P U B S

PEARL-orientiertes, universelles Binder-System

Jörg Grössler

TECHNISCHE UNIVERSITÄT BERLIN
Prozeßrechnerverbund-Zentrale

21.7.1980

ZUSAMMENFASSUNG

Das PEARL-orientierte, universelle Binder-System (PUBS) ist ein Hilfsmittel bei der PEARL-Programmentwicklung. Seine Aufgaben lassen sich wie folgt zusammenfassen:

- *Binden mehrerer getrennt übersetzter PEARL-Module und Überprüfung der Konsistenz der Modul-Schnittstellen.*
- *Anlegen von Systemtabellen für das PEARL-Betriebssystem.*
- *Zuordnung von Code- und Datenbereichen des PEARL-Programmes zu Speicherbereichen des Zielrechners.*

PUBS entsteht im Rahmen des Projekts "Einführung und Anwendung der Programmiersprache PEARL in verschiedenen Anwendungsbereichen der TU-Berlin".

HINTERGRUND/SITUATION

Der vorliegende Bericht beschreibt ein Teilprojekt eines Vorhabens, das sich mit der Einführung von PEARL in den Prozeßrechnerverbund der TU-Berlin beschäftigt. Ein wesentliches Merkmal dieses Vorhabens ist eine Aufgabenteilung bei der Implementierung des PEARL-Programmiersystems: Zu einem in Auftrag gegebenen Compiler wird an der TU-Berlin eine Betriebssystemeinbettung (im Folgenden kurz 'PEARL-Kernel' genannt) erstellt.

Das einzuführende PEARL-Programmiersystem sollte die Modularität von PEARL voll unterstützen. Insbesondere sollte ein Zusammenbinden getrennt übersetzter PEARL-Module möglich sein, ohne dabei auf die Schnittstellenüberprüfung zwischen den Modulen zu verzichten.

Ein weiteres Problem stellt die Schnittstelle zwischen PEARL-Compiler und PEARL-Kernel dar. Der Kernel arbeitet mit Verwaltungsstrukturen (Systemtabellen),deren Anzahl und Initialisierung vom jeweiligen PEARL-Programm abhängen (Anzahl der TASKs, SEMAs, usw.). Legt der Compiler diese Tabellen an (wie in vielen Implementationen üblich), muß bei jeder Änderung oder Erweiterung der Tabellen der Compiler ebenfalls geändert werden. Dies ist besonders bei einer Aufgabenteilung zwischen Compiler- und Kernelerstellung mit erheblichem Mehraufwand verbunden.

Ähnliches gilt für ein Problem, das im Folgenden kurz 'Speicherverwaltung' genannt wird. Hierbei geht es um die Anordnung der vom Compiler erzeugten Code- und Datenbereiche im Hauptspeicher des Zielrechners. Die Entscheidung über diese Anordnung ist Ergebnis der Kernel-Implementation. Im vorliegenden Fall, in dem eine Anpassung der Kernel-Funktionen an ein bereits vorhandenes Realtime-Betriebssystem vorgenommen wird, sind einige Kriterien der Anordnung von Code- und Datenbereichen von diesem bereits vorhandenen System vorgegeben. Es erscheint daher auch hier nicht sinnvoll, wenn der Compiler diese Aufgabe übernimmt.

Aufgrund dieser Anforderungen entsteht ein System zwischen PEARL-Compiler und darunterliegendem Betriebssystem (d.h. hier dem Binder des vorhandenen Systems), das durch die Anpassung beider Systeme ein Höchstmaß an Komfort bei der Erstellung komplexer Systeme in PEARL anbietet und 'PEARL-orientiertes, universelles Binder-System (PUBS)' genannt wird.

BESCHREIBUNG DES BINDER-SYSTEMS

Um die drei Aufgaben

- Schnittstellenüberprüfung
- Anlegen von Systemtabellen
- Speicherverwaltung

erfüllen zu können, muß dem Binder zusätzliche Information zur Verfügung gestellt werden.

