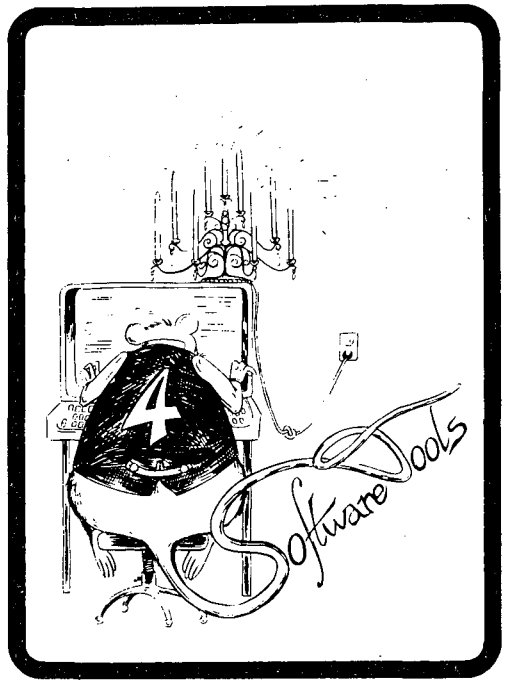
DISCLAIMER

Software Tools COMMUNICATIONS

NUMBER 11 DECEMBER 1983

In This Issue



This work was supported in part by the United States Department of Energy under contract no. DE-AC03-76SF00098.

The First Basic Tape Revision

Nancy Deerinck

A frequent question we are asked on the STUG Hotline is whether on not a particular version of the Basic Tape is "the latest version". A quick glance at the Tape Order Form indicates that there are many "versions" of the Basic Tape, sparing some implementation procedures for different operating systems. What is not immediately apparent is that there has never been a major revision. We are still distributing Version 1 of the Basic Tape.

Much effort has already gone into producing Version 2, including the formation of the STUG Standardization Committee, chaired by Bill Meine. In view of how much effort goes into an actual revision, it was decided to set a Software Tools Standard before it will be produced. Under review is the proposed standard Ratfor as well as all the machine-independent sub-routines and many machine-dependent primitives.

This effort represents the first Software Tools specification ratified by a large body of users. The focus of the committee is to isolate the truly technical objections which will affect all future work. Bill Meine assures STUG members that a "fixit" program will be available to upgrade all existing applications as much as possible.

The Standards Committee is composed of over 50 implementors, and it is intended that the new Basic Tape will provide the best possible Tools for the widest range of operating systems. Standarization will greatly enhance the portability and future development of all our Tools.

Joe Sventek
Bob Upshaw

The majority of the rejections concerned typos and not enough information in the manual entries. We plan to use all of these useful comments to upgrade the manuals for the next ballot pass. The next ballot, containing the final proposals and counter-proposals for standardization, will be mailed out in November.

Providing there are no delays in getting the ballots out and in tallying up the votes, it is hoped that there will be documentation and manuals describing the Software Tools Standard at the January UNIFORUM conference in Washington D.C. We will also make these manuals available through the mail. See the next newsletter for more details.

Because of the standards effort, it is doubtful that a new release of the basic tape will be available at the January UNIFORUM conference. However, some of the implementors at the meeting talked about putting together a "toys" tape for the January conference. This tape would contain such toys as yacc, lex, lisp, the new rat4 compiler, etc. These toys would be in "pre-release" form, i.e. the final versions released on the next tape may be different. However, they will be there for you to play with in the meantime.

P.S. Don't forget to read Carousel's article on Software Tools in the November issue of BYTE.

Implementors Notes

Theresa Breckon

On Monday, Nov 9, an implementors meeting was held in Berkeley. All 8 hours (some of us stayed for 11 hours) were spent discussing the responses to the standards package mailed last month. All implementors were sent a letter asking if they wanted to participate in the standardization of the Software Tools library routines and primitives and the ratfor language. Those who responded affirmatively were also invited to the Implementors meeting. The STUG members who attended were:

Theresa Breckon
Barbara Chase
Phil Davidson
Nancy Deerinck
Richard Harrington
Richard Karpinski
Dave Martin
Bill Meine
Ken Poulton
Debbie Scherrer

Call for Papers

1984 Winter UNIFORM Conference

Presentations are invited for the Software Tools Users Group meeting in Washington, D.C. on January 17, 1984. The Software Tools Users Group meeting will again be held in conjunction with the Usenix meeting, January 17–20, 1984 at the Washington Hilton. There will be a one registration fee for the Software Tools and Usenix sessions.

Talks may include descriptions of projects using Ratfor and/or the Tools, newly-created or enhanced tools, software portability and programming environments, thoughts about future directions for the tools, or other areas of interest to the tools community. Suggestions for other types of presentations are also solicted. Abstracts for proposed talks should be submitted to:

Nancy Deerinck
c/o STUG
1259 El Camino Real #242
Menlo Park, CA 94025

They must contain the following information:

Title
Name of author
Name of company
Mailing address
Phone number (network address, if any)
Audio-visual requirements

The conference committee intends to produce a proceedings consisting of short (less than 10 pages) papers by the authors on the subject of their presentation, as well as all abstracts. Submission of a paper is not required, although it is strongly recommended. Papers will be collected in camera-ready form at the conference.

Unicom Proceedings

Nancy Deerinck

We still have copies of the UNICOM Conference Proceedings (Winter 1983, San Diego) available for $25.00 ($35 overseas). In addition to a wide range of UNIX related articles, the Proceedings contains the following papers which were presented by the Software Tools Users Group.

- Users Group Status Report
- Rockies Assoc. for Tools Report
- Software Tools in C?
- Enhancements to format
- A Portable Mail System for Software Tools
- Interactive Data Analysis using the Software Tools
- New Tools for the Virtual Operating System
- Update on Software Tools Implementation
- Tools in Australia
- LISP for the Software Tools VOS
- West Coast Implementors Group Proposed Standards

Copies can be ordered directly from STUG. An order form is included in this newsletter.

New Pattern Matching Algorithm

Nancy Deerinck

A new algorithm for the pattern matching routines will be available on the next release of the basic tape. Although the algorithm has been improved, the pattern matching subroutines and their arguments remain the same, with one exception. The last 2 arguments, the tagged pattern arrays, of the routine AMATCH were deleted. To be more compatible with UNIX pattern matching syntax, the symbols used with substrings (i.e. tagged patterns) were changed. Curly braces in makpat and amatch were replaced with '@(' and '@)'. The syntax for substituting the value of a matched substring was changed from '$n', where n ranges from 1 to 9, to '@n' in the routines maksub and catsub.

The old pattern-matching algorithm involved backtracking when a partially successfully match failed. This backtracking method had three main faults; execution was slow, it tried to store pointers in the character pattern array, and extra working space was required in the encoded pattern array to do the matching. The new algorithm is essentially the one used by UNIX ex(1), grep(1), etc. It was originally developed by Ken Thompson of Bell Labs. We have extended Thompson's original algorithm to handle substrings (tagged patterns). Backtracking is not required and so the encoded pattern array is free of pointers and the extra space previously necessary for matching. In other words, the encoded pattern can safely be stored in character array.

The routine makpat still "compiles" the regular expression into an encoded pattern. However, it now compiles it into a non-deterministic finite state machine (NDFA). The routine amatch then simulates the execution of this "machine" to determine if a string matches. Amatch examines each character in the line to be matched against a list of possible characters. During this examination a new list of all possible NEXT characters is built. When the end of the current list is reached, the new list becomes the current list, the next character from the line is obtained, and the process continues. This technique of only comparing against legal matches makes the algorithm very fast.

