statusregister	Der Betriebszustand einer Station wird durch ein Statusregister ablesbar gemacht. Unter Statusregister muß nicht notwendigerweise ein Hardware-Register verstanden werden. Entscheidend ist lediglich die Zugriffsmöglichkeit auf die Statusinformation. Werte des Statusregisters werden in dieser Norm durch Bitketten repräsentiert (Anmerkung: die Menge der zulässigen Bitketten ist implementationsabhängig).
Verbindungsstruktur	Gesamtheit der logischen Datenkanäle zwischen Sende- und Empfangsports, über die Botschaften übertragen werden. Je nach Anzahl der Sende- und Empfangsports in einer Verbindung ergibt sich eine unterschiedliche Topologie (1-->n, n-->1 - Verbindungen).

6 Beschreibungsteil für Mehrrechnersysteme (ARCHITECTURE_description)

Syntax:

Ein Mehrrechner-PEARL-Programm besteht aus einer Menge von Moduln und einer übergeordneten Beschreibungsliste (ARCHITECTURE_description). Die Beschreibungsliste stellt ein Abbild des Mehrrechnersystems dar und ist wie folgt aufgebaut:

```
ARCHITECTURE_description :
    'ARCHITECTURE' ';' ARCHITECTURE_divisions 'ARCHEND' ';' .
ARCHITECTURE_divisions :
    STATION_division
    NET_division
    SYSTEM_division
    CONFIGURATION_division .
```

Semantik:

Aus der Syntax geht hervor, daß die Architekturbeschreibung in vier Hauptkomponenten aufgeteilt ist:

- Der Stationsteil (siehe Abschnitt 6.1) ordnet allen Stationen einen Namen und charakterisierende Attribute zu.
- Die Verbindungsspezifikation (siehe Abschnitt 6.2) enthält die physikalischen Verbindungswege zwischen Stationen und ordnet diesen Namen zu. Auf diese Weise kann eine Wegeauswahl für bestimmte Botschaften vorgenommen werden (siehe Abschnitt 6.4.3).
- Die Peripheriebeschreibung (siehe Abschnitt 6.3) besteht im wesentlichen aus dem bekannten PEARL-Systemteil, der in Mehrrechner-PEARL einer Station zugeordnet wird (siehe Abschnitt 6.3.1). Weiterhin kommt der stationsübergreifende Gerätezugriff hinzu (siehe Abschnitt 6.3.2).
- Die Beschreibung der Software-Konfiguration und der dynamischen Rekonfiguration nach Fehlern besteht aus Ladeanweisungen und Anweisungen zur Aufnahme logischer Verbindungen, die beide in Abhängigkeit vom Betriebszustand durchgeführt werden.

Erläuterungen:

Alle Anweisungen der Architekturbeschreibung werden nicht in Maschinencode umgesetzt, sondern dienen lediglich der Parametrierung von Systemprogrammen des Programmerzeugungs- und Ablaufsystems. Als Implementierungshintergrund kann ein Pre-Prozessor gesehen werden, der die Liste bearbeitet und die jeweils benötigten Systemprogramme aktiviert.

6.1 STATION-Teil: Beschreibung von Stationen

Syntax:

Der Stationsteil beschreibt alle Stationen des Mehrrechnersystems durch Angabe ihres Namens, ihrer Attribute und ihrer möglichen Betriebszustände:

```
STATION_division :
    'STATIONS' ';' STATION_descr + 'STAEND' ';' .
STATION_descr :
    STATION_names
    STATION_attribute *
    STATION_state_id * .
```

Die Beschreibung einer Station wird eingeleitet durch das Namensattribut:

```
STATION_names :
    'NAME' ':' (STATION_id // ',') ';'
STATION_id :
    ['(' FIXED_const_denot ')'] id / FIXED_const_denot
```

Die weitere Notation bis zur nächsten Angabe des Schlüsselwortes NAME umfaßt zunächst die (optionale) Angabe weiterer Attribute:

```
STATION_attribute :
    STATION_attribute_id ':' STATION_attribute_values
```

Es können beliebig viele Attribute mit frei gewählten Bezeichnern angegeben werden. Die Angabe der Attributwerte unterliegt einer freien Syntax.

Den Attributen folgen (optional) die Betriebszustände der Station.