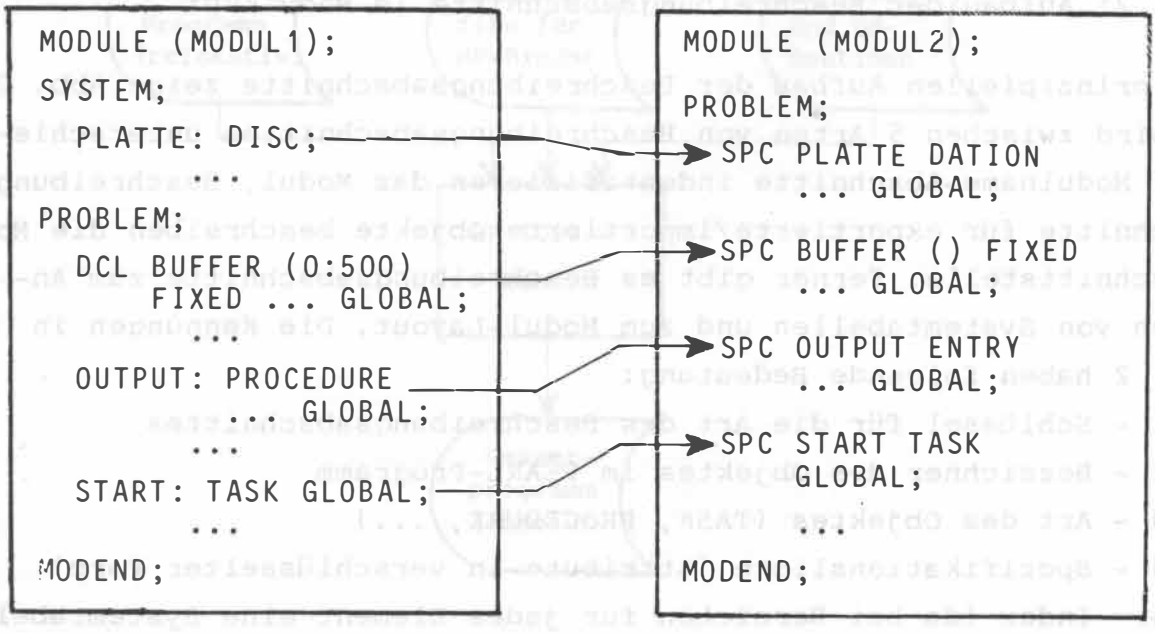

Abb. 1: Import/Export-Beziehung zwischen PEARL-Modulen

Dies hat zur Einführung einer neuen Art 'relokativer Programm-Module' als Ausgabe des PEARL-Compilers geführt. Diese relokativen Programm-Module bestehen aus einem Modulkopf und einem Rumpf. Im Modulkopf wird dem PEARL-Binder in sog. 'Beschreibungsabschnitten' Information über Modulschnittstellen, anzulegende Systemtabellen und die Speicherverwaltung mitgeteilt. Der Modulrumpf enthält den relokativen Code des Moduls.

Abb. 1 zeigt den Mechanismus von Export und Import globaler Objekte in PEARL. Zur Überprüfung der Modulschnittstellen enthalten die Beschreibungsabschnitte im Modulkopf die in den Deklarationen bzw. Spezifikationen enthaltenen Attribute in verschlüsselter Form.

