EIN VERFAHREN FÜR DEN GROBENTWURF VON PEARL-PROGRAMMEN

von L. Frevert, Bielefeld

Zusammenfassung

Der interaktive Entwurf von PEARL-Programmen wird durch ein PEARL-Programm unterstützt. Während ein Programm entwickelt wird, wird seine Struktur in einer Hierarchie von "Verdrahtungsplänen" gezeigt, deren Layout so gewählt wurde, daß sie gut mit Schnelldrucker oder Terminal gezeichnet werden können. Als Kästchen gezeichnete Subsysteme werden durch Informationskanäle verbunden und dann weiter in Subsysteme aufgeteilt. Es gibt spezielle Subsystem-Typen analog zu PEARL-Objekten (Module, Tasks, Dations usw.) und spezielle Informationskanal-Typen (Prozedur-Aufrufe, Task-Aktivierungen, Einplanungen, Interrupts, Release-Request usw.). Auch Informations-Kanäle können weiter verfeinert werden, indem Teilkanäle eingeführt werden (analog zu Bussen oder Kabelbäumen). Das endgültige Dokument wird mit Querverweisen versehen, um die Lesbarkeit zu erhöhen.

Schlüsselwörter: Programmentwurf, Top-down-Entwurf, Dokumentation.

Summary

Interactive top down design of PEARL programs is aided by a program written in PEARL. While a program is designed, its structure is shown by a hierarchy of "wiring diagrams" with a layout well fitted to be depicted by line printer and terminal. Subsystems depicted by blocks are connected by information channels and parted into further subsystems. There are special types of subsystems in analogy to PEARL objects (modules, tasks, dations etc.) and special types of information channels (procedure calls, task activation, schedules, interrupts, release-request etc.). Information channels may be refined too, by introducing subchannels (analog to busses or cable harnesses). The final document is supplied with references to enhance readability.

Keywords: Program design, top-down-design, documentation.

1. Einleitung

Jeder Ingenieur kennt die Worte: "Ein Bild sagt oft mehr aus als tausend Wörter" und "Konstruieren bedeutet Radieren". Die letzte Aussage verliert jedoch zunehmend an Gültigkeit: konstruiert wird nicht mehr mit Bleistift und Radiergummi, sondern am Bildschirm eines Rechners.

Es gibt eine ganze Anzahl verschiedener Verfahren für Spezifikation, Entwurf und Dokumentation von Software. Wenn man diesbezügliche Zusammenfassungen durchsieht (/1/) (/2/), fällt auf, daß die verwendeten Graphiken ihre Herkunft aus Handzeichnungen nicht leugnen können. Das bringt einen Nachteil mit sich: entweder sehen die Bilder aus wie ein Notbehelf, weil nämlich die Linienzüge von Handzeichnungen nur unvollkommen mit Schnelldruckern nachgebildet werden können, oder man muß teure Graphik-Bildschirme und Plotter verwenden, wobei letztere den zusätzlichen Nachteil haben, daß sie vergleichsweise langsam arbeiten. Ein weiterer Nachteil dieser aus Handzeichnungen abgeleiteten Darstellungen liegt darin, daß sie meist keine Rücksicht auf die durch Papierformate auferlegten Beschränkungen nehmen.

Die Möglichkeit, Graphiken auf sehr einfache Weise ändern zu können, scheint dem Spieltrieb des Menschen sehr entgegen zu kommen. Sie regt dazu an, sozusagen "spielerisch" nach besseren Strukturierungen zu suchen und hat bei der Anwendung für den Grobentwurf deshalb auch bessere Programme als Ergebnis. Insofern dürften Verfahren, bei denen primär Texte eingegeben und Graphiken erst nachträglich als Option erzeugt werden, nicht so günstig abschneiden.

Praktisch alle Entwurfsverfahren für Software sind so konzipiert, daß sie unabhängig von der zur Kodierung zu verwendenden Sprache sind. Das muß jedoch insbesondere beim Entwurf von Echtzeit-Programmen zu Schwierigkeiten führen, denn man wird wegen der völlig unterschiedlichen Koordinations-Mechanismen das Tasking von Ada-Programmen anders entwerfen müssen, als dasjenige von PEARL-Programmen. Um noch ein Beispiel zu nennen: Ein Programmentwurf, der keine Rücksicht auf die Modul-Struktur von PEARL-Programmen nimmt, kann das Modul-Konzept von PEARL nicht optimal nutzen; wie die Erfahrung zeigt, ist die vernünftige Aufteilung eines größeren Programmes in Module eine sehr anspruchsvolle Aufgabe.

