Software Tools COMMUNICATIONS

NUMBER 4 OCTOBER 1980

--- MERGER (OR LACK THEREOF) WITH USENIX ---

In the previous newsletter a suggestion was made that the Software Tools group merge with the Usenix group. From the responses we received, it appeared that a merger would be acceptable if the Software Tools group could maintain its identity and independence within Usenix. The Usenix people have been quite supportive of the software tools group, especially in organizing the meetings. Nevertheless, it was felt that a merger would be premature.

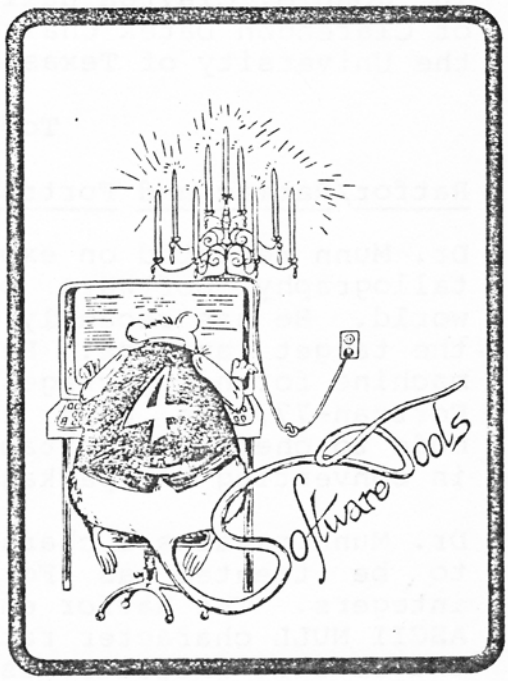
The Usenix group is an organization of installations who have entered into a contractual agreement with Western Electric to run the Unix operating system. Individuals can join the group for purposes of receiving the newsletter. They are not allowed to vote.

The Usenix Board of Directors is in the process of re-evaluating their By-Laws. In the future, the Usenix group could include individuals and organizations running not only the software tools, but other Unix look-alike systems as well. Merger at that time would seem to make more sense

--- BASIC TAPE ---

Work on the basic tape is about two thirds completed. A version of Ratfor with capabilities similar to the 7th Edition Unix Ratfor has been acquired and enhanced to include some special features necessary to allow tools from the University of Arizona, Lawrence Berkeley Laboratory, and Georgia Institute of Technology to pass through it.

The latest tools from the University of Arizona and LBL are currently at Georgia Tech, where they will be tested and merged with the GT tools. The tape should be ready for distribution later this fall.



--- NEXT SOFTWARE TOOLS MEETING ---

The next Software Tools general meeting will be held on Tuesday, January 20, 1981 in San Francisco, again in conjunction with the Usenix group. (The Usenix meeting will run January 21-23 at the same place.) Suggestions should be directed to:

George Pajari
Clarendon Datex, Ltd.
73 Water Street, 4th Floor
Vancouver, BC V6B 1A1 Canada
(604) 688-1515

--- FUTURE DIRECTIONS FORUM ---

During the Delaware conference, a group of users got together to discuss organization of the Software Tools Users Group. An informal survey is now being conducted. Please direct comments to:

Skip Egdorf
P.O. Box 735
Los Alamos, New Mexico 87544

(505) 667-5576
(FTS) 843-5576

--- MINUTES FROM DELAWARE MEETING ---

The Delaware Software Tools Meeting was held on [illegible] June 1980 in John M. Clayton Hall at the University of De[illegible]. George Pajari of Clarendon Datex chaired the session. [Thanks go to Wally Wedel of the University of Texas for these notes.]

Tools and Primitives Session

Ratfor/Ratmac and Fortran 77 -- Robert Munn, University of Maryland

Dr. Munn reported on experience gained in developing the X-ray Crystallography Software package, XTAL, and distributing it around the world. He has recently completed moving the package to Fortran-77 as the target language. He uses a DEC VAX running VMS as the development machine for his package. In the absence of a PFORT utility for Fortran-77, he uses the DEC Fortran 4 Plus processor with character mode to check for portability of Fortran 77. Several problems arose in converting his package to use Fortran 77 as the language base.