This algorithm is described in the June 1968 CACM article by Ken Thompson called *"Regular Expression Search Algorithm"*. It was implemented at LBL by Whei-Ling Chang, of RTSG. She also enhanced the algorithm to handle substrings. Her paper, *"Pattern Matching in the Software Tools"*, which describes the enhanced algorithm, will be available early next year.

New Hot-Line Number

Nancy Deerinck

We have had a tremendous response to the STUG hotline, and encourage all members to use it for questions, referrals or suggestions. We no longer have the direct line available, so all calls must go through the RTSG switchboard. Calls to the old number will be forwarded, but only after many extra rings.

There are several people available to help with STUG matters, so please ask for a Software Tools Consultant when you reach the switchboard.

The Number: (415) 486-6411
The Times: 7a.m. to 6p.m. (Pacific Time)
Monday — Friday
(It may be possible to leave
a message at other times)

Membership Renewals

If you are uncertain as to when your membership expires, check the current expiration date on the address label of this newsletter. 3Q85 for example, means your membership is good until the third quarter of 1985. Yearly memberships extend one year from the quarter you applied, instead of for a fiscal or calendar year.

RatsNest Revisited:
The KERMIT File Transfer Protocol

Dave Martin

In our last episode (you may recall) we had tentatively decided to use the MMDF link-level protocol for our dial-up networking project. Since that time we have become aware of another popular protocol which has some major advantages, not the least of which is the ready availability of several implementations. This protocol is called KERMIT and was developed by Columbia University in response to a communications requirement between a DECSYSTEM-20, IBM 370-series mainframes and various microcomputers.

Although KERMIT is somewhat less flexible in the host-host capability negotiation stage than MMDF, it appears adequate for the great majority of known machines and does have the advantage of providing conventions for both ASCII and stream binary file transfers.

(The following is an excerpt from the KERMIT PROTOCOL MANUAL)

The KERMIT protocol requires that:

- The host can send and receive characters using 7- or 8-bit ASCII encoding over a EIA RS-232 physical connection, either hardwired or dialup.
- All printable ASCII characters are acceptable as input to the host and will not be transformed in any way. Similarly, any intervening network or communications equipment (TELENET, terminal concentrators, port selectors, etc) must not transform or swallow any printable ASCII characters.
- A single ASCII control character can pass from one system to the other without transformation. This character is used for packet synchronization. The character is normally Control-A (SOH, ASCII 1), but can be redefined.
- If a host requires a line terminator for terminal input, that terminator must be a single ASCII control character, such as CR or LF, distinct from the packet synchronization character.
- When using a job's controlling terminal for file transfer, the system must allow the KERMIT program to set the terminal to half duplex, infinite width (no "wraparound" or CRLF insertion by the operating system), and no "formatting" of incoming or outgoing characters (for instance, raising lowercase characters to uppercase, transforming control characters to printable sequences, etc). In short, the terminal must be put in "binary" or "raw" mode, and, hopefully, restored afterwards to normal operation.
- The host's terminal input processor should be capable of receiving a single burst of 40 to 100 characters at normal transmission speeds. This is the typical size of a packet.
- If a host requires padding, the padding character must be a control character (including DEL).

(End of excerpt)

Quite a few systems seem to be able to meet the above requirements and implementations are available for the following systems:

VAX-11/7xx running VMS
DecSystem-10 running TOPS-10
DecSystem-20 running TOPS-20
IBM PC running MS-DOS
Apple][running DOS
Heath Z100 (Intel 8086)
CP/M 2.2 (Intel 8080/Zilog Z-80)
CP/M 3.0
DEC PDP-11 running RT-11
IBM 370-series running VM/CMS
Various machine running UNIX

These implementations are written in a variety of languages, including assembly language, BLISS, C and Pascal. We are, of course, working on a RatFor implementation for distribution on the Software Tools tapes, but in the mean time we are planning to make arrangements with Columbia University to become an alternate source for their distribution tape.

Notes on the Archiver

Paul Howson
Desmond Fitzgerald & Assoc.
East St.Kilda
Australia

There are two versions of the archiver in circulation which are not compatible:

(1) Those based on the original book version, using character counts.

(2) Those based on Allen Akin's archiver from the 1981 distribution tape, using headers and trailers.

We are currently using the header/trailer archiver for the reasons that (i) we only want to use *one* type of archiver and (ii) only the header/trailer archiver produces files which are portable between machines (via magtape) and which can be edited or otherwise filtered (e.g. with the **style** tool) without being corrupted.

It is obvious from the VAX tape (which we collected at Unicom in January and subsequently mounted on a Prime), that the STUG VAX sites are in the habit of using the character count archiver. It is also evident that a new family of useful tools based on the character count archiver have been developed (acat, alist, asam, asplit, axref, rar). We had great difficulty unbundling the archived files from the VAX tape using the character-count archiver. The counts were randomly corrupted due to the effects of trailing blanks somewhere in the chain between STUG's VAX and our Prime.

The "-s" option proved inadequate because most archives were nested and we had to modify the archiver so that "-s" option could cope with nested archives. This is the sort of fudge we dislike having to spend time on.

We appreciate the benefits of the character count archiver:

(1) It is faster when there are efficient file-size, file-skip-by-size and file-copy-by-size primitives available (much faster).

(2) It allows the archiving of binary files (i.e., files possibly not containing newlines at all).

We agree that these advantages are considerable and this is what has prompted this article.

We would like to place before the standards committee of STUG a proposal for a hybrid archiver that uses the best of both character count and header/trailer methods.

Basically, there would be a header containing a character count at the start of each archive member, just like there is now in the character count archiver and a trailer at the end of each archive member, just like there is in the header/trailer archiver. That way, one could use the character counts *most of the time* for speed and efficiency. But if for some reason they became erroneous (due to editing the file or transferring it via magtape, etc), then it would be easy to restore the counts with the help of the header/trailer pairs and a special "fix it" tools (perhaps the "asplit" tool or an **s** option on the archiver). At the moment, without the trailers, there is no way of restoring corrupted character counts in a foolproof way in nested archives (a ubiquitous case).

Now, while on the topic of the archiver, here are a couple of points of which you may not be aware, which also impact on *how* the character count archiver is implemented. They relate to some idiosynchracies of the Prime architecture, of which VAX users would be unaware.

The problem is this: on the Prime, files are just like Unix files... long strings of bytes (actually 16-bit words). Very nice for Software Tools. However, a blank compressin scheme is used. Runs of three or more blanks are encoded as a flag byte (RHT), plus a count byte.

This means that the *size* of a file inbytes has two meanings: (i) the size as measured by multiple calls to **getch** which counts the decoded blanks and (ii) the physical number of bytes in the file. Very confusing!

Can you see the consequences of this for the character count archiver? It means that if you implement "primitives" like **acopy** (copy n bytes from file1 to file2), **fsize** (size of file in "bytes") and **fskip** (skip **n** bytes on a file), which are very easy to implement provided we stick to *physical bytes,* then you can't go and mix these counts with counts using **getch** and **putch.**

So really, what we're saying is that the archiver using counts should talk about file sizes in units known only to these file copy/size/skip primitives and should not mix these with counts using **getch** and **putch.** Of course, having done the copy or skip or whatever, it is perfectly permissable to use **getch** and **putch** for the headers and trailers.

Lastly, we would like to suggest some additional archiver tools and/or options.

(1) The archiver should be able to use the date/time stamp to update only those files which are newer than the existing archiver versions.

(2) How about a change-name function to change the name of an existing archive member. This case occurs typically when you might change the name of a subroutine but still want it to go in the same place in the archive so as not to confuse the TCS system.

Matters of UNIXness, VAXiness and currect practice aside, these suggestions at least have the merit of being sensible.