Da an der Fachhochschule Bielefeld ohnehin kein Programmpaket für den Grobentwurf von Software zur Verfügung stand, wurde beschlossen, eines zu entwickeln, das die oben angedeuteten Nachteile vermeidet.

2. Grundsätzliche Forderungen

Der Ablauf eines Programmes stellt einen technischen Prozeß dar, bei dem Informationen transportiert und umgewandelt werden. Die eigentliche Aufgabe des Entwerfens besteht deshalb darin, das Programm in Teilsysteme zu gliedern, die durch Informationskanäle verbunden sind, und zu beschreiben, welche Informationen in den Informationskanälen übertragen und wie sie in den Teilsystemen umgewandelt werden. Um dabei Übersichtlichkeit zu gewährleisten, ist es unumgänglich, Teilsysteme weiter in Teilsysteme zu gliedern; dasselbe gilt für die Informationskanäle. Mit anderen Worten: sowohl Teilsysteme als auch Informationskanäle müssen schrittweise verfeinert werden.

Ein Programmentwurf besteht daher notwendigerweise aus einer Hierarchie von graphischen Darstellungen und beschreibenden Texten. Um in den Texten auf die Teile der Darstellungen Bezug nehmen zu können, müssen letztere benannt sein. Die hierarchische Struktur des Entwurfes muß außerdem so auf die lineare Struktur eines gedruckten Dokumentes abgebildet werden, daß der Leser die Übersicht behalten und leicht von einer Stelle des Hierarchiebaumes zu einer anderen findet.

Layout der Graphiken

Graphische Darstellungen auf den Bildschirmen normaler Terminals dürfen nicht breiter sein als 80 Zeichen; die andere Dimension darf jedoch beliebig groß sein, wenn die Möglichkeit besteht, Ausschnitte anzuwählen (beim Druck müssen dann jedoch die Anschlüsse leicht von Seite zu Seite zu verfolgen sein). Waagerechte und senkrechte Linienzüge werden in ihnen traditionsgemäß durch Bindestriche bzw. Ausrufezeichen oder Doppelpunkte dargestellt; es stellt jedoch keinen zusätzlichen Aufwand dar, anstelle dieser Zeichen Buchstaben oder Ziffern zu verwenden, die dann zusätzliche Informationen liefern können.

B i l d 1 zeigt die Darstellung des Teilsystems POET aus einem Systementurf (PEARL-orientierte Entwurfs-Technik). Es enthält weitere Teilsysteme; die Umrandung mit S sagt aus, daß es sich um Systeme im allgemeinen Sinn handelt, nicht etwa um Tasks oder Prozeduren oder Module, die mit T bzw. P oder M umrandet wären. Die senkrechten "Linien" links von den Untersystemen stellen Informationskanäle dar, deren Bezeichnungen unten mit entsprechender Staffelung stehen. Die Ziffern in den Linien dienen als "Farben". Die Untersysteme haben "Klemmleisten"; die Zeichen & bzw. * deuten an, daß es sich um einen Quellpunkt handelt, an dem das Untersystem Information in den betreffenden Kanal einspeist. Wenn auf dem Schnittpunkt

