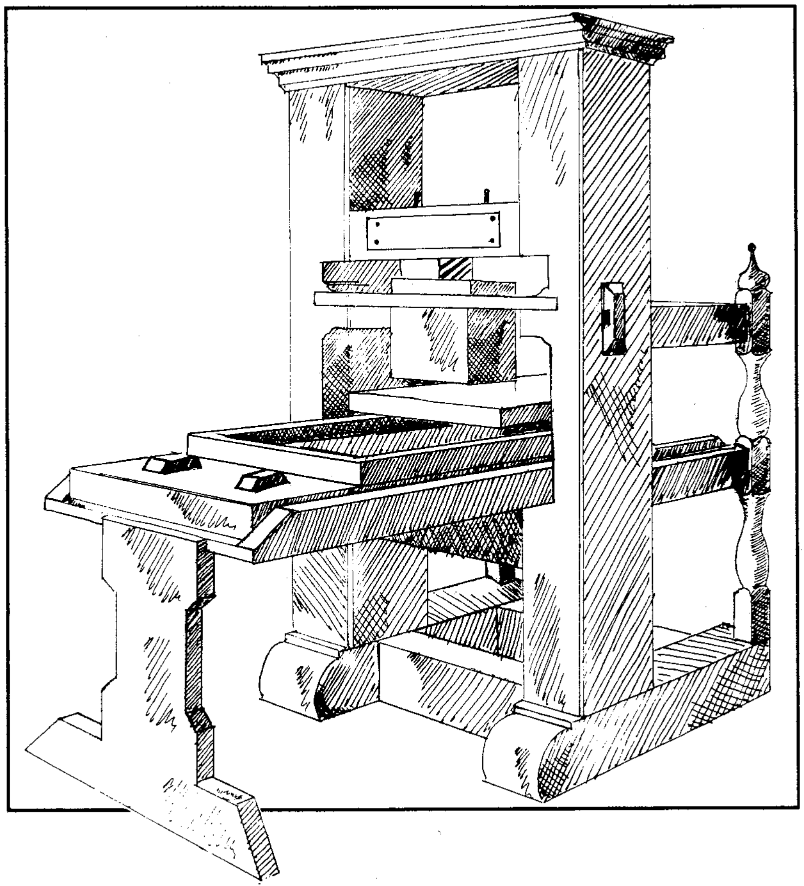
TUGboat

The Communications of the TEX Users Group



Volume 12, Number 1, March 1991
TEX90 Conference Proceedings

TEX Users Group

Memberships and Subscriptions

TUGboat (ISSN 0896-3207) is published four times a year plus one supplement by the TEX Users Group, 653 North Main Street, P. O. Box 9506, Providence, RI 02940, U.S.A.

1991 dues for individual members are as follows:

- Ordinary members: $45
- Students: $35

Membership in the TEX Users Group is for the calendar year, and includes all issues of *TUGboat* for the year in which membership begins or is renewed. A membership form is provided on page 183.

Subscription rates: United States $45 a year; all other countries, $45.

Second-class postage paid at Providence, RI, and additional mailing offices. Postmaster: Send address changes to the TEX Users Group, P. O. Box 9506, Providence, RI 02940, U.S.A.

Institutional Membership

Institutional Membership is a means of showing continuing interest in and support for both TEX and the TEX Users Group.

The privileges of institutional membership include:

- designating individuals to receive *TUGboat* subscriptions and be accorded the status of individual TUG members, the number of such designees depending on the category of membership chosen;
- reduced rates for the purchase of additional subscriptions;
- reduced rates for TUG meetings/courses for *all* staff members, and for rental/purchase of videotapes.
- acknowledgement in every issue of *TUGboat* published during the membership year.

For further information, contact the TUG office.

Addresses

General correspondence:
TEX Users Group
P. O. Box 9506
Providence, RI 02940

Payments:
TEX Users Group
P. O. Box 594
Providence, RI 02901

Parcel post,
delivery services:
TEX Users Group
653 North Main Street
Providence, RI 02904

Telephone

401-751-7760

Fax

401-751-1071

Electronic Mail

(Internet)
General correspondence:
`TUG@Math.AMS.com`
Submissions to *TUGboat*:
`TUGboat@Math.AMS.com`

TeX90 Conference Proceedings
University College
Cork, Ireland, September 10–13, 1990

TUGBOAT

COMMUNICATIONS OF THE TeX USERS GROUP
TUGBOAT EDITOR BARBARA BEETON
PROCEEDINGS EDITOR MARY GUENTHER

VOLUME 12, NUMBER 1 • MARCH 1991
PROVIDENCE • RHODE ISLAND • U.S.A.

Production Notes

The TEX source code for most articles in this issue of *TUGboat* were transmitted via network or floppy diskette to the editors, who then generated `dvi` files with TEX 3.0 on their IBM 3090 for printing on an IBM 3827 or an HP LaserJet. These edited articles were subsequently transmitted to the TUG office via tape and run on an IBM PC-compatible 386 with PC TEX and μTEX. The resulting `dvi` files were shipped to the American Mathematical Society via modem, and camera copy was produced on the Society's APS μ-5.

A few items and articles were produced locally by authors and then mailed to the editors for incorporation in these *Proceedings*. Camera-ready copy for the entire articles was supplied by Les Carr, Olivier Nicole, and Friedhelm Sowa.

Several articles required special fonts from METAFONT. Fonts for the articles by Alan Hoenig, Yannis Haralambous, and Dean Guenther and Janene Winter were generated using pcMF except for Haralambous' initials. These were created using MacMETAFONT on a Macintosh at the American Mathematical Society. Special fonts for Adrian Clark's and Mícheál Ó Searcóid's articles were produced locally and these authors supplied camera-ready copy which was cut in.

Roughly one-third of the articles were written using LATEX macros; the remainder used the `plain`-based `tugproc.sty`.

Trademarks

Many trademarked names appear in the pages of *TUGboat*. If there is any question about whether a name is or is not a trademark, prudence dictates that it should be treated as if it is. The following list of trademarks which appear in this issue may not be complete.

APS μ5 is a trademark of Autologic, Inc.

DOS and MS/DOS are trademarks of MicroSoft Corporation

LaserJet, PCL, and DeskJet are trademarks of Hewlett-Packard, Inc.

METAFONT is a trademark of Addison-Wesley Inc.

PC TEX is a registered trademark of Personal TEX, Inc.

PostScript is a trademark of Adobe Systems, Inc.

TEX and AMS-TEX are trademarks of the American Mathematical Society.

UNIX is a trademark of AT&T Bell Laboratories.

Other Conference Proceedings

Europe

Proceedings of the First European Conference on TEX for Scientific Documentation. Dario Lucarella, ed. Reading, Mass.: Addison-Wesley, 1985. [16–17 May 1985, Como, Italy.]

Proceedings of the Second European Conference on TEX for Scientific Documentation. Jacques Désarménien, ed. Berlin: Springer-Verlag, 1986. [19–21 June 1986, Strasbourg, France.]

TEX88 Conference Proceedings. Malcolm Clark, ed. Chichester, England: Ellis Horwood, 1990. [18–20 July 1988, Exeter University, Exeter, England.]

North America

Conference Proceedings: TEX Users Group Eighth Annual Meeting. Dean Guenther, ed. *TEXniques* No. 5. Providence, Rhode Island: TEX Users Group, 1988. [24–26 August 1987, University of Washington, Seattle, Washington.]

Conference Proceedings: TEX Users Group Ninth Annual Meeting. Christina Thiele, ed. *TEXniques* No. 7. Providence, Rhode Island: TEX Users Group, 1988. [22–24 August 1988, McGill University, Montréal, Canada.]

Conference Proceedings: TEX Users Group Tenth Annual Meeting. Christina Thiele, ed. *TUGboat* 10, no. 4. Providence, Rhode Island: TEX Users Group, 1989. [20–23 August 1989, Stanford University, Stanford, California.]

Conference Proceedings: TEX Users Group Eleventh Annual Meeting. Lincoln Durst, ed. *TUGboat* 11, no. 3. Providence, Rhode Island: TEX Users Group, 1990. [18–20 June 1990, Texas A&M University, College Station, Texas.]

Introduction

Barbara Beeton

TEX90, meeting in Cork from September 10–13, 1990, was the fifth EuroTEX conference, and the first with which the TEX Users Group was associated as a sponsor. As were the previous meetings in Europe, TEX90 can be counted a success in both the diversity and quality of the papers presented.

The conference facilities were located at University College, Cork (Coláiste na hOllscoile, Corcaigh, to the locals). Opened in 1849 as the "Queen's University", UCC was incorporated as a constituent college of the National University of Ireland in 1908. Long before that, however, Cork was known as a center of learning, and monks gathered there from the 12th century onward (and from Europe and the New World, as a rather confused tour guide informed us). Today, UCC has over 6,500 full-time students and more than 1,000 staff. Among the more famous faculty members was George Boole, without whom the craft of computing would probably be quite different.

A current UCC staff member of some renown in the TEX community is Peter Flynn, the local member of the TEX90 Program Committee; to him we owe appreciation for the very fine local arrangements. Our praise for putting together an excellent program go to him and to the other committee members — Dean Guenther (co-coordinator with Peter Flynn), Peter Abbott, Johannes Braams, Malcolm Clark, Bernard Gaulle, Roswitha Graham, and Joachim Lammarsch. And for additional organizational efforts, we thank Kees van der Laan and Ray Goucher.

The official program opened Monday morning with a lively introduction by Michael Smith, Ireland's Minister for Science & Technology. Not only had he traveled quite a distance to give this talk, but he had also done his homework, and was able to describe how TEX fits in well with the nation's plans for building a strong technological base to attract international commerce. The President of UCC followed with some remarks, including an appreciation for a recent budget increase and other support from the Ministry, and a short history of UCC and George Boole.

After introductions of the attending officers of TUG and the European TEX user associations, the technical program began with a session on databases and hypertext. This appears to be a very fruitful area for research, and one with great promise for TEX applications in specific areas. (Every time I see a demonstration of a multidocument system with hypertext links between them, I want one for organizing myself.) The first afternoon was filled out with presentations on efforts in the Netherlands and the U.K. to build facilities adapted to their local needs. Working Group 13 of NTG (the Nederlandstalige TEX Gebruikersgroep) has been working on building LATEX styles that are more acceptable than the originals are in Europe. Adrian Clark described a prototype system for documenting the Aston archive's holdings for users. These needs may be local, but the results will be much more widely useful.

On Tuesday, the first topic addressed was the æsthetics and practicalities of typographics. This included a discussion by Victor Eijkhout of why a document style designer should not work directly in (LA)TEX, but should be provided with separate tools that express the variables of formatting in a more natural way. The next session presented suggestions for including graphics in TEX, or, in the case of Tim Murphy, turning the process inside out and including TEX in graphics. Just before lunch, we were treated to the next installment of the ongoing saga "Towards LATEX 3.0", by Rainer Schöpf. The last presentation of the day introduced an integrated, documented, public domain TEX system for VAX/VMS. The attendees then proceeded to coaches waiting to take them for a trip to Blarney Castle, dinner, and entertainment by a group of Irish balladeers and a troupe of young folk dancers.

(As an erstwhile folk dancer, I can vouch for their talent and skill, and appreciated that they seemed to be enjoying themselves rather than regarding their performance as just a job to be done.)

Wednesday began with Malcolm Clark recounting all the things that can go wrong with editing and publishing the Proceedings of a TEX conference; how familiar! A short introduction to SGML (the Standard Generalized Markup Language) was followed by birds-of-a-feather sessions, and then by two presentations on graphics and TEX. Alan Hoenig built line drawings in METAFONT and devised a technique for passing back to TEX the information needed to apply labels in the proper locations using the regular fonts; this is a good example of one more use that I suspect Don Knuth never envisioned. After lunch, some more traditional, in fact, *very* traditional uses of METAFONT were described, with Yannis Haralambous presenting old German typefaces (and being acclaimed as the presenter of the best paper at the conference), and Mícheál Ó Searcóid telling the history of the written Irish language and showing his work with the traditional Irish alphabet. The font session was rounded out by Dean Guenther's paper on the International Phonetic Alphabet. After tea and a "guru session", Adrian Clark ended the day speaking on halftone output from TEX.

Thursday began with a description by Nico Poppelier of how SGML and TEX (and several other composition systems) coexist symbiotically in the production system of a commercial publisher; this, I think, is an eminently sensible arrangement, and one that makes the best of the strengths of each approach. The next two papers were more technical, with Amy Hendricksen delving into the nitty-gritty of macro techniques, and Frank Mittelbach offering a shopping list of proposals for extending BIBTEX. Two papers directed toward capturing and training new users gave us food for thought. How *not* to write a manual was the topic of Angela Barden's paper, and Konrad Neuwirth, himself a secondary school student (albeit an atypical one), proposed that although TEX is a useful tool, it doesn't teach principles that are appropriate for secondary school students. The final session, after lunch, concentrated on the aesthetics of documents, with one paper a practical course in how to produce a book with fonts other than Computer Modern, the second a more theoretical review of possible page shapes, and the third giving another approach to integrating graphics with TEX.

The technical program was complemented by various social events, support for which was provided by a wide variety of sponsors: Aer Lingus, Apple Computer Sales (Cork), ArborText, Blue Sky Research, Cepadues, the Cork Examiner, Cork-Kerry Tourism, the Cork Taxi Co-op, Guy & Co., Horizon Applecentre, MID/ILG Heidelberg, Northlake Software, Personal TEX, Ronnie Moore Ltd., TEXcel, TEXworks, and UniTEX (Sheffield). Thanks to all of them.

I thoroughly enjoyed my stay in Cork, and found it a most worthwhile adventure. Those of us from the U.S., where TEX began, have both a good example, and a tough act to follow, from this most active European TEX community. TEX is alive and flourishing in this environment, and I look forward to my next visit to a EuroTEX conference.

As a sponsor, TUG has the honor of publishing the *Proceedings* of the TEX90 conference. We hope you agree that this is a worthy volume, deserving of your attention and interest.

Finally, I wish to thank Mary Guenther, the *Proceedings* editor, who has made it possible for me to sit back and simply enjoy this issue.

◇ Barbara Beeton
American Mathematical Society
P. O. Box 6248
Providence, RI 02940
Internet: `BNB@Math.AMS.com`

Quick and Dirty Databases with Nice Output: AWK and TEX

Erich Neuwirth
Institute for Statistics and Computer Science, University of Vienna
Universitätsstraße 5-9, A-1010 Vienna, Austria
Bitnet: a4422dab@awiuni11

Abstract

This paper will describe an easy-to-use set of tools which enables moderately to well experienced TEX-users to produce formatted output from flat data files. Usually all you need is AWK, TEX, and a sort program.

A First Example

Let us assume we have a text file containing addresses in more or less random order. Let us further assume that the main purpose of the file is to produce a printed address list and directory, possible in different versions sorted according to different criteria. Let us further assume that the data file has a very simple structure, namely a number of lines for each address and different addresses being separated by blank lines. Additionally, we assume that the items to be used in sorting occur on the same line in every address block, e.g. the surname is always on the first line, the city on the third line, and so on.

Now we would like to format this material into a two-column format for an address book to be typeset by LATEX. For the first step, let us assume that the data already are sorted in the sequence we need for the printed version. Then the following little AWK program does the trick:

```
BEGIN   {
        RS=""
        FS="\n"
        printf "\\documentstyle[twocolumn]"
        printf "{article}\n"
        printf "\\begin{document}\n\n"
        }
        {
        for(i=1;i<=NF-1;i++)
            printf "%s  \\\\ \n",  $i
        printf "%s \n\n", $NF
        }

END     {
        print ""
        printf "\% %d records transferred \n\n"
            ,NR
        print "\\end{document}"
        }
```

If you keep the program in a file address.awk and execute the following command,

```
awk -f address.awk address.dat>address.tex
```

then the file address.tex will contain a LATEX file which prints the address book.

The command line will work with a UNIX or an MS-DOS system, if you have AWK. We will talk about the availability of AWK later.

Let us look at this program. It consists of three parts.

The first part, starting with BEGIN, is executed before any data is read from the input file. So, this is the place to put the LATEX preamble and any definitions you need for page layouts, (like page sizes, \parskip, etc.).

The middle part of our program, the one with the for loop, is executed for each record read in from the data file. This is where we can reformat the unformatted records from our primary data file.

The third part of the program, beginning with END, is executed after all input data was read, so this is the place for any final housekeeping activities and for \end{document}.

Here we are using the fact that AWK has an implicit loop running through all records in the data file. It also has easy-to-use tools for structuring the records into fields. "$i" in our program refers to the ith field of each record, so AWK by itself breaks the records into fields which are exactly the units we want to deal with for the TEX output.

In this program we use print and printf the same way it is used in C; the print command uses default printing formats; printf explicitly needs a formatting string.

The program by itself should be understandable to anybody with a little knowledge of C. The only additional knowledge is about the implicit loop over records.

If we want to inhibit page or column breaks for each entry in our address book, we simply add the AWK commands to make a "minipage" for each entry, so the middle part should look like this:

```
{
print "\\begin{minipage}{\\textwidth}"
for(i=1;i<=NF-1;i++)
    printf "%s  \\\\ \n",  $i
printf "%s \n", $NF
print "\\end{minipage}"
printf "\n"
}
```

The double backslashes are needed because in AWK, like in many C-like languages, the backslash is the escape character, so a double backslash is needed to produce a single backslash in the output file. If we want the first entry to be printed in boldface, only a slight change in our program is needed:

```
{
print "\\begin{minipage}{\\textwidth}"
printf "{\\bf %s }", $1
for(i=2;i<=NF-1;i++)
    printf "%s  \\\\ \n",  $i
printf "%s \n", $NF
print "\\end{minipage}"
printf "\n"
}
```

Of course this model can be generalized; we can extract any field from the record and add special formatting commands to the field. Using this little program as a skeleton, we already are able to reformat a simple text file with addresses to get a booklet with typeset quality.

Extending the Example

Besides the implicit loop, AWK has a strong pattern matcher directly built into the language. It uses the regular expression syntax of UNIX (like grep and other text-oriented tools in UNIX) and allows selection of records according to certain patterns. If, in our example, we wanted to print the address book for the inhabitants of Cork only, we would only have to change the middle part of the program very slightly:

```
/Cork/  {
        for(i=1;i<=NF-1;i++)
            printf "%s  \\\\ \n",  $i
        printf "%s \n\n", $NF
        }
```

So we only added the /Cork/ expression which tells AWK to perform the printing actions only to records containing the string Cork.

In this case, the command printing the number of records into the TEX file would print the number of records in the input file and not the number of addresses formatted. To correct this we have to modify our program again; we simply add a variable which counts the number or records transferred to the output file,

```
/Cork/  {
        nofout++
        for(i=1;i<=NF-1;i++)
            printf "%s  \\\\ \n",  $i
        printf "%s \n\n", $NF
        }
```

and we change the print command for NR in the end part of our program to:

```
printf "\% %d records transferred \n\n"
        ,nofout
```

AWK initializes variables with zero, so the counting works correctly. We will not explain the pattern-matching mechanism of AWK in this paper, that would go beyond both the scope and the space limits of the paper. We only wanted to demonstrate that selection of records is already built into AWK, so we do not need a special tool for that.

Sorting the Data

AWK by itself does not offer built-in capabilities for sorting data. It would be possible to write a sorting program in AWK, but since AWK is interpreted, performance would be low. It also would be a waste of energy for the programmer, since any reasonable operating system has a sort utility. The only task we have to perform is to write a small AWK program which reformats the input file in such a way that the sort program can sort it according to our criteria. We might have to write another small program which transforms the sorted data back into the original form. We need a little bit of additional information. In the example program in the beginning, we had the following code:

```
RS=""
FS="\n"
```

The variable RS defines the record separator; the variable FS defines the field separator. These two separators refer to the input file. The corresponding two variables, ORS and OFS, refer to the output file. Using these variables, we can transform multiline records into oneline records which can be better dealt with by sort programs. Consider the following program:

```
BEGIN   {
        RS=""
        FS="\n"
        ORS="\n"
        OFS="\t"
        }
```

```
{
printf "%10s  ", $3
print $1,$2,$3,$4,$5
}
```

The BEGIN part defines the record and field separators, the main loop writes the third variable to a fixed length field at the beginning of each output record and then writes all variables with the new output separator (a tab character) into the output file. So the output file contains one line per record. Now we simply use a sort program to sort the intermediate file into the order we need. Then we have to write a similar small AWK program which cuts off the first field of our new records and copies the remaining fields restoring the old multiline structure by using,

```
ORS="\n\n"
OFS="\n"
```

(ORS="" does *not* work). We now have the sorted data in the original format and can print the address book in the order we need.

General Considerations

The examples in the previous sections show that it is very easy to use AWK as a front end between data files and TEX. It is very easy to write short AWK programs which perform the simple formatting tasks needed for preparing data for TEX output. We even might state that we have a tool modeled after WEAVE, which is used in preparing TEX itself and which transforms "raw" input in the form of program parts and comments into a format which allows TEX to typeset itself and other programs with TEX typographical standard. In our case, we use AWK to transform "raw" data into TEX input.

We do not have the full power of database management systems, but if the data file only functions as a central repository of the data and the main final output is printed material, it is worthwhile to consider AWK as a solution.

Availability of AWK

AWK is part of UNIX in all its varieties, so it is widely available already.

The Free Software Foundation and its GNU project have produced a version called GAWK which is completely free and available on many file repositories on electronic networks. It also has been ported to many different operating systems, and it is available for MS-DOS.

The literature needed to handle projects like the ones described is: TEX related books, which the author is not going to cite just to be different from almost any other paper in these proceedings, and *The AWK Programming Language.*

Bibliography

Aho, A., Kernighan, B., and Weinberger, P. *The AWK Programming Language*, Addison-Wesley 1988.

TeX & Hypertext — The Future of Electronic Publishing?

Christine Detig
Kranichweg 1, D-6074 Rödermark, FR Germany
Bitnet: `xiticdet@ddathd21`

Abstract

The usage of computers allows writing and publishing documents electronically. This article is oriented towards the support of authors.

Hypertext is a popular method of electronic document manipulation. It supports authoring by representing a document as a net of linked text fragments in which the structure can be visualized and manipulated. Publishing a hypertext means no longer printing. Instead, the document remains in the computer and is read electronically.

Can hypertext concepts be used for traditional electronic publishing, i.e., for the production of printed documents? And is it possible to integrate them in an environment that uses typesetting systems like TeX for the electronic authoring and for the printing? These questions are discussed.

Introduction

Talking about the rôle of TeX in the process of electronic publishing should reflect that TeX is only a tool for solving part of the problems that arise. We should not enlarge it to handle everything, but instead, integrate it in a set of adequate tools. This article will reflect on the usability of TeX in such a publication environment and under what circumstances it can be used. It shall be a contribution to the development of TeX towards its real power. Otherwise, TeX will be buried under all the other "easy-to-use" text processors.

Electronic publishing, the interwoven process of editing, viewing and revising that results in a printed version of the document, can be separated into electronic authoring and electronic printing. The preparation of a certain group of documents demands more from both parts than an editor and a formatting system. These documents can be classified as being large, long-living and determined to be under regular revision. They may exist in different versions and in several variations. It may also be that several authors work on them. Examples are manuals, lexica, tutorials, etc. They are best handled with a special authoring system that supports long-time management and flexible manipulation.

There exists a technique of document preparation, *hypertext*, that offers support for the interactive authoring of structured documents. In the following, a short overview about hypertext and some typical applications will be given. For more details, see Conklin [1987] or Meyrowitz et al. [1985]. Following the overview, the usability of the hypertext technique in an authoring environment for printed documents in connection with TeX shall be discussed.

What is Hypertext?

In addition to the "classic" electronic typesetting methods, there are other ways to use a computer for electronic document preparation. One of them, quite popular now, is hypertext. Hypertext is a technique that supports electronic authoring by offering an interactive environment that allows one to visualize and manipulate the structure of a document, which is built up from text *fragments* that are connected by *links*. Publishing a hypertext means no longer using printed paper, but only providing on-line access: The document remains in the computer and is read electronically, usually via a network. For visualization, the fragments are formatted, usually by formatters with graphics support.

Hypertext systems. The first concepts for hypertext were developed by V. Bush in 1945 who designed a machine that should manage documents like text, pictures or notes. It allowed the construction of paths between related information which could be used for browsing. This should allow working with stored information analogous to human thinking.

"The human mind [...] operates by association. Man cannot hope fully to duplicate this mental process artificially, but he certainly ought to be able to learn from it. One cannot hope to equal the speed and flexibility with which the mind follows an associative trail, but it should be possible to beat the mind decisively in regard to the permanence and clarity of the items resurrected from storage." (Bush [1945])

Hypertext systems may be classified by their intended usage. To give an overview about existing hypertext systems, I will give a short description of four categories of hypertext applications: bibliographic, idea, information and notecard systems.

Bibliographic systems. Bibliographic systems are used to maintain on-line libraries with various documents. They offer a simple and comfortable interface, not only for reading the documents, but also for working with them. This includes the creation and critical review of texts so that an original version and the reader's modifications may both exist in the system. Finally, this leads to the vanishing of the distinction between reader and writer.

Quite often, the propagators of these systems plead for the infinite computerization of the world and for the abolishing of printing. They do forget the problem of access, which may be easily monopolized and controlled. Financial resources, modern equipment and technical skills would be necessary requirements for taking part in academic (and social) life, resources that are available for only minority of mankind. I do not believe that the social circumstances of new technical systems automatically become democratic and fair (or, at least, free of charge), and any propagation of a single technique as a universal solution lacks realism.

Idea systems. A second group of hypertext systems is designed for the interactive work of one person or several people on a problem. They help collect ideas and organize them so that nothing gets lost and different views on a problem are possible. They may be compared with the notebook of an author or developer that offers possibilities for retrieving and grouping. (Of course, you need your computer next to you all the time in case you get ideas at unusual places.)

Information systems. Information or browsing systems are small bibliographic systems, used for special purposes such as demonstrations, providing for courseware or a collection of information about a special item (e.g., all information about a certain bomb or the time table of a railway station). Usually, they are made for reading only, so that after an initial phase of authoring the content is static. Quite often, they provide only a limited overview about the structure of the document, and you can only follow the keywords for more information. Disorientation is a frequent consequence of this system.

Notecard systems. These systems are not designed for special purposes, but can be used for various applications for which they provide reading, writing, maintaining, and browsing facilities. Systems that are currently popular are NoteCards by Xerox and HyperCard by Apple which are designed as electronic card-index boxes which allow several sorting criteria of the same data.

Concepts of hypertext systems. All hypertext systems have some basic concepts in common. Hypertext documents are non-linear. They offer information that can be dynamically arranged by a desired criteria. They allow the incorporation of ideas in a document and to connect them later in another step so that the author can add and rearrange components freely.

The fragments are the nodes of the graph structure that constitutes the document. They represent a single idea, a specific item. As they are meant to be arranged in various sequences, they should be independent from others, except by connections that can be represented by links. They may be named, can have attributes, or be of different types.

The system supports the management of the fragments and linking them together. Fragments can be viewed or manipulated (i.e., edited), and the structure is represented and can be browsed. When viewing a fragment, the reader can follow or ignore links to other fragments that are related, or entirely different paths may be selected in the document representation. Links may lead to texts that belong to a keyword, annotations, supplementary information, or successor fragments. They may be named and can have attributes, too.

One can use references that are non-hierarchical (e.g., cross references), but they may as well be hierarchical so that a substructure of related information is built. While the first ones are usually represented by (graphical) markup in the text, the latter ones exist on the management level and are represented in the document browser.

Using a hypertext system. Basic hypertext features are the editing abd viewing of the fragments. If they contain not only text, but also pictures, tables or video, more powerful processors are needed to represent the contents in a formatted form. Some systems also offer multi media facilities like audio or video components.

Another aspect is the representation of the document structure. Maps of the graph are used in which one can navigate. They show the actual position and sometimes have different levels of detail to reduce the complexity.

Next to viewing and editing, the systems usually offer features for searching and selecting. Many hypertext systems offer query techniques for the retrieval of information. As well, views are possible that filter the whole document according to some criteria. Both are usually done by attributes of nodes and links, but there also exist content retrieval systems.

Another feature is the possibility of versioning. This allows one to maintain different versions of text or to keep modification logs. If the system supports access control and offers synchronizing mechanisms, co-authoring is possible as well.

For authors, hypertext offers support for design, structuring and presentation. Nevertheless, the necessary actions for adding new parts may also lead to diversion, and provisional structures may result that are not revised later. In general, human association is not explicit, but subconscious. It is not only determined by the problem's inherent logic, but also by aspects like social environment. Hypertext forces authors to build associations explicitly. This may be quite difficult and irritating (just like learning the first programming language) because a way of thinking must be practiced that is not "natural" and demands a higher degree of detail than the human is used to.

Readers (the author is also a reader) may browse through the text in any variation and with any level of detail they like, supported by tools for navigation. But that also means making a lot of decisions in a short period of time, not knowing whether they are sensible, because no "glimpse" is possible. Following a keyword or selecting a node only by its name may lead to many impasses. Following several paths at a time is quite stressful, too.

The visualization may be helpful for orientation, but in a heavily linked structure it will be difficult to find an adequate and clear representation. Clipping or reducing the complexity will be necessary which destroys information that might be needed.

Hyper documents. The above explanations show that hypertext is not equally useful for all kinds of documents. Hypertext is developed for non-linear documents, such as a notebook, a reference, or a lexicon. These hyper documents consist of information fragments that are more or less independent from each other, having a relation that is marked by a link.[2] The resulting document is dynamic. The reader builds his own text by finding his individual way through a database of interconnected information.

This means that hypertext systems are information systems with an interactive environment that supports structured authoring and dynamic reading of the available material. On the one hand, the reader no longer depends on structures designed by the author. He may freely find the structural representation and the parts of information he is looking for.

On the other hand, the author will group and structure the information in a representation he likes best, though he knows that the reader may look at the hyper document in a totally different way. He can try to avoid this by creating fixed structures that are absolutely necessary for comprehension.

The consequence is that hyper documents may offer a very different degree of freedom to the reader: The more the author fixes structures in it, the less variability is left for the reader, until we end up with totally fixed systems in which the paths cannot be modified, or the information is hidden from the reader (depending on his classification). These methods are often used in tutorial systems.

Hyper database systems. Next to applications that are based on documents, hypertext is being used more and more as the interactive interface for nonstandard database systems. It serves for retrieval, expert system requests, and the like. While there exist facilities for the transformation of normal database output into TEX to produce lists of the selected objects, the problem of publishing the contents of a non-standard database is yet unsolved because the mapping of objects to markup is not trivial. As more and more information of all kinds will be managed in databases because they support multi-access, versioning and consistency, this will be an important request of the future.

Hypertext and the Traditional Publishing

As we have seen, hypertext is typically used for a specific kind of document that is different from standard text. Nevertheless, the question remains: Can hypertext concepts be used for traditional electronic publishing, i.e., for the production of printed documents? And is it possible to integrate them in

[1] That this is not trivial is described by Raymond and Tompa [1988].

an environment that uses typesetting systems like TEX during the electronic authoring as well as for printing?

Hypertext and printing. We do not want to discuss here the usage of TEX as a formatter for a hyperbased on-line information system. Printing means that one plans to reach a widespread readership. Hypertext limits the potential reader to a small group with connections to the computer system that holds the text. Presently, interactive access over wide area networks which work with a lot of documents is not without problems. And reading from the screen is not always what we want, even if access and all technical problems are solved.

Hypertext documents are not made for printing. Of course, fragments may be extracted and printed on paper, but the non-linear structure of the whole document would force one to choose a mapping on a sequence that will be inadequate in many cases. The sequence of reading can no longer be chosen interactively, and a visualization of the links must be established so as not to destroy the structure. Contrary to this, a printed document usually has a linear structure and the author intends a sequence of reading. This sequence influences the contents of the text components because they may depend on information given before.

Database demands. New demands arise from the request of publishing the contents of non-standard databases. Hypertext features with their interactive concept are not sufficient. Paper versions are still needed, e.g., for the management. A markup is not enough either. It supports only the printing aspect and not the authoring. An integration of both markup and hypertext will lead towards maintainable systems for both the interactive maintenance of the contents and the publishing of it.

Authoring with hypertext. Now, let's take a closer look at the authoring aspect of hypertext. There are some good reasons to use hypertext for the authoring of printed documents. Hypertext allows the management of structured documents that are variable in both structure and content in a way that adds a new quality. It is this support of variability that makes it much more powerful than a structure editor.

The interactive environment is very helpful for the maintenance of a document, especially for those that are long-living and are intended to be modified and revised again. Annotations can be added easily, and versioning is usually supported or can be introduced by the author himself. Structural elements such as glossaries, footnotes or an index can be mapped to the hypertext concept. The problem of finding an adequate representation of the text structure in the hyper document will no longer be severe, because there are no longer readers of this document form. Problems are also reduced by the possibility of structure visalization. Navigation helps to locate points of interest that the author, who is more familiar with his text than a reader would be, may identify more easily. This is especially true because the named fragments offer better identification criteria than files with naming restrictions.

Hypertext is very useful for the authoring of a specific kind of document, that is, those that exist in different variations but are maintained and updated together. They are 'hyper-texts' with a non-linear structure and may contain free nodes, but the resulting printed document will be linear.

An example is standards in different versions with history and rationals. Hypertext allows one to realize such a document as a base of text fragments through which different paths exist with common parts. This considerably reduces updating problems, and the reorganization of the document structure can be done easily and coordinated by just modifying the paths. Another example is manuals of the same software for different computers, which leads to very similar documents, yet with different components (sometimes naming conventions only). For this application — maintaining a non-linear document that consists of several linear, printable documents — we developed the prototype of an authoring system with hypertext concepts at the Technische Hochschule Darmstadt.

An authoring system for printed documents that uses hypertext concepts needs some modifications and additions to standard hypertext features. The interactive graphical user interface will be used, but it must support the extensions and adaptions to the special requests of linear text. First, the whole document must be accessible, not only single fragments. Viewing the whole document is as important as printing it. But fragmentation offers more flexibility on this point. Features can be added for selecting parts of documents which can be viewed, selected, browsed, and printed. It must be assured that the selected fragments can be transformed to an acyclic, connected graph with one root.

Specific problems with linear texts in a hypertext based environment arise from the interdependence of the fragments in a traditional document. The possibility of rearrangements is restricted, but the restrictions can be specified in the hyper document by using special fragments for annotations or by using attributes that lay them down. And the feature of (hierarchical) structuring and composition may be used to build components that belong together and should not be rearranged except with modifications in the contents.

TEX and Interactive Authoring Systems

Previously we talked about printing without thinking about the realization. Of course, what we want is to use TEX for formatting so that we can view and print documents (and parts of them) by using `dvi` drivers. To use TEX in an interactive environment, some reflections about the demands of TEX must be made.

Requirements. To use the TEX and driver facilities, the authoring system should contain features of a TEX menu so that the printing and viewing process can be parametrized, or search paths can be specified. This should be designed with the same "look-and-feel" philosophy as the rest of the interactive environment. This requires either a special TEX authoring system or one that can be enlarged by the formatting components.

TEX is a batch processor which requires complete documents for processing. Complete means that the grouping levels are balanced and that mode switches are reset again. All initializations like the selection of formats and style options or user defined macros must be available, and the termination sequences are needed, too.

Extensions. For the authoring system described before, that means that the viewing and printing mechanism must add special user defined pre- and post-information into a document for TEX to handle. This document must be extracted to a file as a whole so that TEX can process it.

To allow viewing and printing of any fragment (or document), the restriction of balanced TEX usage in fragments must be forced (probably by a structure editor), or composed structures must be built with components that belong together and cannot be handled separately. This is very difficult to control because of the various group symbols and the possibility of grouping via macros.

Conclusion

TEX was designed as a typesetting machine. The last section discussed the integration of TEX in an authoring system. It is this aspect of TEX which has the consequence that TEX and hypertext is *not* the future of electronic publishing.

TEX is both — optical and logical markup. The fragmentation and structuring of a document is incompatible with optical markup. Fine tuning of the output within the hypertext system is impossible. Problems will arise with error location and handling. Syntax error free TEX input from humans cannot be assured because a structure editor would be necessary that is a complete TEX interpreter to deal with the user changeable lexical analysis and the dynamic binding of token.

In a hypertext system, the user should not input original TEX. Instead, we should encourage the separation of optical and logical markup in TEX by providing markup systems like LATEX (or SGML with a TEX converter) and a variety of styles for the author. Mapping to the optical markup should be provided by the system, leaving the design to professionals.

The separation of author, typesetter and printer has proved to be useful over hundreds of years. Abolishing this splitting of responsibility for contents, form and realization has not led to satisfactory results in the desktop publishing area nor in hypertext systems. Modern technologies allow us to realize these demands as an interwoven electronical process with hypertext and logical markup on one side and TEX and drivers on the other. Let us fill this possibility with life.

Bibliography

The following articles give an introduction into the hypertext technology, present existing hypertext systems and discuss the advantages and problems of their usage.

Bush, V. "As we may think." *Atlantic Monthly* 176, pages 101 – 108, 1945. reprinted in: CDROM: The New Papyrus. Microsoft Press, 1986.

Conklin, E.J. "Hypertext: An Introduction and Survey." *IEEE Computer* 20(9), pages 17 – 41, 1987.

Meyrowitz, N., A. van Dam, N. Yankelovich. "Reading and Writing in the Electronic Book." *IEEE Computer* 18(10), pages 15 – 30, 1985.

Raymond, Darrel R., Frank Wm. Tompa. "Hypertext and the New Oxford Dictionary." *Communications of the ACM* 31(7) (Special Issue Hypertext), pages 871 – 879, 1988.

Experiments in TEX and HyperActivity

L. Carr, S. Rahtz and W. Hall
Department of Electronics and Computer Science
University
Southampton S09 5NH
email: `L.Carr@uk.ac.soton.ecs`

Abstract

Publishers are starting to explore ways of making their traditional books available on screens, using techniques grouped together as "hypertext," and have tended to concentrate on the problems of linkage and navigation. We believe that both structured writing and traditional typesetting skills have an important part to play in the screen-oriented books to come, and we present the results of various experimental systems which use TEX in the process of creating a hypertext document. Problems tackled include the use of TEX to format text being displayed on screen in variable sized windows, embedding generic hypertext navigation tools in the source and using a single source to generate both printed and hypertext versions of a document.

Intro

Ted Nelson (*Literary Machines* [1987] is a good introduction to his ideas) coined the term *hypertext* to describe "non-sequential writing," or works which could not be expressed as a simple linear sequence of their contents. This includes:

branching texts where the reader is presented with a choice between several paths to follow through the work

interconnected texts which make reference to other parts of themselves or parts of other, separate works

active texts which modify themselves according to some particular criterion, e.g., by recomputing one of its own tables of figures from the latest available stockmarket information.

multimedia texts which are composed of material from disparate information media (such as text, graphics, sound and video). These works are known more generally as "hypermedia" documents.

Hypertexts predate computers — novels frequently have concurrent threads of action (branches) as well as flashbacks (local interconnections) and literary cross-references (external interconnections; for example, we cannot read David Lodge's *Small World* without reference to a huge body of older literary forms) — but the computer has made hypertexts much more practicable in three ways:

presentation A reader may follow branching pathways and connections to other documents by selecting the appropriate material (usually with a mouse) and pressing a button.

speed The computer may retrieve information several orders of magnitude faster than a human can turn pages in a book or fetch a new book from library shelves.

information unity The computer acts as a control centre, allowing the many media available to it to be manipulated in a common fashion.

Hypertext systems are characterised by the kind of information that can be stored and the ways in which the information can be interconnected. Information is usually divided into discrete chunks (or *nodes*) which are connected through *links*. Some systems such as HyperTIES use fixed-sized nodes which only contain textual data, others (such as InterMedia, described by Yankelovich [1988]) allow nodes to contain arbitrary amounts of textual and graphical information. The linking capabilities vary widely: some link from a specific point within a node to a destination node, others allow a connection to be established between stretches of text within nodes. Some links also have information associated with them: a name, a type or a title.

The similarity between creating a document using a generic markup system like LATEX and creating a hypertext document is quite pronounced. In both activities the author is committing a set of ideas and thought processes to a medium and attempting to arrange the information within the constraints of the structure provided by that medium. In LaTeX, the structure is that of technical authorship and is essentially *hierarchical* (chapters of sections of subsections and subsubsections) whereas a general hypertext document has no controlling structure, and is usually referred to as a *network*.

The aim of LaTeX is to allow the author to concentrate on the logical structure of the text and not to be concerned about details of its physical representation. Various different physical representations can be achieved by the use of the `\documentstyle` command. The LaTeX document is, in fact, a hypertext because it exhibits both branching (sections are composed of many subsections which are not necessarily meant to be read in sequence) and interconnection (through the use of `\ref`, `\label` and `\cite` commands), and so by the application of a different document style it should be possible to view a technical report or book as a hypertext network.

The next sections describe several attempts to use LaTeX as a tool for producing hypertext documents.

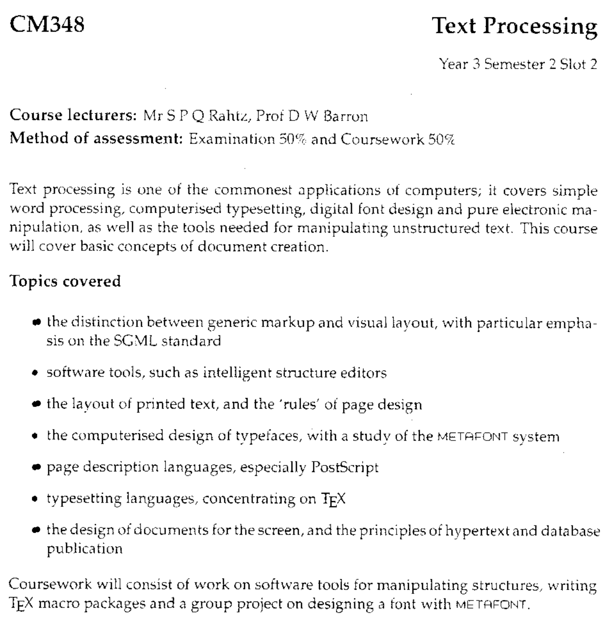
Two Approaches

The function of LaTeX is to take a logical structure (the document's contents with embedded markup) and from it to produce a physical structure (the device independent description of a set of pages). If a hypertext network is required instead then two options are available: to use LaTeX as either

1. a structural markup language which is to be interpreted by an independent system
2. a formatting engine for preparing the individual nodes of a hypertext network

The first alternative views LaTeX as a document interchange standard in competition with SGML (e.g. Bryan, [1988]) or Microsoft's document interchange Rich Text Format (RTF). The document is processed by a *pseudo*LaTeX which is mainly concerned with mapping the continous stream of text onto a network of discrete nodes which will be managed by an independent hypertext system.

An example of this process is Southampton University's Computer Science Prospectus as shown in Figures 1, 2 and 3. The prospectus was originally written in LaTeX so that printed copies could be sent to applicants but a requirement grew for an "interactive guide" to the Department (like that successfully developed for Glasgow Computer Science, and several other sites) and so information from the


CM348 **Text Processing**

Year 3 Semester 2 Slot 2

Course lecturers: Mr S P Q Rahtz, Prof D W Barron
Method of assessment: Examination 50% and Coursework 50%

Text processing is one of the commonest applications of computers; it covers simple word processing, computerised typesetting, digital font design and pure electronic manipulation, as well as the tools needed for manipulating unstructured text. This course will cover basic concepts of document creation.

Topics covered

- the distinction between generic markup and visual layout, with particular emphasis on the SGML standard
- software tools, such as intelligent structure editors
- the layout of printed text, and the 'rules' of page design
- the computerised design of typefaces, with a study of the METAFONT system
- page description languages, especially PostScript
- typesetting languages, concentrating on TeX
- the design of documents for the screen, and the principles of hypertext and database publication

Coursework will consist of work on software tools for manipulating structures, writing TeX macro packages and a group project on designing a font with METAFONT.

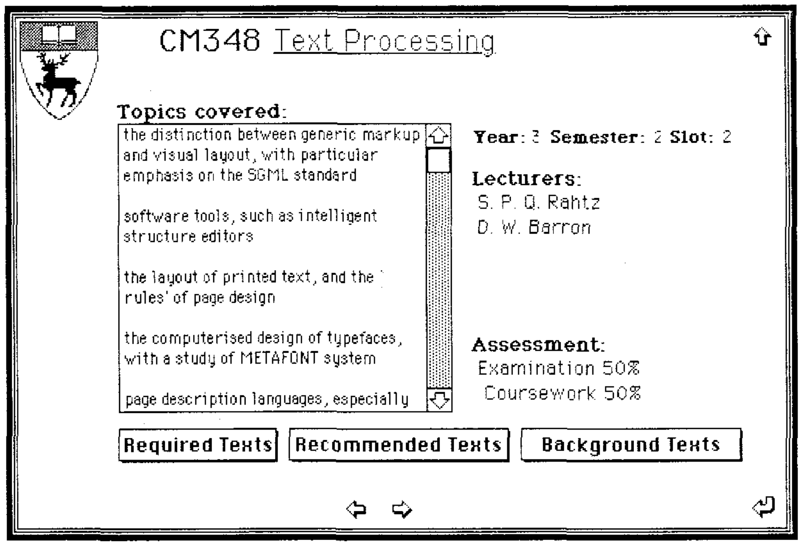
Figure 1: A Page From the Computer Science Prospectus

prospectus had to be merged into a HyperCard stack along with photographs and maps.

To do this a simple *pseudo*LaTeX interpreter was written in HyperCard's programming language HyperTalk (see Figure 4 for a brief extract). The interpreter scans through the document filling nodes with text and creating new nodes whenever a new structure (chapter, section or subsection) is encountered. Subsequent processing allows a table of contents to be built from the titles of each node and reference/label pairs to be resolved.

This approach is advantageous when the document in question already exists in LaTeX format, otherwise it would be more convenient to use a simpler markup scheme. The disadvantage is that specialised markup (such as tabular and math environments) cannot be dealt with automatically. Because the current version of HyperCard does not allow for the use of multiple fonts or font sizes within a field then none of the visual cuing provided by typographic differentiation is possible. This restriction is removed in competitors to Hypercard like Asymetrix *Toolbook* for Microsoft Windows 3, and is about to be remedied in new releases of Hypercard.

A successful example of a relatively complete and advanced pseudoTeX interpreter is the Gnuemacs "info" processor for producing help systems within the Emacs editor; this takes the source of software documentation written in a TeX-based (or LaTeX) generic markup (prescribed by the


CM348 Text Processing

Topics covered:

the distinction between generic markup and visual layout, with particular emphasis on the SGML standard

software tools, such as intelligent structure editors

the layout of printed text, and the 'rules' of page design

the computerised design of typefaces, with a study of METAFONT system

page description languages, especially

Year: 3 **Semester:** 2 **Slot:** 2

Lecturers:
S. P. Q. Rahtz
D. W. Barron

Assessment:
Examination 50%
Coursework 50%

Figure 3: A Screen from the HyperCard Prospectus

```
\code{CM348}\year{3}\semester{2}\slot{2}
\coursetitle{Text Processing}
\headingsection
\lecturers{Mr S P Q Rahtz, Prof D W Barron}
\assessment{Examination 50\%
            and Coursework 50\%}

\topics

\begin{enumerate}
\item the distinction between generic
markup and visual
layout, with particular emphasis
on the SGML standard
\item software tools, such as
intelligent structure editors
\item the layout of printed text,
and the ``rules'' of page design
\item the computerised design of
typefaces, with a study of
the \METAFONT\ system
```

Figure 2: LATEX source for Figure 1

```
repeat
    read from file "LaTeX input"
         until return
    if it is empty then exit repeat
    if char 1 to 9 of it is "\section{"
      then
        delete char 1 to 9 of it
        delete last char of it
        doMenu "New Card"
        put it into field "Title"
        put sectionNumber into field
             "sectionNumber"
        add 1 to sectionNumber
        put 1 into subsectionNumber
        put subsectionNumber into field
             "subsectionNumber"
    else if char 1 to 12 of it is
             "\subsection{" then
        ...
    else
        put it&space after field "Text"
    end if
end repeat
```

Figure 4: Simple PseudoLATEX HyperTalk Script

Gnuemacs team) and reformats it to produce a set of "help" nodes in a hierarchical form comparable to a printed manual. Navigation tools are provided to move up and down and sideways in the tree structure, using the normal Gnuemacs editing commands in "read-only" mode. Alternatively, the same source can be put through LaTeX or TeX to produce a standard printed manual (and in this case effects like figures, tables or maths are activated, rather than being ignored by the "info" system).

The second of the listed alternatives makes use of "true" LaTeX to interpret all the markup in the usual fashion, making use of a modified document style to specify layout for an appropriately sized node in a standalone hypertext system. Once the device-independent file has been produced three different paths have been explored:

1. Use the `dvi2tty` program to produce the page as an ASCII plain text which is fed into a HyperCard field by a simpler HyperTalk program. In this case the logical structure has already been interpreted, so that the HyperTalk script only needs to turn the pre-existing page breaks into new nodes. The table of contents has been created by LaTeX, and all the cross-references have been resolved. Buttons and links may be added in a semi-automatic fashion based on the contents of the text.

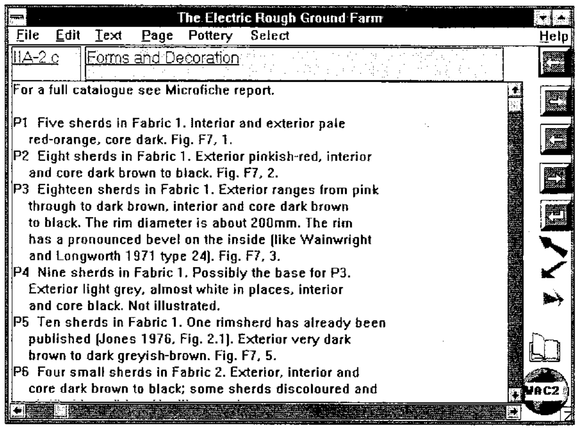
   This approach has been used for *The Electric Rough Ground Farm* (Rahtz and Allen [1990]), implemented on an IBM PC under Windows 3 using Toolbook; an example page is shown in Figure 5. It is necessary, of course, to have a special document style to cope with the following problems:

   (a) page width; we can either decide how wide the field is to be, and format to that width, or we can assume the text will be wrapped by the application. But in the latter case, we lose LaTeX's careful layout.

   (b) we need to suppress running headers and footers, as they are irrelevant in this context;

   (c) it is almost certainly necessary to format using a fixed width font, so that the results will be predictable when converted to ASCII;

   (d) the ASCII codes are missing many useful ligatures and accented letters; getting a 100% useable result from `dvi2tty` is not easy, and requires some modifications to the source;


The Electric Rough Ground Farm
File Edit Text Page Pottery Select Help
IIA-2.c
Forms and Decoration
VAC2

For a full catalogue see Microfiche report.

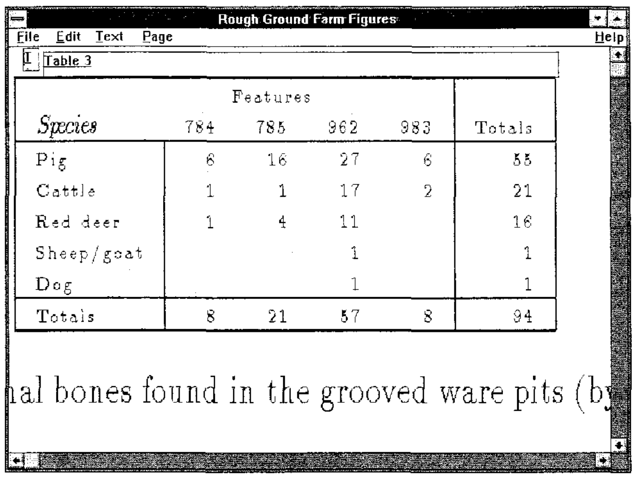
P1 Five sherds in Fabric 1. Interior and exterior pale red-orange, core dark. Fig. F7, 1.
P2 Eight sherds in Fabric 1. Exterior pinkish-red, interior and core dark brown to black. Fig. F7, 2.
P3 Eighteen sherds in Fabric 1. Exterior ranges from pink through to dark brown, interior and core dark brown to black. The rim diameter is about 200mm. The rim has a pronounced bevel on the inside (like Wainwright and Longworth 1971 type 24). Fig. F7, 3.
P4 Nine sherds in Fabric 1. Possibly the base for P3. Exterior light grey, almost white in places, interior and core black. Not illustrated.
P5 Ten sherds in Fabric 1. One rimsherd has already been published (Jones 1976, Fig. 2.1). Exterior very dark brown to dark greyish-brown. Fig. F7, 5.
P6 Four small sherds in Fabric 2. Exterior, interior and core dark brown to black; some sherds discoloured and

Figure 5: A page from *The Electric Rough Ground Farm* in *Toolbook*, formatted by LaTeX

   (e) tables and mathematics cause hopeless problems, to which there is no obvious solution.

   A variant approach adopted in one experiment with archaeological excavation reports (Rahtz, *et al.* [1989]) was to run the LaTeX source through a program to convert it to `nroff` input, using the *mm* macros. The generic structures can be translated unambiguously, and `nroff` can be used to create ASCII output, from which we can proceed in the same way as if we had followed the `dvi2tty` route described above.

2. Save the images from a `dvi` previewer, and use the results as a graphic to cover each node in the hypertext document. Thus we can use `dvipict` to create a bitmap for a HyperCard stack, or Eberhard Mattes' `dvimsp` to create a Windows Paint file for Toolbook (as in Figure 6). This has the advantage of faithfully reproducing the page design (including fonts and spacing) but loses both the structure and the actual text itself. No editing or searching can take place, and buttons and links have to be inserted manually. A solution demonstrated by Wilkins [1990] was to keep the generic LaTeX source behind the scenes in the Hypercard stack, and use a parallel bitmap to show the reader. The underlying ASCII text was used for searching and determination of location, but was never presented to the user. This approach is elegant, in that the source and the "compiled" `dvi` version are kept together, but suffers from the major drawback that TeX has to be invoked to reformat pages if any edits are made (and it is not clear what model to adopt



	Features				
Species	784	785	962	983	Totals
Pig	6	16	27	6	55
Cattle	1	1	17	2	21
Red deer	1	4	11		16
Sheep/goat			1		1
Dog			1		1
Totals	8	21	57	8	94

al bones found in the grooved ware pits (b

Figure 6: Part of table from *The Electric Rough Ground Farm* in *Toolbook*, showing a bitmap from `dvimsp`

for allowing the reader to choose the text to edit).

3. Use a dvi previewer that has been enhanced to deal with hypertext nodes and links. In this case no conversion process is necessary to accommodate the document within a foreign system. This is the route which we have used in the LACE system (described in further detail in the next section) by developing a `hyperdvi` program which converts TEX's device independent format into a form suitable for manipulation as part of a hypertext network.

LACE

At the simplest level, LACE[1] provides a TEX previewing service. Unlike the other experiments described above, LACE is implemented on a SUN workstation using the NeWS (now OpenWindows) windowing software. Based on the PostScript language, this allows compatibility between the formats of documents and pictures that are previewed on-screen and printed on a LaserWriter. A very similar environment exists on the NeXT workstation, but requires greater effort for a proposed implementation for the X Window system, as the X "page description" is lower level than PostScript.

[1] LACE is *not* an acronym for anything; it is partly a pun on the author's initials but more importantly is supposed to reflect the beauty of a structure of thousands of tiny knots of thread forming a coherent and structured whole.

One of the main goals of the LACE project is that there should be little (if any) alteration to the *authoring* process so that a large body of existing documentation can be used unmodified. This goal has been achieved by using a "hypertext" document style option which makes the following changes to the standard LATEX environment:

1. The definitions of all structuring commands are modified to add markers to the `dvi` file, so that they can be extracted from the formatted output.
2. A new command `link` is defined. This takes two parameters — the first is a piece of text over which a button will be placed, the second is the LACE address of a document part. For example, the command `\link{see section 3}{me:section 3}` will make a new window in which this section will "pop up" when the user clicks on the (transparent) button over the text "see section 3."
3. The table of contents, list of figures and list of tables all have buttons over each of the lines, so that clicking on any line will bring up a window with that part of the document in it.
4. The definitions of table, figure and footnote have been changed so that they take up *no space* on the page, but instead inhabit a separate window that is brought up when pressing a button over a reference to them. For example, a footnote window is brought up when the reader presses the button over the footnote marker in the main body of the text. Similarly a figure window is displayed when the reader presses a button over some text like "see figure 3."
5. A new environment *aside* is defined. This behaves very much like a footnote: the LATEX fragment

   ```
   \begin{aside}{Click Me For More
                 Information}
   Green Hypertext allows texts to
   be re-used and has no long term
   effect on the environment
   \end{aside}
   ```

   will produce a button over the words "Click Me For More Information" which, when pressed, will bring up a window with the text that is in the body of the environment.
6. The `label` and `ref` commands have been extended in the form of the `LACElabel` and `LACEref` pair. These two commands differ from the standard LATEX forms in that they

save not only the *number* of the current environment, but also its *type e.g., section 3.4* or *table 2.* As well as doing this, the reference text has a link to the item it references.

7. Any use of the `cite` command to produce a bibliographic citation in the main body of the text has a button over it that brings up a window containing the full bibliographic entry.

When the authoring cycle is completed the document is *published* by entering it into a host-wide database of documents which is presided over by a librarian process, the LACE daemon. The database holds information about each document: its title, keywords, access permissions and location in the host's filestore. Each document is considered to be a database of logical elements (in the structural markup sense). The LACE daemon and the `hyperdvi` process cooperate to provide access to any addressable element of any document published on the local host. Requests for document parts are of the form `<document specifier>:<element specifier>` where the document specifier is either the document's title or a shorter nickname and the element specifier is a string such as `chapter 1`, `section Vehicle Repairs` or simply `Introduction`. In this way a LaTeX document can include the command,

```
\link{See the User's Guide}{guide:section
          Introduction}
```

to reference the introduction from a section of a different document or

```
\link{See Table 2}{me:table 2}
```

to cross-reference a table in the current document.

The `hyperdvi` process is responsible for resolving references to structures within a document and displaying text from the dvi file, while the LACE daemon is responsible for resolving references to documents (potentially across local- and wide-area networks). The LACE daemon also integrates access to other kinds of textual, graphic and video material which may be appropriate to the document being perused.

Figure 7 shows a LACE session. The main window (to the left) was displayed in response to a command-line invocation of `lace Humanities Computing`. The menu (appearing in response to the user's mouse-click) has the usual previewing options of page selection and the common hypertext facility of stepping backwards through the list of elements that have been browsed. The *Contents* and *Tables* submenus are derived from the document's Table of Contents and List of Tables, also seen in this window. The *Add to Trail* item allows the reader to add the current element to a list of "interesting" elements for later browsing. (The trail is a plain text document containing references to each element which can be browsed by the LACE daemon or edited with a text editor. An *involuntary trail* document is also maintained which holds a list of every element that the users ever browses.)

Buttons are transparent patches which cause the cursor to change shape as it is moved over them. Buttons have *actions* associated with them which are invoked when a mouse-press occurs while the cursor is inside them. The actions may be an arbitrary computation (expressed in PostScript), but are usually requests to the LACE daemon to resolve a `document:element` pair.

