Echte Programmierer meiden Pascal

Vor langer Zeit, in der Goldenen Ära der Computer, war es noch einfach, die Männer von den Softis zu unterscheiden (mitunter auch „Echte Männer“ und „Müsli-Fresser“ genannt): Während dieser Periode waren die Echten Männer diejenigen, die Computer programmieren konnten, und die Müsli-Fresser diejenigen, die es bleiben ließen. Ein echter Computerprogrammierer sagte Dinge wie „`DO10I=1,10`“ oder „`ABEND`“, und der Rest der Welt nörgelte „Computer sind mir zu kompliziert“ oder „Ich kann zu Computern keine gefühlsmäßige Bindung aufbauen — sie sind so unpersönlich“. Dabei zeigte schon Remy Eyssens Buch „Echte Männer mögen kein Müsli“*, daß Echte Männer zu nichts und niemandem eine „gefühlsmäßige Bindung“ aufbauen und daß sie auch keine Angst davor haben, unpersönlich zu sein.

Aber die Zeiten ändern sich: Heute stehen wir einer Welt gegenüber, in der kleine alte Damen vollcomputerisierte Mikrowellenherde kaufen können, in der Echte Männer von zwölf Jahre alten Grünschnäbeln bei „Asteroids“ und „Pac-Man“ in die Ecke gestellt werden und in der jeder seinen eigenen Heimcomputer kaufen und sogar verstehen kann. Der Echte Programmierer ist gefährdet, von Studenten mit TRASH-80-Rechnern† ersetzt zu werden!

Es gibt allerdings einige Unterschiede zwischen dem typischen „Pac-Man“-spielenden Gymnasiasten und einem Echten Programmierer. Die Kenntnis dieser Unterschiede kann den Heranwachsenden ein Ziel geben, nach dem sie streben können — ein Vorbild, eine Vaterfigur. Außerdem schützt sie die Echten Programmierer vor der Arbeitslosigkeit.

Am einfachsten erkennt man Echte Programmierer an der von ihnen verwendeten Programmiersprache: Echte Programmierer benutzen FORTRAN. Müsli-Fresser benutzen Pascal. Niklaus Wirth, der Schöpfer von Pascal, wurde einmal gefragt, wie man seinen Namen ausspreche. „You can either call me by name, pronouncing it ‘Veert’, or call me by value, ‘Worth’“, antwortete er. Diese Bemerkung zeigt sofort, daß Wirth ein Müsli-Fresser ist. Der einzige Parameter-Übergabemechanismus, den Echte Programmierer akzeptieren, ist „call-by-value-return“, wie er in den IBM/370-FORTRAN-G- und -H-Compilern implementiert ist. Echte Programmierer brauchen keine abstrakten Konzepte, um ihre Arbeit zu erledigen: Sie sind vollkommen glücklich mit einem Lochkartenstanzer, einem FORTRAN-IVCompiler und einem Bier. Echte Programmierer schreiben Programme für Listenverarbeitung, Zeichenketten-Manipulationen, Abrechnungswesen (wenn sie sich mit so etwas überhaupt abgeben!) und künstliche Intelligenz in FORTRAN.

Wenn man etwas nicht mit FORTRAN machen kann, muß man es mit Assembler versuchen. Wenn man es mit Assembler auch nicht schafft, ist es sowieso wertlos, sich damit zu beschäftigen.

---

* Heyne Taschenbuch Nr. 6290

† Werden in Deutschland auch unter dem Namen „MURX-II“ vertrieben (Anmerkung des Übersetzers)