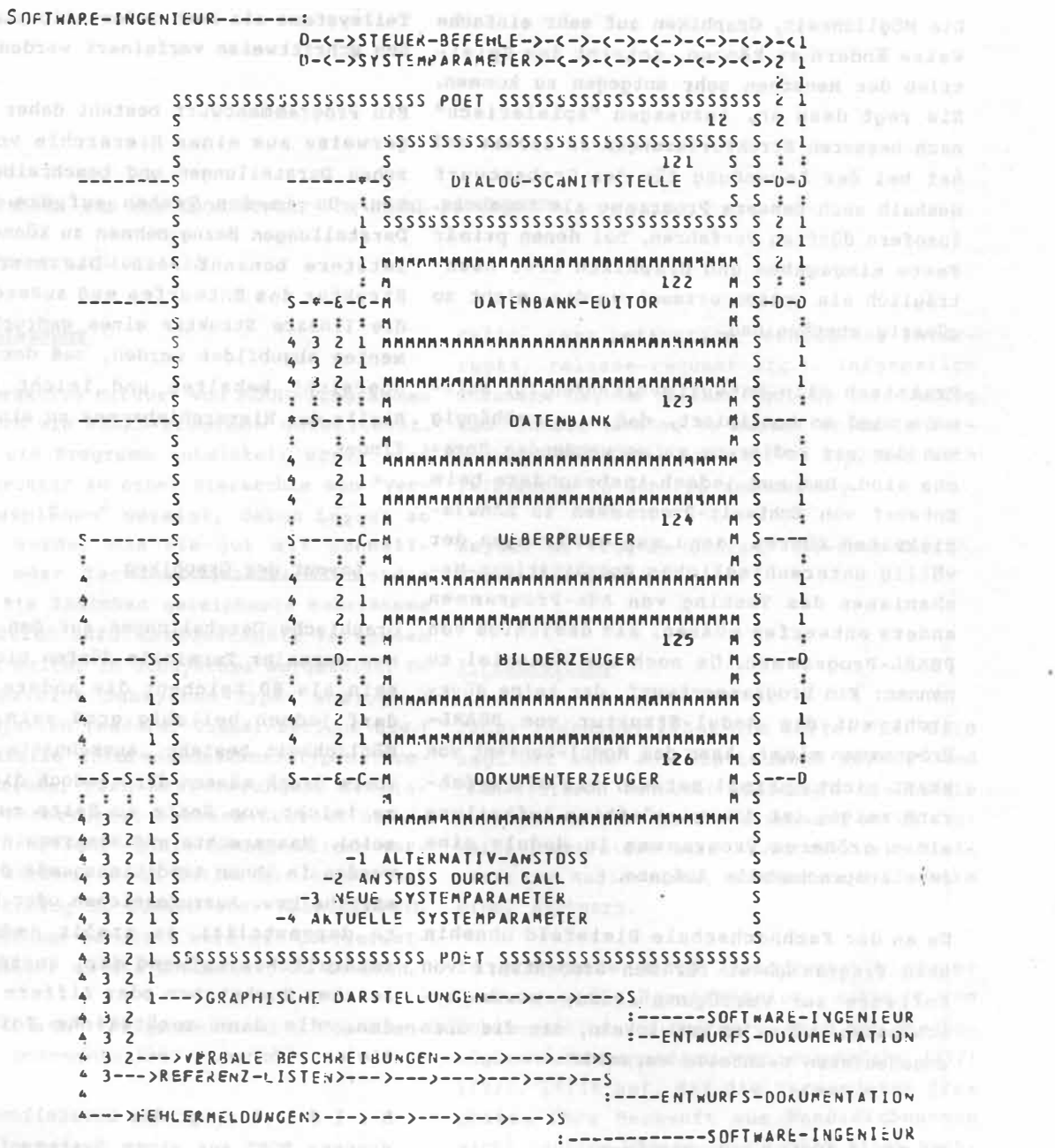

Bild 1: Struktur des POET-Programmes. Es besteht aus dem Teilsystem Dialog-Schnittstelle und den Moduln Datenbank-Editor usw.. Der Informationsaustausch geschieht mit den Kanälen Alternativ-Anstoß usw., die links von den Moduln als "Drähte" dargestellt sind. Einspeisungen in die Kanäle sind mit & und * eingetragen, Zielpunkte durch S, D und C, die gleichzeitig den Kanaltyp bezeichnen. Oberhalb des Kastens POET sind dessen zwei Input-Kanäle gedruckt, die beide den Software-Ingenieur als Informations-Quelle haben. Auf den "Klemmleisten" rechts sind die genauen Zielpunkte eingetragen. Unten sind entsprechend 4 Output-Kanäle.

eines Kanales mit einer Klemmleiste ein Buchstabe steht, ist das ein Zielpunkt und außerdem eine Aussage über den Typ des Kanals (Simplex-Übertragung nur in einer Richtung, Duplex(Dialog-)Übertragung in beiden Richtungen; es gibt, entsprechend den PEARL-Möglichkeiten, auch Aktivierungs-Kanäle usw.. Semaphore werden als Kanäle dargestellt, bei denen RELEASE und REQUEST Quell- bzw. Zielpunkte sind.) Ein Kanal kann mehrere Quell- und Zielpunkte haben.

Oberhalb des Kastens POET stehen die Kanäle, die Informationen nach POET übertragen, und zwar ganz links die Namen der Teilsysteme, aus denen die Information stammt, und in der jeweils nächsten Zeile der Name des Informationskanals. Jeder Name steht in einer Extra-Zeile, damit beim Druck der Gesamt-Dokumentation Hinweise an den rechten Zeilenrand gedruckt werden können, auf welcher Seite die zugehörige Graphik zu finden ist. Rechts neben den Teilsystemen sind wieder Klemmleisten, mit deren Hilfe angegeben wird, in welches Teilsystem die Information fließt. Analog zu den Input-Kanälen stehen die Output-Kanäle unten in der Graphik.