Eingeleitet durch das Attribut-Schlüsselwort STATEID folgt jeweils eine Statusdefinition:

```
STATION_state_id :
    'STATEID' ':' '(' (state_def // ',') ')' ';'
state_def :
    state_id ':' state_register_descr
    [':' state_register_dontcare_descr]
state_register_descr :
    BIT_string %% the description
state_register_dontcare_descr :
    BIT_string %% may use a type different
                %% from BIT-string
state_id :
    id
```

Semantik:

Mit dem Namensattribut wird einer (oder mehreren) Station(en) ein Identifikator zugeordnet. Mit der Angabe von Feldern oder Namenslisten kann auf zwei Arten dieselbe Stationsbeschreibung zur Identifizierung mehrerer Stationen verwendet werden (homogenes System).

Einige häufig wiederkehrende Attributnamen sind vordefiniert (Anhang A).

Mit Hilfe des Attributschlüsselworts STATEID wird einem bestimmten Betriebszustand einer Station jeweils ein Bezeichner (state_id) zugeordnet. Der Betriebszustand wird in einem Statusregister festgehalten (siehe Erläuterungen in diesem Abschnitt).

Im Konfigurationsteil können diese Bezeichner zu boole'schen Ausdrücken verknüpft werden, wobei auf Bezeichner einer Station mit Hilfe der Selektion "Stationsidentifikator.Statusidentifikator" zugegriffen werden kann. Eine solche Selektion wird als "Statusdenotation" bezeichnet (siehe Abschnitt 6.4.3).

Der Betriebszustand, der einem Statusbezeichner durch eine Statusdefinition zugeordnet wird, wird als "Statusregisterwert" bezeichnet. Seine Beschreibung erfolgt durch Angabe der Bitkette, die das Statusregister in diesem Betriebszustand einnimmt. Optional kann eine weitere Bitkette zur Benennung der bedeutungslosen Positionen für die Zustandsbeschreibung angegeben werden. Die gesetzten Bits markieren dabei die zur Beschreibung des Zustands relevanten Positionen. Fehlt diese Angabe, so sind alle Bits des Statusregisters für die Zustandsangabe relevant.

Anwenderprogramme können auf das Statusregister mit Hilfe der vordefinierten Prozeduren SETSTATE und GETSTATE (siehe Anhang C) zugreifen. Mit SETSTATE kann der Status softwaremäßig gesetzt werden, wodurch Rekonfigurationsmaßnahmen (siehe Abschnitt 6.4.3) ausgelöst werden. Mit GETSTATE kann der aktuelle Betriebszustand erfragt und somit der Grund der Aktivierung eines Programmteiles (Initial- oder Wiederanlauf) in Erfahrung gebracht werden.

Erläuterungen:

In einem fehlertoleranten Mehrrechnersystem kann auf Hard- oder Softwarefehler (zum Beispiel Ausfall von Stationen) mit dynamischer Rekonfiguration von Software - notfalls mit Inkaufnahme eines Leistungsabfalls - reagiert werden (siehe Abschnitt 6.4.3). Dabei können vielfältige Überwachungsmechanismen in das System eingebaut sein. Die Schnittstelle der Fehlerdetektionsmechanismen zum Anwendungsprogramm - zur Formulierung einer anwendungsspezifischen Rekonfigurationsstrategie - bildet ein Statusregister in jeder Station, in dem der Betriebszustand der Station festgehalten wird. (Anmerkung: das Wort "Register" impliziert nicht notwendigerweise ein Hardwareregister, die Implementation kann eine Verallgemeinerung durch einen Zustandsvektor mit spezifischer Komponentenart soft- oder hardwaremäßig vorsehen). Die Statusregister der Stationen können unterschiedliche Wortbreite besitzen. In jeder funktionsfähigen Station müssen die aktuellen Betriebszustände aller anderen Stationen verfügbar sein. Bei Änderung dieser Betriebszustände kann die Herstellung einer konsistenten Systemsicht Zeit beanspruchen. Wird von einem Überwacher das Statusregister einer Station verändert, so muß ein im System vorgesehener Aktualisierungsmechanismus die Statusregisterlisten aller Stationen in einen konsistenten Zustand bringen.

Die Einführung mnemotechnischer Bezeichner für Betriebszustände ermöglicht die Programmierung von Rekonfigurationsstrategien im Konfigurationsteil (siehe Abschnitt 6.4.3).