Dr. Munn retains a character macro to determine whether characters are to be treated as Fortran-77 character data type or as Fortran-77 integers. The Ratfor end-of-file definition was changed to be the ASCII NULL character rather than -1. This change was made to keep the end-of-file indicator as a member of the character set. Also the

ASCII tilde character was redefined to represent the EOS character. (As an aside, I might point out that Pascal faced a similar problem in early definitions and replaced the end characters with the notion of an end-of-line predicate. This solution might be a better way to go with Ratfor rather than stealing characters from the character set.) Also some macro symbols must be kept out of the character range. Dr. Munn chose to use the ASCII 'DEL' character as a flag to escape into other character codes. The BLOCKDATA subroutines are unused in the character version of the system. Numerous utility routines were developed to address problems encountered in this development. The most important of these appeared to be routines to manage the macro system better.

Defining the Software Tools Primitives -- Skip Egdorf, Los Alamos

Skip reported on the efforts of the Primitives SIG to define a set of Software Tools Primitives. The results of their efforts are presented in the April 1980 Software Tools Newsletter. The biggest problems people seem to be encountering with these primitives are in markl() and seek(). Work continues to further develop and define the primitives.

Evaluating the Ratfor Preprocessor -- Walter Brown, Moravian College

Dr. Brown presented a list of about 50 suggestions for Ratfor enhancements which have been gathered by the Ratfor SIG and asked people to evaluate them on a scale of A (for most desirable) to F (for least desirable). The list and results can presumably be obtained from Dr. Brown.

Ratfor-T - _ ray, Analytic Discipines, Inc.

Dave reported on the development of a system to provide predicate testing using Ratfor source code. The processor inserts a software probe at each path of Ratfor branches. Traversal of the path is recorded during test executions of the software. The idea is to ensure that all paths get executed during program testing in the hopes of catching errors as early as possible in the software development cycle. They plan to use LBL Ratfor as their standard Ratfor until a new definition emerges from the Ratfor SIG.

A Minimal Operating System Interface -- Bill Plauger, Whitesmiths

Dr. Plauger described the efforts at Whitesmiths to define a minimal set of operating system primitives to support the C language systems they are developing. The systems named were all either DEC or DEC-like operating systems or UNIX or UNIX-like operating systems. He estimated that 500 to 1000 lines of code are required to move the C system between the seven operating systems on which it runs. This code was estimated to comprise about 1/10 of the total in the system.

Implementation Issues

DEC and Software Tools -- Greg O'Brien, Digital Equipment Corp.

Greg used a very entertaining set of slides to describe the efforts within the Telco group at DEC to provide software tools. The efforts have taken the form of developing a common BLISS, and then to develop tools within that language environment. Most of the work is being done on VAX/VMS. They have the LBL tools working on VMS. They will be undertaking a project beginning in July to evaluate the LBL tools environment versus the UNIX environment to try to develop a programming environment within VAX/VMS which UNIX programmers could find suitable.

Control Data Cyber Implementation -- Wally Wedel, University of Texas

My presentation described the work that John Howard, Bill Lee and myself have done in developing a Tools environment for Control Data Cyber computer systems. Our goals in doing this work were to make a full ASCII environment available on the Cyber system, to improve interactive productivity, and to explore the usability of Fortran preprocessors. The problems faced by anyone implementing the Software Tools on a Cyber system arise first because the native mode character set is not ASCII and second because CDC operating systems are typically good batch systems on which an interactive system has been implemented. Problems with the native mode character set include a curious end-of-line convention, different graphics from ASCII, and a different collating sequence. The operating system primitives which are most noticeable by their absence include a sub-process or sub-control point facility and an interprocess communication facility.

Back to UNIX -- Neil Groundwater, Analytic Discip[illegible], Inc.

Neil described ADI's effort in moving the LBL Software Tools back onto UNIX using the University of British Columbia Fortran-77 as the target compiler. The effort was described as generally successful. A few problems with the UBC Fortran appeared which were quickly fixed.

Strategies for IBM OS Implementation -- Mike O'Dell, LBL

Mike described strategies for implementing Software Tools primitives on IBM OS systems in a TSO environment. He asserted that the easiest things to implement were those dealing with program control and status, and file naming. The more difficult primitives were those dealing with input/output, especially those dealing with file access. Mike had no software actually implemented as this design work was done at the University of Oklahoma before they obtained their UNIX system.

MIDAS -- Morris Bader, Moravian College

Dr. Bader presented a differential equation solution package written in Fortran which runs on UNIX. The package is called MIDAS. He is converting the package to Ratfor to place it in the Tools environment.

