# PDV Berichte

**Projekt Prozeßlenkung mit DV-Anlagen**

KFK-PDV 110

## Tagungsband zum Aussprachetag PEARL in Augsburg

März 1977

GESELLSCHAFT FÜR KERNFORSCHUNG MBH KARLSRUHE

## PDV-Berichte

Die Gesellschaft für Kernforschung mbH koordiniert und betreut im Auftrag des Bundesministers für Forschung und Technologie das im Rahmen der Datenverarbeitungsprogramme der Bundesregierung geförderte Projekt Prozeßlenkung mit Datenverarbeitungsanlagen (PDV). Hierbei arbeitet sie eng mit Unternehmen der gewerblichen Wirtschaft und Einrichtungen der öffentlichen Hand zusammen. Als Projektträger gibt sie die Schriftenreihe PDV-Berichte heraus. Darin werden Entwicklungsunterlagen zur Verfügung gestellt, die einer raschen und breiteren Anwendung der Datenverarbeitung in der Prozeßlenkung dienen sollen.

Der vorliegende Bericht dokumentiert Kenntnisse und Ergebnisse, die im Projekt PDV gewonnen wurden.



Druck und Verbreitung:
Gesellschaft für Kernforschung mbH
7500 Karlsruhe 1, Postfach 3640
Printed in Western-Germany

TAGUNGSBAND ZUM

# Aussprachetag

# PEARL

9. MÄRZ 1977
AUGSBURG, KONGRESSHALLE
M O Z A R T S A A L

Veranstalter:

VDI/VDE- GESELLSCHAFT
MESS- UND REGELUNGSTECHNIK (GMR)
Ausschuß 4.4 "Echtzeitprogrammiersprachen"

GESELLSCHAFT FÜR KERNFORSCHUNG (GfK)
Projekt "PDV"
(Prozeßlenkung mit Datenverarbeitungsanlagen)
des BMFT

# Programm des Aussprachetages:

**Wissenschaftliche Gesamtleitung und Diskussionsleitung am Vormittag**
Professor Dr.-Ing. R. Lauber, Stuttgart

---

**Programmkomitee**
K.-F. Bamberger, Karlsruhe
P. Elzer, Erlangen
J. Heger, Mannheim
W. Heusler, Berlin
M. A. Kaaz, Düsseldorf
R. Lauber, Stuttgart
T. Martin, Karlsruhe

---

8.30 **PEARL – Einführung und Übersicht**
(Tutorial für Teilnehmer, die sich vor der Aussprache in PEARL einführen lassen möchten)
R. Lauber, Stuttgart

9.45 Kaffeepause

**Offizieller Beginn des Aussprachetages**

10.00 **Begrüßung:** R. Bertuleit, Bonn

**Die Förderung von PEARL im Projekt PDV des 2. und 3. DV-Programmes des BMFT**
(Historie – Stand – Zukunft – Standardisierung)
T. Martin, Karlsruhe

**Diskussion**

11.30 **PEARL im Vergleich mit anderen Echtzeitsprachen**
A. Schwald, R. Baumann, München

**Diskussion**

12.30 **Mittagspause**

14.00 **Diskussionsleitung**
H. Eckert, Karlsruhe

**Erfahrungen mit bisherigen PEARL-Subsets**
(an Modellprozessen und im industriellen Einsatz)
Referenten: P. Elzer, Erlangen, G. Müller, Brühl,
H. Mittendorfer, Karlsruhe,
W. D. Blaum, Frankfurt/M.,
T. Pfeifer, Aachen

**Diskussion** mit den Referenten

Ende etwa 15.30 Uhr

Inhalt

Dieser Tagungsband enthält die Langfassung folgender auf dem Aussprachetag gehaltener Vorträge:



---

+ Der Beitrag wurde zu spät eingereicht, um noch in den Tagungsband aufgenommen werden zu können

# Die Förderung von PEARL im Projekt PDV (Prozeßlenkung mit Datenverarbeitungsanlagen) des 2. und 3. DV-Programms des BMFT

von T. Martin, Karlsruhe

Inhalt: Seite

# Die Förderung von PEARL im Projekt PDV (Prozeßlenkung mit Datenverarbeitungsanlagen) des 2. und 3. DV-Programms des BMFT

von T. Martin, Karlsruhe

## Stellenwert von PEARL im Projekt PDV

Ziel der Förderungsmaßnahmen im Rahmen des 2. und 3. DV-Programms der Bundesregierung ist, die Leistungskraft der Wirtschaft und der öffentlichen Dienstleistungen unseres Landes zu stärken. Das bedeutet für das Projekt PDV (Prozeßlenkung mit DV-Anlagen) als industrie- und anwendungsbezogene Teilmaßnahme der DV-Programme, daß die erforderlichen Hilfsmittel und Werkzeuge für den Einsatz der Datenverarbeitung zur Steuerung und Automation technischer Prozesse entwickelt und in der Praxis erprobt und demonstriert werden, sodaß alle erforderlichen Voraussetzungen für eine Anwendung und Nutzung dieser wichtigen Schlüsseltechnologie in weiten Bereichen der Technik gegeben sind.

Die Kosten für die Produktion und die Pflege von Anwendersoftware haben bekanntlich einen wachsenden relativen Anteil an den Gesamtkosten eines Automatisierungssystems mit Prozeßrechnern. Ein Hauptbestreben von PDV muß es daher sein, diejenigen software-technologischen Mittel und Verfahren zu fördern, die der Rationalisierung der Anwendersoftwareproduktion dienen.

An erster Stelle ist hier die Entwicklung der universellen höheren Echtzeitsprache PEARL (Process and Experiment Automation Realtime Language) für Prozeßrechner zu nennen.

Man hofft, mit dieser Programmiersprache zukünftig in der Bundesrepublik die Kosten für Anwendersoftware gegenüber den Kosten, die bei Verwenden von maschinenabhängigen Sprachen anfallen würden, für die meisten Prozeßrechneranwendungen drastisch (um 10 bis 50 %) senken zu können.

Die Entwicklung von PEARL ist, was das Fördervolumen anbetrifft, das größte Gemeinschaftsvorhaben innerhalb des Projektes. Daß sich die Ausgabe von Steuergeldern für diesen Zweck voraussichtlich lohnen wird, soll folgende Zahlenbetrachtung zeigen.

In der Bundesrepublik werden jährlich Investitionsgüter im Wert von 26o Mrd. DM produziert; der Anteil der Automatisierungseinrichtungen daran beträgt ca. 13 Mrd. DM (5 %). Ein erheblicher, in Zukunft noch wesentlich wachsender Anteil hiervon ist Prozeßlenkungssoftware - Größenordnung heute 5oo Mio. DM, in absehbarer Zeit 1 Mrd. DM (8 %) /26/. Wenn es gelänge, mit PEARL den Aufwand für diese Prozeßlenkungssoftware drastisch zu senken (auf 9o bis 5o %), so ergäbe das eine Einsparung in der Größenordnung von einigen 1oo Mio. DM. Demgegenüber erscheint der zu erwartende Aufwand für die Entwicklung von PEARL seit 1972 in Höhe von ca. 5o Mio. DM Gesamtentwicklungskosten - davon etwa 3o Mio. DM Fördervolumen (Bild 1) - als vertretbar.

## Charakterisierung der Echtzeitsprache PEARL

Wegen der Vielfalt der Programmiersprachentypen ist es wichtig zu betonen, um welchen Sprachtyp es sich bei der Programmiersprache PEARL handelt.

PEARL wird zu recht als universelle höhere Echtzeitprogrammiersprache bezeichnet.

U n i v e r s e l l ist PEARL im doppelten Sinn: Sie ist zum einen für ein sehr breites Anwendungsspektrum, zum anderen praktisch für alle kommerziellen Prozeßrechner geeignet.

Der Trend zu h ö h e r e n , d. h. maschinenunabhängigen, Programmiersprachen ist allgemein akzeptiert. Sie versprechen eben verminderte Programmherstellungskosten. Die Gründe dafür, daß man sich in der Prozeßdatenverarbeitung so lange mit maschinenabhängigen Sprachen behelfen mußte, sind in der früher vorhandenen Verschiedenartigkeit der Rechner und in einigen Besonderheiten der Prozeßrechnersysteme zu suchen, die programmtechnisch schwierig zu beherrschen sind. Diese Besonderheiten sind - kurz gesagt - folgende:

- Die Programme müssen über spezielle Geräte (Prozeßperipherie) mit dem technischen Prozeß verkehren, die für jedes System individuelle Adressen und Namen haben.

- Die Programme müssen sich selbst schritthaltend mit dem Zeitlauf und den Ereignissen im Prozeß steuern können.

Nur den Programmiersprachen, die Formulierungsmittel für diese Aufgaben im Sprachkern enthalten, sollte man die sogenannten E c h t z e i t fähigkeiten zubilligen, d. h. nur diese sollte man Echtzeitprogrammiersprachen nennen. Die verschiedenen Typen höherer Programmiersprachen für Prozeßrechner sind im Bild 2 jeweils mit Beispielen dargestellt. Man erkennt die Einteilung in anwendungsspezifische und universelle Programmiersprachen /1/.

## Historie der PEARL-Entwicklung

### Vorgeschichte

Seit etwa Mitte der sechziger Jahre gibt es Versuche, höhere Programmiersprachen vornehmlich für spezielle Anwendungen in der Echtzeitprogrammierung zu schaffen. Solche in der Praxis eingesetzten Sprachen sind zumeist FORTRAN-Derivate. Vielfach behilft man sich auch mit vorgefertigten Programmen, Softwarepaketen, die zum Teil durch sprachähnliche Mittel individuell erweitert werden können (MADAM, ARSI etc., siehe Bild 2).

Gegen Ende der sechziger Jahre tauchen dann vor allem in Europa Sprachentwürfe mit universellem Charakter auf, und zwar u. a. RTL/2 in England /2/, PROCOL in Frankreich /3/ und PAS1 /4/ und PEARL in der Bundesrepublik.

Der PEARL-Arbeitskreis tritt im Herbst 1969 im Rahmen der Studiengruppe Nuklearelektronik, einer vom BMBW finanziell geförderten Arbeitsgruppe zur Entwicklung hochwertiger Meßelektronik, zusammen. Sein Motiv ist die Definition einer höheren Echtzeitprogrammiersprache für leichtere Programmierung und als Basis für den Austausch von Prozeß- und Experimentprogrammen. Auch die Arbeit des PEARL-Arbeitskreises wird vom BMBW - bis zur Aufnahme in das PDV-Projekt des BMFT 1972 - finanziell unterstützt, vor allem durch ein am Physikalischen Institut III der Universität Erlangen-Nürnberg laufendes Forschungsvorhaben.

Der PEARL-Arbeitskreis setzt sich von Anfang an aus Anwendern, Herstellerfirmen, Softwarehäusern und Instituten zusammen. Ebenfalls gehört dem Arbeitskreis von jeher ein

Vertreter des Ausschusses Programmiertechnik des VDI/VDE an. In dieser bezüglich der Interessen ausbalancierten Runde entsteht im Herbst 197o der PEARL-Rohentwurf /5/ [1].

## Förderung von PEARL im Projekt PDV

Anfang 1972 nimmt das Projekt PDV PEARL unter seine Fittiche. Mehrere Prozeßrechnerhersteller haben ihr Interesse bekundet, PEARL auf ihren Rechnern zu implementieren. PDV unterstützt diese Pläne und fördert ab 1973 mehrere Vorhaben zur Entwicklung von Compilern und Betriebssystemanpassungen.

Es zeigt sich sehr bald, daß als Voraussetzung für die Sprachentwicklung grundlegende Probleme der Übersetzungstechnik und der Betriebssystemstrukturierung zu lösen sind. PDV fördert daher mehrere diesbezügliche Vorhaben bei wissenschaftlichen Instituten und Softwarehäusern.

Beispiele von Problemlösungen, die im Projekt PDV im Rahmen einer übergreifenden Softwaretechnologie für Prozeßrechner gefunden wurden, sind:

- Spezifikation einer maschinenunabhängigen Zwischensprache beim zweistufigen Übersetzungsvorgang /6/;
- Automatisches Verfahren zur Optimierung des Programmcodes beim Übersetzen /7/;
- Studie über die Struktur eines generellen Betriebssystems für PEARL /8/;
- Anpassung eines Betriebssystems an unterschiedliche Aufgabenprofile /9/.

---

[1] Der Entwurf basiert auf drei unabhängig voneinander entstandenen Vorstudien von V. Haase, H. Mittendorf und P. Elzer.

Anfang 1973 gibt PDV als Ergebnis der Arbeit des PEARL-Arbeitskreises die vorläufige Sprachdefinition von PEARL heraus /10, 11/. Im Laufe 1973 wird die äußere Form (Syntax) der Sprache noch näher an PL/1 angepaßt /12, 13/, um ihre internationalen Chancen zu erhöhen.

Ab Mitte 1973 bildet der PEARL-Arbeitskreis den Ausschuß 4.4 "Echtzeitprogrammiersprachen" des Bereiches 4 "Technik der Prozeßrechner" der VDI/VDE-Gesellschaft Meß- und Regeltechnik (GMR). Gleichzeitig gründet sich unter Vorsitz der Projektleitung PDV - gewissermaßen als Ad-hoc-Arbeitskreis des Ausschusses 4.4 - der PEARL-Subset-Arbeitskreis (SAK). Aufgabe des SAK ist die Implementierung von PEARL auf möglichst vielen Prozeßrechnern und die praxisnahe Erprobung der vorläufig definierten Sprache. (Unter Subset versteht man die Teilmenge einer Programmiersprache, die auf einem Rechner implementiert wird.)

Im SAK sind acht Institutionen als Vollmitglieder - das sind alle Implementatoren und die beiden betreuenden Institutionen - und sieben weitere Institutionen als mitarbeitende Gäste vereinigt (siehe im einzelnen in Tabelle 1).

Bei dem Implementierungen wird in zwei Stufen vorgegangen. Stufe 1: Zwei frühe Implementationen von PEARL nach der vorläufigen Definition von 1973, um damit Erfahrungen zu sammeln. Stufe 2: Anpassung aller Implementationen an die verbesserte und endgültige PEARL-Definition, für die in 1976 Redaktionsschluß ist.

Bei den beiden frühen Implementationen handelt es sich zum einen um den BBC-Subset, genannt PAS2, für das System DP1000, mit dem industrielle Erfahrungen seit 1974 gesammelt werden /14/; zum anderen um den ASME-Stufe 1-Subset, mit dem seit Anfang 1975 auf den Rechnern Siemens 306 und 404/3 und AEG 60-10 und 60-50 PEARL anhand von Modellprozessen im Labor experimentell erprobt wird /15, 16/. Die zahlreichen PEARL-Implementationen und Begleituntersuchungen der Arbeitsgemeinschaft ASME (siehe Tabelle 1) tragen auch wesentlich zur Verbreitung von PEARL bei.

Die in Stufe 1 gewonnenen Erfahrungen münden in Verbesserungsvorschläge für die Sprache PEARL. Bis zum Redaktionsschluß in 1976 werden solche Verbesserungen von den Implementatoren einvernehmlich in das endgültige PEARL aufgenommen.

Dem SAK gehören Vertreter eines weiteren Arbeitskreises an, dem Arbeitskreis PfK (PEARL für Kleinrechner) [1], der seit Ende 1973 besteht. Seine Aufgaben sind:

- Definition eines PEARL-Subset für Kleinrechner
- Implementation dieses Subset bei den zum Arbeitskreis gehörenden Rechnerherstellern
- Erprobung dieses Subset in Pilotanwendungen

Die diesem Arbeitskreis angehörenden Institutionen, deren Vorhaben seit 1974 von PDV gefördert und betreut werden, sind in Tabelle 2 zusammengestellt.

---

[1] PfK ist wie der SAK ein Ad-hoc-Arbeitskreis des Ausschusses 4.4 der VDI/VDE-GMR.

Ein wesentlicher Zweck des Arbeitskreises ist, daß die beteiligten Rechnersteller als Unternehmen der mittelständischen Industrie kostengünstig zu einem PEARL-Compiler gelangen. Das geschieht hier dadurch, daß der von der ASME entwickelte portable Compiler-Oberteil, welcher PEARL in die Zwischensprache CIMIC übersetzt, verwendet wird. Die beteiligten Institute leisten durch Entwicklung von Codegeneratoren, die die Zwischensprache in die jeweilige Maschinensprache der Zielrechner weiterübersetzen (Bild 3), den Herstellern teilweise Zuarbeit. Den Herstellern bleibt es überlassen, die übernommenen Programme an ihre Systemsoftware - insbesondere an das Betriebssystem - anzupassen.