Changes to the the Archiver

Debbie Scherrer

[The following notes were submitted in response to the article above.]

Regarding the archiver: about two years ago the Implementors Group met several times and discussed needed additions to the package. One of the primary concerns was a single archiver, rather than the 2 we had on the original tape. We decided that we needed one that would be able to handle either convention (character counts or header/trailers).

As a result of those discussions, I rewrote the existing archiver to do both. It's default is to use header/trailers, although this default can be changed at compile time by adjusting a macro. If character counts are desired, there is a "-c" flag. If both character counts and trailers are desired, there is a "-ct" flag. It was also adjusted to allow for binary files (using the getbyt/putbyt pair). However, binary files cannot be used in nested archives because there is no way to copy a file which contains both binary and character data. We could not come up with a reasonable set of primitives to do that. In fact, with that limitation one couldn't "cat" an archiver, or do a "find" on it, or anything else than "cp" it. I think we also eventually changed the names of getbyt/putbyt to something which implied less of a byte and more of a binary unit.

We really need a lot more discussion about handling binary data. However, I think the archiver I came up with is what the group concensus seemed to require.

Software Tools in Norway

Peter J. Story
Kongsberg vapenfabrikk
Kongsberg, Norway

Tools on Norsk Data Machines with Sintran III OS

We are now nearing the completion of the first implementation of Software Tools on Nord machines, so I thought we would at least register the fact in case anyone else was thinking about it.

The current state is that all the portable tools are installed with exception of ED, SEDIT, and MACRO (lack of time) and SHELL (haven't figured out how to do this sensibly yet). In addition, we have taken the TEXT Control System commands ADMIN, GET and DELTA from the VMS tape, and with almost no problems added them to the toolset on Nord.

Problems which was have had include:

1) Not reading all the newsletters carefully enough to pick up notes on problems encountered by others.

2) Sintran is somewhat old fashioned in its treatment of commands and insists on converting everything to upper case characters. An irritating habit which makes commands like FIND a bit tricky.

   No solution to this at present, so we'd be interested to hear from other implementors with similar operating systems. We had thought of:

   - prompting for arguments as input
   - using some kind of unshift escape character
   - converting all arguments to lower case, and using some kind of shift escape character
   - but none of these really appeal.

3) The convention with filenames in Sintran is that "new" files must be explicitly indicated as such (by surrounding the filename in quotation marks).

   We have ignored this — new files may be created by tools regardless of whether they are quoted. However, for fanatics, we have said that if quotation marks are given, then the file MUST not exist already.

4) Although Sintran can start a command from a program, this command replaces the calling command, and you cannot therefore wait (in the parent) for the child process. This makes pipelines tricky and is the main reason for not having done anything with the shell.

   Pipes will almost undoubtedly turn out to be intermediate files, unless the Nod "xmsg" interprocess communication turns out to be something other than we currently believe it to be.

The main motivation for implementing the tools was to get some kind of text control system, which we have acheived. In the process we have gained the ability to step outside Sintrans "flat" file system with a 256 file/user limite, by using "ar". Whether the groups in question realise the wider potential of the tools they have been given is an interesting issue to which we'll come back later!

Tools on VMS

One of our groups has installed the tools on VMS — about half an hours work! A nice package!

Network?

I see that there is some interest in an electronic mail facility for Software Tools users. Given the common interests of Usenix and STUG, I would very much like to see this in connection with USENET/EUNET, if we can get a license free version of uucp going. We are on the point of joining the net, and if all goes as planned, the address should be:

...decvax!mcvax!kvport!vax-4!pete

Software Tools in Pascal

Guy Lapalme
Univ.de Montreal

Computer science students at the University of Montreal are taught programming by a sequence of two courses: the first one gives the fundmentals using PASCAL based on the book by Keller[1]. The second one extends those ideas and aims at having the students involved in bigger projects; they will acquire a discipline of programming, modify already existing systems and reuse other people's work. Thus, they become more aware of the need of having good programming style and good documentation.

I have used "Software Tools in Pascal" as a textbook for this second course. The programs from this book are examples of good programming and good documentation and are available in machine readable form from Addison-Wesley. In fact, they are a bit too good in that respect in the sense that they constitute a much higher standard of what is done in the industry so the students are a bit "spoiled" in having everything spelled out in detail. On the other hand, by reading those programs they should, and have, developed by "osmosis" good programming habits.

In practical terms for the last semester, I had the students modify the FORMAT program so that it could run on our CDC system and implement the .ul command on a line printer (this was a 2 week assignment). The next assignment (3 weeks) was to modify the same program but this time using a local extension of the CDC character set and also to implement an hyphenation procedure which, fortunately, is a straightforward task in french. All in all, this experiment was a success. The students were able to learn methods and "tricks" by seeing how things should be done instead of a trail and error (and correction) scheme; they were also able to produce *systems* instead of *programs in a few weeks time.*

The simplest way to modify a system is through a text editor which is available on all computers but his approach can cause a proliferation of quick patches which usually lead to the destruction of the initial structure of the program. Fortunately, Control Data computers (6000 and CYBER series) offer a very attractive alternative: a source management tool called UPDATE [2] which enables us to keep only one copy of the original text and to make temporary and reversible changes to the file. We simply create a file indicating which source lines have to be changed, UPDATE applies those corrections to the file giving a temporary file which can be compiled. A "stream editor" found on most systems could also be used, but in that case, the corrections must be in increasing line number which is a bore; but more important, in that case, we cannot put together all corrections relating to a single goal. With UPDATE, corrections for dealing with our strange character corrections for hyphenation, corrections for adding commands can be logically separated even if they refer to lines over the whole program. The appendix gives the major steps that were taken to transform the tape available from Addison-Wesley to the format used by UPDATE. The students (and teaching assistants) were, at first, annoyed that they could could not use the test editor and had to retype the whole line in order to correct a few characters; but using the system, they realized its benefits which have already been reported by Mordechai Ben-Ari[3]. Another benifit of this scheme is that it is very easy to spot the modifications which have been made, so the correction task (for the T.A.) is much easier.

This experiment was a success and we plan to redo it next year, using other programs, of course. STiP is in fact a gold mine of assignment projects.

Bibliography

[1] *Keller, A.,* **"A first course in Computer Programming"** using Pascal, McGraw-Hill, 1982

[2] **UPDATE Version 1 Reference Manual,** Control Data Corp, rev.E, publication no. 60449900, 1982.

[3] *Mordechai Ben-Ari,* **"Why you should not time-share",** Software Practice and Experience, Vol.9, pp.339–340 (1979).

Appendix

Conversion of the STiP archive tape to an UPDATE program library.

UPDATE deals with a program library file which is structured in "decks" and "common decks" but all in a one level directory. This scheme is very well adapted for FORTRAN programs but less so for PASCAL. The following sketches the main steps we have done to squeeze the software tools into this mold.

As there is only one level of directory in which all deck names must be distinct, we must remove duplicate files which are present in different directories of STiP: that implies keeping the "manual" files. Each remaining file is then transformed to a "common deck" whcih can be "called" (i.e., included) from other common decks. We finally create a "deck", the "wrapper" (cf STiP p.322) from which each program can be created by inserting in it the appropriate "call".

We have transformed the ASCII into the display code (6 bits) of the CDC even if a new version of UPDATE can handle ASCII characters because other programming tools in the CYBER do not handle ASCII; another reason is the fact that the Universite de Montreal computing center uses a local ASCII coding for french character sets. This

combination revealed a breach in the portability effort of STiP; in fact, if the character set of the program is not the same as the one of the date (i.e. N <> ord (char(N)) then many procedures will fail; for example; getcmd, initdef ... In fact, everywhere a "character" variable is initialized with "ord('X')", one should use "LETX" instead. This was done in the original Software Tools in RATFOR!