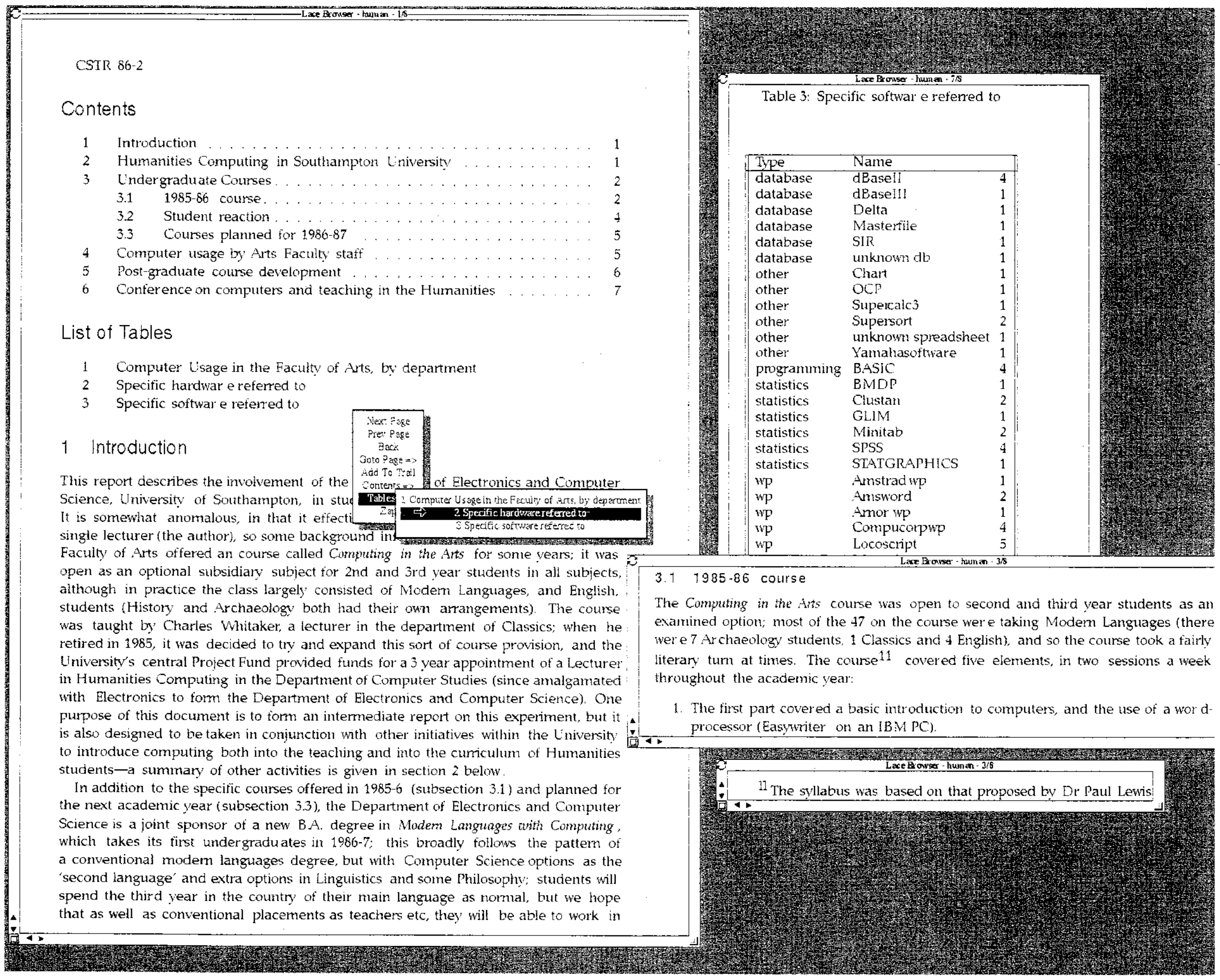
Since the buttons are transparent it is the document style's responsibility to provide some visual cueing for the reader, usually by putting the button's text in a distinctive font. Some buttons may also be deduced by semantic context, for example, "See table 6 for further details".

The remaining windows in Figure 7 have been brought up from buttons in the main window.

Conclusion

One of the problems of producing a hypertext from pre-existing documentation is partitioning the information between separate nodes and finding the links between them. LaTeX allows authors to express the logical structure of their documents and the relationship between the different elements of that structure: this provides a useful basis for constructing a hypertext network. It is also has the crucial advantage that we know we can generate very high quality printed versions of what we write; until screen-reading comes of age, a great many people will much prefer to do sustained reading from a printed page.

Any scheme of logical markup would be suited to the job: SGML is in some ways a more obvious choice (Niblett and vanHoff [1989] deal with work using SGML as a the source language for hyperdocuments; the Perseus Project at Brown University, and the Oxford English Dictionary at Waterloo are well-known examples of large-scale projects based on SGML; it is also relevant to note that the international *Text Encoding Initiative* is proposing an SGML tagset, and if this is adopted it will mean an ongoing supply of SGML-compatible documents). However, LaTeX has the advantage of producing a high quality physical representation (on paper or screen) especially for technical and scientific docu-


Lace Browser - human - 1/8
Next Page
Prev Page
Back
Goto Page =>
Add To Trail
Contents =>
Tables
Zap
1 Computer Usage in the Faculty of Arts, by department
2 Specific hardware referred to
3 Specific software referred to
Lace Browser - human - 7/8
Lace Browser - human - 3/8
Lace Browser - human - 3/8

CSTR 86-2

Contents

1 Introduction 1
2 Humanities Computing in Southampton University 1
3 Undergraduate Courses 2
3.1 1985-86 course 2
3.2 Student reaction 4
3.3 Courses planned for 1986-87 5
4 Computer usage by Arts Faculty staff 5
5 Post-graduate course development 6
6 Conference on computers and teaching in the Humanities 7

List of Tables

1 Computer Usage in the Faculty of Arts, by department
2 Specific hardware referred to
3 Specific software referred to

1 Introduction

This report describes the involvement of the of Electronics and Computer Science, University of Southampton, in stu It is somewhat anomalous, in that it effecti single lecturer (the author), so some background in Faculty of Arts offered an course called *Computing in the Arts* for some years; it was open as an optional subsidiary subject for 2nd and 3rd year students in all subjects, although in practice the class largely consisted of Modern Languages, and English, students (History and Archaeology both had their own arrangements). The course was taught by Charles Whitaker, a lecturer in the department of Classics; when he retired in 1985, it was decided to try and expand this sort of course provision, and the University's central Project Fund provided funds for a 3 year appointment of a Lecturer in Humanities Computing in the Department of Computer Studies (since amalgamated with Electronics to form the Department of Electronics and Computer Science). One purpose of this document is to form an intermediate report on this experiment, but it is also designed to be taken in conjunction with other initiatives within the University to introduce computing both into the teaching and into the curriculum of Humanities students—a summary of other activities is given in section 2 below.

In addition to the specific courses offered in 1985-6 (subsection 3.1) and planned for the next academic year (subsection 3.3), the Department of Electronics and Computer Science is a joint sponsor of a new B.A. degree in *Modern Languages with Computing*, which takes its first undergraduates in 1986-7; this broadly follows the pattern of a conventional modern languages degree, but with Computer Science options as the 'second language' and extra options in Linguistics and some Philosophy; students will spend the third year in the country of their main language as normal, but we hope that as well as conventional placements as teachers etc, they will be able to work in

Table 3: Specific software referred to

Type	Name	
database	dBaseII	4
database	dBaseIII	1
database	Delta	1
database	Masterfile	1
database	SIR	1
database	unknown db	1
other	Chart	1
other	OCP	1
other	Supercalc3	1
other	Supersort	2
other	unknown spreadsheet	1
other	Yamahasoftware	1
programming	BASIC	4
statistics	BMDP	1
statistics	Clustan	2
statistics	GLIM	1
statistics	Minitab	2
statistics	SPSS	4
statistics	STATGRAPHICS	1
wp	Amstrad wp	1
wp	Amsword	2
wp	Amor wp	1
wp	Compucorpwp	4
wp	Locoscript	5

3.1 1985-86 course

The *Computing in the Arts* course was open to second and third year students as an examined option; most of the 47 on the course were taking Modern Languages (there were 7 Archaeology students, 1 Classics and 4 English), and so the course took a fairly literary turn at times. The course^{11} covered five elements, in two sessions a week throughout the academic year:

1. The first part covered a basic introduction to computers, and the use of a word-processor (Easywriter on an IBM PC).

^{11} The syllabus was based on that proposed by Dr Paul Lewis

Figure 7: A LACE session

ments, and there is large body of documents which have been authored using it.

LaTeX's disadvantages should be obvious:

1. its syntax is too complex
2. the document structures it defines are restricted (a simple hierarchy of chapter, section, subsection...)
3. it is a batch-oriented text-processing system which differentiates between the text and its physical representation
4. the TeX system itself is much too large, slow and over-sophisticated for most of the work we will put it to.

It is clear that TeX in a hyperactive world can only survive in a generic markup form such as LaTeX. Much of its power (such as pagination) is irrelevant in this context, and its implementation language will continue to put off many potential designers. But TeX retains a raw beauty of its own, and if we decide that the hyper systems we build must have a formatting engine behind them, we are confident that TeX will continue to be the first choice for many years to come.

Bibliography

Bryan, M. 1988 *SGML: an author's guide.* Addison Wesley.

Nelson, Theodor H. 1987. *Literary Machines.* T.H. Nelson, Swathmore, PA, 87.1 edition.

Niblett, Tim and Arthur van Hoff. 1989. "Programmed Hypertext and SGML." unpublished MSS available from authors at Turing Institute, University of Strathclyde.

Rahtz, Sebastian and Tim Allen. 1990. "Dynamic excavation reports, for good or evil," in *Pre-circulated papers for Information Technology themes at World Archaeological Congress 2, Venezuela, September 1990.* World Archaeology Congress.

Rahtz, Sebastian, Les Carr and Wendy Hall. 1989. "Creating Multimedia Documents: hypertext-processing," in *Hypertext state of the art*, ed. Ray McAleese and Catherine Green, Intellect, Oxford, UK, pp. 183–192. York.

Wilkins, Rob. 1990. "A LaTeX viewer for Hypercard." 3rd year computer science project, Department of Electronics and Computer Science, University of Southampton.

Yankelovich, N., B. Haan, N. Meyrowitz and S. Drucker. 1988. "Intermedia: The Concept and the Construction of a Seamless Information Environment," *IEEE Computer*, pp. 81–96.

The Dutch National LaTeX Effort

Johannes Braams
PTT Research Neher Laboratories
P.O. Box 421
2260 AK Leidschendam
email: `JL_Braams@pttrnl.nl`

Victor Eijkhout
Department of Mathematics
University of Nijmegen
Toernooiveld 5
6525 ED Nijmegen
email: `U641001@hnykun11.bitnet`

Nico Poppelier
Elsevier Science Publishers
Sara Burgerhartstraat 25
1055 KV Amsterdam
email: `n.poppelier@elsevier.nl`

Abstract

In this article, an overview is given of the activities of Working Group 13 WG13 of the "Nederlandstalige TeX Gebruikersgroep" (Dutch TeX Users Group). This working group is also called "Neerlandica," and is interested in anything that has something to do with using LaTeX (and TeX) in a non-American environment. The topics tackled so far range from the design of a page layout suitable for A4 paper by adapting the American layout of `article.sty` to Dutch typographical tastes, to the implementation of a new letter style called "brief."

Introduction

At its founding meeting, the NTG decided to establish a number of "working groups," which would tackle some of the problems encountered by members of the NTG. Some of the subject are "education," "drivers," "TeX for personal computers." At the second NTG meeting, another working group was started. The task of this group was to investigate the TeXnical problems Dutch-speaking LaTeX users encountered and to suggest solutions to these problems.

The first activity of this group was to decide on which problems would be tackled and in what order. We had a number of subjects:

1. The original LaTeX document styles are designed for American-sized paper. The dimensions of this paper differ from A4 paper, which is used most commonly in Europe. The available style option files at the time were not very satisfactory.
2. In Dutch texts, commands that produce text, like `\chapter` or `\abstract`, should produce *Dutch* texts instead of *English*. A way to solve this problem had already been pointed out by Hubert Partl *et al.*
3. The design of the standard document styles provided by Leslie Lamport is very "American" and, at least to our Dutch eyes, a bit "loud." We decided to develop replacement styles that would be more adapted to Dutch typographical standards.

After identifying these tasks, we set ourselves a short-term and a long-term goal. The short-term goal was to provide the members of the NTG with an acceptable page-layout, adapted to A4 paper and to provide a document-style option that redefines commands like `\chapter` to use macros like \chaptername. This seemed a reasonably simple task that would be welcomed by a lot of Dutch LaTeX users. The long-term goal was to develop new document styles, plug-compatible with the styles provided by Leslie Lamport. To this, we added the need for the implementation of a document style for letters that follows Dutch guidelines for the design of letters.

In the following sections we will discuss these topics in more detail. Some of this work has led to articles published or to be published in *TUGboat*.

Page Layout for A4 Paper

As described before, one of the first problems tackled was that of adapting the standard document styles to A4 paper. The solution we sought was to provide a document style option file, and not to modify document styles for this. This was done mainly to avoid having to maintain extra document styles that differ only marginally from the originals.

Obviously, we were not the first LaTeX users to identify the problem, and we knew that document-style options were available in the international TeX community. However, as we were not satisfied with the results these options gave us, we decided to adapt them. This has led to the file `A4.sty` described by Poppelier and Braams. This file started as a combination of two options that were already available. The combination of these two files didn't satisfy our demands, which were:

1. The width of the text should be chosen in such a way that no more than sixty to seventy characters appear on a line of text.
2. When a document is printed two-sided, the texts printed on both sides of one sheet of paper should overlap.
3. The "inner" margin of the document should be wide enough to allow for the binding of the document.
4. The "outer" margin should be wide enough for marginal notes.

After the publication of our article on `A4.sty`, we received a comment that it doesn't handle texts with more than one column correctly. We still have to look into this, but it is a nice example of Gödels principle[1].

The "Loud" American Design of `article.sty`

As discussed before, the modification of the design of the standard document styles was a long-term project. As a first attempt, a document-style option "`sober.sty`" was developed. This file modifies the amount of white space around section heads and lists. It also modifies the fonts used for the various levels of section heads.

Braams, Eijkhout and Poppelier began a more fundamental approach with the development of new document styles, derived from `article.sty` and `report.sty`. The design of these "new" styles was based on discussions with a Dutch typographical designer and books by Treebus and Miles on typographical design. One of the ideas behind the redesign of `article.sty` was to minimize the number of "implied left margins." By an implied left margin we mean a non-zero distance from the actual left margin that is used in more than one element of the document. Examples of implied left margins are:

1. the paragraph indentation,
2. the left margins of items in an "itemize" or "enumerate" list construct,
3. the left (or right) sides of the numbers and labels in such list constructs and
4. the left side of the text of a section heading.

In the standard styles of LaTeX all of these four distances are independent and are different from one another. In the style we have developed, it was decided to strengthen the visual coherence of the layout by taking the same value for each of them whenever possible.

Another idea that was implemented with `artikel1.sty` is that the white space separating a section heading and the text following it should bear some simple relation to the baseline skip, and should not have any stretch.

A third major modification in the design of the article style is a new layout of the table of contents. Both Treebus and Miles are very explicit in their opinion about the layout of a table of contents. They find a layout like the one implemented in `article.sty` "old fashioned" and even confusing. So, a completely different layout was implemented.

As the new layout of the table of contents met with some reservations from users, the old layout is still available through an option.

Besides `artikel1.sty`, we also implemented two other article styles that have a layout that differs from the layout of `artikel1.sty`. `artikel2.sty` is a layout designed to show what can be accomplished by modifying a few parameters in a document style. The third article style is like `artikel1.sty`, but with one major design decision changed. In `artikel3.sty` paragraphs are not indented, but they are separated by vertical space.

[1] See the chapter about the *Contracrostipunctus*.

The design of the document styles `artikel1` and `artikel3` can also be implemented for reports, as `report1.sty` and `report3.sty` respectively, and there is a `boek` style based on the design of `artikel1`.

Modifying the Standard Styles Without Modifying Them

As discussed in the Introduction, one of the topics WG13 should be working on was a solution to the problem of English terms appearing in Dutch texts. The basic idea behind our first solution was already implemented by Partl in `german.sty`. We adopted the idea and created `dutch.sty`. In some ways this file is much simpler than `german.sty`; in another way it is more complex. A major difference between the two files is that `german.sty` just provides parameters and parameter values for the various terms and states in the comment that the user should provide modified document style files that make use of these parameters. The file `dutch.sty` also contains the necessary redefinitions for the various LaTeX macros. This implies that `dutch.sty` can be used with the standard *unmodified* document style files as they are included in every LaTeX distribution.

While implementing `dutch.sty`, it occurred to us that it had some code in common with `german.sty`. This, combined with some discussions at the 1989 EuroTeX meeting in Karlsruhe, led to the idea of building a more universal system of style option files, called the babel system.

Braam's babel system consists of one file, with macro definitions common for all languages and a language-specific file for every language that is to be part of the system. It offers the possibility to switch between languages while processing a multilingual document. Because we wanted this system of style option files to be compatible with the original `german.sty`, the files are implemented so that they can also be used with plain TeX. This is useful because the language-specific files can (and do) contain more than just parameters for LaTeX terms. For instance, for the German language as well as for the Dutch language we have an extra active character. This active character is, among other things, used for controlling the hyphenation of words containing accented letters or explicit hyphens by inserting `\discretionary` commands.

All the user needs to specify is the (main) language used in his document as an option to the `\documentstyle` command. This will instruct TeX to read the appropriate language-specific file. This file checks whether the core of the babel system, `babel.sty`, has been read before. If this turns out not to have happened it inputs `babel.sty`.

When the user wants to add the definitions for another language to the environment in which his document is processed he can use the command `\addlanguage`[2] with the name of another language-specific file as an argument. So the preamble of his document might look like:

```
\documentstyle[11pt,dutch]{artikel1}

\newlanguage{germanB}

\begin{document}
```

So, this example document contains Dutch and German texts. Because the function of the extra active character (") is different for `dutch.sty` and `german.sty`, the user wants to switch this definition when he starts a German part of the document. This can be achieved by adding,

```
        \selectlanguage{german}
```

in front of the German text. When he wants the Dutch settings to be restored, he simply uses the same command with the parameter `dutch`.

As the babel system has been developed in a pre-TeX 3.0 environment, it doesn't use any of the features of TeX 3.0. Perhaps the switching of hyphenation tables and other TeX 3.0 features might be added to the definition of `\selectlanguage`.

The Design of a New Letter Style

In one of its meetings, the "Neerlandica" group decided to try to implement a document style for letters that should conform to Dutch standards for the layout of letters. We have consulted four such standards, from the Dutch standardization institute, the Nederlands Normalisatie Instituut (NNI). They are:

1. NEN-1026 for letters,
2. NEN-1025 for envelopes,
3. NEN-3162 for the structure of documents and
4. NEN-3516 for the design of forms.

The result is a design that cannot be judged by taste: it just implements the standards. This seems quite rigid, but some freedom is left to the user to adapt certain parts of the layout to his own wishes.

This new style is *not* "plug compatible" with the LaTeX `letter` style, although equivalents of some of the macros from `letter.sty` have been provided. The main reason for its "incompatibility"

[2] —"newlanguage— would be more appropriate, but this has become a TeX primitive, so the macro was renamed.

with `letter.sty` is that we have quite a lot of new user commands to either modify some parts of the layout or to fill in some of the fields in the "reference lines." The reference lines contain fields such as "Your letter of" and "Date."

Some of the features of this document style are:

1. If the user doesn't have printed letter paper, he can provide his own letterhead by writing his own macro `\briefhoofd` or he can use the macro `\maakbriefhoofd` to adapt the default letterhead provided with the document style.
2. An option is provided to print short horizontal rules on the sides of the paper as an indication where to fold the letter.;
3. The address is positioned such that it can be used with "window envelopes." The window can be either on the left side (default) or on the right side of the envelope.
4. A user-command `\voetitem` is provided for information about the sender at the bottom of the letter.

An example of this document style can be found at the end of this article.

Conclusion

Working Group 13 of the NTG has had a busy year. We have

1. produced a number of replacement document styles for the standard LaTeX document styles,
2. developed a new document style for letters that implements Dutch standards for the layout of letters,
3. presented a new and improved document style option file for use with A4-size paper and
4. produced a document style option file system that supports multiple languages in one document and provides additional features for specific languages.

In short: the category code of this working group has been `\active`.

Bibliography

Braams, Johannes, Victor Eijkhout and Nico Poppelier. "The development of national LaTeX styles," *TUGboat* 10 (1989) #3, p. 401–406.

Braams, Johannes. "Babel, a multilingual style-option system for use with LaTeX's standard document styles," *To be published in TUGboat.*

Hofstadter, Douglas R. "Gödel, Escher, Bach: een eeuwige gouden band," Uitgeverij Contact, Amsterdam 1985, Dutch translation of *Gödel, Escher, Bach: an eternal golden braid*, Basic Books, New York 1979.

Knuth, Donald E. *The TeXbook*, Addison-Wesley, 1986.

Lamport, Leslie. *LaTeX, A document preparation System*, Addison-Wesley, 1986.

Lamport, Leslie. in: TeXhax Digest, Volume 89, #13, 17 February 1989.

Miles, John. *Ontwerpen voor Desktop Publishing, Alles over layout en typografie op de personal computer*, Uitgeverij Mingus, Baarn 1988, Dutch translation of *Design for Desktop Publishing: a guide to layout and typography on the personal computer*, Fraser, London 1987.

Partl, Hubert. "German TeX," *TUGboat* 9 (1988) #1, p. 70–72.

Poppelier, Nico, and Johannes Braams, "A style option to adapt the standard LaTeX document styles to A4 paper," *TUGboat* 11 (1990) #1, p. 98–103.

Schrod, Joachim. "International LaTeX Is Ready To Use," *TUGboat* 11 (1990) #1, p. 87–90.

Treebus, K. F. *Tekstwijzer, een gids voor het grafisch verwerken van tekst*, SDU Uitgeverij ('s-Gravenhage, 1988). A Dutch book on layout design and typography.

Documenting a TEX Archive

Adrian F. Clark
University of Essex
email: `alien@essex.ac.uk`

Abstract

There are a number of TEX archives in existence: the master sources at Stanford, the LATEX (and other TEXware) collection at Clarkson, Don Hosek's archive at Claremont, and so on. The author is one of a group of volunteers who maintain the TEX archive located at Aston University in the UK. The Aston archive is probably the biggest of the TEX archives, holding complete distributions for Unix, VMS, MS-DOS and the Macintosh, as well as LATEX style files, support utilities, fonts, and so on — a few hundred megabytes in total. Documenting this volume of information is no mean task; indeed, the archive maintainers frequently have trouble remembering where things are!

There is growing interest in coordinating these various archives. While this can only be good for the user, an archive is only as good as its documentation: there's no point having a piece of software archived if no one can find it! The primary purpose of this paper is to raise awareness to the various needs of archive users and maintainers in the hope that it will provoke both discussion and involvement.

Accessing an Archive

The first thing one must know about an archive is *where* it is! Although this may seem a trivial point, a fair proportion of the messages flying around electronic mail and news networks are of the "Where do I find this?" variety.

The second, and by far the most significant, point is *access* to the archive. The majority of TEX (and other) archives are only accessible via anonymous file transfers; a few, such as Clarkson and Aston, run mail-servers, which interpret requests in incoming electronic mail messages and respond with file transfers, directory listings, and so on. But what of the community without electronic access at all? The author is aware of only four publicized sources of TEX on physical media: the "official" Maria Code distributions for several systems from Stanford, the University of Washington's UNIX TEX tapes, Jon Radel's PC distribution on floppies, and the Aston archive, which will send out VMS, UNIX, PC and Mac kits, as well as a complete tape of the entire archive (one fairly full VMS "backup" tape at 6250 bpi).

However, since it is by far the most common means, let us concentrate on electronic access for a few moments. In the Internet world, the standard means of accessing an archive is by anonymous FTP. This is an interactive program which allows one to "log in" to a remote file system (via the username `ANONYMOUS`), list directories and copy files back to the local machine. Although FTP implementations do not typically provide facilities for typing files to the screen, it is fairly trivial to do this in practice on most operating systems. Hence, FTP permits reasonably interactive access to an archive.

The UK academic network, JANET, is based on a rather different set of protocols and supports a rather different type of remote file access. When a request is made to copy a file from a remote machine, it is handled *asynchronously:* the file simply "appears" at some time after making the request. JANET style file transfer (NIFTP) means that a user cannot interactively scan through the filestore (directory) of the remote machine, thereby improving security; hence, directory listings have to be left lying around on the archive machine, stored in files with standardized names. These directory listings have to be updated frequently (this is a good habit for an archive, in any case); in the case of the Aston archive, listings of the entire archive and of the recently created files are generated daily.

For cross-network traffic (*e.g.,* between Bitnet and JANET), neither FTP or NIFTP is possible, and one must resort to other approaches. The most common solution is a *mail server,* a program which receives messages containing requests for directory listings, requests to send files to the requestor, and so on. (The author must accept the stigma of having written the mail server program which fronts the Aston archive.)

Finding the Right File

With only these types of access available, finding the file one wants can be a non-trivial exercise. Perhaps the simplest approach is via the "`whereis`" feature supported by several mail servers, one simply sends off a mail message containing, for example, `whereis recipe.sty`. The mail server program then scans through the archive for any files which have this name and returns their locations in a return mail message to the requestor. Of course, this only works if the name of the required file is known, and this is frequently not the case, and this brings us back to the problem of documentation.

If we consider an archive to be analogous to a book, we can think of the listing of the directory hierarchy as the electronic analogue of the table of contents, since it tells us where each file (topic) is located. However, there is no true analogue of a book's *index*. As anyone who has written a book will confirm, compiling the index is a task which cannot satisfactorily be automated. The closest approach to an index is in the hierarchical help systems offered by several operating systems (this most definitely *excludes* the UNIX on-line manual, which is a total abomination).

To be able to support all types of potential archive user, two forms of documentation are needed: hard copy documentation which can be sent out to people without electronic access, and on-line documentation which facilitates locating files by functionality rather than by name. To save the archivist's valuable time, the most sensible approach is to generate both of these from a common source file. Of course, such systems already exist, such as Digital's VAX Document and the Free Software Foundation's "TEXinfo" (a "LATEXinfo" has recently been devised too). Both these products use TEX as the typesetting engine for printed manuals. Document uses an enhanced version of `runoff` for generating VMS Help files, while TEXinfo uses GNU Emacs (a programmable editor) to generate "info" files, which can be scanned with an Emacs subsystem. The problem with info files is that they can only be read from within Emacs, and not all operating systems run it; indeed, many people prefer other editors, even on systems on which Emacs does run. The author has prototyped a similar system and investigated its use for documenting small sections of the Aston archive.

A Prototype Archive Documentation System

In order to evaluate the above approach to generating an archive "index," the author prototyped a documentation system in the `AWK` language. This system had a (deliberately) very restricted set of commands, which covered operations such as titles, lists, font changes (emphasis and teletype) and keyword specification. General descriptions of the contents of several directories (`README` files) were marked up in this way and processed through GNU `AWK` to generate the following types of documentation:

- LATEX input (each `README` file being a separate section of a larger document)
- `nroff` input, to generate UNIX manual pages
- `runoff` input, to generate VMS Help
- `info` format (from the VMS Help, for use with a stand-alone `info` program written by Nelson Beebe)
- plain text, to be inserted into the individual directories as `README` files

The keywords specified in the marked up text were used to generate index entries for the LATEX document and a specific sub-topic (`keywords`) for the Help. A public-domain Help program was then modified to support searches through only the titles (subject headings), through the keywords, or through the bodies of the help modules. The software was tested on a few volunteers who do not know their way around the Aston archive, and it was found that locating specific items in the archive was then much faster, particularly if the choice of keywords was made carefully. Of course, on such a small-scale experiment, it is dangerous to make sweeping conclusions; nevertheless, user response was favourable.

As a further experiment, a sub-section of each entry also contained a list of the relevant files in the archive, and this was used to automatically generate a set of (NIFTP) file-transfer commands to retrieve from the archive all the files needed to get the particular piece of software working. Alternatively, the same information could be used to prepare a printed request to the archive maintainers to specify the files on magnetic media.

A Real Archive Documentation System

While the prototype outlined above could not be used in the real world, it does indicate a practicable way of documenting an archive. However, one can take the idea a step further. There is no reason why the Help system which one uses to scan through the archive's contents should reside only on the archive machine. Given a machine-portable Help program, one could arrange that its databases (i.e., the descriptions of various archives' contents) could be kept up-to-date automatically. This could be achieved either by updating the local databases on (say) a weekly basis by automatic jobs requesting modifications, or by "registering" the local system with the archive and having it send out modifications as and when required. (The latter approach is similar to the one used in the UK for distributing the database of network names and addresses for certain types of computer, while the latter is similar in concept to the distribution mechanism for network news.)

There is no reason why the above scheme should be limited to an archive of TeX-related material. What it would require would be the willingness of archive maintainers *and* people making submissions to archives to settle on a common but painless markup format for `README` files, and the standardisation of a Help program across several platforms.

LaTeX-Paragraphs Floating around Figures

Thomas Kneser
Gesellschaft für wissenschaftliche
Datenverarbeitung Göttingen
Am Fassberg
D-3400 Göttingen, FRG
Bitnet: `tkneser@dgogwdg1`

Abstract

A LaTeX-environment is presented which has the following property: Figures which are substantially narrower than `\textwidth` are placed automatically right or left justified within one or more paragraphs. Such figures can be set as easily and in the same way as LaTeX's standard figures.

When to Use the FLOATFIG Style Option

Figures often do not fill the full page width. If the width of such figures is only half of the page width or even less, lines of text should be set beside the figures, or—from another point of view—figures should "float" in paragraphs. Figures 1 and 2 show examples of such "Floating Figures".[1] They can be set by using our `FLOATFIG` style option.

The macros which make up the `FLOATFIG` style option are based on PLAIN-TeX macros developed by Thomas Reid (*TUGboat*Vol. 8 # 3 page 315), who showed how to set figures *right justified* with paragraphs floating around them. For such layout Reid chose the term "Floating Figures".

This choice could cause confusion, when we adapt these macros for LaTeX, since Leslie Lamport uses the term "float" for objects which are realized as TeX-`\inserts`. While Lamport's floats are floating in the "main vertical list", floats introduced by Reid are floating within paragraphs. For what follows we adopt the latter definition.

How to Use It

The Floating Figures style option is fully compatible with LaTeX's standard figure facility:

1. Floating Figures and standard figures may be requested in any sequence,
2. Floating Figures can be captioned like standard figures,
3. Captioned Floating Figures are inserted in the list of figures which may be printed by the standard `\listoffigures` command.

[1] The pages shown are a resetting of some pages from HERMANN WEYL'S book on Symmetry (German edition, Birkhäuser Verlag 1955)

A Floating Figure may be requested as follows:

```
\documentstyle[floatfig]{article}
\begin{document}
\initfloatingfigs
           .
           .
\begin{floatingfigure}{5.6cm}
\vspace{6.0cm}
% optional !
\caption{Intermolecular potential K-Xe}
\end{floatingfigure}
           .
           .
\end{document}
```

It is essential that `\initfloatingfigs`, which initializes the `floatingfigure` environment, follows `\begin{document}` immediately. Otherwise some formatting errors will occur.

A Floating Figure may only be requested in vertical mode, that is between paragraphs. "`5.6cm`" in our example specifies the width of the figure space.

A Floating Figure will be set as soon as possible after the request for it has been encountered by LaTeX. That means, it will be tested whether there is enough vertical space on the current page; if not, the figure will be moved to the next page.

Floating Figures are set *alternating*, that is on the right hand side on odd and on the left hand side on even numbered pages.

Restrictions

1. The `FLOATFIG` style option may not be combined with the `TWOCOLUMN` style option,

2. A Floating Figure cannot appear in a paragraph which begins on top of a page.

How It works

We have extended the macros designed by Reid with regard to:

1. fitting them into the LaTeX context,
2. justifying Floating Figures right and left, and
3. the generation of warning messages for "collisions" of two Floating Figures.

Fitting into the LaTeX context The PLAIN-TeX implementation by Reid is based on a redefined `\output` routine:

```
\edef\oldoutput{\the\output}%
\output={\the\outputpretest
  \ifoutput\oldoutput\fi}
\outputpretest={\outputtrue}
```

If a Floating Figure is requested, the content of the `\outputpretest` token register then decides:

1. if there is enough vertical space to set the Floating Figure,
2. if setting of another Floating Figure is already in progress, or
3. if indeed the current page is ready to be sent to the DVI file.

TeX has to deal with more than one paragraph until a Floating Figure is completely processed. During this process, the redefined `\output` routine is called at the beginning of *every* paragraph; this is done indirectly by expanding the control sequence `\tryfig`. Therefore, the `\everypar` token list is prepared by the following command sequence:

```
\edef\oldeverypar{\the\everypar}
\everypar={\tryfig\oldeverypar}
```

Now `\tryfig` triggers the (modified) `\output` routine, which then makes the decisions mentioned above.

Adopting this concept when using the macros in the LaTeX context, we are faced with the following problems:

1. At the time `FLOATFIG.STY` is read in, the `\output` routine is still undefined, and remains undefined until `\begin{document}` is expanded; so the redefinition of the `\output` routine has to be done after `\begin{document}` by the command `\initfloatingfigs` (see section "Known Problems" below).
2. There are situations where LaTeX decides to redefine the `\everypar` token list without saving the former content; this occurs, for instance, when expanding a `\section` control sequence. We overcome this by redefining `\everypar` whenever the `\floatingfigure` environment is entered. So to avoid problems, a Floating Figure should be requested early enough before any sectioning control sequence (see also subsection "Misleading collision warnings"). Furthermore, the conflicting definitions of `\everypar` are the reason why Floating Figures cannot move across section boundaries.

Justifying figures right and left The problem to be solved is whether a particular figure has to be set left or right justified. This decision has to be made according to the value of the page count (left if even, right if odd). This is because we are dealing with the well known problem of associating a certain part of input text with the number of the page on which it will be finally set .

As pointed out by Donald Knuth in *The TeXbook*, this association is made at `\output` routine time. Therefore the problem is not so hard to solve, since in Reid's version there is already a modified `\output` routine which decides if a particular figure will fit on the current page. As a by-product of this decision one easily gains the information "odd" or "even" for the page count of the current page. So our problem is reduced to the following simple decision:

```
\ifodd\count0 %
   \hbox to \hsize{\hss\copy\figbox}%
   \global\oddpagestrue
 \else% leftsetting
   \hbox to \hsize{\copy\figbox\hss}%
   \global\oddpagesfalse
 \fi% \ifodd\count0
```

Collisions of Floating Figures We define a collision as a situation where:

1. a Floating Figure is requested before a predecessor has been finished, or
2. some sectioning is requested before a Floating Figure has been finished.

While the `FLOATFIG` style option cannot avoid such collisions, it will recognize them. For diagnostic purposes we have therefore defined the switch `\iffigprocessing` and another count register called `\ffigcount`. This count register is used to attach a sequence number to each Floating Figure, so they can be identified uniquely in collision warning messages. These sequence numbers are not to be confused with the figure count maintained by standard LaTeX.

Known Problems

Need of Initialization The present version of the style option needs to be initialized by the control sequence `\initfloatingfigs`, as mentioned above.

We hope a later version will initialize itself when the first request for a Floating Figure is encountered. One problem with such an automatic initialization seems to be that one is grouped down, being inside a LaTeX environment. So far, I must confess, we have failed to make the respective settings `\global`.

Misleading collision warnings As mentioned above, Floating Figures do not move across section boundaries. If a Floating Figure is requested near the end of a section, the actual figure will be truncated, if it does not fit into the current section.

If this has occurred, for instance, with Floating Figure number 4, a collision will be reported when a request for Floating Figure number 5 is encountered. But the message will tell us that there is a problem with figure 4 without further explanation. So one has to check in each case that the problem is *not* caused by collision with figure 5 but with a section heading. What is even worse: if there is no Floating Figure 5, but only a truncated 4, one gets no warning message at all! At the moment, one can only check this on the printout. We pointed out in section "Fitting into the LaTeX-Context" why this unfortunate situation cannot easily, if at all, be remedied.

A minor blemish is due to a retardation caused by the `\everypar` mechanism. Floating Figures in consecutive paragraphs seem to be disapproved. The warning messages generated in this connection can be ignored.

Conclusions

Working on `FLOATFIG.STY` we had some unexpected problems which were caused by LaTeX's somewhat unsafe assignments to the `\everypar` token list on the one hand, and by the fact that the use of this token list is fundamental to the algorithm designed by Thomas Reid on the other hand.

We did expect problems in fitting the Floating Figures in with LaTeX's handling of figure captions: that is, to achieve a single figure caption numbering for standard figures and Floating Figures and to get both types listed together by the `\listoffigures` control sequence. But this problem was easily solved by the following local definiton within the `FLOATFIG` environment:

```
\def\@captype{figure}
```

Obviously this is due to the fact that LaTeX's caption apparatus is thoroughly parameterized.

The `FLOATFIG.STY` file is stored in the `EARN/BITNET` listserver in Heidelberg;
send the command:

```
GET FLOATFIG ZOOUUE LATEXSTY
```

to `LISTSERV` at `DHDURZ1` to obtain a copy of the style option file.

The Document Style Designer as a Separate Entity

Victor Eijkhout
University of Illinois, CSRD, 305 Talbot Lab., 104 S. Wright St., Urbana, IL 61801
`eijkhout@s12.csrd.uiuc.edu`

Andries Lenstra
Department of Mathematics, University of Nijmegen, Toernooiveld 1, 6525 ED Nijmegen, The Netherlands
`lenstr@sci.kun.nl`

Abstract

An argument for the need for a programmable meta format: a format that introduces a new syntactic level in TEX for document style designers.

TEX has a number of characteristics that set it apart from all other text processors. Its unsurpassed quality of text setting and its capabilities for handling mathematics are some of the more visible aspects. On a deeper level, however, the extreme programmability of TEX is just as big an asset. Any layout can be automated to an arbitrary extent. (It is strange that *The TEXbook* gives almost no hint of this.)

The form such automation usually takes is what is called 'generalized markup'. The person who keys in the text has at his or her disposal commands that describe the logical structure of the document, and as little as is possible of the visual structure.

Document styles as they appear in LATEX are examples of this. Here the layout is not merely automated, it is completely hidden from the end user. In particular, the same input can produce widely different output by letting it be interpreted by different styles.

With LATEX, however, the problem is the production of document styles. This is a major task, and consequently most people use either the standard styles, or minor modifications of these. Furthermore, LATEX does not offer sufficient tools to produce even moderate variations on the layout of the standard styles.

For scientific use of TEX one can become reconciled to this situation. A scientist should be more concerned with the contents than with the looks of a document, so if there is a format that offers all the tools to get those contents across to the reader, the visual appearance is of secondary concern. We may conclude that, for scientists, LATEX fits the bill.

When a document is not a scientific article, however, the inflexibility of LATEX renders it useless. The alternative would seem to be plain TEX, but there the objection is the long and slow learning curve.

One way out of this dilemma is the 'front end to TEX' approach taken in *The Publisher* from Arbortext and *Grif* from Gipsi. Both systems present almost a 'wysiwyg' (what you see is what you get) interface to the user (the term 'wysipn', what you see is pretty neat, has been used), and allow altering style parameters via dialog boxes and menus. In both cases, however, programming the basic style structure and appearance is still less than trivial.

In this article we describe an approach which brings down the complexity of programming a style to that of using it. We propose a front end programming language for style design, which is itself implemented in TEX.

The 'Checklist' Approach to Style Definition

If one compares TEX to mouse-and-menu text processor packages, one runs into two basic characteristics of programming that constitute a disadvantage when learning TEX.

The obvious first point is that TEX has a *syntax*. Every TEX programmer knows that you can have no end of fun with missing or mismatched braces. Mouse riders are not bothered by this. One can, at most, click the wrong item, but one cannot click in the wrong way.

The second point is more subtle: programming involves and imposes algorithmic thinking. Consider the ordinary loop statement

```
for $i$:=$1$ to $n$
```

In many cases, all that is meant is

```
for all $i$ between $1$ and
        $n$ inclusive
```

Thus, the syntactic formulation contains a sequentiality that may semantically not be present. Similarly, the statements in a macro definition are sequentially ordered, even when the corresponding actions are in no such relation.

But even when actions are sequentially ordered, it should not always be necessary for a style designer to specify them in that same order. For instance, in the design of a list environment the amounts of white space above and below the list, and the amount of indentation of the list, should be specifiable in any order, even though they correspond to sequentially ordered actions. Also, the decision whether or not to break a page above a certain type of heading should be specifiable at any point in the definition.

Such actions do not really correspond to design decisions, rather they are the specific incarnations of general parameters and switches. Thus, it would be a valid approach to style design to offer the person implementing the style a small number of basic constructs (these could for instance be headings, lists, and page layouts), each of which has a 'checklist' of parameters and switches controlling the final layout.

Abandoning sequentiality, and making most specifications optional with default values, is then a sensible way to facilitate TEX programming. An implementation of these demands could take the form of lists of keyword-value pairs. Such lists can be presented in extremely simple syntax, and as matching is performed by keyword instead of by position, such an approach would meet some of the criticism of algorithmic specification above.

As an example, the layout of unnumbered section headings could be given in Lisp 'property list' syntax as,

```
\defineheading:section
    (white_above 6pt plus 2pt minus 1pt)
    (white_below 6pt plus 2pt minus 1pt)
    (caption (size 12pt) (style bold)
             (shape ragged_right))
```

but any other syntax is just as valid. It is advisable, however, to steer clear of the idiosyncrasies of the pure TEX syntax.

Metaformats

Checklists may capture a large part of the variation in basic constructs, but for any set of parameters there will be a layout that eludes classification in terms of these parameters. Thus, it seems inevitable that a style implementer needs to do some programming. However, it is possible to make a very smooth transition between marking a checklist and programming macros. For this, we need the distinction between formats and 'metaformats'.

Let us denote by the term 'format' any collection of macros that gives the end user commands of a higher level than those of pure TEX. By 'metaformat,' we will mean a format such that the commands for the end user are in majority not defined in the format. Metaformats offer the tools with which a style implementer constructs the commands for the end user.

In a restricted sense, the LATEX format is a metaformat. A good example here is the `\@startsection` macro, which is basic to LATEX, and in terms of which commands like `\section` are defined in the style files. The parameters of this command are mainly numeric parameters determining the layout. One parameter functions as a switch.

As another example, the `\list` command, which is the basis for various environments in LATEX, offers the style designer the possibility for programmable extension. Such extensions are specified by passing a piece of TEX code as the second parameter, that is, LATEX does not really provide the style implementer with a separate syntactic level.

Calling LATEX a 'parameterized metaformat,' we can also envision 'programmable metaformats.' There is no objection against implementing a new programming language in TEX which would be easier to learn and to use.

The challenge is then to find primitives that allow formulation in a simple syntax and that are sufficiently powerful for producing a wide range of layouts.

By any other name ...

One choice for the primitives of a metaformat is immediately clear: the boxes and glue of TEX itself. Any TEX programmer knows that you can do all typesetting with boxes and glue, so why not give them to a style designer? The problem of course is how to simplify their declaration. It is here that the writer of the metaformat takes certain decisions.

Consider, as an example, section headings such as they appear in the LATEX article style. These

1. Section title

1.1. Subsection title

take the form (but not the implementation) of two boxes on the same line: one containing the number, the other containing the text. Also the headings of the 'artikel' style can be described thus. But the

1. Section title
1.1. Subsection title

box of the number then has a prescribed width. In both cases, however, the sum of the width is the text width. We can therefore imagine that these two layouts were specified as

```
\defineheading:section
    (inline
       ((value sectionnumber)
        (spaces 2))
       (title))
```

for the standard LaTeX heading, and

```
\defineheading:section
    (inline
       ((value sectionnumber)
        (FillUpTo labelwidth))
       (title))
```

Here the metacommand 'inline' is just an `\hbox to \hsize` in disguise. It takes all boxes and sets them at natural or specified width, and the width of the box containing the actual heading is calculated to fill the remainder of the text width.

Headlines and footlines provide a nice example of how a programmable metaformat can make intelligent use of TeX's glue. Like section headings, footlines are a disguised `\hbox to \hsize`. A footline with just a left aligned page number can be specified like

```
(footline (value pagenumber))
```

where the format supplies a trailing `\hfil` to prevent an underfull box. Cases where the number should be right aligned can be specified as

```
(footline (whitespace fillerup)
          (value pagenumber))
```

where the 'whitespace' is an `\hfill`, squashing the trailing `\hfil` at the end of the line.

A syntax and instruction set, such as sketched in these examples, tax the programming capabilities of the format designer, but not those of the style designer. In effect, the format designer implements a new programming language on top of TeX with a simple syntax, a small instruction set, and at the same time, sufficient generality to produce a wide range of layouts.

Obviously, a smaller instruction set makes learning the format easier. Another advantage is less immediately clear: it reduces the chance of errors. More compact instructions will likely have a more defined function; so on the one hand the style implementer need not specify those actions that must be taken anyway, and on the other hand the format can perform some consistency checking on the intentions of its user.

Conclusion

We have argued the need for a 'programmable metaformat:' a format that introduces a new syntactic level in TeX for document style designers. Such a level should probably have a different syntax from pure TeX, and it should contain a relatively small instruction set in which the actual user macros are written. We have indicated that elements of such a format can, to a large extent, be captured in 'checklists.' Others can take the form of intelligently disguised TeX primitives. We believe that a format incorporating these principles is possible, and that it can be taught to people with little knowledge of TeX.

As an illustration of these ideas we present the style definition of this article in the 'Lollipop' format.

Example

The following piece of code is the style definition for this article given in the 'Lollipop' format.

```
\typeface:Computer
\fontsize:10 \fontstyle:roman

%declare \parindent
\distance:indentation=20pt
%white space around text elements
\distance:whiteline=6pt plus 2pt
         minus 2pt
\distance:leadingwhite=whiteline
\distance:trailingwite=0pt

%lots of defaults used here
\defineheading:section
         fontstyle:bold stop

%very simple page layout
\definelayout:twocolumn
    height:22.5cm width:16.5cm
    band:start column
         whitespace:0.6cm
         column
         band:end
    stop
\twocolumn %install output routine

%paragraph shape
```

```
\defineparagraph:flushright indent:yes
    rightjustified:yes stop
\flushright %install paragraph shape

%bibliography
\definelist:literature counter:1
    item:left
         litteral:[ value:itemnumber
         litteral:]
    item:end stop
\externalreferences:yes
```

Bibliography

Braams, Johannes, Victor Eijkhout, and Nico Poppelier, "The development of national LaTeX styles," *TUGboat*, Vol 10(4), 1989.

Grif manual / Les languages de Gipsi, Gipsi 1989

Knuth, Donald E. *The TeX book*, Addison-Wesley Publishing Company, 1984.

Lamport, Leslie. *LaTeX, a document preparation system*, Addison-Wesley Publishing Company, 1986.

Improving the Æsthetics of Mixed-Font Documents

Philip Taylor
Royal Holloway and Bedford New College, University of London, Egham Hill, Egham, Surrey, United Kingdom.
Tel: +44 784 443172 Internet: P.Taylor@Vax.Rhbnc.Ac.Uk

Abstract

The author has recently been involved in typesetting a 700 page book in a mixture of PostScript and Computer Modern fonts. This paper describes some of the techniques developed to improve the æsthetics of the book, with particular reference to the techniques necessary to integrate fonts from two distinct families. Reference is also made to techniques for improving the visual consistency of the book, such as direct editing of PostScript inserts and dynamic calculation of table widths.

During the period 1987 – 1989, I spent considerable time collaborating with Professor I.L. and Professor R.S. Gibson of the Universities of Waterloo and Guelph respectively on the typesetting of a major new work on the principles of nutritional assessment. The book was to be published by an internationally renowned publishing house, and the design was, to a certain extent, dictated by house standards, but the publishers allowed us considerable flexibility in the implementation of those standards, and the resulting work owes as much to original design as it does to house styles.

The trim and text dimensions of the book were fixed by the publisher as 36 pc × 55.5 pc and 27 pc × 45 pc respectively, and it was felt that these dimensions dictated a single-column style (there are also many unresolved difficulties in the implementation of multi-column styles, where floating and non-floating objects may span an arbitrary number of columns). We decided that the text font should be Adobe Times-Roman, with its variants Times-Italic, Times-Bold and Times-BoldItalic. The book was by no means maths-free, and rather than attempt to re-implement the maths-oriented definitions of `plain.tex` (or `lplain.tex`) we decided to retain the use of the Computer Modern Symbol, Maths Italic and Maths Extension fonts. Greek majuscules and minuscules were taken from the Adobe Symbol font, as upright rather than italic minuscules were desired.

Having chosen the fonts to be used, we next needed to decide on the most appropriate size and leading; this, to my mind at least, led to one of the most significant design decisions of the entire work. We experimented with ten, eleven and twelve point Times-Roman, on a variety of leadings, and felt that none of these offered the perfect combination of characters per line and lines per page. Ten-point Times-Roman had too many characters per line, eleven-point had just about the right number, and twelve-point had too few, while 10/12 point offered the ideal number of lines per page; 11/12 was too cramped vertically, 11/12.5 was about right but gave too few lines per page, and of course 11/13.6 and 12/14.5 gave even fewer. It seemed that we wanted the impossible: a font that was as wide as an eleven-point Times-Roman which could nonetheless be set on a twelve-point leading.

At this point, inspiration struck: as we were using PostScript fonts, on a PostScript output device (a Linotronic 300), there was no reason why we should not scale the fonts *anamorphically* — in other words, shrink the font vertically while leaving it unchanged horizontally. A few experiments (and a 'phone call to John Gourlay of ArborText, to find out how to retain the anamorphic scaling when their standard prelude insisted on turning it off again) later, we had a range of anamorphically scaled fonts available for proofing. It was immediately obvious that anamorphic scaling had to be subtle to be effective: 22/25 was awful; 23/25 OK, and 24/25 excellent. An anamorphic scaling of 24/25, applied to an 11/12.5 point font/leading, yielded a font that was a little over 10.5 points vertically, while of course still being eleven points horizontally, on a twelve-point leading. This gave us the ideal number of characters per line, lines per page, and adequate leading: 10.5/12 it was to be.

Of course, we still had to have the publisher's approval to use this font, so we carefully prepared sample pages of each of the potentially acceptable font/leading combinations, together with a sample page in our new anamorphic 10.5/12 Times-Roman, and sent these off for approval. We were overjoyed when the publisher not only selected 10.5/12, but pronounced it "one of the best examples of Times Roman which he had seen."

Anamorphic scaling of PostScript fonts was one thing, but anamorphic scaling of Computer Modern was another. We were not using Graham Toal's PostScript outlines of Computer Modern, but 1270 dpi bitmaps (`pk` files); how were we to re-scale these? The solution was simplicity itself: rather than scale the fonts individually, we would scale the whole document. Fonts which are presented as PostScript bitmaps are subject to exactly the same set of manipulations as any other PostScript object, and thus can be anamorphically scaled. Of course, scaling of a Computer Modern font bitmap undoes at a stroke all the hard work that METAFONT and the font designer have together achieved, in terms of the exact pattern of pixels output by METAFONT for a particular character, design size, magnification and mode definition, but the amazing thing is, it just doesn't seem to matter! Admittedly the scaled Computer Modern fonts form only a small proportion of the total glyphs typeset, but it would take a very careful and trained eye to spot that they are not produced from the canonical bitmaps.

At this point, it is worth digressing slightly and discussing the implementation language(s). Of the two of us actually typesetting (rather then writing) the book, one (Ian Gibson) is a confirmed though flexible LaTeX user, while the other (the present author) is a totally bigotted Plain TeX addict, who will admit to no merit whatsoever in any of the so-called higher-level formats — needless to say, this lead to some interesting discussions... Nonetheless, it was amicably agreed that the book would be set in LaTeX, but that any changes to the default style files (and, as it turned out, to `lplain.tex`, `lfonts.tex` and `latex.tex`) would be implemented through the medium of Plain TeX. In practice, this led to very few problems. Just occasionally an obscure look-ahead involving `\futurelet` and/or `\afterassignment` failed unexpectedly, and this lead to a few problems in debugging the code, but in general Plain TeX and LaTeX co-existed satisfactorily without any real difficulties. PostScript modifications were, of course, implemented in PostScript.

The book also included a very large number of illustrations and graphs, neither of which are TeX's *forte*, and these were prepared using either Harvard Graphics or Adobe Illustrator. These packages are both capable of producing (encapsulated) PostScript output, and of using the Adobe font set. In practice, the graphics packages were configured to use Adobe Helvetica, and captions typeset in TeX were similarly set in Helvetica. The mechanism by which these figures were incorporated is (very) loosely based on a macro which was first published as `\PrintChart` by ArborText in the documentation accompanying DVILASER/PS; the most recent version of the revised macro (`\PostScript`) is attached as Appendix A. The macro allows the coordinates of the desired portion of the original figure (with respect to the lower left corner of the page) to be specified, and also allows the final height and width to be specified — this does, of course, allow the aspect ratio of the figure to be altered during insertion, but this feature is rarely desirable or used. More important, however, is that it allows the inserted figure to be scaled up or down from the original, but this can have an adverse effect on the æsthetics and consistency of the book, for the following reason: the originals will almost certainly have been prepared without allowance for the possibility of subsequent scaling, and will typically have a consistent set of line sizes; after scaling, lines which were originally the same size (we chose 0.4 pt to match the default rules of TeX) would have been of varying size, and it was therefore necessary to edit the PostScript files produced by Adobe Illustrator and Harvard Graphics to pre-scale the line thicknesses to allow for subsequent re-scaling. The font sizes similarly had to be pre-scaled by editing. This caused one amusing problem: the editor used by Ian to make these changes to the PostScript files had an uncontrollable urge to write a `<control-Z>` at the end of each file, and a special PostScript macro definition for `<control-Z>` had to be written, to prevent otherwise insuperable difficulties.

Returning to the question of fonts, the next issue to be addressed was visual font matching. Remember that we were using Adobe Times-Roman for the text, Computer Modern Maths Italic, Symbol and Extension for mathematics, and Adobe Symbol for Greek majuscules, minuscules and various other characters. The first problem was simply one of achieving consistent perceived size. While Adobe Times-Roman (scaled `\magstephalf`) and Computer Modern Maths Italic and Symbol tenpoint (scaled `\magstephalf`) are very well matched for size, we soon discovered that the Adobe Symbol font, similarly scaled, was far too large. The question was, by how much to reduce it? If design size was an inadequate criterion, perhaps the x-height of the font would be a better metric. We tried again, this time adjusting the scaling of Adobe Symbol such that its x-height (as measured by `\fontdimen␣5`) exactly matched the x-height of Adobe Times-Roman; *still* the fonts appeared visually to be of different sizes. Finally we tried matching the heights of a pair of individual characters from the two fonts: as the μ from Symbol was the most common of the Greek minuscules, we matched its exact height to that of the character from the Times-Roman font with which it most frequently occurred in juxtaposition (lower-case g), and this technique was found to give the best visual match. The code for dynamic font matching is given in Appendix B.

A further problem manifested itself: in designing the Computer Modern fonts, Knuth had obviously been influenced by the requirements of mathematical typesetting. In particular, the height of maths operators such as plus, minus, greater-than, less-than, etc. had been optimised to match the heights of the lower-case letters between which they were most likely to occur. In a scientific (but *not mathematical*) text, maths operators tend to occur between numbers, upper-case letters and lower-case letters with approximately equal frequency, and when seen between numbers or upper-case letters, the maths operators tend to look rather too low. Barbara Beeton was consulted on this problem, and suggested that one obvious solution would be to use METAFONT to generate alternative glyphs for the maths operators, raising them by an appropriate amount from the baseline. Whilst this was undoubtedly the right approach, it was potentially very time-consuming, and a pure TeX solution was sought. We also wished to avoid making changes to the text of the document, and thus a solution involving the replacement of all occurrences of `+` by `\+`, `-` by `\-`, etc., together with re-definitions of `\+`, `\-`, etc., was ruled out. This left us with only one apparent option: the maths operators had to be made active and defined as macros, with their replacement text being that necessary to raise the original operator by an appropriate amount, while otherwise retaining the operator's original semantics including its maths class (`\mathord`, `\mathop`, `\mathbin`, `\mathrel`, `\mathopen`, `\mathclose`, `\mathpunct`). Rather than write an individual definition for each maths operator, we developed a single macro `\mathsraise`, which, when applied to a maths operator, made it active in maths mode and re-defined it in terms of its original meaning; we similarly developed an alternative form, `\raisemaths`, which had an analogous effect when applied to named maths operators such as `\pm`, `\ge`, `\le` and `\times`. We could then simply list the operators to be raised, as in:

```
\mathsraise <
\mathsraise >   %  also +, -, = and /
\raisemaths \pm
\raisemaths \ge %  also \le and \times
```

Thereafter each operator so listed had its new meaning in maths mode only, retaining its original meaning outside of maths mode. With hindsight, it might have been desirable to be able to specify individually the amount by which each operator was to be raised, but in practice a common factor of 0.15 `ex` seemed both necessary and sufficient. The `\mathsraise` macros are given in Appendix C.

Many other changes proved desirable in the production of this book; a new 'the-book' style was derived by extensive modification from the standard 'book' style for LaTeX, and PostScript-specific versions of `lplain.tex`, `lfonts.tex` and `latex.tex` were generated (called `pslplain`, `pslfonts` and `pslatex`, respectively). `PSlfonts`, in particular, was considerably modified from its progenitor, firstly to change all possible references to Computer Modern fonts to the nearest PostScript equivalent, but subsequently to pre-load far fewer fonts, as the implementation placed severe restrictions on the number of fonts that could be concurrently loaded. Special `\struts` were defined for use in ruled tables, to ensure adequate vertical white space above and below each row, and automatic table sizing was implemented, which ensured that any table which exceeded 0.85 `\textwidth` was automatically stretched to occupy `\textwidth`, thereby improving the visual consistency of the book. This last technique required an iterative approach to circumvent the limitation imposed by `\multispan`, whereby all additional space is dumped in the last column spanned. A new version of `\caption` was implemented to make use of information saved by the `\PostScript` and `\SetTable` macros; this enabled us to set captions to the same width as the figure or table to which they referred.

All of this took time — considerable time — but in the opinion of the author it was time well spent. The finished book bears neither the traditional TeX stamp of Computer Modern, nor the traditional LaTeX stamp of overly large fonts and excessive white space. It more closely resembles a traditionally typeset book, and this was, in essence, the underlying aim: the typography and design were intended to be totally transparent to the reader, serving solely to draw his or her attention to the content. We believe that we have achieved this aim.

