ERFAHRUNGEN MIT PEARL - Erweiterungswünsche am Beispiel von zwei Anwendungen

Dipl.-Math. K.Mangold, Konstanz

Zusammenfassung:

Zwei Anwendungen von PEARL werden dargestellt.
Am Beispiel dieser Anwendungen werden Erweiterungswünsche an PEARL dargestellt, die vorwiegend Datenstrukturen, asynchrone Ein-/Ausgabe, Mehrrechnersysteme und Botschaftsmechanismen betreffen.

Summary

Two applications are presented. Useful extensions of PEARL in respect of these applications are discussed, esp. data structures, asynchronous Input/Output facilities, multiprocessor systems and message mechanisms.

Gliederung:

1. Einleitung

Am Beispiel von zwei Anwendungen von PEARL, dem Wetterdaten- und Informations-Anzeige-System und dem Heeres-Flugabwehr- und Aufklärungs-Führungssystem, sollen Erfahrungen mit PEARL dargestellt werden.

Obwohl bei beiden Vorhaben derselbe Rechner und dieselbe PEARL-Implementierung eingesetzt werden, soll versucht werden, Anregungen für die Weiterentwicklung der Sprache zu geben und nicht implementierungsspezifische Punkte zu kritisieren.

Diese Wünsche wenden sich sowohl an die Entwicklungsumgebungen, die inzwischen durch Ada unter der Bezeichnung APSE stark an Bedeutung gewonnen haben, als auch an den Sprachumfang.

Hierbei wird allerdings nicht auf Basic PEARL eingegangen, da dieser Umfang als zu wenig mächtig angesehen wird, sondern es wird unterstellt, daß praktisch Full PEARL verfügbar ist, wobei aber auch hier noch Erweiterungen wünschenswert wären.

Nicht eingegangen wird auf das Problem fast aller 16-Bit-Maschinen, nämlich den inzwischen billigen und in größerem Umfang vorhandenen Speicher mit dem verfügbaren 16-Bit-Adreßraum zu bearbeiten, da dies kein unmittelbares PEARL-Problem ist.

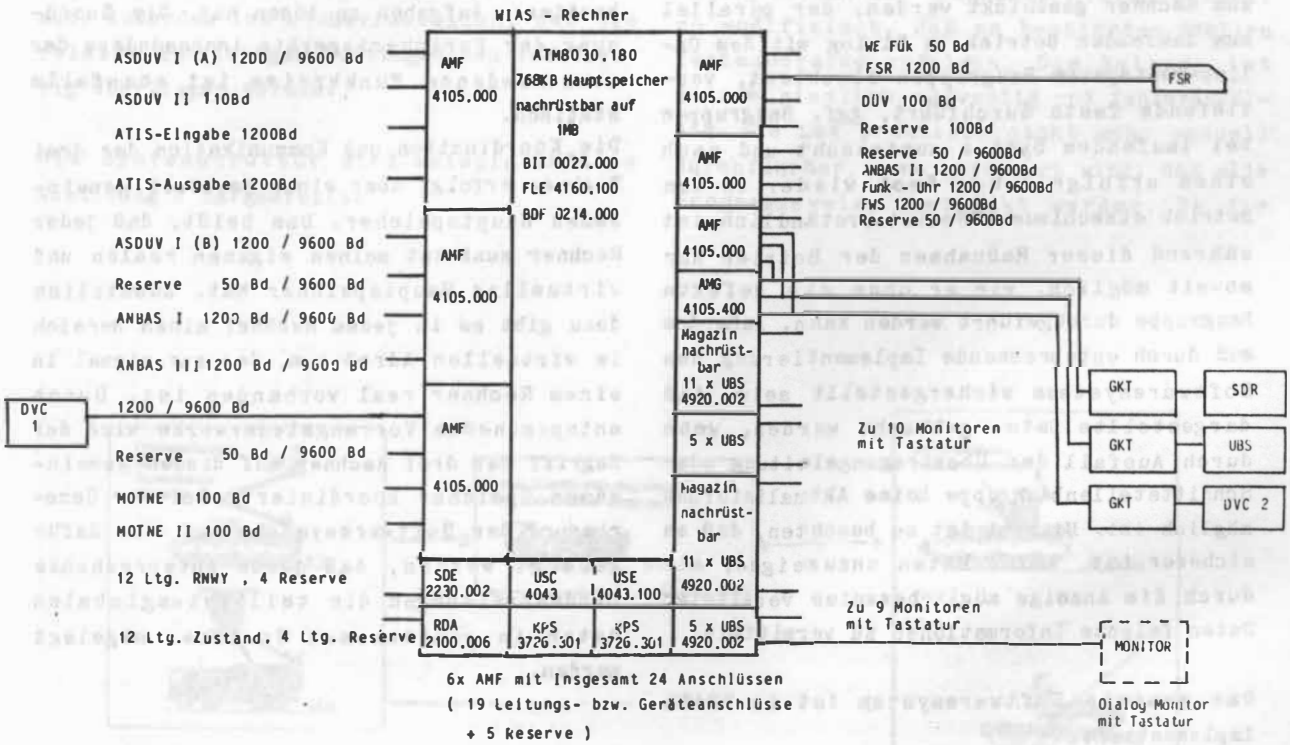
Es wird erst dann zum Problem für den PEARL-Programmierer, wenn die entsprechenden Systemstrukturen im PEARL-Programm berücksichtigt werden müssen.