Bei Verwendung eines verallgemeinerten Statusregisters müssen die einzelnen Komponenten nicht notwendigerweise durch Bitketten realisiert werden. Der Anschaulichkeit wegen wurde jedoch diese Betrachtungsweise hier beibehalten, was in folgenden Abschnitten - insbesondere Abschnitt 6.4.3 und Anhang C - zu berücksichtigen ist.

6.2 NET-Teil: Beschreibung der physikalischen Verbindungswege**Syntax:**

Zur Beschreibung der Netzwerktopologie dient die Angabe von Verbindungsspezifikationen (connection_spc, DIN 66253 Teil 2) mit einer erweiterten Syntax:

```
NET_division :
    'NET' ';' connection_spc * 'NETEND' ';' ;
```

Semantik:

Die Semantik einer solchen Verbindungsspezifikation legt das Vorhandensein einer direkten Verbindung zwischen den Endpunkten fest.

Erläuterungen:

Die Beschreibung beliebiger Netzwerktopologien ist durch einen Anweisungstyp möglich, mit dem zwei Endpunkte miteinander verbunden werden können. Ein solcher Anweisungstyp ist in PEARL nach DIN 66253 Teil 2 bereits vorhanden: mit Hilfe einer Verbindungsspezifikation (Systemteil) kann eine Verbindung zwischen (stationslokalen) Geräten beschrieben werden. Den Endpunkten einer solchen Verbindung (endpoint_element) können Benutzernamen (user_id) zugewiesen werden. Die Benutzernamen für die Endpunkte einer Verbindung werden als "Leitung" bezeichnet. Leitungen können uni- oder bidirektional sein.

Zur Beschreibung stationsübergreifender Verbindungen muß die Angabe eines Endpunktes lediglich um einen Stationsidentifikator (siehe Abschnitt 6.1) ergänzt werden. Zur syntaktischen Eingliederung von Stationsidentifikatoren in Systemteilanweisungen wird auf Abschnitt 6.3 verwiesen. Die vollständige Syntax des erweiterten Systemteils ist in Anhang E zu finden.

Bei der Beschreibung logischer Verbindungen (siehe Abschnitt 6.4.2) kann eine Abbildung auf physikalische Verbindungen vorgenommen werden. Als Bindeglied dient die Angabe von Leitungen; die Beschreibung einer Strecke über mehrere Stationen (mehrstufige Leitung) kann durch eine Liste von Endpunkten vorgenommen werden.

6.3 SYSTEM-Teil: Beschreibung des Zugangs zur Peripherie

6.3.1 Zuordnung zur Station

Syntax:

Die Zuordnung von Systemteilanweisungen zu Stationen geschieht auf gleiche Weise wie die Zuordnung von Attributen zu Stationen im Stationsteil: einer Menge von Anweisungen wird jeweils ein Namensattribut mit der Angabe eines Stationsidentifikators vorangestellt. Sind die Systemteile für mehrere Stationen identisch, so kann auch ein Feld oder eine Namensliste angegeben werden.

Die Eingliederung des Namensattributes erfordert die Ergänzung der Syntaxvariablen `sx_SYSTEM_division` nach DIN 66253 Teil 2 um folgende weitere Ableitung:

```
SYSTEM_division :
    'SYSTEM' ';' STATION_SYSTEM_division * 'SEND' ';' .
STATION_SYSTEM_division :
    STATION_names (device_or_connection_spc ';' ) * .
```

Die Ableitung der Variablen `STATION_names` wurde in Abschnitt 6.1 eingeführt.

Aus Kompatibilitätsgründen darf der Systemteil weiterhin dem Modul untergeordnet sein; es gelten die Gültigkeitsbereichseinschränkungen. Für ein Mehrrechner-PEARL-Programm muß jedoch eine der beiden Möglichkeiten ausschließlich verwendet werden.

Semantik:

Die Station, auf der die beschriebenen Geräte und Anschlüsse physikalisch liegen, wird festgelegt.