Appendix A: The \PostScript macro, for the inclusion of PostScript figures

```
%
%       A very arcane TeX/PostScript macro
%       ==================================
%

\newif \ifoutline
\newdimen \border

\def \PostScript file:#1;
                 xmin:#2; ymin:#3; xmax:#4; ymax:#5;
                 width:#6; height:#7; options:#8%
{%
        \outlinefalse
        \border = 0 pc
        \setbox 0 = \hbox
               {\def    \file   {#1 }%
                \def    \xmin   {#2 }%
                \def    \ymin   {#3 }%
                \def    \xmax   {#4 }%
                \def    \ymax   {#5 }%
                \def    \width  {#6 }%
                \def    \height {#7 }%
                \def    \options{#8}%
                \def    \Xmean  {\dimen 0 }%
                \def    \Ymean  {\dimen 1 }%
                \def    \Xsize  {\dimen 2 }%
                \def    \Ysize  {\dimen 3 }%
                \def    \Width  {\dimen 4 }%
                \def    \Height {\dimen 5 }%
                \def    \0{\count 0 }%
                \def    \1{\count 1 }%
                \def    \2{\count 2 }%
                \def    \3{\count 3 }%
                \def    \4{\count 4 }%
                \def    \5{\count 5 }%
                \def    \6{\count 6 }%
%
% The options macros
%
                \def    \box {\outlinetrue}%
                \def    \outline {\outlinetrue}%
                \options
                \message {Border is \the \border}%
                \6 = \border
                \ifoutline
                        \message {Figure will be enclosed in a box}%
                        \def \rule
                               {\ifhmode
                                        \vrule
                                \else   \ifvmode
                                                \hrule
                                        \else
                                \message {Are you sure you
                                          should be trying to
                                          draw rules in this
                                          mode ?}%
                        \fi
                \fi
               }%
\else
        \message {Figure will not be enclosed in a box}%
        \def \rule
               {\ifhmode
                        \vrule width 0 sp
                \else   \ifvmode
                                \hrule width 0 sp
                        \else
                                \message {Are you sure you
                                          should be trying to
                                          draw rules in this
                                          mode ?}%
                        \fi
                \fi
               }%
\fi
\Xmean = \xmin
\advance \Xmean by \xmax
\divide \Xmean by 2
\0 = \Xmean
\Ymean = \ymin
\advance \Ymean by \ymax
\divide \Ymean by 2
\1 = \Ymean
\Xsize = \xmax
\advance \Xsize by -\xmin
\2 = \Xsize
\Ysize = \ymax
\advance \Ysize by -\ymin
\3 = \Ysize
\Width = \width
\4 = \Width
\Height = \height
\5 = \Height
\def    \xmean  {\the \0 }%
\def    \ymean  {\the \1 }%
\def    \xsize  {\the \2 }%
\def    \ysize  {\the \3 }%
\def    \width  {\the \4 }%
\def    \height {\the \5 }%
\def    \border {\the \6 }%
\vbox to \Height
       {\vss \rule
        \hbox to \Width
               {\hss
                \rule height 0.5 \Height depth 0.5 \Height
                \hskip 0.5 \Width
```

```
                                \special
                                       {ps::[asis,begin]
                                        0 SPB
                                        /ChartCheckPoint save def
                                        Xpos Ypos translate
                                        \width \xsize div
                                        \height \ysize div
                                         scale
                                        /sptobp
                                           {65536 div 72 mul 72.27 div} def
                                        \xmean sptobp neg
                                        \ymean sptobp neg
                                         translate
                                         newpath
                                        /border {\border sptobp} def
                                        \xmean sptobp \ymean sptobp moveto
                                        \xsize sptobp 2 div neg border add
                                        \ysize sptobp 2 div neg border add
                                         rmoveto
                                        \xsize sptobp border 2 mul sub 0 rlineto
                                         0 \ysize sptobp border 2 mul sub
                                                  rlineto
                                        \xsize sptobp border 2 mul sub neg 0
                                                  rlineto
                                         closepath
                                         clip
                                         newpath
                                         255 dict begin
                                        /showpage {} def
                                        }%
                                \special {ps:plotfile \file asis}%
                                \special
                                        {ps::[asis,end]
                                         end
                                         ChartCheckPoint restore
                                         0 SPE
                                        }%
                                \hskip 0.5 \Width
                                \rule height 0.5 \Height depth 0.5 \Height
                                \hss
                               }%
                        \rule \vss
                       }%
        }%
        \message {The dimensions of the figure are:
                        {ht: \the \ht 0}
                        {dp: \the \dp 0}
                        {wd: \the \wd 0}}%
        \copy 0
}
```

Appendix B: The \LoadFont macro, for the dynamic matching of font sizes

```
\newfam \symfam
\newcount \symfactor
\newcount \xheight
\newcount \fudgefactor
\newcount \scalefactor
\newcount \canonicalxheight

\def \loadfont #1=#2 [#3]
       {
        \fudgefactor = 10
        \scalefactor = 10
        \setbox 0 = \hbox {\the #3 0 g}
        \canonicalxheight = \ht 0
        \symfactor = 1000
        \font #1 = #2 scaled \symfactor
        \setbox 2 = \hbox {#1 \char '155}
        \xheight = \ht 2
        \ifnum \xheight = 0
        \then
                \relax
        \else
                \multiply \symfactor by \canonicalxheight
                \divide \symfactor by \xheight
                \multiply \symfactor by \fudgefactor
                \divide \symfactor by \scalefactor
        \fi
        \message {Loading font #1 = #2 scaled \the \symfactor}
        \font #1 = #2 scaled \symfactor
       }

\viiipt

        \loadfont \viiisymtext = pssym [\textfont]
        \loadfont \viiisymscript = pssym [\scriptfont]
        \loadfont \viiisymscriptscript = pssym [\scriptscriptfont]

\ixpt

        \loadfont \ixsymtext = pssym [\textfont]
        \loadfont \ixsymscript = pssym [\scriptfont]
        \loadfont \ixsymscriptscript = pssym [\scriptscriptfont]

\xipt

        \loadfont \xisymtext = pssym [\textfont]
        \loadfont \xisymscript = pssym [\scriptfont]
        \loadfont \xisymscriptscript = pssym [\scriptscriptfont]

\@addfontinfo \@viiipt {\textfont \symfam = \viiisymtext
                \scriptfont \symfam = \viiisymscript
                        \scriptscriptfont \symfam = \viiisymscriptscript}
```

```
\@addfontinfo \@ixpt {\textfont \symfam = \ixsymtext
                \scriptfont \symfam = \ixsymscript
                        \scriptscriptfont \symfam = \ixsymscriptscript}

\@addfontinfo \@xipt {\textfont \symfam = \xisymtext
                \scriptfont \symfam = \xisymscript
                        \scriptscriptfont \symfam = \xisymscriptscript}
```

Appendix C: The \mathsraise macros, for the height adjustment of maths operators

```
\catcode `\@ = 11               % make `@' a letter ...

\gdef \mathsraise               % define the main procedure with no parameters
{%                              % as we have to change catcodes immediately
    \begingroup                 % will be closed as first thing \m@thraise does
    \def \@ctivateall           % define a macro to make all characters
    {%                          % except <space> active
        \count 0 = 0                                    % for i := 0 to 127
        \loop   \catcode \count 0 = \active             % \catcode[i] := \active
        \ifnum  \count 0 < 127 \advance \count 0 by 1   % endfor
        \repeat \catcode 32 = 10\relax                  % \catcode <space> = 10
    }%                          % \@ctivateall now defined
    \def \m@thraise ##1%        % this procedure does all the real work, raising
    {%                          % the character while retaining its maths class
        \endgroup               % restore normal catcodes a.s.a.p.
        \mathchardef \t@mp = \mathcode `##1%            % get the mathcode of #1
        \mathcode `##1 = 32768\relax                    % make it maths-active
        \def \@ftermathchar ####1"{"}%                  % \mathchar" -> "
        \def \m@thselect   ####1% a procedure to select maths classes
        {%                      % the parameter is normally a hex string
                \count 0 = ####1\relax   % get the mathcode into \count 0
                \divide \count0 by 4096  % only interested in the top three bits
                \ifcase \count0 \mathord % use these to select the maths class
                        \or     \mathop
                        \or     \mathbin
                        \or     \mathrel
                        \or     \mathopen
                        \or     \mathclose
                        \or     \mathpunct
                        \or     \mathord
                \fi
        }%                      % \m@thselect now defined
        \edef \t@mp {\expandafter \@ftermathchar \meaning \t@mp}% gets hex value
        \edef ##1%              % define parameter-1 to yield the same character
        {%                      % raised by 0.15ex but retaining its maths class
                \noexpand \m@thselect {\t@mp}%
                {\mathchoice
                 {\raise 0.15ex \hbox {$\displaystyle \mathchar\t@mp$}}%
                 {\raise 0.15ex \hbox {$\textstyle \mathchar\t@mp$}}%
                 {\raise 0.15ex \hbox {$\scriptstyle \mathchar\t@mp$}}%
                 {\raise 0.15ex \hbox {$\scriptscriptstyle \mathchar\t@mp$}}}%
        }%                      % parameter-1 now defined
    }%
    \@ctivateall \m@thraise     % make everything bar <space> active, then
                                % get and process the next character
}%                              % \mathsraise now defined

\def \raisemaths #1%            % this procedure does all the real work, raising
{%                              % the character while retaining its maths class
   \def \aftermathchar ##1"{"}% % \mathchar" -> "
   \def \mathselect   ##1%      % a procedure to select maths classes
   {%                           % the parameter is normally a hex string
```

```
          \count 0 = ##1\relax     % get the mathcode into \count 0
          \divide \count0 by 4096  % only interested in the top three bits
          \ifcase \count0 \mathord % use these to select the maths class
          \or     \mathop
          \or     \mathbin
          \or     \mathrel
          \or     \mathopen
          \or     \mathclose
          \or     \mathpunct
          \or     \mathord
          \fi
      }%                           % \mathselect now defined
      \edef #1{\expandafter \aftermathchar \meaning #1}% gets hex value
      \edef #1%                    % define parameter-1 to yield the same character
      {%                           % raised by 0.15ex but retaining its maths class
            \noexpand \mathselect {#1}%
            {\mathchoice
             {\raise 0.15ex \hbox {$\displaystyle \mathchar#1$}}%
             {\raise 0.15ex \hbox {$\textstyle \mathchar#1$}}%
             {\raise 0.15ex \hbox {$\scriptstyle \mathchar#1$}}%
             {\raise 0.15ex \hbox {$\scriptscriptstyle \mathchar#1$}}}%
      }%                           % parameter-1 now defined
   }%                              % \raisemaths now defined

   \mathsraise <
   \mathsraise >
   \mathsraise +
   \mathsraise -
   \mathsraise =
   \mathsraise /

   \raisemaths \pm
   \raisemaths \ge
   \raisemaths \le
   \raisemaths \times

   \end % of text
```

ArchiTEX A Preliminary International Page Pattern Maker

Alan E. Wittbecker
Digital Equipment, ZKO1-2/C21, 110 Spit Brook Road, Nashua, NH 03062–2698
603-881-0042. Internet: wittbecker@vaxuum.dec.com

Abstract

TEX was created in the tradition of typesetting printed pages for bound volumes. For hundreds of years, printers considered the page as a block of type surrounded by margins. As adequate margins were required for binding, practical considerations became aesthetic needs applied by strict formulas. Earlier concepts of pages, such as certain Egyptian hieroglyphs, Roman scrolls, and medieval illuminated manuscripts, considered the entire page area as a pattern, to be filled or not, depending on numerous aesthetic, practical, or economic criteria.

New methods of production, such as photomechanical plates, or types of display, such as video terminals, mean that page design and formatting need not be limited by traditional conventions. Instead, pages can be considered for loose-leaf books (especially with on-demand printing), where text, graphics, and space are arranged over multi-page spreads. Or, in another instance, pages can be treated as small blocks, called up and presented in a multi-window environment on a video terminal.

The limitations of the printed page are not intrinsic to TEX. TEX macros can be modified to present text, graphics, and space in blocks on a complete or partial page block (that can be combined or recombined). TEX macros can also free reading order from European conventions. The same macro, for instance, can produce lettered lists using any alphabet in a cartesian coordinate system. A prototype markup language, ArchiTEX, uses select TEX macros to produce flexible, modular page patterns for a variety of output devices.

The evolution of print from rock faces to liquid crystal displays can be characterized in four formats.

- Archaic (rock, bone)
- Classic (scroll, book)
- Modern (newspaper, magazine)
- Hypernian (computer, laser)

Typographic presentations evolved from the process of inscription, which has been limited by tools (such as chisel, pen, laser) and media (such as stone, paper, cathode ray tube). Each format can be characterized by media, speed of transmission, and ubiquity. Each format is also part of a historical sequence as well, and its particular uniqueness is the result of the interaction of unique characteristics. After ten thousand years or more, some archaic peoples still put expressions on rock faces; even modern peoples do so on the walls of buildings in cities—it's called graffiti or art. Books have been printed on clay, wood, leather, and paper for over three thousand years. News format, in newspapers and magazines, for several hundred. Hypermedia, coordinated by computer, for two decades. All four formats exist concurrently.

Traditional Page Shapes

Classic. In Egypt, the pith of papyrus stalks were cut into thin strips, which were dried and then laid out in a row with edges overlapping. Another row was laid crosswise, then the two layers were moistened and pounded together. The sheet was sized, dried and glazed; it was supple and flexible—very suitable for being rolled. Sheets could then be glued together side-by-side to form long sections (possibly several hundred feet long). Ordinarily sheets were 6-7 inches high. The side with horizontal strips is called the recto side; the vertical is called verso. The recto side was preferred for writing and became the inner side when rolled. The blank verso became the outside. Egyptian papyrus scrolls were written in hieratics (a simplified form of heiroglyphics, and later further simplified in demotics) in vertical columns separated by thin black lines; sometimes illustrations accompanied the text along the top or bottom of the scroll. As illustration assumed more importance it was placed in the text. Many scrolls had a horizontal frieze marked off from text by double ruled lines—for important scenes, the enlarged and extended completely into the text area the full height of the scroll (see Figure 1).

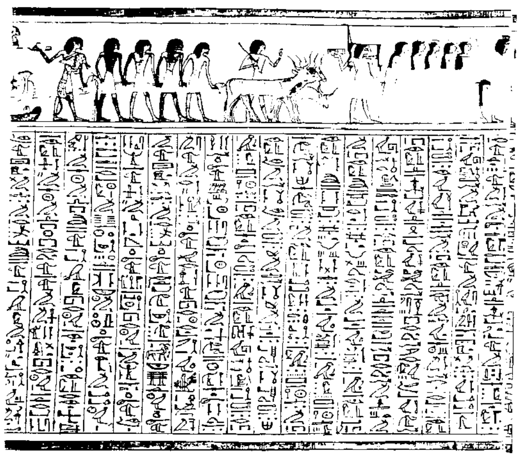


Figure 1. Egyptian *Book of the Dead*

Several hundred years later, the text was presented horizontally from right to left in columns, and the illustrations were put in the columns of text where they were related, sometimes narrower but rarely wider than the column. This format persisted through Greek and Roman manuscripts to medieval and modern books. Thus, columns of text, with illustrations, forms one of the oldest conventions of book formatting. A book was read by unrolling it. The division into columns effectively divided the book into "pages". The text began at the extreme right and moved from right to left. When the book was finished it had to be rerolled.

Although the Chinese (at about the same time) started writing on shell and bone, they made wooden tablets, and later, paper (a silk, cotton, leaf composite) pages into books. Writing began in the upper right corner and ran downward; lines moved from right to left.

Greek scrolls were smaller than Egyptian ones. Although Greek scrolls were also written in columns, characters were presented continuously, in capitals, without breaks between words. Illustrations were included within text (see Figure 2). Punctuation was usually nonexistent. Breaks in thought were sometimes indicated by an underlining stroke known as a *paragraphos* or by a small blank space. Although there was blank space at the beginning of a scroll, to protect the first part of the roll, titles, if any, appeared at the end of the text. As books were more widely read, top and bottom margins became wider to protect the text as the papyrus wore at the edges.

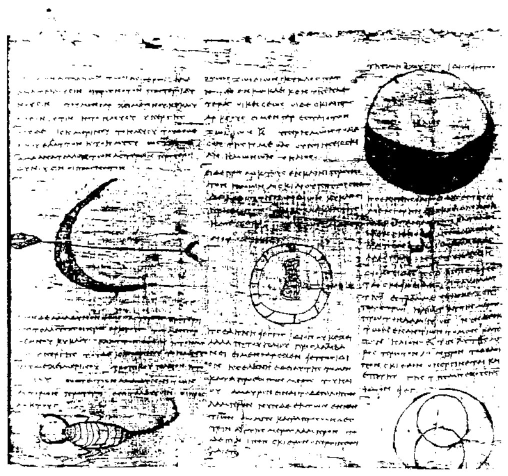


Figure 2.*Eudoxus* Roll

The Greeks copied papyrus books, but also used small tablets of wood for writing exercises. Two or more tablets were bound together sometimes by scribes or traders. After vellum (from animal hide) became available, the form of tablet books was adopted by the Romans—and called a codex. Vellum books were used for small and less expensive editions, especially since both sides could be used. The Egyptians and Greeks also adopted this form of book. Some codices were folded like fabric, but were awkward to read and refold (like maps). Soon, books were divided into simple folds tied together by string (as later book signatures were to be done). The format of such books was relatively small (the ratio of width to height was about 2:3). The practice of placing the title at the end was continued. By the fifth century A.D., the title was placed at the beginning as well; larger formats were also more common with wider margins, especially at the top and bottom. Foliation was introduced, usually on the front side of each sheet, since the page order could be confusing. Pagination came over a thousand years later as a printer's convenience (*pagina*=page, *folium*=sheet).

The Irish developed an angular, compressed minuscule hand (from the Roman half uncial form), which was disseminated by monks on their travels. After the monks cut vellum into sheets, they scratched guide lines with an awl or drew them with red ink or a graphite pencil to contain the characters (the first baselines and possibly grids); the distance between lines was marked in the margin. There was little standardization, otherwise. The scribe determined the column width (usually 2–3 inches with a half-inch gutter). Larger white areas were left as margins on the more splendid scrolls. Upper and lower margins, as well as gutters, were generous. The entire physical area was regarded as space to be filled, with words, pictures, illuminations, detailing, notes, and designs (see Figure 3).

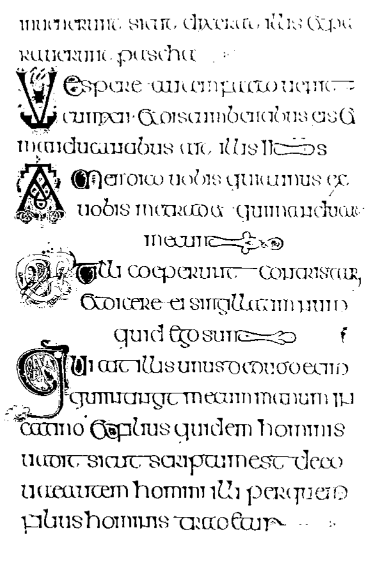


Figure 3. *Book of Kells*

Modern. Moveable type, in the form of a clay disk dating from 1500 B.C., was found in the ruins of the palace of Phaistos on Crete. Later, in China in A.D. 1041, Pi-Sheng developed type characters from hardened clay. Clay, however, did not hold up well under repeated impressions. By 1397 in Korea, type characters were being cast in bronze. Then, in 1440, Johann Gutenberg demonstrated for Europe the commercial possibilities of graphic reproduction with metal type. Gutenberg created a practical apparatus for casting the types. The small sizes of these rectangular blocks of metal type required a fine measuring system (the point system established by le Juene in 1737).

The first printers took manuscripts as their models; the arrangement of the page was followed closely (see Figure 4). Those features that could not be printed (initials or decorations) were added by hand. Pictorial production changed with technology, from drawings and woodcuts to lithograhic processes, photographic etchings, and scanning. In only fifty years, Aldus Manutius had transformed the Gothic type to a roman face and integrated illustration into a "perfect" book, the *Hypnertomachia Poliphili* (see Figure 5). Often, however, graphics dominated print. In many renaissance or rococo books, the illustrations stood out at the expense of the text. This was especially true of botanical or travel books. Later, print dominated graphics, often for economic reasons.

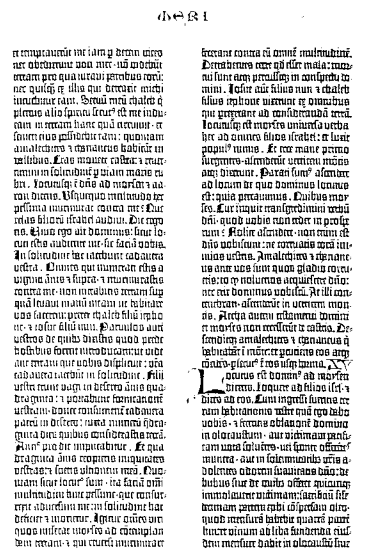
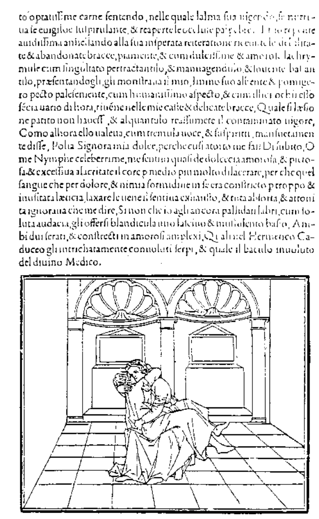


Figure 4. 42-line Gutenberg *Bible*



to optatiſſime carne ſentendo, nelle quale lalma ſua uigen[illegible]
ua ſe euigiloe ſuſpirulante, & reaperte le occluſe palpebre. [illegible]
auidiſſima amplexando alla ſua inſperata reiteratione [illegible]
te & abandonate bracce, piamente, & cum dulciſſime & amoroſe lachry-
mule cum ſingultato pertractantilo, & manuagendulo, & ſouente baſan
tilo, præſentandogli, gli monſtraua il mio, [illegible] ſuo al[illegible]ente & pomige-
ro pecto paleſeniente, cum humaniſſimo aſpecto, & cum illici ochii ello
ſecia uario di hora, riuéne nelle mie calide & delicate bracce, Quale ſi læſio
ne patito non haueſſ, & alquantulo reaſſumere il contaminato uigore,
Como alhora ello ualeua, cum tremula uoce, & ſuſpiritti, manſuetamen
te diſſe, Polia Signora mia dolce, perche coſi a torto me fai? Di ſubito, O
me Nymphe celeberrime, me ſentiua quaſi de dolcecia amoroſa, & pieto-
ſa & exceſſiua alacritate il core p medio piu molto dilacerare, per che quel
ſangue che per dolore, & nimia formidine in ſe era conſtricto p troppo &
inuſitata lætitia, laxare le uene il ſentiua exhauſto, & tuta abſorta, & attoni
ta ignoraua che me dire, Si non che io agli ancora pallidati labri, cum ſo-
luta audacia, gli offerſi blandicula uno ſaluio & mulſulento baſio, Am-
bi dui ſerati, & conſtrecti in amoroſi amplexi, Quali nel Hermetico Ca-
duceo gli intrichatamente conuoluti ſerpi, & quale il baculo inuoluto
del diuino Medico.

Figure 5. Manutius' *Hypnertomachia*

From Gutenberg's time, the page was regarded as a block of type surrounded by a frame of white space. The white space, margins, were left as printing and binding requirements for single pages. Later, printers put two or four pages on one sheet, which changed the ratio of inside to outside margins. As is so often with technology, practical convenience became interpreted as aesthetic theory, so that "pleasing" margin proportions are taught to apprentices still (as of 1971, when I was a printer's apprentice).

Modern Page Conventions. On a page, a group of elements is arranged in a hierarchy according to emphasis, which is often achieved by contrasts. Each element is related in the whole page.

The difference between the type page (text area) and page size (paper area) is the margin area. Deluxe and expensive editions of a book still tend to have wider margins. A traditional margin ratio, as the result of printing four-up (or more), is two units for the inside, three units for the top, four units for the outside, and six units for the bottom; this series (2/3/4/6) of column-to-margin ratios is based on the golden section (1/1.618) and can be found in some classic and medieval manuscripts.

There is a relatively small range of book sizes, centered around a comfortable human scale; few archaic or classic books were larger than nine by thirteen inches. The size of a book can be manipulated by type size and style; some typefaces have long alphabet lengths and may require extra leading between lines. The placement of headings, graphics, and white-space can add or subtract to the length of a book.

Ophthalmological studies show that reading is easier when letters are different from each other. Serif letters are easier to read than sans-serif; mixed upper and lower case easier than upper case. Size is also important. Very large or small sizes are difficult to read as text. Type size is recommended to be 10 to 12 points (for adults) for legibility. Small type may make word recognition more difficult; large type may make sentence recognition more difficult (by focusing perception on a small section of the whole). Weight (stroke thickness) affects legibility, as does kinds of face (italic, bold) and amount of leading.

The measure, or line length, is important to achieve a pleasant reading rhythm. Lines too short or too long may be tiresome to read. There are a number of rules to determine the appropriate line length: Mergenthaler Linotype suggest 40 characters at any size; Skillin et al. suggest a line range of 18 to 24 picas for 10-point type, with the ideal width being 22 picas; an alphabet length (which is the horizontal measure, in points, of the lower case alphabet set in type of one size and face), can be used to describe an optimum text width; it has been set to 1.6 alphabet lengths.

After the width of the page is determined, the length is chosen, usually intuitively, to be of good proportion. Skillin et al. note that a ratio of 1 to the square root of 2 (1.418) is pleasing.

Color combinations affect legibility. Although one government study showed that dark brown ink on light brown paper is easiest on the eye, black on white is traditional and considered the most legible combination. The typographic properties of text are partly defined by the white space on a page. The tone is set by the lightness or darkness of type.

Eye movement is influenced by the visual qualities of text. The text area can have different kinds of alignment (the way text lines up on a column or page): align left (also called flush left or raggedright), align center, align right, or justify (flush right and left). Ragged right provides visual cues that increase legibility. The rhythmic line breaks provide visual points of reference.

The basic findings of legibility research are still valid: interword spacing should be constant, lines of words should be optimum length (see previous formula), and interword space should be less than interline space. Often the readability and "friendliness" of a work can be increased with good margins, especially as long as books continue to be presented in bound paper form. Margins are not necessarily obselete for design purposes.

There is no one way of designing a document. Many ways are effective, and many are ineffective; the effectiveness depends largely on the purpose: communication, impression, shock value, instigation, inspiration, formality. Typography is usually serial and straight-line. The magazine, *BLAST*, put out by Wyndham Lewis (1914) broke most typographic conventions with its style. Each letter could have a separate baseline, at random or circular for instance (see Figure 6). Some books, notably those by Marshall McLuhan on the media of communication and the architectural notebooks of Paolo Soleri (see Figure 7), consciously violate the conventions of margin and typography, preferring to communicate as much between the lines. Changing styles influenced book design. The functionalism in the 1920s, with its principle that the practical is the aesthetically correct, divided text into irregular sections and favored unsymmetrical arrangements (to emphasize the important material). The elemental typography of that period was used for commercial purposes, like posters. It had the effect of freeing books from dependence on traditional typefaces. The democratization of typesetting, with new programs, may also free us from conventions, as many practitioners are ignorant of the history and standards.

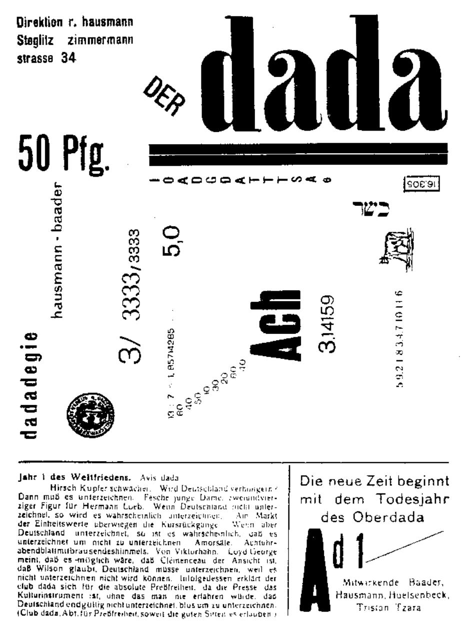


Figure 6. Hausmann's *Der DaDa*

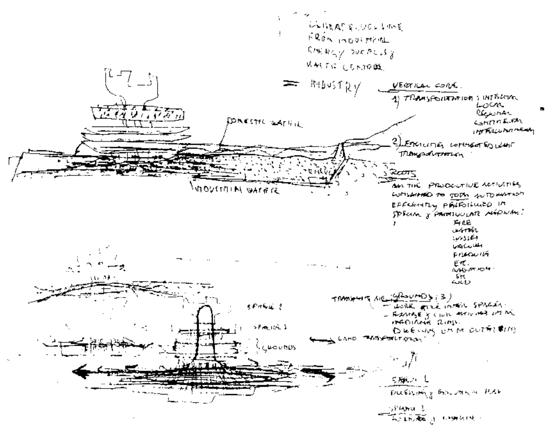


Figure 7. Soleri's Cosanti II

Technological Platforms.

Limitations Early concepts of "pages", such as certain Egyptian hieroglyphs, Roman scrolls, and medieval illuminated manuscripts, considered the entire page area as a pattern to be filled depending on numerous aesthetic, practical, or economic criteria. The illustrations of Newton and da Vinci were integrated into the text of their manuscripts (which were done by hand). Printing technology, however, found some copy, illustrations and tables for instance, difficult to integrate into text (and they charged higher prices for such penalty copy). Graphics was done separately from text; the positioning was often determined by page layout and so the graphics were often displaced from the text.

Traditionally, the size of paper (or tablets or papyrus) was determined by what was available and affordable. With typesetting, pages became larger, especially newspages, reference books, or folio (de luxe versions or coffee table) books—or smaller; Aldus Manutius started a printing business to issue critical editions ("pocketbooks") of classical authors. At that time, most books were done in a folio (two-up) or quarto (four-up) format, but Manutius printed in an octavo (eight-up) format with a reduced typeface based on cursive handwriting and adapted to the smaller size of pages (referred to as Aldine by Italians and italic by everyone else). Computer design seems to be limited by standard output devices, which use predominantly type A pages in North America or A4 in Europe. If the recommendations of legibility research are followed, then most designs are limited to two-columns or one column with very wide margins.

Potential Three basic structures for presenting information can be identified: words, tables, and graphics, according to Tufte (as well as combinations of the three). From early times, the author was responsible for putting thought into graphic and typographic form. With the advent of traditional typesetting, the form of the thought was redesigned within the limitations (physical and financial) of the technology. With computer technology, the form is often determined first and depends on the limitations of the program (as well as of the devices and finances).

New forms of display, such as video terminals, mean that page design and formatting need not be limited by traditional conventions. Instead, pages can be considered for loose-leaf books (especially with on-demand printing), where text, graphics, and space are arranged over multi-page spreads. Or, in another instance, pages can be treated as small blocks (that have pop-up tables and graphics blocks), called up and presented in a multi-window environment on a video terminal (as in Digital's on-line bookreader). or in a specific hypermedia context.

As computation is adapted more to human needs, interfaces may become friendlier, then transparent. In a virtual world, publications may be more accessible and manipulatable. Publications may regain a concern for overall pattern. Importantly, pattern recognition is one way of dealing with information overload.

Trends

Writing, arising from a phonetic alphabet, is an abstract kind of tool. The sound and letter elements of writing are divorced from meaning. *Printing is the mechanization of writing.* As writing became more mechanized, words became more like data, from the dead and living alike, with seemingly equal weight. Mechanization had an effect on the style of publication. With technological changes, the shape of publications, as well as attitudes towards them, has changed. Technology drove publications to a simpler appearance. Speed in production has often been detrimental to the quality and aesthetics of bookmaking. According to McLuhan, typing reduced expression from art to craft, from personal to the impersonal. It transcribed thought instead of expressing it. Linear and rectilinear layout of words in books was efficient and fast. Thought transposed into type became published, removed from the personal to the public sphere. Typing, or keyboarding, also changes the form of expression, favoring shorter sentences and more colloquial, less thoughtful expression.

Then, *the telegraph provided the electrification of writing*, according to McLuhan. And, the newspaper mirrored the form of telegraphic communication. The news format dislodged the book format as the format of perception. News format created daily snapshots across society, simultaneous, not sequential or historical. The news page set up many book pages on one, then continued them elsewhere, simulating simultaneity and nonlineality. It telegraphed information in blocks. The news format has influenced artists from Browning to Poe, Mallarme, Dickens, and James Joyce. The newspaper form structures awareness in its own *patterns*, from important to less, from front to back, from section to section. The application of the form of newspapers shaped meaning.

Finally, computer-based *hypertext is the electronification of literary connections* (or "nonsequential writing") according to Ted Nelson. It is just a small step from news forms to a hypermedia of computer data. The hypermedium seems to bypass the reading step and proceed directly to reference. Most hypertext systems rely on advanced hardware such as workstation windows and the possibility of connection nodes of different media, such as text, data, video, audio, graphics, and spreadsheets. Currently most output is designed for CRT screens, although printing is possible and other forms, such as holographic displays, are talked about. Hypermedia is a conceptual connection of different forms in an explicit medium.

Each of the four formats, archaic, classic, modern, and hypernian, has advantages and disadvantages, and each encourages a unique style of thought and expression. Inscription on rock had mystical overtones or unknown purposes. The sense of participation must have been high. Often, the rock surface was in caves and not easily accessible. Reading the inscriptions might have been a formal rite.

Books have their own romance, from wonderful textures to experienced imaginary dialogues. Book production is fast and standardized (the standards encourage ease of reading for the most part). The type is standardized and interchangeable, a precursor of manufacturing assembly lines. Books themselves are portable and familiar. They require a level of sophistication and participation of the imagination. Many books are not limited by the linearity of the presentation; double page spreads with graphics and notes can create a form of hypertext. Even sequential texts can produce a feeling of hypertext, where the reader makes a complex conceptual model of the subject (—this sort of involvement McLuhan regarded as characteristic of a hot medium). But, they can be static and simple—even dull.

Newspapers and magazines are often more visual and interesting. The news format presents the surfaces of many subjects. Their production is even faster than books. They are more flexible in format. But, the news format often is limited by the amount of commercial and dated material.

Computer-based hypertext can be comprehensive and dynamic in a way that no book or magazine ever can be. Hypertext is consciously nonlinear and pluralistic. A typical (planned) hypertext system offers a great diversity of content from a variety of media. The material can be shared interactively, altered constantly, tailored to individual needs, and displayed in sophisticated ways. But, its development is slow and incomplete; nodes are often too large and not object-oriented; the lack of standardization or traditional order can overwhelm users with information; the screen fonts are poor compared to print fonts; and, effective output is limited to screens.

Some trends, speed for instance, are easily distinguishable in the history of patterns. Reading a manuscript roll or codex was laborious and slow. The book speeded up reading as a form of communication, but was a more solitary activity. In general, simple books offer a linear perspective, with a single tone and attitude. Early manuscript culture, based on scarcity of materials, encouraged memorization. Then, manuscripts became designed for speed of reading. No more dallying through abbreviations and notations. The scope of historical awareness increased with printing, until with the newspapers and television, the past 24 hours are history (see Figure 8). Books were fast and convenient to reference. They could be read fast, then keep for reference. Hypermedia promise immediate referencing.

Material has become more accessible to readers (or viewers, interpreters, or writers). Readers have more control over what they read. The tools and the media (and possibly the ideas) have become more complex. But, technology has also opened the possibility of returning the cycle. Writing became remote from the authors, printing even

AN INTERNATIONAL DAILY NEWSPAPER WEDNESDAY, AUGUST 22, 1990

THE CHRISTIAN SCIENCE MONITOR

GULF CRISIS

Iraq and Jordan Count Cost of Gulf Conflict

Iraq's Arsenal: Terror, Sabotage, And Propaganda

By Peter Grier and George D. Moffett III

Jordan Mobilizes For Possible War In Persian Gulf

By Lamis Andoni

RELATED STORIES

- Turkish government halts food shipments to Iraq Page 3
- UN debates limits of sanctions against Iraq Page 3
- Bush's options are narrower as Saddam disperses hostages Page 6
- Economist sees drawn-out siege of Iraq would be worst scenario for the West Page 9

Rents Soar In Paris
Immigrant families face evictions, as the city's last working-class neighborhoods experience housing boom
4

New Jersey's Revenue Stew
In bowling alleys, hair salons and shops, a tax revolt is bubbling
7

Mexico Privatizes
President Salinas plans to sell state-run banks and telephone company
8

The Life And Art of A 'Titan'
Robert Snyder talks about his new film that uses Michelangelo's own letters and poems
10

The Next Frontier of Peace
Long-time activist couple sees post-cold-war shift to managing complex group conflicts
12

Tribute to Richard Stroui
The Monitor's veteran Washington reporter, who passed on recently, had the best leads in town
18

INSIDE

Figure 8. The *Monitor*

more so; now, technology is offering author control over many stages. Computer technology is not necessarily a limitation, if the author participates in the programming and design as well. New methods of production, such as photomechanical plates, need not adhere to the mechanical limitations of the hand press. New programs, like TEX, offer much potential. Other possibilities of shape and position of type areas can be pursued. The electric age hints at an organic metaphor from a mechanical one. Voice into print is possible; perhaps words may become three-dimensional and multi-layered when the output is hologrammatic. The words may reflect pauses and volume typographically. The form of print has been culturally biased. Possibly new technology can free us from the bias.

Requirements for an International Program

Any program that is to be used across cultural boundaries has to be able to express the direction and alphabets of the languages in use.

Direction. Most occidental languages move from left to right and top to bottom. Semitic languages, like Arabic, are written from right to left and move from top to bottom. Expressions may be useful from bottom to top, in either direction (see McLuhan's books).

Alphabets. A language often has its own unique alphabet, which is usually hard-coded into macros for alphabetical lists. Some languages use ideograms instead of phonograms to represent characters. A language representation should not be tied to a specific keyboard.

The Advantages of TEX

TEX is part of a long history in putting printed words on a page. Many of the conventions of book and journal production, such as paragraphs and columns, have roots in Mesopotamia and Egypt, Greece and Italy. Many TEX terms, from baseline to points, are based on the special nomenclature of typography.

The limitations of the printed page, however, are not intrinsic to TEX. TEX macros can be modified to present text, graphics, and space in blocks on a complete or partial page block (that can be combined or recombined). Regardless of the formal elements, TEX has the ability to put them on paper. TEX macros can free reading order from European conventions. A special TEX macro can produce lettered lists using any alphabet in a cartesian coordinate system.

Knuth and MacKay mention that TEX can handle documents that are read from left-to-right and top-to-bottom—English and other Western languages (MacKay, 1986, Knuth and MacKay, 1987). They also say, "If such documents are turned 90 degrees, they can also be read from top-to-bottom and right-to-left, as in Japan. Another 90 degree or 180 degree turn yields documents that are readable from right-to-left and bottom-to-top, or from bottom-to-top and left-to-right, in case a need for such conventions ever arises." They then describe a way to mold TEX to handle languages, like Hebrew or Arabic, which are right-to-left and top-to-bottom. They clarify the issues involved in mixed-direction documents and consider changes to TEX to extend it for bidirectional formatting. Digital uses TEX as the formatting engine for its documentation program, VAX Document, to take advantage of this directional capability.

At Digital, the Online Bookreader can display VAX Document (TEX formatted) files in an indefinite (but not infinite) series of windows. The book is chosen from an online library, then opened in a directory window, which includes contents, index, figure, table, and example icons (that can be contracted or expanded). Clicking on a chapter title in the contents, for example, results in a topic window opening. The text in the topic window contains hot spots and extensions that can be explicit or implicit. If a hot spot, a formal figure for instance, is clicked on, a subtopic or popup window opens with the figure. The TEX macros for the Bookreader have been modified for a unique screen environment. The concept of individual pages no longer exists, although each paragraph causes a page eject (so that TEX pages can be mapped to information chunks that make up topics). The fonts are larger for readibility on a screen. The Bookreader has experimental hooks for hypertext.

TEX has the capability to produce many kinds of designs. Often, authors and the programmers are ignorant of design, while designers are ignorant of programming. The vocabularies and styles are different. But, more often then not, neither group formally articulates complete design requirements; most designers mock-up typical pages, often leaving out many less common cases. TEX can be integrated into interfaces that solve this difficulty.

ArchiTEX as Page Pattern Maker

Requirements. A prototype markup language, ArchiTEX, uses select TEX macros to produce flexible, modular page patterns for a variety of output devices. These page patterns should be able to mimic book, newspaper, and hypermedia formats. The macros should be able to: use any alphabet for text as well as lettered lists; put text in any location on a page; describe page patterns for each individual page; and, continue text groups onto subsequent pages.

Samples.

Alphabets Creating an alphabet series for any alphabet is fairly simple. Referring to a clue in the *The TEXbook* (page 379) about stripping characters off a string, one need only define a string that can be stripped and recombined in numerous ways. The alphabet string needs to be defined first, with any special characters or accents (in Figure 9).

```
\def\bahbah{\\a\\b\\d\\g\\{jk}\\x}
```

Figure 9. Definition of Alphabet

An item in list can be selected by its position from the left and stripped off. Then, the alphabet length can be counted and assigned to a counter (see Figure 10).

Now, you can make a macro that offers the author a choice in numbering schemes, either a-z, then ab, ac, az—or aa, bb, zz, up to several hundred items (an English alphabet, with 26 letters results

```
\def\outofrange{ }% to prevent err
\def\pullet#1\of#2\to#3{%
    \def\#3{\outofrange\}%
    \long\def\\##1{\advance#1-1
    \ifnum\#1=0 \def#3{##1}\fi}#2}
\def\findalen#1\to#2{\#2=0
    \long\def\\##1{\advance#2 by1 }#1}
\findalen\alphasoup\to\alphalen
```

Figure 10. Alphabet Set-up

```
\def\alphacon{\global\advance\thingnum
              by 1
\ifnum \thingnum < \alphalen
\pullet\thingnum\of\alphasoup%
              \to\signifier
\unskip\signifier%
\else
%...
\}%
```

Figure 11. Alphabet Style Macro

in 703 items, before going to a star signifier). The code is in Figure 11.

The macro, alphacon, could then be put in a list macro for alphabetic lists (as in Figure 12) or combined with page numbers to indicate added pages. The complete macros are found in Appendix A.

ccc. Carneades
ddd. Protagoras
eee. Antiochus Posidonius

Figure 12. Alphabet Macro Output

Text location. Putting the text on any part of the page is a problem. Knuth gives a clue in *The TEXbook* (page 389—PICTEX and other programs use this clue for graphics); the labelling of points in a cartesian coordinate system can be expanded to place text in boxes with the top left corner of a box as the reference point. The first thing to do is set up the page to be the entire print area without margins. Then, set up units for moving, either using scaled points or baselineskips. And, define paragraph and page characteristics (in Figure 13).

The construction in Figure 14 starts in upper lefthand corner of the page, then moves down and right by units, then places a zero-width box at the specified x/y coordinates. This makes it easy to emulate a grid, a graphic device useful in the composition of pages (a recent development by Swiss designers, as a conscious application of imaginary lines to divide space, it imposes discipline on the

```
\hoffset=-60pt\voffset=-60pt% no margs
\hsize=612pt  \vsize=792pt  % h=51,v=66
\baselineskip=12pt          % standard
\newdimen\unit  \unit=\baselineskip
% ...
\def\pargoods{\raggedright
  \tolerance=5000\hyphenpenalty=-50
  \parindent=0pt\parskip=0pt\}% close
% ...
\long\def\go#1 #2 #3\stop{%
  \vbox to 0pt{\kern#2\unit%
    \hbox{\kern\#1\unit
    \vtop{\leftjump=#1\unit
          \downjump=#2\unit
          \for=\uhsize-\leftjump
          \dow=\uvsize-\downjump
          \hsize=\for\vsize=\dow
          \goods \#3\}\}\vss\}%
    \nointerlineskip\}% close go
% ...
```

Figure 13. Text Placement

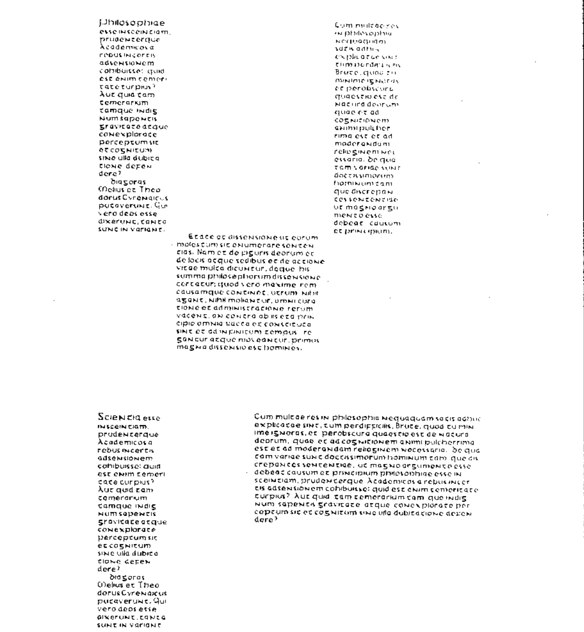


Figure 14. Text Placement Ouput

designer). Many patterns, from Egyptian scrolls to the *Book of Kells* and *Der DaDa*, can be examined and explained in terms of a grid.

Page patterns. Although usually every page can have a typical pattern, each page may have a unique pattern. The text, however, needs to flow sequentially from page to page or from section to section. Page layouts are defined after a scheme by August Mohr (see Figure 15). Each page has to be defined according to a format. A run file loads the definitions and text. A separate file describes how the finished page is to be built.

```
\def\I{1\}      \def\II{2\}
% ...
\def\beginsheet#1{\xdef\sheet{#1}%
     \setcount0 \sheet%
     \ifx\sheet\I  {\setIvone}
\else{\ifx\sheet\II {\setIIvone}
% ...
\def\setXIvdefs{%
   \gdef\setvone{\setXIvone}
% ...
\def\setXIvone{\vsize 99pc
   \hsize\twowide} % unboxed in set7
% ...
```

Figure 15. Page Macros (after Mohr)

The pieces are put together in the output routine, where the vsizes are defined. The output routine also adds the headers and footers.

Breaking and continuation. Breaking material on one page and continuing it on a later page also poses difficulties. Alan Hoenig set up a series of macros for newspaper layouts that takes care of breakage and continuation (see Figure 16). The entire story page is shaped, although the two pieces, the lead and the jump, may have different widths, and the story is divided with the vsplit command. The second half of the story is placed explicitly with a keyword-coded jump command.

```
\def\beginstory[title:#1][key:#2][main:%
   #3][jump:#4]{%
\setbox0=\hbox{\quad See #2 on page ??}
\setbox0=\vbox to2\baselineskip{\hbox
  to\mainhsize{\hss #1\hss}\vss}%
% ...
  \ifnum\count10\>0
   \goalheight=\count10\baselineskip
   \n=\count10    \else
   \whereami \computegoalheight
   \n=\count10
  \fi
  \advance\n by1
  \createparshapespec\makeboxident{#2}%
  \global\keyword={#2}%
  \partoks=% control typesetting
    {\tolerance=5000\pageshape=%
             \n\the\parshapespec}
  \putinvbox{#2}%
\} % end beginstory
% ...
```

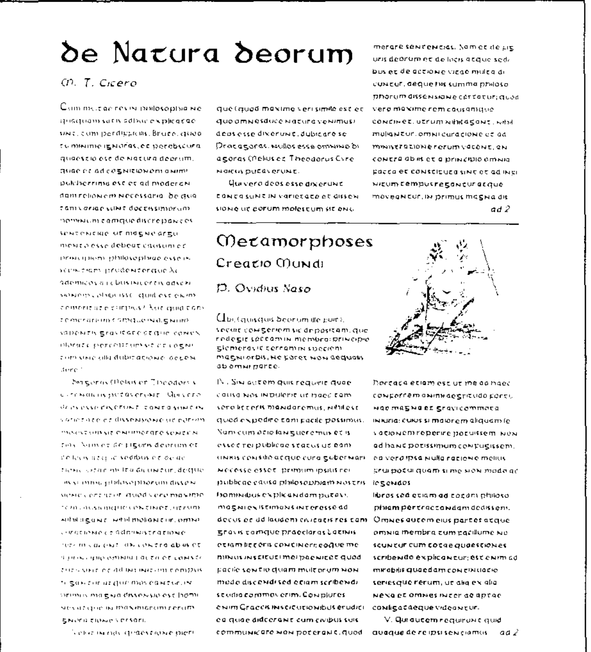
Figure 16. Page Splits (after Hoenig)

Hoenig also creates a macro that is useful for defining page shapes. Macros from Thomas Reid are also included. A sample for input specifies the title, depth, width, and continuation page (see Figure 17).

```
\beginstory[title: Natura Deorum][key:
  cicero][main: XI][jump: XXIV]%
\input cicero.tex
\endstory
\beginstory[title: Natura Universum][key:
  lucretius][main XIII][jump: XXV]%
\input lucretius.tex
\endstory
```

Figure 17. Sample file

The output on the first page is presented in Figure 18. The second page, much like any newspaper page, has blocks of continued articles (see Figure 19).


De Natura Deorum

M. T. Cicero

[illegible]

Metamorphoses

Creatio Mundi

P. Ovidius Naso

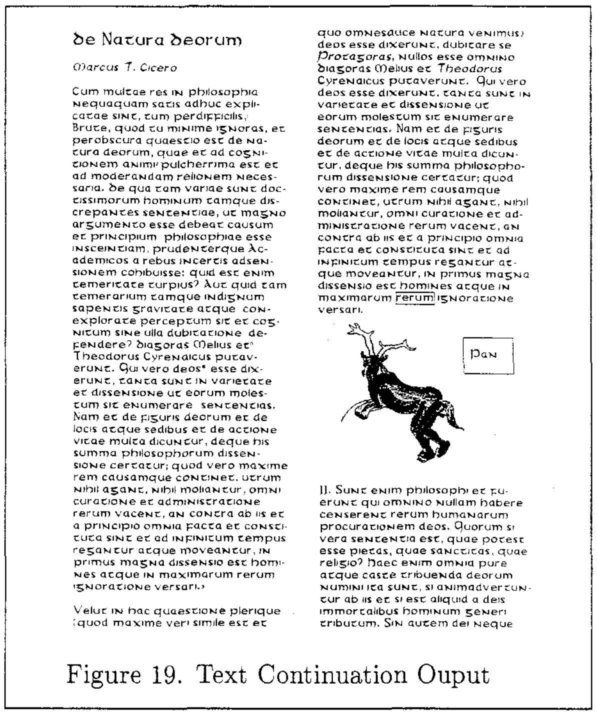
[illegible]

Figure 18. Text Placement Ouput

Summary

Page patterns have not changed greatly over three thousand years; most thoughts are presented in paragraphs in columns with graphics, and the page size seems to be relatively constant at seven by ten inches. Even hypertext pages seem to follow these conventions, although such pages also include cue identifiers and hot-spots.

The ArchiTEX program is intended to be a multi-directional, international page pattern maker.



de Natura deorum

Marcus T. Cicero

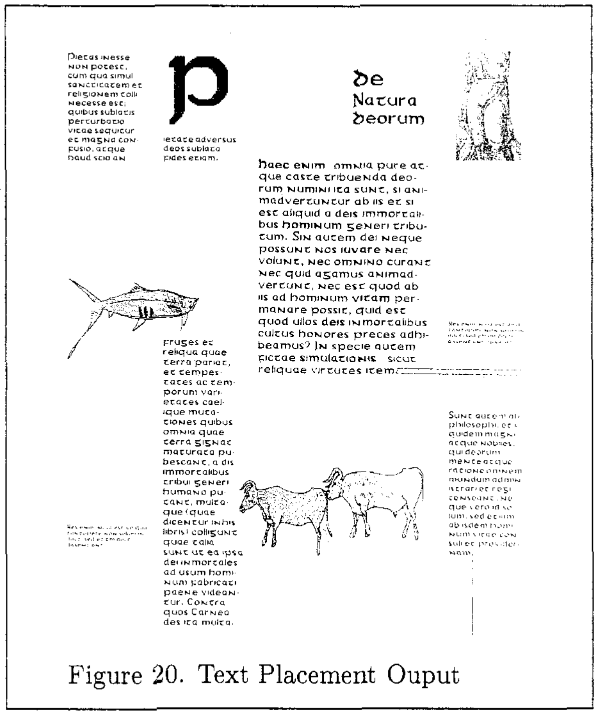
Cum multae res in philosophia nequaquam satis adhuc explicatae sint, tum perdifficilis, Brute, quod tu minime ignoras, et perobscura quaestio est de natura deorum, quae et ad cognitionem animi pulcherrima est et ad moderandam relionem necessaria. de qua tam variae sunt doctissimorum hominum tamque discrepantes sententiae, ut magno argumento esse debeat causum et principium philosophiae esse insceintiam, prudenterque Academicos a rebus incertis adsensionem cohibuisse: quid est enim temeritate turpius? Aut quid tam temerarium tamque indignum sapentis gravitate atque conexplorate perceptum sit et cognitum sine ulla dubitatione defendere? diagoras Melius et Theodorus Cyrenaicus putaverunt. Qui vero deos* esse dixerunt, tanta sunt in varietate et dissensione ut eorum molestum sit enumerare sententias. Nam et de figuris deorum et de locis atque sedibus et de actione vitae multa dicuntur, deque his summa philosophorum dissensione certatur; quod vero maxime rem causamque continet, utrum nihil agant, nihil moliantur, omni curatione et administratione rerum vacent, an contra ab iis et a principio omnia facta et constituta sint et ad infinitum tempus regantur atque moveantur, in primus magna dissensio est homines atque in maximarum rerum ignoratione versari.)

Velut in hac quaestione plerique (quod maxime veri simile est et quo omnesauce natura venimus) deos esse dixerunt, dubitare se *Protagoras*, nullos esse omnino diagoras Melius et *Theodorus* Cyrenaicus putaverunt. Qui vero deos esse dixerunt, tanta sunt in varietate et dissensione ut eorum molestum sit enumerare sententias. Nam et de figuris deorum et de locis atque sedibus et de actione vitae multa dicuntur, deque his summa philosophorum dissensione certatur; quod vero maxime rem causamque continet, utrum nihil agant, nihil moliantur, omni curatione et administratione rerum vacent, an contra ab iis et a principio omnia facta et constituta sint et ad infinitum tempus regantur atque moveantur, in primus magna dissensio est homines atque in maximarum rerum ignoratione versari.

Pan

II. Sunt enim philosophi et fuerunt qui omnino nullam habere censerent rerum humanarum procurationem deos. Quorum si vera sententia est, quae potest esse pietas, quae sanctitas, quae religio? haec enim omnia pure atque caste tribuenda deorum numini ita sunt, si animadvertuntur ab iis et si est aliquid a deis immortalibus hominum generi tributum. Sin autem dei neque

Figure 19. Text Continuation Ouput



Pietas inesse non potest, cum qua simul sanctitatem et religionem tolli necesse est; quibus sublatis perturbatio vitae sequitur et magna confusio, atque haud scio an

p

ietate adversus deos sublata fides etiam.

de Natura deorum

haec enim omnia pure atque caste tribuenda deorum numini ita sunt, si animadvertuntur ab iis et si est aliquid a deis immortalibus hominum generi tributum. Sin autem dei neque possunt nos iuvare nec volunt, nec omnino curant nec quid agamus animadvertunt, nec est quod ab iis ad hominum vitam permanare possit, quid est quod ullos deis inmortalibus cultus honores preces adhibeamus? In specie autem fictae simulationis sicut reliquae virtutes item

fruges et reliqua quae terra pariat, et tempestates ac temporum varietates caelique mutationes quibus omnia quae terra gignat maturata pubescant, a dis immortalibus tribui generi humano putant, multaque (quae dicentur in his libris) colligunt quae talia sunt ut ea ipsa dei inmortales ad usum hominum fabricati paene videantur. Contra quos Carneades ita multa.

Sunt autem alii philosophi, et i quidem magni atque nobiles, qui deorum mente atque ratione omnem mundum administrari et regi censeant. Neque vero id solum, sed etiam ab isdem hominum vitae consuli et provideri.

Figure 20. Text Placement Ouput

It defines one or more alphabets for text. It creates flexible, modular page patterns for single pages and places text in a grid. It also creates blocks for on-screen reference.

Alas, ArchiTEX is not finished. Placing continued text in large blocks across individually designed pages is still a major implementation problem (small tests work, however). Placing directional text with other directional text has not been implemented, yet, although others have done so already. Other parts of the program are still planned. Cues can be added for hypertext use (see Figure 20). ArchiTEX's bookmaker interface (in C) is planned to permit designers to manipulate the objects elements visually, and to enforce both formality and completeness for designers and programmers. Since ArchiTEX is a part-time, independent project, there is no time-frame for its completion.

Bibliography

Hoenig, Alan. "Line-Oriented Layout with TEX." Report to TUG, Providence, RI, 1988.

Knuth, Donald E. *The TEXbook*. Reading, MA: Addison-Wesley, 1984.

Knuth, Donald E. and MacKay, Pierre A. "Mixing right-to-left texts with left-to-right- texts." *TUGboat* 8(1), Pages 14–25, 1987.

MacKay, Pierre A. "Typesetting problem scripts", *Byte* Feb 86 Pages 201–216.

McLuhan, Marshall and Quentin Fiore. *The Medium is the Massage*. New York: Random House, 1967.

Mohr, August. "Multi-Column Layout in TEX80." *TUGboat* 6(2), Pages 87–95, 1985.

Nelson, Theodor H. "All for One and One for All." *Hypertext '87 Proceedings*. New York: ACM, 1987.

Skillin, M. E. et al. *Words Into Type*. New York: Appleton-Century-Crofts, 1964.

Soleri, Paolo. *Arcology*. Cambridge: The MIT Press, 1969.

Soleri, Paolo. *Sketchbooks of Paolo Soleri*. Cambridge: The MIT Press, 1971.

Tufte, Edward R. *The Visual Display of Quantitative Information*. Cheshire, CT: Graphics Press, 1982.

Appendix A: Alphabet Macros

```
% alphanum.mac  a macro for alphabetical counts using
% NA and foreign alphabets as defined in language strings
\def\english{\\a\\b\\c\\d\\e\\f\\g\\h\\i\\j\\k\\l\\m\\n\\o%
   \\p\\q\\r\\s\\t\\u\\v\\w\\x\\y\\z}
\def\bahbah{\\a\\b\\c\\{jk}\\x\\y\\z}
%
% define signifier for the text lists and signs for double letters
\def\signifier{ }
\def\alphasoup{\english}
\let\language=\alphasoup
%
\def\outofrange{ }% added 5/5 to prevent undef error
\def\pullet#1\of#2\to#3{\def#3{\outofrange}%
  \long\def\\##1{\advance#1-1 \ifnum#1=0 \def#3{##1}\fi}#2}
% sample: \pullet\MLa\of\alphasoup\to\signifier
%
% start new counter for the length of the alphabet
%\newcount\alphalen     \alphalen=30
% count alphabet length (after K378)
\def\findalen#1\to#2{#2=0 \long\def\\##1{\advance#2 by1 }#1}
% use immediately
\findalen\alphasoup\to\alphalen
%
% optional method to create aa, ab--az, or  aa, bb--zz
% start new counters for internal fixing of double letters
\newcount\SL        \SL=0         % letter
\newcount\MLa       \MLa=0        % double letter part
\newcount\MLb       \MLb=0        % double letter part
\newcount\SLm       \SLm=0        % multiplicand for mult signifiers
\newcount\thingnum  \thingnum=0   % list, update pages, etc
\newcount\maxthing  \maxthing=121 % maximum multiple alphabet lengths
   \maxthing=\alphalen \multiply\maxthing by \alphalen
   \advance\maxthing by \alphalen  \advance\maxthing by 1
% e.g., English (26 letters) results in 703 max
% new dimensions and boxes for letters, which are actually produced
% as leaders, not translated numbers
\newdimen\eachletwd   \eachletwd=10pt
\newbox\eachlet
\newif\ifab           \abfalse       % if letters are to be aa-az
%
\def\alphacon{\global\advance\thingnum by 1
    \ifnum \thingnum < \alphalen        % single letter
      \pullet\thingnum\of\alphasoup\to\signifier%  same for both
      \unskip\signifier%
    \else
      \ifab                             % for aa-az, za-zz
        \ifnum\thingnum < \maxthing     % less than absolute max
          \SL = \thingnum               % it's a double letter
              \divide\SL by \alphalen   % hold first letter
          \MLa = \thingnum              % prepare second
          \MLb = \thingnum
          \divide\MLb by \alphalen      % check for last letter
          \multiply\MLb by \alphalen
             \ifnum \thingnum = \MLb
                \advance\SL by -1        % back to last letter
                \MLa = \alphalen
             \else\advance\MLa by -\MLb  % identify second
             \fi
          \pullet\SL\of\alphasoup\to\signifier%
          \unskip\signifier%
          \pullet\MLa\of\alphasoup\to\signifier%
          \unskip\signifier%
        \else
          \message{Maximum alphabet use, * used} % too many things
          \signifier{*}
        \fi
      \else                           % for aa-zzzzz
        \MLa = \thingnum              % prepare second for mult.
        \MLb = \thingnum              % only second used
        \divide\MLb by \alphalen
               \SLm = \MLb            % get number to mult signif
               \advance\SLm by 1
        \multiply\MLb by \alphalen
           \ifnum \thingnum = \MLb
               \MLa = \alphalen
               \advance\SLm by -1   % back to end of alphabet
           \else\advance\MLa by -\MLb
           \fi
          \pullet\MLa\of\alphasoup\to\signifier% get letter
             \setbox\eachlet=\hbox{\signifier}%  put in box
             \eachletwd=\wd\eachlet%             measure width
             \multiply\eachletwd by \SLm%        multiply
             \unskip\hbox to \eachletwd{\unskip\leaders\hbox{%
                 \signifier}\hfill}%
     \fi%
  \fi%
}%
%
% define macro to make the alphabetized list.
% do by counter but strip off letters to replace numbers
% \long\def\allist{\hangindent100pt\noindent
%                    \hbox to 100pt{\alphacon\hfill}}%
% [or]
%            \number\pageno.\alphacon
% [or]
% \ifdoingpopps\alphacon\else\number\pageno\fi}
```

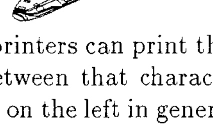
Integration of graphics into TEX

Friedhelm Sowa
Heinrich-Heine-Universität Düsseldorf
Universitätsrechenzentrum
Universitätsstraße 1
D–4000 Düsseldorf
FRG
Bitnet: `tex@dd0rud81`

Abstract

This paper is a practical approach to methods of integrating graphics into TEX. The problems that result from this integration and possible solutions are also discussed.