2. Das System WIAS

Das Wetterdaten- und Informations-Anzeige-System WIAS, das von AEG-Telefunken für die Bundesanstalt für Flugsicherung in Frankfurt realisiert wird, übernimmt von etwa einem D u t z e n d a s y n c h r o n e r Datenfernübertragungsleitungen unterschiedliche Daten, vorwiegend Wettermeldungen. Je nach Art der Meldungen werden unterschiedliche Übertragungsraten und Datensicherungsverfahren eingesetzt. Auch die Meldungshäufigkeit ist auf den einzelnen Leitungen stark unterschiedlich.

Diese Meldungen werden entgegengenommen, auf Plausibilität geprüft und ausgewertet. Auf Grund der übermittelten Daten werden Informationseinheiten aufgebaut, die zu Darstellungseinheiten (Bilder) zusammengestellt werden. Dabei wird durch Betriebsparameter gesteuert, welche Daten aus welchen Meldungen automatisch an bestimmte Stellen vordefinierter Bilder eingeblendet werden. Andere Daten können von einem Dialogarbeitsplatz aus manuell verteilt werden.

Diese Bilder (Text- und Semigraphik) dienen primaer zur Information der Fluglotsen. Sie können über Anwahltastaturen von bis zu sechsundneunzig Arbeitsplätzen ausgewählt und abgerufen werden. Die Darstellung erfolgt auf Farbfernsehmonitoren mit RGB-Anschluß. Durch entsprechende Parametrierung kann das Anpaßwerk softwaremäßig so umge-



AMF = Asynchrone Mehrfach - Fernbetriebseinheit
SDE = Spontan - Digitaleingabe
RDA = Relais - Digitalausgabe
BIT = Selbstprüfeinrichtung
FLE = Fernladeeinrichtung
BOF = Bedienfeld
USC = Umlaufspeicher
KPS = Kassettenplattenspeicher
UBS = Universeller Bildspeicher
DVC = Digital Video Converter
GKT = Gerätekabeltreiber
AMG = Gleichstromübertragungseinrichtung

Abb. 1 System-Übersicht WIAS

stellt werden, daß die gesamte, mehrfarbige Information über den Grünkanal ohne Informationsverlust an Schwarz-Weiß-Monitore übertragen wird. Zusätzlich zu diesen Darstellungsfunktionen enthält das System noch Dokumentations- und Archivierungsfunktionen auf Drucker und Plattenspeichern.

