Software Tools COMMUNICATIONS

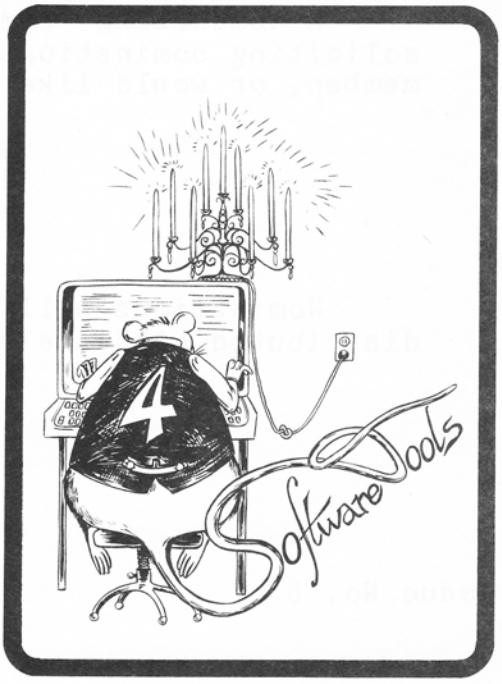
NUMBER 6 **AUGUST 1981**

--- NOMINATIONS FOR DIRECTORS OF SOFTWARE TOOLS USERS GROUP ---

Interested members met on June 22 in Austir, Texas to discuss organization of the Users Group. These members felt that the Group could best be run by a 4-person Board of Directors, responsible for all services, activites, and administrative business. Members of the Board would be elected and would serve two-year terms. After election, each Board member would assume one of the following offices and would be responsible for setting up an appropriate committe, if necessary, to fulfill the function:

Co-ordinator

Assures overall functioning of, and assumes final responsibility for, the Group and the Board. Co-ordinates activites of the other Board members and of the Special Interest Groups. Assumes final responsibility for accounting, money flow, and other business details.



This work was partially supported by the Applied Mathematical Sciences Research Program of the Office of Energy Research U.S. Department of Energy under contract W-7405-ENG-48.

PUB-402 8-81/1500

Liaison

Maintains membership list, including collecting applications and preparing renewals. Monitors incoming communications and coordinates information flow to other officers and members. (This may include setting up an information clearing house.) Assists with publications production by making available mailing labels and providing other membership information as needed.

Software Co-ordinator

Collects and co-ordinates software for the distribution tapes, including receiving new submissions and bug reports. Working with the SIGS, make final decisions on software to be included on the tape. Prepares tapes for release and assures the distribution is handled smoothly.

Publications Supervisor

Provides for editing, production, and mailing of the newsletter, the software catalog, and other future publications.

In addition, the Board of Directors will be responsible for organizing and registering the Group in an appropriate legal status, setting and collecting registration fees, appointing meeting chairman, appointing a nominating committee for elections, assuring cooperation and information flow with other organizations such as USENIX, and evaluating the needs and services of the Group.

A nominating committee has been formed and is currently soliciting nominations. If you are interested in becoming a board member, or would like to suggest a nominee, contact:

Skip Egdorf	505-667-5576
Neil Groundwater	703-893-6140
Dave Martin	213-648-9927

Nominations will close October 23, 1981 and ballots will be distributed with the next newsletter.

--- AUSTIN TEXAS SOFTWARE TOOLS USERS GROUP MEETING MINUTES ---

The Software Tools Users Group Conference was held on June 23, 1981. The meeting was attended by about 400 people. Like previous meetings, the meeting was held preceding the USENIX Conference. Thanks to Allen Akin for his excellent conference notes.

Software Tools in Pascal - P. J. Plauger (Whitesmiths, Ltd.)

Pascal was chosen to support Software Tools for three reasons: it is a popular teaching language, it offers better data structuring and checking than Fortran, and its use demonstrates tool portability in a new arena. Unfortunately, Pascal presents several problems: there are no standards for separate compilation, fetching command arguments, file naming, or raw I/O. Various circumlocutions are necessary to avoid data typing restrictions and to make the limited set of control structures more usable.

The new edition of Software Tools is not simply a rewrite. Manual pages for each of the tools have been included; expression evaluation has been added to the macro processor; the Ratfor preprocessor and the exclusive-or encryption program have been deleted.

As for the tools themselves, a number of changes have been made. The [illegible]o processor has been used to provide a standard "wrapper" for each system environment; modularization has been accomplished with the "#include" while special features have been added via "#define". A minimum environment including command lines with redirection, standard input line editing, and named files is now assumed. Standard data types character, string, filedesc, and mode have been defined. The usual collection of interface routines can be used to manipulate objects of those types.