ANSI Text Processing -- Charles Card, Chairman X3J6

Dr. Card presented a brief history and progress report on the ANSI X3J6 text processing standardization project. They are in the middle of defining a standard text processing language. A standard text definition language is also under development to provide for the interchange of documents among computer systems. They expect to have a draft standard for public review and comment in the Fall of 1981. They are addressing problems of editing and of formatting text. There are some other related ANSI efforts also underway currently. Among these are the X3H3 Graphics Standard effort and the X4A12 Page Image Format standard. An international effort is also underway under the direction of ISO TC97 Technical Subcommittee 5. There is close cooperation between the ISO committee and ANSI X3J6.

Future Directions

Debbie Scherrer of LBL asked for discussion on the matter of a merger of the Tools group with the Usenix Association. Debbie had some trouble arranging this meeting since seven speakers cancelled out during the final week before the meeting. This raised questions as to whether a full day meeting was really justified. There did not seem to be much sentiment to merge the Tools group with Usenix until it becomes clear that Usenix is going to really work. Several people indicated that the speaker cancellations appeared to be more a consequence of a shaky economy than a lack of interest in the Tools work. More discussions are clearly required.

[See the items elsewhere in this issue on the subjects of the Usenix merger proposal and the forum for future directions of the Tools group. --]

--- TOOLS ON THE SEL ---

The Software Tools package has been successfully adapted to run with a SEL 32/77 under MPX 1.3. MPX is a primitive operating system, requiring maximum sizes be specified for files, and having differing file types and formats depending on file contents. The 32/77 is an inexpensive and fast 32-bit mini with an asymmetric and partial instruction set (e.g., no block transfers). Every tool, including a screen editor, but not including a shell, from the Tools distribution tape has been successfully implemented and run. "Pipes" are available and, since MPX has a rudimentary macro capability, the shell has not been missed too dreadfully.

The sources are available. Send a magtape to:

Walt Donovan
NASA Ames Research Center
Mail Stop 240-8
Moffett Field, CA 94035

and you will get the files you need in FILEMGR format. Caveat: the code will have to be changed somewhat if you use passwords with your MPX.

--- DOCUMENT SYSTEM PROPOSAL ---

Ben Domenico of NCAR in Boulder, Colorado is interested in developing tools which will be of use in a fully-integrated document production and retrieval system. (He will be working with George Pajari on the Tools Text Processing SIG.) The projects in which Ben is interested include:

1) A text formatter that produces device-independent output, indices, and includes graphics capabilities.

2) A picture language processor for producing figures for documents.

3) A set of metacode translators--one for each output device to be used.

4) An online doucmentation retrieval mechanism.

Though Ben's work is centered upon a UNIX-based system, he would also like to determine whether there exists any significant degree of interest within the Tools community for the definition of a somewhat standardized system along the lines described in the paper he presented at the June 1980 Usenix conference. [This paper will appear in the forthcoming conference proceedings.] For further discussion on this subject, you may contact Ben at:

Ben Domenico
National Center for Atmospheric Research
P. O. Box 3000
Boulder, CO 80307

--- TOOLS INTEREST WITHIN DECUS ---

A recent issue of the Canadian UNIX Users Group Newsletter reported interest within DECUS (the Digital Eqiupment Corp. Users' Group) in a Unix SIG. This would have as one of its objectives implementation of Software Tools on DEC computers not using Unix. According to the article, they are interested in the presentation of symposia papers and/or panel discussion on Tools-related subjects. For details, contact:

Mark Bartelt
Caltech 356-48
Pasadena, CA 92215
(213) 795-6811 x2663

--- TAPE DISTRIBUTION ---

Optimized versions of the Software Tools are now available for VAX/VMS and PDP 11/RSX-11M systems. A non-optimized or "portable" version is also available for those wishing to conquer new systems. These do not include the new Ratfor described in this newsletter.

RSX and VMS tapes will be written in a format appropriate for the given operating system. If you want the "portable" tape, please specify the desired blocking and density (default is 800 BPI and 512 characters per record).

To obtain a copy, send a 2400' tape with return postage to:

VICKI VAX
Software Tools
D.F.S.E.C.
U. S. Air Force Academy
USAFA, CO 80840

These tapes will be replaced by the basic tape when it becomes available.