Schließlich gelingt in 1976 im Rahmen aller im Projekt PDV integrierten PEARL-Aktivitäten neben der endgültigen Definition der vollen Sprache (Full-PEARL) auch die Festlegung eines gemeinsamen PEARL-Subset für alle Implementatoren, Basis-PEARL genannt. Dieser ist identisch mit dem PEARL für Kleinrechner.

Nach gegenwärtigem Stand implementieren lediglich die Firmen AEG, BBC und MBP einen über Basis-PEARL hinaus gehenden Umfang von PEARL.

Dokumentation von PEARL

Der SAK hat die endgültige Definition von Full-PEARL in 25 (manchmal als mehrtägige Klausur abgehaltenen) Arbeitssitzungen erarbeitet. Diese Definition ist inzwischen in ihrer Syntax und in ihrer Semantik genau spezifiziert worden. Sie wird von der Gesellschaft für Kernforschung, Karlsruhe, als PDV-Bericht "Full PEARL Language Description" in Englisch herausgegeben.

Außerdem wird die Sprachbeschreibung von Basis-PEARL in ähnlicher Form ebenfalls von der Gesellschaft für Kernforschung als PDV-Bericht herausgegeben.

Neben diesen generellen, für Alle verbindlichen Sprachbeschreibungen entstehen PEARL-Handbücher bei den Implementatoren.

Stand der PEARL-Implementationen

Die Institutionen, die PEARL innerhalb des Projektes PDV implementieren und die Prozeßrechner, auf denen der jeweilige Compiler läuft, sind nachfolgend aufgeführt (siehe ausführliche Namen in den Tabellen 1 und 2). Alle diese PEARL-Compiler sollen in 1977 fertig entwickelt sein.

| Institution | Prozeßrechner | Bemerkungen |
|---|---|---|
| AEG | 8o-2o | Anfang 1978 ebenso für 8o-4o/6o |
| BBC | DP 1ooo | /17/ |
| Siemens | 33o | /18/ |
| Dietz | Mincal 621 | ) Compiler-Oberteil von der |
| Krantz | Mulby 3 | ) ASME (ausgeliefert durch |
| Krupp-Atlas | EPR 11oo | ) Fa. GPP) /19/ |
| MBP | HP 3ooo | im Rechenzentrum des IRT, München |
| MBP | Siemens 4o4/3 | für die DFVLR, Oberpfaffenhofen |

Einige dieser Compiler können auch auf Gastrechnern ablaufen.

Die Implementation der Fa. MBP ist ein portables PEARL-System, dessen Compiler wie bei der ASME auf dem zweistufigen Übersetzungskonzept beruht. Die beiden bisher ausgelieferten Systeme tragen PEARL in zwei wichtige, von PDV geförderte Pilotanwendungsbereiche hinein: Am IRT (Institut für Rundfunktechnik), München, sollen für die Rundfunk- und Fernsehanstalten in der Bundesrepublik zentral Automationsprogramme mit PEARL entwickelt werden; an der DFVLR (Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt), Oberpfaffenhofen, werden mit PEARL Experimentsteuerungen und -auswertungen programmiert, die für das Spacelab bestimmt sind.

PEARL wird im Projekt PDV auch bei der Lösung zahlreicher weiterer anwendungsorientierter Automationsverfahren eingesetzt.

Schließlich sei das wachsende Interesse der öffentlichen Auftraggeber an PEARL erwähnt. Beispielsweise hat das BMVg (Bundesministerium für Verteidigung) großes Interesse an PEARL bekundet, besonders für Anwendungen in der Luftfahrt (Avionics) /20/.

Ein weiteres Beispiel ist das Spacelab-Projekt. Die deutsche Seite dieses Projektes, vertreten durch das BMFT (Abteilung 5) und dem Projektträger DFVLR, sieht für Experimente, die in der Bundesrepublik durchgeführt werden, vor, PEARL für die Programmierung des Experiment-Datenmanagements (Erfassung, Verarbeitung, Steuerung) zu verwenden. Darüberhinaus sind Gespräche mit der Europäischen Raumfahrtbehörde über die Verwendung von PEARL für die Nutzung von Spacelab im europäischen Rahmen im Gange.

Internationale Sprachentwicklungen für Prozeßrechner

Die Einführung von PEARL in der Bundesrepublik als einheitliche Echtzeitprogrammiersprache in einem Land steht derzeit in der Welt einzig da. In anderen Ländern sind die Verhältnisse unübersicht-

licher: In England konkurriert RTL/2 gegen CORAL 66 /21/, die der nationale De-facto-Standard ist, obwohl es sich dabei nicht um eine zukunftsweisende Sprache mit Echtzeiteigenschaften handelt; in Frankreich bleibt PROCOL gegen LTR /22/, die von der Armee gefördert wurde, auf der Strecke; in USA gibt es die alte Bestrebung, PL/1 um Echtzeiteigenschaften zu bereichern. Ohne Rücksicht auf letztere ist kürzlich im amerikanischen Militärbereich ein Projekt mit dem Ziel installiert worden, eine moderne Echtzeitsprache schnellstmöglich zu definieren, zu implementieren und zu standardisieren.

Daneben gibt es vor allem in USA, aber auch in Europa, noch Interessenten für Prozeß-FORTRAN (dem für Prozeßrechnerzwecke um einige Routinen erweiterten FORTRAN), die diese Sprache in der Zeit des Übergangs zu Echtzeitsprachen als Zwischenlösung benutzen möchten.

Leider gibt es noch keine Einigung auf ein einheitliches Prozeß-FORTRAN, was die Befürchtung nährt, daß Prozeß-FORTRAN seine Rolle als nützliche weltweite Zwischenlösung nicht wird spielen können. Für die Erweiterungsroutinen gibt es u.a. folgende drei Vorschläge:

- der amerikanische ANSI-Standard "Industrial Computer System FORTRAN";

- der deutsche Vorschlag "Prozeß-FORTRAN 75", der über die amerikanische Norm hinausgeht, aber als notwendiges Minimum angesehen wird (vorgeschlagen vom Ad-hoc-Arbeitskreis Prozeß-FORTRAN des Ausschusses 4.2 der VDI/VDE-GMR).

- Die europäischen Interessenten haben im Rahmen des Purdue Workshop Europe das "Technical Committee 1 on Realtime FORTRAN" gegründet, welches Vorstellungen entwickelt, die wesentlich über die beiden anderen Vorschläge hinausgehen.

Allgemeine Standardisierungsbestrebungen

Trotz (oder wegen) der geschilderten Programmiersprachenvielfalt gibt es seit langem die Bestrebung nach einer einheitlichen Programmierung von Prozeßrechnern. Sie knüpfte in Europa an den 1969 gegründeten "Purdue Workshop on Standardization of Industrial Computer Languages" an der Purdue University, Lafayette, Indiana, USA, an.

Die Europäer gründeten 1974 eine eigene Sektion, den Purdue Workshop Europe, um ihre eigenen Gesichtspunkte besser berücksichtigen zu können. Leider gibt es allerdings bis heute keine europäische Normungsbehörde mit einem Gewicht wie etwa ANSI (American National Standards Institute).

Innerhalb des Workshop beschäftigt sich das "Long Term Procedural Language (LTPL) Committee" mit der Standardisierung künftiger Echtzeitsprachen für Prozeßrechner. Durch die Förderung seitens PDV (und neuerdings auch der Europäischen Kommission) war eine intensive Mitarbeit deutscher Vertreter im LTPL-Komitee gesichert. (Diese stellen den Vorsitzenden sowohl im internationalen wie im europäischen Komitee.)

Zwei große Erfolge des europäischen LTPL-Komitees sind

- das Aufstellen von technischen Anforderungen (functional requirements) an eine Echtzeitsprache (die PEARL gut erfüllt);

- ein detailliertes Vergleichen existierender Sprachen /23/ (bei dem PEARL als fortschrittlichster Kandidat angesehen wird).

Dennoch konnte sich der Workshop auf keine existierende Sprache als Kandidat für einen weltweiten Standard einigen. Deshalb entwickelte er die Philosophie des "melt of best features", die besagen will, daß man bei der Definition einer neuen Echtzeitsprache die besten Eigenschaften der existierenden Sprachen - und damit zwangsläufig die meisten von PEARL - zusammenfassen sollte.

Augenblicklich startet die Europäische Kommission, Brüssel ein europäisches Echtzeitsprachenprojekt. In den zunächst vorgesehenen ersten zwei Jahren soll eine international standardisierbare Echtzeitsprache für Prozeßrechner definiert werden.

Standardisierung von PEARL

In der Bundesrepublik wurde Ende 1973 der Arbeitskreis 5.8 "Programmiersprachen zur Steuerung technischer Prozesse" des FNI (Fachnormenausschuß Informationsverarbeitung) im DIN mit der ausdrücklichen Aufgabe eingesetzt, für eine normgerechte Beschreibung von PEARL zu sorgen. Die vorläufige Sprachdefinition von 1973 /1o/ erfüllte noch nicht die Anforderungen.

Um die Grundlagen für eine normfähige Beschreibung von PEARL zu legen, fördert PDV seit Ende 1974 beim Institut für Software-Technologie der GMD ein Vorhaben, das Beschreibungsmöglichkeiten namentlich der Echtzeitmittel von PEARL klären soll.

Der Arbeitskreis 5.8 des FNI ist auch das zuständige deutsche Spiegelgremium zu der Mitte 1976 gegründeten internationalen Arbeitsgruppe TC97/SC5/WG1 "Programming Languages for the Control of Industrial Processes (PLIP)" der ISO (International Organization for Standardization). Die Arbeitsgruppe PLIP ist dafür zuständig, Echtzeitsprachen, die von nationalen Normungsorganisationen vorgeschlagen werden, auf Normungswürdigkeit zu prüfen.

Bisher sind von den Engländern CORAL 66 und von den Amerikanern "Industrial Computer System FORTRAN" als Kandidaten eingereicht worden. Ihre Aussichten sind, zumal sie nur auf den jeweiligen nationalen Märkten Bedeutung haben, nicht sehr gut.

Was eine eventuelle Normung von PEARL betrifft, so gehen die Ansichten über diese Frage recht weit auseinander. Während die einen behaupten, daß jede deutsche Normungsaktivität für PEARL das sicherste Mittel sei, PEARL den Weg zu einem internationalen Standard zu verbauen, gibt es auch Stimmen, die zu höchster Eile mahnen, weil anzunehmen sei, daß sonst die Amerikaner als erste einen Normvorschlag für eine Echtzeitsprache einbringen werden, oder daß das oben erwähnte europäische Echtzeitsprachenprojekt bald zu einem Normvorschlag führen wird.

Das Projekt PDV hat seine Anstrengungen bis heute auf das Ziel konzentriert, PEARL erst einmal fertigzuspezifizieren und eine möglichst weitgehende Einheitlichkeit bei den Implementationen in der Bundesrepublik herbeizuführen. Die Hoffnung ist berechtigt, daß sich daraus ein De-facto-Standard ergeben wird, wenn PEARL 1977 einheitlich auf

dem Markt erscheint.

Darüberhinaus muß in der Bundesrepublik die Diskussion darüber fortgesetzt werden, wie im Interesse des Anwenders die Einheitlichkeit von PEARL über eine Norm abgesichert werden könnte.

## Zukünftige Aktivitäten um PEARL

Um PEARL-Programme effektiv schreiben, übersetzen, laden, testen und bedienen zu können, benötigt der Anwender auf dem jeweiligen Prozeßrechner eine Reihe von unterstützenden Werkzeugen (Systemprogramme, Softwareproduktionssystem). Es ist durchaus im Interesse der Anwender, daß diese Werkzeuge in ihren Benutzereigenschaften möglichst einheitlich sind, damit der Übergang von einem Rechner auf einen anderen leichter möglich, d.h. die Portabilität, besonders die der Programmierer, realisiert wird.

Besonderes Augenmerk verdient das Testen von Prozeßrechnerprogrammen. Das Beseitigen von Fehlern in Echtzeitprogrammen ist trotz des enormen Fortschrittes durch Verwenden einer höheren Sprache noch immmer eine anspruchsvolle Aufgabe und ein wesentlicher Kostenfaktor. Noch schwieriger ist der Richtigkeitsnachweis für Automationsprogramme, die Anlagen steuern sollen, deren Betrieb mit einem Sicherheitsrisiko verbunden ist. Im Projekt PDV bildet deshalb die Entwicklung von Verfahren für den Test und die Verifikation von Echtzeitsoftware auch künftig einen Schwerpunkt. Der hierfür gegründete Arbeitskreis steht weiteren Interessenten zur Mitarbeit offen.

Ein neuer Anwendungsaspekt für PEARL sind Mikroprozessoren im Zusammenhang mit billigeren Speichern und der sich wandelnden Struktur der Prozeßrechensysteme. PEARL wird als hierfür geeignet angesehen. Die Untersuchung der eventuellen Erweiterung und zweckmäßigen Benutzung von PEARL bei der Programmierung von Echtzeitrechnersystemen mit räumlich verteilten Prozessorstationen wird ebenfalls vom BMFT gefördert /24/.

Viele Anwender betonen, daß die mit PEARL gewonnenen Anwendererfahrungen nicht verpuffen dürfen, sondern in einer geeigneten Organisation gesammelt und ausgewertet werden sollten. PDV beabsichtigt, einen Vorschlag für eine solche PEARL-Anwender-Organisation zu machen.

## Zusammenfassung

Die universelle, höhere Echtzeitprogrammiersprache PEARL ist im Zeitraum 1969 bis 1976 in einer beispiellosen Zusammenarbeit von Anwendern, Prozeßrechnerherstellern, Softwarehäusern und wissenschaftlichen Instituten in der Bundesrepublik entwickelt worden. Alle wesentlichen Prozeßrechnerhersteller wollen im Jahr 1977 ihre PEARL-Implementationen einheitlich herausbringen. Das BMFT hat diese Arbeiten in seinem Projekt PDV, das von der Gesellschaft für Kernforschung mbH, Karlsruhe, als Projektträger durchgeführt wird, bis 1976 mit ca. 30 Mio. DM Fördervolumen, ausgegeben an ca. 25 Institutionen, unterstützt.

Im Projekt PDV werden die mit PEARL zusammenhängenden Probleme auch weiterhin zentral erfaßt und betreut. Dazu gehören:

- die einheitliche Dokumentation
- die Entwicklung unterstützender Werkzeuge, besonders für das Testen und Verifizieren von Programmen
- die Untersuchung der Einsatzfähigkeit von PEARL für Systeme mit verteilten Prozessoren
- die Betreuung der PEARL-Anwender

Mit der Programmiersprache PEARL können und sollen nicht alle Programme für Prozeßrechner geschrieben werden /25/, jedoch ist die Zeit für den Übergang von der maschinenabhängigen Anwenderprogrammierung zur maschinenunabhängigen Anwenderprogrammierung überreif. Ein prozentual wachsender Anteil an portablen Anwenderprogrammen, wie er in der kommerziellen und technisch-wissenschaftlichen Datenverarbeitung bereits 15 Jahre früher verlaufen ist, wäre volkswirtschaftlich wünschenswert (siehe Bild 4).

Schrifttum

/ 1/ Martin, T.: Höhere Programmiersprachen für Prozeßrechner, PDV-Spalte, Regelungst. Praxis, 18 (1976), S. 251.

/ 2/ Barnes, J. G. P.: RTL/2 - Design and Philosophy. Verlag Heyden & Sohn, Rheine/Westf., 1976.

/ 3/ Ritout, M. u. a.: PROCOL - a programming system adapted for process control. 5. IFAC-Weltkongress, Paris, 1972.

/ 4/ Prozeßautomationssprache PAS1, Sprachbeschreibung. Brown, Boveri & Cie., Mannheim, 1971.

/ 5/ Brandes, J., Eichentopf, S., Elzer, P., Frevert, L., Haase, V., Mittendorf, H., Müller, G., Rieder, P.: PEARL - The Concept of a Process- and Experiment-oriented Programming Language. Elektronische Datenverarbeitung 1o (197o), S. 429-442.

/ 6/ Mühlhahn, A. u. a.: Spezifikation CIMIC/1. PDV-Bericht KFK-PDV 75, Gesellschaft für Kernforschung mbH, Karlsruhe, 1976.

/ 7/ Brunner, P. J., Hinderer, W., Werum, W.: Optimierungsverfahren für die Systemprogrammierung. PDV-Bericht KFK-PDV 62, wie oben.

/ 8/ Brunner,P. J. u. a.: Universelles PEARL-Betriebssystem. PDV-Bericht KFK-PDV 55, wie oben.