In den einzelnen Kästen stehen rechts oben Nummern, die vom System automatisch erzeugt werden; sie dienen bei der interaktiven Entwicklung eines Entwurfes zur Anwahl des zugehörigen Bildes. Die Länge der Nummern entspricht der Verfeinerungstiefe; Nummer 129 ist das 9. Teilsystem des Teilsystems 12. Die Nummern von Informationskanälen enthalten einen Bindestrich; Nummer 12-A ist der 10. Informationskanal, der Teile des Teilsystems 12 verbindet.

3. Gliederung des POET-Systems

Wie man sieht, besteht POET aus einer Dialog-Schnittstelle, aus einem Datenbank-Editor, einer Datenbank, einem Überprüfer, einem Bilderzeuger und aus einem Dokumenterzeuger. Editor, Überprüfer und Dokumenterzeuger werden von der Dialog-Schnittstelle alternativ angestoßen; der Bilderzeuger wird entweder vom Editor oder vom Dokumenterzeuger angestoßen. Die Funktion des Bilderzeugers ergibt sich aus dem Namen; der Dokumenterzeuger gibt alle Bilder und die zugehörigen Beschreibungen über Schnelldrucker aus, nachdem er sie mit Querverweisen versehen hat. Zu jedem Dokument gehören außerdem eine Legende und Referenzlisten. Die übrigen Teile werden weiter unten kurz beschrieben.

Bild 2: Struktur des Gesamtsystems für Grobentwurf und Dokumentation von PEARL-Programmen mit den Informationskanälen, welche die drei Teile verbinden. Die Verfeinerung von POET wurde bereits in Bild 1 gezeigt.

B i l d 2 zeigt die Logik des Gesamtsystems: der Software-Ingenieur, der mit POET einen Grobentwurf entwickeln möchte, gibt Befehle und Parameter in das Programm ein, die zu Änderungen und Ergänzungen in der Datenbank führen. Der Bilderzeuger zeigt ihm die Resultate auf dem Terminal-Bildschirm. Nach (Teil-)Fertigstellung eines Entwurfes wird das Entwurfs-Dokument gedruckt und einer Revision unterzogen, was wieder zur weiteren Befehlen usw. führen kann, um den Entwurf weiter zu verbessern. Das Entwurfs-Dokument besteht außer aus den Bildern und zugehörigen Beschreibungen aus Referenzlisten. Jede Beschreibung wird normalerweise zweimal gedruckt, weil jedes Teilsystem (mit Ausnahme der Wurzel und der Blätter des Hierarchie-Baumes) in zwei Bildern auftaucht. Durch diese Redundanz wird das Entwurfs-Dokument zwar relativ umfangreich,

aber leichter les- und prüfbar.

Für jeden Entwurf wird von POET eine Datenbank angelegt. Sie besteht aus 5 Teilen: der Teilsystemdatei, der Informationskanal-Datei, der Quellpunkt-Datei, der Zielpunkt-Datei und einer Datei, welche die verbalen Beschreibungen der Teilsysteme und Kanäle enthält. Die Dateien sind so verzeigert, daß die zu einem Bild gehörende Information möglichst schnell gefunden werden kann.

4. Die Dialog-Schnittstelle

Die Dialog-Schnittstelle wurde so gestaltet, daß die richtige Bedienung praktisch kein Nachschlagen in Handbüchern erfordert und sich aus dem Umgang mit dem System selbst ergibt. Die Bediendialoge sind hierarchisch aufgebaut: der Rechner stellt Fragen, die vom Software-Ingenieur mit Ja oder Ne(in) beantwortet werden können. Angeboten wird eine Standard-Antwort, die mit "Return" übernommen werden kann. Auf jeder Bediendialog-Stufe werden die Antworten zunächst mit Nein angeboten; falls keine mit Ja beantwortet wird, wiederholt das System die Fragen noch einmal und springt dann in die nächsthöhere Hierarchie-Ebene zurück, wenn wieder keine Frage bejaht wird. Das System merkt sich Antworten und bietet sie beim nächsten Mal als Standard-Antwort an; da aufeinander folgende Bedienvorgänge häufig sehr ähnlich sind, besteht die Bedienung überwiegend aus dem Drücken der Return-Taste. Auch Parameter, wie z.B. die Kanal- und System-Typangaben, werden auf diese Weise angefordert und eingegeben.