Erläuterungen:

Der Systemteil nach DIN 66253 Teil 2 umfaßt zwei Aufgabenbereiche:

- a) Beschreibung von Geräten mit Zuordnung von Benutzernamen
- b) Beschreibung der Verbindungswege zwischen Geräten mit Zuordnung von Benutzernamen zu den Anschlußpunkten

In PEARL nach DIN 66253 Teil 2 besteht ein Modul aus einem Systemteil und einem Problemteil, wobei einer der beiden Teile entfallen kann. Ist der Systemteil vorhanden, so ist der Gültigkeitsbereich der deklarierten Systemteil-Benutzernamen das gesamte PEARL-Programm. In Mehrrechner-PEARL können Moduln zu Modulgruppen (Collections, siehe Abschnitt 6.4.1) zusammengefaßt werden. Der Gültigkeitsbereich aller Objekte umfaßt maximal eine Collection; um diese Beschränkung für bestimmte Systemteilobjekte aufheben zu können, werden syntaktische Erweiterungen vorgenommen.

In einem fehlertoleranten Mehrrechnersystem mit automatischer Rekonfiguration von Software ist die Gerätebeschreibung des Systemteils für einen Programmteil abhängig von der Station, auf welche dieser Programmteil geladen wird. Es können Aktivierungseinplanungen für unterschiedliche Betriebszustände auf unterschiedlichen Stationen vorgesehen sein (siehe Abschnitt 6.4.3). Es ist dafür zu sorgen, daß einem Benutzernamen (`user_id`) auf jeder dieser Stationen jeweils ein Systemname (`SYSTEM_id`) zugeordnet ist.

Die Beibehaltung der bisherigen Programmstruktur, in der der Systemteil dem Modul untergeordnet ist, ist deshalb nicht ausreichend, da für jede Station ein anderer Systemteil benötigt werden kann. Für rekonfigurierende Mehrrechner-PEARL-Programme kann der Systemteil daher aus den Moduln ausgelagert und der übergeordneten Architekturbeschreibung zugeschlagen werden. Dabei werden die Systemteilanweisungen nach Stationen aufgeteilt. Aus den Ladeanweisungen des Konfigurationsteils (siehe Abschnitt 6.4) ist bekannt, auf welche Station eine Modulgruppe unter bestimmten

Betriebszuständen geladen wird; die Problemteile dieser Modulgruppen müssen mit den jeweiligen Systemteilen für diese Station vorgebunden und so die entsprechenden Ladeversionen erzeugt werden.

Um die vorliegende Norm eigenständig lesbar zu halten, ist die gesamte Syntax des Systemteils mit den Erweiterungen in Anhang E zusammengefaßt.

Bei der Abarbeitung von Collection-Definitionen (siehe Abschnitt 6.4.1) wird im allgemeinen der Binder aktiviert. Es werden Ladeversionen für alle Stationen erstellt, auf die eine Collection geladen werden kann (Ladeanweisungen, siehe Abschnitt 6.4.3). Der jeweilige Systemteil beginnt bei der Namensanweisung, die den Identifikator der Station enthält und endet bei der nächsten Namensanweisung.

6.3.2 Stationsübergreifender Gerätezugriff

Syntax:

Der implementationsabhängige Systemname eines Gerätes wird ausschließlich aus der Syntaxvariablen `sx_SYSTEM_id` abgeleitet.

Durch Einbringen einer weiteren Ableitung der Variablen `sx_SYSTEM_id` und Verwendung eines Stationsidentifikators (siehe Abschnitt 6.1) wird die Ansprache von abgesetzten (nicht stationslokalen) Geräten ermöglicht:

```
SYSTEM_id :
    [STATION_id '.' ] local_SYSTEM_id
local_SYSTEM_id :
    id ['('enclosure')'] / id '('element_number')'
```

Die Ableitung von `local_SYSTEM_id` entspricht der ursprünglichen Ableitung von `SYSTEM_id`.

Semantik:

Mit den `SYSTEM_id`'s können in PEARL nach DIN 66253 Teil 2 (stationslokale) Geräte in Geräte- (`sx_device_spc`) und Verbindungsspezifikationen (`sx_connection_spc`) angesprochen werden, in Mehrrechner-PEARL zusätzlich nicht-stationslokale.

Die Implementationshandbücher müssen die Verwendungsmöglichkeiten und die jeweiligen Einschränkungen festlegen.