Report on Toronto Unicom

Tom Strong
Editor of ;login:
USENIX Assoc.

The following reports on presentations given at the Toronto The following is a report on the presentations at the Software Tools Users Group meeting that was held on July 12, 1983, in conjunction with the USENIX Association Toronto *Conference. My notes were supplemented by copies of papers submitted for the proceedings and viewgraphs, when they were available.*

Topic: A Bit of History

Speaker: Brian W. Kernighan
Bell Labs, Murray Hill, N.J.
[uucp: research!bwk]

This talk presented the early history of the Tools, up to the time the Tools Group was formed. The speaker started working at Bell Laboratories about 1969. The Labs had been working on portable Fortran for a while, even then. In April 1973, he proposed a "rational Fortran" (**ratfor**). The criteria for the language were: (1) the language should make it easier to write correct programs, and (2) it should be easy to translate into portable Fortran. Ratfor provides control structures from the languages B and C and new syntax elements like free-form input, "define", and "include". Ratfor does not know about Fortran and everything that is not recognized is passed through unchanged.

In April of 1975 Mr. Kernighan and P. J. Plauger began working on a book on how to program well. The emphases were on making programs to help develop other programs, on programs that work together, and on programs that are portable. For portability they defined a small set of "primitives": basic operations for accessing the local operating system. By using only the primitives to access the operating system, operating system dependencies are localized and programs using them are portable.

The book, *"Software Tools"* was published by Addison-Wesley in 1976. The publisher also offered a tape with the software described in the book for a nominal fee. These were picked up by people at Lawrence Berkeley Laboratory, University of Arizona, Georgia Institute of Technology, and University of Maryland. These groups expanded the software from the book, made rock-hard primitives, and eventually got the Users Group going.

Topic: A Command Line Scanning Package

Speaker: Jerry J. DeRoo
Univeristy of Toronto
Ontario, Canada [uucp!utzoo!utteeth!jerry]

This talk described work originally done while the speaker was employed by Garmaise & Associates, Ltd. in Toronto. The company formed a development group to provide the capabilities necessary for generating high quality production systems. The group chose ratfor as their implementation language, and used the code on the Addison-Wesley tape as the basis for their implementation of their software development environment. They made several additions to the original tool set: more extensions to the *macro* and *ratfor* preprocessors, additional string handling routines for the library, and a version of the UNIX Standard I/O system, much like the Tools', but implementing token recognition on all streams.

Then they found out about the Software Tools Users Group! They purchased the VMS implementation tape and added those tools to their environment. They kept their own versions of their original tools, but undertook to maintain compatibility with the Tools group.

They found a need for a consistent and general command line processor and developed the following criteria.

(1) Positional arguments were to be avoided unless specifically useful, as in *copy*.

(2) Flag arguments could be used and were to have mnemonic names, with abbreviations allowed.

(3) Keyword arguments could be used, with the same restrictions as flag arguments.

(4) Flags and keywords that have the same function in more than one tool were to have the same syntax.

(5) Un-recoverable errors were to produce a useful message and a valid example.

(6) Required missing arguments could be prompted for if the tool was being run interactively.

They implemented these criteria with a set of four user callable functions and a command line template.

invoc examines the command line and compares it with the template. The template defines the type of each allowable argument (flag, keyword, etc.), and the name and abbreviations, in any.

gargc determines the number of arguments.

gkey gets the value of the specified argument.

gargn gets the type and value of a positional argument.

They found that using their command line processor eliminated a lot of coding drudgery and made the code easier to write and maintain.

An implementation of the routines is being prepared for evaluation by STUG.

Topic: Minimal Test Cases for the VOS Software Tools

Speakers: Ben Dominico, Russell Rew
National Center for
Atmospheic Research
Boulder, Colorado

This talk described a set of tests designed to indicate whether or not each tool functions at a minimal level. There are tests of all the tools on the STUG distribution except *sh* and *spell*, and for *tcs* and some locally-developed tools. Russell Rew wrote the tests and Ben Domenico tested and packaged them.

The set of test cases were written for the following purposes.

(1) To determine for each tool whether it works at all. Such minimal testing, in their experience, uncovers the vast majority of problems that develop.

(2) To check out new versions of the operating system, the Fortran compiler, *ratfor*, the library, and/or the tools.

(3) To check out new Virtual Operating System (VOS) implementations.

(4) To provide simple examples of tool usage.

Each test is a self-contained script; no external test data is needed. Each generates a one line message stating whether or not the test was successful. The tests are archived in the order necessary to account for dependencies on other tools in the tests.

[The speaker has submitted a paper for the conference proceedings. It includes a listing of the test scripts. STUG plans to include these test scripts on the New Basic Tape.]

Topic: Software Tools at Moravian College

Speakers: Bader & Bader

This talk described many additions made to the basic STUG distribution by the speaker and his son. Their main thrust has been tools for numeric processing. A partial list of the tools described:

- many input and output routines oriented to numeric processing
- a code beautifier
- primitives for a Tektronics software simulator for the ADM-3A
- a simulator for a Calcomp plotter for output on a line printer or CRT
- various math packages including a dynamic simulation package for linear or non-linear ordinary differential equations
- many other tools for mathematics.

Topic: Imposing Character-Oriented I/O on a Record-Oriented System

Speaker: Douglas Orr

This talk described three methods tested to improve the efficiency of processing character I/O. The methods tried were:

(1) data compaction (e.g., by substituting for extra blanks),

(2) using virtual files

(3) hand-coding (i.e., not using the tools library)

They found that their benchmark of the original tools library took 122 seconds and the three methods above took (1) 102 seconds, (2) 47 seconds, and (3) 48 seconds.

Topic: Environments and Search Paths in the Software Tools

Speaker: Theresa Breckon
Lawrence Berkeley Laboratory
Berkeley, California

This talk described various ways that "environments" are used on UNIX and proposed a way that a similar facility could be portability implemented in the Software Tools.

In UNIX, environments are a way for a user to specify names *("environmental variables")* and associated strings for use by programs. The system provides mechanisms to define, change, and un-define the variables and their values. The currently defined variables are available to all programs that choose to use them. Environmental variables are typically used to hold file and directory names, such as the home directory of the user and the default shell, and information about the type of terminal being used. The most commonly used environmental variable is called **PATH**; it contains the list of directories where the

shell is to look for each command entered (the "search path").

Environmental variables are typically set at login time but may be changed at any time. For example, the PATH variable usually contains a list of installation-defined directories where commands are kept. Users who use scripts or write their own commands will usually add the name of the directory containing them to the list of directories in the search path. That way the shell does not need to know specific directory names and the user does not need to enter full pathnames for personal commands and/or scripts. When a user wants, for example, to test some new commands (e.g., a new release of the Tools) the directory containing them may be temporarily added to the search path. Then the new commands will be used, even though they have the same names as the old versions of the commands.

More generally, environmental variables are useful because they allow programs to be written without specific file names, or other information that the user might want to change, built into them. The program is written to use the value of the appropriate environmental variable if it is defined, or otherwise to use a pre-defined default. Programs written this way are more portable and easier to implement.

The set of routines required to support environments in the Software Tools is small and needs to be implemented only once. New environmental variables can be added as needed without affecting existing tools. A Tools implementer can choose to use the environment or not, as appropriate for the system.

The environment model presented is based on the concept of a symbol table: the environment is simply a set of names and associated values. The symbol table is manipulated using existing portable tools. Values may be any reasonable string.

Two basic commands are proposed for use with the environment:

setenv: a builtin shell command to set or un-set variables, and

printenv: to print out the values of the variables in the current environment.