5. Der Datenbank-Editor

Der Datenbank-Editor dient dazu, Entwurfsteile hinzuzufügen, zu streichen, zu verändern oder umzuordnen. Die ersten drei Tätigkeiten sind selbstverständlich beim Erstellen eines Entwurfes; Umordnen ist notwendig, um die hierarchische Gliederung des Entwurfs zu ändern, indem Zwischenstufen eingeführt oder gestrichen werden, oder um Teilsysteme in besserer Weise zu "Baugruppen" zusammenzufassen. Die Auswahl der Editor-Funktionen geschieht auf die oben beschriebene Weise in einem Frage-Antwort-Spiel; angeforderte Parameter werden selbstverständlich so weit wie möglich auf formale Richtigkeit geprüft, bevor sie in die Datenbank übernommen werden.

6. Der Überprüfer

Der Überprüfer prüft den Inhalt der Datenbank auf Vollständigkeit und formale Richtigkeit und komprimiert die Datenbank nach größeren Streichungen. Derartige Prüfungen sind beispielsweise notwendig, weil beim Umordnen von Systemteilen Informationskanäle "abreißen" können und dann neu mit Teilsystemen verbunden werden müssen. Der Überprüfer meldet solche Defekte und erspart dadurch visuelle Prüfungen durch den Software-Techniker.

7. Teilkanäle

Es muß noch kurz auf die Rolle der Teilkanäle eingegangen werden. Aus dem Layout der Teilsystembilder ergibt sich, daß Input-, Output- und interne Verbindungskanäle eines Teilsystems in ihrer Anzahl auf je 15 beschränkt sind. Es ist daher notwendig und auch sinnvoll, Informationskanäle zu "Kabelbäumen" oder auch "Bussen" zusammenzufassen und den Aufbau dieser Konstruktions-Elemente in Bildern genauer darzustellen. B i l d 3 zeigt ein derartiges Kanalbild. Wir erkennen dadurch, daß die Steuerbefehle für das POET-System in verschiedenen Programmteilen enden, und daß die Dation Terminal (an der D-Umrandung als Dation erkennbar), sowie einige Prozeduren als Schnittstellen dienen, in denen die im Kanal übertragenen Informationen in andere Formen umgewandelt werden. Die in einem Kanalbild aufgeführten Teilsysteme sind übrigens Blätter des System-Hierarchiebaumes, die normalerweise in verschiedenen Hierarchie-Ebenen liegen; in den Teilsystembildern ist hingegen der Übersichtlichkeit halber dargestellt, wie die Informationskanäle die Teilsysteme auf gleicher HierarchieEbene miteinander verbinden. Dadurch enthalten die Kanalbilder in dieser Hinsicht genauere Informationen.