Akademische Computerwissenschaftler sind in den letzten Jahren auf's Abstellgleis der Strukturierten Programmierung geraten: Sie behaupten, daß Programme verständlicher werden, wenn bestimmte Sprachkonstrukte und Programmiertechniken benutzt werden. Sie können sich natürlich nicht einigen, welche Konstrukte am besten geeignet sind, und die Beispiele, an denen sie ihren speziellen Standpunkt aufzeigen wollen, passen ausnahmslos auf eine einzige Seite irgendeiner obskuren Zeitschrift. Als ich die Schule verließ, dachte ich, ich sei der beste Programmierer der Welt: Ich konnte ein unschlagbares Tic-Tac-Toe-Programm schreiben, beherrschte fünf verschiedene Programmiersprachen und schrieb fehlerfreie 1 000zeilige Programme. Dann kam ich raus in die wirkliche Welt: Meine erste Aufgabe bestand darin, ein 200 000zeiliges FORTRAN-Programm zu lesen, zu verstehen und um den Faktor 2 zu beschleunigen. Jeder Echte Programmierer wird einem versichern, daß die gesamte Strukturierte Programmierung der Welt bei der Lösung eines derartigen Problems nicht hilft — hier braucht man wirklich Talent. Einige Beobachtungen zum Thema „Echte Programmierer und Strukturierte Programmierung“:

- Echte Programmierer haben keine Angst davor, `GOTO`s zu verwenden.
- Echte Programmierer können fünf Seiten lange `DO`-Schleifen schreiben, ohne durcheinander zu geraten.
- Echte Programmierer lieben arithmetische `IF`-Anweisungen*, weil sie den Code interessanter machen.
- Echte Programmierer schreiben selbstmodifizierende Programme, besonders dann, wenn sie damit in einer inneren Schleife 20 Nanosekunden einsparen können.
- Echte Programmierer brauchen keine Kommentare, das Programm ist selbstdokumentierend.
- Da FORTRAN strukturierte `IF`-, `REPEAT`... `UNTIL`- und `CASE`-Anweisungen nicht kennt, braucht der Echte Programmierer auch kein schlechtes Gewissen zu haben, daß er sie nicht benutzt.
- Nebenbei bemerkt, können sie nötigenfalls mit Hilfe von „Assigned `GOTO`s“† simuliert werden.

Auch Datenstrukturen waren in der letzten Zeit in der Diskussion: Abstrakte Datentypen, Records, Pointer, Listen und Zeichenketten sind in gewissen Kreisen populär geworden. Wirth, der Müsli-Fresser, verfaßte sogar ein ganzes Buch‡, in dem er behauptete, daß man Programme schreiben könne, die auf Datenstrukturen aufbauen, statt es umgekehrt zu machen. Wie jeder Echte Programmierer weiß, gibt es nur eine wirklich nützliche Datenstruktur, nämlich das Array. Zeichenketten, Listen, Records

* Das mit den drei Ausgängen (Anmerkung des Übersetzers)

† Zuweisung einer Sprungmarke an eine Variable (Anmerkung des Übersetzers)

‡ Algorithmen und Datenstrukturen, Stuttgart: B. G. Teubner, 1975

und Mengen sind allesamt Sonderfälle von Arrays und können auch so behandelt werden, ohne dadurch die Sprache zu verkomplizieren. Das Schlimmste an den ganzen schönen Typen ist außerdem, daß man sie deklarieren muß, während Echte Programmiersprachen, wie man weiß, den Typ anhand des ersten Buchstabens eines maximal sechs Zeichen langen Bezeichners implizit festlegen.

Welches Betriebssystem der Echte Programmierer benutzt? CP/M? Gott bewahre! Das ist doch im Grunde ein Spielzeug-Betriebssystem. Selbst kleine alte Damen und Hauptschüler können CP/M benutzen und verstehen.

UNIX ist natürlich schon viel komplizierter — der typische UNIX-Hacker weiß nie, wie das `PRINT`-Kommando diese Woche heißt — aber wenn man es genau nimmt, ist UNIX auch nur ein verherrlichtes Telespiel. Niemand arbeitet auf UNIX-Systemen an ernstzunehmenden Dingen — man schickt kleine Witzchen mit USENET rund um die Welt, oder schreibt ein neues Adventure-Spiel oder Forschungsberichte.