/ 9/ Wettstein, H. u. a.: Ein modernes, modulares Betriebssystem für Prozeßrechner und seine Generierung. PDV-Entwicklungsbericht PDV-E 71, wie oben.

/1o/ Timmesfeld, K. H. u. a.: PEARL, a proposal for a process and experiment automation real-time language. PDV-Bericht KFK-PDV 1, Gesellschaft für Kernforschung mbH, Karlsruhe, 1973.

/11/ Eichenauer, B. u. a.: PEARL, eine prozeß- und experiment-orientierte Programmiersprache. Angewandte Informatik 15 (1973), S. 363-372.

/12/ Sammel, B., Timmesfeld, K. H., Rieder, P.: Überarbeitung von PEARL zur Erhöhung der Übereinstimmung mit PL/1. PDV-Entwicklungsbericht PDV-E 39, Gesellschaft für Kernforschung mbH, Karlsruhe, 1974.

/13/ PDV-Entwicklungsbericht PDV-E 1o: ist identisch mit /1o/, nimmt jedoch zusätzlich Bezug auf /12/.

/14/ Krüger, B.: Erfahrungen mit dem Programmiersystem PEARL. 2. Fachtagung Prozeßrechner 1977, Augsburg, 7./8. März 1977.(Tagungsband erscheint demnächst im Springer-Verlag in der Reihe Lecture Notes in Computer Science.)

/15/ Elzer, P. und Holleczek, P.: ASME-PEARL-Compiler wird der Öffentlichkeit vorgestellt. Regelungstechnik 23 (1975), S. 433-436.

/16/ Lauber, R.: Experimentelle Untersuchung von Spracheigenschaften der Prozeßrechnersprache PEARL anhand von Modellprozessen. In: Praxis von Programmiersprachen, -systemen und -generatoren, Reihe Applied Computer Science, Band 3, Carl Hanser Verlag, München/Wien, 1976.

/17/ Heger, J., Koch, G.: Erfahrungen bei der Erstimplementierung eines PEARL-Subset. Lecture Notes in Computer Science, Band 12, Springer Verlag, Berlin/Heidelberg/New York, 1974, S. 4o1-412.

/18/ Rieder, P.: Effiziente PEARL-Implementierung für den PR 33o (erscheint wie /14/).

/19/ Eichenauer, B.: Ein portabler Übersetzer für einen Subset der Prozeßprogrammiersprache PEARL. Lecture Notes in Computer Science, Band 12, Springer Verlag, Berlin/Heidelberg/New York, 1974, S. 576-585.

/2o/ PEARL-Subset for Avionic Applications, Language Description, June 1976. Elektronik-System-Gesellschaft mbH, München.
(Vorläufige Fassung; Anpassung an PEARL-für-Kleinrechner wird noch vorgenommen.)

/21/ Official Definition of CORAL 66. Her Majesty's Stationary Office, London, 197o (Standard for Military Programming).

/22/ Parayre, P., Trocello, M.: LTR, un système de réalisation pour l'informatique temps réel. AFCET-Konferenz über Echtzeitsprachen, Paris, 19./2o. Nov. 1975.

/23/ LTPL (European Group) Language Comparison of Programming Languages ALGOL 68, CAMAC-IML, CORAL 66, PAS1, PEARL, PL/1, PROCOL, RTL/2. Dokument III/437/76-E (zu beziehen durch: Europ. Kommission, Direktion III-D, Rue de la Loi 2oo, B-1o4o Brüssel).

/24/ Steusloff, H.: Zur Programmierung von Echtzeitrechnersystemen mit räumlich verteilten Prozessorstationen. In: Programmiersprachen, Informatik-Fachberichte, Band 1, Springer-Verlag, Berlin/Heidelberg/New York, 1976

/25/ Hoseit, H.: PEARL versus UNPEARL.
Online 14 (1976), S. 237-242

/26/ Stams, D.: Gegenwärtige und künftige Struktur des Marktes der Prozeßlenkungstechnik - Strukturpolitische Förderungsgesichtspunkte. PDV-Bericht KFK-PDV 85, Gesellschaft für Kernforschung mbH, Karlsruhe, 1976.

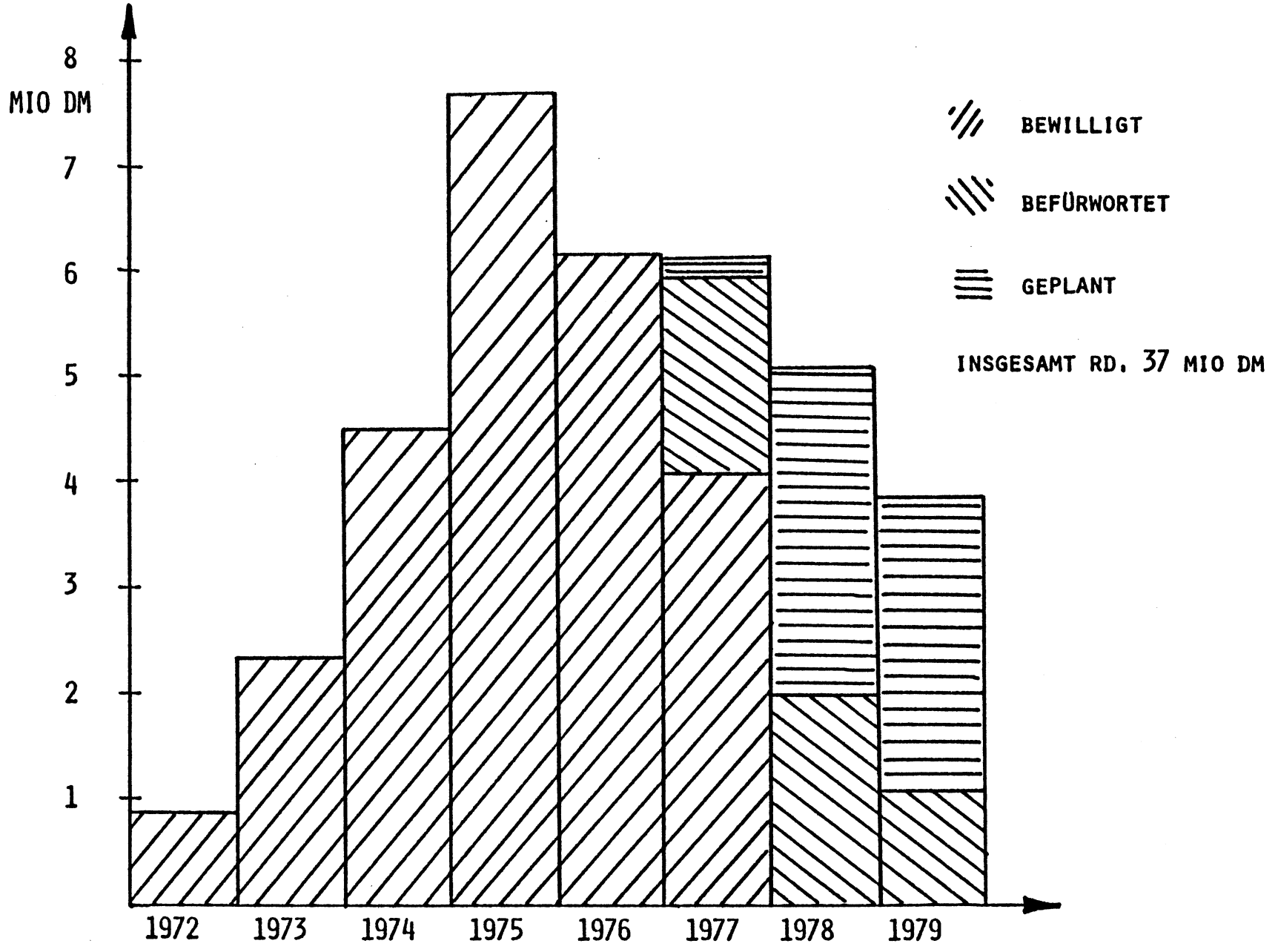



BILD 1 : AUFGEWANDTE FÖRDERMITTEL FÜR DIE PEARL-ENTWICKLUNG

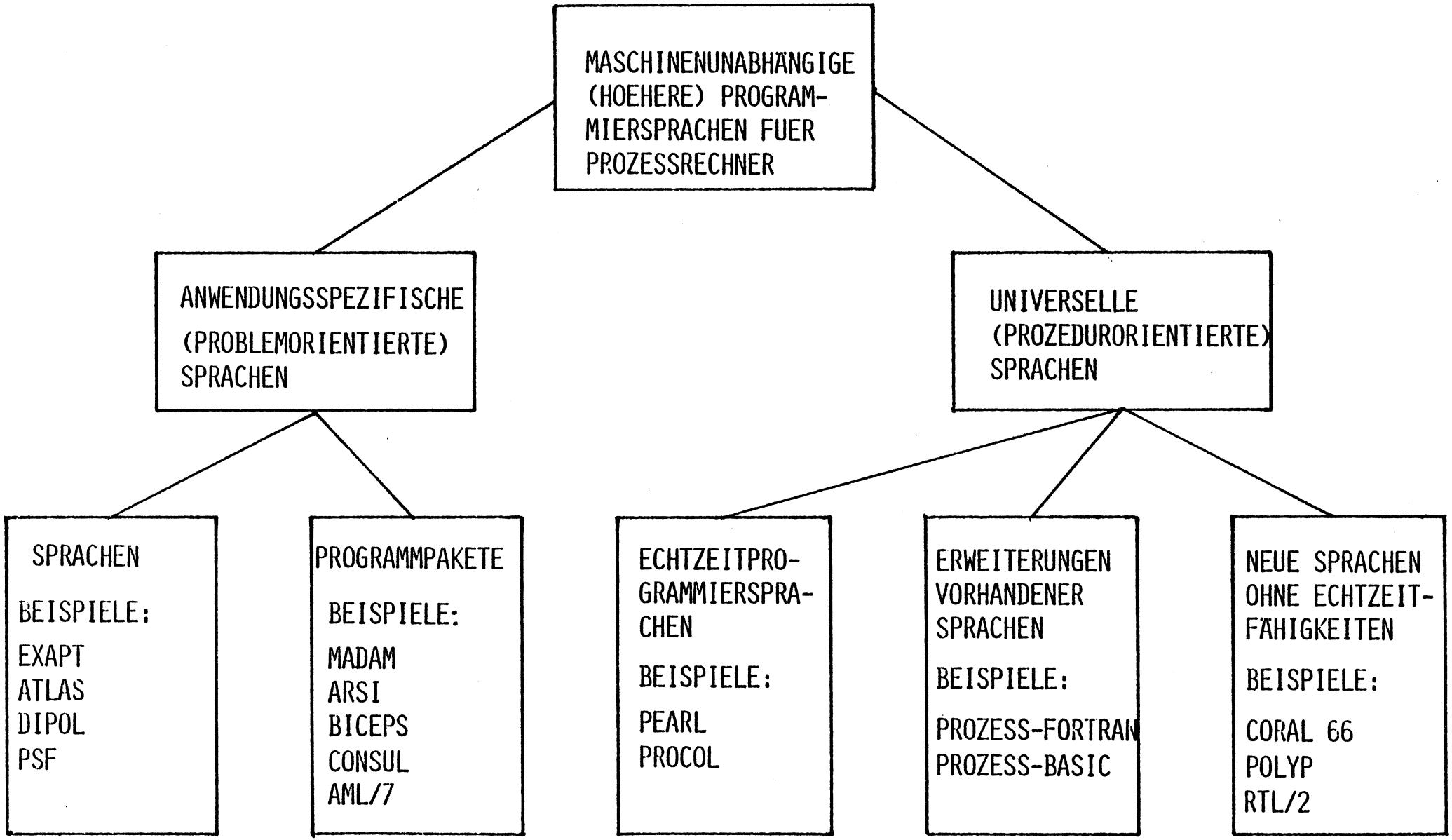



BILD 2 : HÖHERE PROGRAMMIERSPRACHEN FÜR PROZESSRECHNER

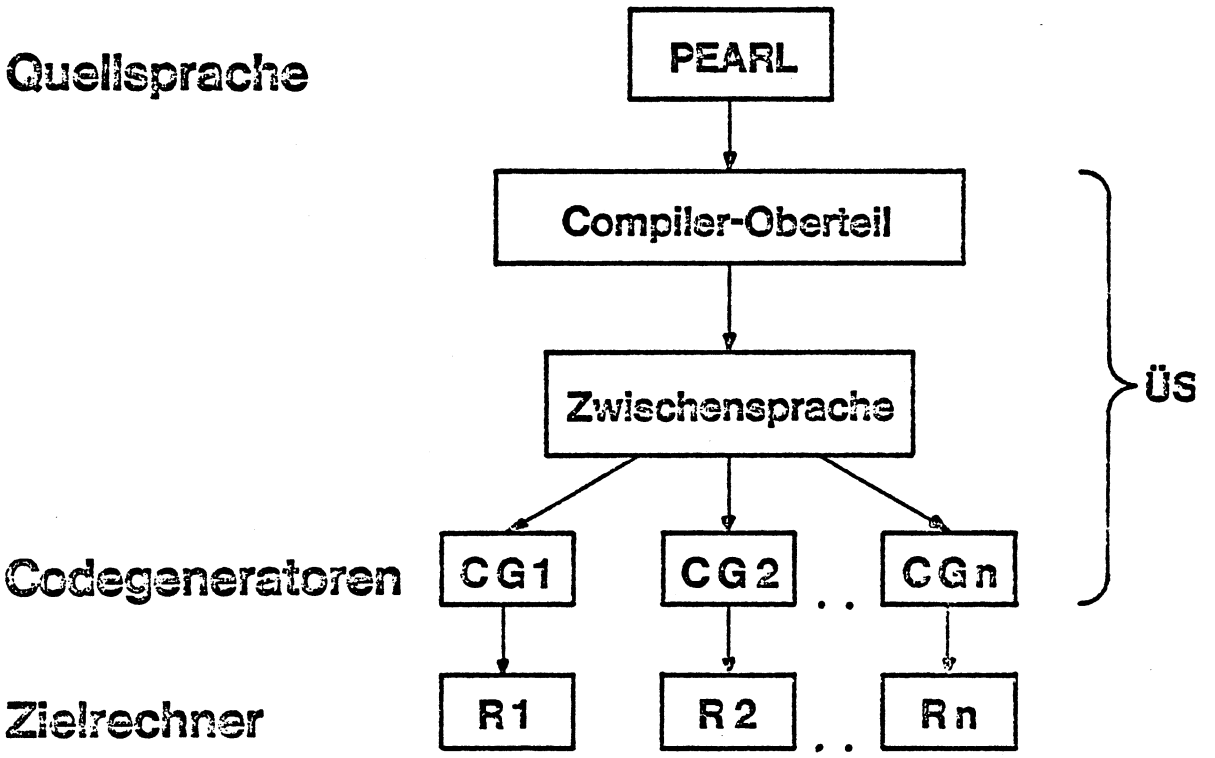



BILD 3 : ZWEISTUFIGES ÜBERSETZUNGSSYSTEM (ÜS) FÜR PEARL

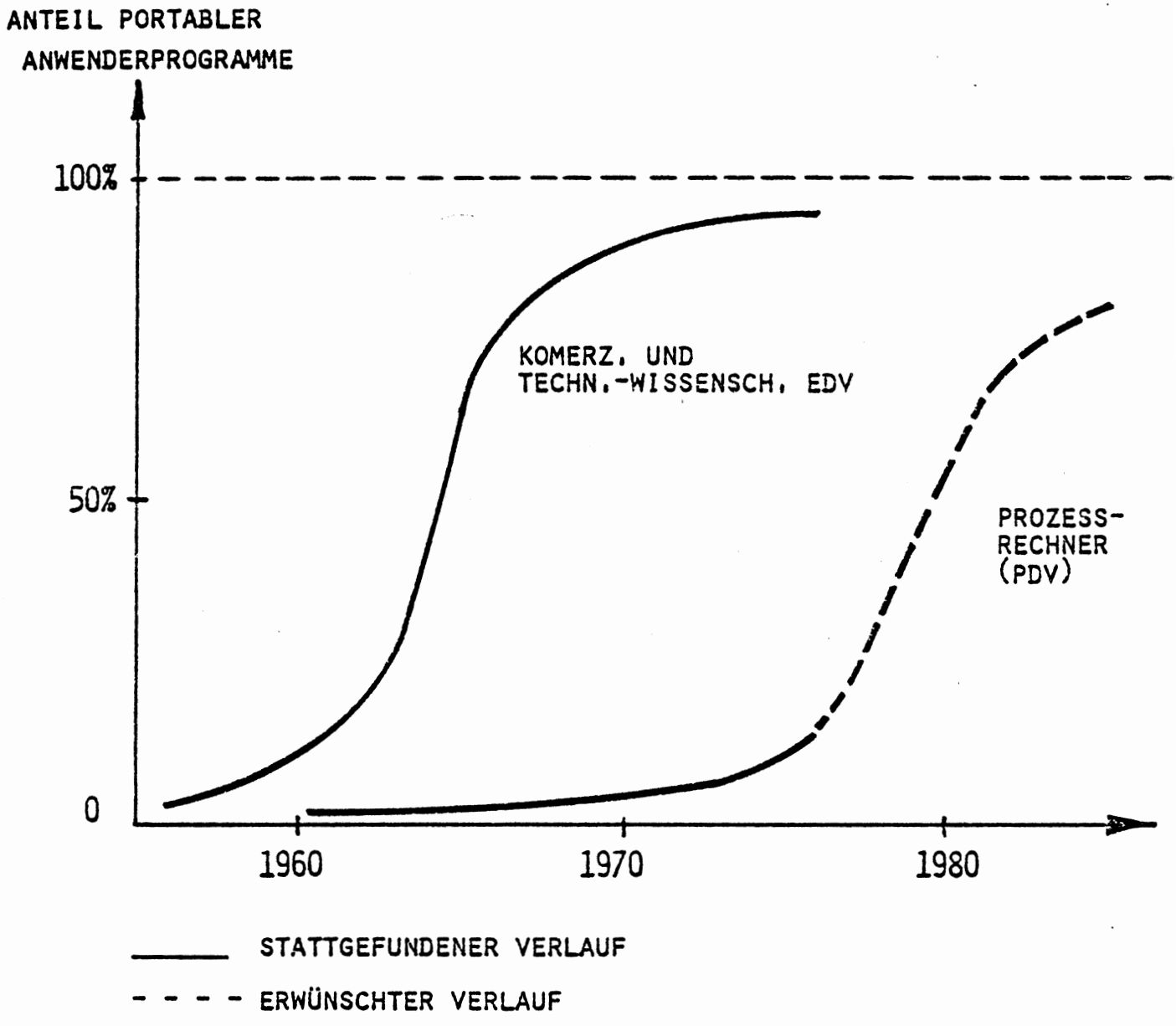



BILD 4 : ÜBERGANG VON DER MASCHINENABHÄNGIGEN ANWENDERPROGRAMMIERUNG ZUR MASCHINENUNABHÄNGIGEN (PORTABLEN) ANWENDERPROGRAMMIERUNG