Da das System eine hohe Verfügbarkeit haben muß, wurde umfangreiche Unterstützungssoftware zur Erkennung und Lokalisierung von Fehlern implementiert. Dieser sogenannte Online-Test beruht auf einem hardwaremäßig realisierten Built-in-test-equipment (BITE), welches sowohl beim Systemstart, als auch während des Betriebs die zentralen Hardwarekomponenten überprüft und Fehler lokalisiert und anzeigt. Zusätzlich werden softwaremäßig sämtliche Peripherieschnittstellen regelmäßig getestet. Werden hierbei Fehler entdeckt, oder treten bei der betrieblichen Nutzung Fehler auf, so werden diese, ebenso wie von BITE erkannte Fehler einem Fernwirksystem übermittelt. Von dieser zentralen Stelle aus kann dann ein Wartungstechniker zum Rechner geschickt werden, der parallel zum laufenden Betrieb im Dialog mit dem Online-Testsystem Baugruppen blockiert, vertiefende Tests durchführt, ggf. Baugruppen bei laufendem System austauscht und nach einem erfolgreichen Test wieder in den Betrieb einschleust. Selbstverständlich ist während dieser Maßnahmen der Betrieb nur soweit möglich, wie er ohne die defekte Baugruppe durchgeführt werden kann. Außerdem muß durch entsprechende Implementierung des Softwaresystems sichergestellt sein, daß dargestellte Daten gelöscht werden, wenn durch Ausfall der Übertragungsleitung oder Schnittstellenbaugruppe keine Aktualisierung möglich ist. Hierbei ist zu beachten, daß es sicherer ist, keine Daten anzuzeigen, als durch die Anzeige möglicherweise veralteter Daten falsche Informationen zu vermitteln.

Das gesamte Softwaresystem ist in PEARL implementiert.

3. Das System HFlaAFüSys

Das Heeres-Flugabwehr- und Aufklärungs-Führungssystem HFlaAFüSys, das von der SIEMENS AG mit mehreren Unterauftragnehmern im Auftrag des Bundesamtes für Wehrtechnik und Beschaffung entwickelt wird, ist ein Verbundsystem mit drei wesentlichen Komponenten:

- mehreren Sensor-Teilsystemen
- einem Führungs-Teilsystem
- mehreren Waffen-Teilsystemen

- Die Sensor-Teilsysteme

Hier wird in Radarstationen eine für das jeweilige Teilsystem aktuelle Luftlage ermittelt und ggf. dargestellt. Diese sog. Einzel-Luftlage wird zusammen mit den per Datenfunk eintreffenden Luftlage-Informationen anderer Stationen zu einer Gesamtluftlage verschmolzen. Die Aufgaben werden in jedem Sensorsystem von drei gekoppelten Rechnern erledigt. Die Aufgabenverteilung auf die Rechner erfolgt als statischer Lastverbund, wobei jeder Rechner bestimmte Aufgaben zu lösen hat. Die Zuordnung der Peripheriegeräte insbesondere der verschiedenen Funkkreise ist ebenfalls statisch.

Die Koordination und Kommunikation der drei Rechner erfolgt über einen partiell gemeinsamen Hauptspeicher. Das heißt, daß jeder Rechner zunächst seinen eigenen realen und virtuellen Hauptspeicher hat. Zusätzlich dazu gibt es in jedem Rechner einen Bereich im virtuellen Adreßraum, der nur einmal in einem Rechner real vorhanden ist. Durch entsprechende Vorrangsteuerwerke wird der Zugriff der drei Rechner auf diesen gemeinsamen Speicher koordiniert. Bei der Generierung der Softwaresysteme muß nun dafür gesorgt werden, daß durch entsprechende Bindeanweisungen die teilsystemglobalen Daten im gemeinsamen Speicher abgelegt werden.