These are supported by three functions:

envset: to install a name and value in the symbol table,

envget: to retrieve a name and value from the table, and

envrm: to remove a name and value.

The speaker described how environments have been implemented in the Real Time Systems Group and how the search path is used. A new version of *open* called *pathopen* was written. It is just like *open* except that if the file name is prefixed by *+name/* and if there is an environmental variable *name* its values will be used with the specified file name to construct a pathname. If there is no variable *name* the file name specified will be used, just as in *open*. For example, to cause the shell to search for *cat* first in the users directory *bin*, then in the directory */newtools/bin* then in the system directory */usr/bin*

```
setenv PATH ~/bin /newtools/bin /usr/bin
```

The shell would then open the file using pathopen with the following filename:

```
+PATH/cat
```

The use of an environment and search paths offers increased portability for the Software Tools, easier installation for the Tools implementer, and better control for the Tools user. RTSG plans to submit the portable environment model for distribution in the extensions section of the next Software Tools basic tape. After refining the model, based on input from Tools users, they plan to submit the environment package, along with search path code for the shell, to the standards committee.

[The speaker has submitted a paper for the conference proceedings.]

Topic: Performanace of Tools: Minis versus Micros

Phil Scherrer
Carousel MicroTools
El Cerrito, California

Carousel MicroTools is doing ports of the Software Tools to microcomputers. Their implementation on CP/M-80 has been running for some time. They are working on an implementation for the MS_DOS system.

The speaker made a simple script file to test his feeling that the Tools ran faster on CP/M than he expected. The following table was presented.

Machine	Cost†	OS	Users	Tries	Time (sec.)	User Cost‡	Notes
11/780	25	UNIX	2	10	15. ±1	380	UNIX; not the Tools
11/780	25	UNIX	2	10	50. ±3	1300	a quick & dirty Tools implementation
11/780	35	VMS	23	6	247. ±85	8500	a busy day
11/34	5	RSX	1	1	154.	770	
8085	1.2	CP/M	1	2	335.	400	6MHz, 20Mb 8″ hard disk
8085	0.8	CP/M	1	2	260.	210	6MHz, floppies plus M-Drive
Z80	0.5	CP/M	1	2	480.	240	4MHz, 10Mb 5″ hard disk

† around $10000 per unit
‡ Cost × Time

Implementors List

The list of Implementors continues to grow as more new members are successful in bringing the Software Tools up on a variety of operating systems. Although we receive many applications from veteran implementors, the majority of applicants are simultaneously ordering the Basic Tape and have not yet gotten their fingers into implementation.

We would like to hear from you after you have installed the tools. Please take a moment to send us your name and the machine(s) on which you have implemented the tools or call the STUG Hotline (particularly if you have broken ground on an obscure machine!).

Mark Aaker
NASA Ames
Mail Stop 241-3
Moffett Field, CA 94035
(415) 965-6410
+VAX 11/780

Eric Arman
The Brookings Institution
1775 Massachusetts Ave. N.W.
Washington, D.C. 20036
202-797-6185
+DEC-10

Dr. Michael Baker
14 Lissel Road, Simpson
Milton Keynes MK6 3AX
ENGLAND
0908 670346 (home)
0908 653938 (work)
+DEC-20

David A. Baumann
3340-18 Cheviot Drive
Fort Wayne, Indiana 46816
(219) 461-5760
+PDP-11,RSX11-M

Steve Bearman
Scripps Inst. of Oceanography
DSDP A-031
U.C.S.D.
La Jolla, CA 92093
(619) 452-3526
+HP-1000

Nelson Beebe
Department of Physics
University of Utah
Salt Lake City, Ut 84112
(801)581-6901
+DEC TOPS-20

Frank Benjamin
c/o C.W. Holeman
1160 W. Chase Ave.
El Cajon, CA 92020
619-444-0446
+DEC RT-11

Lewis J. Bornmann
General Electric Co.
1277 Orleans Drive
Sunnyvale, CA 94089
(408) 734-4980
+VAX(VMS/UNIX)
+CDC Cyber 700(NOS)

Michael Bourke
The Wollongong Group
1135A San Antonio
Palo Alto, CA 94303
Work: 415-962-9224
ARPA: bourke@sri-unix
+Perkin-Elmer
+UNIX and OS/32

Frank Bradford
Foundation for Medical Care
952 South Mount Vernon
Colten, CA. 92324
714-825-6053
+PDP 11/34

Theresa Breckon
Real Time Systems Group
Bldg. 46A
Lawrence Berkeley Laboratory
#1 Cyclotron Road
Berkeley, Ca. 94720
+VAX, MODCOMP

Walter E. Brown
Director, Computer Center
Moravian College
Bethlehem, PA 18018
251-861-1300
[Ratfor]
+PDP 11

Robert Calland
U.S. Navy
Code 62
Naval Ocean Systems Center
San Diego, CA. 92152
619-225-2413
619-225-6231
Calldand@ISIC
+VAX-VMS

John Campbell
732 7th Street
San Pedro, CA 90731
213-831-3938
+HP-1000 w/ RTE

Layne Cannon
Pacific Northwest Laboratory
P.O. Box 999
Richland, WA 99352
509-375-2822
+VAX 11/780

Max Chandler
1030 S. Winchester Blvd.
Suite 205
San Jose, Ca. 95128
+Microcomputer 8080-Z80's

Nigel Chubb
Mitel Corp.
350 Legget Drive
Kanata, Ontario
Canada K2K 1X3
(613)592-2122
+VAX 11/780
+VAX 11-750
+PDP-11/70

Ron Church
Mitel Corp.
350 Legget Drive
Kanata, Ontario
Canada K2K 1X3
(613)592-2122
+VAX 11/780
+VAX 11-750
+PDP-11/70

Allen Cole
Univ.of Utah Computer Center
3116 MEB
Salt Lake City, UT 84112
581-8805
+UNIVAC 1100/60

Michel Cornier
Department d'Informatique
Universite de Montreal
C.P. 6128, Succ "A"
Montreal, Quebec H3C 3J7
CANADA
514-343-7382
514-343-7220
+CDC-Cyber 173

John Cowan
Kidder, Peabody and Co. Inc.
20 Exchange Place
8th Floor
New York, New York 10005
(212)635-5262
+Tandem Non-Stop II

Ben Cranston
Systems Group
Computer Science Center
University of Maryland
College Park, MD 20742
301-454-2946
+Univac

Kent Crispin
Lawrence Livermore Lab
P.O. 808
Livermore, CA
415-422-4273
+CRAY-1

Norman C. Crowfoot
Computer Services
NAU Box 15100
Flagstaff, AZ 86011
602-523-2971
+HONEYWELL DPS8

Jerry J. Deroo
Faculty of Dentistry
University of Toronto
Biometrics Section
124 Edward Street
Toronto, Ontario CANADA M5G1G6
416-978-5396
+UNIX Version 7

Ben Domenico
N.C.A.R.
P.O. Box 3000
Boulder, CO 80307
303-497-1293
303-494-5151 x559
+IBM VM/CMS

Walt Donovan
NASA/AMES Research Center
Moffett Field
Mountain View, CA 94035
Walt@bbnc or GAYDOS@BBNB
415-965-6368
+S.E.L. MPX1.4 and 2.0

William J. Donovan, Jr.
12815 S.W. 112 Terr.
Miami, FL. 33186
+IBM 370 ES

D.T. Doyle
Honeywell, Inc.
Honeywell Plaza
Minneapolis, MN 55408
MN26-2187
(612)870-2984
+VAX 11/780

Larry Dwyer
Hewlett-Packard
11000 Wolfe Road
Bldg. 430
Cupertino, CA
408-257-7000 x 2095
+HP-1000 w/ RTE