Nein, der Echte Programmierer benutzt OS/370. Ein guter Echter Programmierer kann die Beschreibung des Fehlers „`IJK305I`“ in seinem JCL-Handbuch* finden und verstehen. Ein großartiger Echter Programmierer kann JCL schreiben, ohne je ins Handbuch zu sehen. Ein wahrhaft außerordentlicher Echter Programmierer kann Fehler in einem 6-Megabyte-Hexdump finden, ohne einen Taschenrechner zu benutzen.

OS/370 ist wirklich ein bemerkenswertes Betriebssystem: Mit einem einzigen falsch plazierten Leerzeichen kann man die gesamte Arbeit mehrerer Tage zerstören, was die Wachsamkeit im Programmierteam ungemein fördert. Der beste Weg zum System ist der Kartenlocher. Zwar behaupten einige Leute, es gäbe ein Timesharing-System unter OS/370, aber nach sorgfältigen Nachforschungen bin ich zu dem Schluß gekommen, daß sie sich irren.

Welche Werkzeuge ein Echter Programmierer benutzt? Nun, theoretisch könnte er seine Programme über die Maschinenkonsole eingeben und laufen lassen. In den frühen Tagen der Computerei, als Computer noch Maschinenkonsolen hatten, wurde dies auch gelegentlich getan. Der typische Echte Programmierer wußte den System-Urlader Bit für Bit auswendig und tippte ihn ein, sobald er von seinem Programm zerstört worden war. Damals war Speicher auch noch Speicher — der war nicht einfach leer, wenn der Strom ausfiel. Hauptspeicher von heute hingegen vergessen entweder Dinge, die sie behalten sollten, oder halten Informationen, die schon lange weg sein sollten. Aber zurück zum Thema: Die Legende sagt, daß Seymour Cray, der Erfinder des Cray-1-Supercomputers und der meisten anderen Rechner von Control Data, selbst das erste Betriebssystem für die CDC7600 an der Maschinenkonsole eingetippt hat, als sie das erste Mal eingeschaltet wurde. Cray ist selbstverständlich ein Echter Programmierer.

Einer der Echten Programmierer, die ich am meisten bewundere, arbeitete als Systemprogrammierer für Texas Instruments. Eines Tages erhielt er ein Ferngespräch von einem Benutzer, dessen System mitten in einer wichtigen Arbeit abgestürzt war. Der Bursche reparierte dann den Schaden tatsächlich über's Telefon. Er brachte den Be-

* Job Control Language, Batch-Kommandosprache (Anmerkung des Übersetzers)

nutzer dazu, an der Maschinenkonsole Disk-I/O-Instruktionen einzutippen, Systemtabellen in Hexadezimal zu reparieren und Registerinhalte über Telefon durchzugeben. Die Moral von der Geschichte: Obwohl ein Echter Programmierer normalerweise Kartenlocher und Schnelldrucker benutzt, kommt er im Notfall auch mit Maschinenkonsole und Telefon aus.

In einigen Firmen besteht die Programmeingabe allerdings nicht mehr aus zehn schlangestehenden Ingenieuren, die auf einen 029-Locher warten: In meiner Firma z. B. steht kein einziger Kartenlocher. Der Echte Programmierer muß in diesem Fall seine Arbeit mit einem Texteditor erledigen. Auf den meisten Rechnern stehen verschiedene Editoren zur Verfügung, und der Echte Programmierer muß aufpassen, daß er einen erwischt, der seinen persönlichen Stil wiedergibt. Viele Leute glauben, daß die besten Editoren der Welt am Xerox Palo Alto Research Center geschrieben wurden und auf Alto- oder Dorado-Computern laufen. Unglücklicherweise würde jedoch kein Echter Programmierer einen Computer mit einem Betriebssystem benutzen, das „SmallTalk“* heißt, und sicherlich auch nicht über eine Maus mit einem Rechner kommunizieren.

Einige Konzepte der Xerox-Editoren sind mittlerweile in Editoren eingeflossen, die unter sinnvoller benannten Betriebssystemen arbeiten, so wie EMACS oder VI. Das Problem mit diesen Editoren ist, daß Echte Programmierer das Konzept des „Du kriegst, was Du siehst“ für schlecht halten. Der Echte Programmierer will einen „Du hast es so gewollt, da hast Du's“-Editor, einen, der kompliziert ist, kryptisch, leistungsfähig, gnadenlos und gefährlich. EDOR, um genau zu sein.