- Das Führungs-Teilsystem

Es empfängt Luftlagen, die von den zugeordneten Sensorteilsystemen per Datenfunk übertragen werden. Diese Luftlagen werden

ausgewertet. Aus der aktuellen Lage wird die Bedrohungssituation ermittelt, dann erfolgt die Zielzuweisung an die zugeordneten Effektoren. Auf Grund der jeweiligen Rückmeldungen der Effektoren kennt das Führungssystem sowohl die Position, als auch den Zustand der Effektoren und kann diese Größen bei der Zielzuweisung berücksichtigen.

- Die Waffen-Teilsysteme

Sie erhalten von ihrem jeweiligen Führungssystem die Aufträge und führen diese selbstständig aus. Erst nach Ausführung eines Auftrages erfolgt die Ergebnisrückmeldung an das Führungssystem.

Die Kommunikation der Teilsysteme erfolgt in verschiedenen Funkkreisen über Datenfunk direkt zwischen den in PEARL programmierten Rechnern. Dabei sind entsprechende Mechanismen vorzusehen, um Kollisionsfälle, dh. gleichzeitiges Senden mehrerer Rechner im selben Funkkreis zu vermeiden oder aufzulösen. Außerdem ist sicherzustellen, daß die relativ kurzlebigen Luftlagedaten rechtzeitig übertragen werden.

Die Systemstruktur wird beispielhaft in Abbildung 2 dargestellt.

4. Entwicklungshilfsmittel für die Implementierungssprache PEARL

Im Rahmen der vorgenannten Projekte wurden zwei Entwicklungshilfsmittel implementiert, die heute für jede Implementierungssprache vorhanden sein sollten.

- Ein quellbezogenes Testsystem
  Dieses Testsystem QPTS auf dem operationellen Rechner, das schon früher [1] ausführlich dargestellt wurde, sei hier nur genannt.

- Ein Programm-Analysator zur Messung der C0- bzw. C1- Testüberwachung.

Während das Testsystem primär die Fehlersuche unterstützt, dient der Analysator zum Nachweis, daß ein Programm mit verschiedenen Datensätzen ausgeführt wurde und daß dabei korrekte Ergebnisse erzielt wurden.

Häufig werden die Programme hierzu manuell so modifiziert, daß an bestimmten Stellen Testausdrucke erfolgen. Die Methode ist jedoch ziemlich aufwendig und fehleranfällig. Sie ist praktisch nicht mehr manuell durchführbar, wenn gefordert wird, daß alle Programmzweige überdeckt werden. Da die