Erläuterungen:

Eine Anwendungsmöglichkeit des abgesetzten Gerätezugriffs besteht bei der Programmierung von Fehlertoleranzstrategien: bei der Verlagerung von Programmteilen nach einem Stationsausfall können - falls hardwaremäßige Voraussetzungen gegeben sind - die Automatisierungsfunktionen der ausgefallenen Station durch die Ersatzstation nur dann wahrgenommen werden, wenn diese Zugriff zu den E/A-Geräten der ausgefallenen Station hat. Dazu müssen die Benutzernamen für Geräte auf einen Systemnamen mit vorgeschaltetem Stationsidentifikator führen.

Einen zusammenfassenden Vergleich der Gestaltungsmöglichkeiten des Systemteils gibt Tabelle 1.

Tabelle 1. Zusammenfassender Vergleich der Gestaltungsmöglichkeiten des Systemteils.

	Systemteil in			
	Modul	Architektur- beschreibung		
Grund für Einführung	bestehende Übersetzungssysteme (Kompatibilität)	Zusammenfassung der deskriptiven Teile, Zuordnung zur Station		
vorwiegende Verwendung	in nicht rekonfigurierenden Systemen	in rekonfigurierenden Systemen		
Gültigkeitsbereich von Problemteilobjekten mit GLOBAL-Attribut	Collection	Collection		
Gültigkeitsbereich von Systemteilobjekten (user_id)	Stations-identifikator vorhanden	nicht vorhanden	Stations-identifikator vorhanden	nicht vorhanden
	alle Stationen	Collection	alle Stationen	eigene Station

6.4 CONFIGURATION-Teil: Beschreibung der Software-Konfiguration

Syntax:

Die Beschreibung der unterscheidbaren Betriebszustände erfolgt durch logisch verknüpfte Statusdenotationen (siehe Abschnitt 6.1), wobei in einen Initialzustand und einen - bei aufgetretenen Störungen einzunehmenden - Rekonfigurationszustand unterschieden werden kann (siehe Abschnitt 6.4.3). Dementsprechend sind die Anweisungen des Konfigurationsteils in einen Initialteil und einen Rekonfigurationsteil unterteilt:

```
CONFIGURATION_division :  
    'CONFIGURATION' ';' CONFIGURATION_stmts 'CONFEND' ';' .  
CONFIGURATION_stmts :  
    initial_part [reconfiguration_part] .  
initial_part :  
    COLLECTION_definition +  
    initial_LOAD_stmt +  
    CONNECT_stmt * .  
reconfiguration_part :  
    reconfiguration_stmt + .
```

Semantik:

Die Anweisungen des Konfigurationsteils geben eine vollständige Beschreibung der Software-Konfiguration eines verteilten PEARL-Programmsystems für jeden definierten Betriebszustand. Der Betriebszustand für jede Station ist in einem Statusregister festgehalten (siehe Abschnitt 6.1).

Erläuterungen:

Im Initialteil werden zunächst die konfigurierbaren Einheiten einschließlich ihrer Portschnittstelle definiert (siehe Abschnitt 6.4.1). Die Erzeugung des Initialzustandes geschieht durch Ausführen der Ladeanweisungen (siehe Abschnitt 6.4.3) und anschließendem Aufbau logischer Verbindungen (siehe Abschnitt 6.4.2) zwischen Ports. Die logischen Verbindungen können dabei auf physikalische Verbindungen (siehe Abschnitt 6.2) abgebildet werden.

Im Rekonfigurationsteil (siehe Abschnitt 6.4.3) werden die möglichen Betriebszustände sowie die zugehörigen Maßnahmen zur Rekonfiguration (Verbindungsabbau, Verlagern von Collections, Verbindungsneuaufbau) beim Eintreten beziehungsweise Verlassen der Betriebszustände festgelegt, wobei auch Zustandsübergänge von einem Fehlerzustand in einen anderen berücksichtigt werden.

6.4.1 COLLECTIONS: Konfigurierbare Einheiten**Syntax:**

Bei der Erstellung eines verteilten PEARL-Programmsystems ist eine vollständige Einordnung der Moduln des Programmsystems in Modulgruppen (Collections) obligatorisch.

