Anwendung von unterschiedlichen PEARL-Umgebungen in einem Projekt

Dr. M. Ammann, Tettnang

Zusammenfassung

Dornier entwickelte und implementierte für das rechnergestützte Waffeneinsatzsystem ARES die Anwender- und Testsoftware in PEARL nach dem Phasenmodell mit Unterstützung durch EPOS.

Zum Einsatz kamen zwei verschiedene PEARL-Entwicklungsumgebungen, nämlich für ATM 80-PEARL und das DEA (GPP)-PEARL. Das Arbeiten mit diesen verschiedenen Umgebungen wird anhand von Aspekten aus ARES diskutiert. Aus den Erfahrungen werden Schlußfolgerungen gezogen.

Stichworte

PEARL, Entwicklungssystem, Sprachumfang, Systemleistung

Summary

Dornier developped and implements application software and test software for the compute-aided weapon operation system ARES with respect to the phase model and with support by EPOS.

Two different PEARL development systems were used, one for ATM 80-PEARL and one for DEA (GPP)-PEARL. Working with these different systems is discussed with respect to aspects from ARES. Out of the experiences there will be done some conclusions.

1. Einleitung

Das Artillerie-Raketen-Einsatz-System ARES hat die Aufgabe, als rechnergestütztes Waffeneinsatzsystem in der Führungsebene einer Batterie die taktische und technische Feuerleitung zu unterstützen. Dornier entwickelt und implementiert für dieses System die Anwender- und Testsoftware in PEARL.
Die Softwareentwicklung erfolgte nach dem Phasenmodell mit Unterstützung durch EPOS.

Der geforderte Funktions- und Leistungsumfang wird durch ein 3-Rechner-System (mit zwei verschiedenen Rechnertypen) realisiert, dessen Komponenten verschiedene Teilaufgaben bearbeiten unter Berücksichtigung von Redundanz und Ausfallverhalten. Für weitere Einzelheiten wird auf [1] verwiesen.

Für Software-Entwicklung und Test wurden zwei unterschiedliche PEARL-Umgebungen eingesetzt, die mit ihren Eigenschaften und Leistungen sowie den daraus zu ziehenden Schlußfolgerungen vorgestellt und diskutiert werden.

2. Verwendete Entwicklungsumgebungen

Für den einen Rechnertyp, einen MR 8020, wird ein ATM 80-Entwicklungssystem eingesetzt; Bild 1 zeigt die verwendete Konfiguration, die sowohl als Gast- wie auch als Zielrechner eingesetzt wird.

Folgende Software wird dabei eingesetzt:

- ATMOS-Betriebssystem
- AMBOS-Produktionssystem
- PEARL-Compiler/-Testsystem
- BAPAS-DB (Fa. Werum)

Für den anderen Rechnertyp, ein DEA 2020 (Basis Intel 8086), wird das PEARL-System der Firma GPP eingesetzt auf zwei verschiedenen Gastrechnern. Das zuerst verwendete System bestand aus einer PDP 11-Konfiguration für den PEARL-Compiler sowie einem Intel Serie III-Entwicklungssystem, angekoppelt über eine V24-Verbindung. Zielsystem ist ein DEA 2020 mit Drucker, Kommunikationsschnittstellen und Display, ohne Externspeicher.