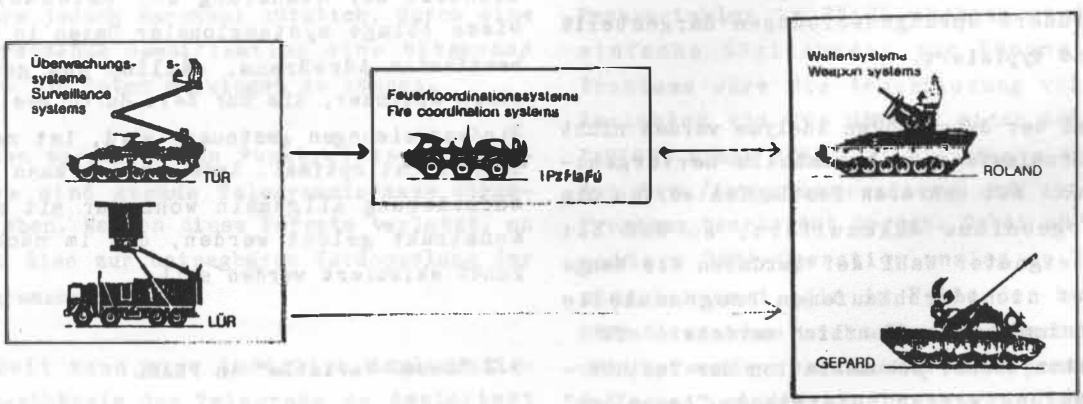


Abb. 2 System-Übersicht HFlaAFüSys

Qualität eines Tests weniger von der Menge der Testdaten abhängt, als von der Verteilung der Testdaten, ist eine so umfassende Aussage notwendig, um die Qualität des Tests zu sichern. Es muß sichergestellt werden, daß bei den Tests auch Fehlerbehandlungen, Sonderfallbehandlungen und andere nicht regelmäßig durchlaufene Programmteile angesprochen werden.
Es muß ein Instrument verfügbar sein, das sowohl Aussagen über die Verteilung der verwendeten Testdaten als auch über die noch nicht angesprochenen Programmteile macht.

Das hier entwickelte Tool [2] gibt detaillierte, aber auch zusammenfassende Auskunft sowohl über den statischen Programmaufbau, als auch über das dynamische Verhalten während des Testlaufs. Da es sich hier um ein Meßinstrument handelt, können intuitive (Fehl-) Einschätzungen des Testers und des Entwicklers weitgehend vermieden werden.

Die Arbeitsweise ist wie folgt:

1.) Der zu testende Modul wird automatisch instrumentiert, dh. die o.g. Kontrollstatements werden automatisch an den entsprechenden Stellen eingefügt.

2.) Es wird dann ein Testlauf mit den vorbereiteten Test-Daten durchgeführt, wobei statische und dynamische Ergebnisse in Testprotokollen abgelegt werden.
Bei der statischen Analyse werden insbesondere Sprunganweisungen dargestellt und typisiert.

Bei der dynamischen Analyse werden nicht durchlaufene Programmteile hervorgehoben. Bei mehreren Testläufen werden die Ergebnisse akkumuliert, so daß bei geeigneter Wahl der Tetdaten die Menge der nichtdurchlaufenen Programmteile abnimmt und hoffentlich verschwindet.
Neben dieser Dokumentation der Testüberdeckung werden auftretende "Anomalien" dokumentiert. Solche sind beispielsweise nicht programmierte CASE-Alternativen oder nur einmal durchlaufene Schleifenkörper.

5. Erweiterungswünsche an PEARL

Im Rahmen des obengenannten Projektes ergaben sich Erweiterungswünsche an die Sprache PEARL. Diese Wünsche mußten inzwischen projektspezifisch gelöst werden. Im Klartext heißt das, daß zumindest teilweise auf Assemblerprogrammierung ausgewichen werden mußte. Darunter leiden Portabilität und Lesbarkeit der PEARl-Programme.

Die hier genannten Wünsche und insbesondere die Lösungen sind natürlich system- und implementierungsabhängig. Die gewünschten Funktionen erscheinen jedoch so allgemein, daß daraus Spracherweiterungen resultieren könnten.

5.1 Mehr-Rechner-PEARL

Für die in HFlaAFüSys eingesetzten, speicherbusgekoppelten Mehrrechnersysteme sind natürlich Koordinationsmittel notwendig. Prinzipiell würde es hierfür ausreichen, neben den üblichen Variablen von Typ SEMA noch "systemglobale" SEMAs verfügbar zu haben. Zur Zeit kann der Zugriff nur über Assemblerprozeduren koordiniert werden, die mit speziellen nicht unterbrechbaren "Test- und Set-Befehlen" Überholvorgänge durch andere Rechner während der Koordinationsoperationen ausschließen. Die Vorschläge zu Mehrrechner-PEARL [3] sind hier dringend geboten. Die Struktur dieser speziellen Systeme erfordert darüberhinaus die Möglichkeit der Steuerung der Datenablage. Diese Ablage systemglobaler Daten in einem bestimmten Adreßraum, nämlich dem gemeinsamen Speicher, die zur Zeit durch geeignete Bindeanweisungen gesteuert wird, ist natürlich nicht optimal. Andererseits kann diese Anforderung allgemein wohl nur mit einem Konstrukt gelöst werden, das im nächsten Punkt skizziert werden soll.