The tools have been retargeted to UC Berkeley Pascal, UCSD Pascal, and Whitesmiths Pascal, on a variety of different machines. Conclusions: Pascal can be made usable and remain portable, and Software Tools can help even primitive environments. It seems that source code modularization, manipulation of files by name, command line arguments, command line I/O redirection, and uniform input editing rules are

mandantory for a good implementation of the tools.

Sociology and Evolution of Toolkits - John Mashey (Bell Laboratories)

What can be said about the evolution of successful programming environments?

Sociological questions are important: how well an environment adapts to different organizations, how much of it must be swallowed in a single gulp, how much it costs, and how new technology tends to spread in a given organization.

The psychological characteristics of the implementors are critical: there must be a balance between "generators" (those who create code and ideas), "analyzers" (those who watch users to find their unspoken needs and make changes to fit the code to reality), and "consolidators" (those who try to throw away pieces of code to make it parsimonious and consistent).

To reduce "creeping featurism" and other ills, one can build tools rather than monolithic systems. End users see the tools plus connectors, and tend to use them to solve many of their own problems while leaving the tools builders with a system of manageable size. Tools can evolve by enhancement, generalization, function splitting, or function deletion. In fact, one can view tools as species occupying specific ecological niches.

The more tools there are in a system, the harder it is to find the "right" one for a job. Human limitations imply the need for tools subsetting and "roadmaps" for tool us

Systems can be kept small by insuring the proper mix of people, consolidating functionality, fighting top-down redundancy, and perhaps by viewing the system as a lifeboat with limited capacity (if a new feature comes in, an old feature must be eliminated).

Data Management Tools for Large Data Sets - Bob Burnett (Battelle Pacific Northwest Laboratory)

An initial set of data management tools has been developed and implemented at Pacific Northwest Laboratory in support of the Analysis of Large Data Sets (ALDS) research project. A self describing binary (SDB) data file structure has been designed to provide a common data representation for all software components. A set of kernel data manipulation functions and supporting SDB

data access routines has been implemented.

A data editor has been implemented as a major user-level data utility. The ALDS data editor operates directly on data stored in SDB files and provides listing, searching, editing, logical subsetting, random sampling, and data transformation capabilities.

Writing a Compiler in Ratfor (With C as a Case Study) - Dan Forsyth (Georgia Institute of Technology)

The School of Information and Computer Science of Georgia Tech decided to implement a Prime 50-Series C compiler, for three reasons: a systems programming language was needed for several research projects, compiler classes needed a "tweakable" compiler, and several outbreaks of assembly language needed stamping out. Ratfor was chosen as the compiler implementation language, since PL/I runs ten times slower than Fortran 66, the Pascal compiler is unreliable, and assembly language is unthinkable.

During the course of the compiler implementation project, several items were much appreciated. Since Ratfor programs are typically 2 to 3 times larger than equivalent assembly code, ample memory was a necessity. A parser generator was useful in getting around the lack of recursion in Fortran. Heap storage and pointers (implemented with the dynamic storage routines on the Basic Tape) helped implement the compiler's data structures. The macro processor proved useful in faking heterogeneous record structures.

The final version of the C compiler consists of a front-end program of about 128K bytes and a reusable (not C-specific) code generator of about 128K bytes. It produces object code of quality comparable to the Prime Fortran 66 compiler, although compilation is fairly slow (about 700 lines/minute on a Prime 400).

Enhancements to Ed - R. R. Van Tuyl (Sylvania Systems Group)

An editor is a tool for entry, retrieval, modification, and viewing of text. Editors should be easy to use, with natural (intuitive) commands, fast operation, and positive feedback. Furthermore, editors should be easily ported to new systems. The Software Tools editor meets most of these requirements quite well. However, a few additions could help its usability.

A "view" command (v) displays a CRT screenful of text (11 lines on either side of the current line) without modifying the current line. A "print" command (p22) displays the current line and the next 22 lines. New line number syntax simplifies moving backward or forward a given number of lines.

Using only the "backspace" function (available on nearly all CRT terminals), **ed** allows the user to search and edit within a line of text with immediate visual feedback. The in-line editing capablities also provide for insertion of newlines, thus allowing the user to break lines apart manually.

A "join" command (j) permits concatenation of all lines in a contiguous group. Should the resulting line be too long, it will be split into several lines before appearances of "break characters".

Ed requires few additional hardware and operating system features. A non-destructive backspace feature on user terminals is essential. For in-line editing, it is desirable for **ed** to regain control immediately after a control character is entered by the user. Data transmission at 240 cps or higher is definitely a plus, as is an operating system that gives quick response to interactive programs.

MIRAGE: Mellon Institute Research Assembler - Daniel Klein (Mellon Institute)

An assembler can be viewed as a mapping function from mnemonics to binary machine code. The map must be cognizant of various pieces of header information (including information on the target machine architecture), addressing modes, opcodes, pseudo-operations, and literals.