Mitglieder des SAK:

- Fa. AEG-Telefunken, Konstanz
- Fa. BBC, Mannheim
- Fa. Siemens, Karlsruhe
- ASME (Arbeitsgemeinschaft Stuttgart-München-Erlangen) [1)]
- PfK (Arbeitskreis "PEARL für Kleinrechner")
  (siehe Tabelle 2)
- Fa. MBP (Mathematischer Beratungs- und Programmierdienst), Dortmund
- Ausschuß 4.2 "Programmiertechnik" der VDE/VDI-GMR
- PDV, Karlsruhe (Vorsitz)

Mitarbeitende Gäste:

- Institut für Software-Technologie, Gesellschaft für Mathematik und Datenverarbeitung (GMD), St. Augustin
- Fa. IDAS (Informations-, Daten-und Automationssysteme), Limburg
- IITB (Institut für Informationsverarbeitung in Technik und Biologie der Fraunhofer-Gesellschaft), Karlsruhe
- Fa. Entwicklungsbüro Werum, Lüneburg

Tabelle 1: Im PEARL-Subset-Arbeitskreis (SAK) mitarbeitende Institutionen

- Institut für Informatik III, Universität Karlsruhe
- Institut für Angewandte Informatik und Formale Beschreibungsverfahren, Universität Karlsruhe
- Institut für Angewandte Mathematik, Kernforschungsanlage Jülich

---

1) Die Partner der Arbeitsgemeinschaft im einzelnen sind:

- Institut für Regelungstechnik und Prozeßautomatisierung, Universität Stuttgart
- Institut für Verfahrenstechnik und Dampfkesselwesen, Universität Stuttgart
- Fa. Elektronik-System-Gesellschaft, München
- Fa. Gesellschaft für Prozeßrechnerprogrammierung, München
- Physikalisches Institut III, Universität Erlangen

Fortsetzung Tabelle 1: Im PEARL-Subset-Arbeitskreis (SAK) mitarbeitende Institutionen

- WZL (Laboratorium für Werkzeugmaschinen und Betriebslehre), TH Aachen (Vorsitz)
- Fa. GEI (Gesellschaft für Elektronische Informationsverarbeitung), Aachen (technische Koordination)
- Fa. Dietz Computer-Systeme, Mülheim
- Fa. Krantz Computer, Aachen
- Fa. Krupp Atlas-Elektronik, Aachen
- Fa. Software Partner, Darmstadt
- Institut für Verfahrenstechnik und Dampfkesselwesen, Universität Stuttgart
- ASME (siehe Tabelle 1)
- PDV, Karlsruhe

Tabelle 2: Im PfK (PEARL für Kleinrechner)-Arbeitskreis mitarbeitende Institutionen

# Erfahrungen mit Implementation und Einsatz eines bisherigen PEARL-Subsets bei der ASME

P. Elzer

Physikalisches Institut der Universität
Erlangen-Nürnberg

## I. Die Arbeiten der ASME

### I.1. Implementation

Seit August 1975 ist das ASME*)-PEARL-Obersetzungssystem Stufe I im Einsatz. Der implementierte Subset ist als PDV-Bericht [1,2] veröffentlicht. Für diesen Subset wurde ein portabler Obersetzer mit den dazu gehörigen Betriebssystemsänderungen und -ergänzungen und Laufzeitpaketen für die verschiedenen Zielmaschinen implementiert. Tabelle 1 zeigt die verwendeten Obersetzungs- und Zielmaschinen:

| | Obersetzungsmaschinen | Zielmaschinen |
|---|---|---|
| Stuttgart: IRP | AEG/Telefunken 6050 | AEG-Telefunken 6050 |
| IVP | CDC 6600/ CYBER 174 | AEG-Telefunken 6010 |
| München: ESG/GPP | Siemens 4004 ; Siemens 404 | (Siemens 404) |
| Erlangen: PIE | Siemens 306 | Siemens 306 |

Tabelle 1: Obersetzungs- und Zielmaschinen der ASME

Die verwendete Technologie war: Ein oberer Compilerteil, der in FORTRAN geschrieben ist, übersetzt PEARL-Programme in die maschinenunabhängige Zwischensprache CIMIC I [3]. Diese wiederum wird durch mittels Stage II [4] erstellte Codegeneratoren in Zielmaschinencode übersetzt. Diese Technologie wurde bereits mehrfach in Veröffentlichungen beschrieben, so z.B. in [5, 6, 7].

---

*) Arbeitsgemeinschaft Stuttgart-München-Erlangen
Stuttgart: - Institut für Regelungstechnik und Prozeßautomatisierung (Prof. Lauber)
- Institut für Verfahrenstechnik und Dampfkesselwesen (Prof. Quack)
München: - Fa. ESG
- Fa. GPP
Erlangen: - Physikalisches Institut, Abt. III (Prof. Fiebiger)

## I.2. Anwendungen

Die Einsatzerprobung von PEARL innerhalb der ASME findet in drei sich ergänzenden Anwendungsbereichen statt:

- Modellprozesse am IRP in Stuttgart.
  Es werden hier folgende wesentlichen Anwendungsklassen abgedeckt:
  - Betriebsüberwachung
  - DDC
  - Automatische Geräteprüfung
  - Warenverteilsystem mit Hochregallager

  Eine erste Analyse der Erfahrungen aus diesen Einsatzfällen findet sich z.B. in [8].
- Computerunterstützte Meß- und Experimentiertechnik im Beschleunigerlabor (6 MV-Tandem-Van-de-Graaff-Beschleuniger) der Physikalischen Institute in Erlangen (PIE).
  Hier sind als Beispiele zu nennen:
  - Reichweitenmessung von $\alpha$-Strahlen
  - Prüfung von Detektoren für radioaktive Strahlung
  - Rechnerkopplung über CAMAC zur on-line Experimentdatenerfassung
  - Interaktives Programm zur Darstellung und Auswertung kernphysikalischer Messungen

Automatisierung eines 30 MW Kraftwerksblockes im Heizkraftwerk der Universität Stuttgart (IVD). Hierunter fallen z.B.:

- one-line Datenerfassung
- An- und Abfahrmodell des Kraftwerkblockes
- pH-Wert Regelung für Abwasser

Außerdem wurde seitens der beteiligten Universitäten PEARL inzwischen ins Lehrangebot aufgenommen und es werden potentielle Anwender beraten.

# II. Erfahrungen

## II.1. Implementationserfahrungen

Die gewählte Implementationstechnologie erwies sich als recht glücklicher Griff. CIMIC I bewährte sich als saubere Schnittstelle zwischen den einzelnen Teams, Stage II als schnell erlernbares und handliches Arbeitshilfsmittel und FORTRAN, innerhalb seiner Grenzen, auch als Compilererstellungssprache.

Der Änderungsaufwand für die sehr unterschiedlichen Betriebssysteme war dementsprechend auch verschieden, hielt sich aber in Grenzen.

Die zeitweise verbreitete Meinung, eine Sprache wie PEARL könne nur auf Großrechnern übersetzt werden, konnte durch den relativ mäßigen Speicherbedarf des ASME-Stufe I-Obersetzungssystems widerlegt werden. So kommt der Compiler z.B. in Erlangen mit einem Laufbereich von 20 K Worten auf der Siemens 306 aus.

## II.2. Einsatzerfahrungen

Hier sind mehrere Gesichtspunkte zu betrachten:

Die Codeverlängerung gegenüber Assemblerprogrammen kann als im üblichen Rahmen liegend angesehen werden. Aus dem IRP (Prof. Lauber) in Stuttgart liegen hierzu quantitative Vergleiche vor. Die folgende Tabelle ist aus [8] entnommen.

| Task | Speicherbedarf | | Code-verlängerungs-faktor |
|---|---|---|---|
| | Assembler-Programmierung (Befehlsworte) | PEARL-Programmierung (Befehlsworte) | |
| DATEN + FALZYK | 397 | 404 | 1,02 |
| ANEIN | 82 | 174 | 2,12 |
| UMRECH | 92 | 175 | 1,90 |
| ALARME | 133 | 188 | 1,41 |
| GREWE | 140 | 220 | 1,57 |
| STOER | 401 | 556 | 1,39 |
| PROTOK | 361 | 501 | 1,38 |

Tabelle 2: Vergleich des Kernspeicherbedarfs bei Programmierung in PEARL und in Assemblersprache AMRAS am Beispiel der Tasks des Modellprozesses "Betriebsüberwachung"

Aus Erlangen liegen Schätzungen vor, die allerdings von erfahrenen Assemblerprogrammierern vorgenommen wurden. Sie bestätigen im wesentlichen diese Werte.

Man muß hierzu allerdings bemerken, daß im ASME-Stufe I Obersetzungssystem noch keinerlei Codeoptimierung vorgenommen wird.

Bei der Programmerstellungszeit ergaben sich zum Teil verblüffende Verkürzungsfaktoren, die bis zu 1 : 25 reichen, wobei in Erlangen Programmerstellung und Test als Einheit betrachtet wurden. Solche Vergleichswerte sind, wie auch schon in [8] dargestellt, immer mit großen Streuungen behaftet und stark von der persönlichen Erfahrungsbasis des jeweiligen Programmierers abhängig. Daß aber die sonst in der Literatur angegebenen Programmierzeitverbesserung gegenüber Assembler bei Verwendung einer höheren Sprache (ohne Test !) von 1 : 5 so häufig so drastisch unterschritten wurde, legt die Vermutung nahe, daß die Verwendung von PEARL auch in der Testphase sehr günstige Auswirkungen hat.

Auch die Erlernbarkeit von PEARL scheint sehr gut zu sein. So konnten in Erlangen bei einem einsemestrigen PEARL-Kurs mit 2 Semesterwochenstunden + Übungen Studenten nach etwa dem halben Semester durchaus anspruchsvolle Probleme (z.B. mit mehreren Tasks und graphischer Ein-/Ausgabe) selbständig lösen.

Auch aus dem IRP werden in dieser Beziehung sehr günstige Resultate berichtet [8].

Ein sehr schwer zu quantifizierender Begriff ist die Handlichkeit einer Programmiersprache als Werkzeug für die beabsichtigten Zwecke. Es konnten aber eigentlich alle angegangenen Probleme in ausreichend eleganter Form gelöst werden. Darüber hinaus bewährte sich PEARL sogar in Anwendungsfällen, für die es ursprünglich gar nicht entwickelt worden war, z.B. bei der Erstellung eines Assemblers.

Der Dokumentationswert von PEARL-Programmen entsprach den Erwartungen. Ein möglicherweise für das Management von Softwareentwicklungsgruppen interessanter Effekt stellte sich bei der Betreuung von Diplomarbeiten heraus: es war wesentlich leichter, in PEARL geschriebene Programme zu korrigieren und zu bewerten, als dies bei in Assembler geschriebenen Programmen der Fall gewesen war.

In Erlangen wurde mit Erfolg versucht, Methoden des 'strukturierten Programmierens' auf die PEARL-Programmierung anzuwenden.

## III. Schlußfolgerung

Innerhalb der ASME wurden bisher insgesamt Anwenderprogramme im Umfang von über 160 K Worten erstellt. Zusammenfassend kann man wohl sagen, daß PEARL im Rahmen der ASME die in es gesetzten Erwartungen erfüllt hat. Die ersten Erprobungsergebnisse lagen auch so rechtzeitig vor , daß eine große Zahl von Ergänzungs- und Verbesserungsvorschlägen in den im Januar 1976 begonnenen Festschreibungsprozeß von PEARL eingebracht werden konnten.

Literaturhinweise

[1] ASME-PEARL-SUBSET/1, PDV-Bericht KFK-PDV 76

[2] Programmieranleitung für das ASME-PEARL-Subset/1, PDV-Bericht, wird veröffentlicht

[3] Spezifikation CIMIC/1, PDV-Bericht KFK-PDV 75

[4] W.M. Waite: The Mobile Programming System STAGE2, Comm. ACM, 13, 6, (1970)

[5] B. Eichenauer: Entwicklungskonzept für ein Programmsystem zur Erstellung und Erzeugung von System-Software für die Prozeßautomatisierung, PDV-Mitteilungen 1/1971, S. 34 - 50

[6] P. Elzer, P. Holleczek: ASME-PEARL-Compiler wird der Öffentlichkeit vorgestellt
Regelungstechnik 12 (1975) 433 - 436

[7] P. Elzer, P. Holleczek, W. Lindstedt, K. Pelz, F.J. Prester, R. Rössler: The Implementation of a subset of the real-time language PEARL and examples of its application, IFAC/IFIP, SOCOCO-76, 1st International Symposium on Software for Computer Control

[8] R. Lauber: Experimentelle Untersuchung von Spracheigenschaften der Prozeßrechnersprache PEARL anhand von Modellprozessen, Applied Computer Science Band 3: Praxis von Sprachen (D. König, Hrsgr.), Carl Hanser Verlag, München-Wien, 1976

## Inhalte der Arbeiten des Kreises "PEARL für Kleinrechner"

Prof. Dr.-Ing. T. Pfeifer

Aufgrund der Tatsachen, daß Kleinrechner durch ihr ständig verbessertes Preis-/Leistungsverhältnis zunehmend Einsatz in der Prozeßautomatisierung finden, daß die Software jedoch durch steigende Erstellungskosten und fehlende Portabilität mit dieser Entwicklung keineswegs Schritt hält, konstituierte sich der Arbeitskreis "PEARL für Kleinrechner" mit der Zielsetzung, ein einheitliches PEARL-Subset auf Prozeßrechnern verschiedener Hersteller zu implementieren. Dieses Subset ist identisch mit dem sogenannten "Basis-PEARL", das diejenige Untermenge von FULL-PEARL darstellt, die in jeder Implementation enthalten ist, so daß Programme, die im Basis-Subset geschrieben sind, einen voll portablen Problemteil besitzen.

Die Zielrechner sind gekennzeichnet durch einen Hauptspeicherausbau von 20 - 24 K Worten und einer Wortlänge von 16 bit. Wenn die Übersetzung der PEARL-Programme auf den Rechnern selbst vorgenommen werden soll, wird noch ein externer Massenspeicher benötigt.

Die Implementationsarbeiten werden vom Bundesministerium für Forschung und Technologie im Rahmen des 3. DV-Programms (Projektträger: Gesellschaft für Kernforschung, Bereich PDV, Karlsruhe) gefördert.