5.2 "Based-Variable" in PEARL

Die Datenorganisation in PEARL sieht zwar komplexe Datentypen vor, aber sie bietet nicht die Möglichkeit, die Deklaration von Datenstrukturen von der konkreten Realisie-

rung im Speicher, das heißt Adressvergabe, zu trennen. Oft werden, besonders bei Datenfernübertragungssystemen, Eingangspuffer dynamisch vom Betriebssystem vergeben, und mit entsprechener Information gefüllt, dem PEARL-Programm zur Bearbeitung übergeben. Obwohl die bekannte Meldungsstruktur in diesem Puffer bereits vorhanden ist, muß meist aufwendig umgespeichert werden, bevor die Daten strukturiert verarbeitet werden können. Es kann nicht verschwiegen werden, daß diese Art von Daten im allgemeinen Fall zu Fehlern führen kann, die insbesondere durch unterschiedliche Lebensdauern und Gültigkeitsbereichen von Referenzen und Daten bedingt sind. Da die Speicherverwaltung in PEARL-Systemen ohnehin nicht trivial ist, sollte hier eine Lösung möglich sein.

Die bei uns gewählte Lösung geht den umgekehrten Weg, indem sie in PEARL deklarierte und angelegte Daten der Pufferverwaltung des Betriebssystems "unterschiebt".

5.3 Erzeugung von bitgenauen Datenstrukturen

In höheren Programmiersprachen ist es allgemein üblich, die Datendarstellung vor dem Programmierer zu verbergen und damit dem Compiler Optimierungsmöglichkeiten zu geben und dem Programmierer Fehlermöglichkeiten zu nehmen. Im allgemeinen ist diese Strategie sicher richtig.

Es wäre jedoch manchmal nützlich, durch eine entsprechende Spezifikation eine bitgenaue Ablage von Daten erzwingen zu können.

Bei den verschiedenen Funkkreisen in HFla-AFüSys sind genaue Telegrammformate vorgeschrieben. Werden diese Formate verletzt, so führt dies zur untragbaren Verdoppelung der Telegrammanzahl.

Zur Zeit kann hier lediglich implementierungs-abhängig das Telegramm so deklariert werden, daß der jeweilige Compiler die gewünschte Struktur ablegt. Daß diese Strategie gegen Compileränderungen und Programmübertragungen sehr fehleranfällig ist, liegt auf der Hand.

5.4 Botschaftsmechanismen

PEARL sieht vom Taskkonzept her keine Möglichkeit vor, Daten von einer Task an eine andere zu übergeben. Da in praktisch allen Realzeitbetriebssystemen Botschaftsdienste vorhanden sind, liegt es nahe, diese Dienste auf Prozedurebene implementierungsabhängig dem PEARL-Programmierer zugänglich zu machen. Das Hauptproblem bei dieser Lösung besteht jedoch darin, daß beim Empfang von Botschaften Wartezustände auftreten können, die nicht der üblichen PEARL-Taskverwaltung unterliegen.

Außerdem werden diese Botschaftsmechanismen häufig auch von der PEARL-Taskverwaltung implizit benutzt. Dadurch entstehen spezielle Beschränkungen, so daß diese Prozeduren nicht beliebig mit PEARL-Taskoperation gemischt werden können.