During the last two years we have had a lot of discussions during meetings, in publications and within electronic lists, about the question of how to integrate pictures into TEX. It is a very interesting question, indeed! The different answers to this question are also interesting because they lead to the interdependencies between TEX and driver programs. We can say that it is no problem in general to typeset pictures with TEX anywhere in a document by using different methods like the LATEX picture environment or similar macro packages, the `\special` primitive within a box, or something else. But when trying to typeset complex pictures, there is a great probability of failure because of TEX's limited capability in this area. When avoiding that effect by using the `\special` primitive, we surely need a driver that knows how to handle the special information. Nevertheless, it sounds very queer to talk about programs with special features that have to interpret a DeVice Independent `dvi` file. In one of the TEXhax issues of 1989 Stephan v. Bechtolsheim wrote "TEX was made for typesetting text...." Maybe he remembered the operation codes of the `dvi` file format when writing that. He is right, anyway. There is no correct driver in use that is not able to print:

M 

Though it is true that not all printers can print this.

What is the difference between that character on the right and the characters on the left in general, except the dimensions, from a drivers point of view?

The difference in usage between the inch high "M" and the shuttle is illustrated below, where the "M" code is on the left and the shuttle code is on the right:

```
\font\origin=cminch      \font\atesta=atesta
{\origin M}              {\atesta !(}
```

A lot of people have considered converting graphic output from different systems to fonts. This way of integrating graphics into TEX seems to minimize the existing problems.

The Graphic Sources

As this paper should not be a theoretical but a practical approach to the problem, it is necessary to have a look at the typical situation for the need of graphics integration.

Eleven years after the birth of TEX, there is someone, somewhere in the academic world, writing a paper. The visualization of the scientific results is done by a dedicated system. For a lot of different disciplines there are a lot of different programs available on the market which convert data into a picture. Often those systems are embedded in a single purpose system. In any case, it is just that system which is the best for,

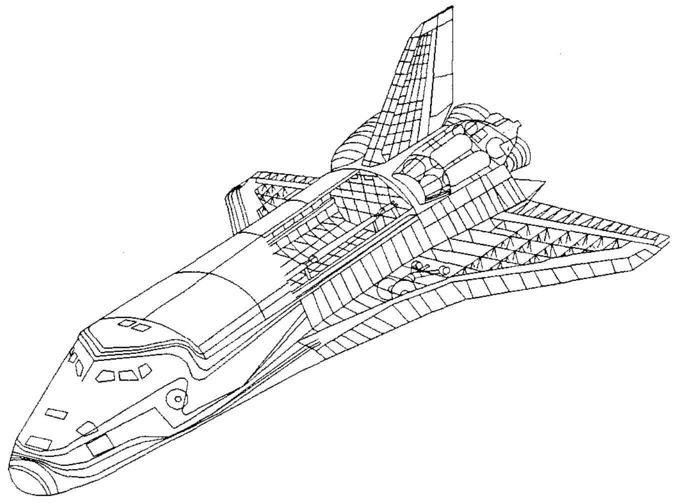
1. chemistry
2. physics
3. mathematics
4. statistics
5. engineering
6. economics
7. geography
8. etc.,

otherwise it would not be used. As long as an institute has computer freaks to do the programming, there will always be a "best" program for any particular area of science. In any case, the system produces pictures, that can be printed on a plotter as well as on a laser printer or a matrix printer. Some systems use a device independent metafile, others produce vector graphics, or even raster graphics immediately.

In the decade of micros, the importance of raster graphics (compressed or uncompressed) grew by the use of scanners, cameras, screen dumps and paint programs. The difficulties of making pictures was decrease at the same time. Even the digitization of "real pictures," like photographies, became possible.

Now back to the lonesome scientist who is using TEX to write his paper and trying to describe what is to be seen on the output of a graphic system. Usually there is someone who asks the question, "Hi, I'm using TEX for my paper and I want to include some pictures. How does it work?"

The individual situation It works, but how is not important here. Only the dimensions of the picture in the final print are important. Either the user wants to print it on his own 300 dots per inch (dpi) laser printer or on the publisher's 1200 dpi typesetter. That is the most important item for the layout of the document. The picture will be included into the document for a certain print device by telling TEX the picture's `tfm` information. Although the files around TEX are device independent the picture in it's original form is not device independent with respect to the document's purpose. Of course, there is a formal device independency concerning the `tfm` files and the `pk` files or `pixel` files, but depending on the resolution, the picture should be printed in different sizes. Using a high resolution device the picture can be generated larger than for a low resolution device because there are more dots available for the same amount of space.



This picture was generated for a 300 dpi device. The same picture on the previous page was generated for a Tektronix display and included for purposes of demonstration. The higher resolution allows for more detail. Expressing this thought in one sentence,

> A complex graphic on a high resolution device requires less space than the same graphic on a low resolution device to see the details.

The user has to decide, depending on the resolution of the output device, what size the picture should be on the page. This is only true if the source of the picture is a vector oriented graphic. Bitmap graphics are fixed in size. Changes to an existing black/white bitmap by scaling will change the quality of the picture too, unless the run of the black bits are converted to vectors. In this case the circle is closed again.

The situation in general. The problem of integrating pictures into a document is heightened by facts, which are basically the same for every author. Those facts are:

1. the kind of the graphic
   (a) vector graphics
   (b) black/white bitmaps
   (c) bitmaps with grey pixels
2. the resolution of the output device
3. the available driver program
4. the ability of TEX to typeset text
5. the motivation of the author

What the author needs is a link between his graphics and his text. There is no question that he himself has to do something to make the link. The simplest way is to start a program that converts the graphic file into a font for the author's output device by generating the `tfm`'s and `pk` or `pixel` files and writes some TEX instructions to typeset the picture. So he only has to say something like `\setpicxy` somewhere in his text. That should be easy enough for everyone.

It was already mentioned that some vector oriented graphic systems generate metafiles like GKS, Phigs etc. Is it wrong to say that every system of that kind generates metafiles in any way when processing a picture? Certainly not. The aim of those systems is to draw some lines or curves on a sheet of paper. This is done by either external driver programs or special subroutines for specified devices. At this point, we find common instructions for printers and plotters. For those devices, a fixed instruction set is defined, exists for a long period of time and does not need any standardization committee.

It is a standard in an isolated area, acknowledged by a lot of driver programs or subroutines.

This is the relevant interface for a font generating program. The program should know the set of instructions of two or three manufacturers to solve the integration problems of about 80% or even 90% of the TeX users who make their pictures with such a graphics system.

There is a similar situation in the world of bitmaps. Many abbreviations like TIFF, PCX, IMG or CUT represent the package of more or less compressed bitmaps. A lot of conversion programs allow one to use the appropriate package for an interesting picture. Here it is probable the problem will be solved up to 99.99% of the time.

The Graphic Fonts

When talking about fonts within the TeX community, everybody remembers METAFONT. The readers of *TUGboat* or the Exeter proceedings know, that there are some programs that use the sister of TeX to make fonts from graphic files. He should also know, that some approaches were made without METAFONT to generate graphic fonts. The main reason to avoid the usage of METAFONT is the ability of TeX to run on any machine without METAFONT. The user knows it, and he typically utilizes TeX, not METAFONT. He only wants one program to include the picture, not a complex system.

Driver compatibility. It was already mentioned that the typical user does not need device independence. He only needs the driver selected to print the final document. This sounds a little bit like a driver restriction. And indeed it is. A correct driver should follow the recommendations of the `dvi` driver standards committee. This is not a wish but a law. Especially on small systems there is a restriction concerning the size of font files. A limit of 64KB is to be found in a lot of programs. This is the smallest common denominator when generating graphic fonts.

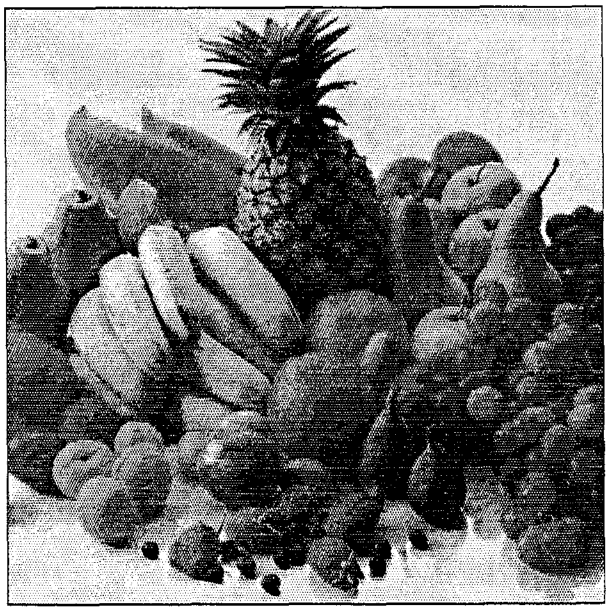
It may be that there is a difference among drivers concerning the ability to print large characters. Here is the next guideline for generating fonts.

As a conclusion of those facts we can say: If a driver program is able to handle

1. the number of fonts TeX allows
2. fonts with characters not larger than 1 inch
3. fonts not exceeding the 64KB limit

then it is able to print documents that include graphics.

The suitable program. A practical approach to the problem of including graphics includes the development of a program too, especially for an employee of a computer center in a university. After gathering information about the structure of `tfm` files, `pixel` files and `pk` files on the one hand and about TIFF and CUT format on the other hand, the foundation stone for BM2FONT was laid. It should be able to handle bitmap formats and simple bitmaps. The latter ones are generated by a special vector conversion program on a central computer system, where the vector approach was done for historical reasons.



Since a lot of students and scientists are working on personal computers BM2FONT was written for DOS. Here we find the origin of the other kinds of bitmaps. After long discussions with the WEB system and the Pascal compiler the program got a consolidated status. It produced a lot of `tfm` files and `pk` files and files like:

```
\newbox\obstbox
\newdimen\obstw
\font\aobst=aobst
\font\bobst=bobst
\font\cobst=cobst
\font\dobst=dobst
\setbox\obstbox=\vbox{\hbox{%
\aobst !()+}}
\obstw=\wd\obstbox
\setbox\obstbox=\hbox{\vbox{\hsize=\obstw%
\parskip=0pt\offinterlineskip\parindent0pt%
\aobst !()+\vskip0pt%
,\bobst !()\vskip0pt%
+,\cobst !(\vskip0pt%
)+,\dobst !}}
```

```
\def\setobst{\box\obstbox}
```

Using `\input obst` to include the TEXnical description of the graphic into the TEX source file and `\setobst` to typeset the picture the job of integration is done.

Everything worked fine and there were no new problems concerning the available driver programs. Only the known errors appeared.

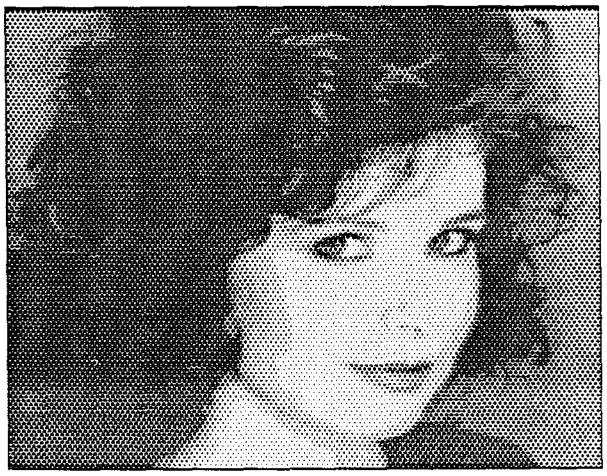
Halftone Graphics

The conversion of black/white bitmaps to fonts is a rather boring job. Pixels p with values in the range $0 \leq p \leq 255$ present additional fun before that conversion. In this case, a temporary bitmap must be generated, where the original pixels are represented by areas partially filled with black or white pixels, which are usually the only types of pixels available on common devices. In ***TUGboat***, there have been interesting articles on this (Volume 8, number 2 and 3 by Don Knuth and Adrian F. Clark.) They used fixed patterns for defined devices within very special fonts. The conclusion of those articles was the need of a big TEX for small patterns. With that solution, the still life on the previous page would require 79,341 characters for the same size picture.

The other suggestion of Knuth was a better one. He proposed a picture specific font that includes the whole picture. By assigning larger areas to the characters of the font, he reduced the amount of characters in the input file. But there was no consideration of driver limitations and, as the father of METAFONT, he was not willing to reconsider this, as it was not a solution for any acceptable output device. Nevertheless it was the right idea to solve the problem.

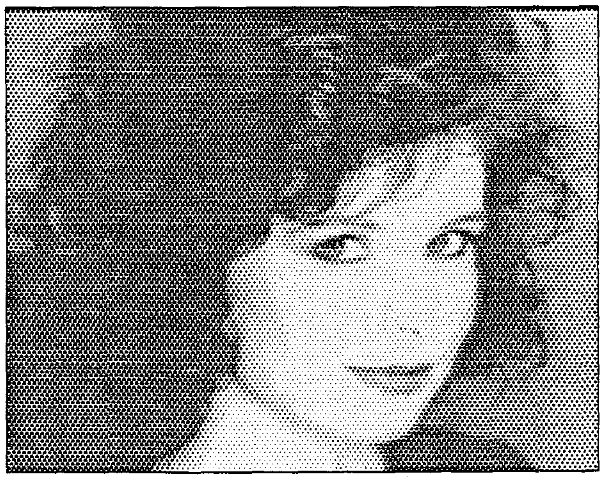
Another aspect of the problem is the subjective behavior of the human eye, which tries to see a homogeneous picture, and not the single pixels. This can be achieved by representing different grey shades by black dots, which differ in size. This is not a new idea, but it is difficult to achieve good results in a general way for different devices with resolutions between 300 and 1200 dpi.

Generating grey patterns. To generate black dots, we take a n x m square of pixels. For every possible grey shade, a new pixel in this square, close to the last blackened pixel, is turned from white to black. This is the principal way to generate black dots with different sizes. The grey values of the source file are to be scaled up or down to the available grey shades, and then the picture is composed with a well structured number of more or less black squares.



Usually the original file comes from a system that uses a scanner or a video camera. To get the details off a picture, it should be digitized with a high contrast. This means in practice that 256 grey values give better results than 16 grey values. Most systems follow this rule, but when scaling the usable grey shades down, depending on the square's size, it is necessary to distribute the rounding errors of the pixel to the neighboring pixels. Experience has shown that error distribution gives a smooth transition between areas of nearly same grey values.

The picture above was taken by a video camera system that uses 256 colors. It was converted by using a 4 x 4 square. The available 16 grey shades are reduced to 12 because the printer seems to overlap the pixels and produces solid black dots quite early. Beside that manipulation, a correction of the lines of patterns was necessary. As the pixels of the camera system are higher, rather than wider, the last line of the square was ignored. This is a good method to avoid long faces. The following example shows the result of error distribution. The changing of colors became smooth, but the contrast in the picture was reduced.



In this example, the distribution was done by the Floyd-Steinberg algorithm, that gives $\frac{3}{8}$ of the error to the side and down and $\frac{1}{4}$ diagonally down.

Simulating more patterns. The last two examples were made with this described method, but a little trick was added. Four pixels of the source file had to build one "grey dot" in the temporary bitmap. This gave the opportunity to use one dither matrix for any size of squares. We had to start in the top left corner of the matrix to find the position of the pixel in the *n* x *m* square which had to become black.

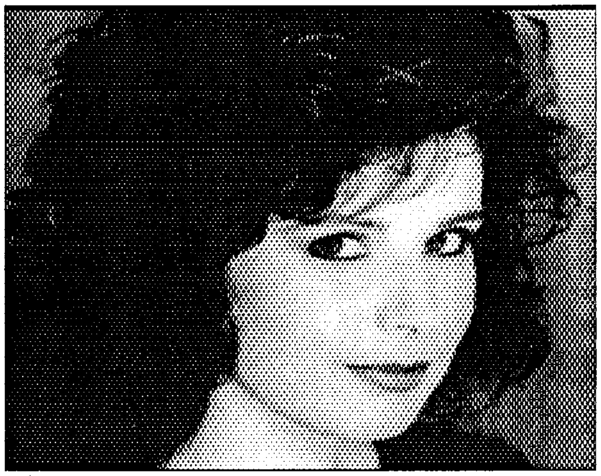
	column						
row	1	2	3	4	5	6	7
1	01	03	06	10	18	26	38
2	02	04	07	12	19	28	40
3	05	08	09	14	21	30	42
4	11	13	15	16	23	32	44
5	17	20	22	24	25	34	46
6	27	29	31	33	35	36	48
7	37	39	41	43	45	47	49

This resulted in four different patterns for each grey value, because the black pixels of one "grey dot" should be close together. The position of pixel 1 in the first pattern is the upper left corner of the square, in the second pattern the lower right corner, in the third pattern the lower left corner and in the last pattern the upper right corner. So the original grey pixels can grow together considering the values of the neighbors.

	pattern			
shade	1	2	3	4
	●			
1				
	●			
2				
		●		
	●			
3				
		●	●	
	●			●
4				
		●	●	
	●			●
5	●			
		●	●	
	●			●
6	●	●		
		●	●	
⋮				

	pattern			
shade	1	2	3	4
	●●	●	●	●●
20	●●	●●	●●	●●
	●	●●	●●	●
	●●●	●	●	●●
21	●●	●●	●●	●●
	●	●●	●●	●
	●●●	●	●	●●
22	●●	●●	●●	●●
	●	●●●	●●	●
⋮	⋮	⋮	⋮	⋮
	●●●	●●	●●	●●●
32	●●●	●●●	●●●	●●●
	●●	●●●	●●●	●●
	●●●	●●	●●	●●●
33	●●●	●●●	●●●	●●●
	●●●	●●●	●●●	●●
	●●●	●●●	●●	●●●
34	●●●	●●●	●●●	●●●
	●●●	●●●	●●●	●●
	●●●	●●●	●●●	●●●
35	●●●	●●●	●●●	●●●
	●●●	●●●	●●●	●●
	●●●	●●●	●●●	●●●
36	●●●	●●●	●●●	●●●
	●●●	●●●	●●●	●●●

Still this method was not good enough. More grey shades were necessary to maintain the contrast in spite of error distribution. In fact, we did not have 16 grey shades available for one "grey dot" but four times as many. So, we multiplied the shades by 4 and changed pattern generation. Instead of making four patterns for one grey shade with the same number of black pixels, we produced four patterns for a certain grey value by decreasing the number of black pixels in every pattern by one. The result was much better than those of the older version.



This version seems to consider the neighboring pixel values best. When making the examples for the effect of error distribution, the old version was used. The new method for the example gave the same results when the error distribution was eliminated in the program.

Bibliography

Burger, Peter, and Duncan Gillies, 1989, ***Interactive Computer Graphics***, Addison Wesley Publishing Co.

Childs, Bart, Alan Stolleis and Don Berryman, 1989, "A portable graphics inclusion" in *TUGboat* Volume 10, Number 1, page 44 – 46

Clark, Adrian F. 1987, "Halftone output from TEX" in *TUGboat* Volume 8, Number 3, page 270 – 274

Heinz, Alois. 1990, "Including pictures in TEX" in Malcolm Clark, ed., ***TEX Applications, Uses, Methods: TEX88 Proceedings***, page 141 – 151, Ellis Horwood Series in Computers and their Applications

Knuth, Donald E. 1986, ***The TEXbook, Computers & Typesetting***, Volume A. Addison Wesley Publishing Co.

Knuth, Donald E. 1986, ***The* METAFONT*book, Computers & Typesetting***, , Volume C. Addison Wesley Publishing Co.

Knuth, Donald E. 1987, "Fonts for digital halftones" in *TUGboat* Volume 8, Number 2, page 135 – 160

Messinger, Heinz. 1982, ***Langenscheidts Großwörterbuch, "Der kleine Muret-Sanders", Deutsch-Englisch***, Langenscheidt

Pickrell, Lee S. 1990, "Combining graphics with TEXon IBM PC-compatible systems and LaserJet printers" in *TUGboat* Volume 11, Number 1, page 26 – 31

Pickrell Lee S. 1990, "Combining graphics with TEXon IBM PC-compatible systems and LaserJet printers, Part II" in *TUGboat* Volume 11, Number 2, page 200 – 206

Rogers, David F. 1989, "Computer graphics and TEX, a Challenge" in *TUGboat* Volume 10, Number 1, page 39 – 44

Simpson, Richard O. 1990, "Nontraditional Uses of METAFONT" in Malcolm Clark, ed., *TEX Applications, Uses, Methods: TEX88 proceedings*, page 259 – 271, Ellis Horwood Series in Computers and their Applications

Wilcox, Patricia. 1989, "METAPLOT: Machine-independent line graphics for TEX" in *TUGboat* Volume 10, Number 2, page 179 – 187

PostScript, **QuickDraw**, TEX

Timothy Murphy
School of Mathematics, Trinity College Dublin
email: `tim@maths.tcd.ie`

Abstract

How can graphical material best be incorporated with TEX? The author's experience with TEX on the Macintosh, interpreted through PostScript or **QuickDraw** (as the printer decrees), has driven him remorselessly towards an heretical view—that TEX must be dissected, and re-assembled as a context-free grammar, with presentation in `DVI` format. For only in that way can TEX and graphic meet on equal terms .

Because of the very problem addressed here—the lack of an agreed standard for marrying graphic with TEX—this article differs substantially from the corresponding talk. For that talk was, in effect, a practical demonstration of TEX on the Mac, and a comparison of **QuickDraw** with PostScript as graphical interpreter. This article might perhaps better be entitled "Some Thoughts Occasioned by an Experiment in PostScript, **QuickDraw** and TEX."

Introduction

The incorporation of graphical material into TEX is the principal and perennial preoccupation of the TEX community. Knuth has taught us how to use the *quill.* Now, like those medieval monks who penned the Book of Kells, we wish to illustrate the text.

But wait! Why do we always speak of incorporating graphical material into TEX? Why shouldn't we, conversely, incorporate TEXnical material into graphics? Why not — those of a sensitive disposition should avert their eyes at this point — why not 'encapsulated TEX'?

A Word of Thanks

The mathematical community owes Donald Kriuth an immense debt of gratitude. He has lifted a grievous load from our shoulders. TEX is our washing machine, microwave oven and dishwasher, all rolled into one.

But debtors were always ungrateful. ("Arthur Guinness has been very good to the citizens of Dublin," the pompous speaker intoned. "And the citizens of Dublin have been very good to Arthur Guinness," piped up Brendan Behan from the rear, a pint in his hand.) In the spirit of that ancient adage, let us proceed to bite the hand that feeds us...

The Grand Panjandrum

TEX is a black box. We feed it with our ideas — fragmentary and ill-formed. We crank the handle, and lo and behold! — our thoughts emerge (quite slowly) from the other end, if not laundered, then so beautifully presented we almost blush.

It is true that Knuth, like a good magician, throws open the box with a flourish — "Look, no tricks!" He even tells us what all those knobs are for. Then, just as we begin to get the hang of it, the lid crashes down on our fingers.

Who does not recall those dread words at the opening of `tex.web` — *Unless your name is D. E. Knuth....* (The words conjure up the image of an ancient Registrar timidly asking, "If I might be so bold as to enquire, Miss, why do you want to change your name to Donald Knuth.")

But I propose that we bite the forbidden fruit, that we disobey the Master. In these days of recombinant DNA, could anyone balk at recombinant TEX?

TEX as a Language

TEX is a two-headed monster — it is a language, and it is a program for interpreting that language. And the program, like the Queen in *Alice in Wonderland,* is the final arbiter. If it says we are talking nonsense, then we *are* talking nonsense.

I'm reminded of the early days of UNIX and C, when the answer to the question, "What is −5/2?" was, whatever the Ritchie compiler said it was.

Then came context-free grammars, and Backus-Naur, and yacc. And they begat Johnson's portable C compiler. And the modern world was born.

Let me put my plea plainly (or should that be lplainly). TEXackers of the world: give us tex.yacc.

Wouldn't this express the *true* meaning of TEX? To see its anatomy dissected in that way — with hboxes packed recursively into vboxes, and vboxes into hboxes, and so *ad infinitum* (like Jonathan Swift's fleas).

The Grammar of LATEX

Is it fanciful to see in a LATEX document style the germ of a context-free grammar? Couldn't we express it most succinctly as a yacc file? Its central section might start something like this:

```
DOCUMENT: HEADING TEXT REFERENCES
  ;
HEADING: TITLE AUTHORS INSTITUTION
         DATE ABSTRACT
  ;
TITLE: title_tok string
  {
    output("\begin{center}
            \font\largebf");
    output(yytext);
    output("\end{center}");
  }
  ;
AUTHORS: /* empty */
  | AUTHORS AUTHOR
  ;
```

Euromath, Grif and TEX

At this point, I should confess that my inspiration is drawn from a bizarre project sponsored by *Euromath,* a consortium of West European Mathematical Societies.

Euromath has contracted Gipsi, a French software house, to develop a 'mathematical editor'. More precisely, Gipsi has undertaken to extend an existing editor — called *Grif* — to include mathematical formulæ, as well as simple graphics[2]. This editor is to interface in some as yet unspecified way with TEX — or more probably, with LATEX. At the very least, we are promised a TEX-driven editor and previewer.

[2] For further information on Grif, contact paoli@gipsi.gipsi.fr.

The relevance of this project to my argument is that Grif envisages the document before it as a context-free grammar. (Gipsi uses the term 'structured information', but it comes to the same thing.)

Unfortunately, the project is doomed to failure, in my view, because Euromath has agreed, for its part, to produce a definitive analysis of mathematical expressions... In other words, it must define a context-free grammar which will include within its ambit everything that any mathematician might say. On that happy day, which will probably occur about the same time as the state withers away, Gipsi will take the grammar as basis for its mathematical Grif.

Encapsulated TEX

But if Grif — or Euromath — could lower its sights, and accept that its task was ended when a mathematical formula was encountered;

And if TEX could come off its high horse, and accept that a single formula in an alien environment was an object worthy of its attention;

Then the two might marry, and be fruitful, and make Euromath happy.

In conclusion, it seems appropriate to speak of 'encapsulated TEX' in this context, by analogy with encapsulated PostScript. For while common-or-garden PostScript defines a whole document, like TEX, encapsulated PostScript describes an isolated graphic — a Mandelbrot set, or a picture of Marilyn Monroe — precisely as required, in fact, for incorporation into TEX.

An Approach to Drawing Circuit Diagrams for Text Books

M.P.Maclenan & G.M.Burns
South Bank Polytechnic, Borough Road, London
email: `burnsm@vax.southbank-poly.ac.uk`

Abstract

The development of a library of pictograms is described. The pictograms are defined using macros embodied in PiCTeX and used to create an application to enable high definition circuit diagrams to be easily included in TeXdocuments.

Introduction

The problems of including pictures within a TeX document have been considered by several people e.g. Norris *et al* and recently by Rogers and by Childs *et al* . Norris imports a PostScript file, Roger's paper suggests creating a mini-graphics package while Childs argues for a specialised hardware. *Textures* which we use on Apple hardware provides a `\special{picture#}` to be included which allows for such combinations of pictures and text to be created as

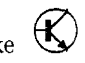
TWICKENHAM
RUGBY
FOOTBALL
CLUB

This facility is excellent if mere representation is all that is required, but there is little flexibility once the picture is imported to TeX and making changes becomes tedious. We consider here a use of PiCTeX to create an application, referred to as CiRTeX for brevity, which might be used to produce electronic circuit diagrams. The use of PiCTeX was considered to have the following advantages:

1. it maintains device independence
2. it allows a mixture of good graphic pictograms with the elegance of TeX typesetting, ideal for circuit diagrams
3. compositors need only be conversant with TeX and the simple process of creating diagrams embodied in CiRTeX to produce diagrams.

Concepts Used in CiRTeX

The simple principle adopted is that small, recurring objects are defined in a co-ordinate system using 1pt as the base dimension of the space. From these simple objects, small pictograms are defined and from these pictograms, complete circuit diagrams can be drawn. Thus

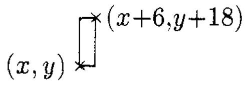
and are used to make 

Section 9.2 of The PiCTeX Manual discusses the use of `\setdimensionmode` to `\put` objects in an absolute space with respect to a given origin. However, by working consistently with 1pt as the base dimension, co-ordinate space was eventually preferred. There are two possible approaches to defining the objects and pictograms, in particular:

1. to define the objects in terms of the absolute coefficients (x, y) of the final diagram
2. to build up a box using co-ordinates relative to a given origin within the box and then to `\put {box} at` x y
   where (x, y) is now absolute within the final diagram.

In the final version of CiRTeX, both approaches are used. It was initially decided to relate *all* objects and pictograms to an origin at their bottom left hand corner, but eventually it was realised that it was convenient if the origin of some of them was logical rather than predefined in this way.

For example, consider a simple vertical rectangle used, according to BS3939, to represent a resistor.



Using approach A, `\Rv` can be defined by

```
\def\Rv#1#2{% #1=$x$,#2=$y$
\cxone=#1 \advance\cxone by 6
\cyone=#2 \advance\cyone by 18
\putrectangle corners at
{#1}{cyone} and {cxone}{#2}}
```

It is then used simply as \Rv{x}{y} where x, y are absolute coordinates in the space of the final diagram.

Approach B follows the suggestion made in Section 8.2 of the PiCTeX manual on how to include pictures in pictures. Thus \Rv can be defined by

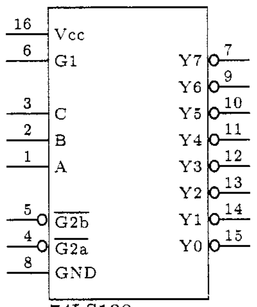
```
\def\Rv{\%
\putrectangle corners at 0 18 and 6 0
}
```

\Rv is subsequently placed in the final picture using \put {\Rv} [bl] at x y. In this case, approach B is preferred because it is simpler and use of \put allows for an x_{shift} and/or a y_{shift} so that the resistor can be located at the known end of a vertical line and offset 3pt to place the origin. Note that we found that the inclusion of \beginpicture $\cdots$ \endpicture as suggested in Section 8.2 was found to be unnecessary in the definition.

An example of logical placement is

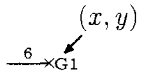
```
\label#1#2#3#4{% #1=$x$ #2=$y$
% #3=pin number #4=label $\cdots$
```

used to draw this IC.



74LS138

Here, each label is placed as



The parameter #4 can contain any text, including symbols, so that the depth of this sub-box is unknown. Since its depth will define the depth of \label, logical placement is essential to maintain alignment.

Once these concepts were understood, a library of objects and pictograms was developed.

Tricks

▷ and o are used to make ▷o

Initially, the inversion bubble that converts a buffer to an inverter was constructed using \circulararc, which took a long time to compile and used considerable amounts of memory. It was noticed that a lower case o in courier font is circular, and at 10pt, of a perfect size to be used as \bubble which can be \put immediately!

▶ and ❙ are used to make ▶❙

Infill of shapes proved to be difficult. The diode triangle is constructed by successively \puting on a centerline vertical rules of diminishing height and offset 0.4pt to each other. The down arrow of the *npn* transistor is drawn similarly by \plotting a series of lines radiating from the tip of the arrow.

Labels of some signals are drawn with a bar over the symbol to indicate that the signals are active low. Given that the symbols are of unknown length, this was solved by

```
\def\actlow#1{\hbox{#1}
\setbox1=\hbox{#1}\kern-\wd1
\raise5.5pt\hbox{%
\vrule height 0.4pt width \wd1
}}
```

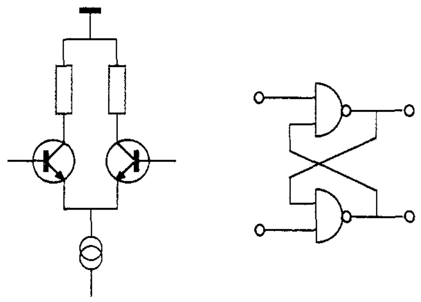
Problems

It must be stressed that problems discussed here relate to our use of PiCTeX in trying to create an application for which it was not intended and not to PiCTeX itself.

1. \linethickness has no effect on PiCTeX macros producing curves so that, for example, the circular outline used to draw the transistor could not be changed.
2. \loop $\cdots$ \repeat does not seem to work within any of our defined macros which involve use of PiCTeX.
3. During development, variable names were chosen which by chance coincided with names used in PiCTeX. To avoid repetition of the problems this caused, all CiRTeX variables now start with a c e.g. \cxone
4. Our initial use of \setbox1 gave very odd results until the valuable advice to always use \newbox was remembered (which produced \box22!).
5. Our attempt at object-orientated development implies deeper and deeper grouping which may contribute to greater use of memory than is necessary.

Conclusion

The extent of development now allows us to quickly draw diagrams like these,



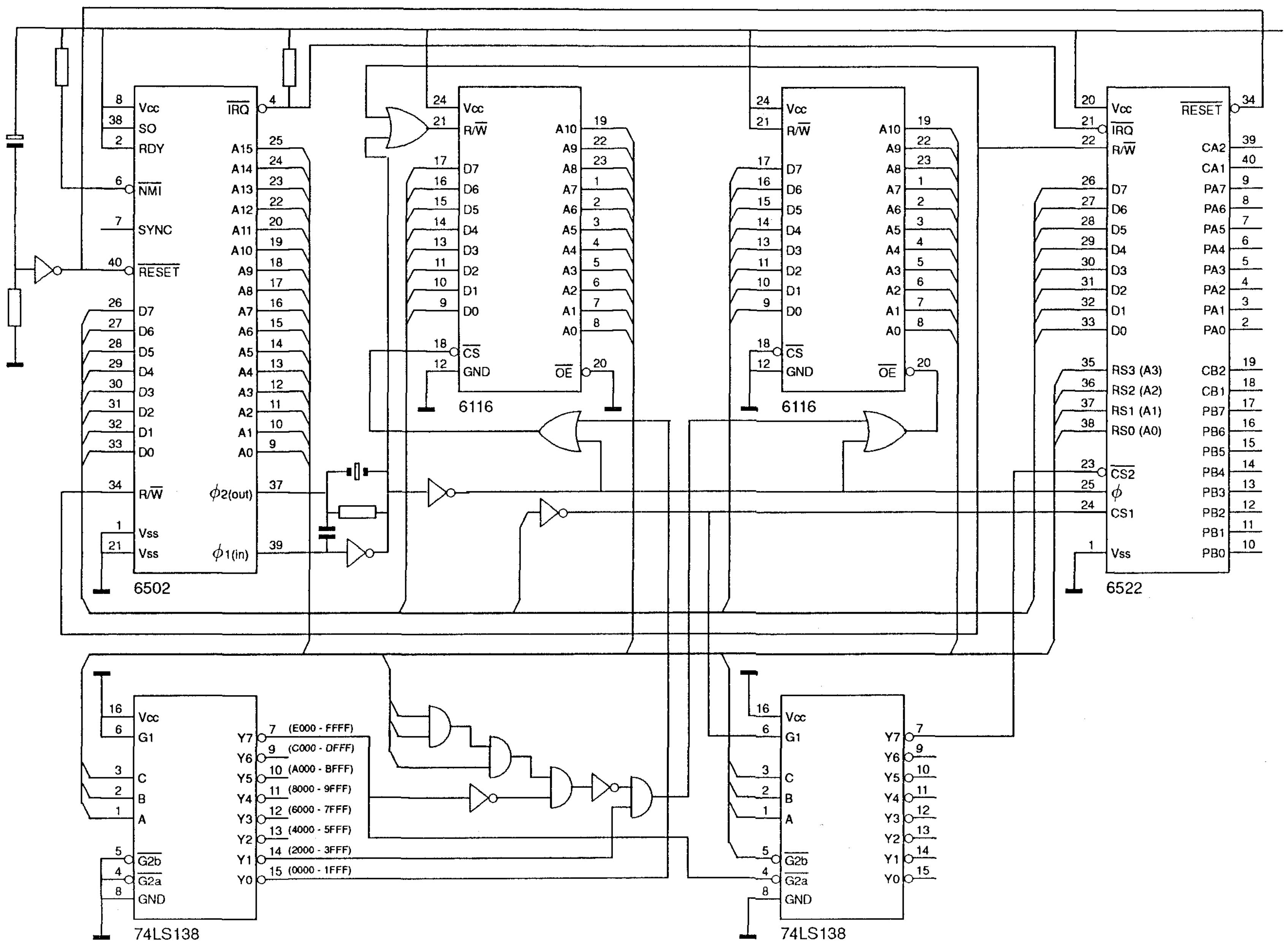
which can be totally integrated into a TEX document. Indeed, it is now possible to produce diagrams such as that in the Appendix, but this takes 37 kbytes of memory and over 6 minutes to compile on a Macintosh II! The reason for this is simple, that PICTEX must calculate many of the basic shapes of the objects. A tantalising reference is made on page 389 of the *The TEXbook* to a font `\qc` which contains four quarter circles as characters but no hint is given as to its availability. If a font which contains *all* the basic object shapes which we use could be developed and combined with the computational power implied in the use of `\put` within PICTEX, then as Rogers suggests, we would be able to produce a package capable of easy production of circuit diagrams of suitable quality to be included in text books.

Bibliography

Norris & Oakley. 1988. "Electronic Publishing," *TEX88 Exeter Proceedings.*

Rogers, David. 1988. *TUGboat* 10(1), April.

Appendix


6502
Vcc
SO
RDY
NMI
SYNC
RESET
IRQ
R/W
ϕ2(out)
ϕ1(in)
Vss
6116
CS
GND
OE
6522
RS3 (A3)
RS2 (A2)
RS1 (A1)
RS0 (A0)
CS2
CS1
74LS138
G1
G2b
G2a
(E000 - FFFF)
(C000 - DFFF)
(A000 - BFFF)
(8000 - 9FFF)
(6000 - 7FFF)
(4000 - 5FFF)
(2000 - 3FFF)
(0000 - 1FFF)

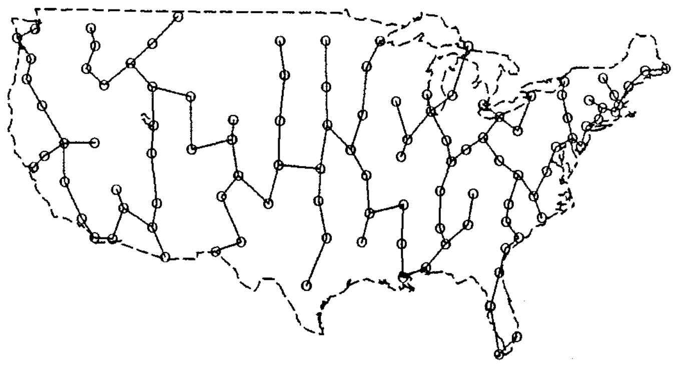


Figure 3: A minimal spanning tree connecting selected US cities.

Wichura's PiCTeX[1987], is composed of a set of TeX macros that allow the user to draw graphics such as curves, histograms, etc. PiCTeX draws those graphics by printing dots so close to one another that they seem to form a continuous line.

Graphics drawn with PiCTeX form an integral part of the text and therefore benefit from all the advantages offered by TeX, e.g., both computer and peripheral independency. TeX's power in mathematics composition can be used in captions.

I deliberately chose not to use PiCTeX facilities to draw vertical or horizontal rules because the relative positions of the rules and the lines made up of dots could not be set to match each other. It is possibly due to the dvi to PostScript driver, but by eliminating the rules I got rid of the problem.

The Driver

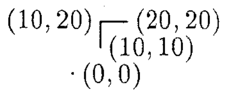
Written in C, the driver is divided into elementary procedures (moving the pen, drawing a line, drawing text). Those procedures are called by S graphic functions while drawing graphics.

Graphics are stored in a file named `Slatex.out`. This file contains PiCTeX commands that will then be included in a TeX text file.

The default size of the graphics is 360×360 points. The S coordinate system uses integer variables. The hundredth of a point was the measure to be used for adequate precision.

When several lines are drawn at the same time, they are concatenated and a single `\put` command is issued. To draw a line with the following coordinates: $\{(10,10)(10,20)(20,20)\}$ the driver will generate the command:

```
\put{\!start(10,10)
     \!ljoin(10,20)
     \!ljoin(20,20)
} [lb] at 0 0
```



The result obtained is much more compact than when using two separate commands, one for $\{(10,10)(10,20)\}$ and another for $\{(10,20)(20,20)\}$.

Text is drawn as strings and not as separate characters; TeX attends to the composition of the final text strings. Text strings can be centred, left- or right-flushed respective to the current point.

As the driver was designed to be used with LaTeX, changing character size is done by the commands `\tiny`,...,`\Huge`; the default size is `\small`. Changing line thickness is done by changing the size of the dots used to draw the lines. A line type change is obtained by the PiCTeX `\setdashpattern` command.

Using the Driver

From S This driver can be used like any S graphic driver. It is called by the new S command **`latex`**. It is possible to change the default graphics width and height with command options. The minimum graphics size is 100×100 points.

When the driver starts, the file `Slatex.out` is opened and PiCTeX graphic commands are added

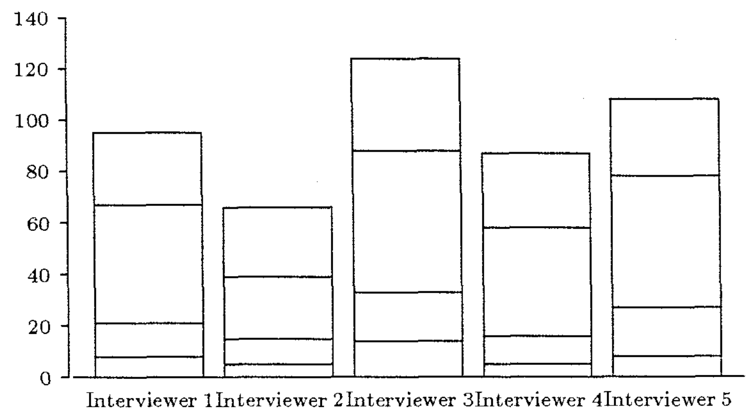


Figure 4: Bar plot of responses received by 5 telephone interviewers.

to the file. If the driver is started a second time, the file is overwritten. This will keep the file from growing indefinitely, but if the user wishes to save the `Slatex.out` file, he will have to rename it. A standard graphic file takes up 10 KB.

The only problem —apparently due to a fault in an S function—I could not do away with, is when the user need to type a \ character in a character string. He will have to type four "\" successively, e.g., "parameter θ_2" has to be typed `parameter $\\\\theta_2$`.

From TEX Before a user can include his graphics in his text file, he should load the PICTEX library. This is done with the three following `\input` commands:

```
\documentstyle{...}
\input prepictex
\input pictex
\input postpictex
```

`Prepictex` and `postpictex` files must be used to run PICTEX with LATEX.

Wherever the user wishes to include graphics, these are loaded by using another `\input` command, for instance:

```
..., these are loaded  by using another
\input command, for instance:

\begin{figure}
\input figfour
\caption{Bar plot of responses...}
\end{figure}
```

The user will take advantage of the `\caption` command to append captions to his graphics. He will then be able to use cross-references with the `\ref` and `\label` LATEX commands.

Two weak points remain. First, composing graphics is very slow, e.g., it takes up to ten minutes on a Sun3/50 workstation. Second, if graphics are too complex, TEX memory capacity is exceeded. Those problems are well known to PICTEX users. Maps of the USA are about the most complex graphics that can be drawn.

All graphic examples are extract from the S introduction manual. Graphics size is 330×203 points.

Finally, I wish to thank Ben O'Ferle for his help with my English.

Bibliography

Becker, R.A. *S: An Interactive Environment for Data Analysis and Graphics.* Wadsworth ed., 1984.

Chahuneau, F, and O. Nicole. "Une op"ration pilote "réseau" à l'INRA." *Technique et Science Informatiques*, 7 #2, 1988.

Wichura, M.J. "The PICTEX Manual." *TEXniques* 6, 1987.

Towards LaTeX 3.0

Frank Mittelbach
Electronic Data Systems (Deutschland) GmbH
Eisenstraße 56 (N15)
D-6090 Rüsselsheim
Federal Republic of Germany
Bitnet: `pzf5hz@ruipc1e`

Rainer Schöpf
Konrad-Zuse-Zentrum für Informationstechnik Berlin
Heilbronner Str. 10
D-1000 Berlin 31
Federal Republic of Germany

Abstract

LaTeX is a very valuable tool for document composition. As a TeX macro package, it is unique in its concept of logical commands, at the same time retaining enough flexibility with visually oriented commands to allow the user a relatively easy correction of an automatically chosen layout. This fact makes it far superior to the plain and AMS-TeX macro packages when it comes to professional applications.

Therefore the LaTeX re-implementation project is certainly one of the most important efforts to "expand TeX's horizon". This is the place to link TeX with the modern developments like SGML. This paper describes the current status of the re-implementation of LaTeX.

1 Objectives of the LaTeX project

LaTeX [6] was developed to serve the specific needs and designs found in documents of natural science, whereas the needs of other fields are more or less neglected. Many layouts and concepts cannot be realized in the present LaTeX, and even those that can be realized are often falsely flagged as "impossible in LaTeX".

One of the main objectives of the LaTeX project is, therefore, to redesign the style file interface by incorporating a broader spectrum of possibilities. At the same time, we try to structure this interface in such a way that it satisfies the needs of a designer. This means that desired layouts should be specifiable preferably through parameters and generic functions that allow a wide range of varieties.

An ensuing objective is the proper documentation of the new interface. It should guide the designer in the evaluation of a new layout, allowing him to use the full power of the interface within a short period of time.

As a third objective, we feel it necessary to reevaluate LaTeX's internal concepts and reverse those that have been proven inadequate.

In our talks at last year's conferences [8, 9, 10, 11], we presented a concept for a re-implementation of LaTeX. After discussing this topic, Leslie Lamport and one of the authors (FMi) agreed on a two-step procedure, first redesigning the style interface, and then enhancing the user interface.[1]

Further discussion throughout this year has shown that this plan is not feasible as the internal style file interface is affected too greatly by enhancements in the user interface, and vice versa. Therefore, we abandoned this idea and decided to merge both steps, at least temporarily.[2] As a result, the discussion then focused on three different major topics, the enhancements and changes to the user inter-

1. This was published as the update section in [8] and [7].

2. We feel that it is still sensible to defer certain parts of the revision to a later stage. However, this will only concern internals of the implementation and not induce any changes to the user or the style-designer interface.

face, the revision of the style file interface, and the re-evaluation of internal concepts.

In the following sections, we will discuss the topics which we have been concerned with since last year's conference and the state of their realisation.

2 The User Interface

Any change to the user interface of an existing program always opens the question of compatibility. While we judge this question as very important, we feel that it is not justified, for the sake of upward-compatibility, to leave all existing features untouched, even when they have proven to be inadequate. This means that we try to keep these sorts of changes small, and devise a possibility to emulate LaTeX 2.09 in the new LaTeX to allow processing of older documents with few or no changes.

2.1 Attribute concept. It has been generally agreed in the ongoing discussion that an attribute concept as supported by DCF GML [4, 5] and SGML [3] would be a great improvement. Since this is a major change, it was discussed whether such a concept would render the optional arguments obsolete. But as the number of LaTeX users is far greater than most people think, such an incompatible change in the syntax is not feasible. In addition, the old optional arguments and star forms provide convenient short forms for the most important attributes. While there have been several proposals for an attribute syntax as well as one prototype implementation, a final decision has not been made as yet. But it seems probable that this concept will only be available for environments, not for ordinary commands. We think that this will be sufficient since it is planned to provide environment forms of all text producing commands (cf. 2.7).

2.2 Robust and fragile commands. The distinction between robust and fragile commands will no longer be present. Instead, all text in moving arguments will be automatically protected against expansion.[3]

To allow the style file writer the specification of text that has to be expanded, there will be a command that removes or partly removes this protection.

We do not consider it necessary to give this feature to the user. Hence, this is not available in the user interface.[4]

2.3 Font selection. The new font selection scheme is already being distributed as beta test version for LaTeX 2.09 and seems to be working very well. It is also the basis for the `amsfonts` style option that makes the AMSFONTS collection available for LaTeX and is part of the AMS-LaTeX distribution [1].

Nevertheless, the current implementation should not yet be regarded as the final product. Time and users' demands will show whether it has to be improved.

2.4 Front matter. The specification of preliminary material is one of the parts which are not handled properly in LaTeX 2.09. Although this topic has not been discussed in depth so far, this might be one of the places where the syntax of the new LaTeX might differ in an incompatible way.

2.5 Tables. The extensions of the `array` and `tabular` environments by Frank Mittelbach [12] seem to be widely accepted. Several further extensions are conceivable, but this needs careful evaluation. There is also a new implementation of these environments by Denys Duchier that includes the extended syntax. This implementation looks very promising and can probably serve a basis for table handling in the new LaTeX.

We will provide a command to specify notes to tables working similar to the `\footnote` command inside a `minipage` environment.[5]

2.6 Math. The `amstex` style option for LaTeX 2.09 implements most of the features of AMS-TeX in LaTeX syntax (such as `\begin{align}...\end{align}` instead of `\align...\endalign`). As this is now being distributed by the AMS, users are able to typeset complicated math formulas in LaTeX without falling back to plain TeX's idiosyncrasies [1].

However, the implementation still has some loopholes that need to be eliminated in future versions.[6]

2.7 Text producing arguments. All commands with arguments in which the user specifies

3. The approach of LaTeX 2.09 to expand everything by default is counter-intuitive and a common source of nasty errors.

4. Expansion is normally necessary to cope with problems presented by the asynchronous output routine mechanism together with macros that change their contents, so that it is essential to write the expansion and not the macro name to a file, etc.

5. Using `\footnotemark` and `\footnotetext` commands inside a table that (additionally) has to be put inside a `minipage` environment is another counter-intuitive concept of LaTeX 2.09.

6. Some features do not work correctly in boundary cases. This is partly due to limitations in the current LaTeX, e.g., the primitive handling of `\begin...\end` (see below), etc.

text to be typeset (e.g., `\fbox`) will also be available in an environment form to allow their use in user-defined environments.

2.8 Verbatim input. The use of the `\verb` command will be possible in all circumstances.[7] A similar extension of the `verbatim` environment is not possible.[8] But this restriction is lessened by the possibility to use the environment form of the respective command in which the `verbatim` environment may be used.

2.9 Float positioning. LaTeX 2.09 was designed for documents containing only relatively few floats.[9] The new implementation will improve the float position algorithm and the user's control. This might include some sort of an 'h' option that really means "here".[10] There have also been proposals to prohibit the use of multiple captions within one float, thus allowing the document style to position the caption according to its own rules. But these topics have not yet been discussed thoroughly enough to present a final concept.

2.10 Bibliographies. The handling of citations and bibliographies will probably change to support several conventions. Since this topic depends on the development of the new BibTeX to some extent, it is not yet clear what is actually implementable.

Specific problems concerning citations and the interaction with BibTeX are discussed in [14] and in [13].

2.11 Omitting environment end tags. The implementation of a prototype for error recovery in case of unmatched `\end` tags 4.1 has shown that it is possible to implement the concept of implied `\end` tags. This feature will help in writing SGML parsers that produce LaTeX output. Whether the detection of implied `\begin` tags, e.g., the omission of the first `\item` in a list environment, can be easily implemented requires further testing.

3 The Style-Designer Interface

As we stated above, the main goal for the design of the style file interface is making it easily applicable. Therefore, the new interface will contain a lot more generic commands allowing for the specification of a wide range of layouts with a minimum of effort.

3.1 International language support. Support for more than one language (US English) was one of the key issues that triggered this LaTeX reimplementation project. In the new implementation, all textual representations in style files will be settable. While this is certainly not sufficient to support the different typographic conventions of different countries, it allows for typesetting foreign texts within the usual typographic conventions.

3.2 Hooks. For the implementation of certain layouts, it is often necessary to carry out specific actions at well-defined points, e.g., the footnote placement algorithm of this article has to be initialized at `\begin{document}`. For this type of application, many of the internal commands will contain hooks that allow the style file writer to add code to these commands without overwriting the original definitions.[11]

3.3 Generic section headers. One goal for the style file interface is to provide the designer with a generic heading macro which implements a broad range of layouts by varying certain parameters. It is clear that a proper balance has to be found between the internal complexity of such a command and the number of different layouts which are specifiable through it. Several different syntax proposals have been discussed so far, but the discussion hasn't reached a satisfactory conclusion, as yet.

The new mechanism will probably provide a specification for headings, in which the designer has complete control over heading layout, e.g., positioning supplied text[12], ornaments[13] and the like. It is planned to support the following general types of headings:

- Vertically oriented headings, where the heading is separated by white space from preceeding and following text.[14] There will be parameters to allow the heading to extend into one or both margins.

7. However, due to limitations of the TeX program itself, the use of multiple blanks in one `\verb` command will not be supported in all cases.

8. In LaTeX 2.09 neither the `\verb` command nor the `verbatim` environment may be used in arguments.

9. If, for example, the space for floats is larger than the surrounding main text, as is often the case in appendices of manuals, etc, LaTeX 2.09 is seldom able to compile the document without running out of memory space, even if the float parameters are given full flexibility.

10. We are aware of the problem of handling previously deferred floats of the same kind.

11. In LaTeX 2.09 a lot of style options are incompatible with each other, simply because they redefine the same internal macro.

12. For example, 'Chapter' in the `\chapter` command.

13. Rules and dingbats, etc.

14. This layout has already been realized to a certain extent in the `\@startsection` command of LaTeX 2.09. But this generic macro is not able to specify the layout of the heading (except setting the used font), so that it can not even be used for standard headings like the `\chapter` command.

- Horizontally oriented headings, where the heading is partially or fully placed into one of the margins beside the following text.[15] A critical problem with this sort of layout is to guarantee that enough space is available and that any necessary hanging indentation (for headings placed only partly into the margin) is applied up to the necessary point.
- Run-in headings, where the text following continues on the same line as the heading.[16]

The designer will be given tools for specifying the heading layout in an easy manner, so that it is possible to vary the layout depending on things like the length of the heading text, the presence or absence of a heading number, etc.

It is planned to allow for the specification of a minimal amount of text that has to follow a heading.[17]

3.4 Generic table-of-contents entries. What has been said in the last subsection about generic section headers applies to the formatting of entries in the table of contents as well. This has not yet been discussed in detail, but we hope that a proper implementation of generic section headers can be used as a starting point for generic toc entry formatting. The same mechanism can then be used for other types of tables as well (`lof`, `lot`, etc.).

3.5 Generic lists. Necessary extensions and corrections of the generic list environment have been already discussed in [8, 11]. There have been a few proposals concerning lists, but this topic needs further attention, as it affects a major part of most style files.

3.6 Parameter tables. There has been a proposal to group certain parameters into tables to make their structure and dependence visible. One item that will probably change on the style designer level in this way is the specification of default parameters for different levels of lists. But it is not yet clear, whether such a concept is implementable in an efficient manner.

3.7 Paragraph design. The specification of paragraph layout (such as different forms of ragged right typesetting) will be improved. The designer will be given the possibility to specify such layouts not only for the main text, but also for footnotes, floats, etc.[18]

3.8 *Toc levels.* There are book designs which require several tables of contents, e.g., one for the whole book and those referring to each chapter. This touches on the topic of auxiliary file handling (cf. 4.2) since the current mechanism does not allow more than one table per document. There will be support to select only certain entries of a tàble of contents for printing. A prototype implementation for this feature was written by Nico Poppelier.

3.9 Documentation. The interface between the style files and the LaTeX kernel will be documented properly. We will provide a complete description as well as a number of examples.

4 Internals

4.1 Error recovery. In the current LaTeX, an omitted or misspelled environment name usually produced a lot of error messages and often suppressed any further compilation of the document. In the new implementation, certain classes of environment errors are detected and corrected without damaging the output. For this complex, a prototype implementation is currently undergoing alpha testing. It implements the following features:

Incorrect `\begin` tag If the user misspells the name of the environment desired inside the `\begin` tag, an error message is generated and the offending `\begin` tag is ignored.[19] However, the user is allowed to insert the correct environment name by specifying `i\begin{`⟨*envir*⟩`}` in response to the error message.[20]

Incorrect `\end` tag When an incorrect `\end` tag is encountered, e.g., `\begin{bar}` ... `\end{foo}`, the new LaTeX tries the following recovery:

1. If `\end{foo}` is unknown, we assume that the user misspelled the name and recover by replacing `\end{foo}` with `\end{`⟨*curenvir*⟩`}`. This will produce an error message but will allow safe continuation of the compilation afterwards.
2. If `\end{foo}` is a legitimate `\end` tag, i.e., if the corresponding internal environment start com-

15. This sort of layout is not supported in the current version.

16. Again, this layout is provided in LaTeX 2.09 but does not allow specifying the layout of the heading, e.g., punctuation marks at the end, underlining, etc.

17. In LaTeX 2.09 this is the fixed amount of two lines of text. It might be possible that a more general implementation will fail in special cases due to TeX limitations.

18. In LaTeX 2.09, this is not possible without redefining several internal macros, because the paragraph shape parameters are reset at several points to fixed defaults.

19. Of course, this mechanism will be triggered only if the user did not exchange one environment name for another.

20. In LaTeX 2.09, this response would result in TeX error messages at the end of the compilation.

mand is defined, we check to see whether there are any unresolved `\begin{foo}` environments.

(a) If so, the currently open environment is closed by inserting an `\end{`⟨*curenvir*⟩`}` tag.[21] Afterwards `\end{foo}` is tried again. This mechanism will close all open environments until the correct one is found. Depending on the status of an internal variable, this will either produce an error message, or a warning message, or will be executed silently to allow for the implementation of implied `\end` tags.

(b) If there is no open `\begin{foo}` we simply ignore the `\end` tag after issuing an appropriate error message. The underlying idea is that this `\end` tag is probably left over after moving some text in the source around.

4.2 Auxiliary file handling. We will implement a two-step approach for `.aux` files where all information is written to an intermediate file which is then copied to the 'real' `.aux` file at the end of the run. As a result, there will be only one `.aux` file instead of the many files produced by LaTeX 2.09 when the `\include` command is used. The new scheme will make it possible to preserve cross-reference information if a compile run ended prematurely.[22] In addition, it will be possible to detect whether `\include` files have to be re-compiled because of changes in other parts of the document, or not.

4.3 References. The use of symbolic references will be extended to include textual references, if desired.[23] Whether this will result in some changes concerning the user interfaces not has not been discussed so far. If it is feasible, we will also provide the possibility for hierarchical references.[24]

4.4 Page selection. To provide easy access to the different parts of a document at the printing level, we plan to record certain document structure information in the `\count` registers 0–9. Besides the usual page counter in `\count` 0, this might include the physical page number, the current chapter, section, or subsection, ... number, to allow the printing of, e.g., Chapter 6 or "all preliminary pages"[25] by giving a simple page selection pattern to the printer driver.[26]

4.5 Plain TeX compatibility. We tend to build up the new LaTeX from scratch, i.e, not to read in the plain TeX format (or a nearly identical variant) as a basis when building a format file.[27] This does not mean the useful functions, `\mathchar` definitions, etc. of `plain.tex` will be discarded, but concepts which are obsolete in the LaTeX environment or macros that can be implemented in a better way will be replaced or removed.

5 Beta testing

It took LaTeX about three years to develop into a stable system. To avoid needing a similar period of time when switching to the new version, we plan to run the new version throughout the development at a few selected sites for beta testing. So, if you are a maintainer of a TeX installation and think that you can persuade your users to play willing (or unwilling) guinea pigs, please, contact one of the authors. Your installation should have the following characteristics:

- A running TeX 3.0 preferably (but not necessary) with drivers supporting the virtual fonts introduced with TeX 3.0.
- A fair amount of LaTeX processing in a fully supported LaTeX 2.09 environment (including a customized Local Guide) and at least one user with some experience in style writing.
- A working eMail connection.
- A maintainer (you) who is willing to
  - provide backups and fast user support in case something goes wrong
  - send in bug reports, if necessary
  - compile new formats when updates arrive.

21. The implementation of this part of the recovery is not as straight forward as it may seem. The term ⟨*curenvir*⟩ does not necessarily refer to the innermost open environment because it may be possible that a user defined environment calls other environments in its body. In such a case, the calling environment has to be closed first, because the inner environments are resolved in its `\end` tag.

22. The current implementation writes directly to the `.aux` file, so that its contents are damaged or lost when an error occurs.

23. In the current LaTeX, references can only be made to counter values, or more exactly, to a combination of counter values. This will lead to problems if document styles decide to use unnumbered headings, for example.

24. In LaTeX 2.09, this has been realized for the `enumerate` environment, to some extent.

25. In LaTeX 2.09, it is often not possible to print a specific set of pages that have the same TeX page numbers as other pages that come earlier in the document.