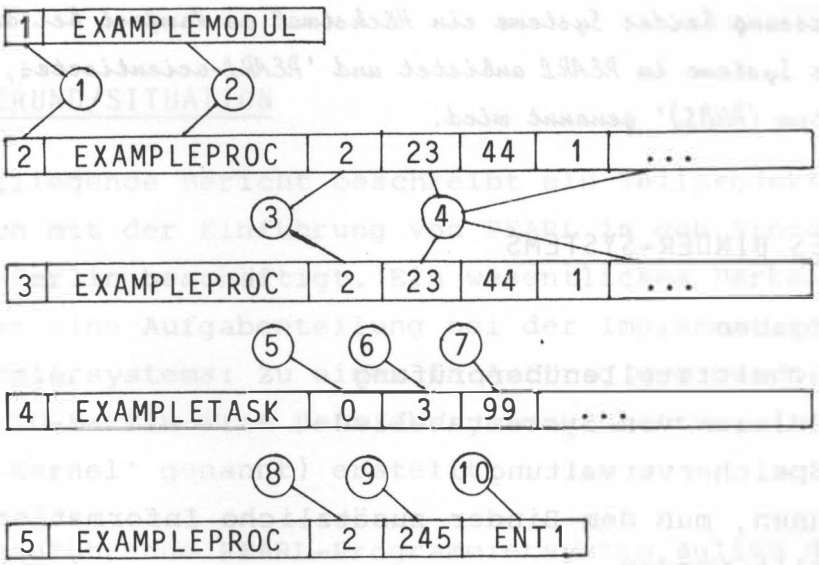

Abb. 2: Aufbau der Beschreibungsabschnitte im Modulkopf

Den prinzipiellen Aufbau der Beschreibungsabschnitte zeigt Abb. 2. Es wird zwischen 5 Arten von Beschreibungsabschnitten unterschieden: Modulname-Abschnitte indentifizieren das Modul, Beschreibungsabschnitte für exportierte/importierte Objekte beschreiben die Modulschnittstelle. Ferner gibt es Beschreibungsabschnitte zum Anlegen von Systemtabellen und zum Modul-Layout. Die Kennungen in Abb. 2 haben folgende Bedeutung:

1 - Schlüssel für die Art des Beschreibungsabschnittes
2 - Bezeichner des Objektes im PEARL-Programm
3 - Art des Objektes (TASK, PROCEDURE, ...)
4 - Spezifikationsliste (Attribute in verschlüsselter Form)
5 - Index (da bei Bereichen für jedes Element eine Systemtabelle)
6 - Art der Systemtabelle (TASK, SEMA, SIGNAL, DATION, ...)

7 - Initialisierung der Systemtabelle (hier z.B. Priorität)
8 - Index (bei Bereichen wird jedes Element getrennt angespr.)
9 - Länge des Codebereiches
10 - Identifikationsadresse (symbolische Adresse im Zielrechner)