Collections werden mit Hilfe von Collection-Definitionen eingeführt:

```

COLLECTION_definition :
    'COLLECTION' COLLECTION_id MODULE_enumeration
    PORT_enumeration ';'
COLLECTION_id :
    id
MODULE_enumeration :
    'MODULES' (MODULE_id // ',')
MODULE_id :
    id
PORT_enumeration :
    'PORTS' (PORT_id // ',')
PORT_id :
    id

```

Die Definition enthält eine Aufzählung der Moduln, aus denen die Collection besteht. Die Moduln werden über ihren Namen angesprochen, der zu diesem Zweck in der MODULE-Anweisung am Anfang jedes Moduls angegeben werden muß. Weiterhin enthält eine Collection-Definition eine Aufzählung der Ports.

Semantik:

Ein Port ist ein neues PEARL-Datenobjekt. Der Deklaration im Konfigurationsteil folgt eine Spezifikation im Problemteil (siehe Abschnitt 7.1), die dem Port eine Reihe von Attributen zuordnet und die auf den Port anwendbaren Operationen des Botschaftenaustausches (siehe Abschnitt 7.2) festlegt.

Innerhalb von Collections gelten bezüglich des GLOBAL-Attributes die Regeln von PEARL nach DIN 66253 Teil 2. Die Kommunikation zwischen Collections erfolgt ausschließlich über Botschaften. Die Ports bilden die zugehörige Schnittstelle.

Der Gültigkeitsbereich eines Portobjekts erstreckt sich über die Collection, das heißt die auf einen Port anwendbaren Operationen können nur innerhalb der Collection ausgeführt werden, der er angehört. Während jedoch für den Ablaufteil eines PEARL-Programms Portnamen Collection-lokal sind, können im Konfigurationsteil Ports verschiedener Collections miteinander verbunden werden (siehe Abschnitt 6.4.2), was den Botschaftenaustausch zwischen Collections ermöglicht. Diese Verbindungen müssen während der gesamten Laufzeit eines PEARL-Programms nicht notwendigerweise konstant sein (siehe Abschnitt 6.4.3).

Es sei C die Menge aller Collections und M die Menge aller Moduln eines Mehrrechner-PEARL-Programms. Für alle c aus C und für alle m aus M gilt: Die Relation "m ist Bestandteil von c" von M nach C ist nicht notwendigerweise eine Funktion, das heißt ein Modul kann Bestandteil verschiedener Collections sein.

Erläuterungen:

Die Abarbeitung der Modulaufzählung kann die direkte Ansteuerung des Übersetzers für die Moduln zur Erzeugung maschinenunabhängigen Zwischencodes zur Folge haben, da Information aus dem Problemteil (Portdeskriptoren, siehe Abschnitt 6.4.2) zur Bearbeitung weiterer Anweisungen des Konfigurationsteils benötigt wird (siehe Abschnitt 6.4.2). Zur Ansteuerung des Codegenerators für die Zielmaschinen müssen aus den Ladeanweisungen (siehe Abschnitt 6.4.3) die Stationsnamen derjenigen Stationen entnommen werden, auf die die Collection geladen werden soll. Weitere Informationen für den Codegenerator kann der Stationsteil (siehe Abschnitt 6.1) enthalten. Abschließend kann bei Abarbeitung einer Collectiondefinition auch ein Binder angesteuert werden, der aus den übersetzten Moduln eine ladbare Einheit erzeugt.

Mit dem hier eingeführten Portkonzept wird die direkte Ansprache von Kommunikationsobjekten in fremden Collections vermieden und Verbindungen zwischen Collections werden nicht implizit festgeschrieben. Auf diese Weise sind Collections in wechselnder Umgebung einsetzbar: zur Ansprache der Anwendungsfunktionen, die durch eine Collection bereitgestellt werden, muß lediglich eine Verbindung zu dieser Collection über ihre Portschnittstelle hergestellt werden.

Die Verwaltung der Portobjekte obliegt der Netzwerkbetriebssystem-Software (NBS) in jeder Station. Um ankommende Botschaften einem Empfänger zuordnen zu können, verwaltet das NBS eine Namensliste der sich aktuell auf der Station befindlichen Ports (Portliste). Der Name eines Ports setzt sich dabei aus dem Collection-Namen und dem eigentlichen - Collection-lokalen Portnamen zusammen. Die Portliste enthält Zeiger auf die Collection-lokalen Portobjekte, die aus der Portspezifikation erzeugt werden. In einem Portobjekt sind die Portattribute (siehe Abschnitt 7.1) und die Verbindungen (siehe Abschnitt 6.4.2) eingetragen, die der Port besitzt. Ein angebundener Port kann durch eine dreistufige Namensgebung beschrieben werden: Stationsname, Collection-Name und Portname.