Dr. Philip H. Enslow, Jr.
School of Information
and Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30322
(404) 894-3187
+PRIME 400 and larger CPUs
+PRIME 50-series

Dr. B. Finkbein
Lehrstuhl fuer
Rechnergest Entwerfen
TU Muenchen, Post Box 202420
D-8000 Meunchen 2
GERMANY
+PRIME 400

Randolph Franklin
Rensselaer Polytechnic Inst
Troy, New York 12181
(518)270-6330
+IBM 3033-MTS
+PRIME (Rev 18)

Joe Gallagher
Director
Scientific Computing
Cleveland Clinic Foundation
9500 Euclid Avenue
Cleveland, Ohio 44106
(216) 444-2551
PDP-15, VAX

Guenther Goerz
Univ. Erlangen-Nuernberg
RRZE
Martensstr. 1
D-8520 Erlangen
West Germany
(9131)85-7031
goerz@sumex
+CDC Cyber 173

Nancy Gow
Los Alamos National Lab
P.O. Box 1663
Los Alamos, NM 87545
(505) 667-4028
+VAX 11/780
+Cyber 825
+BSD UNIX 11/70
+7600
+CRAY-1

Neil Groundwater
Analytic Disciplines
8230 Old Courthouse Road
Suite 300
Vienna, VA 22180
703-893-6140
NPG@SDAC-UNIX
+VAX-VMS
+UNIX-Ver.6

Mel Haas
Bell Laboratories
Room HO 2G-431
Homdel, NJ 07733
(201) 949-1562
+UNIX 4 and 5
+IBM MVS/TS0-3081
+IBM Amdahl Ver.8

Ralph W. Haas
1044 Calle Pecas
Thousand Oaks, CA. 91360
+DEC
+LSI-11-03
+ RT-11
+ FORTRAN IV

Dennis Hall
Computer Science and Math Dept.
Lawrence Berkeley Laboratory
Berkeley, CA. 94730
415-486-6053
Hall@LBL-Unix
+VAX

Richard M. Hambly
Harris Corp
1680 University Ave.
Rochester, New. York 14610
716-244-5830
+PDP11-70/RSX11M plus

Alan Hamilton
7 Devon Dircle
Paoli, PA 19301
(215) 296-8887
+VAX 11/750
+Burroughs 6000/7000

John Hanshew
CompuCode
6147 Aspinwatt Road
Oakland, CA 94611
415-339-9463
+Data General RDOS

David R. Hanson
Assistant Professor
Department of Computer Science
University of Arizona
Tucson, AZ 85721
602-621-6610
+DEC-10 and CDC Cyber

Steve Hathaway
Tektronix, Inc.
P.O. 1000
Del. 63-333
Wilsonville, OR 97070
P.O. Box 500
Beaverton, OR 97077
503-685-3292
+DEC TOPS-20

C. W. Holeman

1160 W. Chase Ave.
El Cajon, CA 92020
619–444–0446
+DEC RT-11

Jung P. Hong
Los Alamos National Lab
Mail Stop D455
P.O. Box 1663
Los Alamos, New Mexico 87545
(505) 667–8495
+UNIX

Ray Houghton
National Bureau of Standards
Technology Bldg., Rm A255
Washington, D.C. 20234
(301)921–3545
+VAX-VMS
+PDP-11/780
+Onyx 8002

Ray C. Houghton, Jr.
Computer Science Dept
Duke University, N.C. 27706
684–5110
houghton@nbs-vms
+VAX-VMS

Jessie C. Howell
6008 Clames Drive
Alexandria, VA 22310
Work: 703–525–6020
Home: 703–9222–7230
+Tandem & DEC IAS

Paul Howson
197 Alma Road
East St. Kilda 3182
Victoria, Austrailia
(03) 527–5881
+PRIME
+PERKIN-ELMER
+HONEYWELL

Van Jacobson
Real Time Systems Group
Bldg. 46A
Lawrence Berkeley Laboratory
#1 Cyclotron Road
Berkeley, Ca 94720
+VAX, MODCOMP

Rob Janes
Senior Scientific Analyst
Computer Applications
Mailcode 50181
Cummings Engine Co, Inc.
Columbus, IN 47201
812–372–7211
+MODCOMP IV
+Honeywell DPS8–70

Dieter Japel
Martensstr. 3
Lehrstuhl fuer Informatik 5
D-8520 Erlangen
WEST GERMANY
9131–85–7894
+PDP11/34-RSX11M
+VAX 780-VMS
+CYBER 703-NOS

Larry R. Jasper
P.O. Box 2006
Ellisville, MO 63011
(314)394–1600
+VAX/780
+PDP-11

Chris Johnson
Software Resarch Associates
P.O. Box 2432
San Francisco, CA 94126
415–957–1441
+Onyx C8002

Joint System Devel. Corp.
Yuseigojyokai — Kotohira Bldg.
1–14–1, Toranomon
Minato-ku, Tokyo
JAPAN
+VAX 11–780
+FACOM OSIV/F4
+IBM MVS/SP3
+ACOS-4
+TSOS, AOS

Richard Kiessig
Intelligent Decisions, Inc.
440 Cesano Apt. 105
Palo Alto, CA 94306
415–949–2306 (home)
+VAX
+Motorola 6800

Marty Kittower
The Software Toolworks
14478 Gloretta Drive
Sherman Oaks, CA 91423
213–986–4885
+Heath
+Kaypro 2
+Xerox 820
+Osbourne 1 and 2

Guy Lapalme
Department of Information and
Research Operations
University of Montreal
Case postale 6128
Succursale "A"
Montreal, PQ, H3C 3J7
CANADA
+CDC Cyber

Bill Lee
University of Texas
Computation Center
Austin, TX 78712
512–471–3242
lee@utexas
+Cyber 177–50

Bob Lewis
CGIS
4231 Norwalk Drive
No. EE312
San Jose, CA. 95129
(415)966–8440 x334
Home: 408–249–5986
+Apollo

John W. Lewis
General Electric Company
Corporate Research and Devel.
1 River Road, Bldg. 37, Rm 561
Schenectady, New York 12301
518–385–1600
(518)385–8247
+VAX

Pat Ma
Mitel Corp.
350 Legget Drive
Kanata, Ontario
Canada K2K 1X3
(613)592–2122
+VAX 11/780
+VAX 11–750
+PDP-11/70

Andrew Maffei
Intermetrics, Inc.
733 Concord Ave.
Cambridge, MA 02138
(617) 661–1840
+GOULD S.E.L. 32/27(MPX)
PDP-11 (RSX)

John Marshall
Mitel Corp.
350 Legget Drive
Kanata, Ontario
Canada K2K 1X3
(613)592–2122
+VAX 11/780
+VAX 11–750
+PDP-11/70

Dave Martin
Hughes Aircraft Company
MS C320
Bldg.R1
P.O. Box 92426
Los Angeles, CA. 90009
213–648–9927 after 11:30
+DEC VAX/VMS

Bill Meine
Sun Microsystems
1333 West 20th Avenue
Suite 200
Denver, CO 80234
(303)450–2324 (work)
(303)989–5442 (home)
+IBM OS/MVS

Webb Miller
Department of Computer Science
University of Arizona
Tuscon, Arizona 85721
(602)626–3685
+VAX/UNiX,
+PDP11–70 UNIX

Michael N. Norred
MINESoft
13271 W. 20th Ave.
Golden, CO 80401
(303) 238–8911
+VAX 11/780 and 11/750
+Harris 800

Ed Orton
Mitel Corp.
350 Legget Drive
Kanata, Ontario
Canada K2K 1X3
(613)592–2122
+VAX 11/780
+VAX 11–750