26. With most printer drivers, the first page to print can be selected by specifying a pattern to be matched against TeX's `\count` registers 0–9. We propose that drivers also allow for the specification of a pattern for the pages to be included.

27. Currently, LaTeX is built on top of `lplain.tex` which differs from `plain.tex` in only a few places. This means that even the funny keyboard support (for SAIL terminals) is included that will give you in certain situations a '⊕' character if you key in `$^^M$`.

The work that has to be carried out by the beta testers should not be underestimated. It is probably a time-consuming commitment since a lot of organizing is usually involved. Nevertheless, we hope that there will be enough people around willing to help us in this stage of development, so that we can finally return a product to the user that will have the same success as the current LaTeX had.

Throughout the beta testing phase, LaTeX will have to show its abilities in real life situations. As TeX and LaTeX are currently finding their way into new areas, we hope that people from these fields will take the opportunity to test whether the new LaTeX matches their needs by participating in the testing phase. This is the time when it is still possible to correct errors before they become established as unfortunate facts.

References

[1] American Mathematical Society, Providence. *AMS-LaTeX Version 1.0 User's Guide*, July 1990.

[2] Brüggemann-Klein, Anne, editor. *1989 EuroTeX Conference Proceedings*, 1990. To appear.

[3] Bryan, Martin. *SGML: an author's guide to the standard generalized markup language.* Addison-Wesley, Woking, England; Reading Massachusetts, second edition, 1988.

[4] *DocumentComposition Facility: Generalized Markup Language Starter Set User's Guide.* New York, fifth edition, May 1987.

[5] *Composing Documents with the Generalized Markup Language.* International Business Machines Corporation, New York, second edition, March 1988.

[6] Lamport, Leslie. *LaTeX: A Document Preparation System.* Addison-Wesley, Reading, Massachusetts, 1986.

[7] Mittelbach, Frank and Rainer Schöpf. "A new font selection scheme for TeX macro packages — the basic macros." *TUGboat*, 10(2):222 - 238, July 1989.

[8] Mittelbach, Frank and Rainer Schöpf. "With LaTeX into the nineties." In Thiele, Christina, editor, *1989 Conference Proceedings*, volume 10#4 of *TUGboat*, pages 681 - 690. TeX Users Group, December 1989.

[9] Mittelbach, Frank and Rainer Schöpf. "LaTeX dans les années 90." *Cahiers GUTenberg*, (6):2 - 14, July 1990. French translation of [11] and of some parts of [10].

[10] Mittelbach, Frank and Rainer Schöpf. "LaTeX limitations and how to get around them." In *1989 EuroTeX Conference Proceedings*, Anne Brüggemann-Klein [2].

[11] Mittelbach, Frank and Rainer Schöpf. "With LaTeX into the nineties." In *1989 EuroTeX Conference Proceedings*, Anne Brüggemann-Klein [2].

[12] Mittelbach, Frank. "A new implementation of the array - and tabular - environments." *TUGboat*, 9(3):298 - 314, December 1988.

[13] Read, David. "Towards BIBTeX style-files that implement principal standards." *TeXline*, (10):2 - 8, May 1900.

[14] Wonneberger, Reinhard and Frank Mittelbach. "BIBTeX reconsidered." In Guenther, Mary, editor, *TeX 90 Conference Proceedings*, volume 12#1 of *TUGboat*, pages 111 - 124. TeX Users Group, March 1991.

Public-Domain, Documented Implementations of TEX and METAFONT for VAX/VMS

Brian Hamilton Kelly
Royal Military College of Science
Shrivenham
Janet: `tex@uk.ac.cranfield.rmcs`

Adrian F. Clark
University of Essex
Colchester
Janet: `alien@uk.ac.essex`

Abstract

In this paper, a VMS implementation of the most recent versions of TEX, METAFONT and all the important support utilities are described. Some of the important features include TEX sensitive VMS editors, command-line interfaces, and complete documentation, both HELP and printed manuals.

Two public-domain implementations of TEX on VAX/VMS have been in use for some time: the first was by David Fuchs of Stanford University and the second [*TUGboat*, V10], derived from Fuchs', was by one of the authors of this paper. The latter provided an interface to two of the most popular editors on VMS (TPU and EDT) and an expanded memory. However, most of the additions were written in Fortran rather than Pascal. In this paper, we present a compatible but greatly enhanced VMS implementation of TEX 3.0. Equivalent changes to the latest version of METAFONT are also summarised. This implementation was carried out by the first author, again derived from Fuchs' earlier work, with occasional assistance from Niel Kempson, also of RMCS; the second author merely hassled for particular features to be included.

Editing after Errors

The most significant enhancement again concerns the editor interface. When TEX encounters an error in an interactive run, typing "`e`" (or "`E`") to its prompt invokes an editor. The editor which is invoked may be one of the standard set of VMS editors (TPU, EDT, LSE, `TECO`), or an unsupported editor such as `SOS`. The one to be used is specified via the logical name `TEX$EDIT`; this is analogous to other VMS utilities such as `MAIL`. (If this logical has not been defined, or expands to something that this implementation can't handle, the default TEX response ensues, the user being told whereabouts to edit his/her file.) The four standard editors are specified by setting `TEX$EDIT` to one of `CALLABLE_TPU`, `CALLABLE_EDT`, `CALLABLE_LSE` or `CALLABLE_TECO`, so the desired editor could be defined *e.g.*

```
$ DEFINE TEX$EDIT CALLABLE_TPU
```

As these names suggest, the appropriate editor is dynamically linked to TEX and then invoked. Wherever possible, the cursor is moved to the point in the text at which TEX detected the error.

- For TPU and LSEdit, the cursor *is* always positioned correctly.
- In the case of TECO, the user is provided with a macro (`M1`) which moves to the correct place when it is executed.
- Since EDT does not really support the ability to position the cursor from outside its "change" mode, this is done by sleight-of-hand. The method adopted *does* position the cursor, but results in EDT generating an error message on entry; a message which can safely be ignored.

The alert reader might ask "What about users' initialisation files?". In the case of LSEdit and TPU, the editor interface need only invoke the editor and pass in the desired cursor position with the `/START_POSITION` qualifier, so any initialization file defined through the relevant logical will still be included. With EDT, the `/COMMAND` qualifier is used to pass the name of the file which positions the cursor correctly; this will be written to `TEX$EDTINI.EDT`; thus it may be written elsewhere if `TEX$EDTINI` is defined as a suitable logical name. If the logical name EDTINI is defined, then the initialization file arranges to execute the commands in the file indicated by this logical after the cursor has been positioned. With TECO, things get more complicated: according to the documentation, the callable version of TECO is supposed to accept a parameter defining the initialization file; however, this doesn't work. Many hours spent poring over a trace of the execution of TECOSHR revealed that the code to interpret this third parameter is bypassed. Therefore, TeX redefines the logical TEC$INIT to reference its initialization file (which is written to TEX$TECOINI.EDT). Any user's initialization previously defined in this logical will first be included in the new file, before the cursor positioning macro is defined into register Q1. After editing is completed, the original definition of TEC$INIT is restored, if there was one.

Any other translation of TEX$EDIT is considered to be a DCL command to be executed in a subprocess. (This may, of course, be a command procedure, in which case the equivalence string will commence with an "@ character.) Three arguments are passed to the DCL command: the name of the file to edit, and the line and column at which the error was detected. (These latter numbers are one-based, *i.e.* the first character of the file is at line 1, column 1.) TeX examines the status returned by this DCL command or command procedure. Any error status causes TeX to revert to its default behaviour, exiting after telling the user where the file should be edited.

After the error has been corrected, the editor may be exited, creating a new version of the input file. At this point, control returns to TeX, so that the user may fix the problem within TeX and continue with the run if he or she so wishes. This feature permits several errors to be corrected in a single TeX execution; however, it is important to realise that TeX continues to process the *original* (*i.e.,* incorrect) file. However, further invocations of the editor will read in the *latest* version of the source file. (In the case of LSEdit and TPU, this implementation is able to recognize that the user has quit from the editor without writing a new file, and under these circumstances the original file will continue to be read until a new version is written.) It is not possible to make this discrimination with other editors, although a warning status returned by an editor run in a sub-process will be interpreted in this same manner. The dvi and log files continue to be written in the usual fashion.

Some criticism has been directed at the authors for continuing to process the TeX source after performing an edit. It has been suggested that the description of the "E" option in *The TeXbook* [p.32] implies that TeX *must* exit after performing the edit, just like typing "X." However, experience has shown the usefulness of being able to perform further edits (not the least of which is that LaTeX's auxiliary files get completed correctly). The user can always type Control-Y to abort TeX if he/she considers that the error is too severe to permit sensible recovery after editing. All files created by TeX itself in the current run (dvi, lis, aux, etc.), will then be discarded, although the new source file(s) written by the editor will be retained.

Diagnostic Feedback whilst Editing

Another feature of this implementation, relating to the language-sensitive editor, LSE, is unconnected with the use of that editor from TeX's error prompt. However, it *does* prove extremely useful when used the other way round, *i.e.* when TeX is invoked from *within* the language-sensitive editor by use of the latter's COMPILE command. The definition of "language" LaTeX for LSE includes the following:

```
/CAPABILITIES=DIAGNOSTICS -
/COMPILE_COMMAND= -
   "LATEX /BATCH 'LSE$FILE'" -
```

(Similar definitions could be provided for the "languages" TeX and SliTeX.) As TeX runs, in batch mode, all error messages are written in TeX's normal format to the log file, and are written additionally in the diagnostics file in a format defined by DEC. After processing has been completed, control is returned to LSEdit and the latter's REVIEW command will then read in the diagnostics file. By using the NEXT STEP and PREVIOUS STEP commands, each error report in the REVIEW window may be highlighted; the GOTO SOURCE command will then position the cursor at that point in the relevant source file at which the error was detected. (It is *not* recommended that LSEdit be invoked from within TeX when generating a diagnostics file, because the latter will not be available to LSEdit until TeX completes its processing.)

Command-Line Interface

Another major improvement offered by the new implementation is that it provides a proper command-line interface to DCL via a .CLD (command definition) file. The DCL commands defined in this file may be made available to the user's process by saying:

```
$ SET COMMAND TEX
```

This is usually done in the command procedure which establishes logical names and DCL symbols needed by TEX. However, this DCL operation is quite slow, so it is better performed (once only) by the system manager and installed in the standard SYS$SHARE:DCLTABLES.EXE file.

The DCL command TEX supports the following qualifiers:

/BATCH which tells TEX to start running in batch mode

/BIG to use the big-memory version of TEX

/DIAGNOSTICS=*filename* for specifying the name of the LSE diagnostics file. If no file specification is provided, TEX will use the name of the first file input, with an extension of .DIA

/DVI_FILE=*filename* for overriding the default name of the resulting dvi file. If this qualifier is not supplied, the name of the dvi file is the same as that of the first file input, with an extension of .DVI
This qualifier may be negated (/NODVI_FILE) and will then suppress generation of the dvi file. This can be useful when making the first one or two passes over LATEX source files to generate all cross-references and BIBTEX citations. The log file will report that no dvi file was generated, but it will also say how large it would have been.

/FORMAT=*filename* to specify the name of the format file; this defaults to
TEX$FORMATS:PLAIN.FMT
Other DCL verbs can be defined such that LATEX invokes the image of TEX with /FORMAT=TEX$FORMATS:LPLAIN.FMT as default, etc.

/INITEX to invoke iniTEX rather than TEX; iniTEX is used to build format (.FMT) files. As an amusing aside, when this command definition file was first written, it defined INITEX as a verb. Later that night the operators discovered that attempting to INITIALIZE a magnetic tape for backup actually invoked iniTEX: DCL only looks at the first four characters of a command verb!

/LOG_FILE=*filename* to override the default name of TEX's transaction log-file. If this qualifier is not supplied, the name of the log-file is based on that of the first file input, but with an extension of .LIS.
This qualifier may be negated (/NOLOG_FILE) to suppress generation of a log file.

/TRIP invokes a version of TEX which has arrays of the correct sizes for the TRIP test.

The TEX command also accepts a single parameter, which is normally the name of the file to be processed. Its default extension is, of course, .TEX. However, the entire command line is used for this parameter, and is folded to lowercase (DCL having "kindly" converted it to uppercase) before TEX sees it, so that one can include most TEX commands on the DCL command line. (They must, of course, be commands which contain no upper-case letters). Recent development work on METAFONT revealed that it was easier to require that strings containing multiple commands, with embedded spaces, should be entered as a standard DCL quoted string, replacing the usual single parameter of a file name. Since this quoting will preserve lowercase letters correctly, a future update of this TEX implementation will remove the folding to lowercase; this new version *will* permit the insertion of uppercase letters where desired in command lines.

If any input file cannot be found in the current directory, TEX looks in the directory (or all of the directories in a comma-separated list) specified by the logical name TEX$INPUTS. Similarly, TEX uses the logical name TEX$FONTS to specify the directory or directories in which to look for font (.TFM) files.

TEX returns a status to VMS as it exits. This will be one of STS$K_SUCCESS, STS$K_WARNING, STS$K_ERROR or STS$K_FATAL, depending on whether TEX detected errors and how they were handled. Although this status can be tested in the usual way, TEX inhibits VMS from outputting the corresponding error message.

Changing change files. As the changes are supplied, there is a master change file, TEX.CH and subsidiary change files, such as TEX-INITEX.CH for building iniTEX, and TEX-BIGTEX.CH to enlarge the size of many of TEX's internal tables. There is an accompanying program, WMERGE.C, which can be used to merge the change file and subsequent modifications to it.

This WEBmerge program must originally have been written in WEB, since the original C source contains references to wmerge.web, but there is no indication of the author. It was extended (and debugged) by BHK, who added facilities such that it could match WEB's various @x,@y and @z commands within change sections themselves. By applying these subsidiary change files to the master file, change files are created for generating the different variants from the one master TEX.WEB.

The reason for distributing the big TEX changes separately is that the version with smaller memory requirements runs a little faster on smaller VAXen (approximately 16% faster on a VAX-11/750).

At present, no provision has been made for applying the changes to generate TEX-XET, but it is under consideration.

METAFONT

A similar implementation of METAFONT has also been performed. This supports an essentially identical editing interface, but controlled by the logical name `MF$EDIT`. It also has a similar command-line interface, supporting the following qualifiers:

`/BASE=`*filename* specifies the name of the base (analogous to TEX's preloaded format) file to be loaded. This defaults to the file `MF$BASES:PLAIN.BASE`. Again, another verb can be defined to run with CMplain by default.

`/BATCH` indicates that the run is to take place in batch mode.

`/DIAGNOSTICS=`*filename* specifies the diagnostic file for use with LSE.

`/GF_FILE=`*filename* specifies the name of the file to receive the *generic font* file, which can be further processed to a `pk` or `pxl` file. This defaults to the name of the first file read in, but with an extension of `.`⟨*mag*⟩`GF`, where ⟨*mag*⟩ reflects the pixels/inch of the file; the default will thus be `.2602GF` if `mode=proof`.

`/NOGF_FILE` suppresses output of the `gf` file.

`/INIMF` to invoke iniMETAFONT, which can build base (`.BASE`) files

`/LOG_FILE=`*filename* to specify the name of the file to be used for METAFONT's log-file. This defaults to the name of the first file read in, but with an extension of `.LIS`

`/NOLOG_FILE` suppresses output of the log file.

`/TRAP` invokes a version of METAFONT which has arrays of the correct sizes for the TRAP test

Analogously with TEX, METAFONT uses the default extension `.MF` on input files. The current directory is scanned for an input file. If it is not found there, the comma-separated list of directories specified by the logical name `MF$INPUTS` is searched.

Graphics support. There is one major difference between TEX and METAFONT. When designing fonts, some means of interactively viewing the glyphs is invaluable. This implementation allows previewing on several graphics terminals; the device to be used is selected by assigning a mnemonic to the logical name `MF$TERM` before invoking METAFONT. If this logical is not defined, or its definition doesn't correspond with one of the recognized values, then graphics output is suppressed. The recognized mnemonics are (note that these *are* case-sensitive):

`go140`	GraphOn 140 terminal
`gp`	Northwest Digital Technology's *Graphics Plus* terminal
`tek`	Tektronix 4105 (or compatible) terminal
`Vis`	Visual Technology's *Visual 550* terminal

Future Directions

Both these implementations have been in use for a reasonable amount of time at a number of sites and no problems have been reported. They are available from the UK TEX archive at Aston, both as change files and as object modules. Don Hosek has recently been extending the change files so as to permit the use of different logical names (some people believe that they should obey DEC's stricture that logical names containing dollar signs are private to DEC themselves). He has also made a major contribution by removing graphics support into separate shareable images (one for each terminal type supported) which means that a new graphics display may be supported without the necessity of revising the METAFONT change file and recompilation. His extensions to this implementation will then become the default for the VMS-world.

Bibliography

Clark, Adrian. "An enhanced TEX-editor interface for VMS," *TUGboat* 10(1), April 1989, pp.14–15.

Knuth, Donald E. *The TEXbook*, Addison-Wesley, 1986.

Post Congress Tristesse

Malcolm Clark
Malcolm Clark, *TEXpert systems ltd.*, PO Box 1897, London NW6 1DQ
email: `malcolmc@uk.ac.pcl.mole`

Abstract

Almost every conference now has its proceedings made available to attendees, and perhaps even published to a wider audience. This is a description of the production of one conference proceedings, charting progress and pitfalls from the conference itself to the final bound proceedings being mailed to the attendees. It attempts to suggest that some aspects of the publishing process are perhaps more awkward than we might expect, but equally that there are unexpected rewards as well.

Introduction

Because we deal with an electronic typesetting system we may subscribe to the widespread myth that the progression of documents from manuscript to published form is now easier and more straightforward than it was formerly. The original text may be captured directly at the keyboard, then massaged into its correct grammatical and syntactical form until eventually something emerges which we may happily present to a publisher (or perhaps a local print shop). Because most of the process is (almost) under our own control we begin to wonder what all the fuss is about. On the other hand, few people really do complete the path from manuscript to book, although most of us are sure that it represents only a few more small steps forward.

The gulf between this make believe world and an instance – the production of the proceedings of the TEX88 conference – may help to expose the myth (and create other ones). It is a single instance, and of course is straightforward to dismiss through its own blend of chance and circumstance.

Perhaps I should have learned by my own previous experiences. The first conference proceedings I edited and produced through TEX was in 1984. This took an agonising 3 years (April 1981 to May 1984), but I at least had the excuse of having to key in the manuscripts myself, struggle with a buggy implementation of TEX80 and placate (apparently ungrateful) authors. Preview devices were few and far between and my only output device was an Autologic APS-μ5. In other words, I went straight to bromide, and often from there to the bin. But eventually the proceedings were published, and I still think they are quite attractive. There are some things I would change, but the constraints of CDC Cyber/TEX80/APS-μ5 would have made such refinements extremely difficult. Shortly after, I typeset a book of thermodynamic tables using the Cyber/TEX80/APS-μ5 combination which were subsequently published by Pergamon.

In 1985 – 6 I had the temerity to write a Fortran book, which I produced entirely through the medium of TEX on a PC. The very first versions of TEX on IBM personal computers had appeared. This was my first encounter with stable and reliable TEX. The Fortran book was laser "typeset" on an Imagen 8/300, and I had a preview facility on the PC. Of course, in those heady days, we thought that laser printed output was good.

By 1986 I therefore had some experience not only with producing conference proceedings, but also with producing a whole author-prepared book through TEX. I thought I had solved most of the problems inherent in the publishing process. A publisher was now merely the outfit handling printing, binding and distribution. Everything else could be under my control. This of course was the sort of image being presented at the time as "desktop publishing" took hold. Having "published" from the desktop before the terms were invented, I felt I knew it all.

Theory

Obviously a conference proceedings is at least as easy to produce as a book. After all, the manuscripts are prepared by someone else, so the creation or origination is already taken care of. All we have to do is take the "compu-scripts" and pipe them through TEX, and lo, a few minutes later, out will come the proceedings, which can then be dispatched to the publisher. Within a few weeks of the conference, the bound volume will be winging its way to the participants. Excellent theory. But consider the following.

Ennui. If the conference organizer is also the editor of the proceedings, he or she may feel that once the last participant finally departs that it's over. The adrenalin levels reduce and you can start normal life again. Wrong. The difficult bit is just beginning. First you have to extract the papers from the authors, while at the same time fending off the participants who want to know when the proceedings will be out. The few authors who have already presented you with their papers also cannot see why there is any delay.

The paper isn't finished yet. The authors may not yet have finished writing their papers. What they presented is likely not quite what they wrote down. As an editor, you can of course take a really hard line and demand that all papers have to be in before they may be presented. In my view this is unrealistic and counter productive. TEX users are known by their cooperative and amiable natures (the one or two exceptions are all the more cuddly and lovable for not adhering to this norm).

If you are trying to arrange a conference, you want as many papers as possible. It's an eerie feeling scheduling three days of talks long before anyone says they are coming. The first few months of organisation are very lonely. In general you are delighted to extract a plausible title and an abstract. Demanding a fully written paper is expecting a great deal. In any case, I would argue that the paper might so benefit from the delivery that it requires amendment, enhancement, improvement, or even rewriting. If the conference is any good it will expose the authors to some new ideas which likely have relevance to their own specialties. So at best we can expect some editing by the authors, which will take up some time; at worst, we can expect to see the paper actually rewritten.

TEX or LATEX? Since we have (at least) two "standards," TEX and LATEX, some authors prefer one, and some the other. At the time of the conference I had very little experience with LATEX style files, and didn't really want to convert to LATEX. The "standard" LATEX book style seemed clumsy and foreign to me. At the time it seemed less arduous to change the LATEX "encoded" documents to plain than vice versa. Fortunately no-one submitted an article in something completely bizarre. At no time did I assume I could produce a heterogeneous volume which combined both plain TEX and LATEX – although that is what *TUGboat* does.

Of course it was always the intention to produce the whole proceedings with TEX. As a matter of principle, but also because many of the papers discussed aspects of TEX which could only be illustrated by using TEX itself.

The Papers Trickle In

Almost all the papers arrived electronically, either as floppy disks (of both persuasions), or as electronic mail. I had no difficulty in transferring all of these to the Macintosh with Kermit. There was still one slight problem, since one of the major relays used by the UK for the transmission of electronic mail is Rutherford, where due to an incompatibility between ASCII and EBCDIC certain characters are mapped incorrectly. Incorrect mapping is no great problem, provided it is one to one. It isn't. Fortunately, knowing the problem, it is possible to correct it manually.

And one or two came on paper, or not at all. This last category represents a problem. Naturally I wanted the conference proceedings to reflect what went on at the conference – or, at least, the formal part of what went on. I was reluctant to erect some sort of refereeing structure. It was not an academic conference, and refereeing seems inappropriate. Therefore the papers are not going to be screened beforehand, except in a very rudimentary way. Once a paper has been presented it should be represented in the proceedings, if the proceedings are supposed to represent the conference. Two papers which were on the program but not presented do not appear in the proceedings. One extra paper does appear, but I plead extenuating circumstances and editor's privileges (there's no point having power if you don't abuse it). The extra paper was from Poland, and discussed some aspects of the adaptation of TEX to Polish typographic needs. This seemed so appropriate and apposite for a European conference that, despite the fact that the authors could not present the paper, I included it.

This still leaves the problem of the papers which don't arrive. There are several reasons why this might occur. Some papers just never do get written, and some did fall in that category. But I still had the abstract. So the abstracts went into the proceedings. The other non-papers were those which were written, but went elsewhere because of the delay. Really, only one fell into that category.

The particular paper contained a great deal of rather Unix-specific, LaTeX-specific graphics and had proved to be beyond the capabilities of any driver to process (in its entirety). It was therefore a paper I put to the bottom of my pile, and when I came to retrieve it, it had gone (it is available in another form now).

A Matter of Style

I had decided not to issue a set of style instructions to authors, except to indicate that provided they could provide an ASCII text, with or without markup, I would handle it. This seemed an almost achievable base level. My own experience, borne out with conversations with others, seemed to be that authors would ignore the stylistic recommendations anyway. There are two stylistic features at work here. There is the 'consistency' aspect of style, and the "use of language" aspect

Consistency. The "house style" may require that we always refer to a device independent file as a `dvi`, or `.dvi` or even `dvi` file. We can always say that provided we encode the phrase as `\dvi` it really doesn't matter. The "style" will pick it up and put it in correctly. But first you have to have the author encode it as `\dvi`. They might, and they might not. Were it not for authors and their whims and vagaries, producing books would be simple. There are also the other elements of expanding "*e.g.*" to "*for example*" and "*i.e.*" to "*that is*."

Language. "Use of language" is a fascinating area. The language of the conference was English, for which I make no apologies; English is sufficiently well-understood in the rest of Europe to make it close to a *lingua franca*–ignoring the obvious oxymoron. But as an international conference, the first language of many of those offering papers was not English.

What then do we do with the infelicities of grammar and style? I find this a thorny problem. I have always resented editors who turn my own idiosyncratic style into bland Euro- (or perhaps Oxo-) english. I prefer to think that what I am reading was written by an individual, and by neither a committee nor a machine. My criterion was therefore "is the meaning clear?" The reader will be able to find sentences in the proceedings which are a little quirky, and which may even amuse. I argue that this is no bad thing, unless it obscures the meaning for those who do not appreciate the full weft and warp of English. These considerations are as applicable to those who claim to possess English as a first language.

I would like to think that the reader can detect those instances where the authors are not native English speakers, and will therefore appreciate all the more the heroic job those authors will have done in expressing themselves in an alien mode.

My own addition to house style was to remove as many accents as possible. Just because TeX allows you to say "naïve" instead of "naive" does not mean you should. English had the foresight to dispense with all this fancy foreign frippery, while accomodating hundreds of "foreign" words. There is no need to lard on the accents to a language which does not use them. If we can distinguish and pronounce rough, cough, chough, lough, through, bough, hough, enough, slough, Slough and tough, as well as bow, cow, low, row, throw, how, hoe, enow, tow, toe, sloe and slow, we really don't need ï in "naive."

Look and feel. There is a rather grander aspect to style, the sort that is enshrined in the look of the book. We each have feeling of what constitutes a sentence or a paragraph, and how we assemble these elements into larger units. It should not come as too much of a surprise that there is little agreement on these "feelings." Some people write very long sentences or very long paragraphs. TeX hardly encourages paragraphs over about two or three pages long. The subdivision of text into units (like sections and subsections) is a personal business. An article made up of lots of sections and subsections looks very different from one with few. The differing "granularity" may mean that the interplay of white space obscures what the editor fondly thought was the underlying homogeneity of layout style. A conference proceedings is likely to be an extreme example. A book by a single author is far less likely to exhibit these pathological symptoms.

Publish, Please

Although it may not have been obvious to the authors, I was beavering away, assembling, editing and negotiating. I had approached a couple of publishers with the project, choosing those who I knew to have some "interest" in TeX. Addison Wesley (who produced the first European TeX Conference Proceedings) didn't want to know; similarly, Wiley's "didn't do conference proceedings." So I went back to my old reliable publisher, Ellis Horwood. Provided I produced the typeset pages, they would do the rest. Naturally I would present them with some interim examples to confirm that I was not too far from their house style.

The printing. In the beginning, I thought we would laser print the final copy. Although I knew a few people who could take TEX dvi and typeset – a few of them had been at the conference: Cambridge University Press, Imprimerie Louis-Jean, Stürtz, American Mathematical Society. I had not done it myself since the days of the APS-μ5. This is the one area where the delay in production was an advantage. Had I managed to get the proceedings out by Christmas 1988 (once an idle dream), they would have been laser set. The extra effort in going to typeset in late 1989/early 1990 was minimal. The increase in perceived quality seems enormous.

I had become convinced that laser printing was only suitable as a proofing stage. "Masterpieces of the publishing art" were simply not possible with 300 dpi printers. In addition, I had come to the conclusion that Computer Modern looked really good at typeset resolutions, but that it had never been intended for low resolution printing.

I admit I look at the other TEX conference proceedings (the two European conferences and the last two TUG conferences), and I'm inclined to think that they do not show TEX at its best. Having said that, I open myself up to similar criticism. At worst, I think *my* proceedings are merely mediocre. They are still the best produced of the bunch.

I decided to phototypeset, using the University of London Computer Centre's Linotron 300. Members of the typesetter's user group had used it to produce TEX output. A couple of friendly Vaxes had been monopolised for a weekend to produce the 1270 dpi TEX fonts, which were subsequently used in a number of publications. I was assured that it was a well-trodden path.

Production Problems

I was fortunate that none of my authors provided manuscripts which required one of the currently fashionable "gross" TEX versions. Everything could be processed on my 1 Mbyte Mac Plus with *Textures.* By the end of the project I also had access to a 4 Mbyte Macintosh IIci, which was appreciably faster. The last paper to arrive (by email from Austria), Michael Ramek's, was processed entirely on the IIci. I had always expected that Michael's paper would present a problem. I had erroneously supposed that he used a "gross" TEX. The complexity of the graphs and chemical structures which he produced through TEX macros encouraged my view. In the event he made such an excellent job of coding the macros that there was no problem with a "standard" memory version.

The typesetter strikes back. Although I had not finished editing all the papers, most were done by the time I had arranged to use the ULCC typesetter. I therefore started to produce some bromides. After all, there should be no problem, others had done it before me. To my surprise and amazement I discovered that "TEX" fonts meant just that. None of the LATEX fonts were available, nor were the logo fonts used for the METAFONT logo. After a great deal of ferreting around I found the necessary fonts at the Open University and was able to ship them not only to ULCC, but also to the Aston Archive.

But even shipping the dvi to ULCC had presented a problem. My IIci had software which allows a large file to be shipped in a matter of seconds to my host Vax. The Vax could then be used to FTP the file to ULCC, again in a matter of seconds. Unfortunately, this two stage process appeared to damage the file in some way, so that it was not recognisable when it reached the ULCC microVax. I could have solved the problem eventually, but instead I choose to use another route, using Kermit to transfer the dvi file directly from the Mac to the microVax which was the typesetter's front-end. This worked, but it took an enormously long time: a couple of hours for a file. I had been advised to send files of no more than about 40 pages. Needless to say, everything went through at least twice. Just as it is impossible to see the mistakes in preview on the screen, it is also impossible to see all the mistakes in the laser printed copy. Once in bromide, some more of the remaining mistakes leap out before you.

An exceptional story. Rick Simpson's was one of the exceptional papers. The story is a little involved. His paper discussed three ways of inserting diagrams into a TEX document. Two of the ways involved using a prepocessing program which generated both METAFONT code and LATEX instructions. The METAFONT code was then processed to produce a single character which did all the things that LATEX couldn't – for example, graph elements. The third way was to take a scanned image and convert it to "packed pixel" format – in other words, another character which could be handled "in the normal way." Somehow I had managed to acquire an IBM 6150 running AIX, as well as Rick's own port of TEX to that machine. I could therefore edit his paper, which he provided on floppy, readable by AIX. This still didn't solve the problem of output, but I thought I might always either "cut and paste" or have him laser print it from my edited version. I managed to move the text of the paper to a DOS disk (AIX allows you to create DOS disks), which could then be moved, via Kermit, to my Mac. Of course, the diagrams were not going to travel quite so easily. I reasoned that since I had the METAFONT files it should be possible to run these through METAFONT to produce a complete set of pk files which a suitable driver could process.

I persuaded Philip Taylor to generate the pk files: I knew he had some experience with METAFONT, and since he used Vaxes I was confident that the file transfer to ULCC would not be difficult. He duly produced both 300 dpi and 1270 dpi pk versions of the METAFONT files, and ran the paper with the illustrations and the other pk file on his local machine to produce 300 dpi output. When he tried to generate the 1270 dpi version, the device driver fell over and died. Reading Rick's paper a little more closely, I noted that the device driver would have to be able to handle very large characters. Evidently this one had been unable to do so. I would have been happy to have some of the paper typeset and to include as much of the pk "bits" as possible, but unfortunately Phil had lots of other things to do. In the meantime, Rick had sent me an up-to-date laser printed copy of his paper. I resorted to typesetting the text and using cut and paste for the graphics.

I see no conceptual or ethical problems here. Cut and paste has been in the armoury of book producers for a long time. If it is appropriate, I shall continue to use it. It had already become evident that cut and paste was going to be used for the diagrams in Angela Oakley and Tony Norris' paper, since the PostScript files they generated were long gone. This is another instance where speedier production would have been appropriate.

Cut and paste was used only in the two papers mentioned above. All the other "graphical" elements, like those in Michael Ramek's paper, Anne Brüggeman-Klein and Derick Wood's paper, Alois Heinz' paper and Jörg Winckler's were handled by a mixture of TEX and *Textures*' `\specials`. Anne and Derick's paper was provided in LATEX, and made heavy use of the picture environment. Fortunately they also provided some equivalent macros for plain TEX. These included the picture environment from LATEX, which is sufficiently modular that it can be removed from LATEX and used more or less independently. It was used for a diagram in Jörg's paper, and also, with some modification, for a large diagram in Angela and Tony's paper.

The clouds part. Alois Heinz' paper, as noted above, made use of two `\specials`. Fortunately, he had also used *Textures*, but because I was sending the `dvi` file to a very different `dvi`-to-PostScript convertor, I couldn't expect to handle the pictures correctly, and I had expected to cut and paste them, but at this point I had two pieces of good fortune. First, Blue Sky, the authors of *Textures* made an early release of their PostScript outline Computer Modern fonts available. The second stroke of luck was to be summoned to Cheltenham to use one of Linotype's LN300 machines. I took Alois' paper with me. It proved to be embarassingly simple to generate the paper, `\specials` included. No doubt purists will now be taking their magnifying glasses to prove that there are significant differences between the outline Computer Modern and the bitmapped Computer Modern. I know which I prefer to use. There is something attractive in having an LN300 attached to your Macintosh. Who needs laser printers?

Keying in. One paper was about a system based on TEX. Our original intention had been to use that system to produce bromide, emulating the style of the rest of the proceedings, but permitting the differences to be seen by the comparison that would be manifest. This one had been electronically lost. Fortunately I had a paper copy, so I rekeyed it, although one or two small changes had to be made in the text to reflect the reversion to TEX.

The End is Nigh

By this time, everything had fallen into place, and only the front matter and end matter were still to be done. Previously when I had created indexes for books I had done it myself. I had soon concluded that indexing was not only difficult, but it was also time consuming. I felt that an author (or editor) was far too close to their subject to be able to make the "correct" sorts of judgements about the indexed material. Indexing must be kept for mature reflection, preferably someone else's. I hired someone else to do the job – a professional indexer. Judge for yourself. The preface and table of contents were simple enough to do, although time consuming. The last few jobs still take time. The most difficult decision was whether to have a dedication. I wanted one, but decided it would create too much embarrassment.

All this time, my publisher had been remarkably patient. Ellis Horwood was then bought out by the US publisher, Simon and Schuster. I had a momentary shudder when I heard this. Oh dear, there goes my contract. But no.

At length the bromides, together with what I hoped were clear instructions on the paste-ups, were sent off to Ellis Horwood in Chichester. The proceedings arrived in my hands on April 30th (twenty-one months after the conference!).

A Few Lessons to Ignore

Editing. Editing takes time. Do not underestimate the task. Perhaps insisting that authors adopt a standard style would have simplified things. I'm not sure about this at all. Part of my doubt relates to the fact that a TEX conference, in particular, will likely contain papers which extend any style to its limits, or will require extensions to the style to accommodate features the editor/style writer has not considered.

The sheer physical task of editing can be arduous and time consuming. It can be frustratingly difficult to spot typographic errors on the screen. Anyone who has had anything published has had the experience of picking up the end result and instantly seeing a typo. Professional copy editors have a rare skill, which we should not underestimate. Indeed, if publishers have one great benefit to bestow on us, it is in copy-editing.

I think the weakest part of the proceedings are the bibliographies. I thought of having a single bibliography, but this is awkward unless you are able to use a tool like BIBTEX. Since I was not using LATEX, that was not possible. I am conscious that the bibliographies are not as consistent as they might be. A version of BIBTEX which worked independently of LATEX, together with the ability to generate chapter-at-a-time bibliographies would have helped.

Leave well alone. Timeliness might be preferred to production values. Certainly, quite a few authors would have been better pleased to see their offerings in print much sooner. I would too. Equally, a number of others seemed quite pleased that something would be produced, however late it was. But one of the features of having everything available electronically is that there is always a tendency to keep refining beyond the point at which the refinements are noticeable. A commercial venture does not have this problem: it goes bankrupt.

More cooks. It is very difficult to surrender control to someone else. There are only one or two others I could have shared this with. The few times I had to go outside my own resources proved to be counter productive. While this rather reinforces the notion that you have to do it all yourself, I don't think that is what I am implying. Where there are properly definable tasks, they can be assigned. Like the phototypesetting, the indexing, and the stuff the publisher did. I think the copy-editing should be separated out, too.

It's a lonely job. Nobody loves the editor. The authors generally can't understand why it's taking so long; wives and/or lovers begin to wonder who you are; your publisher takes to phoning you to ask when the bromides will be ready, or starts writing "bank-manager" looking letters. It's not for the squeamish or the sensitive. And don't expect thanks.

Conclusion

In common with so much of TEX, the conference and the proceedings were an amateur affair. Recall that *amateur* need not be a perjorative term, any more than *professional* implies excellence. This was a labour of love. I hope I never lose my amateur status.

Would I do it again? Of course. Some people never learn! But next time I'll get it right.

Bibliography

Clark, Malcolm. ed. *TEX88 Conference Proceedings.* Chichester, England: Ellis Horwood, 1989. [18–20 July 1988, Exeter University, Exeter, England.]

Clark, Malcolm. ed. "TEX: Applications, Uses, Methods." *Proceedings, Third European TEX Conference.* Chichester, England: Ellis Horwood, 1990. pg 250.

Clark, Malcolm. *pc-Portable Fortran.* Chichester, England: Ellis Horwood, 1986. pg 228.

Clark, Malcolm. *Coastal Research: UK Perspectives.* Geo Books. Norwich, UK. 1984. pg 131.

Angus, S, B. Armstrong and K. M. de Reuck. *Chlorine Tentative Tables.* Pergamon Press. 1985. pg 162.

SGML (, TEX and ...)

C.G. van der Laan
Rekencentrum RUG
Landleven 1, 9700 AV
Groningen, The Netherlands
050-633374/8080

Abstract

An introductory review is given of what SGML (and DTDs), TEX (and formats), and their relation, is all about. Coupling SGML to TEX is considered via interfacing and transformation. Transformations of the tags of the *basic* as well as the *complete* SGML marked up compuscript are dealt with in detail for the examples: a letter, bridge lay-out, some mathematics and a simple table. At the end some guidelines are provided in order to decide when SGML, TEX, or both, might be beneficial, along with a conclusion. It is a 3-in-1 paper: (1) what SGML and TEX is all about, aimed at novices; (2) examples of marked up copy in SGML and (La)TEX and the coupling issues, for those already familiar with SGML and TEX, but like to be informed about transformation issues, when mathematics or tables are part of the compuscript; (3) finally, a literature compilation, for those who consider bibliographies at hand useful.

What is a Document?

The Association of American Publishers (AAP) *Reference Manual on Electronic Manuscript Preparation and Markup* defines a document as:

> A document is an organized collection of smaller pieces of text (such as chapters) and images (such as figures) that are called elements. The elements in a document have a relationship to each other which gives the document a definite organization called document structure.

Lifecycle-phases of documents. Compuscript preparation requires that additional information be interspersed within the text to aid any subsequent processing. That information, called markup, is usually specific to a particular publisher, system or printing device. A universal standard method of marking up electronic compuscripts, however, offers many advantages.

The *complete Lifecycle* of a document can be thought of as:
- preparation,
- distribution,
- reading,
- storing (Paper? Electronically? Optically?),
- other usage, reuse?[1]

[1] Publishers like reuse, authors rework!

Standard Generalized Markup Language (SGML) supports the complete Lifecycle, where *future* usage of the document is not necessarily restricted to printing. SGML must be complemented, however, by generally accepted Document Type Definitions (DTDs). The Association of American Publishers [AAP, 1987 and 1989] and the British National Bibliography [Smith, 1987] have provided some DTDs. In order to serve the primary aim of publishing, the coupling to formatters must be supported too.

TEX supports formatting and electronic document exchange.

What is SGML?

SGML provides a language to describe documents and has made it possible to achieve two goals:
1) establish a standard means of identifying and tagging parts of an electronic compuscript so that computers can differentiate between these parts; and
2) provide some logical ways of representing special characters, symbols and tabular material, using only the ASCII character set usually found on standard keyboards.

SGML is defined in ISO8879 [1986]. Some introductions to SGML are: Barron [1989], gentle SGML in Sperberg-McQueen & Burnard [1990], and the books Bryan [1988] and van Herwijnen [1990]. Examples are provided in among others ISO/TR9573 [1988].

Purpose. The purpose of SGML is to facilitate *information* exchange, and reusability (in other contexts, even yet unknown contexts),

— *Then and There* [2] —

via a description *language*, where information is packed in compuscripts, containing, text, graphics, formulas, tables, etc.

Meta Language. SGML is a *meta* language, which can be used to define an arbitrary number of markup languages in a standardized way. This means, for any class of documents, markup rules can be prescribed by SGML, yielding a *language*, the Document Type Definition, for that class. The parser checks compliance of the marked up copy to the DTD.

Standard.

Formerly: No consensus on markup codes (WordPerfect, Wordstar, MacWrite, ...; Scribe, troff, TEX, LATEX, ...)

Presently: SGML ISO standard

Standard $\overset{\text{def}}{=}$ It can be used to define an arbitrary number of markup languages in a *standardized* way.

Entails:

General applicability,
Extended lifetime,
Improved reusability,
Enhanced exchange possibilities.

Generalized.

Formerly: (Typeset) *marks* for *specific* here-and-now printers, via *direct* markup.

Presently: Marks are *generic*.[3] This is done with procedural markup. Macro calls are inserted as markup tags, where the implementation of the macros (the format or style file) represents the style, accounts for the fonts, etc. The printer hardware is shielded by intermediate languages. Intermediate language copy is printed via drivers. Change of style needs another style file, no modification of tagged copy is necessary. Change of printer hardware needs another driver, no modification of tagged copy nor modification of format file!

[2] The essential issues of portability: portability with respect to time (then) and place (there).

[3] As opposed to specific print/plot/photo typeset hardware.

Generalized $\overset{\text{def}}{=}$ Abstraction from the specific (printing) to the general (other usage), by emphasizing the *structure* of a document and to specify intent without regard for appearance.

Markup.

Formerly: (Typeset) *marks* in the margin (Marks are bound to a version; no "data-integrity")

Presently: Marks are integrated along with copy (Data-integrity of markup code is preserved.)

Markup $\overset{\text{def}}{=}$ Term used to describe codes added to the electronically prepared document.

Author's point of view. Authors have to markup their copy with

1) Awareness of the DTD which applies to the document; either the DTD must be understood or templates must be available;
2) Knowledge of which (begin) tags to use where and how;
3) Knowledge of ranking, attribute use, tag minimization;
4) Knowledge about how to create entity references.

These aspects are treated in author's guidelines. The above can be alleviated by providing an SGML *environment*, or better, a document preparation environment, supported by menus and templates with prompts. Thus authors can concentrate on structure and content by using a standard (generic) markup language, or a sufficiently advanced document workbench, leaving formatting issues to publishers, or software vendors. Because of this "separation of concerns" the author's task is simplified.

Publisher's point of view. Publishers make use of sufficiently accepted DTDs. They provide authors with guidelines and proofing tools. DTD writing requires knowledge of the various types of markup, such as presentational, direct, procedural and descriptive markup.

Example markups.

No markup. In order to remind the unpleasant look of documents with just words, we start with a no markup example.

```
TeX A system for formatting text
TeX and the accompanying macro
package LaTeX provide powerful means ...
```

Presentational markup. Documents with tabs, indentations, and in general positional control make use of what is called presentational[4] markup, in order to convey the meaning.

[4] Some editors prefer submission of this form for simple text!

```
TeX:
A system for formatting text.

  TeX and its accompanying macro
package LaTeX provide
powerful means of formatting
text to be output on either
  - a simple matrix printer,
  - a laser printer or
  - a photo typesetter.
```

Presentational markup is functional with poetry, such as Alice's mousetail as mentioned by Malcolm Clark [1989] or DEK's favourite poem of Piet Hein [*The Errors of TEX*, 1989], where the words are arranged along an ellipse.

Direct markup. When specific print instructions are included, we get direct markup:

```
@T:                            []
A system for formatting text.[]
                               [I]
@T and its accompanying macro
package @LT provide
powerful means of formatting
text to be output
on either                      [I]
 - a simple matrix printer,    [I]
 - a laser printer or          [I]
 - a photo typesetter.
```

[I] is a print instruction indicating to go to the next line and *indent*; @<name> stands for a process with a special formatting effect.

Procedural (LATEX) markup. A markup command, where the implementation of the command contains print instructions, is considered a procedural markup command; when the printer is changed the implementation has to be changed too, not the marked up copy. LATEX markup of the example reads

```
\subsection*{\TeX}
A system for formatting text.
\par
\TeX\ and its accompanying macro
package \LaTeX\ provide
powerful means of formatting text
to be output on either
\begin{itemize}
\item simple matrix printer,
\item a laser printer or
\item a photo typesetter.
\end{itemize}
```

Descriptive (SGML) markup. Descriptive markup goes even further and uses markup which describes the structure and intent of the various parts of the document:

```
<h>&TeX;
<p>A system for formatting text.
<p>&TeX; and its accompanying macro
package &LaTeX; provide
powerful means of formatting
text to be output on either
<li>
<it>simple matrix printer,
<it>a laser printer or
<it>a phototypesetter.
</li>
```

Formatting information with SGML. It is possible to convey formatting information via SGML. This is done with element attributes or with Processing Instructions. Other symbols than those in the ASCII character set are often denoted by an entity reference to the font containing those symbols, with appropriate loading of the font elsewhere. With respect to attributes, one can think of specifying open space in order to include illustrations from other (electronic) sources. Also, indication of a representation choice is possible if properly accounted for in the DTD. Consider for example the representation of labels of list items: alphabetical or roman/arabic numeral.

```
<li number=alpha>
<it>a simple printer,
<it>a laser printer or
<it>a photo typesetter.
</li>
```

It is possible in SGML to include document parts which already contain formatting information. The parser must be told to lay back. For a notation to be allowed it must be declared via e.g.

```
<!NOTATION TeX SYSTEM>
```

for TEX formatted copy. Appropriate entity and attribute specifications are also needed in the DTD. For an author the equation formatting with the type attribute (value=TEX) has to be supplied as

```
<eqn type=tex>
  $$X\cap(A\cup B)=(X\cup A)\cap(X\cup B)$$
</eqn>
```

Processing Instructions (PIs) can be used to tell the local system how it should process data contained within a document. For example, SETM typography markup instructions:

```
<p><?[s24][sec][rm]>T<?[pri][rm]his>
         paragraph ...
```

In this case the SETM instructions are in brackets, preceded by the PI open delimeter <?. The meaning of this instruction is to treat "**T**" as "**24**pt" initial letter to be set using the **rom**an version of the face currently defined in the **sec**ondary type family.

Availability. As mentioned by Herwijnen [1990], Sobemap, The Publisher and IBM's SGMLDCF, are some SGML systems that are already available.

Support. Support for SGML is done by the companies, as part of automation projects. There also exists a Dutch chapter of the SGML Users Group.[5]

Courses. Courses are provided by private companies, and the National Normalization Institutes.

What SGML is not! SGML is not:

- WYSIWYG (pronounced wĭsēwĭg, and stands for what you see is what you get),
- A formatter, certainly not a standard formatter.

What is TEX?

TEX stands for $\tau\varepsilon\chi$, the first three letters of the Greek word for *technique*, which also means art. TEX is a *machine independent* formatting language designed by Don Knuth, [*The TEXbook*, 1990 (Version 3.0)]. Michael Doob gives an easy start to TEX in *A Gentle Introduction to TEX* [1989]. A systematic treatment of the commands is given by Abrahams [1990]. There also is an introduction in French by Seroul [1989] and various German introductions: Appelt [1988], Schwarz [1989]. Von Bechtolsheim [1990, in English] is impressive. LATEX, by Leslie Lamport [1985], is a macro collection for simplified use of TEX. LATEX uses *procedural* markup. Buerger [1990] gives a LATEX introduction. In German there are Kopka [1989] and Wonneberger [1987]. Bruin [1989] gives a Dutch introduction to LATEX.

Purpose. The stated purpose of TEX is "making beautiful books."

Processing (La)TEX. LATEX is processed in three steps: edit the copy, format the copy to create a `dvi` file, and print the resultant `dvi` file. A diagram of this looks like:

copy $\xrightarrow{editor}$ `ASCII` $\xrightarrow{(La)TeX}$ `dvi`-file $\xrightarrow{driver}$ results.

The more steps to the process, the more cumbersome is correction handling because of larger production *loops*. This is the case when the use of TEX is combined with SGML markup. The SGML parsing and linking extends the loop.

[5] SGML-Holland secretary: D. van Wijnen, Wolters Kluwer. P.O. Box 989, 3300AZ Dordrecht. 078-334933; e-mail: surf003@kub.nl.
SGML User's Group secretary: S. G. Downie, Softquad Inc, 720 Spadina Avenue, Toronto, Ontario M5S 2T9, Canada.

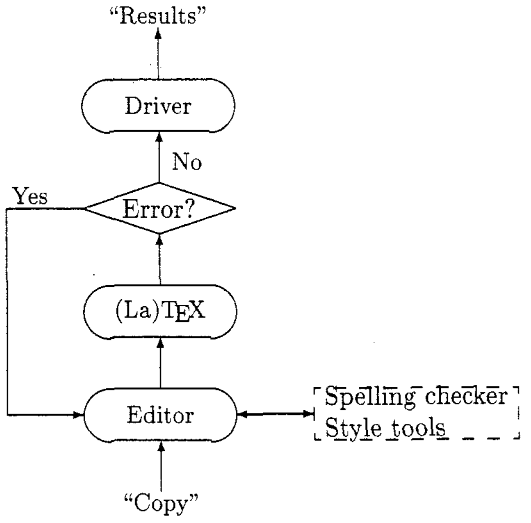


Figure 1: Correction cyclus

Availability. TEX is available on many computers under various operating systems with a variety of drivers for previewing (such as VDU), printing, and photo typesetting. Documents written in TEX or LATEX can be ported easily. Exchanging documents via e-mail is also generally possible except for the incorporated graphics. When graphics are part of the document, TEX can be combined with (encapsulated) Postscript, which is used within the TEX community and elsewhere. Portability is restricted to places with Postscript printers of course. TEX is in the *public domain*. Drivers are generally not. They do, however, generally have value added by the companies you buy the driver from. See the ads in any TUGboat. Moreover, TEX systems can make

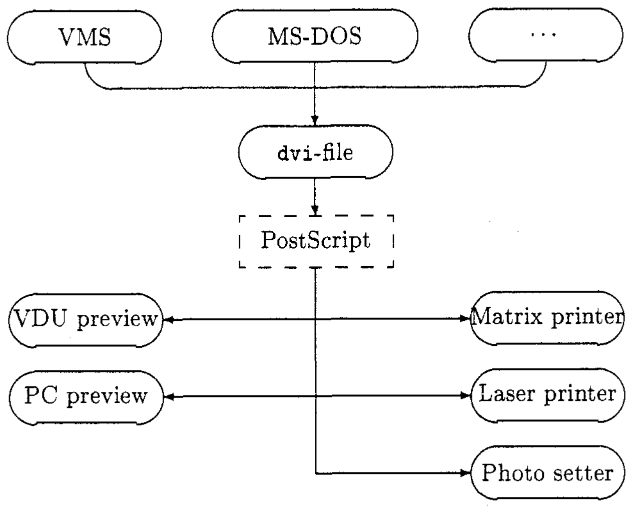


Figure 2: (La)TEX's use

use of fonts from various sources, such as Adobe's PostScript fonts and, of course, Metafont.

Support. Support is organized by the various users groups. Software, style files, macros etc. are distributed (via e-mail, ftp or floppy disks) by the TEX Users Group (TUG)[6]; in the Netherlands it is distributed by NTG[7]; in France by GUTenberg; in Germany by DANTE; in the United Kingdom by ukTEXug; in the Nordic countries by "Nordic TuG". There is also the recent TUGlib (analogous to NETlib from the numerical mathematicians) service in Utah: e-mail fileserver, literature surveys (a.o. what has appeared in TUGboat), and the who-is database.

Courses. Courses are organized by TUG and other TEX user groups, especially in conjunction with their main meetings.

Relationship: DTDs, SGML, TEX, formats and ...

The relationship of TEX, SGML and other applications is illustrated in the diagram below. An integrated[8] implementation is Arbortext's "The Publisher," which has AAP's DTDs built-in, and runs on a.o. SUN hardware. Note that the backarrows denote some of the work in progress by Elsevier Science Publisher, Bleeker [1989], Poppelier [1990].

SGML ór TEX sufficient? NO, needed are format files and DTDs as well! If a compuscript is printed with TEX for personal use only, there is nothing to worry about. When no reuse of a document is in sight, but remote publishing and electronic exchange are considered, it pays to use standard format/style files — which reflect the lay-out of the document type — along with TEX. When reuse or abstract structuring are being used, standard DTDs will be crucial.[9]

Moreover friendly user-interfaces are needed. Grootenhuis (priv. comm.) provided on top of the SGML system he used, the possibility to input letter copy in a natural way with tags hidden: awareness of the DTD used is not necessary, nor does the user see or have to type in any tags. (Of course a sample of how the letter looks is needed.) This kind of system needs a high degree of foolproof-ness: the spaces etc. have *side effects* and when errors are made the (SGML) messages are quite confusing. Another possibility is to prompt the user for the required copy, while the tags are provided by the system.

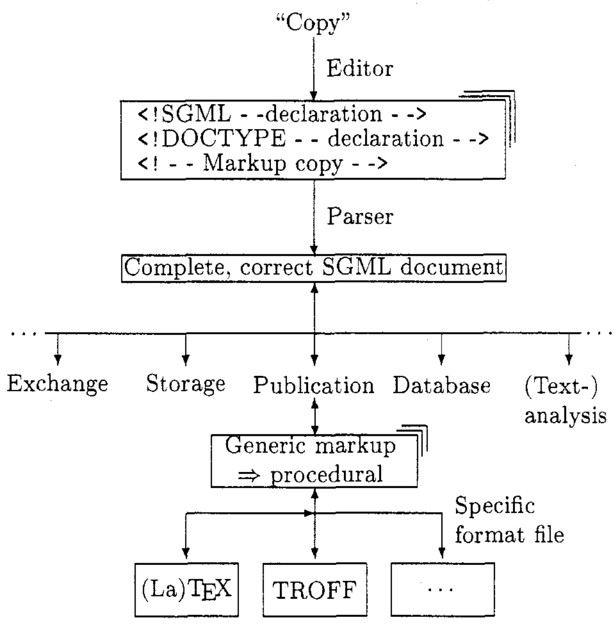


Figure 3: Relation SGML and (La)TEX

Interfacing vs. transformation. Interfacing copy marked up by any formatter to SGML is possible in SGML via the NOTATION mechanism. Of course, it has to be incorporated into the DTD and appropriately implemented: the parser should lay back and leave the formatting of the TEX marked up copy to TEX.
Transformation SGML into TEX is different. Using TEX as a back-end formatter to SGML can be done. It is supported by the link mechanism of SGML. Needed is at least a table of corresponding notations in order to substitute the markup tags from SGML into TEX. The other way round has to be done by a separate program. In the sequel we will study example document elements marked up by SGML and (La)TEX; transformation issues will be addressed as well.

[6] Editorial and TUG address: TEX Users Group, P.O. Box 9506, Providence RI 02940, USA. email: TUGboat@Math.AMS.com.

[7] NTG: Nederlandstalige TEX Gebruikersgroep. Secretary: G.J.H. van Nes, Postbus 394, 1740AJ, Schagen, 02246 - 4185; e-mail: vannes@ecn.nl, ntg@hearn.

[8] Ikons user interface, SGML layer, TEX layer, Postscript handling (optionally); with SGML, TEX and dvi files as intermediate results

[9] The flexibility and open-endedness of SGML are strong points, everybody can write or modify DTDs; this is also a weak point because of incongruent DTDs.

Examples

Simple text. As an example let us take the simple text given earlier. The (basic) SGML and LATEX markups have been given in previous sections. Coupling comes down to a change of representation, except for the omitted endtags. This direct approach needs the substitutions:

SGML	⇒ TEX
`<h>`	`\section{`
`<p>(first)`	`}`
`<p>`	`\par or blank line`
`<li>`	`\begin{itemize}`
`</li>`	`\end{itemize}`
`<it>`	`\item`
`&TeX;`	`\TeX␣`
`&LaTeX;`	`\LaTeX␣`

This suggests systematic coupling of all entities and tags to equivalents in LATEX. The explicit endtags are more natural to handle than the omitted ones. The handling of the first and following occurences of `<p>` have to account also for respectively finishing the heading `</h>` and ending a paragraph `</p>`, which have been omitted.

Letter. A typical letter has the general aspects:

- Background
  Heading (Logo, address, phone, ...)
  Footer (numbering, ...)
- Context (running heads next pages, ...)
- Reference
- Your reference
- Date
- Addressee (name, company, address, zip code)
- Beginning (Dear...)
- Contents
- End matter (Salutation, name, position)
- Additions (PS, enclosure, cc)

The above aspects are generally ordered in a tree for detailing the hierarchical structure and writing the DTD from.[10] To overcome the optional and repetition notational difficulties in the structure tree for document parts, signs are added to the knots: + for repetition, and * for an optional element. (This mechanism accounts also for 'elements in arbitrary

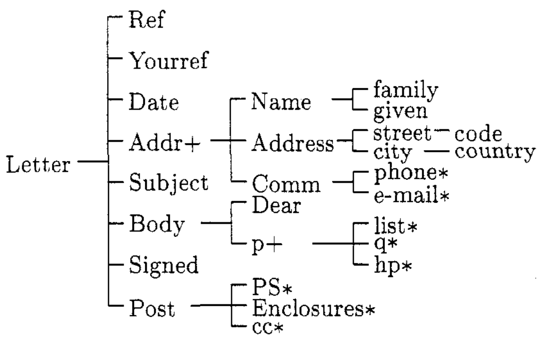


Figure 4: Hierarchical letter structure

order.') Attributes are not reflected in such a tree, neither are included and excluded elements. It is not clear to me why the tree structure is chosen instead of the syntax flowchart notation as used, e.g., along with the definition of the Pascal programming language. From such a tree a DTD is generally written. Only the bridge example DTD is used in this article, because the rigorous DTDs, how interesting and instructive they might be, would lead us too far away into SGML details.

SGML markup. The SGML markup for a typical letter might look something like:

```
<!DOCTYPE letter PUBLIC
 -- DTD to be used  --
  "-//NTG//DTD Letter//EN">
<letter -- start-tag -->
<ref> CGL/Ba/B89-007
<yourref> MC/L1/L89-001
<date> 4 august 1989
<address> Malcolm Clark, ICRF
<email>
     malcolm@icrf.ac.uk
</email>
<dear>Malcolm
<p> Thank you very much ...
...
<p> Some details about the course ...
...
<signed name=CGL>
</letter -- end-tag -->
```

LATEX specification. The same letter using LATEX markup might look like:

```
\documentstyle[12pt]{letter}
  \address{% return address
          C. G. van der Laan   \\
          \ldots}
  \signature{Kees}
\begin{document}
```

[10] A difficulty is what to exclude from the structure and to consider as a formatting issue. In the letter example I have considered the header (with logo) and the headerlines on the pages 2 etc., and the footers as formatting issues. They could have been included in the DTD as included elements; this cannot be represented in the structured tree.

```
\begin{letter}{% address
         Malcolm Clark       \\
         \dots}
% no ref or your ref
% date is handled automatically
\opening{Dear Malcolm}
\par
Thank you very much \ldots
\begin{quote}
$\vdots$
\end{quote}
Some details about the course \ldots
\begin{quote}
$\vdots$
\end{quote}
\closing{Best regards}% Handles signature
%ps, cc, enclosure all possible
\end{letter}
\end{document}
```

Letter result. Because a sample LaTeX letter could not be processed simultaneously in this paper (I don't have access to Postscript facilities in order to include the separately produced result), the printed letter has been omitted.