So wurde beobachtet, daß EDOR-Kommandofolgen dem Leitungsrauschen ähnlicher sind als lesbarem Text. Eines der unterhaltsameren Spiele, die mit EDOR möglich sind, besteht darin, zwischen jeden Buchstaben des eigenen Namens ein Ausrufezeichen zu setzen, das als Kommando einzugeben und zu raten, was dann passiert. So ungefähr jeder mögliche Tippfehler kann dank EDOR das gerade editierte Programm zerstören, oder schlimmer noch, kann kleine mysteriöse Fehler in einstmals funktionierende Unterprogramme einbringen.

Aus diesem Grund editieren Echte Programmierer nur sehr widerwillig Programme, die schon fast laufen. Sie finden es viel einfacher, den binären Objektcode direkt zu ändern, für gewöhnlich mit einem wundervollen Programm, das SUPERZAP heißt (auf Nicht-IBM-Rechnern entsprechend anders). Dies funktioniert so gut, daß viele laufende Programme auf IBM-Systemen keine Ähnlichkeit mit den ursprünglichen FORTRAN-Quellprogrammen haben. In einigen Fällen ist nicht einmal mehr das Quellprogramm vorhanden. Wenn dann der Zeitpunkt gekommen ist, so ein Programm zu ändern, würde kein Manager auch nur daran denken, einem geringeren als einem Echten Programmierer diese Arbeit zu übertragen — kein müsli-fressender strukturierter Programmierer wüßte auch nur, wo er mit der Arbeit anfangen soll. Man nennt das „Arbeitssicherungsmaßnahme“.

* Geplapper (Anmerkung des Übersetzers)

Hier eine Liste der wichtigsten Programmierhilfen, die der Echte Programmierer *nicht* benutzt:

- FORTRAN-Präprozessoren wie MORTRAN oder RATFOR: Diese „Haute Cuisine“ der Programmierung eignet sich hervorragend, um Müsli zu produzieren.
- Quellcodeorientierte Debugger: Echte Programmierer lesen Hexdumps.
- Compiler, die Code für Array-Indexprüfungen zur Laufzeit erzeugen: Sie ersticken jede Kreativität, zerstören die meisten der interessanten Anwendungen der `EQUIVALENCE`-Vereinbarung, und machen Änderungen des Betriebssystems mit Hilfe negativer Indizes unmöglich. Und schlimmer noch, solcher Code ist ineffizient.
- Programm-Pflege-Systeme: Ein Echter Programmierer hält seine Software als Kartenstapel unter Verschluß, denn dies zeigt, daß der Besitzer seine wichtigen Programme nicht unbewacht lassen kann.

Wo der typische Programmierer arbeitet? Welche Art von Programmen derart talentierter Individuen würdig ist? Nun, man kann sicher sein, daß man nie einen Echten Programmierer beim Schreiben von Buchhaltungsprogrammen in COBOL erwischen wird, oder gar beim Sortieren von Abonnentenadressen des SPIEGELs. Nein, ein Echter Programmierer braucht Aufgaben von weltbewegender Bedeutung.

Echte Programmierer arbeiten für das Los Alamos National Laboratory und schreiben dort Atomkriegs-Simulationen auf CRAY-1-Supercomputern, oder sie arbeiten bei der National Security Agency und entschlüsseln russische Funksprüche.

Nur weil tausende Echter Programmierer für die NASA gearbeitet haben, waren „unsere Jungs“ eher auf dem Mond als die Kosmonauten. Die Computer im Space Shuttle wurden von Echten Programmierern programmiert*, und auch die Betriebssysteme der Cruise Missiles der Firma BOEING wurden von diesen Echten Programmierern entworfen.