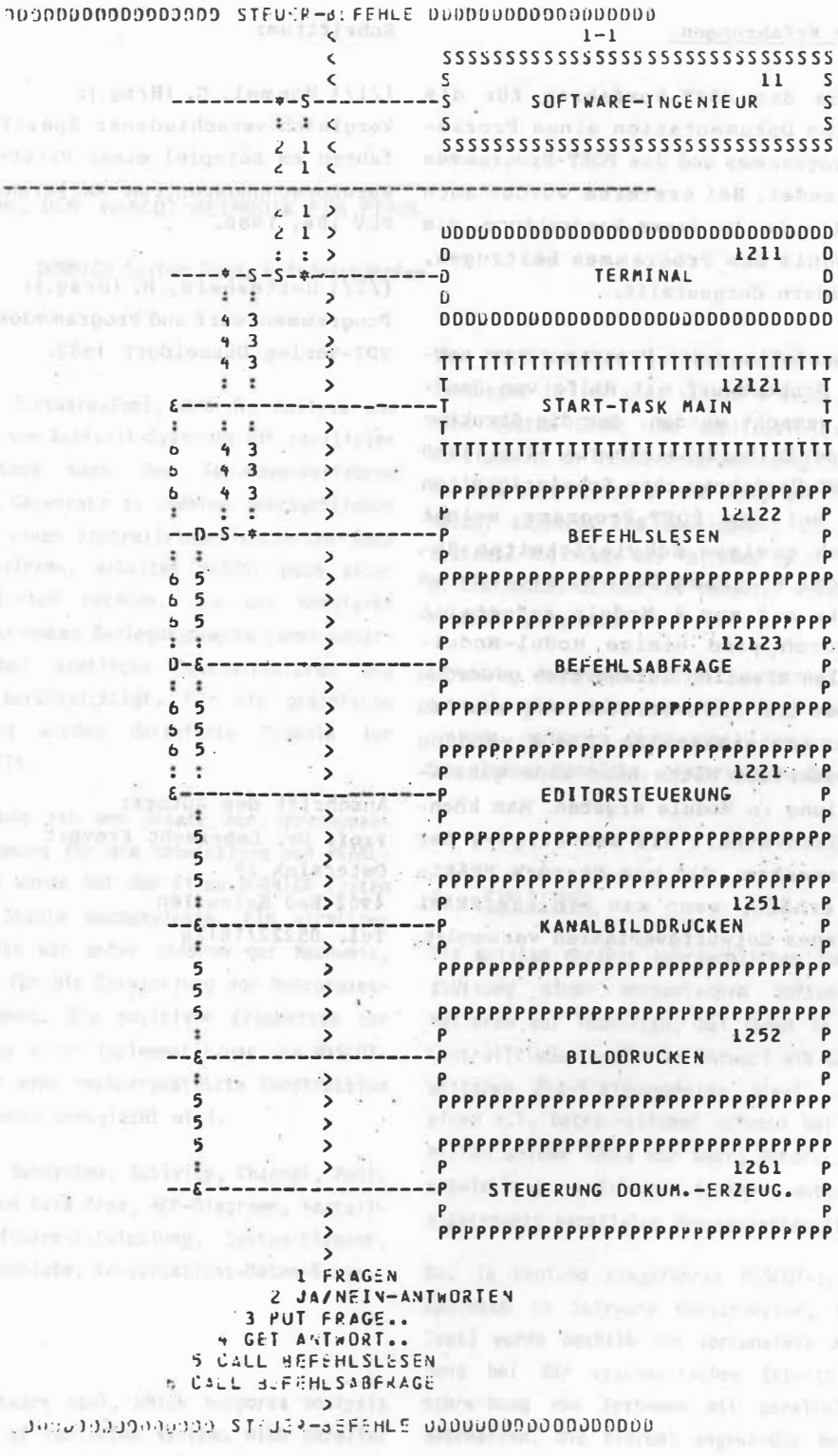

Bild 3: Verfeinerung des Informationskanals 1-1 "Steuerbefehle" aus Bild 1 und 2. Man erkennt, daß der Informationsaustausch zwischen dem Software-Ingenieur und den Teilen des POET-Systems mit Hilfe von Prozedur-Aufrufen und der Dation Terminal erfolgt.

8. Bisherige Erfahrungen.

Bisher wurde das POET-Verfahren für die nachträgliche Dokumentation eines Prozeß-Steuerungsprogrammes und des POET-Programmes selbst verwendet. Bei ersterem wurden auch einige Details der Hardware-Verdrahtung, die zum Verständnis des Programmes beitrugen, mit POET-Bildern dargestellt.

Für das Prozeßsteuerungs-Programm war seinerzeit ein Grobentwurf mit Hilfe von Handzeichnungen gemacht worden, der die Struktur von PEARL-Moduln berücksichtigte. Hier ließ sich das POET-Verfahren ohne Schwierigkeiten einsetzen. Bei dem POET-Programm selbst ergaben sich gewisse Schwierigkeiten dadurch, daß das Programm über 90 Prozeduren enthält, die auf nur 6 Module aufgeteilt waren. Dadurch sind einige Modul-Modul-Schnittstellen ziemlich umfangreich geworden und bestehen in POET-Darstellung aus zu vielen Informationskanälen. Die Anwendung des POET-Verfahrens hätte hier eine günstigere Aufteilung in Module ergeben. Man könnte diesen Sachverhalt als Bestätigung der Vermutung ansehen, daß man bessere PEARL-Programme erhält, wenn man ein auf PEARL zugeschnittenes Entwurfsverfahren verwendet.

Schrifttum:

(/1/) Hommel, G. (Hrsg.):
Vergleich verschiedener Spezifikationsverfahren am Beispiel einer Paketverteilanlage;
Kernforschungszentrum Karlsruhe GmbH., KfK-PDV 186, 1980.

(/2/) Nettesheim, H. (Hrsg.):
Programmentwurf und Programmdokumentation;
VDI-Verlag Düsseldorf 1982.

Anschrift des Autors:
Prof. Dr. Leberecht Frevert
Ostersiek 29
4902 Bad Salzuflen
Tel. 05222/10126