RUHR - UNIVERSITÄT BOCHUM

Arbeitsberichte

des

Rechenzentrums

Direktor: Prof. Dr. H. Ehlich

Nr. 7101

AIDA, eine Dialogsprache für den TR440

von

K.-H. Mohn, M. Rosendahl, H. Zoller

Bochum, Februar 1971

Zoller

AIDA, eine Dialogsprache für den TR440

Karl-Heinz Mohn, Manfred Rosendahl, Hanspeter Zoller

Selbst wenn ein Teilnehmer-Rechensystem wie etwa TR440 [1,2] mit Übersetzern für die gängigen Programmiersprachen (ALGOL, BCPL, COBOL, FORTRAN, ...) ausgestattet ist, so vermissen doch immer wieder Benutzer des Systems einen "hürdenfreien" Zugang zum Rechner. Sicherlich sind Kenntnisse sowohl einer formalen Sprache als auch der Kommandosprache [3] geeignet, gerade diesem System beachtliche Leistungen zu entlocken; aber viele Probleme könnten ohne solche Kenntnisse gelöst werden, und noch mehr Benutzer könnten an eine sinnvolle Konsolennutzung herangeführt werden, wenn ein Dialog ohne Vorkenntnisse möglich wäre. Diese Lücke schließt das am Rechenzentrum der Ruhr-Universität in Bochum entwickelte Dialogsystem AIDA [4,5], weil dem Konsolbenutzer ein ganzes Spektrum von "Einstiegsebenen" angeboten wird.

Als einfachste wäre die Tischrechnerebene zu nennen. Ein eingetasteter arithmetischer Ausdruck wird berechnet und das Ergebnis ausgegeben (-Ausgabenanweisungen entfallen-), wobei alle Standardprozeduren von ALGOL 60 [6] verwendet werden dürfen. Eine sehr flexible interne Datenstruktur ermöglicht Operanden beliebiger Struktur wie Vektoren, Matrizen (-auch nicht rechtwinklige-) oder Zeichenketten (Strings). So kann ein Benutzer, ohne AIDA zu kennen, durch einfaches Probieren das System interaktiv kennenlernen. Spezielle Operatoren ermöglichen eine sehr verdichtete Schreibweise beim Eingeben an der Konsole, ähnlich wie in der IVERSON-Notation von APL [7,8]. Jedoch benützt AIDA nur Sonderzeichen des 54-Zeichen-Fernschreibercodes SC1 [9].

Wer zur Formulierung seiner Probleme nicht ohne formale Sprache auskommt, kann ALGOL 60 - Statements [6,10] eingeben (-übrigens auch in abgekürzter Form-), welche interpretativ abgearbeitet werden. Dabei muß nicht auf Rekursivität oder komprimierte Schreibweise (s.o.) verzichtet werden, wohl aber können Deklarationen (wie integer, real, array, ...) weggelassen werden, weil die anpassungsfähige Datenstruktur eine dynamische Typveränderung erlaubt. Die interpretative Bearbeitung birgt natürlich gewisse Nachteile bezüglich interner Rechengeschwindigkeit.

Jedoch sei an dieser Stelle hervorgehoben, daß AIDA keinen Übersetzungs- oder Montagevorgang kennt. Für solche Operatorläufe des Betriebssystems wird selbst dann keinerlei Zeit benötigt, wenn bei einem einmal eingegebenen Programm Änderungen an Typ und Besetzung von Variablen oder mit den eingebauten Korrekturhilfen Änderungen am Quelltext vorgenommen werden, bevor es neuerlich ablaufen soll. So bleibt der Benutzer stets in direktem Kontakt mit seiner Problemstellung.