5.5 Asynchrone Ein-/Ausgabevorgänge

Die gesamte Parallelverarbeitung steckt in PEARL im Taskmodell. Bei Systemen wie z.B. WIAS, wo eine Vielzahl von Geräten quasiparallel betrieben werden soll, verbietet es sich jedoch für jedes Gerät eine eigene PEARl-Task einzurichten. Auch die dynamische Zuordnung von Tasks aus einem Taskpool zu den Geräten ist nicht machbar, da keine Taskvariablen in PEARL möglich sind. Eine einfache Möglichkeit zur Lösung dieses Problems wäre die Assoziierung von einer Variablen vom Typ SEMA zu einer Asynchron-DATION. Diese SEMA-Variable könnte dann von der Ein-/Ausgabeverwaltung und vom PEARL-Programm bearbeitet werden. Dabei wäre eine weitere SEMA-Operation analog zu REQUEST wünschenswert, die im Falle der belegten SEMA-Variablen nicht in einen Wartezustand führt, sondern mit einer entsprechenden Meldung das Weiterarbeiten des Programms ermöglicht. In Ermangelung dieser Lösung blieb uns nichts anderes übrig, als das starke Sprachmittel des SVC-Befehls (-Betriebssystem-Aufruf) als PEARL-Intrinsic zu benutzen.

5.6 Sprachmittel zur Bildschirmprogrammierung

Nachdem es verschiedene, praktisch inhaltsgleiche Normen zur Steuerung von Bildschirmen gibt [4], wäre es wünschenswert, die Darstellungsattribute wie z.B. Farben, Schriftarten etc. symbolisch angeben zu können. Die Verpackung von CSi-Steuerzeichenfolgen in geeignete, benannte Konstanten und deren Verwendung in E/A-Listen ist mühsam und undurchsichtig. Besser wäre die Einführung von entsprechenden CONTROLs, die dann von der E/A-Verwaltung in die entsprechende Steuerzeichenfolgen umgesetzt werden könnten. Eine spezielle Lösung dieses Problems, insbesondere unter Berücksichtigung der oben erwähnten Mehrfachausgabe gleicher Bilder, stellt das sogenannte Hauptspeicher-DATION [5] dar. Dabei werden die Konvertierungsleistungen der PEARL-Ein-/Ausgabe nicht in einen Gerätepuffer, sondern in ein im PEARL-Programm deklariertes Speicherstück wirksam. Über Prozeduren werden Informationen über die DATION-Verwaltung dem PEARL-Programm zugänglich gemacht. Damit können nach der Konvertierung und vor dem Transfer zum Gerät noch Manipulationen am auszugebenden String vom PEARL-Programm aus durchgeführt werden.

5.7 Datenbankzugriffe in PEARL

Wenn auch in keinem der beiden vorgestellten Projekte eine Datenbank eingesetzt wird, besteht doch häufig der Bedarf, mit Sprachmitteln Datenbanken zu bearbeiten. Spezielle Prozeduren sind hier nur ein Notbehelf und nicht dem Sprachkonzept angepaßt. Ich möchte hier daran erinnern, daß eine so viel geschmähte Sprache wie COBOL diese Elemente schon seit mehr als zehn Jahren enthält.

6. Literatur

[1] K. Mangold: Ein quellbezogenes Testsystem für PEARL auf einem Prozeßrechner.
PEARL-Rundschau, Heft 6, Band 2, Dez. '81

[2] -: SQUIRREL ein Test- und Dokumentations-Werkzeug zur Messung der Co- und C1-Testüberdeckung, Benutzerhandbuch, IDAS GmbH, Limburg, 1983

[3] H.Steusloff: Nutzbarkeit von PEARL für Mehrrechnersysteme; in: Konzepte moderner Programmiersprachen am Beispiel von Ada und PEARL, Mannheim 1984

[4] DIN 66254, Entwurf, Zusätzliche Steuerfunktionen für zeichendarstellende Geräte, Berlin, Mai 1983

[5] -: PEARL für Computersysteme ATM80-10, ATM80-30
Benutzerhandbuch Compiler
" Laufzeitorganisation
" Quellbezogenes Testsystem
ATM GmbH,Konstanz, 1984

Verfasser: K. Mangold
AEG-Telefunken
A16/E-KN
Postfach 2154
7750 Konstanz