--- NEXT NEWSLETTER ---

Articles for the next issue of the Software Tools Communications should be sent to Neil Groundwater and Gary Trujillo at:

Analytic Disciplines, Inc.
[illegible] Courthouse Road, Suite 300
Vienna, Virginia 22180
(703) 893-5140

--- The 'STANDARD' RATFOR PREPROCESSOR ---

Beginning on the following page is the user document for the Ratfor preprocessor which will be provided on the basic tape. This Ratfor was orginally developed at the University of Arizona, with enhancements by LBL. Before final distribution, Georgia Tech will add long variable names, which would be converted by the preprocessor to 6-character unique Fortran-acceptible names. This feature is necessary for processing the tools Georgia Tech has donated for the basic tape.

Ratfor User Documentation

NAME

Ratfor - Ratfor preprocessor

SYNOPSIS

Ratfor [files ...] >outfile

DESCRIPTION

Ratfor translates the Ratfor programs in the named files into Fortran. If no input files are given, or the filename '-' appears, the standard input will be read.

A file containing general purpose software tools definitions (e.g. EOF, NEWLINE, EOS, etc.) will be automatically opened and processed before any of the files specified are read.

Syntax:

Ratfor has the following syntax:

```
prog:   stmt
        prog stmt
stmt:   if (expr) stmt
        if (expr) stmt else stmt
        while (expr) stmt
        repeat (expr) stmt
        repeat stmt until (expr)
        for (init expr; test expr; incr expr) stmt
        do expr stmt
        do n expr stmt
        break
        break n
        next
        next n
        return (expr)
        switch (expr) {
        case expr:   stmt
          ....
        default:     stmt
        }
        digits stmt
        { prog }  or  [ prog ]
        anything unrecognizable (i.e. fortran)
```

where 'stmt' is any Fortran or Ratfor statement. A statement is terminated by an end-of-line or a semicolon.

Character Translation:

The following character translations are performed:

```
<                .lt.
<=               .le.
==               .eq.
!=, ^=, ~=       .ne.
>=               .ge.
>                .gt.
|                .or.
&                .and.
!, ^, ~          .not.
```

Included files:

The statement

```
include file        or
include "file"
```

will insert the contents of the specified file into the Ratfor input in place of the 'include' statement. Quotes must surround the file name if it contains characters other than alphanumerics, underscores, or dots.

Macro Definitions:

The statement

```
define(name, replacement text)
```

defines 'name' as a macro which will be replaced with the indicated text when encountered in the source files. Any occurrences of the strings '$n' in the replacement text, where 1 <= n <= 9, will be replaced with the nth argument when the macro is actually invoked. For example:

```
define(bump, $1 = $1 + 1)
```

will cause the source line

```
bump(i)
```

to be expanded into

```
i = i + 1
```

The names of macros may contain letters, digits, periods and underline characters, but must start with a letter. Upper case is not equivalent to lower case in macro names.

The replacement text is copied directly into the lookup table with no intepretation of the arguments, which differs from the procedure used in the macro utility. This "deferred evaluation" has the effect of eliminating the need for bracketing strings to get them through the macro processor unchanged. A side effect of the deferred evaluation is that defined names cannot be forced through the processor - i.e. the string "define" will never be output from the preprocessor. The inequivalence of upper and lower case in macro names may be used in this case to force the name of a user defined macro onto the output - i.e. if the user has defined a macro named mymac, the replacement text may contain the string MYMAC, which is not defined, and will pass through the processor.

(For compatibility, an "mdefine" macro call has been included which interprets definitions before stacking them, as does the macro tool. When using this version, use "$(" and "$)" to indicate deferred evaluation, rather than the "[" and "]" used by the macro tool.)

In addition to define, four other built-in macros are provided:

arith(x,op,y)	performs the "integer" arithmetic specified by op (+,-,*,/) on the two numeric operands and returns the result as its replacement.
incr(x)	converts the string x to a number, adds one to it, and returns the value as its replacement (as a character string).
ifelse(a,b,c,d)	compares a and b as char strings; if they are the same, c is pushed back onto the input, else d is pushed back.
substr(s,m,n)	produces the substring of s which starts at position m (with origin one), of length n. If n is omitted or too big, the rest of the string is used, while if m is out of range the result is a null string.

Note: the statement

```
define name text
```

may also be used, but will not always perform correctly for macros with parameters or multi-line replacement text. The functional form is preferred.

Conditional Preprocessing:

The statements

```
ifdef(macro,text)       and
ifnotdef(macro,text)
```

conditionalize the preprocessing upon whether the macro has been previously defined or not.