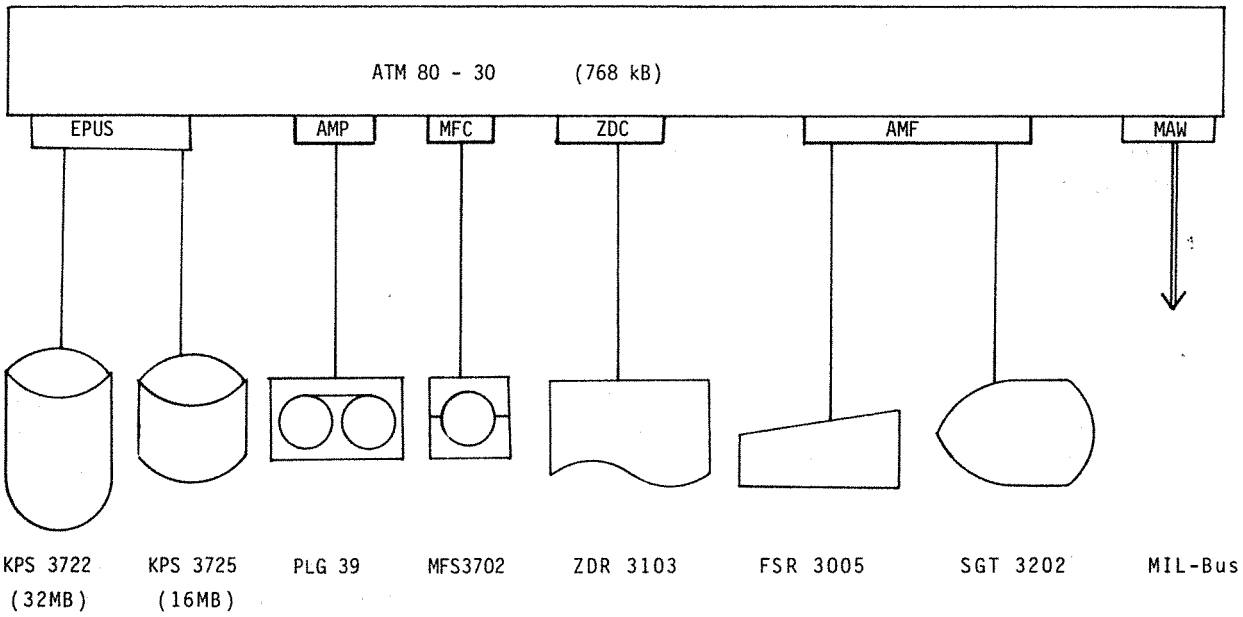

Bild 1 Entwicklungssystem ATM 80-30

Folgende Software wird dabei eingesetzt:

RSX 11-M V 4.1
PEARL-Compiler, -Systemdatengenerator > PDP 11
ISIS II
PEARL-System > Serie III

Zur Steigerung des Durchsatzes werden weiterhin Entwicklungssysteme vom Typ Intel SBC 86/3xx eingesetzt; Bild 2 zeigt die verwendete Konfiguration.

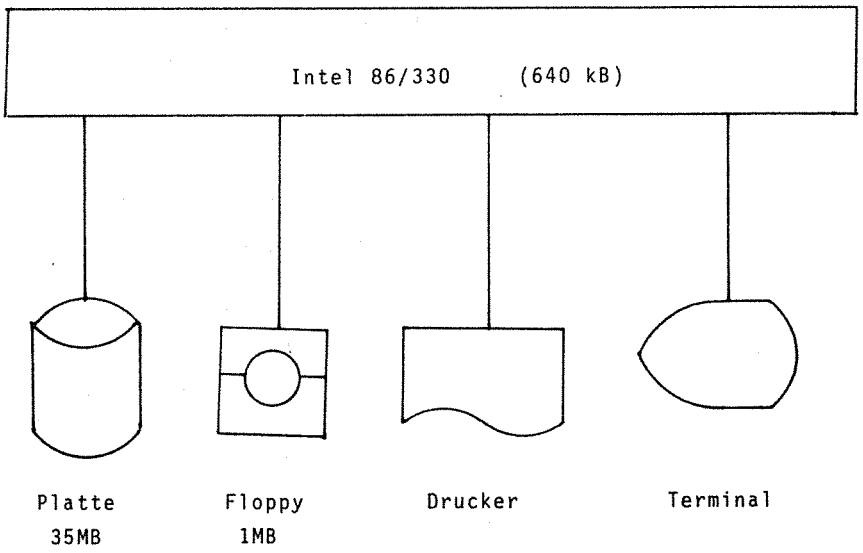

Bild 2 Entwicklungssystem Intel 86/330

Diese Konfiguration kann auch als Zielsystem verwendet werden in Testphasen, die noch keine Peripherie benötigen. Ansonsten ist wieder ein DEA 2020 das Zielsystem. Folgende Software wird hier eingesetzt:

iRMX 86 V6.0
PEARL-Compiler, -Systemdatengenerator
PEARL-System

Für das DEA 2020 ist das Projekt ARES Erstanwender.