Transformation. What comes to mind when looking at both representations of marked up copy is the difference in the sequence order of tagged items in SGML and LaTeX. With *complete* marked up SGML copy to be transformed into procedural TeX, there is no problem: in TeX macros, strings can be stored for later usage. This will be shown along with the tabular example in the section 'transformation revisited.'

Bridge card deal. In TUGboat 11#2 I have described typesetting bridge using TeX.

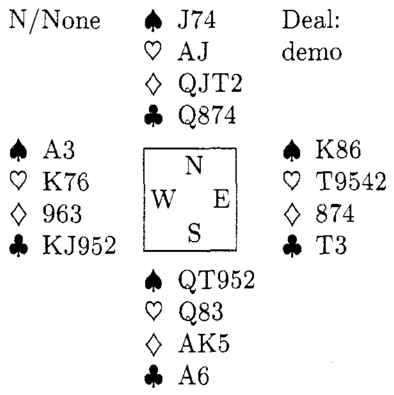


Figure 5: Bridge deal

SGML markup. The SGML markup for Figure 5 above might look like:

```
<deal><vuln>N/None
      <comm>Deal: demo
<hand n><spades>J74
        <hearts>AJ
        <diams> QJT2
        <clubs> Q874
<hand e><spades>K86
        <hearts>T9542
        <diams> 874
        <clubs> T3
<hand s><spades>QT952
        <hearts>Q83
        <diams> AK5
        <clubs> A6
<hand w><spades>A3
        <hearts>K76
        <diams> 963
        <clubs> KJ952
</deal>
```

(La)TeX specification. The procedural TeX markup for Figure 5 might look like[11] :

```
\crdima{N/None}{\vtop{\hbox{Deal:}
                      \hbox{demo}}}%
 {\hand{J74}{AJ}{QJT2}{Q874}}%N
 {\hand{K86}{T9542}{874}{T3}}%E
 {\hand{QT952}{Q83}{AK5}{A6}}%S
 {\hand{A3}{K76}{963}{KJ952}}%W
```

Transformation. The transformation comes down to

SGML	⇒ TeX
`<deal>`	`\crdima`
`<vuln>N/None`	`{N/None}`
`<comm>DEAL: demo`	a suitable `\vtop`
`<hand x>`	`{\hand`

and all the cards per colour surrounded by curly braces, with an extra "}" after the clubs. Although once again a simple[12] example, the translation table is not natural. Note. This is a bottom-up example: descriptive markup TeX macros were already available. Starting from descriptive TeX marked up copy is simple, and a candidate for table-driven substitution, apart from some exceptions like the suitable \vbox here.

TeX macros. Figure 5 is created by using the TeX macros:

```
\def\hand#1#2#3#4{%
%Example: \hand{AKJ765}{AK9}{--}{T983}
\vtop{\hbox{\strut\s\enspace#1}
```

[11] \minipage is not used because of the more general \vtop command, which can be used as well in LaTeX as in TeX.

[12] If one ever can call nested tables simple.

```
\hbox{\strut\h\enspace#2}
\hbox{\strut\d\enspace#3}
\hbox{\strut\c\enspace#4}}%end \vtop
}%end \hand
%
\def\crdima#1#2#3#4#5#6{%
%purpose: layout bridge hand
%#1 left upper·text
%#2 right upper text
%#3, #4, #5, #6: N, E, S, W hands
\vbox{\halign{                &##\quad\cr
             #1&         #3&      #2\cr
 $\vcenter{#6}$&$\vcenter{\copy\NESW}$&
                      $\vcenter{#4}$\cr
               &         #5&        \cr
              }%end \halign
      }%end \vbox
}%end \crdima
%
\def\NESWfig{%
\vbox{\font\small=cmr9
\def\str{\vrule height2.2ex%
    depth.75ex width 0pt}
\offinterlineskip\tabskip0pt\hrule
\halign{\vrule\hskip2pt\relax
##\hfil\tabskip3pt&  \str\hfil##\hfil&
##\hskip2pt\relax\hfil\vrule
                        \tabskip0pt\cr
  &      \hbox to 0pt{\hss\N\hss}&  \cr
\W&                      \phantom{N}&\E\cr
  &  \str\hbox to 0pt{\hss\S\hss}&  \cr
       }%end \halign
\hrule}%end \vbox
}% end \NESWfig
\setbox\NESW\hbox{\NESWfig}
```

SGML requirements. The following DTD is needed:

```
<!ENTITY % ISOpub PUBLIC
 "ISO 8879-1986//ENTITIES Publishing//EN">
<!ELEMENT deal - - (vuln, comm?, hand+)>
<!ELEMENT (vuln|comm) - O CDATA>
<!ELEMENT hand - O (spades,
                    hearts, diams, clubs)>
<!ATTLIST hand nesw (n|e|s|w) #REQUIRED>
<!ELEMENT (spades|hearts|diams|clubs)
                    - O CDATA>
```

Note. In the DTD we could have imposed sequence ordering by changing hand+ into (handn, hande, hands, handw). But this requires that all the hands are needed and that is too restrictive.

Some math. The following examples of mathematical formulas are borrowed from the "Mathematical Formulas" report [van der Laan, Coleman, Luyten, 1989]. In this report, SGML and LATEX markup are supplied for formulas from various fields: elementary mathematics, set theory, geometry, functional analysis, calculus (differential equations, special functions, continued fraction), statistics, algebra (tensor calculus), homology (diagrams) and quantum mechanics. A few simple ones are selected here.

LATEX results. The following was formatted with LATEX markup:

$$X \cap (A \cup B) = (X \cup A) \cap (X \cup B)$$

$$x \notin A \not\subset B$$

$$\|a(x+y)\| \leq |a|.(\|x\| + \|y$$

$$\int \frac{1}{\sqrt{1+x^2}}\, dx = \log(1+\sqrt{1+x^2})$$

(Basic) SGML markup. To accomplish this with SGML markup, you might enter:

```
<fd>X&cap;(A&cup;B) =
     (X&cup;A)&cap;(X&cup;B)</fd>

<fd>x&nisin;A&nsub;B</fd>

<fd><fen d>a(x+y)<rp d</fen>&le;
  |a|.(<fen d>x<rp d</fen>
    +<fen d>y<rp d</fen>)
</fd>

<fd><in><opd><fr>1</><rad>1+
  x<sup/2/</rad></fr>dx</in>=
  <rf/log/(1+<rad>1+x<sup/2/</rad>)
</fd>
```

The DTD used is an adapted version of AAP's DTD by D.C. Coleman.

(Direct) TEX markup. LATEX and TEX markup are very similar for these examples:

```
X\cap(A\cup B) =(X\cup A)\cap(X\cup B)

x \notin A \not\subset B

\|a(x+y)\| \leq |a|.(\|x\|+\|y\|)

\int\bfr1</>\sqrt{1+x^2}\efr dx =
                 \log(1+\sqrt{1+x^2})
```

Transformation. To accomplish the SGML to TEX transformation, some general substitutions are needed:

SGML	⇒ TEX
`<fd>`	`\[ or $$`
`</fd>`	`\] or $$`
`<sup/2/`	`^2`
`etc.`	

For the first set theory example the following substitutions are additionally needed

SGML	⇒ TEX
`&cap;`	`\cap␣`
`&cup;`	`\cup␣`

Functional analysis required moreover

SGML	⇒ TEX
`<fen d>`	`|`
`<rp d>`	`|`
`&le;`	`\leq␣`

For the integral the following definition (format) is needed for the fraction, where use is made of `</>` as parameter separator:

```
\def\bfr#1</>#2\efr{{#1\over#2}}
```

Also needed are the substitutions

SGML	⇒ TEX
`<in>`	`\int␣`
`<opd>`	`\relax`
`<fr>`	`\bfr`
`</fr>`	`\efr`
`<rad>`	`\sqrt{`
`</rad>`	`}`
`<rf/log/`	`\log␣`

Note. A translation table is once again not straightforward; unnatural are the SGML difference in norm open and closing, and the fancy use of `</>`, i.e. null endtag for numerator and omitted opening tag for denominator. The short reference for the modulus sign is neat.

(Complete) SGML markup. The sobemap parser yielded the following (visually edited) result for the set theory example:

```
<FD DCN="GEO.FORM">
<FL>
X&cap;<FEN STYLE="S" LP="PAR">
        A&cup;B
      <RP STYLE="S" POST="PAR"></FEN>
=
<FEN STYLE="S" LP="PAR">
   X&cup;A
<RP STYLE="S" POST="PAR"></FEN>
   &cap;
<FEN STYLE="S" LP="PAR">
   X&cup;B
<RP STYLE="S" POST="PAR"></FEN>
</FL>
</FD>
```

This shows that complete SGML is verbose. For example, consider the complete tags for parentheses "(" and ")". Thanks to the short reference mechanism the input can look natural. Coupling of the above to TEX can be done. How to automatically handle attributes in the best way is not yet clear to me. It is not efficient for parentheses, "(" and ")", to be expanded first by the parser into a fence tag with the appropriate attribute value, followed by substitution at the TEX level into "(" and ")" again.

Because matrices are treated similarly to tabular material, we have omitted a matrix example and refer to the next sections, where a table is studied.

AAP's simple table. A simple table is characterized by: simple table entries, one header row, no header subrows, no footer, no intra referencing, and no caption. From the SGML technical point of view no attributes are used. AAP's example simple table [Markup of Tabular Material, 1989], even more simplified, is reproduced below.

Table AAP Job Changes: 1973–1980

	Gain/Loss of Hospitals since 1973	Total No. of CEO Job Changes 1973–80	Survival Rate of CEO's
Texas	+20	—	22%
Maryland	+ 5	42	24%

Source: David Kinzer, "Turnover Of Hospital Chief Executive Officers: A Hospital Association Perspective," *Hospital and health Services Administration* May–June 1982.

Figure 6: AAP's simplified table

(Basic) SGML markup. The SGML markup for Figure 6, with a minor structural adaptation and some layout modifications, reads:

```
<tbl  -- start of table               -->
<no>Table  AAP
<tt>Job Changes: 1973&ndash;1980
<th    -- heading                     -->
<th>Gain/Loss of Hospitals since 1973
<th>Total No. of CEO Job Changes
    1973&ndash;80
<th>Survival Rate of CEO's
<bdy  -- table body                   -->
@Texas|20||22%
@Maryland|5|42|24%
<ft    -- table foot                  -->
<au>David Kinzer
<atl>Turnover Of Hospital Chief
     Executive Officers: A Hospital
     Association Perspective
<nme>Hospital and health Services
     Administration
<dt>May&ndash;June 1982
</tbl>
```

A DTD for simple tables is not separately provided by AAP; it is incorporated as part of the complex table DTD. The "simple table"-example obeys the following SGML structure description

```
<!ENTITY row    STARTTAG "row"          >
<!ENTITY column STARTTAG "c"            >
<!ELEMENT tbl       - - (hd, bdy, ft)   >
<!ELEMENT hd        - 0 (no?, tt?, th+) >
<!ELEMENT bdy       - 0 row+            >
<!ELEMENT row       - 0 c+              >
<!ELEMENT ft
        - 0 (au|src|atl|nme|dt)         >
<!ELEMENT (th, c, au, src, atl, nme, dt)
        - 0 (%t.cs;) -- Character string-->
<!SHORTREF tablemap "@" row
                    "|" column          >
```

Note. Some tags are presumed part of the general DTD, e.g. no, tt.

(Direct) TEX markup. As expected, some formatting commands had to be included in order to reproduce the published results.

The approach has been to supply a template (preamble) line for the layout of the contents proper and to handle the header row as an exception to the template, *manually!* Separators between header, body and footer have to be incorporated as well. A translation table is clearly insufficient. The TEX markup, then, would look like:

```
\newdimen\entrywidth%\entrywidth=<default>
\newdimen\tablewidth
\tablewidth=.5\hsize%default
\def\nl{\par\noindent}
\def\ndash{--}
\def\tablerule{\noalign{\hrule}}
\newdimen\digitwidth\setbox0=\hbox{\rm0}
         \digitwidth=\wd0
%?-command for nonsignificant leading
%         zeroes, Knuth p241
\catcode`?=\active
\def?{\kern\digitwidth}
%\btbl %AAP's simple example with direct
%       TeX markup
\entrywidth=2cm
\tablewidth=4\entrywidth
\vbox{\hsize=\tablewidth
\halign{\hbox to\entrywidth{#\hss}\hfil&&
    \hbox to .5\entrywidth{\hss#}\hfil\cr
%preamble line
   \tablerule\noalign{\vskip1ex}
   \omit{\bf  Table AAP}
        Job Changes: 1973--1980
   \hidewidth\cr
   \tablerule\noalign{\vskip1ex}
   \omit           &
   \omit\vtop{
        \noindent\hsize=\entrywidth
        Gain/Loss\nl
        of Hospitals \nl
        since 1973}&
   \omit\vtop{
        \noindent\hsize=\entrywidth
        Total No.    \nl
        of CEO       \nl
        Job Changes  \nl
        1973--80}  &
   \omit\vtop{
        \noindent\hsize=\entrywidth
        Survival     \nl
        Rate of      \nl
        CEO's}     \cr%end header row
\noalign{\vskip.5ex\hrule\vskip.5ex}
%head-body separation
   Texas    &$+$20&---& 22\%\cr
   Maryland&$+$?5&42 & 24\%\cr
\noalign{\vskip1ex}%body-foot separation
   \noalign{Source: David Kinzer,
   ``Turnover Of Hospital Chief Executive
     Officers:
     A Hospital Association Perspective,''
     {\it Hospital and health Services
     Administration\/} May--June 1982.
    }%end \noalign
}%end \halign
}%end \vbox
```

Transformation revisited.

Complete SGML markup with procedural TEX markup representation can be achieved via the SGML link mechanism, or by any automatic substitution process (programmable editor, preprocessor). Translation into TEX can be done *direct* or via procedural markup. For the mathematical examples given in van der Laan, Coleman [1989], this has been done by Grootenhuis [1990]. He has used the SGML link mechanism for "substitution" and did direct coupling to LATEX. The *direct* coupling approach, without procedural (LATEX) markup, has the disadvantage that change of formatter requires (TEX) source adaptation. In order to abstract from any particular format, we have considered procedural TEX markup as an intermediate phase, in van der Laan, Coleman [1990].

SGML⇒TEX using procedural markup. The "generalized markup ⇒ format" process can be characterized by the following four levels,

$$\begin{tikzcd}
\text{(basic) Generalized markup (SGML)} \arrow[d] \\
\text{Complete generalized markup (SGML)} \arrow[d, Rightarrow] \\
\text{Procedural (\TeX) markup} \arrow[d] \\
\text{Formatted document}
\end{tikzcd}$$

with the input phase and output (print) phase before and after. (The dots separate the SGML bound layer from the TEX bound layer.) The interfacing with procedural markup is illustrated below, with AAP's simplified table as example. This four level process resembles the coupling of higher level programming languages such as PASCAL and ADA to FORTRAN (numerical libraries). For more on the latter, I refer to Einarsson&Gentleman [1984] and other early work of mine [1984].

Note. Of course, one can also work the other way round: start from procedural marked up copy and couple that to SGML.

SGML markup, AAP's table. (Completely tagged) The complete SGML markup version — complete means expansion of short references into tags and addition of omitted (end) tags — of AAP's simple table reads:

```
<tbl>
<no>Table AAP</no>
<tt>Job Changes: 1973&ndash;1980</tt>
<hd>
   <th></th>
   <th>Gain/Loss of Hospitals
       since 1973</th>
   <th>Total No. of CEO Job Changes
       1973&ndash;80</th>
   <th>Survival Rate of CEO's</th>
</hd>
<bdy>
<row><tsb> Texas</tsb>
     <c>20</c><c></c><c>22%</c>
</row>
<row><tsb>Maryland</tsb>
     <c>5</c><c>42</c><c>24%</c>
</row>
</bdy>
<ft>
   <au>David Kinzer</au>
   <atl>Turnover Of Hospital Chief
        Executive Officers: A Hospital
        Association Perspective</atl>
   <nme>Hospital and health Services
        Administration</src>
   <dt>May&ndash;June 1982</dt>
</ft>
</tbl>
```

The above tagged table describes contents and structure. The variety of presentations for printing must be catered for with either a TEX format or a LATEX style file.

(Procedural) TEX markup, AAP's simple table. We have limited ourselves to "translating" SGML markup into procedural TEX markup (no LATEX markup). (Mainly: `<name>` into `\bname` and `</name>` into `\ename`; the transformation of the tags can be table-driven, but in the header row `\nl` commands have to be inserted manually, guided by taste and aesthetics and limited by the value of `\entrywidth`. Note that the data have to be adapted too: insertion of ? for suppressed 0, and \ before %.)

```
\entrywidth=2cm
\tablewidth=4\entrywidth
\btbl %AAP's simple example with TeX
      %procedural markup
\bno Table AAP\eno
\btt Job Changes: 1973--1980 \ett
\bhd
   \bth\eth
   \bth Gain/Loss\nl
        of Hospitals \nl
        since 1973 \eth
   \bth Total No.    \nl
        of CEO       \nl
        Job Changes  \nl
        1973--80 \eth
   \bth Survival     \nl
        Rate of      \nl
        CEO's \eth
\ehd
\btby
\brow\btsb Texas\etsb\bc$+$20\ec
     \bc\ec\bc 22\%\ec
\erow
\brow\btsb Maryland\etsb\bc$+$?5\ec
     \bc 42\ec\bc 24\%\ec
\erow
\etby
\bsrc
   \bau  David Kinzer\eau
   \batl Turnover Of Hospital Chief
   Executive Officers:
   A Hospital Association Perspective\eatl
   \bnme Hospital and health Services
   Administration\enme
   \bdt May--June 1982\edt
```

```
\esrc
\etbl
```

Note. We still have to supply the values for entrywidth and tablewidth along with each particular table, once again, manually.

TEX format macros. In order to reproduce AAP's representation the following (format) macros were written

```
%TeX ``format'' for AAP's simple table.
\newdimen\entrywidth %\entrywidth=<default>
\newdimen\tablewidth
\tablewidth=\hsize%default
\def\nl{\par\noindent}
\def\ndash{--}
%? command for nonsignificant
%  leading zeroes, Knuth p241
\catcode`?=\active
\def?{\kern\digitwidth}
%
\def\tablerule{\noalign{\hrule}}
\def\btbl{\bgroup
   \def\bno##1\eno{{\bf##1}}
   \def\btt##1\ett{{##1}\hidewidth\cr}
   \def\bhd{\tablerule\noalign{\vskip1ex}}
   \def\ehd{\cr
       \noalign{\vskip.5ex}\tablerule
       \noalign{\vskip.5ex}}
   \def\bth##1\eth{\vtop{\noindent
              \hsize=\entrywidth ##1}&}
   \def\btby{\noalign{\vskip1ex}}
   \def\etby{\noalign{\vskip1ex}}
   \def\btsb##1\etsb{\hbox to
             \entrywidth{##1\hss}\hfil}
   \def\bc##1\ec{&\hbox to .5
             \entrywidth{\hss ##1}\hfil}
   \def\brow##1\erow{##1\cr}
   \def\bsrc{\noalign\bgroup}
   \def\esrc{%Source information is
             %handled conform AAP's
             %representation
            Source: \gau, ``\gatl,''
            {\it \gnme\/}
                    \gdt.\ \gobi
            \egroup}
   % Next items are ``stored'' via gdefs
   \def\bau##1\eau{\gdef\gau{##1}}
   \def\batl##1\eatl{\gdef\gatl{##1}}
   \def\bnme##1\enme{\gdef\gnme{##1}}
   \def\bdt##1\edt{\gdef\gdt{##1}}
$$\vbox\bgroup\hsize=\tablewidth
    \halign\bgroup &##\cr%preamble line
    \tablerule\noalign{\vskip1ex}
}%end\btbl
\def\etbl{\egroup%\halign
          \egroup$$%\vbox
          \egroup%\btbl
         }%end \etbl
%end AAP simple table format
```

The above listed format macros take care of the final results in print: appropriate separators and good order and format of the 'source' items. The table entries are centered and aligned on the last digit. This required knowledge of the column width. In order to handle the footer suitably the tablewidth had to be known. The chosen approach allows flexible formatting of the footer. Variability of column widths has not been incorporated in the provided macros but can be addressed.

Difficulties with AAP's *complex* table DTD. Although not the case in the above elaborated example, we experienced difficulties with header rows which contain "halflines." In my opinion, halflines belong to the structure. Confusion arises when the `br` (begin row) and `er` (end row) attributes are used together with halflines. According to the DTD, halflines don't account for a line in the formatted result, in TEX formatting they do, jeopardizing the prescribed br- and er-values.
Note that author etc. information is stored in gdefs in TEX, in order to cope with the difference in sequence order of this items in SGML (AAP's DTD) and TEX (independent) marked up tables.

Graphics. Coupling graphics is not (yet) elaborated, because graphics in SGML is left to other sources. Various graphic sources are interfaced to SGML.[13] The only structuring aspect deals with open space (to electronically paste up the illustration) which must not be split over a page break. The difference with mathematics possibly is that formulas are also part of the text while illustrations are more or less separated from the text.

Developments

A survey of the development of SGML is given by Barron [1990].

Usage. Among the users of SGML there are:

- DOD (Automated Technical Order System)
- European Communities (FORmalised EXchange of Electronic Documents; office official publications)
- Publishers (AAP, British Library, KNUB-(Elsevier, Kluwer, ...), ...)

[13] CGM (Common Graphics Metafile) is adopted by the SGML community, as communicated by Malcolm Clark, Idle by the Thames, TeXline X.)

[21] Guittet, C.(1986): FORMEX: une mise en practique des normes internationales. SGML user's group. Bulletin, 1, 2.

[22] Herwijnen, E. van (1988): Electronic submission of Physics articles to publishers. De 1^e Nederlandse SGML conferentie. SGML: De Consequenties. (Also published in: Computer Physics Communications 57 (1989) 244 - 250: The use of text interchange standards for submitting physics articles to journals.). In the context of this paper the discussion of SGML related to TEX is relevant.)

[23] Herwijnen, E. van (priv. comm.): Streamlining publishing procedures.

[24] Herwijnen, E. van (1988): TexT Processing at CERN I. SGML Users' Group Bulletin, 3, 2, 39–54.

[25] Herwijnen, E. van (priv. comm.): Streamlining publishing procedures.

[26] Herwijnen, E. van (1990): Practical SGML. Kluwer-Academic.

[27] ISO8879 Information Processing — Text and Office Systems — Standard Generalized Markup Language (SGML). 1986–10–15.

[28] ISO/TR9573 Information Processing — SGML Support Facilities — Techniques for using SGML. 1988–09–12.

[29] Knuth, D.E. (1989): The Errors of TEX. Softw. Prac. Exp. 19, 7. 607 - 685.

[30] Kopka, H. (1989): LATEX, Eine Einführung. Addison-Wesley.

[31] Laan, C.G. van der (1984): (Graceful) Mixed Language Programming. Argonne National Laboratory Workshop.

[32] Laan, C.G. van der (1990): Typesetting Bridge via TEX. TUGboat, 11#2, 265 - 276.

[33] Laan, C.G. van der, D.C. Coleman, J.R. Luyten (1989): SGML - (La)TEX. 1. Mathematical Formulas. RC-RUG report 24.

[34] Laan, C.G. van der, D.C. Coleman (to appear): SGML - (La)TEX. 2. Tabular material.

[35] Lamport, L. (1985): LATEX a Document Preparation System. Addison-Wesley.

[36] Poppelier, N.A.F.M.(1990): SGML and TEX in Scientific Publishing. EuroTEX90, Cork.

[37] Seroul, R.(1989): Le petit livre de TEX. Inter-Éditions. Paris.

[38] MARK-IT (1989): SGML Parser, version 2. Sobemap NV, Place du Champ de Mars 5, Bte 40, 1050 Bruxelles.

[39] Schwarz, N.(1989): Einführung in TEX. Addison-Wesley. (Translated into Dutch and English)

[40] Scheller, A. (1989): Experience with SGML in the real world. Advisory Group on Computer Graphics workshop, Rutherford. (Work reported from DAPHNE-project: Document Application Processing in a Heterogeneous Network Environment.)

[41] Smith, J.M.(1987): The standard generalized markup language (SGML): Guidelines for editors and publishers. British National Bibliography Research Fund Report 26. ISBN 0-7123-3111-5.

[42] Smith, J.M.(1987): The standard generalized markup language (SGML): Guidelines for authors. British National Bibliography Research Fund Report 27. ISBN 0-7123-3112-3.

[43] SGML User's Group Newsletters. [15]

[44] Sperberg-McQueen, C.M., L. Bernard (1990): ACH-ACL-ALLC. Guidelines for the encoding and interchange of machine readable texts.

[45] Vignaud, D.(1989): L'édition structrée dans les documents, SGML applications `a l'édition française. Éditions du Cercle de la Librairie. Paris.

[46] Warmer, J., S. van Egmond (1989): The implementation of the Amsterdam SGML Parser. EP-ODD, 2, 2, 65 - 90.

[47] Warmer, J. (priv. comm).

[48] Wittbecker, A.(1989): TEX enslaved. Proceedings TUG89. Stanford. (Advantages and disadvantages of TEX-formatter with SGML "front-end" are discussed, related to DEC's VAX Document.)

[49] Wonneberger, R. (1987): Kompaktführer LATEX. Addison-Wesley.

[15] Editorial address: Pindar Infotek, 2 Grosvenor Road, Wallington, Surrey SM6 0ER, UK.

SGML and TEX in Scientific Publishing

N.A.F.M. Poppelier
Elsevier Science Publishers
Physical Sciences and Engineering Division
R&D Department
email: `n.poppelier@elsevier.nl`

Abstract

Elsevier Science Publishers has for a few years investigated the possibility of accepting compuscripts, a manuscript in electronic form, created with TEX, LATEX and a few other text processing systems, and converting these to *SGML* form. This paper will discuss the current status of these activities, the reasons for converting compuscripts to *SGML* form, and the various ways in which TEX is used.

Introduction

Until a few years ago the results of scientific research were always published in the following way: the author writes an article using whatever method he prefers. Some authors use TEX, some authors a word processor, others prefer a typewriter, and there are still authors who consider the pen to be the best desktop publishing tool that is available.

When the author is satisfied with his work, he or she submits the manuscript, say, to the editor of a journal, usually in duplicate or triplicate. The editor enters the manuscripts into his administration, and sends a copy to one or more "referees," who review and judge the manuscript. This is one of the places on the long road from manuscript to published paper where the publisher adds something to the paper. The system of peer reviewing, which provides a kind of scientific certification, is one example of "added value." The quality of the printed product and the world-wide distribution of an article in a journal can also be labelled "added value."

The referee(s) can either recommend acceptance of the manuscript for publication, possibly with suggestions for alterations, or rejection of the manuscript. If it is accepted, and after the author has put it into final form, the manuscript is sent to the publisher, where it is adapted to the style of the journal by a technical editor. Then the manuscript is re-typed (typeset) and, in most cases, stored in electronic form. Because of the fact that the material has been re-typed, it may contain errors. These errors are removed by proof-reading, which is a task that can be performed, for example, by the technical editor or the author.

Finally, the paper is published, together with other papers, in the journal and after a while the author sees his article in print.

The time that passes between submission of the manuscript and publication of the article in printed form can vary between a few weeks and a year, or even longer. In some fields of science this period of time is too long and authors sometimes ask publishers why this time cannot be reduced.

Modern Times

The conventional method of publishing, which I have sketched above, can probably not be made any more efficient by using conventional means. However, many authors use their own text processing systems to produce their papers and reports. This text can be transmitted in electronic form. In principle, this makes it possible to

- publish the paper within a shorter length of time;
- extract bibliographic information and abstracts, and store this information in a database.

In the first case we see that, by making use of manuscripts in electronic form, which I will call *compuscripts* in the rest of this paper, that is by making use of modern electronic means, the conventional method of publishing scientific journals *can* be made more efficient. Furthermore, the production of secondary publications, such as abstract journals, can be made more efficient since the abstracts do not have to be re-typed.

An important consequence is that the publisher can now add extra value to the journal or to his entire range of scientific publications, by making the material available in various electronic forms, such as hypertext books, databases on CD-ROM, etc.

However, a publisher can only accept a compuscript if the instructions of the text processing system of the author can, in one way or another, be converted to that of the publisher, which in most cases is a computer-driven phototypesetter. Since a publisher wants to work as efficiently as possible, this conversion must be automatic or nearly automatic. This means that the compuscript must be prepared by the author according to a set of rules. In most cases, of course, a technical editor still has to read a draft printed on paper to mark appropriate changes to the text, the typography or the layout. Especially if the paper contains tables or mathematical formulae, this is an intricate task.

Research at ESP

Elsevier Science Publishers (*ESP*) has for a few years been investigating the possibility of accepting compuscripts created with modern text processing systems. The aim of our present research is to develop a method for manipulating the compuscripts in such a way that the material can be published more efficiently than before, and is also stored for re-use at a later stage.

We take the view that the transition from manuscript-based to compuscript-based publishing must at least preserve the flexibility of the current submission procedures and, where possible, improve upon it. This means, for instance, that the author should not be bothered with compuscript preparation instructions that are significantly different between publishers, or even between the various journals of one publisher. It also means that publishers should be able to accept compuscripts produced by a range of text processing systems, and submitted either by electronic network or on a diskette, via ordinary mail.

SGML

As Figure 1 shows, the Standard Generalized Markup Language (SGML) [1, 2, 3] plays a central role in our approach. SGML promises to become a very important tool, since it allows us to (i) separate the logical structure of the document from its visual structure, i.e. its appearance; (ii) treat all accepted compuscripts, once they have been converted to SGML form, in a uniform working environment; (iii) handle the compuscripts independently of output medium and output format; (iv) store the text, wholly or partially, in a database and use the database contents again at a later stage for various other forms of publications; (v) standardize compuscript input formats between the various scientific publishers, thus helping to preserve the flexibility of submission mentioned above.

It is not our intention to ask authors to submit SGML-coded compuscripts yet. However, we foresee that, as a consequence of the fact that the departments of the US Government as well as major industries have adopted SGML as a standard for document interchange, developers of text processing software will provide their customers with facilities for conversion from their particular text format to SGML. Authors will then be able to produce their compuscripts with the text processing system of their choice and to provide the publisher with an SGML version without additional effort.

Until then, in order to have all material submitted in the form of compuscripts available in SGML-coded form, conversion facilities must be developed. The scheme we would like to realize is indicated in Figure 1.

On the one hand, Elsevier Science Publishers wants to allow authors to use their favorite text processing system. On the other hand we know that there are dozens of text processing systems that are being used by authors in various disciplines.

This means that we have to make a choice. Two factors influence this choice: (i) the text processing systems chosen must correspond to a fair proportion of the total number of authors, and (ii) a compuscript prepared with a certain text processing system must, ideally, be convertible to SGML form. The first point is something that should be investigated separately for every discipline. The second point implies that we must choose text processing systems that, whenever possible, produce texts where the logical structure is, at least partially, indicated. In other words: a Word compuscript based on a style sheet is to be preferred over a plain Word file, and LaTeX is preferred over plain TeX.

TeX Compuscripts

TeX will be one of the text processing systems included in our electronic publishing scheme. Later this year, as a first step towards having a TeX compuscript scheme for all appropriate journals, *ESP* will start accepting LaTeX-coded compuscripts for a few of their physics journals. For various reasons we have chosen LaTeX as the most convenient variety of TeX.

- Since according to Lamport [4], "the primary function of almost all the LaTeX commands ... [is] to describe the logical structure of [a] docu-

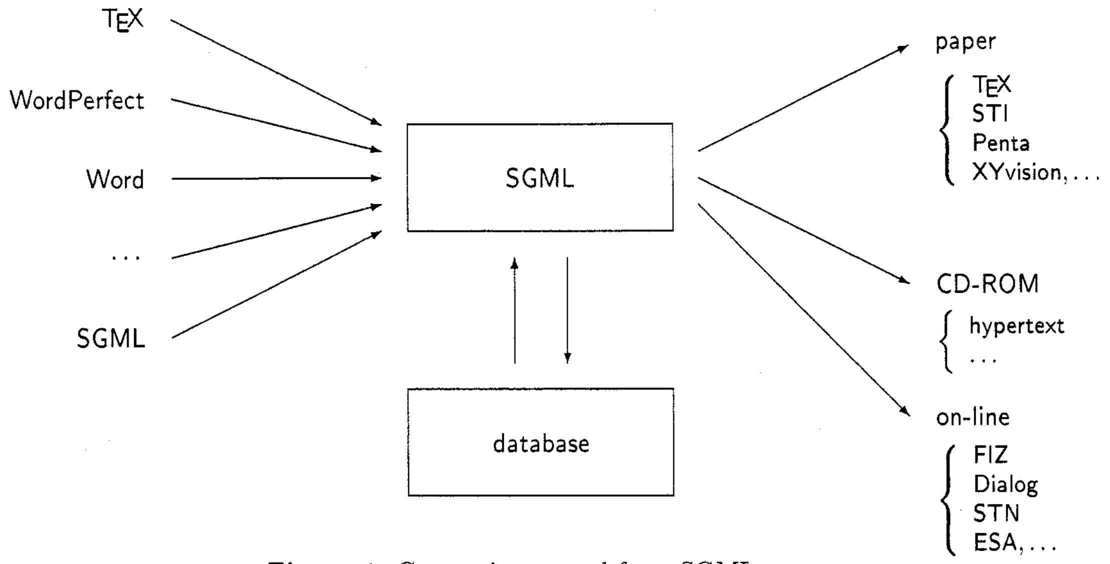


Figure 1: Conversion to and from SGML.

ment" conversion of a LATEX-coded compuscript to SGML form is possible.

- The idea of a document style, which is fundamental to LATEX, enables us to produce camera ready copy for a particular journal by changing only the `\documentstyle` command in the submitted compuscript.
- LATEX is relatively easy to learn.
- LATEX is described in a book, so that it can be considered as a *de facto* standard.
- Especially for large review articles and for books, the following LATEX tools come in very handy
  - cross-referencing with symbolic keys,
  - automatically generated table of contents,
  - index and bibliography tools.

Status Quo

So far we have achieved several things. First of all, we have created LATEX document styles for four of *ESP*'s scientific journals. These new document styles exactly reproduce the layout of the corresponding journals, which so far are still produced in the conventional way.

We have also been able to develop a set of programs for complete automatic conversion of LATEX documents with mathematical formulae to SGML form. To be more precise: we convert LATEX documents with mathematical formulae to a *document type definition* [1, 2] for scientific papers that is based on the one developed by the *Association of American Publishers* (*AAP*).

A document type definition is a description of the logical structure of a certain class of documents, using a method that resembles the specification of the syntax of a programming language. As an example, the document type definition of a novel is represented in a diagrammatic manner in Figure 2.

Conversion of LATEX-coded tabular material to SGML form, i.e, to the *AAP* document type definition or possibly a different document definition, is currently under investigation. Another automatic conversion that is currently under investigation is the conversion from Word, with an appropriate style sheet, to SGML.

Book Projects with TEX

Recently the Physical Sciences and Engineering Division of *ESP* has published several books that have been written in plain TEX or in LATEX, and the number of TEX-coded and LATEX-coded books that we will publish in the near future will probably increase. Our experience is that LATEX is an excellent tool for these larger projects due to the separation of logical structure and visual structure, the ability to do cross-referencing with symbolic keys and the presence of tools for index and bibliography; these advantages were already mentioned under the section *TEX Compuscripts*.

However, we have also noticed that making a TEX-coded or LATEX-coded book ready for publication requires more time when the authors do not have to follow a set of instructions, but are free to type things the way they think is best.

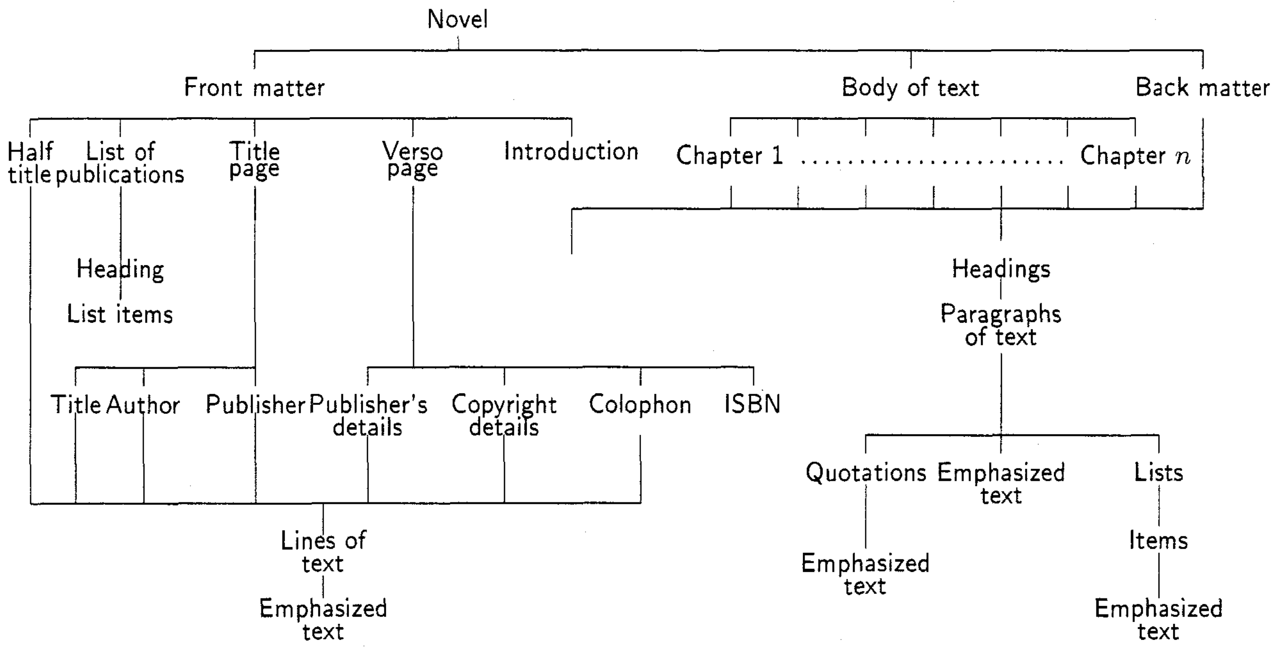


Figure 2: Document type definition of a novel.

Other Uses of TEX

Lots of programs, such as database programs, have output capabilities that are somewhat meager. After all, a database program is meant to store data, not to print it in a sophisticated manner. The fact that TEX is a programmable text formatting system makes it an excellent "print engine" for such programs. As long as the database program is capable of inserting some fairly simple texts in its output – most database programs have report generators that are able to accomplish this – TEX's capabilities as a text formatter can be used to print the output in any desired way.

An example of this approach is symbolic mathematics packages that can compute complicated formulae and generate the TEX codes for formatting them. Another example is provided by the database-publishing activities of the *Excerpta Medica Publishing Group* (*EMPG*), which is part of the Biomedical Division of *ESP*.

The *EMPG* employs a database of article openings, abstracts and citation lists that have been derived from articles in a selected set of biomedical journals. The information in this database is stored in a structured form. With this information the *EMPG* produces abstract journals that are called the *Core Journals in* $\langle \ldots \rangle$, where $\langle \ldots \rangle$ is, for instance, *Opthalmology*, *Cardiology* or *Neurology*. The *EMPG* has been experimenting with using LATEX for the production of *Core Journals*: the relevant portions of the database are extracted and converted to LATEX. The LATEX-coded form of this information can then, in principle, be used to produce the camera ready copy for the journal, using a document style that was developed in-house.

In the near future, starting this year, a similar system will be implemented for the article openings of all journals that are published by *ESP*. The purpose of this project, which is called *CAPCAS*, is to have the article openings, i.e. title, author(s), abstract, keywords and publication history, available in SGML-coded form and to store this information in a database. From this database we can then create secondary publications of various kinds, such as volume indexes, author indexes and abstract journals. Also for this type of database-publishing, TEX is used to format the structured information on paper.

Conclusion

In this paper I have sketched the ideas that *Elsevier Science Publishers* have developed concerning the handling of author prepared manuscripts in electronic form. In this scheme, SGML will play a central role, and TEX, in one or more of its varieties, will also play a part.

Furthermore, I have given a few examples of other ways in which TEX's text formatting capabilities can be put to good use, namely in the fields of book production and database publishing.

Summarizing, I think it's no exaggeration to say that, because of its programmability, TEX is eminently suitable for large-scale text production, both directly as a text processing system, and indirectly as the output component of a database-publishing system. Considering that

- the quality of material typeset in TEX is considered satisfactory even by very demanding users,
- TEX runs on an impressive number of different types of computers, and
- TEX output can be printed on a wide range of output devices,

the number of uses of TEX will undoubtedly increase in the years to come.

Bibliography

[1] International Standard ISO 8879: *Standard Generalized Markup Language (SGML)*. ISO (1986).

[2] Martin Bryan: *SGML, an author's guide to the Standard Generalized Markup Language*. Addison Wesley, (Workingham, 1988)

[3] Eric van Herwijnen: *Practical SGML*. Kluwer Academic, (Dordrecht, 1990).

[4] Leslie Lamport: *LATEX, a document preparation system*. Addison Wesley, (Reading MA, 1986).

Getting TEXnical: Insights into TEX Macro Writing Techniques

Amy Hendrickson
TEXnology Inc., 57 Longwood Avenue, Brookline, MA 02146
Internet: `amyh@ai.mit.edu`

Amy Hendrickson presented her paper, Getting TEXnical: Insights into TEX Macro Writing Techniques, at the TUG meeting at Texas A&M in June, and at the TUG meeting at the University of Cork in Ireland in September. For a copy of this presentation, please refer to TUGboat Volume 11, Number 3, September, 1990. — Ed.

BibTeX Reconsidered

Dr. Reinhard Wonneberger
Electronic Data Systems (Deutschland) GmbH
Eisenstraße 56 (N15)
D-6090 Rüsselsheim
Federal Republic of Germany
Bitnet: `qzdmgn@ruipc1e`

Frank Mittelbach
Electronic Data Systems (Deutschland) GmbH
Eisenstraße 56 (N15)
D-6090 Rüsselsheim
Federal Republic of Germany
Bitnet: `pzf5hz@ruipc1e`

Abstract

The article contains a general discussion of BibTeX, the proposals discussed at the EuroTeX89 meeting in Karlsruhe, and further proposals for extensions and changes to BibTeX 0.99c.

Contents

1 Introduction **123**

2 Alternative approaches to citing **124**
- 2.1 Presentational citing. 124
- 2.2 Short title citing. 124
- 2.3 BibTeX in the industrial field. 126

3 Interaction with LaTeX **126**
- 3.1 Multiple bibliographies. 126
- 3.2 Title formatting. 126
- 3.3 Formatting commands. 127
- 3.4 Decentralized bibliography. 127
- 3.5 Sequential citations. 127
- 3.6 Internal names. 127

4 BibTeX input **127**
- 4.1 Source organization. 127
- 4.2 Supporting multiple bibliographies. . 128
- 4.3 Language considerations. 128
- 4.4 Style considerations. 129
- 4.5 Entries and fields. 130
- 4.6 Crossreferencing. 131
- 4.7 Compatibility with professional bibliographic databases. 132

5 BibTeX output **132**
- 5.1 'Comefrom'-information. 132
- 5.2 Error messages. 132
- 5.3 Database maintenance. 132

6 BibTeX internal commands **133**
- 6.1 `.bst` Stack language documentation. 133
- 6.2 The incompleteness of the language. 133

7 Implementation and organization **133**
- 7.1 Sharing bibliographic information. . 133
- 7.2 Software maintenance with BibTeX entries. 133
- 7.3 BibTeX status in the TeX community. 134

References **134**

Index of proposals **136**

1 Introduction

The more authors are won by TeX's outstanding quality of typesetting, the more it becomes obvious that typesetting is only one, though the core, step in producing a document.

What is needed in addition may by summarized by the term *document support.*[1]

One important part of document support is the concept of *General Markup* as realized in LaTeX, DCF *Generalized Markup Language* [21], and, on the higher level of a *Document Definition Language*, by SGML [13, 7].

But two standard tasks of Document Support are not included in LaTeX. One is indexing [9], the other one is producing bibliographies. The latter is handled by BibTeX, which was developed by Oren

1. On the following, see the discussion and practical examples in [42][181ff] and [4, 45].

PATASHNIK in close cooperation with LESLIE LAMPORT. BIBTEX was designed to work in close connection with LATEX, but can also be used with other tools or as a stand alone program.[2] A comparison with bibliographic tools of other platforms can be found in [33].

For a long time, BIBTEX was a niche product for experts only. This seems to be changing now:

1. A BIBTEX requirements session was held at the EuroTEX meeting in Karlsruhe [6], resulting in a list of requirements that was spread out in [44] and published in [1, 24]. It is one of the purposes of the present article to elaborate these requirements.
2. BIBTEX was discussed recently in TEXline [36, 34, 14]. The TEXline newsletter also began to print bibliographic information in BIBTEX source format.
3. At our site at EDS, we introduced BIBTEX about one year ago to help us with the large amount of inhouse and vendor documentation, which has to be handled in our industrial environment [47].

Thus it is one of the intentions of this article to help remove some of the obstacles which may prevent others from realizing BIBTEX's benefits.

Although we will speak about alternatives and make quite a lot of suggestions, we should point out right away that we do appreciate what has been achieved so far. BIBTEX is an excellent tool when used for the task it was designed for. Most of the problems arise because there are other needs that were not considered in the original design. The following lines are not more than a first try to present a collection of what might be desirable. Most of them adopt a user's, not an implementor's, point of view.

We are going to begin with a discussion of alternative concepts of citing. Next, we treat the links to LATEX, and finally we deal with some issues confined to BIBTEX itself.

2 Alternative approaches to citing

In many scientific areas, the citing information given in the text is kept to a minimum, e.g. a reference number. In some areas, however, different approaches to citing are required, and we shall consider a few of them here.

2.1 Presentational citing. The standard citing concept of BIBTEX is based on the assumption that references are mainly used in rare circumstances for verification, but not read in the same way as the normal text.

There are, however, quite a lot of applications where bibliographic information is meant to be *read*, either alone or together with normal text. Such cases of *presentational citing* also need a different technical approach.

To get a vivid impression of why presentational citing might be desirable, the gentle reader of this paper might have a close look at footnote 3 on page 124.

2.1.1 Footnote citing. In some cases, authors may want their readers to see immediately those bibliographic items they are referring to.

Assembling such references in the back of the book or article will not do; it only makes sense if they are used rarely, so this case is quite similar to the alternative of footnotes vs. endnotes.

A good way to make bibliographic information easily accessible is to include it into footnotes; for an example, see [42]. This is discussed in more detail in Proposal 10 'Citefull style' on page 127.

2.1.2 Commented bibliographies. A similar question arises with commented bibliographies, where the commenting descriptions are to be mixed with the bibliographic information and *section* commands are to be used.

2.1.3 Publication lists. To document their own publications (*publish or perish!*), all scholars need publication lists which have some technical peculiarities.

Proposal 1 (Publication list)
Implement a `pubslist` *bibliographic style, with the following features: – omission or abbreviation of author's name, – chronological sequence, – entry related display of reviews (cf. Proposal 28 '*`@review`* entry' on page 130).*

2.1.4 Multiple bibliographies. In some cases, especially with textbooks for teaching, one would like to have bibliographies chapterwise. For a more technical discussion of this issue, see Proposal 14 'Selected bibliography generation' on page 128.

2.2 Short title citing. It may help to understand some of BIBTEX's design limitations to have a look at another system, which was used to produce the bibliographies for some books of REINHARD WONNEBERGER.[3] In this system, a bibliographic entry looks like this:

```
:b t=l .label
:b     t=a .Author / Editor
```

2. The first release (up to 0.98) is described under the roof of LATEX [25, 39]. Substantial enhancements were introduced with release 0.99 [32]. The syntax for the bibliographic entries is upward compatible, but the style files have to be redone [31].

3. [49] (explained in [42]), the first edition of [46] and the German edition thereof [41], and also [40] (which was typeset by the author's own Scriptor program), and [48].

```
:b      t=t .Title
:b          .(continuation line, for any type)
:b      t=u .Subtitle
:b      t=r .Series / Journal
:b      t=o .Address
:b      t=p .Publisher
:b      t=j .Year (edition)
:b      t=v .>out of<
:b      t=s .internal information
:b      t=e .(end mark)
```

The `:b .` is a GML-tag [21]; it carries a type attribute to specify the bibliographic field given in the remainder of the line. The '`v. >out of<`' field is used to refer to separate entries describing parts of a book, e.g. articles of a *Festschrift* (see below).

It is quite obvious that this system is far simpler than BIBTEX; it does not include adaptations along the lines of `.bst` bibliographic styles. But, in addition to some technical features worth noting, it may show some different attitudes and requirements when writing in fields like theology, literature, linguistics and the like.

1. With the few fields mentioned above, it was possible to accomodate more than 2,000 entries of very different kind. Typing is quite easy, and the simple input structure helps to get a quick and safe orientation in the source files, cf. Proposal 12 'Structured source layout' on page 128.
2. The `r`-field is ambiguous, being used for journals, e.g. `ZAW 92 (1980) 185-204`, in which case the year is given here, and for series, e.g. `BZAW 123`, in which case a separate year field is used. Though it would need some analysis to extract the year of an article, this scheme is very efficient for input.
3. The most interesting aspect is perhaps the inclusion of *incollection* type items. The need arises because books and the parts of them, e.g. articles, always get different entries. Let us have an abbreviated example, the first entry giving the collected volume, the second one giving the article contained in it:

```
t=l .Weydt Partikeln
    t=a .Weydt, Harald (ed.)
    t=t .Die Partikeln ...    [book]
    t=j .1979
    t=v .>Wonneberger Pragmatik<
t=l .Wonneberger Pragmatik
    t=a .Wonneberger, R.
    t=t .Zur Syntax ...    [article]
    t=r .Weydt Partikeln 488-499
```

   The interesting point is that the bibliography (corresponding to the `.bbl` file) is produced in *one pass.*[4]

4. There is also a difference in the citation mechanism. References to literature are done as index entries, which are also printed in the text; to give an example, the text entry `:s.12 Wonneberger Normaltext 99`, which will appear in print as '*Wonneberger Normaltext* 99', also produces an index entry. This gives an index of quoted literature, which is quite useful in the writing of bigger manuscripts. A file with bibliographic requests is produced during index processing. As entries are sorted anyway for the index, no special sort is needed to get the bibliographic request file into alphabetic order. All special symbols like *umlauts* are replaced by a transcription like 'ae', 'ss' and the like during label processing.

The last point seems to be of general importance. As an author, I do not want to be dependent on several LATEX and BIBTEX runs to get my cite keys resolved. I want to code an explicit short title in my source, the meaning of which is immediately clear to myself and to others working in this field, and I also want to identify who was cited where.

Proposal 2 (Explicit labels)
Allow explicit labels, like "`Wonneberger Normaltext`*", which can be printed directly and used with* `\nocite`. *Umlauts can be normalised.*

Remarks
A similar system is used in linguistics, giving author *and* year, *e.g.* Chomsky (1957c). *This case seems to be more complicated, because punctuation may be subject to stylistic guidelines, and counting relative to the year is context dependent.*

It is important to understand that these different approaches to citing are by no means just a matter of taste. In most cases they reflect a specific functionality and perhaps also general attitudes of a community of scholars.

Speaking in terms of Computer Science, the familiar BIBTEX method is *addressed* citing, whereas these methods can be seen as *associative* citing, which means: addressing by content.

Addressing does not matter as long as it is kept inside the machine. It becomes a handicap if the user is involved.

The question of associative citing also evokes the topic of databases (see below).

4. When the list of bibliographic requests (corresponding to `\citation`) is matched with the bibliographic database, all items are checked for a possible 'v' field. Then the program goes the other way round and checks whether the label framed by angle brackets is requested in the bibliographic requests list. It is not checked, however, whether the `r`-field matches with the book label.

2.3 BIBTEX in the industrial field. At first sight BIBTEX might seem to be an academic tool. This is true because of the academic need for quoting others.

But an important benefit of BIBTEX can only be realized by sharing bibliographic information, and sharing resources might be more common in an industrial environment.[5]

There are some requirements which result from industry needs. Internal labels will not normally be built on authors' names, as many publications do not show their authors at all. In these cases it may be best to build on the publication number, which, as a rule, is also printed on the publication itself.[6]

In an MVS environment one might wish to have only one document identifier which can serve both as a label and as a member name for source and object forms of the document. This, however, would impose the restrictions for member names on the label.[7]

Proposal 3 (Worldwide convention(s) for labels)
Agree on a standard for the formation of cite-keys.

In an environment with demand printing, some publications will be available for general printing, e.g. [47] is made available as [43]. Instead of using two separate entries, it might be nice to have fields that can be switched on and off dynamically. This concept is also important for maintaining additional information within one entry which should be used only sometimes, e.g. prices, etc. While it is possible in principle to write different style files that support different fields, this overhead is error prone.

Proposal 4 (Switchable fields)
Support the inclusion of individual fields on request.

Remarks
This can be achieved with the help of the `@like` *option (cf. Proposal 26 'Like option' on page 130) since unwanted fields can then be identified with ignored fields. Alternatively, it could be supported in a substyle concept (cf. Proposal 22 'Substyle concept' on page 129).*

3 Interaction with LATEX

3.1 Multiple bibliographies. BIBTEX was designed for documents containing one reference list, i.e. one `thebibliography` environment. Therefore it was sensible to read the citation information directly out of LATEX's `.aux` file. If more than one bibliography is necessary, one can circumvent the problem by using the sub-level `.aux` files provided by the `\include` command and reading the resulting `.bbl` files manually.[8]

Alternatively the `\bibliography` command can be changed slightly to read in a `.bbl` file corresponding to the current include file. This will give us one bibliography per `\include` command, a command that will always start a new page. Therefore this mechanism is not adequate for journals in which the articles are printed just one after the other. It is also not usable for manuals, etc., that have several reference lists, one after the other, e.g. 'Further reading', 'Related publications', etc.

A second alternative is to change the `\cite` command to write the citations not into the `.aux` file but into other files, one for each wanted reference list. This will allow one to use the resulting BIBTEX output at any place, but will require a considerable number of redefinitions of LATEX commands. Furthermore, both methods are very time consuming and error prone because BIBTEX has to be started for every file generated.

Proposal 5 (Multiple bibliographies (LATEX))
There are plans to extend the bibliography support in the re-implementation of LATEX in such a way that several bibliographies within one document (all generated by BIBTEX*) are possible. The syntax for this feature is currently under discussion.*

Remarks
This support is only possible in an efficient way if the BIBTEX *input syntax is changed according to Proposal 14 'Selected bibliography generation' on page 128.*

3.2 Title formatting. To use BIBTEX for the formatting of a title page could be another useful and efficient application. It would be an advantage

5. See the corresponding considerations for TEX in [47].

6. The IBM system of form numbers shows how a very big publisher handles this [20]:

`ABBB-CCCC-EE`	Typical IBM Documentation Number
`A...-....-..`	Use Key, e.g. `G`: generally available, `S`: for sale; `L`: for licensees
`.BBB-CCCC-..`	Form Number
`.BBB-....-..`	Prefix, e.g. one number for all DCF publications; specific prefixes denote publication categories like Logic Manuals, Bill of Forms, Microfiche, Pseudonumber, Technical Newsletter, Supplement; for details see [20].
`....-CCCC-..`	Base Number relative to the prefix
`....-....-EE`	Suffix, e.g. 00 for first edition

The first three elements might be used right away as a BIBTEX label, assuming that new editions should be quoted automatically.

7. Max. 8 alphanumerics, starting with a letter (including the national characters `@`, `#`, `$`).

8. The new LATEX will use only one `.aux` file, so that this mechanism will not work any longer. For further information see [27].

that arbitrary information could be gathered in the BibTeX database while each journal provides its own title-`.bst` that picks up the information of interest. This style would then produce an output file containing LaTeX commands for formatting a title according to the house style. An additional benefit would be that each paper produced in this way would already contain its own bibliography entry in BibTeX format.[9]

Proposal 6 (BibTeX and title page)
Evaluate whether such a scheme is feasible and easy to use. If so, devise new LaTeX commands that handle this concept in a user friendly way.

Remarks
This clearly requires multiple bibliographies (cf. Proposal 14 'Selected bibliography generation' on page 128), as one cannot ask the user for an additional BibTeX *run to produce a title page. Such a feature should be clearly considered as a suggested add-on, but a conventional way to produce a title in a document should still be available.*

BibTeX can be particularly helpful in providing title information for software in a complete and standardized way.

Proposal 7 (BibTeX software titling)
Develop a concept for self-contained software title information along the proposal in [2][486–487].

3.3 Formatting commands. With the release of BibTeX 0.99 it is possible to include LaTeX commands in the bibliography database that will be copied by BibTeX to the very beginning of the `.bbl` file. This allows one to specify formatting declarations and to introduce new commands which will be used in the bibliography entry. But LaTeX supports only `\newcommand` and `\renewcommand`. If for example the database contains the line

```
@preamble(\newcommand{\WEB}{{\sc web}})
```

it is not possible to use this database together with the `ltugboat` style option because the command `\WEB` is already defined in this style and the `\newcommand` will therefore balk.

Proposal 8 (Supply command)
Add a `\supplycommand`[10] *macro to LaTeX which behaves like* `\newcommand` *if the command name is previously undefined but does nothing (or prints only a warning) when the command already has a meaning.*

3.4 Decentralized bibliography. Some document types require bibliographic entries to be placed directly into the text or into a footnote, cf. section 2.1 on page 124. To support this, BibTeX should be changed to allow sequential reading of the `.bbl` file.

Proposal 9 (Singularize `.bbl`)
Change the output of BibTeX *by surrounding the whole entry with a pair of braces, i.e.*

```
\bibitem{Kn86}{%
  Donald E. Knuth. \newblock
  ...
  }
```

Remarks
Using the `\read` *command to access the* `.bbl` *information would forbid commands that change* `\catcode` *(like* `\verb`*) in the database.*

3.5 Sequential citations. As mentioned in section 2.1 on page 124, a form of sequential citing, e.g. in footnotes, is a common requirement in certain disciplines. This should be supported by LaTeX at least via a style option.

Proposal 10 (Citefull style)
Write a LaTeX style option that supports this sort of citation. This would necessarily include a new `\citefull`[11] *command that marks the place where the full entry should be cited. The* `\cite` *would then refer to this place.*

Remarks
This requires a different syntax in the `.bbl` *file as explained in Proposal 9 'Singularize* `.bbl`*' on page 127.*

3.6 Internal names. After looking into an `.aux` file many users get confused by the fact that user commands (e.g. `\cite`, `\bibliographystyle`, etc.) have nearly identical counterparts in internal commands to be understood by BibTeX. Unfortunately these commands (e.g. `\citation`, `\bibstyle`, etc.) are user accessible, which may lead to confusion.

Proposal 11 (Change internal names)
Change the internal names that are written by LaTeX into the `.aux` *file, so that they cannot be accessed by the ordinary user.*

4 BibTeX input

4.1 Source organization.

9. This is similar to the practice to print a short bibliographic entry as provided by the national library on the backside of the title page.

10. The name is only a suggestion. A simple definition for macros without parameters could be:

```
\def\supplycommand #1#2{\ifx
  \undefined #1\def #1{#2}\fi}
```

11. The name is chosen to apear near `\cite` in the alphabetical sequence and to allow incremental search in appropriate editors.

4.1.1 Uniform source layout. BIBTEX allows identical information to be coded in different syntaxes. For large bibliographies, it is important to find a standard format of coding, cf. the rather uniform source layout of the Reduce bibliography [12].

Proposal 12 (Structured source layout)
Develop a convention to structure BIBTEX *source files for better readability.*

Remarks
This also implies agreed positions for commas and a convention about braces vs. brackets vs. quotes.

4.1.2 Error recovery. "In practice, we all make mistakes. And one of the most common typographic errors is to forget a '}', ... ". These words of Knuth [23][205] about TEX also apply to BIBTEX. If braces or quotes do not match properly, it can be quite difficult to detect the corresponding location. In some cases, the error will be reported only at the end of the file, and many entries may have been skipped.[12]

Proposal 13 (Entry separator)
Define an empty line to act as an entry separator.

