* * * * * * * * * * * *

Dreamboat

* * * * * * * * * * * *

TEX AS A PROGRAMMING LANGUAGE?

A. E. Siegman
Professor of Electrical Engineering and
Director of the Edward L. Ginzton Laboratory, Stanford University

Many of the future users of TEX will not be computer professionals. The majority of future users, in fact, will be "knowledge workers"— scientists and engineers, editors and writers, humanities scholars, university faculty members, secretaries, and even business professionals.

Many of these users will want to do all their text composition, communication, document preparations, and (very important) their record-keeping using only a local screen editor, an electronic mail system, and TEX— nothing else. One reason for keeping this list of computer tools as short as possible is that the working user of a computer system — in contrast to a computer professional — usually wishes to learn as few systems or languages as possible. This also minimizes the number of file formats to be concerned with, and the number of programs that must be provided on the system.

An important implication of this is that it would be very desirable to develop a TEX *programming language*, separate and distinct from the TEX *typesetting program*. That is, one would like to have a programming language which would retain essentially the same syntax and the same programming capabilities as TEX (plus additional capabilities), but which would work only with ASCII characters and produce only ASCII output. What's important of course is not so much the ASCII aspect, but retaining (and even supplementing) most of the programming language and macro capabilities of TEX, while eliminating everything related to the more sophisticated typesetting and printing aspects.

As one illustration of what this implies, consider a TEX user who keeps bibliographies, address lists, and other sets of records with each record stored in a format like

```
\author{J. Jones}
\title{History of TeX}
\date{December 1982}
```

By defining suitable macros for \title, \author and so forth, it is possible at present not only to introduce these records into documents to be typeset, but also to use TEX to manipulate, rearrange, and reformat these records in a variety of ways, i.e. to use TEX as a programming and record-manipulating language.

Suppose however that the user only needs to manipulate these records, and then output a reordered subset of each record in ASCII lineprinter format, not as typeset output. Reasons for doing this might be to get output on a local hardcopy terminal; or perhaps not printed output at all, just reformatted output to be sent to another file or electronically transferred to a non-TEX-using colleague. This can be done at present by using \send to transfer the formatted output to another file and discarding the pages that TEX produces; but this solution is inelegant, incomplete, and expensive.

A supplementary TEX-like language, which did not require learning an entire new syntax, and which might offer added arithmetic and logical capabilities, would be extremely useful in this situation. It's not necessary here to explore just which capabilities should be retained or added and which should be discarded in such a TEX programming language. Clearly the line-breaking and page-making capabilities of TEX might go; the entire concept of glue might be eliminated; and so forth. A side benefit is that a simplified TEX programming language might be a much less expensive program to run. (This may not seem important to computer professionals in working environments where extensive computer resources are readily available. In environments where users must pay for CPU cycles on a fully costed basis, however, TEX is in fact an expensive program to run).

All this is obviously not something that has much connection with Don Knuth's original objectives in writing TEX. If however TEX is to become a widely used language — as I believe it is — for people who work with words and ideas, then the concept that TEX and its derivatives should meet essentially all the needs for those people becomes important. The importance of minimizing the number of languages that a lay computer user has to learn should not be minimized. This note is to suggest to the Tugboat community that an offshoot of TEX, which uses the same notation and syntax to manipulate strings, numbers, and files, without doing any typesetting at all, is well worth thinking about.