Die beiden kurz vorgestellten Entwicklungsumgebungen werden im folgenden nun näher besprochen im Hinblick auf die konkrete Anwendung und das Handling. Dabei werden die Punkte

- Sprachumfang
- Systemleistung
- Test
- Releases
- Stördienst

angesprochen.

3. PEARL-System ATM 80

Anhand der o.a. Stichworte wird nun die PEARL-Entwicklungsumgebung auf ATM 80 aus der Sicht der erfolgten Anwendung diskutiert.

Sprachumfang: Der ATM-PEARL Sprachumfang geht weit über Basic-PEARL hinaus und umfaßt große Teile von Full-PEARL. Für ARES wurden insbesonders folgende Sprachelemente benutzt:

- Strukturen geschachtelt sowie mit Feldern als Element
- TYPE-Definition
- Identitätsspezifikation SPC ... IDENT
- Character-Selektion und -Handling
- Slices
- Referenzen

Damit war es möglich, sauber strukturierte Programme mit klaren Datenstrukturen zu entwickeln und zu implementieren. Als positiver Nebeneffekt war festzustellen, daß die Programme wesentlich effektiver und kleiner wurden als wenn der zu verwendende Sprachumfang auf Basic-PEARL beschränkt würde. Eine effektive Anwendung des ATM-PEARL ist jedoch im Hinblick auf die unterliegende Implementation nur möglich, wenn der Entwickler vertiefte Kenntnisse über Compiler, Laufzeitsystem und interne Struktur von Programmen und Daten hat.

Systemleistung: Für eine zügig abzuwickelnde Test- und Integrationsphase ist die Übersetzungsgeschwindigkeit des PEARL-Systems (bis zum Vorliegen eines ausführbaren Programms) ein wichtiges Leistungsmerkmal. Gemessene Zeiten ergaben Werte von ca. 150-200 PEARL-Statements pro Minute je nach Umfang der dazuzubindenden Bibliotheken und Objekte. Mit dieser Leistung sind vernünftige Turnaround-Zeiten in der Testphase realisierbar.

Ein weiteres Leistungsmerkmal sind die Compilergrenzen bei der Übersetzung größerer Einheiten in Bezug auf Anzahl Symbole, Anzahl Initialisierungswerte, Operandenkeller. Zu Beginn der Entwicklung wurden relativ kleine Werte festgestellt, die für den geplanten Umfang und Komplexitätsgrad der Software nicht ausreichend waren, ohne größere Auswirkungen auf die Implementation. In Zusammenarbeit mit Fa. ATM wurde hier Abhilfe geschaffen, so daß nunmehr in einem Übersetzungsvorgang maximal 800 Zugriffsfunktionen bei 500 Typattributen bearbeitet werden können. Diese Werte sind jedoch noch immer für mehrere geplante Übersetzungseinheiten zu klein und bedingen eine Aufspaltung mit entsprechendem Overhead.

Das letzte hier angesprochene Leistungsmerkmal sind die zulässigen Größen von Adressräumen für Code (d.h. hier Tasks) und Daten.
Ohne die Entwicklung der letzten Jahre darzulegen sei festgestellt, daß hier (derzeit noch) zwei wesentliche Grenzen existieren: die Größe eines Laufbereichs für eine Task (mit oder ohne Daten) und die Größe des Commonbereichs; beide Werte sind nicht unabhängig. Die erreichbaren Laufbereichsgrößen von 32 kB reichen oftmals nicht aus und erzwingen eine Aufteilung von Tasks. Die Problematik des Common-Bereichs wurde inzwischen dadurch entschärft, daß Anwender-Prozeduren und -Daten in das LZO-Segment bzw. in Zonen verlagert werden können, jedoch mit harten Einschränkungen.

