

Testen von PEARL-Programmen auf PEARL-Ebene

G. Nieder, Siemens AG Karlsruhe

Zusammenfassung:

In diesem Vortrag wird eine neu entwickelte Testhilfe zum Testen von Programmen auf Quellsprache-Ebene vorgestellt. Die Integration von Compiler und Testsystem in das System 300 wird gezeigt und die Schnittstelle zwischen Compiler und Testsystem erklärt. Verschiedene Anwendungsmöglichkeiten werden aufgezeigt.

Schlüsselwörter:

PEARL, Testhilfe für höhere Sprachen
DEBUG, Compiler-Testsystem-Schnittstelle

1 Testsystem für höhere Sprachen

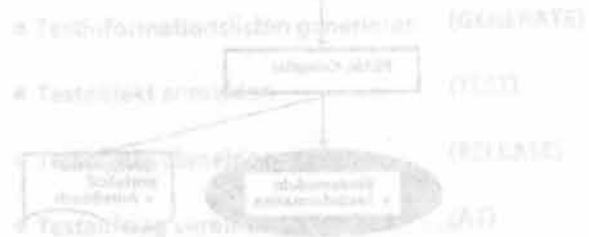
1.1 Bisherige Testmöglichkeiten für PEARL-Programme

Bisher hatte der Siemens-PEARL-Anwender zwei Möglichkeiten, PEARL-Programme zu testen: Ein 'kleines' Testsystem MINITEST und die Assembler-testhilfe TESTS. MINITEST zeichnet sich durch einen geringen Platz- und Zeitbedarf zur Laufzeit aus, bietet aber nur die wichtigsten Funktionen, z. B. Setzen von Haltepunkten oder Ablaufverfolgung. Gegebenenfalls kann zusätzlich die Assembler-testhilfe TESTS zu Hilfe genommen werden. Bei der Compilerbedienung MINITEST wird Testinformation in der Grundsprache abgelegt. Die andere Möglichkeit bestand darin, mit TESTS und der Adreßbuchausgabe des Compilers zu arbeiten (Bild 1) /3/.

1.2 Testsystem DEBUG für höhere Sprachen

Mit zunehmendem Einsatz von PEARL wurde von den Anwendern immer dringender der Einsatz einer Testhilfe gefordert, die die bisherige, aus der Assembler-Denkweise herrührende Testmethodik ablöste und der Leistungsfähigkeit einer höheren Sprache wie PEARL angemessen ist.

Organisatorische Zusammenfassung



Summary:

This lecture deals with a newly developed debugging aid for program testing on source-language level. It describes how the compiler and the debugging system are integrated into the systems 300 and explains the interface between compiler and debugging system. Various applicabilities are demonstrated.

Keywords:

PEARL, debugging aid for high-level-languages,
DEBUG, interface between compiler and debugging system

Folgende Forderungen waren bei der Entwicklung von DEBUG richtunggebend:

- Interaktive Arbeitsweise
- Ansprechen der Variablen über ihre Namen
- Integration von Compiler, Binder, Lader und Testhilfe in ein leistungsfähiges Gesamtsystem
- leichte Erlernbarkeit
- keine Verlängerung des Objektcodes
- Vermeidung von Übersetzungsläufen nur zu Testzwecken
- Fortsetzungsmöglichkeiten bei unkontrolliertem Abbruch
- Überwachung von Programmsystemen
- Integration von Zähleinrichtungen
- Verwendung für verschiedene höhere Sprachen

1.3 Integration in das System 300

Der PEARL-Compiler erzeugt Testinformation bezüglich aller symbolischer Objekte und Zeilennummern gemäß einer einheitlichen Schnittstelle. Diese Schnittstelle wird auch von den übrigen Compilern des Systems 300 bedient, damit ist die Sprachunabhängigkeit