Der Arbeitskreis besteht aus Rechnerherstellern, Softwarefirmen und Hochschulinstituten; im einzelnen handelt es sich um die Rechnerhersteller Krantz Computer (Aachen), Dietz Industrieelektronik (Mülheim/Ruhr) und Krupp-Atlas-Elektronik (Bremen); die Softwarefirmen Gesellschaft für Prozeßrechnerprogrammierung (GPP, München), Gesellschaft für Elektronische Informationsverarbeitung (GEI, Walheim) und Software-Partner (SP, Darmstadt); die Hochschulinstitute Institut für Verfahrenstechnik und Dampfkesselwesen (IVD, Universität Stuttgart) und

das Werkzeugmaschinenlabor (WZL, TH Aachen) mit seinen Abteilungen Automatisierung und Meßtechnik. (s. Bild 1).

Die Arbeiten der Firmen Software-Partner und Krantz ruhen momentan. Die Implementation des Basis-PEARL auf dem Krantz-Rechner Mulby 3 wird eventuell durch das Werkzeugmaschinenlabor vorgenommen. Zielrechner der Gruppe sind Prozeßrechner mit einer Wortlänge von 16 bit, einem Hauptspeicher von 48 k byte und einem externen (random access) Massenspeicher. Es handelt sich im einzelnen um folgende Rechner: (s. Bild 2)

| | |
|---|---|
| Dietz-Industrieelektronik | Mincal 621 |
| Krantz-Computer | Mulby 3 |
| Krupp-Atlas-Elektronik | EPR 1100 |
| Télémécanique | T 1600 |

Für die Implementation sind folgende Aufgaben zu erledigen:

Auswahl eines geeigneten Subsets
Bau des Compilers
Bau der Codegeneratoren
Anpassung der Laufzeitsoftware
Anpassung der Betriebssysteme (s. Bild 3 + 4)

Das ausgewählte Subset entspricht Basis-PEARL bis auf den Bereich der Ein-/Ausgabe. Zu dem Zeitpunkt, als die Arbeiten am Compiler begonnen werden mußten, war das Ein-/Ausgabe-Konzept noch nicht voll ausdiskutiert; der jetzt entwickelte Compiler spiegelt das E/A-Konzept vom Juni 1976 wider. Ende 1977 wird jedoch auch das E/A-Konzept an das einheitliche Basis-PEARL angeglichen sein.

Die Arbeitsteilung im Kreis sieht vor, daß der rechnerunabhängige Compilerteil für alle Zielrechner gemeinsam von der Firma GPP entwickelt wird; er produziert den rechnerunabhängigen Zwischencode CIMIC. Der Aufwand, der nötig ist, um den Compiler auf eine Maschine zu bringen, besteht im Bau eines Codegenerators für CIMIC, da der Compiler als Programmsystem in dieser Zwischensprache vorliegen wird.

Die Codegeneratoren für CIMIC werden von den Hochschulinstituten auf der Basis des Makroprozessors Stage-2 gebaut, d.h. die Abbildung von CIMIC auf den Assemblercode der Zielmaschine geschieht über Makrodefinitionen; der Makrokopf entspricht der CIMIC-Anweisung, der Makrokörper seiner Assemblerauflösung. Ein CIMIC-Programm wird also als eine Folge von Makroaufrufen aufgefaßt und aufgelöst. Die Anpassung der Betriebssysteme und der Laufzeitsoftware an die Bedürfnisse von PEARL wird von den Rechnerherstellern geleistet.

Der Firma GEI obliegen die fachliche Koordination des Kreises und einige rechnerunabhängige Aufgaben, wie z.B. die Entwicklung von Testprogrammen zur Austestung der Codegeneratoren.

Nach Abschluß der Implementationen ist ein Gesamttest des Basis-PEARL-Systems durch Pilotanwendungen in folgenden Bereichen vorgesehen:

1. Verfahrenstechnik

   Automatische Abwasserneutralisation im Durchfluß
   Datenerfassung im überkritischen Dampferzeuger
   (IVD, Prof. Quack)

2. Fertigungstechnik

   Meßwerterfassung (Qualitätskontrolle) in rechnergeführten Fertigungsanlagen
   (WZL/MT, Prof. Pfeifer)

   Steuerung und Auswertung der Meßdaten an einer CNC-geführten 3-Koordinaten-Meßmaschine
   (WZL/MT, Prof. Pfeifer)

   Werkzeugmaschinensteuerung
   (WZL/WZM, Prof. Weck)

3. Lagertechnik

   Materialflußsteuerung
   (GEI)

4. Produktion

Laborautomation
(Dietz-Industrieelektronik)

Stranggußautomation
alternativ: Automatisierung der Werkstoffprüfung
(Krupp-Atlas-Elektronik)

Zum derzeitigen Stand der Arbeiten läßt sich sagen, daß die ersten beiden Compilerläufe (Lexikalanalyse und Syntaxanalyse) fertiggestellt und ausgeliefert sind; die Auslieferung des gesamten Compilers ist für April 1977 geplant. Die Codegenerierung (Abbildung von CIMIC auf Assembler) ist nahezu abgeschlossen; die Anpassungsarbeiten zu den Betriebssystemen sind noch im Gange.

Aus den bisherigen Arbeiten läßt sich sagen, daß die Codegenerierung auf Stage-2-Basis zwar sehr schnell abgeschlossen werden konnte, jedoch wenig Spielraum für Optimierungen läßt und dadurch zwangsläufig relativ viel Speicherplatz benötigt. Für den späteren industriellen Einsatz wird daher ein effizienteres Codegenerierverfahren erforderlich sein; diesbezügliche Untersuchungen sind bei der Firma Krupp-Atlas-Elektronik bereits im Gange.

Darüber hinaus sind Codeineffizienzen aufgrund des Implementierungsprinzips zu erwarten; der Compiler wird einen auf CIMIC-II abgestimmten Code absetzen, der zum Teil rechnerabhängige Optimierungen aufgrund eines gewissen Informationsverlustes verhindern wird. Trotzdem scheint der eingeschlagene Weg der günstigste, zumindest im Hinblick auf die Implementationskosten und auf eine Übernahme der Ergebnisse durch weitere Implementatoren.

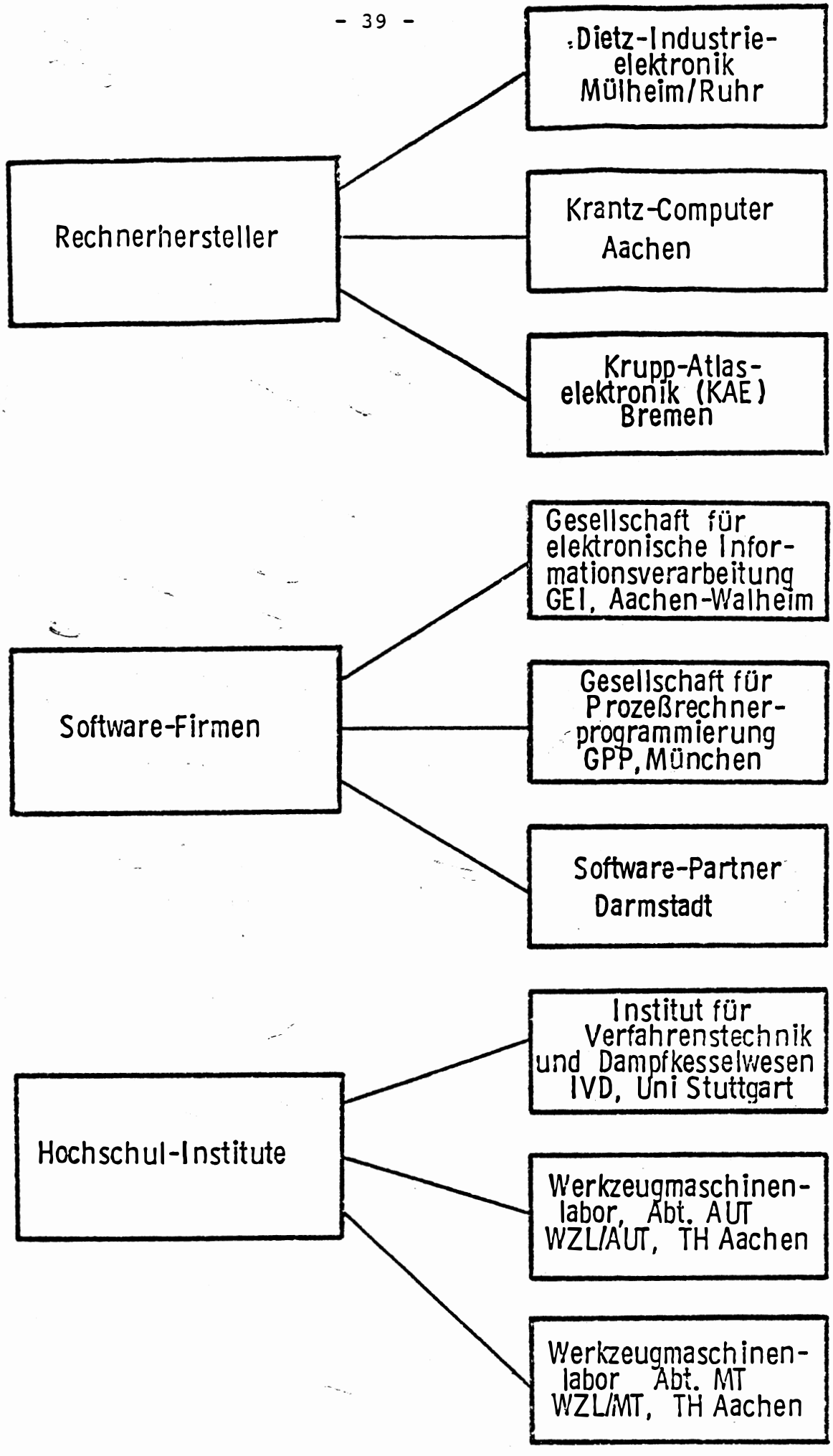

Rechnerhersteller
Dietz-Industrie-elektronik Mülheim/Ruhr
Krantz-Computer Aachen
Krupp-Atlas-elektronik (KAE) Bremen
Software-Firmen
Gesellschaft für elektronische Informationsverarbeitung GEI, Aachen-Walheim
Gesellschaft für Prozeßrechner-programmierung GPP, München
Software-Partner Darmstadt
Hochschul-Institute
Institut für Verfahrenstechnik und Dampfkesselwesen IVD, Uni Stuttgart
Werkzeugmaschinen-labor, Abt. AUT WZL/AUT, TH Aachen
Werkzeugmaschinen-labor Abt. MT WZL/MT, TH Aachen

Bild 1

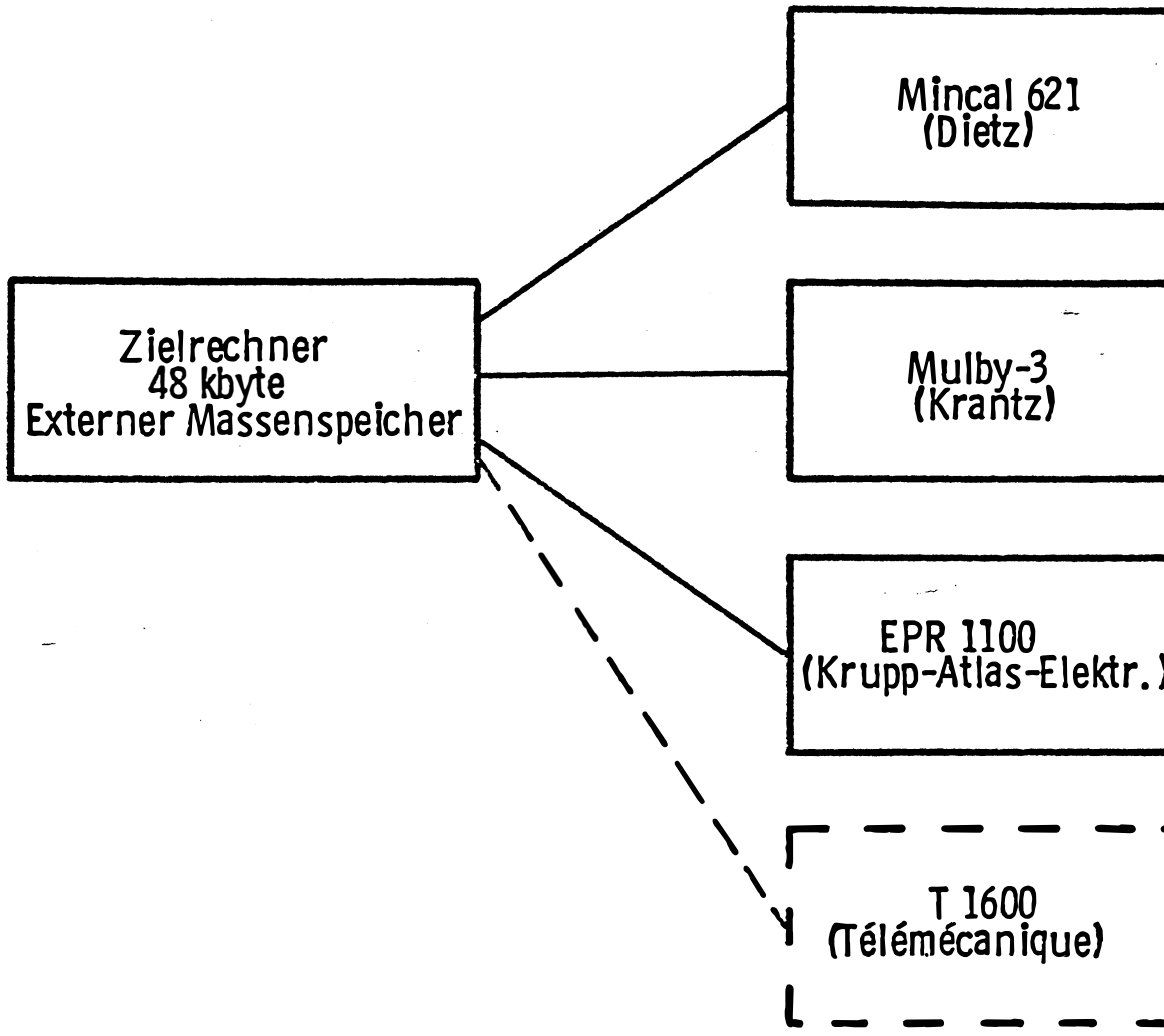

Mincal 621
(Dietz)
Zielrechner
48 kbyte
Externer Massenspeicher
Mulby-3
(Krantz)
EPR 1100
(Krupp-Atlas-Elektr.)
T 1600
(Télémécanique)