Remarks
This strategy is similar to the one adopted for `\par` *in TEX [23][205]. It would only prevent empty lines from occuring inside entries, which does not seem to be a big loss.*

4.2 Supporting multiple bibliographies. As explained in section 3.1 on page 126 one of the important shortcomings of BIBTEX is the missing support of several bibliographies within one document. This could be incorporated by changing the syntax of internal commands in the following way:

Proposal 14 (Selected bibliography generation)
Change the commands recognized by BIBTEX *in the* `.aux` *file so that they accept two arguments with the following meaning:*

- *The first argument is a list of context selectors that will help* BIBTEX *to decide which citations to choose for a particular* `thebibliography` *environment, which data bases to search, and which style to use for the bibliography to generate.*
- *The second argument holds the cite key, the style, or a list of databases, respectively.*
- *A context selector is a unique string of at most eight characters.*[13]

In detail, the internal commands found in the `.aux` *file should be interpreted in the following way:*

`\bibstyle` *If the first argument is empty, generate a bibliography containing all citations searching all databases. Otherwise, the first argument contains a list of context selectors. In this case, generate for every context selector a bibliography considering all citations and databases that have been specified for this selector.*

More than one `\bibstyle` *command in the* `.aux` *file is allowed. But every context selector should be used only once.*

`\bibdata` *If the first argument is empty, the second specifies a list of data bases which should be used to search for* all *citations. Otherwise, the data bases from the second argument are searched only if the current citation contains at least one of the context selectors.*

Again, more than one `\bibdata` *command is allowed. Therefore, the problem of a buffer overflow (when writing to the* `.aux` *file) can be prevented.*

`\citation` *If the first argument is empty, the citation will be included in every bibliography generated by the* `\bibstyle` *commands in the* `.aux` *file. If it contains a list of context selectors the citation will show up in every bibliography generated for them.*

Remarks
The internal command names should be changed according to Proposal 11 'Change internal names' on page 127.

It seems advisable to process different `\bibstyle` *commands sequentially to avoid the problem of memory overflow.*

4.3 Language considerations. A general presentation of language-related problems can be found in [17].

4.3.1 Language flags. With TEX 3.0 it is possible to support several languages within one document. This is important for hyphenation, use of fixed names (like "volume", "and", etc.) but also for direct entry formatting. There is, for example, no decapitalization of titles in German.

Proposal 15 (Language flag)
Add a language flag for entry processing by both BIBTEX *and TEX.*

Remarks
This should be available for every field, but the possibility to specify a language for the whole entry (cf.

12. For the time being, such errors may be detected by changing all entry escape symbols '`@`' into something like '`",})@`'. These closing delimiters will trigger error messages if the matching is wrong; otherwise, they will be ignored.

13. This restriction seems to be necessary to allow a standard naming scheme for the resulting output files. Of course, the possibility to construct the output file names from the TEX job name (e.g. `\jobname.bbl`) has to be given up on most systems.

Proposal 34 'Language field' on page 131) is equally important.

Until this has been achieved, standard styles should be language independent, e.g. *Computer Physics Communications 61* instead of *volume 61 of Computer Physics Communications.*

Proposal 16 (Language independent standard styles)
The BIBTEX *standard styles should be language independent.*

4.3.2 Symbols. Special symbols like *umlauts* play an important role in an international environment.

Proposal 17 (Symbol convention)
Though BIBTEX *makes provisions for some special cases, it should also be able to handle language related symbols (e.g. those from* german.sty *[30, 29]) and allow outputting them in a different syntax.*

To produce output for other typesetting systems, e.g. for DCF, BIBTEX should be able to output language related symbols in a different syntax. To give an example, the umlaut `{\"a}` should be converted to `&a.` for DCF.

Proposal 18 (Symbol conversion)
Set up symbol conversion with the help of an additional style file at run time similar to the way MakeIndex *handles different syntax conventions for input and output [9].*

Proposal 19 (Composed symbols)
Composed symbols should be generated. Roman figures, e.g. should be kept as arabic numbers, with some kind of a transformational prefix like in DCF, *where* `&R'4.` *will give 'IV', whereas* `&r'6.` *will give 'vi'.*[14]

4.3.3 Sorting. Different character sets require different sorting rules [22]. In the current BIBTEX, the use of accented characters in author names or titles often leads to incorrect bibliographic sequences in the output.

Proposal 20 (Sorting rules)
Devise some mechanism to incorporate customizable sorting rules into BIBTEX.

Remarks
The same module should be used for MakeIndex, *where similar problems arise.*

4.3.4 Names. The handling of names in BIBTEX is very sophisticated. Nevertheless it cannot handle all classes well enough for general applications. For example, titles in the name fields are misplaced in certain styles,[15] and monks and nuns should be allowed to have their congregation indicated, e.g. 'James Swetnam, S.J.'.

Proposal 21 (Names and titles)
Reconsider the BIBTEX *name function and extend it in such a way that it can deal with foreign names.*

Under the present scheme, agree to write all names with a comma, so that also Spanish names will be unambiguous, e.g. 'García Márquez, Gabriel'.

Remarks
Alternatively, one could give the style file more access to reprogram the function, but this would not solve the problems of an international data base.

4.4 Style considerations.

4.4.1 Substyles. At the moment a BIBTEX style is responsible for both the logical structure of the input (e.g. which fields and entries are recognized) and the structure of the output (e.g. sorting, label generation, formatting of the entries, etc.). Since these tasks are, at least in principle, unrelated to each other, it would be better to separate them.

Proposal 22 (Substyle concept)
Determine a substyle concept for bibliography styles that allow to change certain aspects of the formatting and input recognition similar to the `\documentstyle` *command of LATEX. This concept could be used to handle the requirements discussed in Proposal 20 'Sorting rules' on page 129, Proposal 34 'Language field' on page 131, and Proposal 26 'Like option' on page 130.*

4.4.2 Special styles. When writing documents it is helpful to have a printed version of the BIBTEX database which shows the names of the citation keys.

Proposal 23 (Cite style)
Add a substyle to BIBTEX*'s standard distribution that additionally shows the citation keys.*

Remarks
Such a style could be derived directly from the standard styles. Using a suitable preprocessor (see Pro-

14. The following macros could be used in a LATEX style file to interpret such input:

```
\def\R'#1.{\@Roman{#1}}
\def\r'#1.{\@roman{#1}}
```

15. Dr. is part of the name in Germany, e.g., so one should get 'Dr. R. Wonneberger', but 'Dr. Wonneberger, Reinhard' or 'Wonneberger, Reinhard, Dr.', if the title is not dropped at all, as it is customary in some academic fields. Curious as it may sound, the title was added in the second edition of [39] without the the author's knowledge.

posal 25 'Style distribution' on page 130), it should be no problem to provide it. An example style was given in [11]. Alternatively, this can also be achieved with a suitable style option in LaTeX.

When submitting a paper like this one for publication, the bibliography will have to be sent in `.bbl` format. It is very difficult then for the editor to make modifications to the bibliographic style, in contrast to the fact that they can be done quite easily with a LaTeX `\bibliographystyle` command.

Proposal 24 (Collect style)
Add a substyle to BibTeX*'s standard distribution that will collect the subset of bibliographic entries required for the document and write it into a separate* `.bib` *file.*

Remarks
Such a style will also allow editors to produce a combined bibliography for the whole volume, provided that labels can be mapped as described in Proposal 38 'Multiple labels' on page 131.

4.4.3 Documentation and distribution. The main bibliography styles (`abbrv`, `alpha`, `plain` and `unsrt`) are derived from one master source (`btxbst.doc`) which is written using a C preprocessor syntax. The individual styles are then extracted by setting individual preprocessor flags. Unfortunately the choice of a C preprocessor seems to be tied to a UNIX environment. This means that installations without a C compiler must rely on supplied `.bst` files. The amount of time involved to customize a set of `.bst` files in such an environment is extremely high because every `.bst` file has to be changed individually.

Proposal 25 (Style distribution)
Supply and document all standard styles with a tool that is generally available in all TeX distributions. Since `WEB` *has the disadvantage of being tied to* PASCAL, *we propose using the extended version of* `doc.sty` *and* `docstrip.tex` *[26] which is capable of processing conditional code.*

4.5 Entries and fields. As shown by David Rhead [34], the names and meanings of entries and fields provided by the standard styles do not conform to the classification given by [8], [38], or [5]. But even if a more standardized set of entries and fields were implemented, one would certainly find exceptional applications of BibTeX that involve just another class of documents. The problem of different naming conventions is that it will be difficult to merge databases. This can be circumvented if BibTeX is able to interpret field and entry names in a database depending on exception rules.

Proposal 26 (Like option)
Add a `@like` *entry that might have the following syntax:*

`@like(`⟨*entry1*⟩`)(`⟨*entry2*⟩`)(`⟨*linklist*⟩`)`

This entry will tell BibTeX *that any* ⟨*entry1*⟩ *in the data base should be treated as an entry with name* ⟨*entry2*⟩*. This sort of command is useful only if both entries have similar fields. The* ⟨*linklist*⟩ *should be used to set up conversions between individual fields, e.g. the command*

```
@like(edsmanual)(manual)
(intnote= note
printnote= note)
```

tells BibTeX *that it should interpret an* `@edsmanual` *entry as a* `@manual` *entry and convert any* `intnote` *or* `printnote` *field into a* `note` *field. The* ⟨*linklist*⟩ *can be empty.*

Remarks
These additional commands can be placed in a separate database that is loaded first. The example above can have the result that after conversion more than one field of the same sort exists. This should be handled according to Proposal 33 'Field loops' on page 131.

4.5.1 New entries. There are many possible additions to the current set of standard entries. A proposal for a rather different set of entries is given in [34]. If the current set of entries is kept as a basis, we would suggest adding at least some new ones.

Proposal 27 (`@journal` entry)
At the moment there is no good way to cite an entire journal. This would be required for cross references but is also of interest to draw attention to general topics covered in a journal without citing specific articles.

Proposal 28 (`@review` entry)
A review will typically have to `crossref` *another bibliographic entry as part of its title.*

Review being a publication class of its own, it should be identifiable by its entry name.

Proposal 29 (`@product` entry)
Citing a product (e.g. program, hardware, etc.) is nearly impossible since no official entry contains suitable fields.

Proposal 30 (`@institution` and `@person` entry)
These entries should accomodate all kinds of addresses and allow `person` *to* `crossref institution`.

4.5.2 Fewer entries. As opposed to adding further entries, it might be better to use only a few fundamental ones like `monograph`, `periodical` etc., cf. [34], and give a more detailed classification in a `type` field.

Proposal 31 (Review entries)
Discuss entry concept.

4.5.3 Stacked entries. In some cases, the *incollection* relation involves more than two levels, e.g. this paper is part of a proceedings volume which in turn is part of a journal.

Proposal 32 (Stacked entries)
Allow stacked entries.

4.5.4 Repeated fields. In many cases it makes sense to have identical fields within one entry. Consider, for example, a citation giving the years for two editions like: "1.Aufl. 1980, 2.Aufl. 1982". It is also useful simply to combine certain information as explained in Proposal 26 'Like option' on page 130.

Proposal 33 (Field loops)
Extend the BIBTEX *style language to handle multiple fields. If desired by the style designer, it should be possible to express that certain fields are bound together in the sense that the designer can access the first occurrence of a number of fields (e.g.* `year` *and* `edition` *in the above example), then the second and so forth.*

Remarks
If a field of an entry is not marked as multiple *in the sense above, all occurrences of this field should be simply concatenated with a programmable separator string.*

4.5.5 Supported fields. A language field should be available for the language of the publication.

Proposal 34 (Language field)
Add an optional field `language` *to each entry, specifying the language used within the entry. If no field is specified, a default language should be used.*

Remarks
The default language can be implemented as part of a substyle concept (cf. Proposal 22 'Substyle concept' on page 129). The language of individual fields (see Proposal 34 'Language field' on page 131) should always take precedence over the general or default language. This should not be confused with the case where the book is in a language different from the bibliographic entry, e.g. Hebrew and Russian publications when quoted with the bibliographic information in English. In this case the original language can be placed into the `note` *field.*

One important piece of information for published items is their *International Standard Book Number* (ISBN) or *International Standard Series Number* (ISSN), which are useful to identify and order publications.

Proposal 35 (Number field)
Support a field for ISBN and ISSN numbers or a general number field for all entries.

Remarks
It is possible that several numbers exist for one entry (e.g. a hard- and softcover version for [23], etc.); see Proposal 33 'Field loops' on page 131 for ideas on handling this situation.

The current use of `booktitle` is difficult to understand.

Proposal 36 (`booktitle`)
A better scheme should be devised for the `booktitle` *concept in connection with* `crossref`. *At least, booktitle should be made optional.*

4.5.6 Unsupported fields. In the current BIBTEX implementation, there is one field, namely `annote`, which is not supported by the standard BIBTEX styles. By defining such names BIBTEX databases are kept portable.

Proposal 37 (Standardize extended field names)
Add names for all fields to the user documentation of BIBTEX *which are of interest in extended styles. This would include fields like* `abstract`, *etc.*

4.5.7 Labels. As long as labels are not defined by the publisher, different labels will be used for one and the same element. This prevents an easy combination of databases.

Proposal 38 (Multiple labels)
Discuss multiple cite labels or alias-fields (to allow different systems of reference, e.g. in different user groups), or a scheme of citing by content [33].

Remarks
An application for this is given in Proposal 24 'Collect style' on page 130.

4.6 Crossreferencing. BIBTEX's possibility to crossreference entries (introduced in version 0.99) is restricted to a fixed order of entries (referenced entries must follow later in the data base). This is necessary since BIBTEX is essentially a one-pass system. On the other hand, this poses great problems in maintaining huge databases.

Proposal 39 (Multi-pass system)
Remove this restriction by reading the data base several times if necessary.

Remarks
A multi-pass system is also necessary to support Proposal 14 'Selected bibliography generation' on page 128 and perhaps also Proposal 22 'Substyle concept' on page 129.

Another possibility is to allow backward references from whole publications to parts. This would allow for processing everything correctly in one pass and help to keep track of consistency.

Proposal 40 (Backward references)
Consider this approach as an alternative to the current concept.

Remarks
This would also allow to identify cited parts of a collect volume under its entry.

The concept of crossreference might be extended to other areas of application.

Proposal 41 (Extended crossreferencing)
Allow more than one crossreference for an entry.

Remarks
This would allow to crossref persons and institutions, e.g. authors with their affiliation, or publishers with their address, in addition to the normal crossreferencing of collect volumes.

4.7 Compatibility with professional bibliographic databases. Further development should consider standards like RAK. RAK (*Regeln für die alphabetische Katalogisierung*) is the present standard for German scientific libraries (and a successor to the so-called *Preußische Instruktionen* (Prussian Instructions). In addition to giving rules on how to deal with publications without an explicit author or editor, this standard specifies the full range of fields that are necessary to describe all types of items held in such libraries.

Proposal 42 (RAK compatibility)
Though this standard is far too complicated for nonprofessionals, it might be useful to think about a way to inherit BIBTEX *fields from a professional catalog entry.*

Remarks
This topic could best be dealt with if we could find a professional librarian who is also familiar with TEX.

5 BIBTEX output

When working with larger bibliographies, it is important to have a working printout which also shows internal data. Lookup in a printout according to our experience is much more efficient than lookup in several files.[16]

5.1 'Comefrom'-information.

5.1.1 Date and time. For long-living documents it is often essential to determine when the document was last updated. On many systems, the automatic date and time stamp of a file is not an adequate source of information, because it might change during physical movement of files, etc.

Proposal 43 (Date and time)
Write the date and time of execution into the `.bbl` *and* `.blg` *files using a TEX acceptable format, i.e. starting lines with a percent sign or using appropriate macros.*

5.1.2 Source. In a large bibliography, it can be difficult to locate a specific entry. It is useful to have a working copy which, in addition to the cite label, also shows the file name for each entry.

Proposal 44 (Source information)
Add the source file information to each `\bibitem` *command on demand.*

5.2 Error messages. If braces or quotes do not match properly, BIBTEX may read bulks of lines in the vain hope of finding a good end delimiter (cf. Proposal 13 'Entry separator' on page 128).

Proposal 45 (Error message entry identification)
Identify the entry and field being processed in BIBTEX *error messages to help with spotting long range syntax errors of the input.*

Proposal 46 (Warning message for unknown fields)
Give a warning message for unknown fields, unless declared before, or switched off (cf. Proposal 37 'Standardize extended field names' on page 131).

5.3 Database maintenance. BIBTEX's databases are simple ASCII files. This has the advantage that such files are easily portable. On the other hand, maintaining these files is difficult. If, for example, two large databases are to be merged, unifying the citation keys is a time consuming task. Using the `\cite{*}` feature of BIBTEX can help a little bit in this regard, but if the database has too many entries it will blow up BIBTEX's memory, e.g. the Reduce bibliography [12] with about 40 pages is not processable on most installations.

Proposal 47 (Nonsorting mode)
Add a feature to BIBTEX *that allows sequential reading of arbitrarily large databases. The only information which should be saved is the citation key and*

16. Similar considerations apply to LATEX, where a general `draft` substyle showing 'comefrom' and crossref information is highly desirable.

its position in the database so that multiple entries can be detected.

Remarks
This is not the same as writing a style file which ignores nearly all fields. The latter can be used only to detect multiple citation keys while the proposed feature would allow for printing all information contained in the database.

5.3.1 Comments. In a large database it is often useful to include comments (about the last update, etc.). While this is possible between entries, as long as one does not use one of BIBTEX's special characters, it is not allowed inside a bibliography entry. If, for example, not all information is available, it would be nice to comment out open fields instead of deleting them from the database. This is especially useful in an environment where the user is guided by templates which show all possible fields.

Proposal 48 (Comment character)
Make the percent character a BIBTEX *comment sign to allow comments in the database.*

6 BIBTEX internal commands

6.1 `.bst` Stack language documentation. The 'unnamed language' as described in [31] is difficult to understand even for experienced programmers since it implements the seldom used concept of a stack language. But the main clients of BIBTEX in the future are probably people with only a small or no knowledge in computer science. They will have to change certain aspects of the standard styles and for that reason have to understand the basic principles of the language.

Proposal 49 (Progammers guide)
Expand the BIBTEX *programmers guide in a way that even inexperienced people are able to customize existing* `.bst` *files.*

In any case it is questionable whether the concept of a stack language is natural for the given problem of producing bibliographies.

Proposal 50 (Style language change)
Consider the possibility of changing the style language of BIBTEX *in the future to a different concept.*

Remarks
If the necessity of customizing supplied style files is lessened by introducing a substyle concept (cf. Proposal 22 'Substyle concept' on page 129), one could perhaps leave the language as it is: as an amusing or frustrating experience for a few gurus.

6.2 The incompleteness of the language. When one of the authors (Frank Mittelbach) implemented a `tugproc.bst` according to the guidelines for TUG Proceedings [10], he ran into several problems. It was pretty difficult to convert explicit dashes (i.e., '`--`' and '`---`') into the commands `\dash` and `\Dash` that should be used instead. Problems arose because `text.prefix$` and `text.length$` operate on special characters. Another dilemma was produced by the requirement that punctuation characters should go into the quotes. As a result the `add.period$` could not be used any longer.

Proposal 51 (Extend internal commands)
Check whether standard requirements for bibliographies can be programmed in a straightforward way with the given set of commands; extend this set if necessary.

7 Implementation and organization

7.1 Sharing bibliographic information. It takes quite a lot of know-how, time, and care to produce a good entry for BIBTEX. Much can be gained by distributing precise bibliographic information, as it is shown by the Reduce bibliography [12].

Proposal 52 (`.bib` distribution)
Discussion lists like [35] and similar installments should encourage distribution of bibliographic information in `.bib` *format.*

Even without the sharing of source files, it is much easier to type a ready-made BIBTEX entry from paper (e.g. [36]) than to reconstruct it from normal bibliographic information.

Proposal 53 (`.bib` printout)
Following the good example of TEXline [36], also other publications like TUGboat [37] should print bibliographic information in `.bib` *format.*

7.2 Software maintenance with BIBTEX entries. The quality of software available to the TEX community varies from exceptional to poor. One of the main problems is that, quite often, it is not easy to determine author, release, last update, etc. for a macro package or a program source. This could be prevented if a standard for documentation is developed which describes the relevant facts. Here BIBTEX would be the ideal tool.

Proposal 54 (Software entries)
Agree on a small set of entries with corresponding fields that describe software. This standard should be used within the TEX community for all software distributed. Authors submitting software to servers

should be asked to add such an entry in front of their contribution. Preferably fields like `abstract` *should be included so that this information can find its way directly into a local guide.*

7.3 BIBTEX **status in the TEX community.** With this paper we want to stress the importance of companion programs to TEX, BIBTEX in this case.

Proposal 55 (Standard TEX support)
BIBTEX, *and* MakeIndex[17] *accordingly, should get the same support by the TEX Users Group (TUG) as TEX and LATEX.*

Some of our suggestions will be quite easy to implement even under the current concept of BIBTEX, some others will need extensions, and still others may require a completely different implementation.

Proposal 56 (Academic research)
Doing further research into the questions raised in this paper might be a challenge for students and teachers who are interested in computer science as well as in librarianship. Such research might also foster cooperation among different disciplines and experienced practitioners.

Remarks
DAVID RHEAD, *known for his work on* BIBTEX *[34], has offered to help with coordinating further work. Anyone who wishes to volunteer, or who is in a position to supervise a research student doing further work, is invited to send details of their interests to David.*[18]

The present authors would be pleased to supply their manuscript to serve as a nucleus for further work on BIBTEX.

Proposal 57 (Living document)
Create a living document to display requests, results of discussion, and the status of implementation for BIBTEX.

References

[1] Bailey, Rosemary. "TEX89 looks at LATEX tools." *TEXline*, (10):8, 1990.

[2] Beebe, Nelson H.F. "From the president." *TUGboat*, 11(4):485 – 487, 1990.

[3] Beeton, Barbara. "Report from the question and answer session." In Durst, Lincoln, editor, *1990 Annual Meeting Proceedings*, pages 455 – 458, Providence, September 1990. TEX Users Group. TUGboat 11 (1990/3).

[4] Brandt, Josef. "Computer-aided production of scientific documents." Pages 163 – 176 in *Man-Machine Interface in the Scientific Environment. Proceedings of the 8th European Summer School on Computing Techniques in Physics. Skalský Dvůr, Czecholsovakia, 19 – 28 September 1989*, J. Nadrchal [28]. Invited Paper.

[5] British Standards Institution. *Citing publications by bibliographic references*, 1987. BS 5605.

[6] Brüggemann-Klein, Anne, editor. *Proceedings of the 4th European TEX Conference, September 11th – 13th, 1989*, Karlsruhe, forthcoming. EuroTUG.

[7] Bryan, Martin. *SGML: An Author's Guide to the Standard Generalized Markup Language.* Addison-Wesley, Woking, England; Reading Massachusetts, second edition, 1988.

[8] Butcher, Judith. *Copy-editing.* Cambridge University Press, 2 edition, 1981.

[9] Chen, Pehong. MakeIndex*: A General Purpose, Formatter-Independent Index Processor.* University of California, Berkeley, June 1988. LATEX version of a file distributed with MakeIndex (MakeIndex adaptation manual); source in 'adeit.mkindex.tex(pehong)'.

[10] Durst, Lincoln K. "Guidelines for *proceedings* of the annual meeting of the TEX users group." Received February, 1990.

[11] Hailperin, Max. "Another response to BIBTEX question." *TEXhax*, 90(40), 22 April 1990.

[12] Hearn, Anthony C. *REDUCE Bibliography.* The RAND Corporation, June 1990. Available as `.bib` source; enquiries to network address: reduce @ rand.org.

[13] Herwijnen, Eric van. *Practical SGML.* Kluwer, Dordrecht, NL, 1990.

[14] Higgins, Christopher P. "Cross referencing the bibliography." *TEXline*, (10):8 – 9, May 1990.

[15] IBM National Language Technical Center. *Designing Enabled Products, Rules and Guidelines.* Volume 1 of NLIDG [17], 1987. IBM Order Number: SE09-8001-00.

[16] IBM National Language Technical Center. *Left-to-Right Languages and Double-Byte Character Set Languages.* Volume 2 of NLIDG [17], 1987. IBM Order Number: SE09-8002-00.

[17] IBM National Language Technical Center, editor. *National Language Information and Design Guide.* IBM, 1987, 1988. Consists of [15, 16, 18, 19].

17. MakeIndex even waits to be converted to WEB, which we feel is important in spite of the growing availability of C compilers in order to allow the standard distribution tools like change files etc. to be used, cf. question 2 in [3].

18. On JANET: `d.rhead@uk.ac.nottingham.ccc.vme`, and from other nets: `d.rhead@vme.ccc.nottingham.ac.uk`.

[18] IBM National Language Technical Center. *National Language Information: Arabic Script Languages.* Volume 3 of NLIDG [17], 1988. IBM Order Number: SE09-8003-00.

[19] IBM National Language Technical Center. *National Language Information: Hebrew.* Volume 4 of NLIDG [17], 1988. IBM Order Number: SE09-8004-00.

[20] International Business Machines Corporation. *Entering an SLSS Subscription*, 6 edition, September 1986. G320-1561-05; SLSS: System Library Subscription Service.

[21] International Business Machines Corporation. *Document Composition Facility: Generalized Markup Language. Starter Set Implementation Guide*, 5 edition, March 1988. SH35-0050-04, Release 3.2.

[22] International Business Machines Corporation. *Document Composition Facility: SCRIPT/VS Text Programmer's Guide*, 6 edition, March 1988. SH35-0069-05, Release 3.2.

[23] Knuth, Donald E[rvin]. *The TEXbook*, volume A. Addison-Wesley Publishing Company, Reading, Mass. etc., 1986. Hardcover: ISBN 0-201-13447-0, Softcover: ISBN 0-201-13448-9.

[24] Kruljac, Gabriele. "BIBTEX und MAKEINDEX." *Die TEXnische Komödie. Mitgliedszeitschrift von DANTE, Deutschsprachige Anwendervereinigung TEX e.V.*, 2(1):23–24, 1990. Abdruck der BIBTEX-Anforderungen von der europäischen TEX-Tagung in Karlsruhe, September 1989.

[25] Lamport, Leslie. *LATEX: A Document Peparation System.* Addison-Wesley, Bonn etc., 1 edition, 1986.

[26] Mittelbach, Frank. "The doc-option." *TUGboat*, 10(2):245–273, July 1989.

[27] Mittelbach, Frank and Rainer Schöpf. "Towards LATEX 3.0." In Gunther, Mary, editor, *TEX 90 Proceedings, TUGboat*, 12(1):74–79, Providence, March 1991. TEX Users Group. to appear.

[28] Nadrchal, J., editor. *Man-Machine Interface in the Scientific Environment. Proceedings of the 8th European Summer School on Computing Techniques in Physics. Skalský Dvůr, Czecholsovakia, 19–28 September 1989*, volume 61 of *Computer Physics Communications.* North Holland Publishing Company; Elsevier Science Publishers B.V., 1990.

[29] Partl, Hubert. "How to make TEX and LATEX international." Pages 190–200 in *Man-Machine Interface in the Scientific Environment. Proceedings of the 8th European Summer School on Computing Techniques in Physics. Skalský Dvůr, Czecholsovakia, 19–28 September 1989*, J. Nadrchal [28]. Invited Paper.

[30] Partl, Hubert et al. *LATEX-Kurzbeschreibung.* Technische Universität Wien, Wien, 1 edition, 1986.

[31] Patashnik, Oren. *Designing* BIBTEX *Styles.* Stanford University, January 31 1988. BIBTEX installation file: 'The part of BIBTEX's documentation that is not meant for general users'.

[32] Patashnik, Oren. BIBTEX*ing.* Stanford University, January 31 1988. BIBTEX installation file.

[33] Rahtz, Sebastian P. Q. "Bibliographical tools." *Literary and Linguistic Computing*, 2(4):231–241, 1987.

[34] Rhead, David. "Towards BIBTEX style-files that implement principle standards." *TEXline*, (10):2–8, May 1990.

[35] "TEXhax." ongoing. Modisett, Tiina and MacKay, Pierre (eds.): Moderated Network Discussion List.

[36] "TEXline. [Nr.1-7:] A newsletter of TEX users in UK and Ireland / [Nr.8ff:] A newsletter of the TEX community.." 1ff(1ff), Started 1985. Edited by Malcolm W. Clark.

[37] "TUGboat. The communications of the TEX users group." 1 ff(1 ff), 1980 ff. Barbara Beeton (ed.); subtitle for vol. 1–8: 'The TEX Users Group Newsletter'.

[38] University of Chicago. *The Chicago Manual of Style*, 13 edition, 1982.

[39] Wonneberger, Dr. R. *Kompaktführer LATEX.* Addison-Wesley Kompaktführer. Addison-Wesley, Bonn etc., 2 edition, Nov. 1988. ISBN 3-89319-152-6 (VVA-Nr. 563-00152-0)

[40] Wonneberger, Reinhard. *Syntax und Exegese. Eine generative Theorie der griechischen Syntax und ihr Beitrag zur Auslegung des Neuen Testamentes, dargestellt an 2.Korinther 5,2f und Römer 3,21-26.*, volume 13 of *Beiträge zur biblischen Exegese und Theologie (BET).* Peter Lang, Frankfurt / Bern / Las Vegas, 1979.

[41] Wonneberger, Reinhard. *Leitfaden zur Biblia Hebraica Stuttgartensia.* Vandenhoeck & Ruprecht, Göttingen, 2 edition, 1986. ISBN 3-525-52180-4.

[42] Wonneberger, Reinhard. ""Verheißung und Versprechen" — A third generation approach to theological typesetting." In Désarménien, Jacques, editor, *TEX for Scientific Documentation. Second European Conference, Strasbourg, France, June [19-21], 1986. Proceedings*, number 236 in Lecture Notes in Computer Science, pages 180–189, Berlin / Heidelberg / London /

etc., 1986. Springer. Describes the typesetting of [49].

[43] Wonneberger, Reinhard. "TEX in an industrial environment." Preprint of [47], EDS, 1989.

[44] Wonneberger, Reinhard. "TEX yesterday, today, and tomorrow." *TEXhax*, 90(5), 7 January 1990.

[45] Wonneberger, Reinhard. "Structured document processing: the LATEX approach." Pages 177-189 in *Man-Machine Interface in the Scientific Environment. Proceedings of the 8th European Summer School on Computing Techniques in Physics. Skalský Dvůr, Czecholsovakia, 19-28 September 1989*, J. Nadrchal [28]. Invited Paper.

[46] Wonneberger, Reinhard. *Understanding BHS. A Manual for the Users of Biblia Hebraica Stuttgartensia*, volume 8 of *Subsidia Biblica*. Editrice Pontificio Istituto Biblico, Roma, 2, revised edition, 1990. ISBN 88-7653-578-0.

[47] Wonneberger, Reinhard. "TEX in an industrial environment." In *Proceedings of the 4th European TEX Conference, September 11th-13th, 1989*, Anne Brüggemann-Klein [6]. forthcoming.

[48] Wonneberger, Reinhard. *Redaktion. Studien zur Textfortschreibung im Alten Testament, entwickelt am Beispiel der Samuelüberlieferung.* to appear.

[49] Wonneberger, Reinhard and Hans Peter Hecht. *Verheißung und Versprechen. Eine theologische und sprachanalytische Klärung.* Vandenhoeck & Ruprecht, Göttingen, 1 edition, 1986. ISBN 3-525-60367-3. Typesetting described in [42].

Index of proposals

`@institution` *and* `@person` *entry (30)*, 130
`@journal` *entry (27)*, 130
`@product` *entry (29)*, 130
`@review` *entry (28)*, 130

Academic research (56), 134

Backward references (40), 132
`.bib` *distribution (52)*, 133
`.bib` *printout (53)*, 133
BIBTEX *and title page (6)*, 127
BIBTEX *software titling (7)*, 127
`booktitle (36)`, 131

Change internal names (11), 127
Cite style (23), 129
Citefull style (10), 127
Collect style (24), 130
Comment character (48), 133
Composed symbols (19), 129

Date and time (43), 132

Entry separator (13), 128
Error message entry identification (45), 132
Explicit labels (2), 125
Extend internal commands (51), 133
Extended crossreferencing (41), 132

Field loops (33), 131

Language field (34), 131
Language flag (15), 128
Language independent standard styles (16), 129
Like option (26), 130
Living document (57), 134

Multi-pass system (39), 131
Multiple bibliographies (LATEX) (5), 126
Multiple labels (38), 131

Names and titles (21), 129
Nonsorting mode (47), 132
Number field (35), 131

Progammers guide (49), 133
Publication list (1), 124

RAK compatibility (42), 132
Review entries (31), 131

Selected bibliography generation (14), 128
Singularize `.bbl` *(9)*, 127
Software entries (54), 133
Sorting rules (20), 129
Source information (44), 132
Stacked entries (32), 131
Standard TEX support (55), 134
Standardize extended field names (37), 131
Structured source layout (12), 128
Style distribution (25), 130
Style language change (50), 133
Substyle concept (22), 129
Supply command (8), 127
Switchable fields (4), 126
Symbol convention (17), 129
Symbol conversion (18), 129

Warning message for unknown fields (46), 132
Worldwide convention(s) for labels (3), 126

Labelling Figures in TEX Documents

Alan Hoenig
Department of Mathematics, John Jay College/City University of New York
Mailing address: 17 Bay Avenue, Huntington, NY 11743 USA; (516) 385 - 0736.
Bitnet: `ajhjj@cunyvm`

Abstract

With practice, it becomes a relatively quick job to use METAFONT to generate figures that TEX can typeset in a document. How, though, may one use TEX to typeset labels that may be needed at various points in the figure? This presentation describes one approach. METAFONT is instructed to record coordinates in the font file which TEX may access and use as kerns for positioning label boxes.

Introduction

Many people perceive a problem in including graphics within TEX documents, but that problem no longer really exists. Thanks to the `\special` command in TEX, it's possible to include a wide variety of graphics within the printed document. The general strategy is as follows.

Ask TEX to leave the proper amount of white space in the document for the figure. Prepare a separate file by some other means which contains instructions for generating the figure. "Pull in" this separate file to the `dvi` file by means of the `\special` command. Finally, use a device driver which is alive to the possibility of external graphics files. Many such device drivers now exist.

There are, though, disadvantages to this approach. Primarily, the graphics file is not really a part of the TEX source file. This means that it will be impossible to preview this file unless you use a particularly smart previewer (one that also responds to `\special` commands and is intelligent enough to know how to display this material on the screen).

Need for Labels

Another disadvantage makes itself felt whenever a figure needs labels. How may they be included so that they are properly positioned? Some workers include the label as part of the graphic, but this leads to a visual inconsistency since the label font is most likely not any of the Computer Modern fonts that TEX is apt to be using. Ideally, we'd like TEX to typeset the labels, so they will be typeset as handsomely as the rest of the document.

In this article I'd like to describe one labelling approach that has worked well for me. I use METAFONT to generate the graphic as a font. While in this process, I decide which points in the figure need labels (although I need not decide on the labels themselves), and METAFONT incorporates the coordinates of these points in the font. Then, when TEX includes the figure, it may access these coordinates which it uses as amounts by which to move right or left and raise up or down a box containing the text of the label.

Other Work in this Area

Little work has apparently been done in this area. Rick Simpson was apparently the first to suggest (at least in print) that METAFONT be used as a drawing engine. His article [1990] is worth searching out.

One other effort in this field is the Metaplot macro package created by Patricia Wilcox [1989]. She showed how it was possible to enlist an interface between the user and METAFONT. In her case, it was a species of commonly used plotter. Various plotter commands were made to correspond to METAFONT commands, and she developed a package whereby it was possible to create sophisticated and pleasing graphics. (She is an individual blessed with considerably above average artistic skills.) These are amply illustrated in the article cited above. (Certain of my strategies were inspired by suggestions from Ms. Wilcox and Tom Rokicki, and I am happy to acknowledge their unwitting help.)

John Hobby [1989] is adopting a different approach. He is tailoring METAFONT so that it directly produces PostScript output rather than the generic font format METAFONT normally produces. Although it will be possible to generate Computer Modern fonts in PostScript format, this MetaPost program is primarily a vehicle for generating figures and logos. His program will use a different approach for incorporating labels within the picture.

The METAFONT Part

We begin by seeing how the METAFONT portion of this project works. I imagine each figure to be the "A" character of a special-purpose METAFONT font that contains no other characters. TEX contains no requirement that an "A" look anything like a *real* "A", and does not have any preconceived notions of the proper size of such a character, so no one complains if the "A" we ask METAFONT to draw looks nothing like a real "A" but like the figure we need in our document. Suppose the METAFONT file containing the figure-drawing instructions is `figfont.mf`. Then, assuming you've done all the METAFONT things properly, you would typeset this figure in your TEX document by declaring the font:

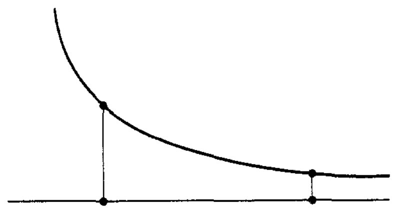
```
\font\figfont=figfont
```

and including the statement

```
{\figfont A}
```

where the figure is to appear. (Don't forget those enclosing braces!)

Here is a figure typical of one that might appear in a math text. (In fact, it appears somewhere in the first volume of Don Knuth's *Art of Computer Programming*.) The curve represents a portion of the graph of the function $f(x) = 1/x$. (Therefore, its METAFONT file is `1onx.mf`.)



It may be helpful to examine the METAFONT code to produce this drawing.

```
mode_setup;
u#=12pt#; nib#=.5pt#;
if (mode=smoke) or (mode=proof):
 u#:=.5pt#; nib#:=.1pt#; fi
define_pixels(u, nib);
beginchar("A", 12u#, 6u#, 0);
pickup pencircle scaled 2nib;
z1=(w/8,h); z2=(w/4, h/2);
z3=(w/2, h/4); z4=(w,h/8);
 % points on the curve
path p; p=z1..z2..z3..z4;
draw p; % draw the $1/x$ curve
z.x=point 2.6 of p; % another point
z5=(x2,0); z6=(x.x,0);
 % points on the $x$-axis
pickup pencircle scaled nib;
draw origin--(w,0); % bottom axis
draw z2--z5; draw z.x--z6;
 % vertical struts
pickup pencircle scaled 6nib;
drawdot z2; drawdot z.x;
drawdot z5; drawdot z6;
endchar; bye.
```

Most of these statements are reasonably self-explanatory, provided you are familiar with the rudiments of METAFONT syntax. The variable `nib#` governs the diameter of the pen we use for drawing, and `u#` is an ad hoc measure of length that's convenient for scaling the entire drawing. Of the other variables in this program, `w` and `h` are the width and height of the figure, and the quantities `z1`, `z2`, and so on (including `z.x`) refer to special points in the figure.

The lines

```
u#=12pt#; nib#=.5pt#;
if (mode=smoke) or (mode=proof):
 u#:=.5pt#; nib#:=.1pt#; fi
```

must be troubling to an ardent METAFONTer; they imply `u#` and `nib#` take on device dependent values, contrary to the METAFONT spirit. This device dependent code is dictated by limitations within my monitor. METAFONT is accustomed to working with pieces of type that are (roughly) ten points square. During on the screen development (when `mode` is either `smoke` or `proof`), METAFONT uses the entire screen to display the character. When the characteristic length of the character is several picas, METAFONT will only display small portions of the letter, unless `u#` is sufficiently small so that there will be room enough to display the character on the video monitor. The lines above attempt to implement this strategy. The characteristic dimensions `u#` and `nib#` have the values we'd like for any font making activity. However, during the development mode, they are much smaller to facilitate reviewing the work on the video screen.

METAFONT Talks to TEX

We could use TEX to provide an identifying caption for this picture, but it would be more helpful to include labels. Knuth provided labels at each of the four circle like dots — can we?

Hackers are used to referring to both TEX and METAFONT as full-featured programming languages, but METAFONT doesn't completely fill that role. For one thing, METAFONT has no ability to create files except for `log` and `gf` files. Therefore, we cannot write the coordinates of the label points to files for use by TEX.

The key idea is to recognize the usefulness of the `fontdimen` parameter. Normally, METAFONT uses them to record universal constants for a particular font, such as the width of a quad or the interword spacing. There appears to be no upper bound on the number of allowable `fontdimen` parameters, nor constraints on the uses to which they be put. I therefore felt free to use them to store the coordinates of each label point. Each point uses two fontdimen parameters — one for the x coordinate and one for the y coordinate.

The file `convert.mf` contains the definitions of macros such as `convertz_` which transform the coordinates into a useful format. Therefore, we need the statement

```
input convert;
```

right after the `mode_setup` command. Before the `endchar` command, we need an additional line

```
fontdimen10: convertz_(z2,z.x,z5,z6);
```

the macro `convertz_` converts the list of pairs to numbers which follow the proper `fontdimen` syntax.

The TEX Part

Once the figures have been properly "METAFONT ed," it's easy to include them in a TEX document, as we have already discussed. But it's also easy to apply labels. For that, we use a set of `\pointing` macros, similar to those in Appendix D of *The TEXbook*. If `\x` and `\y` are TEX dimen registers, it's easy to get their values from the proper `\fontdimen`:

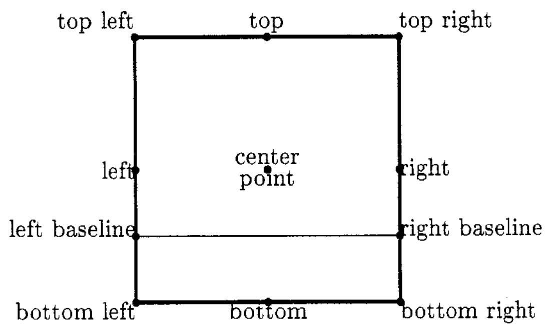
```
\advance\fontdimencount by 1
\x=\expandafter \the\fontdimen
 \the\fontdimencount \figfont
```

(and similarly for `\y`). Then, if `\box\labelbox` contains the text of the label, a construction like

```
\rlap{\kern\x \raise\y \box\labelbox}
```

will put the label in the proper position.

Actually, that's not quite true. There are eleven reference points associated with any label box, as the following diagram makes clear.



It's easy to create a set of macros which position a particular reference point at the point of the label. Thus, for example, the label "top right" was typeset using the command

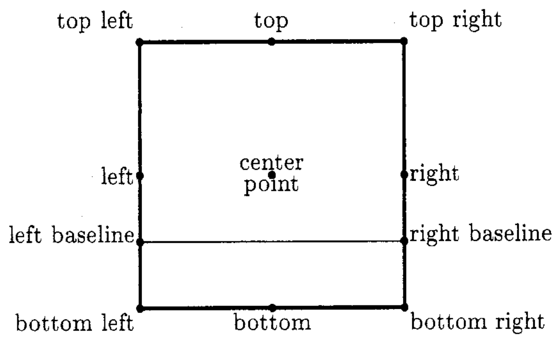
```
\blpoint{top right}
```

After all, the bottom left corner of the hbox containing "top right" must be positioned at the label point.

Still, this is not yet the full story. The reader who examines this figure closely will note that the labels seem somewhat cramped onto the label point. It may be necessary to leave some extra space. Commands like `\hskip2pt` or `\hskip-3pt` will move the label to the left or right, and it is easy to define macros `\up` and `\down` so that (for example)

```
\up3pt
```

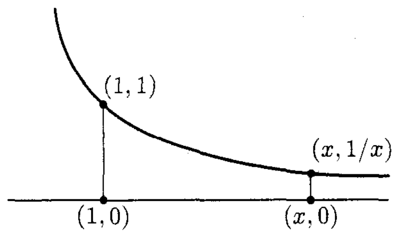
will help adjust vertical placement. We can use these considerations to produce a better reference point diagram



The following table lists all the pointing macros and their associated orientation. Some reference points have two macros. The user needing to position the top left point of a label box may use `\tlpoint` or `\ltpoint` so the question of remembering which of these two is correct does not arise.

Orientation	Macro Name(s)
top left	`\tlpoint`
	`\ltpoint`
top	`\tpoint`
top right	`trpoint`
	`\rtpoint`
right	`\rpoint`
bottom right	`\brpoint`
	`\rbpoint`
bottom	`\bpoint`
bottom left	`\blpoint`
	`\lbpoint`
left	`\lpoint`
left baseline	`\point`
	`\lBpoint`
	`\Blpoint`
right baseline	`\rBpoint`
	`\Brpoint`
central point	`\cpoint`

Using these tools, we can redo the Knuth's figure from *ACP*, this time with labels:

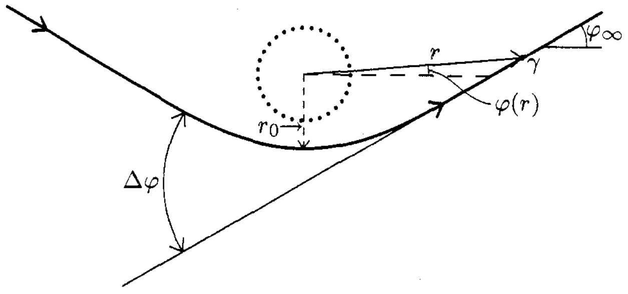


Here are the macro calls that I needed.

```
\font\figfont=1onx
\beginfig
\lbpoint{$(1,1)$}%
\lbpoint{\up2pt $(x, 1/x)$}%
\tpoint{\down 2pt $(1,0)$}%
\tpoint{\down 2pt$(x,0)$}%
\endfig
```

An Example

As a final example, here is a figure drawn from a general relativity text. The actual context or meaning of the figure is unimportant (it relates to the deflection of light due to relativistic effects), but it illustrate a number of interesting effects that are possible with METAFONT.



This example shows that METAFONT can create many important visual aids, such as arrow heads at arbitrary orientations, dotted lines, dashed lines, and so on.

Call to Arms!

One aim of this work is to strike fire into the heart of some ardent METAFONT artists. The TEX and METAFONT community could benefit from the creation of a macro package that stands in the same to METAFONT as does LATEX or AMS-TEX to TEX. Any takers?

Bibliography

Hobby, John D., "A METAFONT-like System with PostScript Output," *TUGboat* 10(4), pages 505–512, 1989.

Simpson, Richard O., "Nontraditional Uses of METAFONT." Pages 259–272 in *TEX: Applications, Uses, Methods*, Malcolm Clark, ed. (London: Ellis Horwood Ltd., 1990).

Wilcox, Patricia, "Metaplot: Machine-independent line graphics in TEX," *TUGboat*, 10(2), pp 179–187, 1989.

Typesetting Old German: Fraktur, Schwabacher, Gotisch and Initials

Yannis Haralambous
U. F. R. de Mathématiques,
Université de Lille-Flandres-Artois,
59655 Villeneuve d'Ascq, France.

Abstract

Typesetting in the old style, with the corresponding types, besides being an art, is also a real pleasure. METAFONT allows the creation of faithful copies of these types and TEX gives the possibility of using them in the most traditional manner. In this spirit, the necessary fonts and macros to typeset in the Old German types: Gotisch (also called Textur), Schwabacher and Fraktur are presented in this paper, together with an historical introduction to each of them. Also, a set of initials is described. Rules for typesetting in these types are given, together with extracts from the original sources.

This paper is dedicated to D. E. Knuth.

This article shows the first results of a longterm project on reconstructing old types and typesetting following the old rules, with TEX and METAFONT. The work presented in this paper has been done on a Mac SE/30 with OzTEX and MacMETAFONT.

General Introduction to the Project: What's the Use of Reconstructing Old Types?

Old types are beautiful. Until now, one could find either modernized copies of them (for decorative use) or facsimiles of historical books. With TEX and METAFONT, at last we have the possibility to approach these types in the manner — and with the care — of a *collectionneur*. Since there is no commercial scope, no compromise needs to be made in the creation of the fonts. And once the METAFONTing is done, we can bring the fonts back to life, by using them in new or old typesetting texts. TEX and METAFONT are strong enough to achieve a faithful reproduction of old works, and what's more, delicate enough to allow a personal tone and new ideas. Thanks to D. E. Knuth's work, typesetting becomes an interpretative art within the reach of everyone. You can believe me, it is the same pleasure to read (esp. typeset) Goethe's poems in Breitkopf's Fraktur as to hear (esp. play) Mozart's Sonatas on a Stein's Pianoforte.

Old German Types

Gotisch

Gutenberg chose the Bible as his first work for merely commercial reasons: only the churches and monasteries could afford to buy quantities of books. Consequently, the first types he created had to imitate manuscript characters, to be able to concurre with the beautiful manuscript Bibles produced by the monasteries themselves. This explains the fact that Gutenberg's font is so elaborate. A similar situation arose with Venetian Greek renaissance types, which had to imitate Alexandrinian and Byzantine Greek handwriting. Hundreds of ligatures were used.

Gutenberg's font had 288 characters: besides the 25 uppercase (there is no distinction between I and J) and 27 lowercase (there are two kinds of s), all the others are variant types, accented characters and ligatures.

The font `ygoth` presented here, is not an exact copy of Gutenberg's font. It merely follows Gutenberg's guidelines on lowercase characters and selects the uppercase ones from different 15^{th} century types. Please note that these uppercase characters are not suitable for "all capitals" typesetting. Here are the basic upper and lowercase characters:

A B C D E F G H
I K L M N O P Q R
S T U V W X Y Z
a b c d e f g h i j
k l m n o p q r
ſ s t u v w x y z

For all old German types, there is no distinction between I and J; also there are two kinds of s: the middle and initial "long s" and the final "short s:"

ſ s

In composite words, a short s is used when some component of the word ends on s:

Ausgang, but **Anſtand**.

Since it's almost impossible for a computer to know if some s is long or short, you have to do it manually; type `s:` for a short s, like in `Aus:gang` or `Alles:`.

The following ligatures are part of the font:

ae ba be bo ch ck ct
da de do ha he ho
ff fi fl ffi ffl ij ll
oe pa pe po pp qq gs
ſſ ſſi ſt tz va ve vu

Beside the ones shown above, there are variant forms

ſſ ſſi ſt

at positions *'052, '057, '075* of the font. Because of the many ligatures, there is no place left for special characters (I used only 128-character fonts). You'll have to switch to `CM` for #, $, %, &, *, +, =, etc. For the vowels a, e, o, u with Umlaut and for the ß, I followed Partl's [1988] convention: just type `"a`, `"e`, `"o`, `"u`, `"s` (ë is used in Flemish) to obtain:

ä ë ö ü ß.

The difference with Partl's approach is that in our case `"a`, `"e`, etc., are ligatures. Since ß historically comes from the ligature s+z (ß is called es-zet), by typing either `"s` or `sz`, you get the same output.

In Appendix A you can find a sample of the font; it is an extract of Luther's Bible (1534), in the original orthograph.

Schwabacher

The name comes from Schwabach, a little German town in the south of Nürnberg. According to Updike [1927],

in fifteenth century German gothic or black-letter fonts, a differentiation of type-faces began to show itself, as we have seen, in the last twenty years of the century, between types that were somewhat pointed and a rounder, more cursive gothic letter, with certain peculiarities — the closed a, looped b, d, h, and l, and a tailed f ans s. The first type was called "fraktur." The second was ultimately known as "schwabacher."

Schwabacher was in some extent the "bold-face" font, compared to the usual Fraktur. The font presented here is called `yswab`. It is based on 18th century types. Nevertheless, some characters (such as the "Hebrew-like" question mark ?) have been taken from a contemporary book: A. Wikenhauser [1948], *Das Evangelium nach Johannes*, where John's text is written in Schwabacher and comments in Fraktur. Here are the basic upper and lowercase characters:

A B C D E F G H I K L M N
O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n
o p q r s ſ t u v w x y z.

The following ligatures are included in the font:

ff, ſſ, ſſ, ſt, ß

For the vowels a, e, o, u with Umlaut, you have the choice between two forms: for the older one (a small "e" over the letter) you need to type a `*` + vowel combination, and for the newer one a `"` + vowel combination. So, by typing `*a`, `*e`, `*o`, `*u`, `"a`, `"e`, `"o`, `"u`, you get

aͤ eͤ oͤ uͤ ä ë ö ü

respectively.

Fraktur

The first Fraktur type was created by Johann Schönsperger in Augsburg to typeset the book of prayers of Kaiser Maximilian (1513). Some years later, Hieronymus Andreæ created a new Fraktur type, used by Dürer for the printing of his theoretical works. In the 17th century, Fraktur had a period of decline. It was only in the fall of the 18th century that some progressive typographers like G. I. Breitkopf and J. F. Unger gave Fraktur a new breath, by creating new fonts with the aesthetic

Gottlob Immanuel Breitkopf

standards of their time. Especially Unger's font seems to lay more in the 19th century spirit.

Gottlob Immanuel Breitkopf (1719 - 1794) lived in Leipzig. He travelled a lot, studied French, English and other foreign fonts and wrote an article (Breitkopf [1793]) on the situation of typographers and typography in Leipzig at the time. In 1754, he was first to use removable types to typeset music. His name is familiar to all musicians and friends of music, because of the famous Breitkopf & Härtel editions of complete works of Bach, Beethoven, etc.

After Breitkopf, the "official" version of Fraktur (newspapers and official documents) didn't evolve very much. In the 19th century, with all its social — and artistic — turbulence, many decorative Fraktur types were made, most of them monstrous (for example, see Knebel [1870]). A final renovative effort was made in the 1920s by artists like Walter Tiemann and others. Unfortunately, the destructive trend of uniformisation of Nazism didn't leave much place for æsthetic improvements or changes.

Texts like Warum deutſche Schrift? (Why german type?) by G. Barthel [1934], and Heraus aus der Schriftverelendung! (No more degenerate writing!) by T. Thormeyer [1934] (...die Rundungen haben nichts mit dem deutſchen Spannungsbedürfnis gemein=ſam. Das Schwelgen in abgerundeten Formen kann man andern Nationen überlaſſen...) show that Nazis tried to use Fraktur as a symbol of the German nation. But — an historical paradox — it was the Nazis themselves who abolished Fraktur in 1941*. In a recent edition of the Brockhaus, one can find the sentence *"Die nationalsozialist. Regierung ließ die Fraktur 1941 aus Zweckmäßigkeitsgründen von Amts wegen abschaffen. Ob sie damit eine Entscheidung traf, die ohnehin im Zuge der Entwicklung lag, ist schwer zu beurteilen..."* (it is hard to say if the Nazi decision of abolishing Fraktur was really in the sense of development...); there is a certain nostalgic in these words.

Today Fraktur is used mainly for decorative purposes (a nice counterexample is the dtv pocket edition of Mozart's correspondence: his letters are in Fraktur and the comments in Antiqua). Also, there are methods for the old German handwriting (Süterlin) which also includes Fraktur (for example,Wir leſen deutſche Schrift, bei A. Kiewel et al [1989]).

Let's return now to TEX: the font `yfrak` which I propose is in the old Breitkopf style. Here are the basic upper and lowercase characters:

A B C D E F G H I K L M N
O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n
o p q r s ſ t u v w x y z.

It contains the same ligatures and Umlauts as `yswab`. The symbols ꝛc (which means "etc") and J (an attempt to differentiate I and J) are in font positions *'044* and *'100* respectively. You can find a sample of the font in Appendix B. It is the begining of the second part of Carl Philipp Emanuel Bach's treatise on the true art of playing the keyboard (which meant the harpsichord and/or clavichord) "Verſuch über die wahre Art das Clavier zu ſpielen" [1762].

* There seems to have been some secrecy around this decision of the Nazis. The only data I could find is a short and cryptical reference in the 1941 DIN-booklet on typographic standards: "Bekanntgebung II EM 8408/41 vom 26 Juli 1941 des Reichswirtschaftsministers an den Deutschen Normenauschluß". I would be very obliged if some reader could provide me with more information.

Initials

The chancery initials, which you can see in Appendix B and C, are a revival of baroque designs. This makes them suitable for both old and new texts. They form the font `yinit`. You have the choice of creating characters with depth zero, or characters with height equal to `cap_height` of `cmr10` (with the corresponding magnification) and the biggest part of the character under the baseline. For this, there is a boolean parameter `zero_depth` in the `yinit.mf` parameter file. To typeset the initial D of Appendix B, I used the macro `\yinitial{D}` as follows (with `zero_depth:=false`)

```
\def\yinitial#1
{\hangindent=2.54cm
\hangafter=-4
\hskip-3.24cm
\lower-2.7mm
\hbox{\yinit #1}
\hskip1.5mm}
```

Of course, all these parameters will need some adjustment, according to the interline skip and the textfont you are using. Note also that `\par` stops the execution of `\hangafter`. It wouldbe better to use `\hfill\break\indent` instead.

Typesetting Rules

In the following text, taken from the Duden (Mülsing and Schmidt [1919]) many fine points of typesetting in Fraktur are explained. The essential points are the following: 1) don't use ligatures in Latin Antiqua words, use them in French Antiqua and in French Fraktur; 2) in a composite word, do not use ligatures between adjacent letters of two components 3) the antiqua ß is to be used in German words and names regardless of the language; 4) the Latin "etc" is to be translated as uſw., and its older form ꝛc should not be used anymore; 5) concerning foreign words in German, use Fraktur when the word has been "germanized", otherwise use antiqua; 6) the hyphen should always be used in Fraktur, except when it appears between two antiqua words; 7) in 1879, Daniel Sanders proposed 𝔍 as an alternative to ℑ for the letter J; it would be nice if authorities recognized it.

Einzelvorſchriften für den Schriftſatz

In dieſem Abſchnitte ſtellen wir einige Einzelvorſchriften zuſammen, deren allgemeine Befolgung für die Einheitlichkeit bei der Herſtellung von Druckſachen ſehr wünſchenswert wäre.

Ligaturen Æ, æ, Œ, œ ſtatt Ae, ae, Oe, oe. In lateiniſchen Wörtern ſind die Ligaturen nicht anzuwenden, z. B. Caelius mons, Asa foetida. In franzöſiſchen Wörtern, die im deutſchen Satz verſtreut vorkommen, muß, wie im franzöſiſchen Satz überhaupt, ſtets Œ und œ geſetzt werden, z. B. Œuvres, sœur. Selbſt bei Frakturſatz darf auf das kleine œ nicht verzichtet werden, z. B. Horsd'œuvre.

Sonſtige Ligaturen. In Wortverſchmelzungen wie Schiffahrt, Schnelläufer, alliebend, d. h. alſo in Wörtern, die von drei gleichen Mitlauten einen ausgeſtoßen haben, iſt die Ligatur anzuwenden, wenn ſie in der betreffenden Schriftgattung vorhanden iſt. Die Ligatur iſt ferner überall da anzuwenden, wo ſie die ſprachliche Richtigkeit nicht ſtört, z. B. benutzen, abflauen, Billard, nicht aber in einfachen Zuſammenſetzungen wie entzwei, Kaufleute, vielleicht.

Der Buchſtabe ß in fremdſprachichem Satz. Wenn aus einem Deutſchen Namen, in dem ß vorkommt, durch Anfügung einer Lateiniſchen Endung ein Lateiniſches Wort gebildet wird, ſo bleibt das ß erhalten, es erſcheint alſo als ß (in Antiqua). So wird z. B. aus Weißenburg: Weißenburgensis (der Codex Weißenburgensis). Ebenſo wird ß geſetzt, wenn deutſche Eigennamen mit ß in fremdſprachlichem Satz erſcheinen, z. B. : Monsieur Aßmann a été à Paris. Ho trovato il Signor Große a Venezia.

uſw. – ꝛc – etc. Im deutſchen Satze iſt „und ſo weiter" der amtlichen Vorſchrift gemäß durch uſw. abzukürzen, und zwar ſowohl in Fraktur wie in Antiqua. Die Form ꝛc, die ſich innerhalb der Lautſchrift wie eine Hieroglyphe, wie ein Vertreter der Zeichenſchrift, aufnimmt, iſt veraltet und nicht mehr anzuwenden.

Die Form etc darf nur im Antiquaſatz angewandt werden, wird aber beſſer durch usw. erſetzt. Für lateiniſchen Satz, alſo innerhalb lateiniſchen Textes, iſt etc. ſelbſtverſtändlich. Ferner ſei erwähnt, daß die Franzoſen und Engländer &c., die Italiener ecc. und die Spanier etc. verwenden, und zwar ſetzen alle ſtets einen Beiſtrich vor dieſe Abkürzungen, was im Deutſchen nicht üblich iſt.