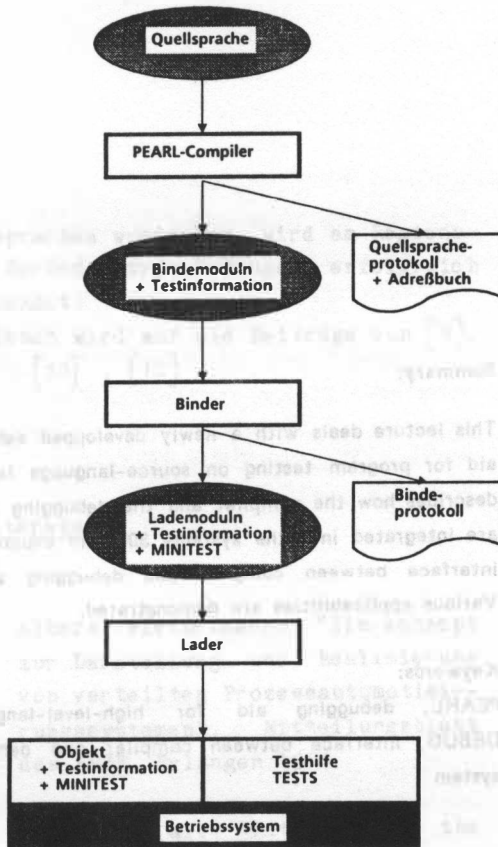


Bild 1. Integration der Testhilfen MINITEST und TESTS in das System 300

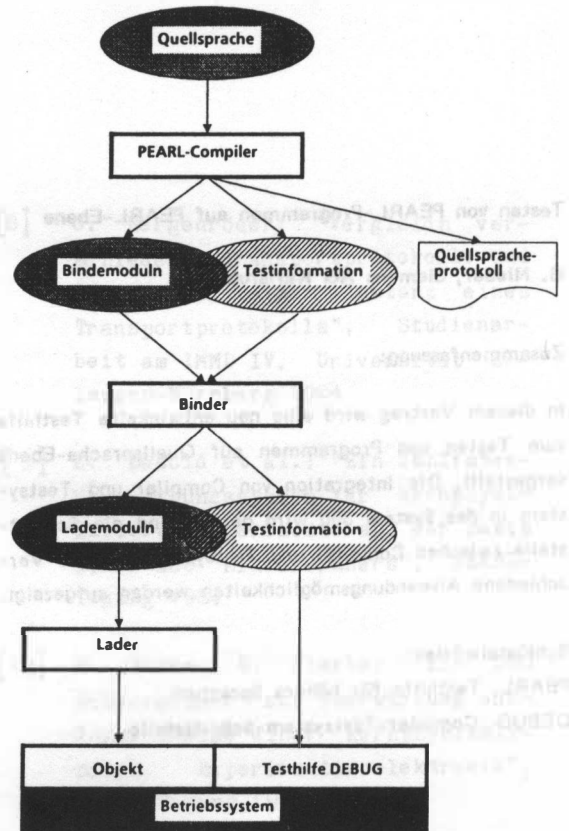


Bild 2. Integration der Testhilfe DEBUG in das System 300 Anwendungsfall PEARL

von DEBUG gewährleistet. Die vom Objektcode unabhängige Testinformation wird mit dem Objektcode vom Binder bearbeitet. Der Objektcode wird nach dem Binden wie bisher vom Lader geladen (Bild 2). Die Testinformation wird von einem speziellen Testlader in Testlisten umgeformt. Aus den Testlisten zieht DEBUG seine Informationen, um beim symbolischen Ansprechen einer Größe aus dem Testobjekt Zugriffe und Überprüfungen durchführen zu können /2/.

2 PEARL-DEBUG-Anschluß

2.1 Schnittstelle zwischen Übersetzer und Testsystem

Der Anschluß von PEARL-Programmen an eine Testhilfe auf Quellspracheebene wird so realisiert, daß der PEARL-Compiler beim Übersetzungsablauf für alle Testobjekte geeignete Zusatzinformation erzeugt. Dabei wird der vom PEARL-Compiler erzeugte Code nicht verändert.

Die vom Compiler erzeugte Testinformation bezüglich der vom Anwender deklarierten Objekte umfaßt die Identifikation, den Typ und den Adressierungsweg dieses Objekts, ebenso semantische Abhängigkeiten wie 'ist Parameter von', 'ist lokale Variable von', den Gültigkeitsbereich usw.

Für Anweisungen werden neben der Zeilennummer laut Übersetzerprotokoll und der Codeanfangsadresse der in dieser Zeile stehenden Anweisung auch Typmerkmale erzeugt, über die DEBUG die Ausführbarkeit von Test-Kommandos steuert. Es lassen sich sowohl mehrzeilige Anweisungen als auch mehrere Anweisungen innerhalb einer Zeile identifizieren /1/.

2.2 Anwendungsgebiete von DEBUG

Dem PEARL-Programmierer wird mit DEBUG ein Testsystem zur Verfügung gestellt, das folgende Anwendungsgebiete überdeckt /1/:

- symbolischer Test von PEARL-Programmen durch Funktionen zur Ablaufverfolgung und zum Ablisten und Ändern von Daten

- Optimierung von Programmen durch Funktionen zum Setzen von Zählpunkten
- Qualitätssicherung durch Überdeckungstest
- Systemtest
- Abnahme, Wartung

2.3 Anwenderschnittstelle von DEBUG

An der Anwenderschnittstelle von DEBUG werden dem Benutzer Daten, Testereignisse und Funktionen zur Verfügung gestellt. Als Sprachmittel dient eine eng an die im System 300 übliche Bediensyntax angelehnte Kommandosprache. Kommandos lassen sich zu Verbundanweisungen und Kommandoprozeduren zusammenfassen.

Ansprechbar sind alle Daten, die durch die Testinformation des Compilers unterstützt werden und die vom Anwender in DEBUG deklarierten Daten.