Bild 2

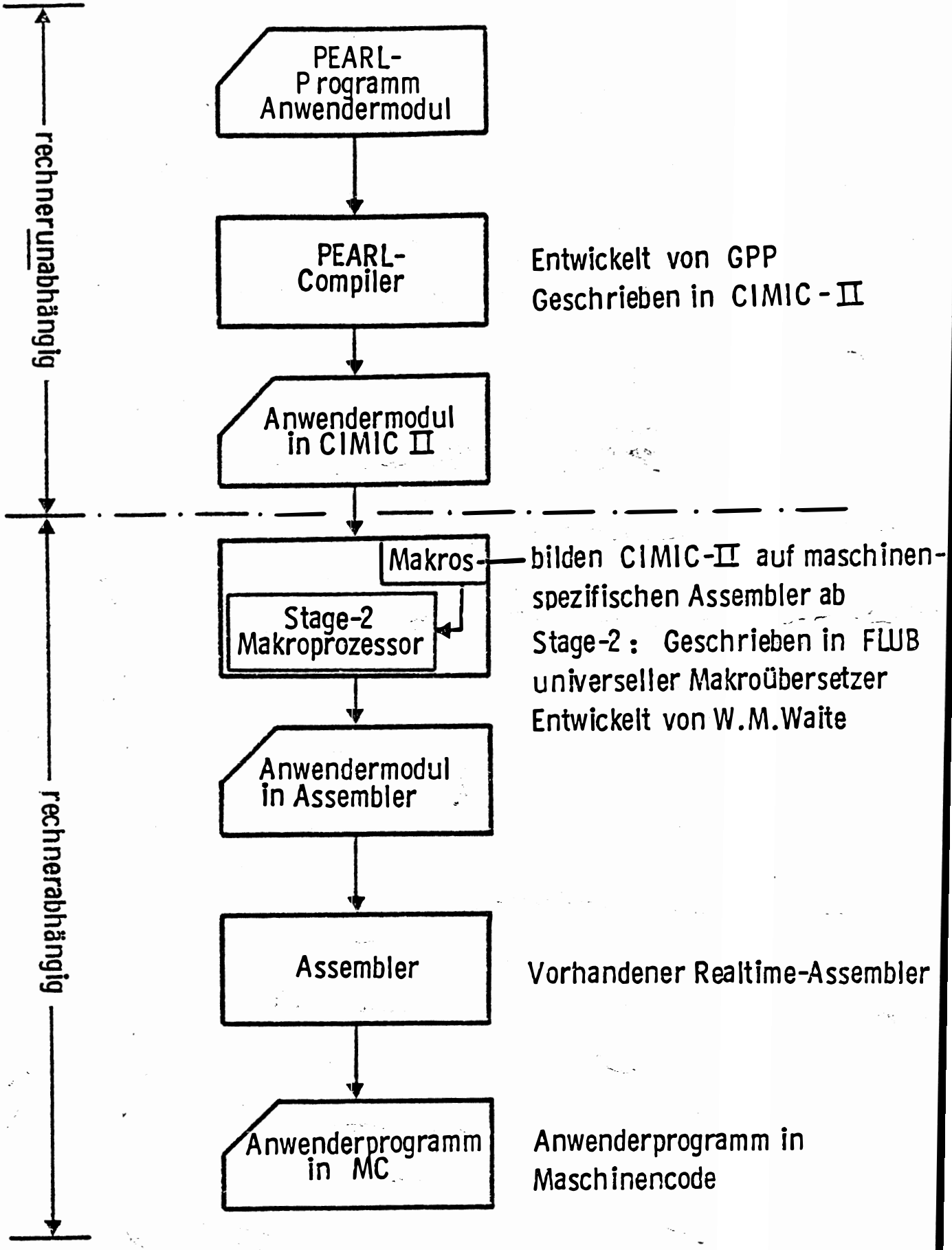

rechnerunabhängig
PEARL-
Programm
Anwendermodul
PEARL-
Compiler
Entwickelt von GPP
Geschrieben in CIMIC - II
Anwendermodul
in CIMIC II
Makros
Stage-2
Makroprozessor
bilden CIMIC-II auf maschinen-
spezifischen Assembler ab
Stage-2 : Geschrieben in FLUB
universeller Makroübersetzer
Entwickelt von W.M.Waite
Anwendermodul
in Assembler
rechnerabhängig
Assembler
Vorhandener Realtime-Assembler
Anwenderprogramm
in MC
Anwenderprogramm in
Maschinencode

Bild 3

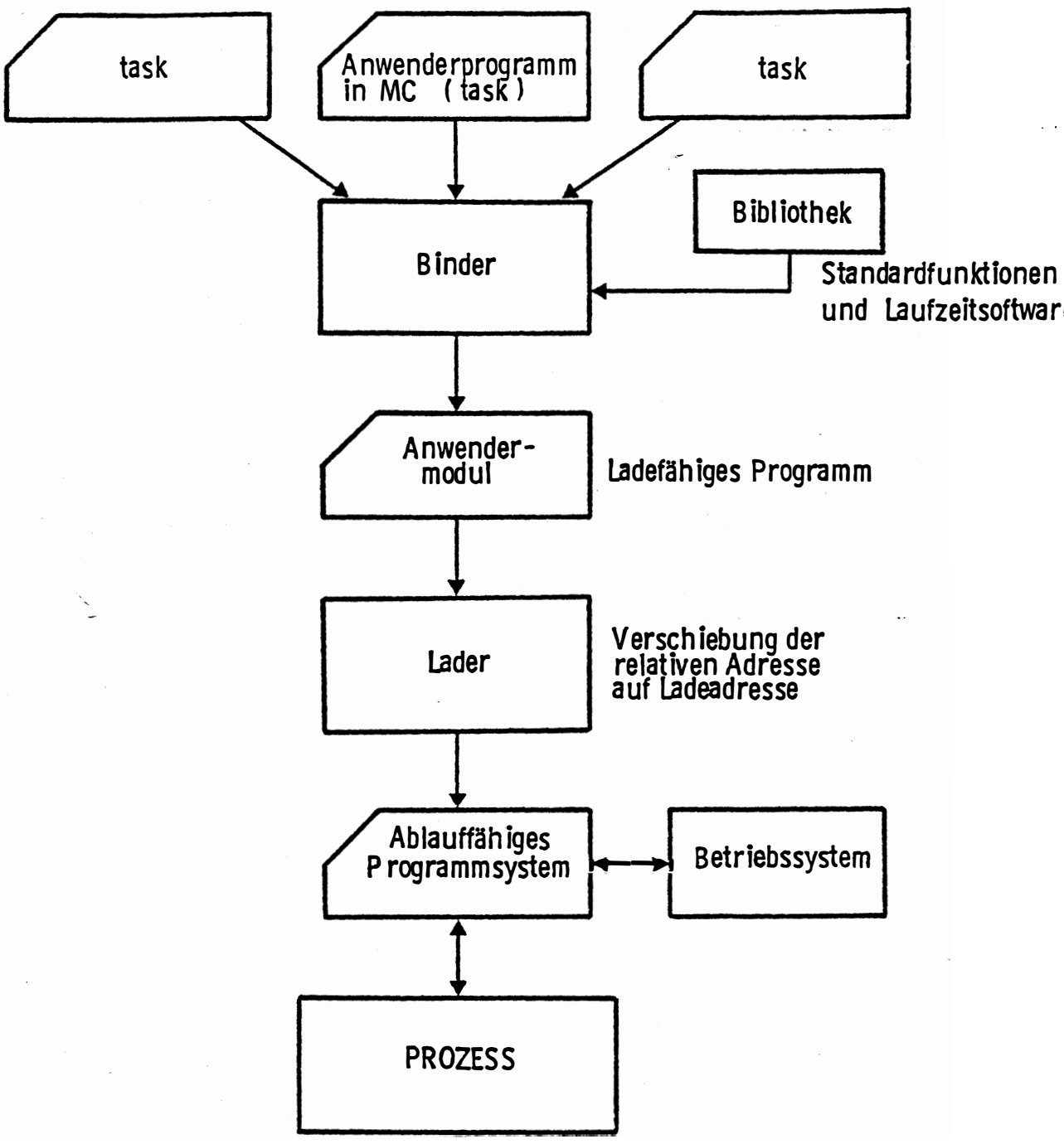

task
Anwenderprogramm in MC ( task )
task
Bibliothek
Binder
Standardfunktionen und Laufzeitsoftwar
Anwender-modul
Ladefähiges Programm
Lader
Verschiebung der relativen Adresse auf Ladeadresse
Ablauffähiges Programmsystem
Betriebssystem
PROZESS

Bild 4

Dr. Horst Mittendorf SIEMENS - E STE 33

Erfahrungen mit bisherigen PEARL-Subsets (an Modellprozessen und im industriellen Einsatz)

Aufgrund der Erfahrung von mehr als einem Jahrzehnt Prozeßrechner-Entwicklung im Hause SIEMENS, gewonnen an mehr als 1000 Rechner-Einsatzfällen, unterteilen wir eine "Prozeßrechner-Software der Gegenwart" in

Programmpakete und individuelle Software (Bild 1).

Die Programmpakete entsprechen dabei vorwiegend wiederkehrenden, technologisch orientierten Aufgaben, die individuellen Programme sind bevorzugt erforderlich bei immer neueren, anlagenorientierten Problemstellungen. Generelles Ziel einer Software-Entwicklung muß sein, den Anstieg der Software-Kosten zu begrenzen.

Die Anwendung dieser Erkenntnisse auf den Bereich heute verfügbarer Programmierungsmittel, abgestuft nach Assembler - Prozeß-FORTRAN - PEARL läßt dann (dargestellt im Bild 2) erkennen, daß in jedem Fall technisches Know-how die unabdingbare Einsatz-Voraussetzung bildet. Die gegenüber Prozeß-FORTRAN zusätzlich verfügbaren Mittel sind bei PEARL dargestellt. Erst die Zuhilfenahme der geeigneten Projektierungsmittel ermöglicht die für die Anwendung rationell verwendbaren technologischen Programmpakete.

Euphorische Aussagen zu PEARL sollten daher anhand noch allgemein weitgehend offener Fragen relativiert werden. So ist zu fragen: Was leistet PEARL bei

| | | |
|---|---|---|
| . Aufgabenanalyse? | wenig! | |
| . Systementwurf? | einiges! | (Systemteil, globale Nahtstellen) |
| . Programmtest? | einiges! | (Testsysteme, weniger Fehler) |
| . Dokumentation? | viel! | |
| . Inbetriebnahme? | wenig! | (Bild 3) |
| . Programm-Wartung? | einiges! | |

Zu ergänzen ist, daß die bisher problematische Frage der begrenzten Portabilität dadurch entschärft wird, daß ab Jan. 77 das "Basis-PEARL" als für die erwähnten Implementatoren verbindlich anerkannt ist.

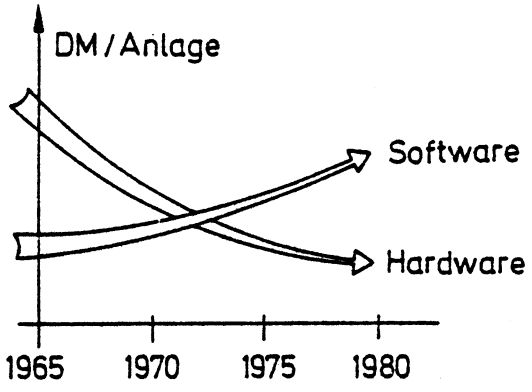
Was PEARL nicht leistet, ist natürlich die Festlegung komplizierter Datenstrukturen auf Grund der Problemstellung selbst (Bild 4), zusätzlich die Gesamtproblematik der Generierungs- und Bedienungsebene. In dem hier gezeigten Beispiel wird die Steuerung einer komplizierten Meßwertverarbeitung über eine (im PEARL-Sinne) strukturierte Liste abgewickelt. Die Festlegung der Bit-Struktur (siehe Lupen-Effekt) erfordert außerordentlichen Aufwand, der durch Verwendung von höheren Sprachen nicht nennenswert zu senken ist. Die zusätzliche Komplexität der Datenvorgabe und der on-line-Bedienung sei nur angedeutet.

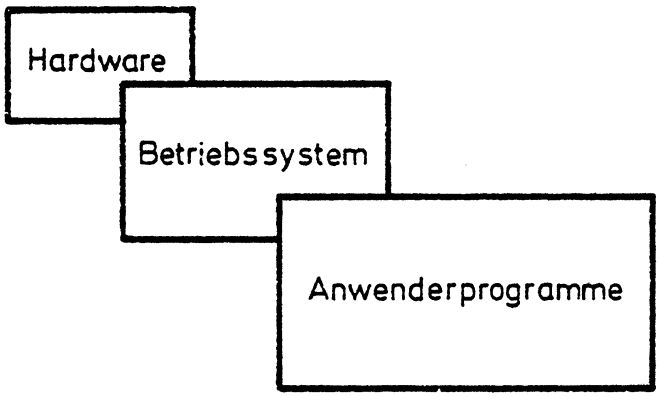
Ist diese sehr umfangreiche Klärungsphase eines Projektes abgeschlossen, so kann die Datenstruktur festgelegt werden. Das ist im nächsten Beispiel (Bild 5) am Eintrag eines Änderungshinweises am Ende eines Textfeldes zu sehen. Der entsprechende Eintrag wird in der PEARL-Prozedur AEND erzeugt und PUFF8 hinterlegt. Da in diesem der industriellen Anwendung entnommenen Beispiel sowohl PEARL- als auch Assembler-Programme vorlagen, ist der Vorteil der PEARL-Schreibweise ersichtlich (Faktor 15). An dieser Stelle einer Projekt-Abwicklung schlägt nun die komfortable Daten-Strukturierung von PEARL positiv zu Buche. Im Beispiel ist das an der Indizierung einer Hinweisliste über ein Index-Strukturfeld aufgezeigt (FIX ist durch TOFIXED zu ersetzen!).

Ein entsprechender Vergleich mit Prozeß-FORTRAN findet sich im nächsten Bild 6. FORTRAN ist hier ungünstiger als PEARL und liegt nicht wesentlich über Assembler-Niveau. Das Fehlen von Datenstrukturen und Wandlungs-Operatoren in Prozeß-FORTRAN bedeutet hier viel Aufwand an Bit-Handling (6 Anweisungen gegen 1 in PEARL).

Die vom PEARL-Compiler abgesetzte Befehlsfolge ist im Bild 7 dargestellt. Hinter etwa 11+2 abgesetzten Befehlen stehen ca. 2*35+10=80 durchlaufene Befehle des Laufzeitsystems ⇒statischer Code nur unwesentlich von Assembler-Lösung verschieden, dynamischer Durchlauf ca. 90/15=6 mal länger!

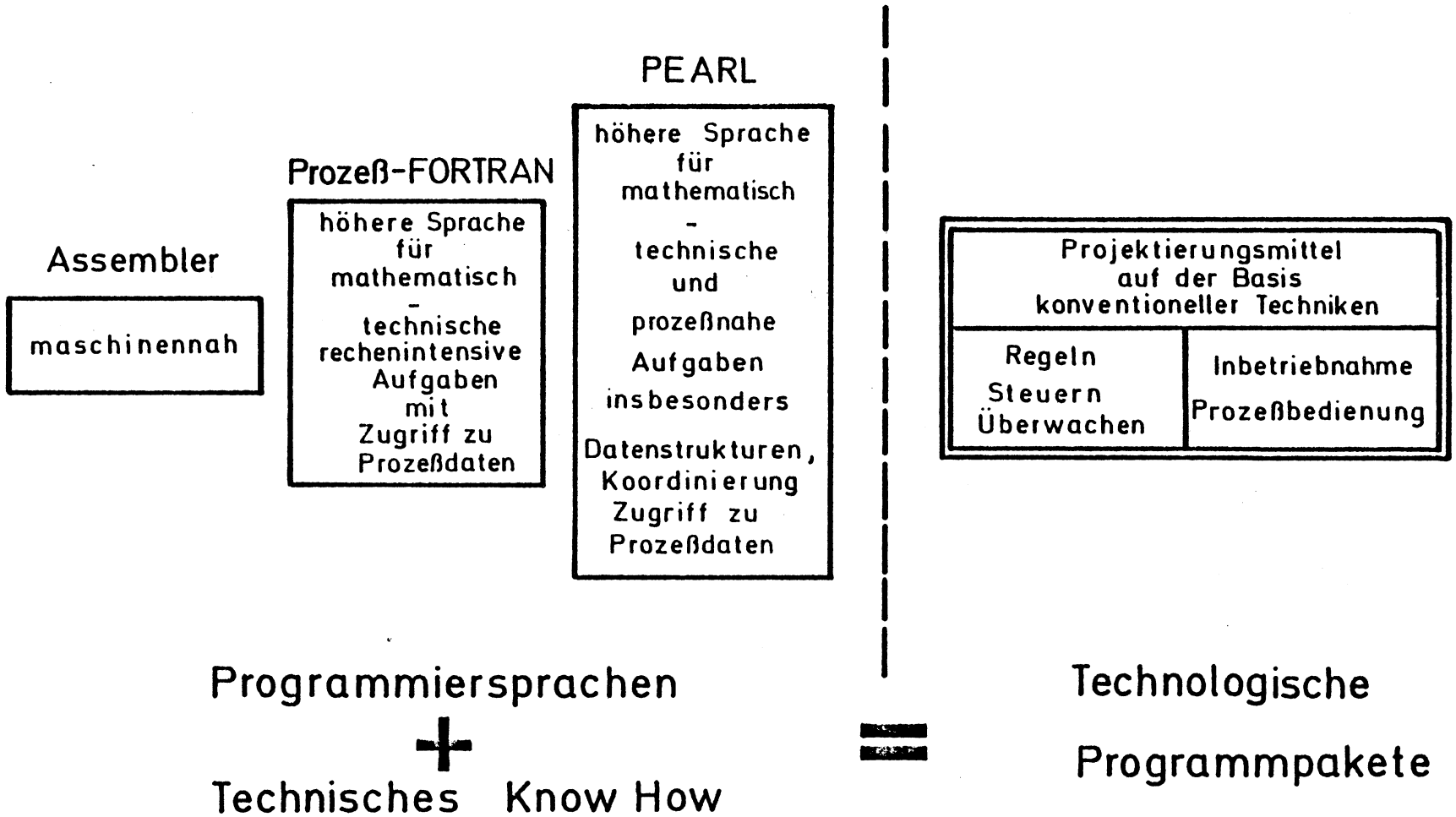
Zusammengefaßt: Wir erwarten bei Verwendung von PEARL für Programmpakete eine nur unwesentliche Verbesserung, bei individuellen Aufgaben dagegen einen echten Durchbruch.