Test: Für das PEARL-System existiert ein Testsystem (QETS), das umfangreiche Funktionen für sprachbezogenes Testen anbietet. Dieses System wurde jedoch nur beschränkt eingesetzt, da seine Anwendung die ohnehin knappen Adressräume zu sehr belastet. Der Hauptteil der Tests wurde mit einer eigenen allgemeinen Testumgebung durchgeführt, die ein effektives Testen ermöglicht.
Diese enthält dazu einen Satz von Grundfunktionen zur Ablaufsteuerung von Tests sowie für Dumps und Protokolle. Sie werden testspezifisch ergänzt um Funktionen, wie spezielle Dumps, Dialog für Parameteränderung und Anstoß spezieller Testhilfsroutinen.

Ein großer Vorteil des ATM-PEARL-Systems ist die Tatsache, daß während Entwicklung, Test und SW-Integration der Test auf der gleichen Anlage stattfindet wie die Übersetzung, d.h., Gast- und Zielmaschine sind identisch. Nachteilig ist, daß während des Tests die Anlage zu einem Einplatz-System wird, was in intensiven Testphasen zu Engpässen führt.

Releases: Die ATM-Systemsoftware inkl. PEARL-System wird ständig gepflegt und weiterentwickelt. In Abständen von ca. 1/2 bis 1 Jahr werden neue Versionen herausgegeben. Der positive Teil dieser Tatsache ist ein nach und nach immer leistungsfähigeres System, was der Entwicklung und Implementation von Software sehr zugute kommt. Die unangenehmere Seite ist zum einen der daraus resultierende beträchtliche Aufwand für Systemgenerierungen und Umstellungen, zum anderen die Feststellung, daß neue Releases zu oft mit Fehlern behaftet sind, die die SW-Entwicklung wesentlich beeinträchtigen und deren Behebung oder Umgehung einen beträchtlichen Aufwand verursacht.
Möglicherweise ist ein Grund darin zu sehen, daß vor Auslieferung eines neuen Release kein Feldversuch stattfindet, der in konkreten Anwendungen Fehler beseitigen hilft.

Stördienst: Für einen guten Projektablauf ist ein wirksamer Stördienst für die verwendete Systemsoftware von großer Wichtigkeit. Hier kann von sehr guten Erfahrungen mit dem ATM-Software-Stördienst gesprochen werden, der in der Lage ist, auf schnelle und unbürokratische Art und Weise Störungen zu beheben oder Umgehungen anzubieten.

4. PEARL-System DEA

Im folgenden sollen nun analog zum vorhergehenden Kapitel Aspekte der Arbeit mit dem PEARL-System DEA näher diskutiert werden.

Sprachumfang: Der Sprachumfang des verwendeten GPP-PEARL entspricht weitgehend Basic PEARL. Es sind jedoch folgende Konstrukte nicht vorhanden

- READ/WRITE (in der neuesten Version vorhanden; für ARES ohne Bedeutung)
- kein Character-Selektor
- Einschränkungen bei globalen Daten
- SIGNAL-Reaktionen nur auf Taskebene

Der fehlende Character-Selektor hatte zur Folge, daß für Stringhandling (dies ist der überwiegende Teil der DEA-Aktivitäten) ein Satz von 90 Assemblerprozeduren erstellt werden mußte. Die große Anzahl resultiert aus der strengen Typprüfung in PEARL, die hier eine effektive Lösung verhindert. Zudem fehlt in Basic-PEARL die Identitätsspezifikation, die elegante Problemlösungen durch Umgehen der Typprüfung erlaubt.

Einschränkungen existieren bei Verwendung des INV-Attributs in Spezifikationen globaler Daten; es darf nur angewendet werden, wenn auch die Deklaration mit INV erfolgte. Damit fehlt die Möglichkeit eines modulspezifischen Schreibschutzes für Variable.