Einige der ehrfurchteinflößendsten Echten Programmierer arbeiten im Jet Propulsion Laboratory in Kalifornien. Viele von ihnen kennen das gesamte Betriebssystem der Pioneer- und Voyager-Sonden auswendig. Mit einer Kombination aus großen, bodengebundenen FORTRAN-Programmen und kleinen, von den Sonden mitgeführten Assemblerprogrammen vollbringen sie unglaubliche Kunststücke der Navigation und Improvisation: So treffen sie nur zehn Kilometer große Fenster nahe Saturn nach sechs Jahren Flug durch den Weltraum oder reparieren bzw. umgehen defekte Sensoren, Sender oder Batterien. Angeblich soll es einem Echten Programmierer sogar gelungen sein, in ein paar hundert Bytes unbenutzten Speichers innerhalb der Voyager-Sonde ein Mustererkennungsprogramm zu pressen, das einen neuen Mond des Jupiters suchte, fand und photographierte.

* Außer bei der Challenger!

Für die Galileo-Sonde ist vorgesehen, daß sie auf ihrem Weg zum Jupiter entlang einer schwerkraftgelenkten Bahn am Mars vorbeizieht. Diese Bahn führt in einer Entfernung von 80 ± 3 km an der Marsoberfläche vorbei. Kein Mensch würde diese Art der Navigation einem PASCAL-Programm oder gar -Programmierer anvertrauen.

Viele der Echten Programmierer dieser Welt arbeiten für die amerikanische Regierung, meist für das Verteidigungsministerium. So soll es sein! In letzter Zeit allerdings erscheinen dunkle Wolken am Horizont der Echten Programmierer: Es scheint, als hätten einige einflußreiche Müsli-Fresser im Verteidigungsministerium entschieden, daß in Zukunft alle Verteidigungsprogramme in so einer Art von großer, vereinheitlichter Programmiersprache namens „Ada“ geschrieben werden müßten. Lange Zeit schien es, als läge Ada im Verstoß gegen alle Regeln der Echten Programmierung: Es ist eine Sprache mit Strukturen, Datentypen, strenger Typenbindung und Semikolons. Kurz, sie ist wie geschaffen, um die Kreativität des typischen Echten Programmierers zu verkrüppeln. Glücklicherweise hat jetzt die vom DoD* ausgewählte Sprache noch genügend interessante Eigenschaften, um dem Echten Programmierer eine Annäherung zu ermöglichen: Sie ist unglaublich komplex, sie enthält Möglichkeiten, um mit dem Betriebssystem herumzumachen und Speicherbereiche neu zu verteilen, und Edgar Dijkstra mag sie nicht. Dijkstra ist, wie man wissen sollte, der Autor von „`GOTO`s Considered Harmful“†, einem Meilenstein der Programmiermethodologie, der von PASCAL-Programmierern und Müsli-Fressern gleichermaßen bewundert wird. Aber trotzdem, ein zu allem entschlossener Echter Programmierer kann in jeder Sprache FORTRAN-Programme schreiben.

Der Echte Programmierer kann allerdings auch Kompromisse in bezug auf seine Prinzipien eingehen und an etwas geringeren Aufgaben als der Vernichtung des Lebens arbeiten, sofern er dafür entsprechend bezahlt wird: Viele Echte Programmierer schreiben z. B. Videospiele für Atari, allerdings spielen sie nicht damit. Ein Echter Programmierer weiß, wie er die Maschine jedesmal schlagen kann, und damit ist es keine Herausforderung mehr. Jeder bei Lucas-Film ist ein Echter Programmierer, denn es wäre doch verrückt, das Geld von 50 Millionen „Star-Wars“-Fans auszuschlagen. Der Anteil der Echten Programmierer im Bereich der Computergraphik ist etwas niedriger als anderswo, was wahrscheinlich daran liegt, daß noch niemand irgendeinen Nutzen von Computergraphiken entdeckt hat. Andererseits werden Computergraphik-Programme überwiegend in FORTRAN geschrieben, daher gibt es einige Leute, die so das Schreiben von COBOL-Programmen vermeiden.