6.4.2 Beschreibung logischer Verbindungen

In Abschnitt 6.2 wurden die Möglichkeiten zur Beschreibung der physikalischen Verbindungswege zwischen den Stationen eines Mehrrechnersystems aufgezeigt. Im folgenden werden nun logische Verbindungen zwischen konfigurierbaren Einheiten auf Portbasis eingeführt. Ebenfalls wird die Abbildung der logischen auf die physikalische Verbindungsstruktur definiert.

Syntax:

Logische Verbindungen werden mit Hilfe von Verbindungsaufbauanweisungen (`CONNECT_stmt`) beschrieben:

```

CONNECT_stmt :
    'CONNECT' global_PORT_list PORT_direction_qualifier
    global_PORT_list [line_map] ';'
global_PORT_list :
    (global_PORT_id // ',')
global_PORT_id :
    COLLECTION_id '.' PORT_id
PORT_direction_qualifier :
    '->' / '<-'
line_map :
    fixed_line / preferred_line
fixed_line :
    'VIA' line_list
line_list :
    (user_id // ',')
preferred_line :
    'PREFER' line_list

```

Semantik:

Eine Verbindungsaufbauanweisung enthält die Angabe zweier disjunkter, nichtleerer Mengen von Portobjekten (`global_PORT_list`), die in Collection-Definitionen vereinbart sein müssen. Die Angabe eines Ports erfolgt dabei in der globalen Form durch die Selektion "Collection-Name.Portname". Der zwischen den beiden Listen stehende Richtungs Pfeil ordnet jedem Port ein Richtungsattribut zu: er zeigt von der Menge der Sende- auf die Menge der Empfangsports. Ein Portobjekt kann in mehreren Verbindungsaufbauanweisungen des Konfigurationsteils genannt werden, es muß jedoch ein eindeutiges Richtungsattribut besitzen.

Außer dem Richtungsattribut besitzt ein Port weitere Attribute, die in seiner Portspezifikation innerhalb eines Moduls (siehe Abschnitt 7.1) angegeben werden müssen. Bezüglich dieser Attribute ergeben sich weitere Einschränkungen für die Zulässigkeit bestimmter Portobjekte in einer Verbindungsaufbauanweisung (siehe Abschnitt 7.1):

- über alle Sende- und Empfangsports in einer Verbindung kann nur derselbe Botschaftentyp übertragen werden
- der Protokolltyp muß bei allen Ports übereinstimmen (No-Wait-Send-, Blocking-Send- oder Send-Reply-Protokoll)
- enthält die Liste der Empfangsports mehr als ein Element, darf das Send-Reply-Protokoll nicht verwendet werden.

Verbindungsaufbauanweisungen können in beiden Hauptabschnitten des Konfigurationsteils eingesetzt werden. Die Verbindungsanweisungen im Initialteil beschreiben die initiale Verbindungsstruktur zwischen allen Collections. Im Rekonfigurationsteil (siehe Abschnitt 6.4.3) wird die betriebszustandsabhängige Veränderung der Verbindungsstruktur beschrieben. Für jeden Port muß es mindestens einen Betriebszustand geben, in dem er mit mindestens einem weiteren Port verbunden ist. Ein Port muß in der Initialkonfiguration nicht notwendigerweise eine Verbindung besitzen.

Die Verbindungsaufnahmen für die Initialkonfiguration müssen vor dem Start der ersten Task des Mehrrechner-PEARL-Programms erfolgen, das heißt beim Anlauf des Programms muß gewährleistet sein, daß alle spezifizierten Verbindungen existent sind. Die Durchführung wird von den beteiligten Sendeports initiiert.

Die Abbildung der logischen auf die physikalische Verbindungsstruktur wird durch Angabe von Leitungen - vereinbart in einer physikalischen Verbindungsspezifikation (siehe Abschnitt 6.2) - ermöglicht und kann auf drei unterschiedliche Arten erfolgen:

- Bei Angabe einer festen Leitung (`fixed_line`) in der Leitungszuordnung (`line_map`) - eingeleitet durch das Schlüsselwort VIA - darf der Datenstrom vom Sende- zum Empfangsport ausschließlich nur über diese Leitung erfolgen.