Systemleistung: Gemessene Übersetzungszeiten führten zu folgenden Werten:

PDP 11 + Serie III	: 9-12	Statements/Minute
Intel 86/3xx mit DEA	: 12-16	" "
Intel 86/3xx	: 16-20	" "

Diese sehr schlechten Werte bedeuten für die Testphase eine wesentliche Beeinträchtigung; eine vollständige Übersetzung der DEA-Software dauert damit mehrere Tage.

Die Symboltabelle des PEARL-Compilers auf PDP 11 erlaubt maximal 300 Einträge. Dieser sehr kleine Wert erforderte eine übermäßige Aufspaltung aller geplanten Übersetzungseinheiten und verhinderte meist das Arbeiten mit Signalen in der Testphase. Auf Intel 86/3xx sind nach einer Änderung nunmehr 800 Einträge möglich, so daß einigermaßen erträgliche Testbedingungen herrschen.

Bezüglich Adressraum nutzt die Implementation die Möglichkeiten des Intel 8086 hinsichtlich Segmente und Register, so daß hier keine Schwierigkeiten aufgetreten sind.

Test: Für das PEARL-System DEA wurde von der Firma GPP ein sprachbezogenes Testsystem entwickelt, das in ARES zum Einsatz kam. Anders als das ATM-Testsystem, das mit Externspeicher arbeitet, müssen hier sämtliche Listen und Daten zusammen mit dem Testling gebunden werden, was für den Umfang Beschränkungen ergibt. Diese sowie Probleme der Generierung auf Intel 86/3xx ließen deshalb nur einen beschränkten Einsatz zu. Wie bei ATM 80-Rechner wurde wesentlich mit der eigenen Testumgebung gearbeitet.

Das PEARL-System mit den beiden Gastrechnern PDP 11 und Intel Serie III erlaubt zwar paralleles Compilieren auf PDP 11, die weiteren Bearbeitungsgänge sind aber im Single-User-Betrieb vorzunehmen, einschließlich der jeweiligen Transfers. Die Systeme

86/3xx sind reine Einplatzsysteme, bieten aber den Vorteil, auch als Zielmaschine dienen zu können.

Releases: Das PEARL-System wird gepflegt und weiterentwickelt. In größeren Abständen (ca. 1 Jahr) werden neue Versionen herausgegeben. Es sind hier ähnliche Aussagen zu machen wie bei ATM, jedoch waren die negativen Auswirkungen nicht immer so ausgeprägt. Insbesondere ist der Aufwand für die Generierung eines PEARL-Systems wesentlich kleiner als auf Seiten ATM 80.

Stördienst: Für das PEARL-System ist ein Stördienst eingerichtet, der gut funktioniert. Verbesserungen sind jedoch noch möglich in Bezug auf Reaktionszeit.

5. Schlußfolgerungen

In den beiden vorhergehenden Kapiteln wurde die Arbeit mit den beiden im Projekt verwendeten unterschiedlichen PEARL-Systemen anhand von für eine Entwicklung wichtigen Aspekte erläutert. Die hierbei gemachten Erfahrungen sollen nun zusammengefaßt werden; dabei können unter Einbeziehung von Anforderungen aus dem Projekt Schlußfolgerungen gezogen werden bezüglich Eignung, Einsatz und sinnvoller Erweiterungen.

Sprachumfang:

Der Vergleich der Arbeiten mit den beiden verschiedenen Sprachumfängen hat klar gezeigt, daß für Projekte der vorliegenden Größe und Komplexität ein komfortabler Sprachumfang wie der des ATM-PEARL sinnvoll und notwendig ist für eine effektive Programmierung. Insbesonders zu erwähnen ist die Identitätsspezifikation, die ein Umgehen der Typprüfung erlaubt und wesentlich für leistungsfähige Software ist. Die Beschränkung auf Basic-PEARL, das u.a. praktisch keine Textbearbeitung kennt, führt in Verbindung mit der starren Typprüfung zu grotesken Umgehungen und Assemblereinschüben. Generell fehlt in beiden Systemen die Möglichkeit von (auch rechnerübergreifender) Prozeßkommunikation, die in praktisch allen größeren Vorhaben notwendig ist. Dieser Mangel wurde durch eigene Entwicklung geeigneter Software-Moduln behoben.