Im allgemeinen spielt der Echte Programmierer wie er arbeitet — mit Computern. Er ist ständig darüber erheitert, daß sein Arbeitgeber ihn tatsächlich für etwas bezahlt, was er nur so zum Spaß sowieso tun würde — allerdings achtet er darauf, diese Meinung nicht zu laut zu äußern. Gelegentlich kommt der Echte Programmierer auch aus seinem Büro heraus, um sich ein wenig frische Luft und ein oder zwei Bierchen zu genehmigen. Hier daher einige Hinweise, wie man den Echten Programmierer außerhalb des Computerraums erkennt:

---

* Department of Defense, amerikanisches Verteidigungsministerium (Anmerkung des Übersetzers)

† CACM 1/1983, S. 73

- Auf Parties stehen Echte Programmierer in einer Ecke herum und diskutieren über Sicherheitsmechanismen von Betriebssystemen, und wie man darum herumprogrammiert.
- Beim Fußballspiel vergleicht der Echte Programmierer die Ergebnisse mit seinen auf grünliniertem Leporello-Papier gedruckten Computer-Simulationsergebnissen.
- Am Strand zeichnet der Echte Programmierer Flußdiagramme in den Sand.
- Ein Echter Programmierer geht in die Disco, um sich die Lichtorgel anzusehen.
- Bei Begräbnissen sagt der Echte Programmierer typischerweise: „Armer Hans-Helmut. Er war mit seinem Sortierprogramm schon fast fertig, als ihn der Herzinfarkt geholt hat.“
- Im Supermarkt besteht der Echte Programmierer darauf, seine Bierdosen selber über das Fenster des Strichcodelesers zu schieben, weil er keinem Kassierer zutraut, dies beim ersten Versuch richtig zu machen.

In welcher Umgebung der Echte Programmierer am besten funktioniert? Nun, dies ist eine sehr wichtige Frage für die Manager von Echten Programmierern: Wenn man bedenkt, wie teuer es ist, einen von ihnen in Betrieb zu halten, dann sollte man ihn oder sie in eine optimale Arbeitsumgebung versetzen.

Der typische Echte Programmierer lebt vor einem Computerterminal. Rund um dieses Terminal liegen Ausdrucke von jedem Programm, an dem er je gearbeitet hat, sie stapeln sich grob chronologisch geordnet auf jeder ebenen Fläche des Büros. Im Zimmer verteilt finden sich über ein halbes Dutzend mit kaltem Kaffee mehr oder weniger gefüllte Tassen. Gelegentlich schwimmen Zigarettenkippen darin herum, in einigen Fällen auch Reste von Orangenschalen. Irgendwo liegen Kopien des OS-JCL-Handbuchs und der „Principles of Operation“ an einer besonders interessanten Stelle aufgeschlagen herum, außer bei extrem guten Leuten. An der Wand klebt ein Schnelldruckerkalender mit Snoopy drauf aus dem Jahr 1969. Über den Boden verteilt liegen Reste der Verpackungen von gefüllten Keksen (der Typ, der schon in der Fabrik so trocken ist, daß er auch bei längerem Liegen im Automaten nicht schlechter wird). Schließlich, in der linken oberen Schublade des Schreibtischs, unter der Schachtel mit den Muntermachern, liegt eine Schablone für Flußdiagramme, die sein Vorgänger dort vergessen hat. Echte Programmierer schreiben Programme und keine Dokumentation, das überläßt man den Typen von der Wartung.

Der Echte Programmierer ist in der Lage, 30, 40, ja 50 Stunden in einem Rutsch zu arbeiten, und das unter hohem Zeitdruck. Genaugenommen mag er es so am liebsten. Schlechte Antwortzeiten regen den Echten Programmierer nicht auf — sie geben ihm die Chance, zwischen zwei Kommandos ein bißchen Schlaf zu ergattern. Wenn die Planung nicht genug Zeitdruck bereithält, dann tendiert der Echte Programmierer dazu, seine Arbeit herausfordernder zu machen, indem er sich die ersten neun Wochen mit einem kleinen, aber sehr interessanten Teil des Problems befaßt, um dann in der letzten Woche seine Aufgabe in zwei oder drei 50-Stunden-Marathonsitzungen zu be-

