ERFAHRUNGEN MIT BASIC PEARL IM PROJEKT NETZLEITSTELLE DEGGENDORF

Ing.grad. D. Struhalla
Siemens AG - E 14 - 8520 Erlangen

Zusammenfassung:

Der vorliegende Beitrag schildert die Erfahrungen, die mit Basic-PEARL bei der Projektabwicklung für die Netzleitstelle Deggendorf der Energieversorgung Ostbayern AG gemacht wurden. Um die Erfahrungen im rechten Licht erscheinen zu lassen, wird zunächst das Projekt anhand eines historischen Rückblicks, der Anlagenkonfiguration und einiger Projektdaten vorgestellt.

Darüberhinaus enthält der Beitrag einige Anregungen für die Abwicklung von 'PEARL-Projekten'.

Die PEARL-Anwendung hat gezeigt, daß die Sprache eine schnelle, fehlerarme und selbstdokumentierende Programmierung ermöglicht.

PEARL-Programme haben, wie alle in höheren Programmiersprachen geschriebenen Programme, einen gegenüber Assembler größeren Speicherplatzbedarf, zeichnen sich jedoch durch eine hohe Zuverlässigkeit aus.

Die diesem Beitrag zugrunde liegenden Arbeiten wurden mit Mitteln des Bundesministeriums für Forschung und Technologie (Förderungszeichen: P 4.2/36, RE-OBA/1) gefördert.
Die Verantwortung für den Inhalt liegt jedoch allein beim Autor.

1. Projektgeschichte

Siemens erhielt am 6.12.76 den Auftrag zur Ausrüstung einer Netzleitstelle für die Energieversorgung Ostbayern AG (OBAG) in Deggendorf.

Bestandteil des Auftrages war die Forderung nach Implementierung der Anwendersoftware in der Programmiersprache PEARL.

Im folgenden werden einige markante Termine genannt, die unsere Arbeiten bestimmten:

12.76 - 03.78	Analyse, Spezifikation
09.77	Beschreibung Basic PEARL (PDV 120/121)
10.77	Beschreibung Full PEARL (PDV 130)
01.78	Freigabe des ersten Basic PEARL-Compilers von Siemens
02.78	Veröffentlichung der OBAG-Funktionsanalyse (PDV-E 108)
04.78 - 09.79	Implementierung, Test
05.78	Beginn der PEARL-Schulung durch die Schule für Prozeßrechnertechnik
06.78	Festschreibung von Basic PEARL (DIN 66253 Teil 1)
12.78	Liefereinsatz des Basic PEARL-Compilers von Siemens (DIN 66253)
10.79 - 11.80	Inbetriebnahme

2. Anlagenkonfiguration

Ein Rechnersystem bestehend aus

- 1 Zentraleinheit R30 mit 384 k Byte Hauptspeicher
- 2 Sichtgerätearbeitsplätzen in der Warte mit je 4 grafischen Sichtgeräten, Format 80 x 48, für Netz- und Übersichtsbilder und je 1 alphanumerischen Sichtgerät, Format 80 x 24, für Betriebslisten
- 1 Dialogarbeitsplatz im Rechnerraum mit 1 grafischen Sichtgerät, Format 80 x 48, zur Bildkonstruktion und 1 alphanumerischen Sichtgerät, Format 80 x 48, zur Datenpflege.

Aus Sicherheitsgründen ist ein zweites Rechnersystem installiert worden. Das sich daraus ergebende Doppelrechnersystem wird im 'stand-by' Betrieb gefahren.

Jeder Rechner hat seine eigenen Massenspeicher und ein eigenes Prozeßelement. Umschaltbar sind lediglich die Arbeitsplätze mit ihren Sichtgeräten.

Mit dem Doppelrechnersystem ist ein sog. Testbetrieb möglich. Hierbei bekommt jeder Rechner einen Arbeitsplatz zugeschaltet. Während der prozeßführende Rechner mit nur einem Pult den Prozeß weiterführt, können auf dem anderen Rechner mit dem zweiten Pult z. B. neue Daten ausgetestet werden.

3. Projektdaten

3.1 Hardware

Das Netzleitsystem ist für 100 Unterstationen ausgelegt. Der derzeitige Auftragsumfang beläuft sich auf 78 Unterstationen.

Die Informationen aus den Unterstationen werden parallel beiden Rechnern zugeführt.

Das statische Informationsvolumen, bezogen auf den Endausbau, beträgt

20.000 Meldungen
1.500 Meßwerte
1.500 Befehle

Unter den technologischen Begriffen Meldung, Meßwert und Befehl werden hier binäre Signale zwischen 1 Bit und 8 Bit Länge verstanden.

Das dynamische Informationsvolumen beträgt bei 100 Bd. Übertragungsgeschwindigkeit und 48 Zentralgeräten

$$\text{permanent } 60 \quad \frac{\text{Telegramme}}{\text{s}}$$

oder anders ausgedrückt, die Telegrammfolgefrequenz beträgt

16 ms

Die Telegrammbreite ist 48 Bit und enthält entweder 4 Meßwerte oder 32 Meldungen.