+PDP-11/70

Bill Patton
Armour-Dial, Inc.
P.O. 1427
Fort Madison, Iowa 52627
(319) 463–7111 x330
+General Automation 440 and 480

Robert D.Perry, Jr.
Tektronix
Information Display Division
P.O. Box 1000 MS63–296
Wilsonville, OR. 97070
(503)685–3567
+DEC-20

Chris Peterson
M/A-Com Linkabit, Inc.
10453 Roselle St.
San Diego, Ca. 92121
619–453–7007 x454
+DEC TOPS-20
Chrisp@lbl-unix]

Doug Porter
Concept Development Systems
3610 N.E. Expressway
Suite 107
Atlanta, Georgia 30340
(404) 458–0158
+Perkin Elmer

Ken Poulton
Terminal Software
3182 Greer Road
Palo Alto, CA 94303
415–856–8659
Alt: 415–857–8461
+HP-3000 w/ MPE IV (or III)

Richard C. Raffenetti
Argonne National Laboratory
9700 South Cass Ave.
Argonne, Ill. 60439
(312) 972–8497
+VAX 11–70, 11/780

Peter Reintjes
Data General Corp.
62 Alexander Drive
Research Triangle Park
NC 27709
919–549–8421
Usenet: duke!unc!pbr
+Data General AOS & AOS/VS
(Pascal)

John Saxer
Logicon, Inc.
3535 Lomita Road
Bldg. 127 Room 1438
Torrance, CA 90505
619–578–0860
ARPA: saxer@NPRDC
+Burroughs B6900 & B7800

Debbie Scherrer
C.S.A.M.
Lawrence Berkeley Laboratory
#1 Cyclotron Road
Berkeley, CA. 94720
(415)486–5881
Scherrer@LBL-UNIX
home(415)881–4489

Dr. Phillip Scherrer
Unicorn Systems
30261 Palomares Road
Castro Valley, CA. 94546
415–881–4490
Alt: 415–497–1505 (emergency)
+Digital Research CP/M

Mike Shapiro
NCR Corportation
16550 W. Bernardo Drive
San Diego, Ca. 92127
+NCR V8000 w/ VRX
+CDC Cyber w/ NOS

Sid Shapiro
Wang Institute
Tyng Road
Tyngs Boro, MA 01879
617–649–9731
+Wang VS-80

Dr. Jerome Silbert
Laboratory Service
V.A. Medical Center
West Haven, CN 06516
203–932–5711 x466
+Data General Eclipse

Dr. James A. Stark
485 — 34th Street
Oakland, CA. 94609
415–658–2566
+CompuPro

David Stoffel
11872 Dunlop Ct.
Reston, VA 22091
(703) 620–4143
+Univac

Peter J. Story
Konsberg vapenfabrikk
Postbox 25
N-3601 Kongsberg
NORWAY
(+473)73 82 50
+VAX-VMS
+NORSK Date/Sintran III

Joe Sventek
Bldg. 50B
Lawrence Berkeley Laboratory
#1 Cyclotron Road
Berkeley, CA. 94720
415–486–5205
Sventek@LBL-UNIX
+DEC RSX-11M

J. Otto Tennant
Cray Research
1440 Northland Drive
Mendota Heights, MN 55120
+CRAY-1 COS

Bob Upshaw
Real Time Systems Group
Bldg. 46A — 1123
Lawrence Berkeley Laboratory
#1 Cyclotron Road
Berkeley, CA 94720
415–486–6411
+MODCOMP w/ MAX-IV
+DEC VAX/VMS

J. Monte Waite
ESL Inc., MS 301
495 Java Drive
Sunnyvale, CA 94086
(408) 738–2888
+VAX-11/780
+PDP-11/RT-11

Stephen R. Walton
Solar Astronomy 264–33
Ca. Institute of Tech.
Pasadena, CA 91125
(213)356–3862
+VAX 11/70

Peter N. Wan
School of ICS
Georgia Tech
225 North Ave., N.W.
Atlanta, Georgia 30322
(404) 894–3658
wan.gatech@Udel-Relay
+PRIME 400
+PRIME 550

Daryl R. Winters
Sanders Assoc.
95 Canal Street
Nashua, NH 03061

Letters to the Editor

This section of the newsletter is devoted to publishing short letters intended for general dissemination among STUG members. Letters sent to STUG should include a request for publication is order to be considered for the newsletter. Long letters may be abridged by the newletter editors to fit the available space.

(The following letter was published in the Unix Connection, and forwarded to us by W. Daugherity.)

Since RSTS/E is designed to support multiple run-time systems simultaneously (e.g., BASIC-Plus, RSX, TECO and RT-11 can all be in use at one time by different users), why hasn't someone come up with a Unix run-time system for RSTS/E? I realize that some mapping of files names, etc. might have to be done, but still a healthy Unix subset should be possible under RSTS/E.

Editor's note: We have a version of the Basic Tape available for RSX, which several members have successfully implemented on RSTS/E already.

We have implemented the Software Tools on a CDC Cyber 173. After having investigated Dave Hanson's implementation for CDC's NOS operating system, we decided to implement an alternative more efficient interface to NOS which is — in contrast to his — fully compatible to the specifications of the Berkeley Cookbook. With the exception of ed, all the supplied tools are operational. Currently, we do not have a working shell; implementation of the PDS is the next goal we are heading to. If there is interest among STUG's membership, we are ready to supply the source code of the interface and the few modifications to the Tools we were forced to realize.

G. Goerz
Friedrich-Alexander-Universitat
Erlangen-Nurnberg
Germany
goerz@sumex

It seems to me that STUG is top-heavy with implementors. Having brought up about a third of the K&P tape on a Harris supermini during my spare time, I can understand the fascination (and the work!) of developing new tools and transporting the system to new computers. However, I recently moved on to a new employer and I am once again Software-Toolless. Readers, Speak up! Am I the only person who has had this problem? There are three reasons that it is difficult to start bringing the tool set up all over again.

First, hardly anyone gets free use of a computer. If I am to acquire the Tools on our IBM-whatsis, I must write a request for funds for either (a) my labor or someone else's, plus computer time, to install the Tools from the portable tape, or (b) locate and procure them ready-to-go from another location (or more likely a combination of (a) and (b)).

Second, hardly anyone knows about the Software Tools, so it's difficult to explain to a non-programming boss just what it is you're trying to do and why.

Third, not enough information is available about the package. The portable tape is cheap enough, but you need a working formatter in order to extract the manuals from the tape.

I would like to suggest a few steps to help newcomers start enjoying the benefits of the Software Tools and VOS. These steps will increase the visibility and availability of the package and bring wider appreciation of its potential to ease the pain of coding in Fortran.

(1) Sell printed copies of the user manuals *separately* from the tape. Many people would be wiling to spend a nominal amount, say $20, for the manuals describing a $50 tape full of "free" software. The reason, of course, is that everyone wants to see whether the finished product is worth the considerable amount of labor it takes to install it.

(2) Produce a professional-quality brochure describing STUG, the Software Tools, and VOS. It should give a little history, a list of utilities and other contents, perhaps a convincing sample of Ratfor and the Fortran code it generate, a list of implementations, and so on.

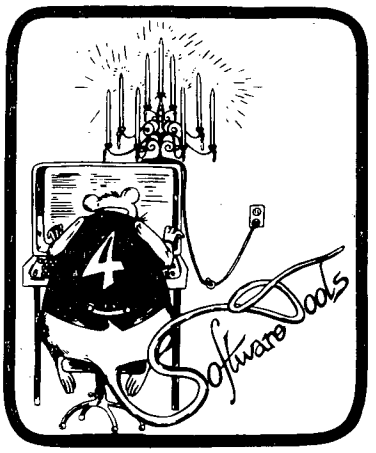
(3) Concerning specific implementations: I realize that STUG works with volunteer labor and it is hard to assure a steady supply of help for new projects. Still, wouldn't it be nice if there were *one person* to call on for help with starting the Tools on each specific brand of machine and operating system? This person would know all about VOS implementations for that computer.