Testereignisse sind an bestimmte Bedingungen im Ablauf des Testobjekts gekoppelt, z. B. Erreichen einer Programmarke, Erreichen eines bestimmten Werts einer Variablen oder eines besonderen Programmzustands.

Testaufträge sind ereignisabhängige Testaktivitäten, die durch DEBUG verwaltet werden. Eine Auswahl möglicher Kommandos zeigt Tabelle 1 /4/.

Um mit DEBUG testen zu können, braucht der Anwender nicht in den Erstellungsweg der Objektprogramme einzugreifen.

Die Compiler erzeugen standardmäßig die notwendige Testinformation, die vom Binder bearbeitet wird. Die erzeugten Moduln werden vom Lader geladen. Der Anwender erzeugt nun mit Hilfe der GENERATE-Funktion Testlisten, mit dem TEST-Kommando stellt er die Zuordnung zwischen Testlisten und Testobjekt her.

Damit kann er mit dem Test seiner Programme beginnen.

Er braucht nun keine speziellen Adressumrechnungen mehr vorzunehmen, er kann unmittelbar auf Ebene der PEARL-Quellsprache z. B. eine Task an bestimmter Stelle anhalten, dort die Werte aller Arten von Variablen ausgeben oder Inhalte von Variablen gezielt verändern.

Beispiel für eine Kommando-Folge:

Testlisten erzeugen:

GENERATE PLSK1-OBJ.PROG TO PLSK1-TLISTE

Zuordnung zwischen Testlisten und Testobjekt herstellen:

Organisatorische Kommandos

-
- | | |
|---|--------------|
| ● Testinformationslisten generieren | (GENERATE) |
| ● Testobjekt anmelden | (TEST) |
| ● Testobjekt abmelden | (RELEASE) |
| ● Testauftrag vereinbaren | (AT) |
| ● Testauftrag löschen | (DELETE) |
| ● Testauftrag aktivieren | (ACTIVATE) |
| ● Testauftrag passivieren | (DEACTIVATE) |
| ● Temporäre Variable vereinbaren | (DECLARE) |
| ● Testobjektbezug und Objektqualifizierung einstellen | (QUALIFY) |
| ● Informieren über Testobjekt und Voreinstellungen | (INFORM) |
| ● Testhilfe beenden | (END) |
-

Testkommandos DEBUG 300 (Auszug)

-
- | | |
|----------------------------------|----------------|
| ● Ausgabe von Variablen | (DISPLAY) |
| ● Ändern von Variablen | (CHANGE) |
| ● Ablaufverfolgung | (TRACE) |
| ● Ausgabe Objektkennndaten | (DESCRIBE) |
| ● HSP-Puffer ausgeben | (BUFFER) |
| ● Abzug von Bereichen | (DUMP) |
| ● Testobjekt starten, fortsetzen | (RUN) |
| ● Testobjekt wartend setzen | (STOP) |
| ● Testobjekt beenden | (TERMINATE) |
| ● Zählpunkte setzen | (COUNT) |
| ● Überdeckungstest | (TRACE, COUNT) |
-

Tabelle 1. Test- und organisatorische Kommandos an die Testhilfe DEBUG

TEST P'PROG WITH PLSK1-TLISTE

Voreinstellung für einen MODUL angeben:

QUALIFY P'PROG.M'MODUL1

Inhalte von Variablen dieses Moduls ausgeben

DISPLAY .VAR1,.VAR2,.VAR3

Variableninhalt verändern:

CHANGE .VAR2=20

Auf Zeile 100 im Programm anhalten, wenn I den Wert 0 hat:

AT P'PROG.'100' (IF I=0; STOP)

Mit diesem Testsystem wird dem PEARL-Anwender ein Werkzeug in die Hand gegeben, das voll in das System 300 integriert ist und das ihm den Test seiner PEARL-Programme und -Programmsysteme wesentlich erleichtert und vereinfacht.

Literatur:

- /1/ Büchner, G.; Schlüter, M.: DEBUG 300, eine Testhilfe für höhere Programmiersprachen. Siemens Energietechnik 6 (1984) S. 93 - 96
- /2/ Schlüter, M.: Prüf- und Testwerkzeuge für die Software von Automatisierungssystemen. Anforderungen und moderne Lösungen. Fachberichte Messen, Steuern, Regeln Bd. 10 (1983), M. Syrbe und M. Thoma (Hrsg.), S. 625 - 635
- /3/ PEARL 300, Anwenderbeschreibung, Bestell-Nr. P71100-D3010-X-X-35
- /4/ DEBUG 300, Anwenderbeschreibung, Bestell-Nr. P71100-E6071-X-X-35

Nieder, Gertrud

Siemens AG, Karlsruhe

Systemtechnische Entwicklung

E STE 33

Östliche Rheinbrückenstr. 50

7500 Karlsruhe 21

Tel.: 0721/595-4693