3.2 Software

Auf dem standardmäßigen Betriebssystem ist ein Prozeß- und ein online laufendes Dialogsystem aufgesetzt.

Das Prozeßsystem enthält die Aufgaben Steuern, Melden, Überwachen, Protokollieren und Bedienen.

Das Dialogsystem beinhaltet die Aufgaben Datenpflege und Bildkonstruktion.

Beide Dialoge, sowohl Datenpflege als auch Bildkonstruktion, sind losgelöst von internen Softwarestrukturen. Dies ist die Voraussetzung dafür, daß der Anlagenbetreiber selbst die Rechner mit seinen Netzdaten 'füttern' kann.

Der Aufwand für das Prozeß- und Dialogsystem beläuft sich auf 170 Mannmonate oder in Programmen gesprochen auf 65 PEARL-Objekte mit 1000 K Byte Länge und 4 Assembler-Objekten mit 44 K Byte Länge.

Der Assembler wurde für die besonders zeitkritische und platzoptimale Fernwirktelegrammbearbeitung eingesetzt.

4. Ergebnisse

4.1 Sprache

Erlernbarkeit

Die Sprache ist leicht erlernbar.

Zum wirklichen Arbeiten mit dem Werkzeug PEARL benötigt man allerdings auch das Verständnis für das zugrundeliegende Sprachkonzept. Als Beispiel sei hier die Typverträglichkeit (Konvertierungsroutinen) genannt.

Sprachumfang

Der Sprachumfang von Basic PEARL hat sich bis auf einige Ausnahmen als ausreichend erwiesen.

Das Siemens PEARL, im folgenden PEARL 300 genannt, ist um einige Sprachelemente im Rahmen von Full PEARL wie z.B. Characterselektion, Gruppenselektion von Charactern und Bits, Initialisierung von Feldern und Prozeduren für Datum und Uhrzeit erweitert.

Die Festlegung des Gültigkeitsbereiches eines Selektornamens in Basic PEARL, der sich nicht auf die zugehörige Struktur

sondern auf den übergeordneten Block bezieht, sehen wir als Nachteil an. Dadurch geht ein wesentlicher Teil der in der Strukturdefinition liegenden Transparenz verloren.

Die Formulierung von Datennahtstellen zwischen einzelnen Programmen - Datei und Common Data - ist zu elementar. Der versierte Anwender vermißt die direkte Datenübergabe zwischen Programmen.

Die Koordinierung des Softwaresystem, Netzleitstelle Deggendorf, geschieht fast ausschließlich über Semaphorvariablen.

Damit PEARL-Systeme noch effektiver arbeiten, müßte die Semaphorvariable dem ACTIVATE-Statement gleichgestellt werden - will sagen, die Semaphorvariable muß um die Zeitmodifikation erweitert werden.

Notation

Die Sprachnotation von PEARL ist übersichtlich, klar, leicht verständlich und selbstdokumentierend.

Assembleranschluß

PEARL 300 arbeitet mit einem variablen blocktiefenabhängigen Keller. Über diesen Keller ist PEARL 300 auf Prozedurebene mit beliebigen Sprachen mischbar.

Zuverlässigkeit

Hier liegt die Stärke der Sprache.

Implementierung und Test haben gezeigt, daß die Fehlerquote bei PEARL-Programmen sehr niedrig ist. Hier erweist sich die bereits zur Compilierzeit erfolgende Datentyp-Prüfung (Konvertierungsroutine) als sehr nützlich.

Die Sprachelemente von PEARL liegen fast ausnahmslos auf höherer Ebene. Die Sprache eignet sich sehr gut als Bindeglied zwischen Systemmann und Technologie. Sie unterstützt guten Programmstil, Modularität und strukturierte Programmierung.

Programmiersprachen sind weder die Ursache noch die Lösung von Softwareproblemen. Wegen ihrer zentralen Rolle bei allen Software-Entwicklungen wirken sie allerdings richtungsweisend.

PEARL bewirkt bei der Lösung von Echtzeitproblemen eine Kosteneinsparung durch

- reduzierte Codier-, Test- und Fehlerbehebungskosten
- geringere Projektführungs- und Wartungskosten aufgrund erhöhter Lesbarkeit und Durchschaubarkeit der Programme

Die Kosteneinsparung beläuft sich gegenüber einer Assemblerprogrammierung zwischen 20 und 40%.

4.2 Compilieren/Binden

PEARL 300 - Module müssen nach der Compilierung mit dem Siemens PEARL-Compiler PC3Ø mit Hilfe des Binders zu einem ladbaren und danach ablauffähigem Programm gebunden werden.

Die Zeit zum Compilieren und Binden liegt im Schnitt bei ca. 5 Min. In Verbindung mit einem leistungsfähigen Drucker ergeben sich somit gute Compilier- und Bindezeiten.