(4) The Virtual Operating System has established a standard of sorts for an operating system interface. STUG should disseminate it to computer manufacturers, software houses, and entrepreneurs. These individuals might produce VOS interfaces to existing operating systems, or even special operating systems designed to support VOS. They might even make a profit at it. And why not? It would be more economical to pay someone $10,000 for a ready-to-go, fully supported, standard-conforming and efficient VOS than to bet six programmer-months on an outcome that is at best uncertain.

In closing, let me advise STUG to remember the noephytes and the suffering Fortran programmers. They have the most to gain from the Software Tools. Since we are a nonprofit organization, I believe that we have a charitable duty to help them if we can.

Charles L. Hethcoat III

Editor's note: (1) we will have printed programmers manuals and cookbooks available in early January. (2) We have an information package available for the asking which includes much of this material. (3) Volunteers?



software tools users group

1259 el camino real #242, menlo park, ca 94025

Application for Membership

Date: _____

Name: _____

Address: _____

_____

State/Country/Zip: _____

Phone: _____

Network Address: _____

Machines and systems on which you use the Software Tools package:

_____

Utilities/library functions you have implemented

_____ The standard package (as distributed by STUG)
_____ The original package (Kernighan and Plauger)
_____ Other: _____

Other systems on which you plan to implement the Tools package:

_____

Special Interests: _____

_____

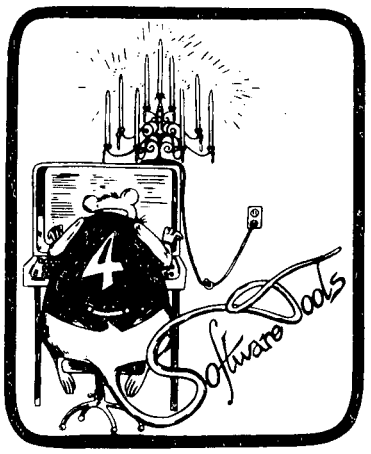
Category of Membership:

_____ Individual Membership $ 15.00
_____ Industrial Membership $ 150.00
_____ Sustaining Membership $1500.00 or more

_____ Overseas Air Mail $ 5.00
_____ Privacy; do NOT want this info. available to STUG members

Amount Enclosed: _____

Make checks payable to Software Tools Users Group, and mail to above address.



software tools users group

1259 el camino real #242, menlo park, ca 94025

Tape Order Form

Date: _____

Name: _____

Address: _____

_____

State/Country/Zip: _____

Phone: _____

Network Address: _____

Target computer(s) for the Tools: _____

Please send the following tapes:

_____ Portable LF Terminated, 2048 cpb ASCII
_____ 800 BPI _____ 1600 BPI

_____ Portable — Blocked Card Image, 3200 cpb ASCII
_____ 800 BPI _____ 1600 BPI

_____ VAX/ Unix 4.1 BSD Tools Tar Format
_____ 800 BPI _____ 1600 BPI

_____ VAX/ VMS Tools (LBL-Hughes) Files-11 Format
_____ 800 BPI _____ 1600 BPI

_____ RSX-11M Tools (LBL) BRU Format
_____ 800 BPI _____ 1600 BPI

_____ UNIVAC 1100 Tools @COPY,G Format

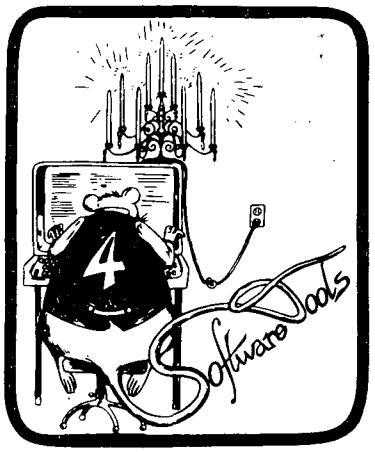
No. of tapes order: Subtotal:

_____ @ $50.00 /tape _____

Overseas Air Mail: @ $10.00 /tape _____

Total Amount Enclosed: _____

Make checks payable to Software Tools Users Group, and mail to above address.



software tools users group

1259 el camino real #242, menlo park, ca 94025

Software Submission Form

Please read the article on Tape Submission included in this newsletter. Your submission should be in archive format and include manual entries, routines, etc. as described in the article.

Machine and System on which you made the tape:

Brief description of tape contents:

Density: _____800 bpi _____1600 bp
(9-track only)

Character Code: _____ EBCDIC _____ ASCII

Blocking Factor:

Software Release
****

I (We) the undersigned give the Software Tools Users Group permission to reproduce and distribute all or any part of the program package material contained on the above tape for the use of STUG members. This material is not subject to copyright.

Submitted by: _____

_____

_____

SIGNATURE(s) _____ DATE _____

_____ _____

software tools users group

1259 el camino real #242, menlo park, ca 94025

Software Problem Report

Date:_____

Originator's Name:_____ Address:_____

Phone:_____ _____

Net Address:_____ _____

_____ _____

Name of Tool(s):_____

Machine/Operating System:_____

Date of Distribution Tape:_____

****

Problem Description: (Check all that apply) ____source ____routines ____manual entry

****

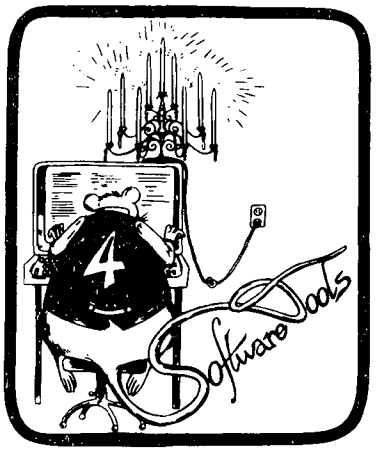
****

Suggested fix, if any:

****

If you have questions, call the
STUG Hotline

****



software tools users group

1259 el camino real #242, menlo park, ca 94025

T-Shirt Order Form

Date: _____

Name: _____

Address: _____

_____

State/Country/Zip: _____

Phone: _____

Please send me the folowing t-shirts:

Size	Color	Number
Small		
Medium		
Large		
X-Large		

No. of t-shirts ordered: _____ Subtotal:

_____ @ $10.00 /t-shirt _____

Total Amount Enclosed: _____

Make checks payable to Software Tools Users Group, and mail to above address.

UNICOM Conference Proceedings
Order Form

Proceedings of the January, 1983 UNICOM Conference are available from the Software Tools Users Group. The deadline for ordering under the conference rates was 31 March 1983. The new rate is $25 per proceeding, member or non-member. *Overseas postage is an additional $10 per copy.*

Due to the unincorporated legal status of the Software Tools Users Group, we cannot accept purchase orders.

Proceedings will be released after 1 July 1983.

Please make checks payable to the Software Tools Users Group and mail them, together with this form, to:

the Software Tools Users Group
1259 El Camino Real, #242
Menlo Park, California 94025
attn: UNICOM Proceedings

Total Number of Proceedings Ordered: _____

Total amount enclosed: $_____
(including overseas postage if applicable)

Ship to:

Name _____

Address _____

_____

City _____

State/Country _____

Zip Code _____

Lawrence Berkeley Laboratory
RTSG – 46A
University of California
Berkeley, CA 94720

PUB-402/12-83/850