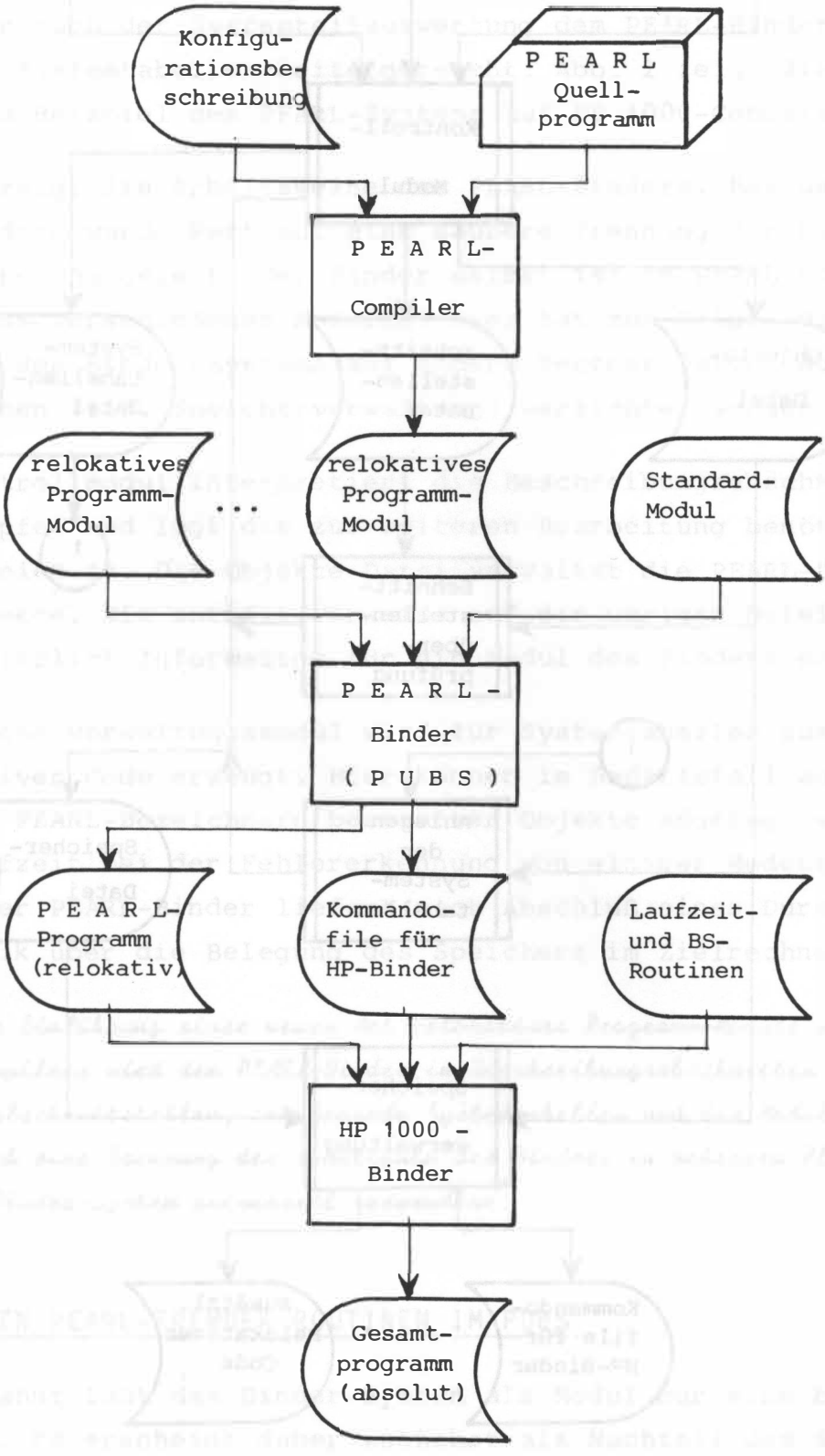

Abb. 3: Aufbau eines PEARL-Programmiersystems mit PUBS

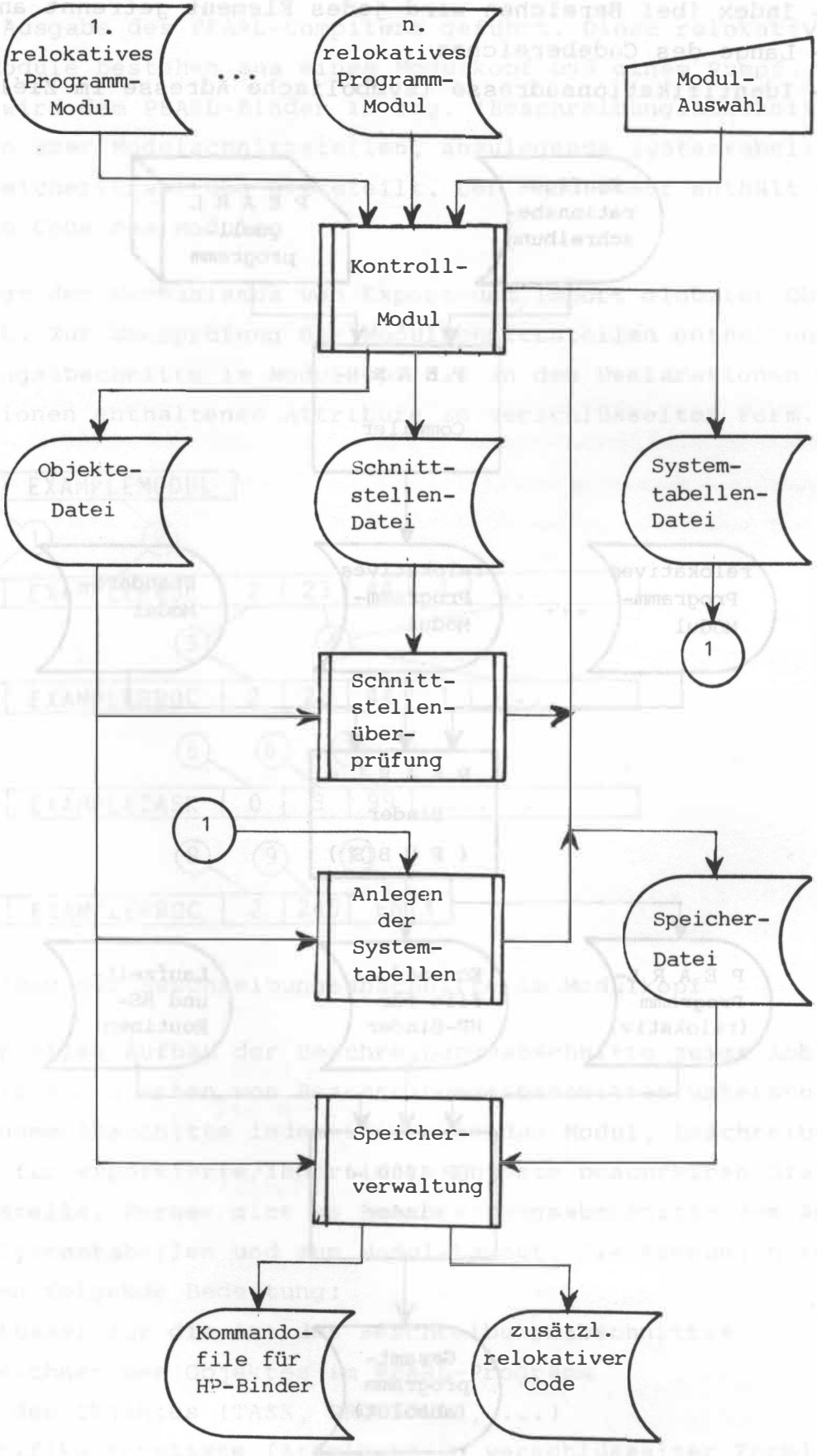

Abb. 4: Aufbau des PEARL-Binders

Abb. 3 verdeutlicht die Stellung des Binder-Systems im Ablauf der PEARL-Programmentwicklung. Alle dem PEARL-Binder als Eingabe dienenden Dateien müssen relokative Programm-Module im Sinne der eben beschriebenen sein. Die in der Konfigurationsbeschreibung enthaltene Information über Geräte, Interrupts und Signale werden vom Compiler nach der Systemteilauswertung dem PEARL-Binder zum Anlegen der Systemtabellen weitergereicht. Abb. 2 zeigt die Verhältnisse am Beispiel des PEARL-Systems auf HP 1000-Computern.

Abb. 3 zeigt die Arbeitsweise des PEARL-Binders. Bei der Erstellung des Binders wurde Wert auf eine saubere Trennung der einzelnen Aufgabenbereiche gelegt. Der Binder selbst ist in PEARL codiert und besteht aus verschiedenen Modulen. Dies hat zur Folge, daß bei Übertragung des Binder-Systems auf andere Rechner leicht auf eine der Funktionen (z.B. Speicherverwaltung) verzichtet werden kann.

Das Kontrollmodul interpretiert die Beschreibungsabschnitte in den Modulköpfen und legt die zur weiteren Bearbeitung benötigten internen Dateien an. Die Objekte-Datei verwaltet die PEARL-Bezeichner der Objekte. Sie enthält Verweise auf die übrigen Dateien, in denen ausschließlich Information für ein Modul des Binders enthalten ist.