enden. Dies beeindruckt nicht nur den Manager, sondern schafft gleichzeitig eine hervorragende Entschuldigung für das Fehlen der Dokumentation. Und überhaupt: Kein Echter Programmierer arbeitet von neun bis fünf, außer denen von der Nachtschicht. Echte Programmierer tragen keine Schlipse. Echte Programmierer tragen keine hochhackigen Schuhe. Echte Programmierer kommen zur Arbeit, wenn andere zum Mittagessen gehen. Ein Echter Programmierer vergißt vielleicht den Vornamen seiner Angetrauten, aber niemals den Inhalt der gesamten ASCII- (oder EBCDIC-) Tabelle. Echte Programmierer können nicht kochen: Da Supermärkte um drei Uhr morgens selten geöffnet sind, müssen sie sowieso von Kaffee und Keksen leben.

Die Zukunft betrachtend machen sich eine Reihe von Echten Programmierern Sorgen, daß die jüngste Programmierergeneration nicht mehr mit der gleichen Lebensperspektive aufwächst wie sie selbst: Viele der Jüngeren haben noch nie einen Computer mit einer Maschinenkonsole gesehen. Kaum ein Schulabgänger kann heute noch hexadezimal rechnen, ohne einen Taschenrechner zu benutzen. Die Studenten von heute sind weich — geschützt vor den Realitäten der Programmierung durch symbolische Debugger, Texteditoren, die Klammern zählen, und benutzerfreundliche Betriebssysteme. Und das Schlimmste ist, einige dieser angeblichen Computer-Spezialisten kommen zu Rang und Namen, ohne je FORTRAN zu lernen! Sind wir dazu verdammt, eine Industrie von UNIX-Hackern und PASCAL-Programmierern zu werden?

Nun, aus meiner Erfahrung heraus glaube ich behaupten zu dürfen, daß das Schicksal den Echten Programmierern wohlgesonnen ist: Weder OS/370 noch FORTRAN zeigen irgendwelche Symptome des Aussterbens, trotz aller Anstrengungen der PASCAL-Programmierer. Selbst subtilere Tricks, wie das Hinzufügen strukturierter Schleifen zu FORTRAN, sind fehlgeschlagen. Sicher, einige Computerhersteller liefern FORTRAN-77-Compiler, aber jeder einzelne von ihnen läßt sich über eine einzige Compiler-Option in einen FORTRAN-66-Compiler verwandeln — mit `DO`-Schleifen, wie von Gott erschaffen.

Selbst UNIX scheint für den Echten Programmierer nicht mehr so schlecht zu sein wie früher: Die neueste UNIX-Version hat das Potential eines Betriebssystems, das eines Echten Programmierers würdig ist. Sie hat zwei verschiedene, leicht inkompatible Benutzer-Schnittstellen, einen geheimnisvollen und komplizierten Teletype-Treiber und virtuellen Speicher. Und wenn der Echte Programmierer die Strukturierung ignoriert, kann er sich sogar mit C anfreunden: Schließlich gibt es keine Typenbindung, Bezeichner sind sieben (zehn? acht?) Zeichen lang, und man hat Zeiger als Bonus. Das ist, als hätte man die besten Teile von FORTRAN und Assembler vereint, von den kreativeren Möglichkeiten des `#define` ganz zu schweigen.

Nein, die Zukunft ist nicht völlig schlecht: So hat sich in den vergangenen Jahren die populäre Presse sogar über die clevere Brut von Computer-Schraten und -Hackern geäußert, die Plätze wie Stanford oder MIT zugunsten der Wirklichkeit verlassen haben. Allen Anzeichen nach lebt der Geist der Echten Programmierung weiter in diesen jungen Männern und Frauen. Und solange es schlecht beschriebene Ziele, bizarre Fehler und unrealistische Zeitpläne gibt, solange wird es Echte Programmierer geben, die bereit sind einzuspringen und das Problem zu lösen, und die sich die Dokumentation für später aufheben. Lang lebe FORTRAN!