PROGRAMM - PAKETE
Wiederkehrende Aufgaben
Technologisch orientiert
Anlagenunabhängig
Mittel zur Projektierung
Inbetriebnahme
Prozeßbedienung

INDIVIDUELLE PROGRAMME
Immer neue Aufgaben
Anlagenorientiert
Mittel zur Programmierung


Assembler
maschinennah
Prozeß-FORTRAN
höhere Sprache für mathematisch - technische rechenintensive Aufgaben mit Zugriff zu Prozeßdaten
PEARL
höhere Sprache für mathematisch - technische und prozeßnahe Aufgaben insbesonders Datenstrukturen, Koordinierung Zugriff zu Prozeßdaten
Projektierungsmittel auf der Basis konventioneller Techniken
Regeln
Steuern
Überwachen
Inbetriebnahme
Prozeßbedienung
Programmiersprachen
+
Technisches Know How
=
Technologische
Programmpakete

# Euphorische Aussagen

→ Softwarekosten
Sinken


JA — beim Programmieren

? — bei Aufgabenanalyse

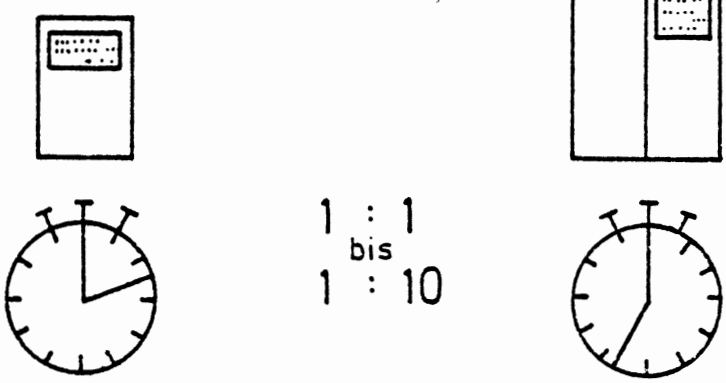
? — bei Systementwurf

? — bei Programmtest

? — bei Dokumentation

? — bei Inbetriebnahme

→ Speicherplatz
wenig mehr
Laufzeit

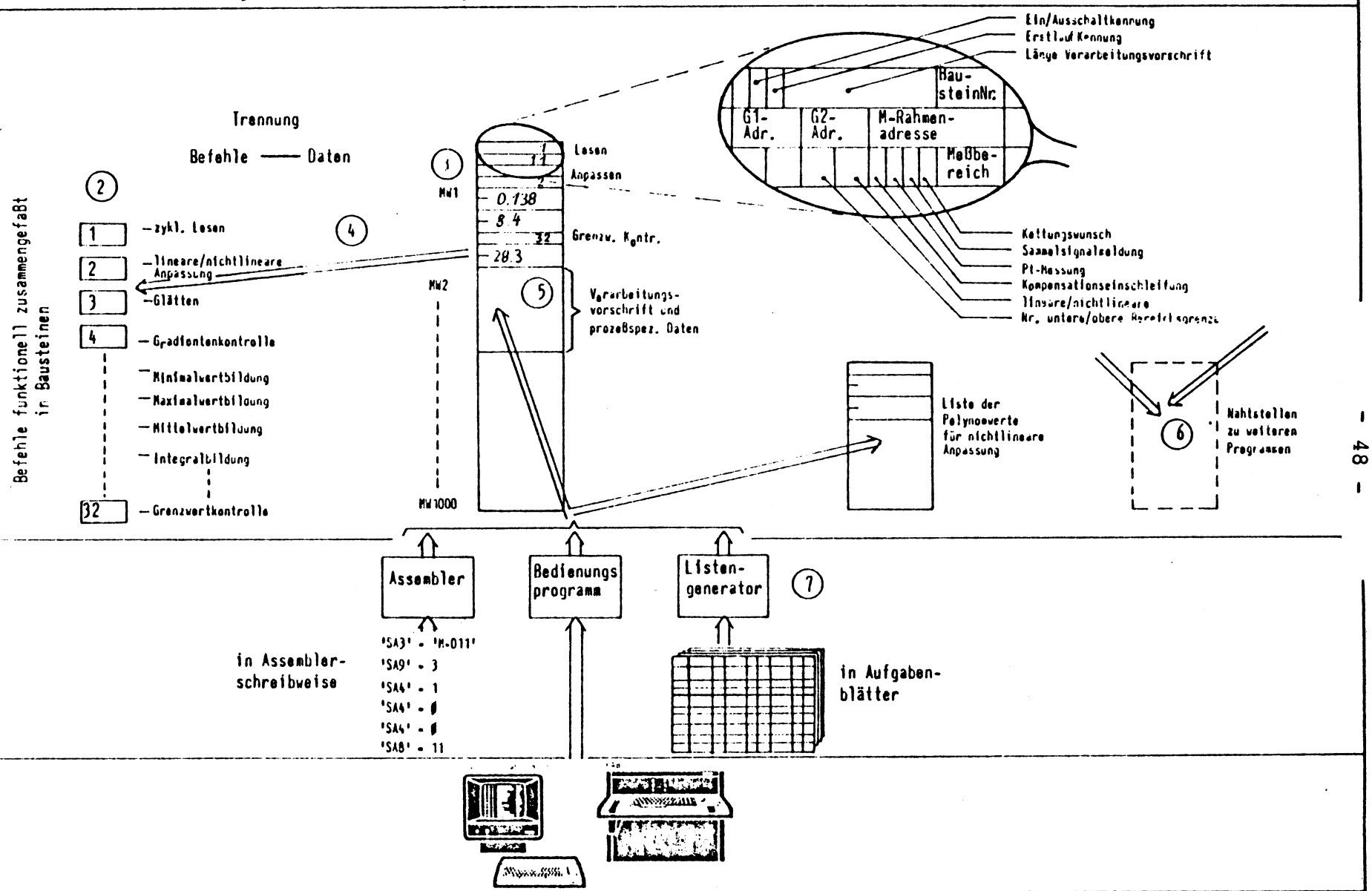



→ Portabilität

JA — theoretisch

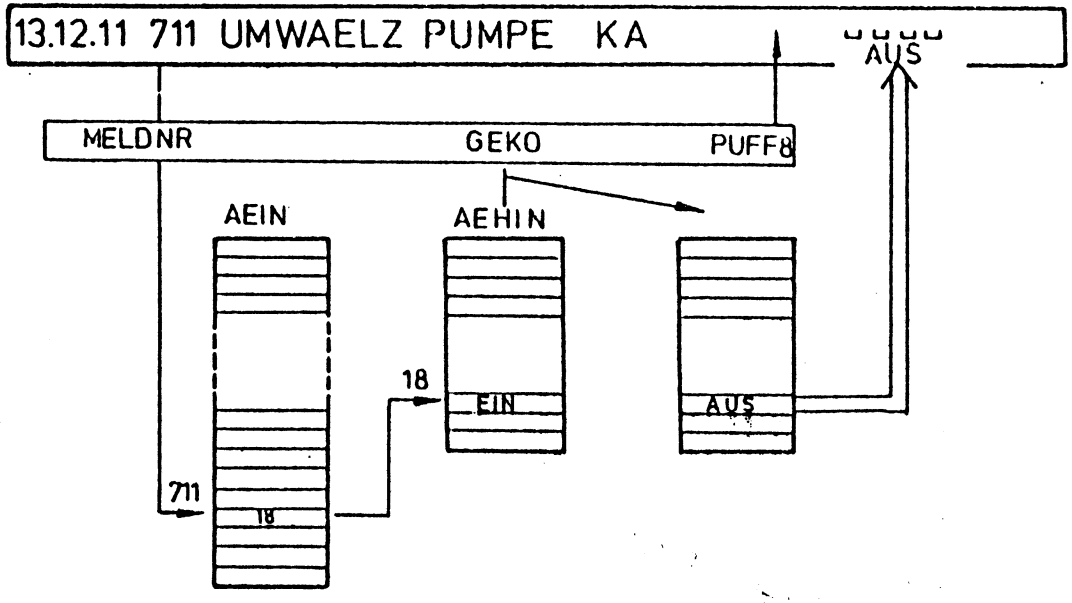
aber praktisch: Subset BBC ≠ ASME ≠ AEG ≠ SIEMENS ≠ mbp

| Aufgabenstellung | Umfangreiche Meßwertverarbeitung 10 - 1000 Meßwerte |
|---|---|
| Programmaufbau | |
| Vorgabe der prozeß-spezif. Daten (durch Projektierungs-ingenieur) | |
| Änderung der Daten auf der Anlage (Bedienung) | |




**E STE 3 Erforderliche Stuktur eines Programmpakets**

Aufgabe: Meldungsprotokollierung, Einfügen des Änderungshinweises



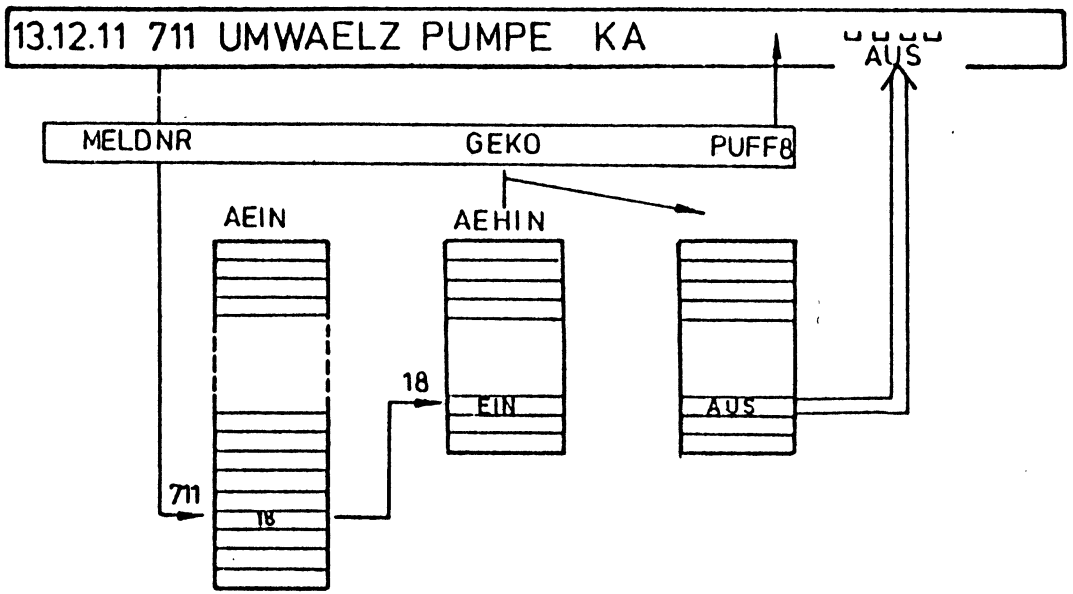

in PEARL

```
DCL  AEIND(1:1024) BIT(5);      /* INDIZESS DER AENDERGSPAARE */
DCL  AEHIN(0:31,0:1) CHAR(8);   /* AENDERUNGSHINWEISE */
                                /* FUELLUNG UEBER LISTENGENERATOR */

AEND: PROCEDURE (MELDNR,PUFF8,GEKO);
/* PROZEDUR ZUR FUELLUNG DES PUFFERS8 MIT AENDERUNGSHINWEIS */
/* IN ABHAENGIGKEIT VON DER MELDUNGSNUMMER UND DEM          */
/* GEHEN/KOMMEN (GEKO)-HINWEIS.                             */
/* DIE INDEXLISTE AEIND UND DIE ZWEIFACH-LISTE DER AENDERGS-*/
/* HINWEISE SIND GLOBALE GROESSEN.                          */
DCL MELDNR FIXED, GEKO BIT(1), PUFF8 CHAR(8);
PUFF8 := AEHIN ( FIX(AEIND(MELDNR)), FIX(GEKO));
END; /* ENDE DER PUFFERFUELLUNG MIT DEM AENDERUNGSHINWEIS */
```

in Assembler

```
                    *** AENDERUNGSHINWEIS-TEXT BEARBEITUNG
                    *** V: R3 ABLEGADR
'A'
'A'  AEND/    G1  :=  (MELDNR)-1
'A'           G1  :=  G1*5              *** BIT ADR
'A'           G0  :=  G1,V-4            *** DIV 16 WORTADR
'A'           G3  :=  (L'AEINADR)*G0    *** ADR INDEX
'A'           G2  :=  (G3')             *** INDEX UEBER
'A'           G3  :=  (G3)              *** 2 WORTE
'F'           G1  :=  G1,U15-11         *** VERSCH.NR
'A'           G2  :=  G2,D G1,U'H=001F' *** INDEX
'A'           G2  :=  G2,V3*(L'AEHINADR)*** INDEX *8 +ANFADR
'A'           G0  :=  4                 *** ANZ.DER UMSPEICH.WORTE
'B'           (0*GEKO(1))=0 ;SP AEND1   *** 1.ANDERUNGSHINWEIS
'A'           G2  :=  G2+4              *** 2.ANDERUNGSHINWEIS
'A'  AEND1/   G1  :=  (G2')
'A'           (R3');= G1
'A'           G0 ;DS AEND1
```




in PEARL

```
DCL  AEIND(1:1024) BIT(5);      /* INDIZESS DER AENDERGSPAARE */
DCL  AEHIN(0:31,0:1) CHAR(8);   /* AENDERUNGSHINWEISE */
                                /* FUELLUNG UEBER LISTENGENERATOR */

AEND: PROCEDURE (MELDNR,PUFF8,GEKO);
/* PROZEDUR ZUR FUELLUNG DES PUFFERS8 MIT AENDERUNGSHINWEIS */
/* IN ABHAENGIGKEIT VON DER MELDUNGSNUMMER UND DEM          */
/* GEHEN/KOMMEN (GEKO)-HINWEIS.                             */
/* DIE INDEXLISTE AEIND UND DIE ZWEIFACH-LISTE DER AENDERGS-*/
/* HINWEISE SIND GLOBALE GROESSEN.                          */
DCL MELDNR FIXED, GEKO BIT(1), PUFF8 CHAR(8);
PUFF8 := AEHIN ( FIX(AEIND(MELDNR)), FIX(GEKO));
END; /* ENDE DER PUFFERFUELLUNG MIT DEM AENDERUNGSHINWEIS */
```

in Prozess - FORTRAN

```
SUBROUTINE AEND (MELDNR,PUFF8,GEKO);
INTEGER *2 MELDNR,GEKO,W1,W2,MS,IND,AEIND, *4 WG
REAL *8 PUFF8,AEHIN
COMMON /GLOBAL/ AEIND(1000),AEHIN(32,2) /HILF/ W1,W2
EQUIVALENCE (W1,WG)
MS=(MELDNR-1)*5
WNR=MS/16
W1=AEIND(WNR+1)
W2=AEIND(WNR+2)
IND=WG.LSHIFT.(MOD(MS,16)-27),AND.'0001F'
PUFF8=AEHIN(IND+1,GEKO+1)
RETURN
END
```

SIEMENS

E STE 3 Programmieraufwand PEARL - Prozess FORTRAN (Beispiel)

Programm in PEARL

```
AEND: PPOCEDURE (MELDNR,PUFFR,GEKO);
/* PROZEDUR ZUR FUELLUNG DES PUFFERS8 MIT AENDERUNGSHINWEIS */
/* IN ABHAENGIGKEIT VON DER MELDUNGSNUMMER UND DEM          */
/* GEHEN/KOMMEN (GEKO)-HINWEIS.                             */
/* DIE INDEXLISTE AEIND UND DIE ZWEIFACH-LISTE DER AENDERGS-*/
/* HINWEISE SIND GLOBALE GNOESSEN.                          */
DCL MELDNR FIXED, GEKO BIT(1), PUFF8 CHAR(8);
PUFF8 := AEHIN ( FIX(AEIND(MELDNR)), FIX(GEKO));
```

aufgelöst

| |
|---|
| INDEX α : = FIX (GEKO); |
| INDEX β : = FIX (AEIND ( MELDNR)); |
| PUFF8 : = AEHIN (INDEX α , INDEX β ) |

abgesetzte Befehlsfolge
(in Assemblerschreibweise)

| |
|---|
| R2:=R2-4<br>GØ:=(GEKO)<br>(*R2):=GØ |
| GØ:=(MELDNR)<br>(*R2):=GØ<br>GØ:=AEINDFD<br>(*R2):=GØ<br>R7:=US FELDZUGRIFF |
| GØ:=AEHINFD<br>(*R2):=GØ<br>R7:=US FELDZUGRIFF |
| R5:=PUFF8ADR<br>R4:=4<br>R7:=US ZUWEIS3 |

UP FELDZUGRIFF *)

35 Befehle

UP FELDZUGRIFF *)

35 Befehle

UP ZUWEIS3

10 Befehle

*) Befehlsfolge wird nur 1x abgesetzt



E STE 3 abgesetzte Befehlsfolge bei PEARL (Beispiel)

SIEMENS

# Programmiersprachen im industriellen Einsatz von Prozeßrechnern

Referent: W.-D. Blaum

Verfasser: W.-D. Blaum / W. Körner

AEG-TELEFUNKEN

Fachbereich Prozeßtechnik

## 1. Einleitung

In der Anwendungs-Software-Abteilung des Fachbereiches Prozeßtechnik von AEG-Telefunken werden seit den Jahren 1975/1976 Untersuchungen angestellt, die bisher übliche Assemblerprogrammierung für Prozeßrechneranwendungen durch eine höhere Programmiersprache abzulösen. Im Laufe dieser Untersuchungen wurden folgende Programmiersprachen berücksichtigt:

1. Prozeß-FORTRAN [1]
2. PEARL [2]
3. AEG 80 Systemprogrammiersprache [3]

Schon in den Anfangsphasen der Untersuchungen bestätigten sich die erwarteten Vorteile von PEARL und AEG 80 Systemprogrammiersprache gegenüber Prozeß-FORTRAN in folgenden Eigenschaften:

1. Modulstruktur
2. Abschnittskonzept
3. Modegebundenheit durch explizite Deklaration
4. Datentypen für Bit-Größen, Referenzen und insbesondere für Strukturen
5. Dynamische Arrays
6. Große Sicherheit gegenüber Programmier- und Laufzeitfehler durch umfangreiche Prüfungen der Compiler.