Der Comiler PC3Ø liefert ein gut dokumentiertes Fehlerprotokoll, das eine schnelle und sichere Fehleridentifikation ermöglicht.

4.3 Programmcode

Die mittlere Programmlänge im Projekt Netzleitstelle Deggendorf liegt bei 18 K Byte, wobei das Spektrum von 4 K Byte bis 44 K Byte reicht. Eine subjektive Beurteilung der Länge von PEARL-Programmen und funktionsähnlichen Assemblerprogrammen ergibt einen Faktor um 2. Dieser Faktor gilt allerdings nur in Verbindung mit einem im Zentralspeicher reentrant bereitgestellten Laufzeitsystem.

4.4 Zeitverhalten

Die Sprache PEARL bietet viele Möglichkeiten der Strukturierung und Dimensionierung von Daten. Beides geschieht auf Sprachebene und ist somit rechnerunabhängig.

Im Laufe der Arbeiten haben sich allerdings folgende Erkenntnisse herauskristallisiert.

- Der Aufbau eines umfangreichen Software-Systems ist auch in PEARL ohne Hardware- und Betriebssystem-Kenntnisse nicht möglich.
- Wenn harte Zeitbedingungen eingehalten werden sollen - und wann ist das nicht der Fall - dann müssen rechnerinterne Strukturen mit berücksichtigt werden.
- Durch geschickte Anwendung der Sprache kann unter Berücksichtigung der vom Compiler abgesetzten Befehle ein Programm ganz entscheidend zeitoptimiert werden.

  Beispiel: Eine falsch placierte Wertzuweisung von Strukturselektoren in einer Programmschleife bedeutet bei Meßwertverarbeitungsprogrammen, daß diese Wertzuweisung u.U. 1500-mal unnütz durchlaufen wird.

4.5 Test

In PEARL 300 ist ein 'online Test' zur Laufzeit möglich. Es sind die Funktionen Zeilentrace, Haltepunkte und Haltepunkte mit Bedingung realisiert.

Das Anschauen und Verändern von Variablen ist in Verbindung mit der standardmäßigen Systemtesthilfe möglich.

5. Anregungen

5.1 Konzept

Die in der Konzeptphase angedachten Datenstrukturen sollten im Hinblick auf eine transparente Bearbeitung folgenden Randbedingungen genügen

- Trennung zwischen Buchführung und Daten (keine Adressverweise in Datensätzen)
- alle Datenstrukturen so vereinbaren, daß eine wiederholbare Datensatzstruktur entsteht.

5.2 Spezifikation

Bereits in der Spezifikationsphase müssen einige übergeordnete Maßnahmen getroffen werden

- Zentrale Erfassung der globalen Daten für das gesamte System (Vereinbarung von Namen und Datentypen)
- Zentrale Erfassung der im System vorkommenden Dateien und Geräte. (Festlegung von Zugriffsrichtung, Länge, Lage und Datentyp gem. der Dateideklaration)
- Anwendung der strukturierten Programmierung (Nassi Shneiderman)
- Bei Programmnahtstellen gleiche Namen und Datentypen für Daten mit gleicher Bedeutung wählen.
- Für die zentralen Nahtstellen des Systems einen online lauffähigen Nahtstellentrace vorsehen
- Die funktionelle Gliederung der Tasks so wählen, daß ggf. einzelne Funktionen mit möglichst wenig Aufwand in eigenständigen Tasks ausgelagert werden können

5.3 Implementierung

Für die Implementierung gelten folgende Empfehlungen

- Zentrale Erstellung des SYSTEM-Teils
- Zentrale Erstellung eines Definitionssatzes der globalen Daten
- Zentrale Erstellung eines Spezifikationssatzes für die globalen Daten und Prozeduren
- Trotz Formatfreiheit den Protokollaufbau so wählen, daß die Blockstrukur erkennbar ist
- Bei Deklaration pro Variable eine neue Zeile beginnen und einzeln kommentieren.

5.4 Test

Um jederzeit einen bequemen online Test durchführen zu können, sollten taskspezifische Tracefunktionen mit einprogrammiert sein.

Bei Einschalten dieser Fracefunktionen sollte es möglich sein, Programmabläufe nicht nur im Dialog mit dem Rechner verfolgen zu können. Anhand dieses Traceprotokolls muß eine 'post-mortem-Analyse' möglich sein.
(Nur zu empfehlen, wenn die Formatschlüssel reentrant im Zentralspeicher stehen).

5.5 Dokumentation

Bei Einhaltung gewisser Notationsregeln stellt das Quellsprachenlisting die Basisdokumentation dar. Sie muß lediglich um eine übergeordnete Funktions- und Datenbeschreibung ergänzt werden.

6. Resümee

Die PEARL-Anwendung hat gezeigt, daß die Sprache eine schnelle fehlerarme und slebstdokumentierende Programmierung ermöglicht.

PEARL-Programme zeichnen sich durch eine hohe Zuverlässigkeit aus - allerdings zu Lasten eines erhöhten Speicherplatzbedarfs.