Darüber hinaus eröffnet das Operatorprozedurkonzept von AIDA neue Möglichkeiten der Unterprogrammtechnik. Selbstverständlich können vorhandene ALGOL-Prozeduren ungeändert in den sogenannten Interpretationsspeicher eingetragen und zur Berechnung aufgerufen werden. Der einmal eingetragene Prozedurname kann fortan als Operator verwendet werden. (Dyadische Operatoren sind pre- oder infix aufrufbar.) Beim Aufruf können auf aktueller Parameterposition neben komplexen Datenstrukturen (s.o.) auch Operatoren auftreten, wie in ALGOL 68 [11] . Zum Austesten von AIDA-Operatorprozeduren sind flexible Eingriffs- und Überwachungsmöglichkeiten implementiert. Interaktiv können so ganze Programmbibliotheken erstellt werden (Rechnen mit komplexen oder triplexen Größen, Statistik, u.s.f.).

Die schon erwähnte interne Datenstruktur ist der bei LISP [12,13] verwendeten sehr ähnlich, so daß es keine große Mühe bedeutete, durch geeignete Systemprozeduren (CAR, CDR, CONS, EQ, ATOM, NULL) in AIDA auch ein interaktives LISP-System zu verwirklichen. Zur Listendefinition darf sowohl dot - als auch list - Notation verwendet werden, womit eine bequeme interaktive Listenverarbeitung ("symbolischer Tischrechner") gegeben ist. Das gesamte System ist in ALGOL 60 geschrieben. Hierdurch wurde Maschinenunabhängigkeit, leichte Änderbarkeit und ein hoher Dokumentationswert erreicht. Mit Systembefehlen (SAVE und LOAD) kann eine Konsolsitzung in AIDA jederzeit unterbrochen, der derzeitige Zustand in der langfristigen Datenhaltung [1,3] gesichert, und die Sitzung an der Unterbrechungsstelle zu einem späteren Zeitpunkt fortgesetzt werden.

Mit dem AIDA-System ist dem "Benutzer ohne Vorkenntnisse" ebenso wie dem Fortgeschrittenen und dem Lehrenden ein Werkzeug in die Hand gegeben, mit dem der Kontakt Mensch - Maschine wesentlich effektiver gestaltet werden kann als bisher.

Schrifttum:

[1] J. Piper, H. Meißner, F. Stetter, M. Heinz
Das Teilnehmer-Betriebssystem BS3. Datenverarbeitung AEG-Telefunken 3, 115-122 (1970)

[2] E. Schmidt, N. Linn, A. Schwald, H. Krainer
Zum Programmiersystem des TR440.
Datenverarbeitung AEG-Telefunken 3, 124-131 (1970)

[3] Unterlagensammlung TR440 Kommandosprache Nr.-N31.D0.01
AEG-Telefunken (1969)

[4] K.-H. Mohn, M. Rosendahl, H. Zoller
AIDA, ein Dialogsystem und seine Implementierung in ALGOL. Vortragsmanuskript zur Fachtagung der GI über Programmiersprachen (1971)

[5] K.-H. Mohn, M. Rosendahl, H. Zoller
AIDA-Manual für den Benutzer (1971)

[6] TR440 - ALGOL 60
Nr. N31.D1.04 AEG-Telefunken (1970)

[7] K.E. Iverson
A Programming Language. Wiley, New York (1962)

[8] A.D. Falkhoff, K.E. Iverson
The APL 360 Terminal System.
ACM Symposium on Experimental Systems for Applied Mathematics. Academic Press, New York, 22-37 (1968)

[9] Unterlagensammlung TAS-Handbuch
Nr. N3/GR61 AEG-Telefunken VS.2 (1968)

[10] P. Naur (Ed.)
Revised Report on the Algorithmic Language ALGOL 60
Numer. Math. 4, 420-453 (1963)

[11] A. van Wijngaarden (Ed.)
Report on the Algorithmic Language ALGOL 68
Numer. Math. 14, 79-218 (1969)

[12] J. Mc Carthy, P.W. Abrahams, D.J. Edwards, T.P. Hart, M.I. Levin
LISP 1.5 Programmer's Manual. MIT-Press, Cambridge (1962)

[13] P.W. Abrahams et al
The LISP2 Programming Language and System.
Proc. AFIPS Fall Joint Comp. Conf., 661-676 (1966)

Anschrift der Verfasser:

RUHR-UNIVERSITÄT BOCHUM
- Rechenzentrum -
4630 BOCHUM
Buscheystraße, Geb. NA