Ein anderer Mangel in PEARL bezüglich E/A zeigte sich während der Entwicklung, nämglich das Fehlen von Anstoßanweisungen sowie keine Möglichkeit für kombinierte E/A, wie sie praktisch überall zu finden ist. Als Folge davon ergaben sich teilweise nicht sachgerechte Lösungen.

Zusammenfassend kann gesagt werden, daß Basic-PEARL für größere Projekte nicht die geeignete Programmiersprache ist. Die Verwendung eines Sprachumfangs, der große Teile von Full-PEARL umfaßt, ist dringend zu empfehlen. Darüberhinaus sollte in jedem Einzelfall geprüft werden, ob PEARL die geeignete Sprache ist.

Systemleistung

Für eine zügige Abwicklung größerer Projekte ist ein entsprechender Durchsatz an den Entwicklungssystemen zu fordern; ein Wert von 200 Statements pro Minute (über alles) ist anzustreben. Das ATM-System erreicht diese Leistung, während das DEA-System um eine Grössenordnung darunter liegt und damit unerträglich lange Durchlaufzeiten verursacht. Eine Leistungssteigerung um mindestens eine Größenordnung ist hier dringend notwendig.

Beiden Systemen gemeinsam sind Compilergrenzen bezüglich Umfang Symboltabelle und Anzahl Initialwerte, was für die Entwicklung u.U. gravierende Folgen haben kann. Mit ein wesentlicher Grund sind speicherresidente Tabellen und Listen, die nicht beliebig erweiterbar sind. Hier wäre eine Auslagerung auf einen Externspeicher angezeigt und erforderlich.

Bezüglich Adressraum bleibt nur zu erwähnen, daß mit der Einführung der neuen Maschinen ATM 80-16 mit Gate-Array-Technologie und Segmentadressierung eine weitere entscheidende Verbesserung der Situation zu erwarten ist.

Test

Beide PEARL-Systeme enthalten ein sprachbezogenes Testsystem mit einem umfangreichen Funktionssatz. Die Anwendung ist jedoch infolge von Rückwirkungen und Einschränkungen für den Testling mit Schwierigkeiten verbunden. Zu fordern ist hier eine praktisch rückwirkungsfreie Implementation, die den Zielrechner/Testling nicht mit Listen belastet. Die Identität von Gast- und Zielrechner muß gewahrt bleiben.

Nachteilig bei beiden Testsystemen ist die Tatsache, daß sie Einplatzsysteme darstellen, was in intensiven Testphasen zu Engpässen führt, die sich nur durch vermehrten Hardwareeinsatz vermeiden lassen. Mehrplatz-Testumgebungen bringen hier Vorteile, obwohl sie auch nicht ohne Probleme sind (Aufwand, Echtzeitverhalten, ...).

Unabhängig von obigen Überlegungen ersetzt ein Testsystem auf keinen Fall eine anwendungsspezifische

Testumgebung, deren Erstellungsaufwand sich durch systematisches und effektives Testen auszahlt.

Releases, Stördienst

Aufgrund der bisherigen Erfahrungen mit Releases ist zu empfehlen, vor Auslieferung einen Feldtest mit geeigneten Anwendungen vorzunehmen. Dies erscheint das angemessene Vorgehen zu sein, um Aufwand und Verzögerungen durch intensive Fehlersuche und Umgehung bei den Anwendern zu vermeiden.

In diesem Zusammenhang ist die Notwendigkeit einer exakten, funktionierenden Konfigurationskontrolle seitens der Hersteller besonders hervorzuheben. Die Vernachlässigung dieses Punktes führt zu sehr unangenehmen Störungen im Projektablauf und ist deshalb unbedingt zu vermeiden.

Schrifttum:

[1] Ammann, Martin: Entwicklung der Software für das Waffeneinsatzsystem ARES
PEARL-Tagung '83, Düsseldorf

Ammann Dr., Martin
Dornier System GmbH, Abt. ZITW
Postfach 1360
7990 Friedrichshafen
Tel. 07545/85472