Im Zuge der weiteren Untersuchungen wurden deshalb zunächst die beiden Sprachen PEARL und AEG 80-Systemprogrammiersprache für verschiedene Anwendungen getestet. Hier werden im folgenden zwei dieser Anwendungen vorgestellt, und zwar:

1. Eine Anwendung aus der Labor-Meßwerterfassung und -verarbeitung. [1]
   (Modellprozeß)
2. Der Analogwerterfassungs- und Verarbeitungsteil eines Standard-Software-Paketes für Anlagenüberwachung, Regelung, Steuerung und Informationsgewinnung.

Die Untersuchungen dieser beiden Anwendungen und ihre Ergebnisse werden im folgenden kurz skizziert.

## 2. Anwendungsbeispiele

### 2.1. Anwendungsbeispiel "Labormeßwerterfassung"

Die Labormeßwerterfassung wurde in unserem Prozeßrechenzentrum als Modellprozeß aufgebaut und getestet. Die Aufgabenstellung war wie folgt gewählt:

Zyklische Meßwerterfassung bis zu einer Maximalzahl von Meßzyklen, wobei Start und Ende der Erfassung durch Unterbrechungseingaben vorgegeben wird.

Die Meßwerte werden nach dem Erfassen konvertiert und zur Auswertung auf dem Externen Speicher abgelegt.

Nach dem Ende der zyklischen Erfassung schließt sich eine statistische Auswertung an. Außerdem werden Meßprotokolle erstellt und ausgegeben.

Bei der Formulierung dieser Aufgabe in PEARL wurde ein Subset benutzt, der für das kleinste Mitglied der AEG 80-Prozeßrechnerfamilie gebildet wurde. Einige wesentliche Erweiterungen dieses Subsets [6] gegenüber dem bekannten ASME I-PEARL-Subset [4] sind:

1. Strukturen als weiteren Datentyp
2. Datentyp für Referenz-Objekte
3. Beliebige dynamische Arraygrenzen
4. Typdeklaration
5. Operatordeklaration
6. Synchronisationsgrößen vom Typ BOLT
7. Dynamische Prioritätsangaben bei ACTIVATE
8. Beliebige Schedules bei allen Taskanweisungen
9. USING Tasksemaphor bei ACTIVATE

## 2.2. Anwendung "Analgwerterfassung und Verarbeitung mit einem Standard-Software-Paket

Bei der Analogwerterfassung und Verarbeitung innerhalb eines Standard-Software-Paketes für umfangreiche Prozesse in Kraftwerken, Chemie-Industrie und EVU-Netzautomatisierungen ist der Aufbau des Datenmodells leistungsbestimmend.

Das Datenmodell, Abbildung des technologischen Prozesses, besteht aus Auskunftselementen, die den Prozeßvariablen (Meßstellen) fest zugeordnet sind. Das einzelne Auskunftselement enthält die physikalische und technologische Beschreibung der Prozeßvariablen und ihrer Verarbeitung im Softwaresystem, z. B. Filterung, Konvertierung nicht linearer Geberkennlinien,

Extremwert-, Mittelwert-, Summenbildung, logische Verknüpfungen, Alarm- und Meldungsaufbereitung etc., sowie im Falle der Regelung weitere Angaben über Regelalgorithmen und Ausgabeprozeduren.

Die Länge eines Auskunftselementes für eine analoge Prozeßvariable reicht von ca. 30 Byte für einfache Meßstellen bis zu 200 Byte für umfangreiche Meßstellen mit anschließender Regelungsverarbeitung. Die Ausdehnung eines Auskunftselementes bleibt nicht immer konstant, sondern kann sich während der Lebensdauer des Systems verändern. Wird zum Beispiel eine analoge Meßstelle infolge einer Verfahrensänderung in technologischen Prozessen mit Regelung ausgestattet, so muß das Auskunftselement ergänzt werden. Die einfachste und häufigste Lösung bei der Datenstrukturierung der Auskunftselemente ist, diese für den maximalen Ausbau auszulegen. Damit erhält man ein "festes" Datenmodell. Bei Auskunftselementen, deren Meßstellen z. B. keine Regelungsverarbeitung erhalten, tritt hierbei ein Verschnitt von bis zu 60 % auf. Daraus ergibt sich, je nach Ausbau, ein Verschnitt von ca. 30 - 50 % der Gesamtausdehnung des Datenmodells. Mit einem festen Datenmodell ergibt sich allein für die ca. 500 - 1200 zyklisch zu erfassenden analogen Prozeßvariablen ein Datenmodellumfang von ca. 200 KByte.
Die einzelnen Auskunftselemente dieses Datenmodells müssen zyklisch vom Hintergrundspeicher in den Arbeitsspeicher transferiert werden. Aus unseren Erfahrungen mit diesen Anwendungen zeigte sich, daß bei solchen Systemen die Grenze der Leistungsfähigkeit nicht in der Verarbeitungseinheit, sondern an der Transferkanalbelastung liegt. Eine Folge der noch zu langsamen Hintergrundspeicher.

Um eine Entlastung des Transferkanals zu erzielen, ist die Verminderung des Datenmodellumfangs und der gruppenweise Transfer in möglichst großen Datenmodellsätzen (ca. 4 - 12 KByte) erforderlich.

Die Verminderung des Datenmodellumfangs erzielt man mit einer variablen Strukturierung der Auskunftselemente. Hierbei hat jedes Auskunftselement nur die Größe seiner zum Zeitpunkt tatsächlichen Belegung und kann jederzeit parallel zum Prozeß beliebig erweitert und verändert werden.

Bei diesem variablen Datenmodellaufbau besteht ein Auskunftselement aus einem festen Kopfteil und der augenblicklichen Belegung. Diese entspricht N verschiedene Datenstrukturen. Die festen Kopfteile werden vom Anfang des Datenmodellsatzes aus abgelegt. Die Datenstrukturen aller Auskunftselemente liegen ungeordnet am Ende des Datenmodellsatzes.

Wird nun während des Betriebes eine Meßstelle und damit ein Auskunftselement um eine Verarbeitungsfunktion erweitert, so wird am entsprechenden Platz in dem Kopfteil nur der Zeiger auf den Freibereich eingetragen und die dazugehörigen Verarbeitungsdaten dementsprechend abgelegt.

Bei der Verarbeitung der Meßstelle läßt sich aus der Stellung des Zeigers innerhalb des Kopfteiles die dazugehörige Verarbeitungsprozedur ermitteln. Als Parameter wird dieser Prozedur nur der Zeiger auf die Verarbeitungsdaten mitgegeben.

## 3. Erkenntnisse aus der Realisierung der beiden Anwendungsbeispiele

### 3.1. Zur "Labormeßwerterfassung"

Die vorgegebene Aufgabenstellung konnte ohne Schwierigkeiten mit allen drei Programmiersprachen formuliert werden. Bei der Realisierung wurden nur die schon bekannten "natürlichen" Vorteile der einzelnen Programmiersprachen bestätigt.

Durch den PEARL-Systemteil und die wirkungsvollen Statements für Prozeß EA ergab sich bei PEARL eine kurze Quelle. Bei der AEG-80-Systemprogrammiersprache mußte dies durch den Aufbau der Strukturen als Parameter für die Systemdienstaufrufe realisiert werden. Dieser Nachteil der längeren Quelle und die erforderliche Kenntnis der Systemdienstschnittstellen wird durch eine kleinere Laufzeitorganisation für die AEG 80 Systemprogrammiersprache wieder aufgehoben.

### 3.2. Zur "Analogwerterfassung und Verarbeitung in einem Standard-Software-Paket"

Die optimale Speicherablage eines variablen Datenmodells ist mit Hilfe bekannter Algorithmen in der Assembler-Programmierung beherrschbar.

Bei der Behandlung des variablen Datenmodells mit einer höheren Programmiersprache ergibt sich die Schwierigkeit, daß sich solche variablen Datenmodelle nicht, oder nur sehr schwierig beschreiben lassen.

Ein Ausweg bei der Anwendung mit PEARL ist, daß der Datenmodellsatz als FIXED-Array erklärt wird, und bei jedem Zugriff zu den einzelnen Daten bzw. Datenstrukturen des Auskunftselementes eine Modewandelnde Operation eingesetzt wird.
Dieses Verfahren ist aber unbefriedigend, da sich z. B. keine Struktur- oder Array-Operationen hierauf anwenden lassen. Außerdem wird das Dokumentationsvermögen des PEARL-Programms hierdurch vermindert.

Vorteilhafter wäre hier ein Verfahren, wie es z. B. die Sprachmittel der AEG 80 Systemprogrammiersprache anbietet. Dem Zugriffsprogramm ist es in einem solchen Falle möglich, aus dem Index des FIXED-Array und dem Mode der Datenstruktur eine gültige Referenz zu bilden. Mit Hilfe dieser Referenz kann jetzt eine modegerechte Verarbeitung der Daten erfolgen. [5] Der Zugriff erfolgt über die gebildete Referenz mit dem zur Compilzeit abprüfbaren Mode. Dieses Verfahren ist auch in anderen Sprachen bekannt. Aus Anwendersicht wäre es wünschenswert, wenn dieses Verfahren auch in PEARL ermöglicht wird.

## 4. Schlußbemerkung

Aus den vorgestellten Ergebnissen und weiteren Erfahrungen ergibt sich bei der Auswahl einer Programmiersprache für die Erstellung von Anwendungssoftware bei uns folgendes:

Für die Aufgaben, bei denen es insbesondere auf die individuelle ingenieurmäßige Lösung des technischen Problems ankommt, wird in Zukunft neben FORTRAN verstärkt PEARL eingesetzt. Rationelle Softwareproduktion und verbesserte Dokumentation sowie der bessere Sprachumfang sind hier hervorzuheben.

Bei der Erstellung unserer großen Standard-Software-Pakete, die die volle Leistungsfähigkeit unserer Hardwaresysteme und der Systemsoftware gezielt ausnutzen sollen und bei denen Flexibilität für die speziellen Anpassungen erforderlich ist, werden wir zur Zeit auf die AEG 80 Systemprogrammiersprache, und/oder evtl. auch noch auf die Assemblersprache noch nicht verzichten können. [5]

Als PEARL-Implementator wird AEG-Telefunken die aus dem praktischen Einsatz gewonnenen Anregungen in den PEARL-Gremien vertreten.

# LITERATUR

[1] Prozess-FORTRAN 75
Eine Erweiterung von FORTRAN für Prozessrechner-Anwendungen.

Vorschlag des Arbeitskreises "Prozess-FORTRAN" der VDI/VDE-Gesellschaft für Mess- und Regelungstechnik vom 5. 12. 1975.

[2] PEARL
PDV-Bericht KFK-PDV 73, April 1973,
Gesellschaft für Kernforschung mbH, Karlsruhe

[3] D. Dürr
SL3 - eine Systemprogramiersprache auf ALGOL-68-Basis als Grundsprache für die Prozeßrechnerfamilie AEG 80
Angewandte Informatik 9/1975, Seite 393 bis 399

[4] ASME-PEARL-SUBSET/1
PDV-Bericht KFK-PDV 76, Mai 1976,
Gesellschaft für Kernforschung mbH, Karlsruhe

[5] R. Emge
Maschinennahe Struktur von Standard-Software und deren Behandlung mittels der höheren prozedurorientierten Sprache SL3.
In Applied Computer Sciene, Seite 87 bis 98, Band 3
Carl Hauser-Verlag München/Wien

[6] Eichentopf
AEG 80-20 PEARL
Interne Entwicklungsbeschreibung AEG-Telefunken

Tagungsband zum Aussprachetag PEARL in Augsburg, März 1977

Gesellschaft für Kernforschung mbH, Karlsruhe
KFK-PDV 11o, März 1977

Die Gesellschaft für Kernforschung mbH, Karlsruhe, Projektleitung PDV, veranstaltet gemeinsam mit der VDI/VDE-Gesellschaft Meß- und Regelungstechnik am 9. März 1977 einen öffentlichen Aussprachetag über PEARL, die einheitliche Echtzeitprogrammiersprache in der Bundesrepublik Deutschland. Der vorliegende Bericht enthält Ausarbeitungen der über folgende Themen gehaltenen Vorträge:

- Die Förderung von PEARL im Projekt PDV (Historie, Stand, Zukunft, Standardisierung),
- PEARL im Vergleich mit anderen Echtzeitsprachen,
- Erfahrungen mit bisherigen PEARL-Subsets (an Modellprozessen und im industriellen Einsatz).

Tagungsband zum Aussprachetag PEARL in Augsburg, März 1977

Gesellschaft für Kernforschung mbH, Karlsruhe
KFK-PDV 11o, März 1977

Die Gesellschaft für Kernforschung mbH, Karlsruhe, Projektleitung PDV, veranstaltet gemeinsam mit der VDI/VDE-Gesellschaft Meß- und Regelungstechnik am 9. März 1977 einen öffentlichen Aussprachetag über PEARL, die einheitliche Echtzeitprogrammiersprache in der Bundesrepublik Deutschland. Der vorliegende Bericht enthält Ausarbeitungen der über folgende Themen gehaltenen Vorträge:

- Die Förderung von PEARL im Projekt PDV (Historie, Stand, Zukunft, Standardisierung),
- PEARL im Vergleich mit anderen Echtzeitsprachen,
- Erfahrungen mit bisherigen PEARL-Subsets (an Modellprozessen und im industriellen Einsatz).

Proceedings of a Symposium about PEARL in Augsburg, March 1977

Gesellschaft für Kernforschung mbH, Karlsruhe
KFK-PDV 11o, March 1977

Both, the PDV Project Management of Gesellschaft für Kernforschung mbH, Karlsruhe, and the VDI/VDE-Gesellschaft Meß- und Regelungstechnik, are organizing on March 9, 1977, a public symposium about PEARL, the unified realtime programming language in the Federal Republic of Germany. This report contains papers of speeches given at the symposium about the following topics:

- The sponsoring of PEARL within the Project PDV (history, status, future, standardization),
- PEARL as compared with other realtime languages,
- Experiences with present PEARL subsets (both in laboratory and industrial environment).

Proceedings of a Symposium about PEARL in Augsburg, March 1977

Gesellschaft für Kernforschung mbH, Karlsruhe
KFK-PDV 11o, March 1977

Both, the PDV Project Management of Gesellschaft für Kernforschung mbH, Karlsruhe, and the VDI/VDE-Gesellschaft Meß- und Regelungstechnik, are organizing on March 9, 1977, a public symposium about PEARL, the unified realtime programming language in the Federal Republic of Germany. This report contains papers of speeches given at the symposium about the following topics:

- The sponsoring of PEARL within the Project PDV (history, status, future, standardization),
- PEARL as compared with other realtime languages,
- Experiences with present PEARL subsets (both in laboratory and industrial environment).