Im Speicherverwaltungsmodul wird für Systemtabellen zusätzlicher relokativer Code erzeugt. Hier können im Bedarfsfall auch Tabellen mit den PEARL-Bezeichnern bestimmter Objekte angelegt werden, was zur Laufzeit bei der Fehlererkennung von einiger Bedeutung sein kann. Der PEARL-Binder liefert nach Abschluß eines Durchlaufes eine Statistik über die Belegung des Speichers im Zielrechner.

Durch die Einführung einer neuen Art relokativer Programm-Module als Ausgabe des PEARL-Compilers wird dem PEARL-Binder in Beschreibungsabschnitten Information über Modulschnittstellen, anzulegende Systemtabellen und das Modul-Layout gegeben. Durch eine Trennung der Funktionen des Binders in mehreren PEARL-Modulen ist das Binder-System universell verwendbar.

EINBINDEN PEARL-FREMDER ROUTINEN IM PUBS

Wie erwähnt läßt das Binder-System als Modul nur eine bestimmte Form zu. Es erscheint daher zunächst als Nachteil des Systems, daß auf diese Weise externe Routinen, die z.B. im Assembler des Zielrechners vorliegen, nicht ansprechbar sind.

Dieses Problem ist jedoch - unter Beibehaltung des Prinzips der korrekten Schnittstellenbeschreibung - in der Weise lösbar, daß die zum Binden notwendig Modulköpfe getrennt erzeugt werden.

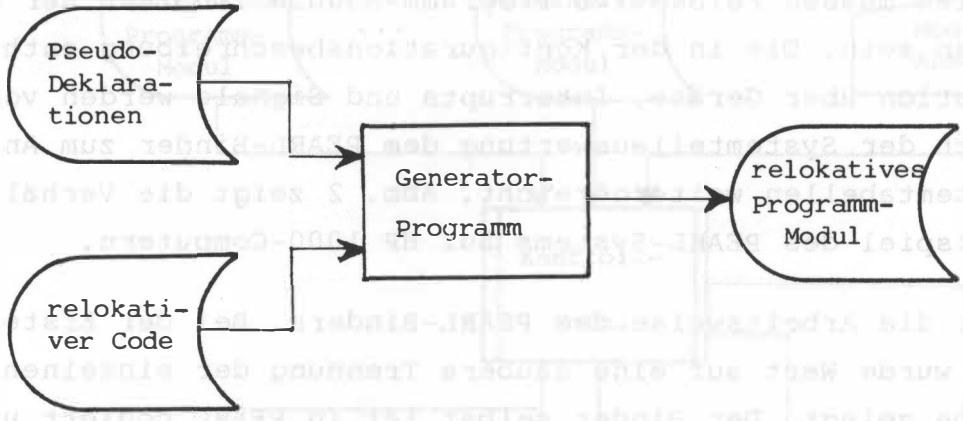

Abb. 5: Umwandlung externer Routinen in relokative Programm-Module

Zu diesem Zweck wurde ein Generator-Programm erstellt, das bestehende Routinen in relokative Programm-Module umwandelt. Als Eingabe dient diesem Generator (Abb. 5) eine Modulbeschreibung, die in PEARL-ähnlicher Form Pseudo-Deklarationen von Objekten und Prozeduren mit Angabe der symbolischen Adresse im Zielrechner enthält. Aus diesen Pseudo-Deklarationen werden dann Beschreibungsabschnitte für den neu erstellten Modulkopf erzeugt.

Es ist darauf zu achten, daß auf diese Weise nur eine bestimmte Art von Objekten in andere PEARL-Module importiert werden kann. (der Import erfolgt durch Spezifikation mit GLOBAL-Attribut). Dies reicht jedoch aus, um bereits vorhandene Software (z.B. Plot-Software) in das PEARL-System zu integrieren.