String Data Types:

The statements

```
string name "character string"
```

or

```
string name(size) "character string"
```

declare 'name' to be a character array long enough to accomodate the ascii codes for the given character string, one per array element. The array is then filled by data statements. The last word of 'name' is initialized to the symbolic parameter EOS, and indicates the end of a string. EOS must be defined either in the standard definitions file or by the user. If a size is given, name is declared to be a character array of 'size' elements. If several string declarations appear consecutively, the generated declarations for the arrays will precede the data statements that initialize them.

String [illegible]ls:

Conversion of in-line quoted strings to Hollerith constants is performed in the following manner:

```
"str"       nHstr
        (where 'n' is the number of characters in str)
```

String literals can be continued across line boundaries by ending the line to be continued with an underline. The underline is not included as part of the literal. Leading blanks and tabs on the next line are ignored.

Integer Constants:

Integer constants in bases other than decimal may be specified as n%dddd... where 'n' is a decimal number indicating the base and 'dddd...' are digits in that base. For bases > 10, letters are used for digits above 9. Examples include: 8%77 (=63), 16%2ff (=767), 2%0010011 (=19). The number is converted to the equivalent decimal value using multiplication; this may cause

sign problems if the number has too many digits.

Lines and Continuation:

Input is free-format; that is, statements may appear anywhere on a line, and the end of the line is generally considered the end of the statement. However, lines ending in special characters such as comma, +, -, and * are assumed to be continued on the next line. An exception to this rule is within a condition; the line is assumed to be continued if the condition does not fit on one line. Explicit continuation is indicated by ending a line with an underline character (_). The underline character is not copied to the output file.

Comments:

Comments are preceded by '#' signs and may appear anywhere in the code.

Literal (unprocessed) Lines:

Lines can be passed through Ratfor without being processed by putting a percent "%" as the first character on the line. The percent will be removed and the line shifted one position to the left, but otherwise will be output without change.

CHANGES

This Ratfor preprocessor differs from the original (as released by Kernighan and Plauger) in the following ways:

The code has been rewritten and reorganize clarity.

A hash table has been added for increased efficiency in searching the definitions list.

The 'string' data type has been included.

The 'define' processor has been augmented to support macros with arguments.

Preprocessing conditional upon the definition (or lack therof) of a symbol has been included.

Extraneous gotos in the resulting Fortran have been reduced.

Blanks have been included in the output for increased readability.

Multi-level 'break' and 'next' statements have been included.

The Fortran 'DO' is allowed, as well as the Ratfor one.

The capability of specifying integer constants in bases other than decimal has been added.

Underscores and dots have been allowed in defined names.

The 'define' syntax has been expanded to include the form:
define name value

The 'return(value)' feature has been added.

Quoted file names following 'include' statements have been added to allow for special characters in file names.

A method for allowing lines to pass through un-processed has been added.

The 'switch' control statement has been included.

Continuation lines have been implemented.

Brackets have been allowed to replace braces (but NOT $(and $)).

The bug causing incorrect line numbers to be printed in error messages has been fixed.

FILES

A general definitions file (e.g. 'ratdef') is automatically opened and read.

SEE ALSO

Ker... Plauger's "Software Tools"
Kernigha... Ratfor - A Preprocessor for a Rational Fortran"
The Unix command 'rc' in the Unix Manual
The tools 'incl' and 'macro'

DIAGNOSTICS

<The actual documentation contains a complete list of error messages and their meaning>

AUTHORS

Original by B. Kernighan and P. J. Plauger, with rewrites and enhancements by David Hanson and friends (U. of Arizona), Joe Sventek and Debbie Scherrer (Lawrence Berkeley Laboratory).

BUGS/DEFICIENCIES

The macro capabilities in this Ratfor are in some ways incompatible with the macro processor tool.

Missing parentheses or braces may cause erratic behavior. Eventually, Ratfor should be taught to terminate parentheses/brace checking at the end of each subroutine.

Extraneous 'continue' statements are generated within Fortran

'do' statements.

There is no way to explicitly cause a statement to begin in column 6 (i.e. a Fortran continued statement), although implicit continuation is performed.

See you in San Francisco!

This work was supported by the Applied Mathematical Sciences Research Program of the Office of Energy Research, Department of Energy under Conract W-7405-ENG-48.

PUB-348 1200/9-80