MIRAGE is a translator for a descriptive language covering these pieces of critical information. In some sense, it resembles a parser generator in that it transforms a largely non-procedural description into a procedure for performing the assembly mapping.

MIRAGE is still being developed; however, assembler descriptions have been prepared for the MOS Technology 6502, RCA 1802, Intel 8080 and 8048, Motorola 6809 and 68000, Data General Eclipse-C, Dec PDP-8, PDP-10, and PDP-11, IBM 370, and Zilog Z80.

Adding Data Structure Support to Ratfor - Wade Shaw (Execucom Systems Corp.)

Execucom routinely handles the porting of a large FORTRAN-based software package to more than 20 radically different target machines. The clarity and portability of this package is sometimes hampered by the limited data structure support of FORTRAN. By using an extended version of the Software Tools macro preprocessor, simple packed and unpacked data structures can be added to the Ratfor preprocessor.

Data structures are frequently implemented in FORTRAN by "equivalencing" a number of data items to the elements of an array. This is generally not acceptable, as it is easy to make a mistake in setting up the equivalence, it is difficult to add or delete structure fields, it is non-portable, and it doesn't work at all if the base array is a subprogram formal parameter.

These problems are overcome with the following scheme. Use a two-level definition mechanism in which a "group" describes the actual record structure, while a "record" is an instantiation of a particular group. The macro processor is used to read environment descriptions and produce correct equivalence statements for the elements of a particular record. In the case of formal parameters, the actual parameter is copied into a local array, where equivalencing can be used.

Packed records require some additional effort. As a by-product of interpreting a group description, the macro processor produce "descriptors" which can be interpreted at run-time to pack or unpack a structure. It is then necessary to perform the usual FORTRAN operations only on members of unpacked data-structures.

An article will appear in the next newsletter giving more details on this approach.

Implementation of Software Tools Under TOPS-20 - Bruce Dawson (Digital Equipment Corp.)

The implementation of the Software Tools under TOPS-20 proved to be a relatively straight-forward task. The major problem encountered is the special meaning of the question mark to TOPS-20. When a question mark is encountered by TOPS-20, TOPS-20 enters a special help mode. This is a problem when trying to redirect error output with the shell.

Implementation of Software Tools on the UT2D Cyber - Curt Webb (U.T. at Austin)

The implementation of the Software Tools on the UT2D Cyber had the typical problems encountered by other groups installing the tools on CDC machines. These problems included character sets and Fortran peculiarities.

Software Tools as the Basis for a Second Software Engineering Course - Walt Brown (Moravian College)

The Software Tools book has been successfully used as the basis for a second course in the undergraduate curriculum at Moravian College. Major advantages are that students are exposed to good engineering practices on non-trivial non-numeric programs early in their career. The course is popular after it is over. The major disadvantage of the book as a text is that since the book is meant for experienced programmers the book tends to be somewhat terse for a second computer science course.

Results of Ratfor Dialect Survey - Walt Brown (Moravian College)

The results of the Ratfor dialect survey were presented. It was interesting to note that many of the "enhancements" represented in the most popular Ratfor dialects have found there way into the Ratfor on the standard tape. An article will appear in the next newsletter giving the details of this survey.

Special Interest Group Meetings

At the end of the meeting, the group s[illegible]nto the text, ratfor, networking, standards, primitive[illegible] and program development aids special interest groups. Much interest was expressed in the networking and standards special interest group.

--- SOFTWARE TOOLS USERS GROUP ADDRESS ---

When writing the Software Tools Users Group please address all correspondence such as change-of-address and newsletter contributions to:

Software Tools Users Group
242-1259 El Camino Real
Menlo Park, CA 94025

Please mark correspondence clearly on the outside indicating its contents with words such as

"Attention: Change of Address".

--- NEW TAPE DISTRIBUTION CENTER ---

The National Energy Software Center at Argonne is considering adding the Software Tools tape to their distribution library. New additions to the library are selected based on the number of outstanding requests. Anyone interested in obtaining the Software Tools through the Software Center should contact:

National Energy Software Center
U.S. Department of Energy
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439
Phone: 312-972-7250

--- SOFTWARE TOOLS CATALOG PUBLISHED ---

The Sp[illegible]981 Software Tools Catalogue has been published by Richard K[illegible]ig of Intelligent Decisions, Inc. The catalogue lists interesting work that has been done using software tools concepts as well as people to contact about specific tools. The catalogue can be ordered from:

Richard Kiessig
Intelligent Decisons, Inc.
1235-C Henderson Avenue
Sunnyvale, Califlornia 94086

Lawrence Berkeley Laboratory
Debbie Scherrer
CSAM — 50B/3209
University of California
Berkeley, CA 94720