Anwendung der Antiqua im Frakturſatz. Um dem biſherigen Schwanken in der Wahl zwiſchen Antiqua und Fraktur ein Ende zu machen, empfiehlt es ſich folgende Grundſätze zu beobachten:

1. Alle Fremdwörter romaniſchen Urſprungs, die nicht durch Annahme deutſcher Biegung oder deutſcher Lautbezeichnung als eingedeutſcht erſcheinen, ſetze man aus Antiqua, z. B. en avant, en arrière, en vogue, in praxi, in petto; a conto, dolce far niente; ferner Verbindungen wie Agent provocateur, Tempi passati, Lapsus linguae, Agnus Dei. Auch alle italieniſchen techniſchen

Ausdrücke aus der Tonkunſt, wie andante, adagio, moderato, vivace, ſetze man aus Antiqua. Die der lateiniſchen Sprache entſtammenden Bezeichnungen Dur und Moll ſind als eingedeutſchte Hauptwörter aufzufallen und daher groß zu ſetzen, z. B. C⸗Dur.

2. Wenn ein Fremdwort deutſche Lautbezeichnung oder deutſche Biegung annimmt oder mit einem deutſchen Worte zuſammengeſetzt wird, ſo ſetze man es ais Fraktur, z. B. adagio, aber: das Adagio, die Adagios; a conto, aber: die Akontozahlung; dolce far niente, aber: das Dolcefarniente.

Anwendung des Bindeſtrichs in Frakturſatz, der mit Antiqua vermiſcht iſt. Wenn in Frakturſatz bei Wortzuſammenſetzungen der eine Teil der Zuſammenſetzung aus Antiqua geſetzt werden muß, ſo ſind etwa vorkommende Bindeſtriche aus der Textſchrift, alſo aus Fraktur, zu ſetzen, z. B. CGS⸗Maßſyſtem. Eine Ausnahme wird nur dann gemacht, wenn der mit dem Bindeſtrich ſchließende erſte (Antiqua⸗) Beſtandteil an das Ende einer Zeile oder in Klammern zu ſtehen kommt; in dieſem Falle iſt der Bindeſtrich aus Antiqua zu ſetzen. In beſonderen Fällen kann auch eine Vermiſchung von Fraktur⸗ und Antiquabindeſtrichen ſtattfinden, z. B. Hoftheater⸗Corps-de-ballet; denn innerhalb des aus Antiqua geſetzten Wortes müſſen auch die Bindeſtriche aus Antiqua geſetzt werden.

J (Selbſtlaut) und J (Mitlaut) in der lateiniſchen Druckſchrift. In der lateiniſchen Druckſchrift wird zwiſchen dem Selbſtlaut und dem Mitlaut J genau unterſchieden, und zwar ſteht I ausſchließlich für den Selbſtlaut, J ausſchließlich für den Mitlaut. Dieſe Unterſcheidung machen alle neueren Sprachen. Daß die deutſche Druckſchrift einen Unterſchied zwiſchen J (Selbſtlaut) und J (Mitlaut) nicht kennt, iſt ein großer Mangel. Dieſen Mangel zu beſeitigen verſuchte ſchon 1879 Daniel Sanders, indem er für den Mitlaut das Zeichen J empfahl. Dieſes Zeichen iſt heute nur vereinzelt in Drucken zu finden, hat ſich alſo nicht allgemein eingebürgert und iſt auch nicht amtlich anerkannt worden. Es wäre ſehr zu wünſchen, daß auch in deutſcher Schrift ein Unterſchied zwiſchen J (Selbſtlaut) und J (Mitlaut) geſchaffen und von der zuſtändigen Behörde anerkannt würde, und zwar um ſo mehr, als er bei den kleinen Buchſtaben ſowohl in deutſcher (i, j) wie in lateiniſcher (i, j) Schrift bereits ſeit langem beſteht.

Availability

Following a tradition of my friend Klaus Thull, these fonts are in the public domain. They should be available at the Aston and Heidelberg archives. Also, you can obtain them at my address. The status of this software is postcard-ware: each satisfied user could send me a nice local postcard for my collection.

References

Bach, Carl Philipp Emanuel. *Versuch über die wahre Art das Clavier zu spielen. Zweyter Theil, in welchem die Lehre von dem Accompagnement und der freyen Fantasie abgehandelt wird.* Berlin: G. L. Winter, 1762.

Barthel, Gustav. "Warum deutsche Schrift?" *Schrift und Schreiben* 4, pages 98 – 130, 1934.

Breitkopf, Johann Gottlob Immanuel. "Ueber Buchdruckerey und Buchhandel in Leipzig." *Journal für Fabrik, Manufaktur und Handlung* 5, pages 1 – 57, 1793.

Faulmann, Carl. *Das Buch der Schrift, enthaltend die Schriftzeichen und Alphabete aller Zeiten und aller Völker des Erdkreises.* Wien: Druck und Verlag der kaiserlich-königlichen Hof- und Staatsdruckerei, 1880.

Glaister, Geoffrey Ashall. *Glaister's Glossary of the Book.* London: 1960.

Kiewel, Albert, Eberhard Dietrich, Inghild Stölting, and Heinold Wachtendorf. *Wir lesen deutsche Schrift.* Hannover: Kallmeyer'sche Verlagsbuchhandlung, 1989.

Knebel, P. *Sammlung der gebräuchlisten Schriftgattungen.* Landshut: Verlag der Jos. Thomann'schen Buchhandlung, 1870.

Mülsing, Ernst and Schmidt Alfred. *Duden, Rechtschreibung der deutschen Sprache und der Fremdwörter.* Leipzig und Wien: Bibliographisches Institut, 1919.

Partl, Hubert. "German TEX." *TUGboat* 9 (1), pages 70 – 72.

Stiebner, Erhardt, Helmut Huber and Heribert Zahn. *Schriften + Zeichen.* München: Bruckmann, 1987.

Thormeyer, Traugott. "Heraus aus der Schriftverelendung!" *Schrift und Schreiben* 4, pages 131 – 136, 1934.

Updike, D. B. *Printing Types.* 1927.

Walther, Karl Klaus. *Lexikon der Buchkunst und Bibliophilie.* Leipzig: Bibliographisches Institut, 1987.

Wikenhauser, Alfred. *Das Evangelium nach Johannes.* Regensburg: Friedrich Pustet, 1948.

Appendix A
Das acht Capitel

Und die menner von Ephraim ſprachen zu yhm, Warumb haſtu uns das gethan; haſt du uns nicht rieffeſt; da du ynn ſtreyt zogeſt widder die Midianiter? und zanckten ſich mit yhm hefftiglich. Er aber ſprach zu yhnen. was hab ich itzt getan das gleich ſey? Jſt nicht eyn rebe Ephraim beſſer. denn die gantze weynernd Abi Eſer? Gott hat die fürſten der Midianiter Oreb und Seb ynn ewr hend gegeben. wie hettkund ich das yhr than hatt. Da er ſolchs redet; lies yhr geyſt von yhm abe Da nü Gideon an den Jordan kam; gienger hynuber mit den drey hunder man; die bey yhm waren und waren müde und lagten nach und er ſprach zu den leutten zu Sücoth; Lieber gebt dem Volck das unter myr iſt ettlich brod; denn ſie ſind müde. das ich nachiage den konigen der Midianiter Sebah und Zalmüna Aber die furſten zu Sucoth ſprachen, Sind diehend Sebah und Zalmüna ſchon ynn deynen henden; das wyr deyner ſchar ſollen brod geben? Gideon ſprach Wolan wenn der herr Sebah und Zalmuna ynn meyne hand gibt will ich ewr fleyſch mit dornen aus der wuſten und mit hecken zu dreſſchen. Und er zoch vondannen hynauffgen Pnüel und redet auch alſo zu yhnen. Und die leutt zu Pnüel anttwortten yhm. gleich wie die zu Sucoth. Und er ſprach auch zu den leutten zu Pnuel. Kom ich mit frieden wider. ſo will ich dißen türn zu brechen; Sebah aber und Zalmüna warren zü Karkar. Und yhr heer mit yhnen bey funfftzehen tauſent, die alle uberblieben waren vom gantzen heer. der kinder vom morgen. Denn hundert und zwantzig tauſent waren gefallen; die ſchwerd außihen kunden. Und Gideon zoch hynauff auf den ſtraſſen da man ynn hutten wohet gegen morgen; gen Nobah und Jagbeha und ſchlug das heer. Denn das heer war ſicher. Und Sebah und Zalmuna flohen. aber er jaget yhn nach; und fieng die zween Konige der Midianiter Sebah und Zalmuna und zur ſchreckt das ganze heer.

An extract from Luther's Bible

Appendix B

Einleitung

§. 1.

Die **Orgel**, der **Flügel**, das **Fortepiano** und das **Clavicord** ſind die gebräuchliſten Clavierinſtrumente zum Accompagnement.

§. 2. Es iſt Schade, daß die ſchöne Erfindung deſ **Holfeldiſchen Bogenclaviers** noch nicht gemeinnützig geworden iſt; man kann dahero deſſen beſondere Vorzüge hierinnen noch nicht genau beſtimmen. Es iſt gewiß zu glauben, daß es ſich auch bey der Begleitung gut ausnehmen werde.

§. 3. Die **Orgel** iſt bey Kirchenſachen, wegen der Fugen, ſtarken Chöre, und überhaupt der Bindung wegen unentbehrlich. Sie befördert die Pracht und erhält die Ordnung.

§. 4. So bald aber in der Kirche Recitative und Arien, beſonders ſolche, wo die Mittelſtimmen der Singſtimme, durch ein ſimpel Accompagnement alle Freyheit zum Verändern laſſen, mit vorkommen, ſo muß ein **Flügel** dabey ſeyn. Man hört leyder mehr als zu oft, wie kahl in dieſem Falle die Ausführung ohne Begleitung des Flügels ausfällt.

§. 5. Dieſes leztere Inſtrument iſt auſſerdem beym Theater und in der Cammer wegen ſolcher Arien und Recitative unentbehrlich.

§. 6. Das **Fortepiano** und das **Clavicord** unterſtützen am beſten eine Ausführung, wo die gröſten Feinigkeiten des Geſchmacks vorkommen. Nur wollen gewiſſe Sänger lieber mit dem **Clavicord** oder **Flügel**, als mit jenem Inſtrumente, accompagnirt ſeyn.

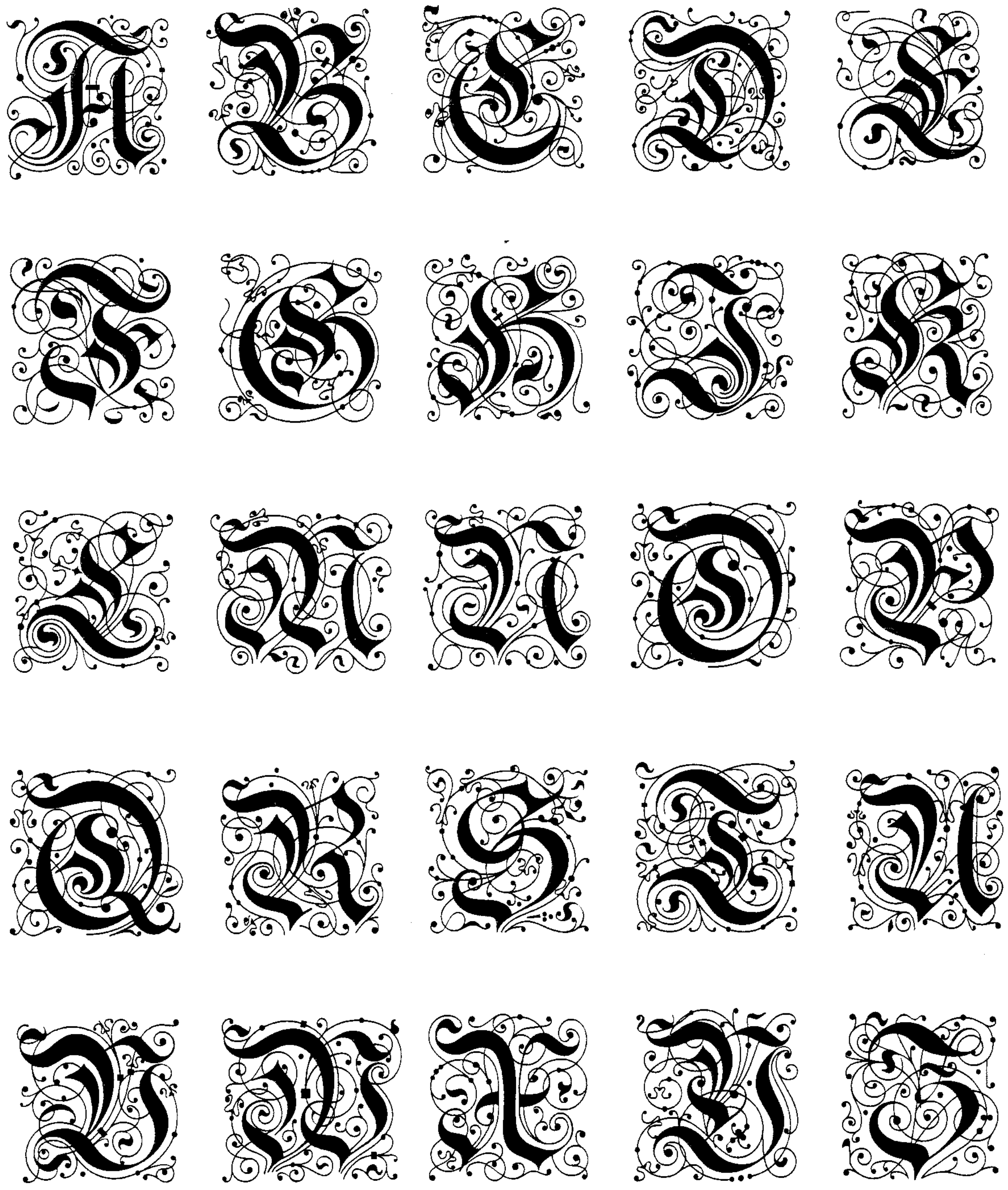
§. 7. Man kann alſo ohne Begleitung eines Clavierinſtruments kein Stück aufführen. Auch bey den ſtärkſten Muſiken, in Opern, ſo gar unter freyem Himmel, wo man gewiß glauben ſolte, nicht das geringſte vom Flügel zu hören, vermißt man ihn, wenn er wegbleibt. Hört man in der Höhe zu, ſo kann man jeden Ton deſſelben deutlich vernehmen. Ich ſpreche aus der Erfahrung und jedermann kann es verſuchen.

§. 8. Einige laſſen ſich beym Solo mit der Bratſche oder gar mit der Violine ohne Clavier begleiten. Wenn dieſes aus Noth, wegen Mangel an **guten** Clavieriſten, geſchiehet, ſo muß man ſie entſchuldigen; ſonſt aber gehen bey dieſer Art von Ausführung viele Ungleichheiten vor. Aus dem Solo wird ein Duett, wenn der Baß gut gearbeitet iſt; iſt er ſchlecht,

wie

An extract from C. P. E. Bach's Treatise on the true Art of playing the Keyboard

Appendix C



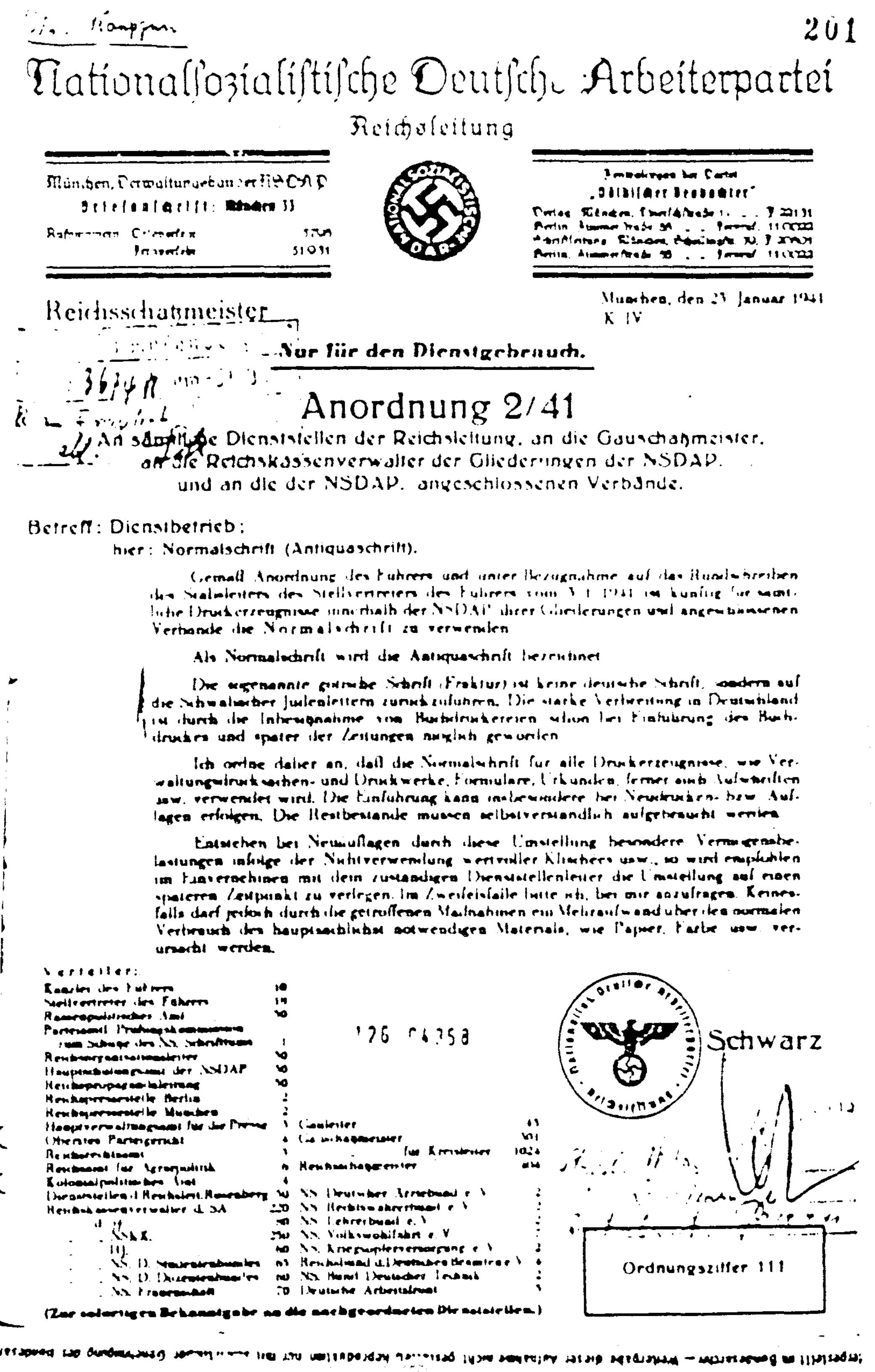
The font yinit scaled 1728

Appendix D
Late Breaking News

I would like to thank Frau Franziska Kaiser for finding me (after many attempts) a copy of the official order of the German Nazist Party, abolishing Fraktur and Schwabacher from all printed items. The argumentation given is typical of the Nazist Party (antisemitic and illogical) :

"*...It is ordered that from now on only the normal type is to be used for all printed documents. As normal type, the antiqua type is meant. The so-called gothic type (Fraktur) is not a german type but goes back to the schwabacher jew-letters. This type has been strongly used in Germany because Jews owned the printing works already since typography was introduced, and later on the newspapers...*"

Rather a sad way for such a beautiful type to die, especially if one considers all the masterpieces of the first centuries of typography! Of course the real reasons of this event —which occured exactly 50 years ago— are not the ones mentionned in the document... perhaps historians will find them some day. My personal opinion is that even if an analogous reform would be applied sooner or later it should have a completely different argumentation (for example the readability of Fraktur in comparison to Antiqua) and should leave printers the freedom of choice: uniformization by force always brings flatness and sterility.



201

Nationalſozialiſtiſche Deutſche Arbeiterpartei

Reichsleitung

München, Verwaltungsbau der NSDAP

„Völkiſcher Beobachter"

Reichsschatzmeister

München, den 23. Januar 1941
K IV

Nur für den Dienstgebrauch.

36144

Anordnung 2/41

An sämtliche Dienststellen der Reichsleitung, an die Gauschatzmeister,
an die Reichskassenverwalter der Gliederungen der NSDAP.
und an die der NSDAP. angeschlossenen Verbände.

Betreff: Dienstbetrieb;
hier: Normalschrift (Antiquaschrift).

Gemäß Anordnung des Führers und unter Bezugnahme auf das Rundschreiben des Stabsleiters des Stellvertreters des Führers vom 3.1.1941 ist künftig für sämtliche Druckerzeugnisse innerhalb der NSDAP, ihrer Gliederungen und angeschlossenen Verbände die Normalschrift zu verwenden.

Als Normalschrift wird die Antiquaschrift bezeichnet.

Die sogenannte gotische Schrift (Fraktur) ist keine deutsche Schrift, sondern auf die Schwabacher Judenlettern zurückzuführen. Die starke Verbreitung in Deutschland ist durch die Inbesitznahme von Buchdruckereien schon bei Einführung des Buchdruckes und später der Zeitungen möglich geworden.

Ich ordne daher an, daß die Normalschrift für alle Druckerzeugnisse, wie Verwaltungsdrucksachen- und Druckwerke, Formulare, Urkunden, ferner auch Aufschriften usw. verwendet wird. Die Einführung kann insbesondere bei Neudrucken- bzw. Auflagen erfolgen. Die Restbestände müssen selbstverständlich aufgebraucht werden.

Entstehen bei Neuauflagen durch diese Umstellung besondere Vermögensbelastungen infolge der Nichtverwendung wertvoller Klischees usw., so wird empfohlen im Einvernehmen mit dem zuständigen Dienststellenleiter die Umstellung auf einen späteren Zeitpunkt zu verlegen. Im Zweifelsfalle bitte ich, bei mir anzufragen. Keinesfalls darf jedoch durch die getroffenen Maßnahmen ein Mehraufwand über den normalen Verbrauch des hauptsächlichst notwendigen Materials, wie Papier, Farbe usw. verursacht werden.

Verteiler:

Kanzlei des Führers	10		
Stellvertreter des Führers	19		
Rassenpolitisches Amt	50		
Parteiamtl. Prüfungskommission zum Schutze des NS. Schrifttums	1		
Reichsorganisationsleiter	50		
Hauptschulungsamt der NSDAP	50		
Reichspropagandaleitung	50		
Reichspressestelle Berlin	2		
Reichspressestelle München	2		
Hauptverwaltungsamt für die Presse	3	Gauleiter	43
Oberstes Parteigericht	4	Gauschatzmeister	301
Reichsrechtsamt	3	für Kreisleiter	1024
Reichsamt für Agrarpolitik	6	[illegible]	404
Kolonialpolitisches Amt	4		
Dienststellen d. Reichsleit. Rosenberg	50	NS. Deutscher Ärztebund e.V.	2
Reichskassenverwalter d. SA	220	NS. Rechtswahrerbund e.V.	2
d. SS	50	NS. Lehrerbund e.V.	2
d. NSKK.	250	NS. Volkswohlfahrt e.V.	2
d. HJ.	60	NS. Kriegsopferversorgung e.V.	2
d. NS. D. Studentenbundes	65	Reichsbund d. Deutschen Beamten e.V.	6
d. NS. D. Dozentenbundes	60	NS. Bund Deutscher Technik	2
d. NS. Frauenschaft	70	Deutsche Arbeitsfront	3

(Zur sofortigen Bekanntgabe an die nachgeordneten Dienststellen.)

126 04358

Schwarz

Ordnungsziffer 111

Order of the Nazist Party
Forbidding the use of Fraktur

The Irish Alphabet

Mícheál Ó Searcóid
Roinn na Matamaitice, Coláiste na hOllscoile, Baile Átha Cliath, Éire
Bitnet: `searcoid@irlearn`

Abstract

The origin, history and present-day usage of the Irish typeface. The brighter future due to TeX and METAFONT. Irish types in a range of sizes and weights unimaginable a generation ago are now being designed with METAFONT and will shortly be ready for use.

In a recent article in the *TUGboat*, Yannis Haralambous made the statement that the Irish language has its own, most beautiful alphabet. I do not disagree with him as regards the beauty. His assertion does, however, require some qualification. Firstly, the use of this alphabet for ordinary printing purposes has not been widespread during the past twenty years. Secondly, what we call the Irish alphabet is no more than a simple variant of the roman one; certainly it has its own distinctive features, but these are not so obtrusive — in the later typefaces, anyway — as to hinder speedy recognition of the letter forms by anyone familiar with standard roman types, or to inhibit comfortable reading by those who know the language.

The Roman Empire never extended as far as the shores of Ireland. Nonetheless, there was considerable traffic by Irishmen across the Irish sea in the days of Roman Britain. The very name by which we Irish call ourselves, Gaeil, and the name of our language, Gaeilge, are in origin Welsh names. Indeed there were Irish settlements and strongholds in Wales during most of the period of the Empire, and it would have been surprising if the Irish had not been affected by some aspects of the roman culture. One of the most important concepts transmitted by that culture was that of writing. The Romans wrote not only for their own benefit, but also for posterity. They erected stone monuments and inscribed on them. The ancient Irish copied the practice. The Irish however preferred in most cases to write their own language in their own alphabet. Their language has survived the mighty Latin of the empire as an everyday tongue, and their is some evidence that their alphabet, known as Ogham, was never completely forgotten.

Each character of the Ogham alphabet consists of strokes, or sometimes dots, numbering one to five, written to the left or the right of a vertical baseline, or across that baseline. That the Ogham alphabet existed before it was put to use in the medium of stone can hardly be doubted. It may have been a finger alphabet used for sign language; its ultimate origin may even be in the use of fingers for counting. Whatever its early history, it was used during the fourth, fifth and sixth centuries A.D. by Irishmen in south-western Britain and in the south of Ireland to make inscriptions on stone. Many of the Ogham stones have been uprooted from their original locations; several are preserved in the cloisters around the quadrangle of University College, Cork.

The Ogham alphabet in its written form is distinctly Irish; but as a written alphabet it is rather clumsy. It has more in common with the finger signs of the american baseball player than it has with fine literature. In the fifth century Christianity came to Ireland, and with it came book-writing. Ogham was not very suitable for writing on vellum or parchment, nor indeed for writing very much more than the terse memorials for which it was used. It is not surprising, therefore, that by the end of the sixth century it had generally fallen into disuse.

By that time, the seeds of Christianity in Ireland had begun to bear fruit. Irish missionaries were already spreading across Europe. They not only took with them the Christian faith; they had also become custodians of a culture which was centred around the written word. For a thousand years their mission continued. It extended from Iceland to the Mediterranean and from Scotland as far east as Kiev. Irishmen were in the forefront of the Christianization of half of Europe, and Europeans in turn travelled to the great monastic universities of Ireland to receive education.

The writings of Irishmen, mostly in Latin but sometimes in Gaelic, are to be found scattered across the vast territory of their missionary activity. These men brought their own particular genius to the way in which they used and ornamented the roman alphabet. The hand of the Irish monk is distinctive; latterly it developed, as did other national hands, alongside of, but separately from, the Carolingian. That it was a neo-Carolingian hand which eventually dominated in Europe is a matter of history; but that the Irish hand still flourished at the advent of printing was important for the future of Irish typography.

At home too, Irishmen were at their books. They committed to writing the early Irish legal system and their medical knowledge, together with some of the folklore of the Irish people. They copied religious works. St Columba, usually known in Ireland as Colm Cille, who, in the sixth century, established the famous monastery on the island of Iona and became the patron of Scottish Christianity, once borrowed a book from Finnian Droma Fhinn and spent days and nights making a copy of it. When Finnian learnt of this, he was outraged and staked his claim to the copy. The case was brought before the King, Diarmaid Mac Cearbhaill, whose judgement that "every calf belongs to its mother, and every copy of a book to the owner of the book" constituted Ireland's first copyright law; its harshness no doubt evokes a variety of responses from today's listeners.

It is to the English reformation, and to the Privy Council of the English Queen Elizabeth I, that we owe our gratitude for establishing Irish letter forms as those proper to the printing of the Irish language. In the hope that "God would, in mercy, raise up some to translate the New Testament into their mother tongue" that Queen ordered that a special typeface be designed, cut and sent to Dublin for use in publishing religious material. The typeface was to be based on Irish written letter forms. She also ordered that an Irish grammar be prepared in order that she herself might learn the language.

Matthew Parker, the man to whom the idiomatic expression 'Nosy Parker' was first applied, had been Elizabeth's Archbishop of Canterbury since the first year of her reign. Anglo-Saxon fonts had already been cut for him by John Day, and it is probable that it was the archbishop himself who determined that a new typeface should be designed to accommodate the Irish language. A brief glance at the hand of the *Book of Kells* and that of the *Lindisfarne Gospels* is sufficient to convince us of the similarity between Irish and Anglo-Saxon manuscript writing*; since the Anglo-Saxons had been taught to write by the Irish, the likeness is not surprising. The wheel turned full cycle with the advent of printing, and it seems that it was Day's Anglo-Saxon typeface which became the inspiration, though not the model, for the new Gaelic one.

Elizabeth's interest in the Irish language may seem strange. Her father, Henry VIII, had issued a statute demanding that all the English living in Ireland learn the English language within a year. The statute, which was translated into Irish for their benefit, had had no noticeable effect. He had also ordered that they shave off their moustaches and wear their beards after the English fashion; one wonders if they were willing to comply even with this much simpler command. Now, Elizabeth was promoting the language! Certainly it was not as remote from her as it had been from her father, for there were Irish nobles at her court who spoke it. She was probably motivated primarily by the desire to save the souls of the Irish people from popery. In England, *The Book of Common Prayer* had replaced the Latin liturgy of the Catholic church by one which the ordinary people could understand. Elizabeth was simply doing the same thing in Ireland.

The new type was cut in London and was in use in Dublin from 1571. A religious poem by Philip Mhac Cuinn Chrosaigh, dated for that year and now preserved in Cambridge University Library, is the oldest surviving document printed with it. The typeface is strong and well-balanced, and the overall effect is extremely neat. However, on closer inspection, it displays better than any of the other early Irish typefaces how slight the differences between the Irish letter forms and the more standard roman ones really are. Some ligatures were cut, but it is doubtful if more than half of the letters were new designs; several clearly belong to a roman typeface, and the 'a' belongs to an italic one. Indeed, it should have been possible to print Irish language material without any new type design, and we are very fortunate that Queen Elizabeth's adminisration ordained it otherwise.

Thus it was English Protestantism that first made use of printing in the battle for the souls of the Gaelic nation; but its adversaries on the continent eventually became stronger and more determined in their continued use of it. Some of Ireland's greatest scholars were Catholic priests

* Examples of these hands and of some of the Irish typefaces are shown on separate pages.

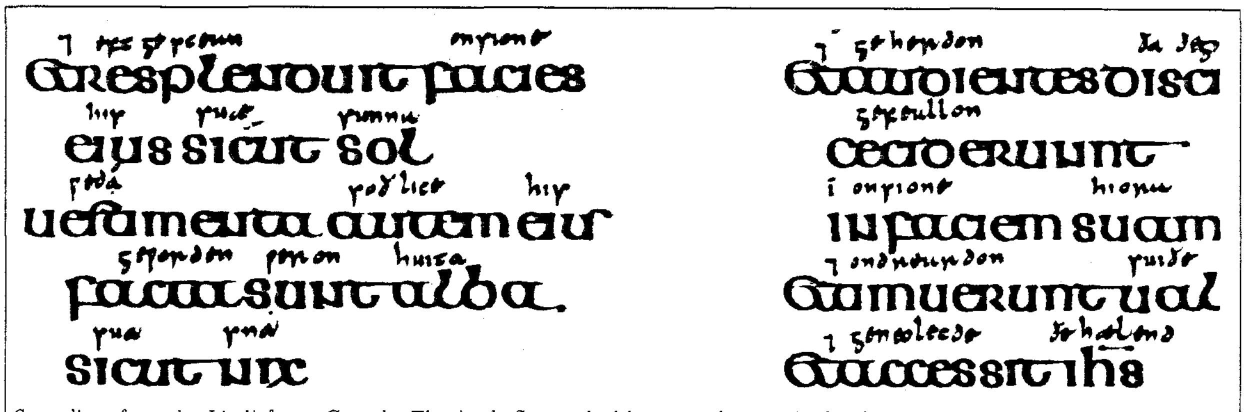
living in exile. In about 1611, the Irish Franciscans in Belgium designed a thoroughly Irish type, based on Irish manuscript letters, which for over a century kept a steady stream of Catholic religious material flowing from Louvain to Ireland. Around 1675, a new Irish type was cut in Rome for Sacra Congregatio de Propaganda Fide and the Rome press became engaged in a like task. In the eighteenth century, a new Gaelic type was cut in Paris and that city too became a centre for the publication of Irish language material. It is curious to read in the preface to the first English–Irish dictionary, that published at Paris in 1732, an apology for any errors that might have resulted from the printers' ignorance of both languages. In 1804, the Rome type of 1675 and a much later one also from Rome were used by J.J. Marcel of the Imprimerie Nationale in Paris. They had been carried off from Rome as part of the booty of Napoleon and deposited by him at the Imprimerie. Had it not been for Napoleon they may well have been lost, as are so many of the other early types.

In time, Queen Elizabeth's prayer that some Irishman might translate the New Testament into his mother tongue was answered. That and *The Book of Common Prayer* were translated by Nicholas Breatnach, Bishop of Ossory, and by Uilliam Ó Domhnuill, Archbishop of Tuam. But it was left to an Englishman, William Bedell, to embark on the task of translating the Old Testament. Bedell arrived in Ireland in 1627 at the age of 56 to take up the post of Provost of Trinity College, Dublin. He set to work at once to learn the language, and he insisted that students who proposed to be clergymen should do likewise. He engaged competent natives to work on the translation, and he himself was responsible for the final form of words, which was approved on a chapter by chapter basis only after careful comparison with the Hebrew and with a polyglot Bible.

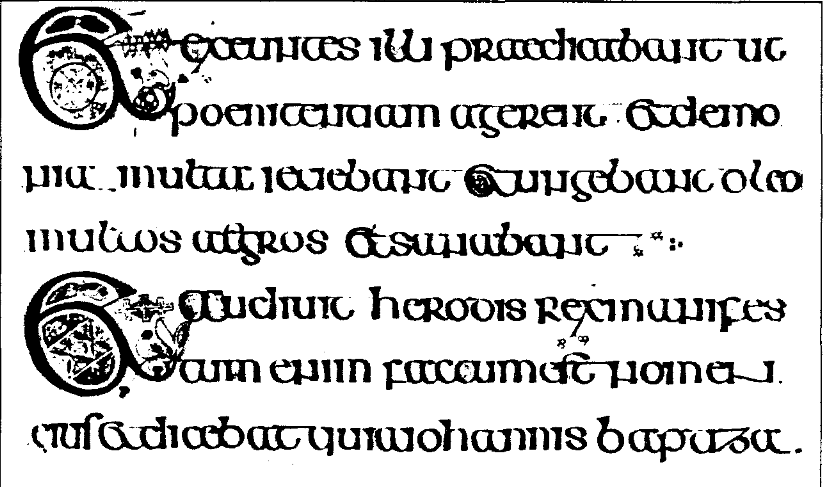
Bedell's Bible was subsequently to be used everywhere the Gaelic language was spoken: in Scotland, in America and in Canada. It is said that it was used even in Catholic liturgy in Ireland during the 1960s and 1970s, before the first complete approved Catholic Bible in Irish was published in 1981. Though Bedell's work was probably finished not long after 1630, and though he himself died in 1642, his Bible was not published until 1685. Indeed, it might never have been published had it not been for the good services of the illustrious scientist Robert Boyle. Boyle was the seventh son of Richard Boyle, Earl of Cork. It appears that by the time of Bedell, the brief affair of the English court with things Gaelic was over. Bedell's Elizabethan frame of mind was out of step with the time, or at least with the attitude of those in power in his day. When Boyle set about printing the Old and New Testaments, Bedell's manuscript was in a sorry condition. Parts of it were illegible, having become damp. Boyle had another difficulty: the Queen Elizabeth type had disappeared. He was informed that the fonts of this type had been stolen by the Jesuits and were being used at Douai to publish Catholic propaganda for shipment to Ireland. In fact, there is no record of any Catholic material printed with that type. Nonetheless, the fonts and all that went with them were gone. Boyle had a new set of Irish characters cast in London. Moxon was the cutter of this type, and it is thought that the model for it was the first Catholic type of Louvain. Boyle footed the entire bill for the printing of the Bible, around £330.

Although some copies of the Boyle printing were used in Gaelic Scotland, it is of interest to note that it was deemed necessary to create a new edition of the Bedell Bible, printed in the Roman character, for that part of Gaeldom. The Scots were never subsequently exposed to Gaelic printing based on the letter forms which their early saints had helped to create.

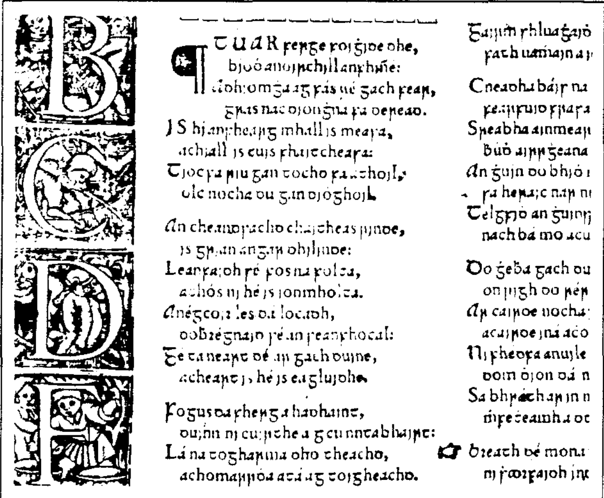
All the types we have mentioned, and indeed all those designed before 1840, were based on a spiky manuscript hand. It was not representative either of the best or of the most widely known of Irish letter forms. The beautiful writing in the *Book of Kells*, for example, is in a neat round hand, which displays moreover some features which have never been included in any Irish type. In the year 1841, the Irish Archaeological Society was founded. It was devoted to the study of the language, literature, history and antiquities of Ireland, and to the publication of material in those areas of interest. The Society arranged for the design and cutting of a new series of types for its publications. The designer of these is thought to have been George Petrie, and they were probably cut and cast in Dublin by James Christie. These fonts were based on the round hand, suitably modified to the medium of printing. They are clean-cut and easy to read, and owe little to the Irish types that preceded them; they can boast both originality and artistry. Moreover, they made a considerable contribution to the design of the important Keating type of 1863 and, through it, to the design of all the Irish types which followed. They were probably also the model for the very attractive twentieth century Colm Cille Gaelic type of Colm Ó Lochlainn.



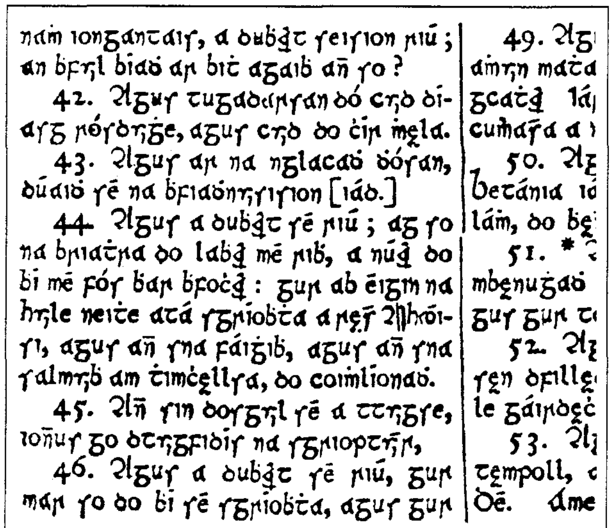
Some lines from the *Lindisfarne Gospels*. The Anglo-Saxons had been taught to write by the Irish.


Et praedicabant ut
poenitentiam agerent. Et demo
nia multa eiciebant et unguebant oleo
multos aegros et sanabant.
Et audiuit herodis rex manifes
tum enim factum est nomen ei
et dicebat quia iohannis baptiza

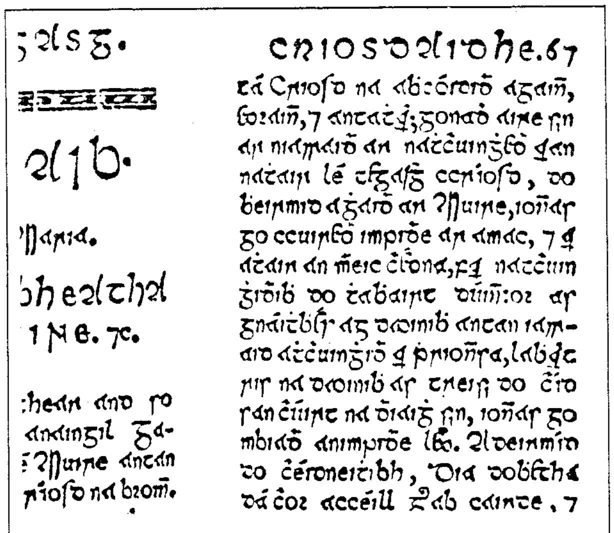
An example of Irish writing from the *Book of Kells*. Note the similarity between this and the much earlier *Lindisfarne Gospels*.



A part of *Tuair Ferge Foighide*, in the Queen Elizabeth type of 1571. This is the earliest known document to have been printed in Irish.



This is part of the New Testament printed using the Moxon type commissioned by Robert Boyle around 1680.



ſ ASG.

AIB.

Maria.

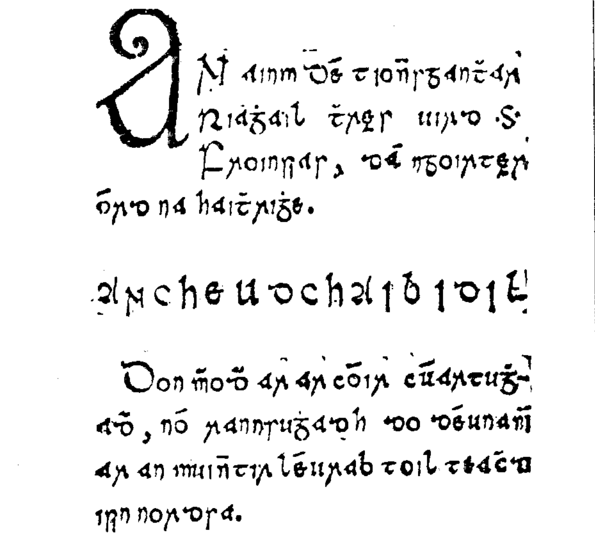
bhealtha INE. 7c.

:hean and ro angil Ga- ; Muire antan riosd na broin.

CRIOSDAIDHE. 67

tá Críosd na ábsóroid agam,
boráin, 7 antaċt; gonad aire ſin
ar niarraid an naċċuingid gan
naċain lé ċ[illegible] ccriosd, do
beirmid agaid an Muire, ionar
go ccuireó impide ar amac, 7 ag
aċain an ṁeic ċlóna, [illegible] naċċuin
ġidib do ṫabairt dúinn: oir ar
gnáiṫbír ag daoinib antan iarr-
aid aċċuingid ar prionsa, labairt
rir na daoinib ar treiſ do ċid
ran ċúirt na diaiġ ſin, ionar go
mbiadó animpide leó. Adeirmid
do ċéidneitibh, Dia dobétha
dá ċoi accéill ġab cainte, 7

Part of a very common catechism printed in Antwerp with the Louvain type of 1611.

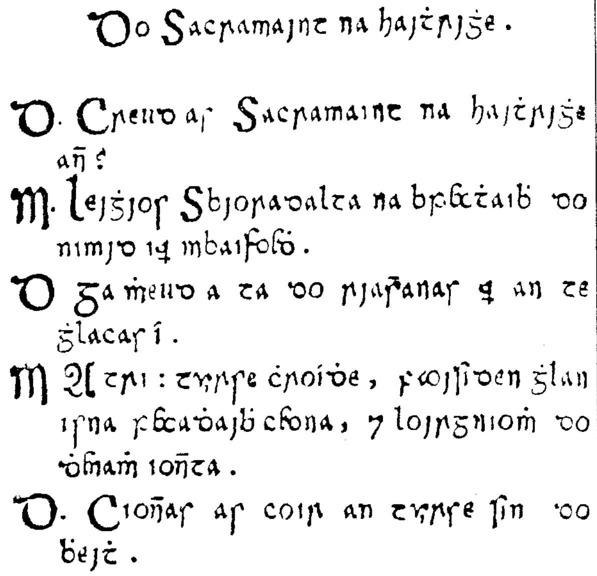


AN ainm Dé tionnsgantar
Riaġail treas uird S.
Froinsias, dá ngoirtear
ord na haiṫriġe.

An cheud chaibidil.

Don mod ar ar cóir cuartuġ-
ad, nó ranntugadh do dhéunamh
ar an muinntir lé ar ab toil teaċt
isin nord-sa.

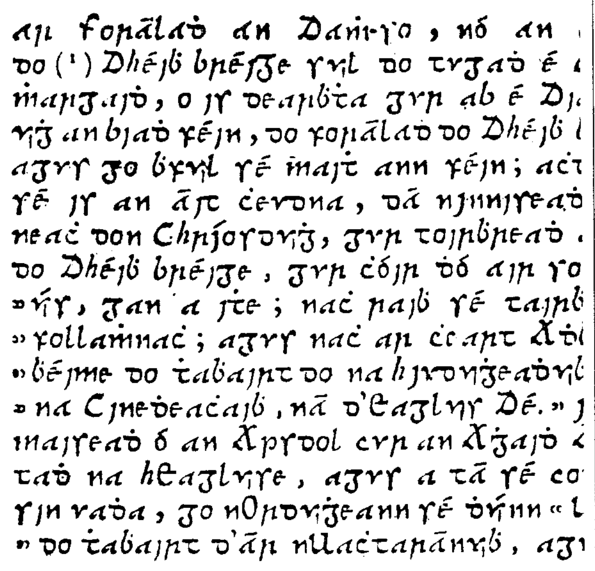
Religious material printed using the Louvain type of 1641.



Do Sacramainte na haiṫriġe.

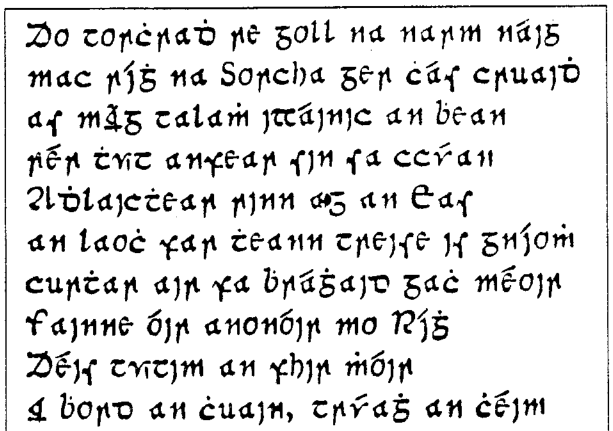
D. Creud as Sacramainte na haiṫriġe
ann?
M. Leiġios Spioradalta na bpeacaib do
nímid iar mbaisdeadh.
D. Ga ṁeud a ta do riaċtanas ag an te
ġlacas í.
M. A tri: truaiġe ċroiḋe, faoisidin ġlan
isna peacaḋaib ċeana, 7 loirgníom do
ḋénam ionta.
D. Cionnas as cóir an truaiġe ſin do
beiṫ.

The Rome type of 1676.



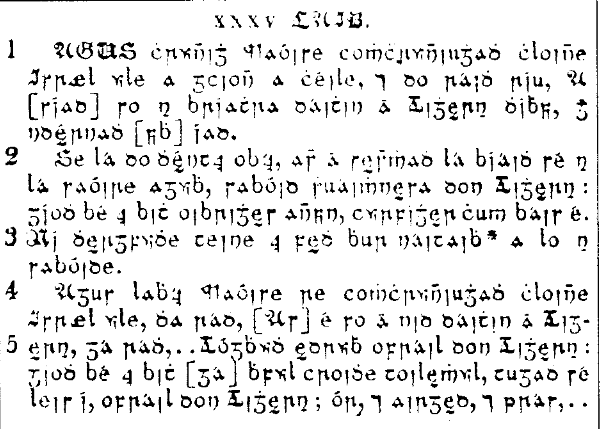
ar foráilaḋ an Daniṡo, nó an
do (¹) Dhéib bréige sul do tugaḋ é
marḃaiḋ, o is dearḃṫa gur ab é Di
ug an biaḋ féin, do foráilaḋ do Dhéib
agus go bfuil sé maiṫ ann féin; aċt
sé is an art ċéudna, dá ninnisteaḋ
neaċ don Chriosduiġ, gur toirbreaḋ
do Dhéib bréige, gur cóir dó air so
»uis, gan a iṫe; naċ raib sé tairb
» follaṁnaċ; agus naċ ar ceart Aḋ
» béime do ṫabairt do na hiodhuiġeaḋuib
» na Cineḋeaċaib, ná d'Eaglais Dé.»
maiseaḋ ó an Abstol cum an Aṫaiḋ
taḋ na hEaglaise, agus a tá sé co
sin uada, go nOrduiġeann sé ḋúinn «
» do ṫabairt d'ár nUaċtaránuib, agu

The Paris type of 1732.



Do torċraḋ re goll na narm náig
mac ríġ na Sorcha ger ċás cruaiḋ
as mág talaṁ iṫṫáinic an bean
rér ṫuit anfear sin sa ccúan
Aḋlaicṫear rinn ag an Eas
an laoċ far ṫeann treise is gníoṁ
curṫar air sa bráġaid gaċ méoir
fáinne óir anonóir mo Ríġ
Déis tuitim an fhir ṁóir
ag bord an ċuain, truaġ an ċéim

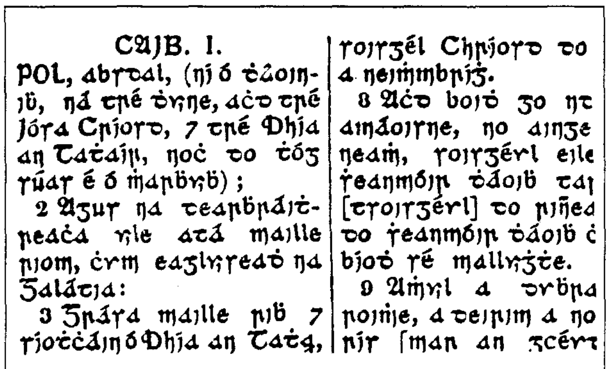
The large well-formed Brooke type of 1789.



XXXV CAIB.

1 AGUS ċruinniġ Maoise coiṁċruinniuġaḋ ċloinne
Israel uile a gcionn a ċéile, 7 do raiḋ riu, A
[siad] so na briaṫra daiṫin a Tiġerna dibh, g
ndéanad [sib] iad.
2 Se lá do déuntar obair, aċt a seaċtṁad lá biaiḋ sé n
lá naoṁṫa agaib, saboid suaimneasa don Tiġerna:
giod bé ag biṫ oibriġes ann, cuirfiġear ċum bais é.
3 Ni deargṫaiḋe teine ag fed bur naitaib* a lo n
saboide.
4 Agus labair Maoise re coiṁċruinniuġaḋ ċloinne
Israel uile, da raḋ, [As] é so a nid daiṫin a Tiġ-
5 erna, ga raḋ, ..Togbaiḋ eadraib ofrail don Tiġerna:
giod bé ag biṫ [ga] bfuil croiḋe toileamhail, tugad sé
leis í, ofrail don Tiġerna; ór, 7 airgeod, 7 prar,..

Fry's type, 1819.



CAIB. I.

POL, abstal, (ni ó ḋaoinib, na tré ḋuine, aċd tré Iósa Criosd, 7 tré Dhia an Taṫair, noċ do ṫóg suas é ó ṁarbuib);

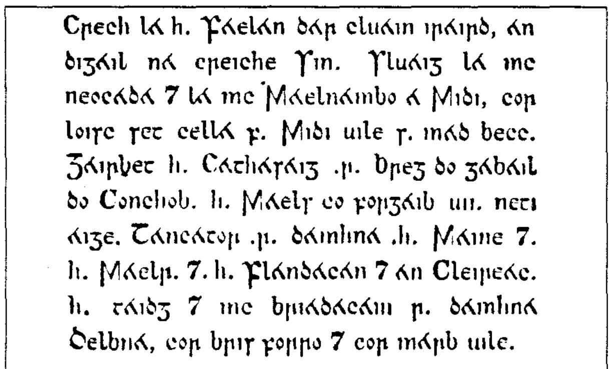
2 Agus na dearbraiṫreaċa uile atá maille riom, ċum eaglaiseaḋ na Galátia:

3 Grása maille rib 7 síoṫċáin ó Dhia an Taṫar, soisgél Chriosd do a neiṁiṁbriġ.

8 Aċd boiḋ go nt aindaoisne, no ainge neaṁ, soisgéul eile ṡeanmóir ḋaoib tai [tsoisgéul] do riñea do ṡeanmóir ḋaoib ċ bioḋ sé malluiġṫe.

9 Aṁuil a dubra roiṁe, a deirim a no nis ſman an sgéu

The Watts type of 1818.



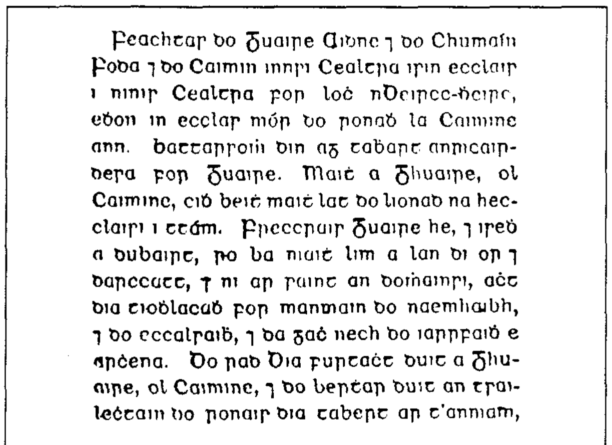
Crech la h. Faelan dar cluain iraird, an digail na creiche sin. Sluaig la mc neocada 7 la mc Maelnambo a Midi, cor loisc sec cella f. Midi uile s. mad becc. Gairbet h. Cathasaig .r. Breg do gabail do Conchob. h. Maels co forgaib uii. neci aige. Tancator .r. damlina .h. Maine 7. h. Maelr. 7. h. Flandacan 7 an Cleirec. h. taidg 7 mc bruadacain r. damlina Delbna, cor bris forro 7 cor marb uile.

The rather unattractive Figgins type of 1825.

P̄S in nostra insola que uocatur hibernia
ostensus est hominibus maximis mirabilibus
que perfecit per felicem celestis uite uirginem
precellentem pro merito magno in mundi circulo

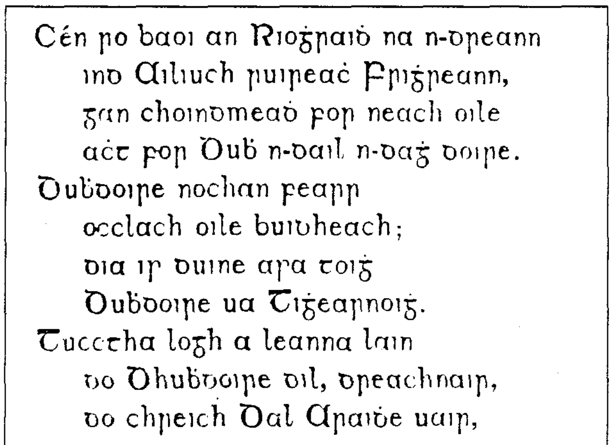
Ymnus iste angelice summeque sancte brigite
fari non ualet omnia uirtutum mirabilia
que nostris nunquam auribus si sint facta audiuimus
nisi per istam uirginem marie sancte similem

Capitals of one of the Irish Archaeological Society's types used as ordinary type, 1855.



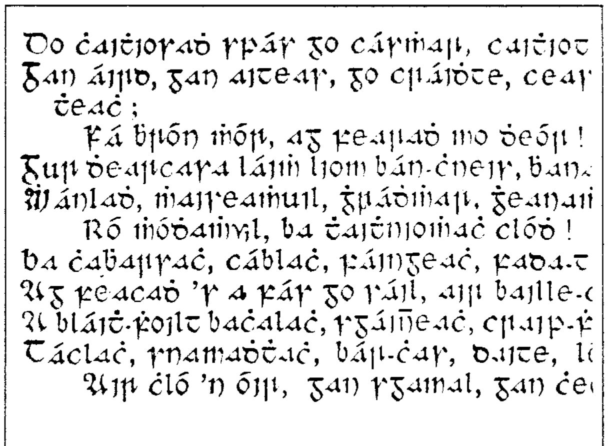
Feachtas do Guaire Aidne ⁊ do Chumain Foda ⁊ do Caimin innsi Cealtra isin ecclais i ninis Cealtra for loċ nDeircc-ḋeirc, edon in ecclas mór do ronaḋ la Caimine ann. Battarsom din ag tabart anmcairdesa for Guaire. Maiṫ a Ghuaire, ol Caimine, ciḋ beiṫ maiṫ lat do lionaḋ na hecclaisi i ccám. Frecrair Guaire he, ⁊ isseḋ a dubairt, ro ba maiṫ lim a lan di or ⁊ darccatt, ⁊ ni ar saint an doṁainsi, aċt dia tiodlacaḋ for manmain do naemhaibh, ⁊ do eccalsaiḃ, ⁊ da gaċ nech do iarrfaiḋ e arċena. Do rad Dia furtaċt duit a Ghuaire, ol Caimine, ⁊ do bertar duit an trai-leċtain do ronais dia tabert ar t'anmain,

Irish Archaeological Society type, 1841.

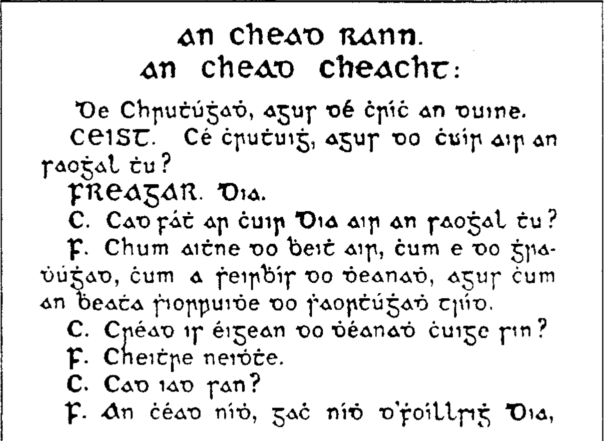


Cén ro baoi an Ríoġraiḋ na n-Ereann
ind Ailiuch ruiseaċ Frigreann,
gan choindmeaḋ for neach oile
aċt for Dub n-dail n-daġ doire.
Dubdoire nochan fearr
occlach oile buidheach;
dia is duine ara toiġ
Dubdoire ua Tiġearnoiġ.
Tucctha logh a leanna lain
do Dhubdoire dil, dreachnair,
do chreich Dal Araiḋe uair,

Another Irish Archaeological Society type, 1841.

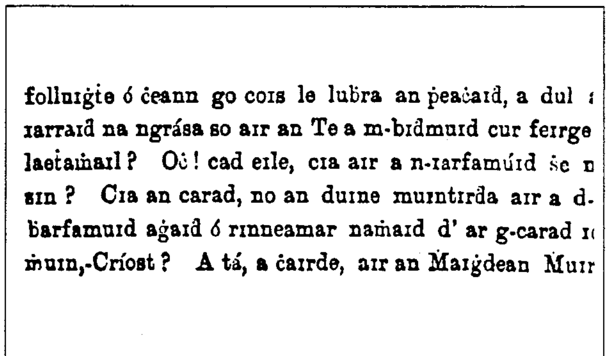

Do ċaiṫiosaḋ sfás go cásṁair, caiṫioc
Gan áirid, gan aiteas, go cráiḋte, ceas
ṫeaċ ;
Fá ḃrón ṁór, ag fearaḋ mo ḋeór !
Gur ḋeascasa láiṁ liom bán-ċneis, ban-
Ṁánlaḋ, ṁaiseaṁuil, ġráḋṁar, ġeanai
Ró ṁóḋaṁuil, ba ṫaiṫnioṁaċ clóḋ !
Ba ċaḃairsaċ, cáblaċ, fáinneaċ, fada-t
Ag feacaḋ 's a fás go sáil, air baille-c
A bláiṫ-ḟoilt baċalaċ, sgáinneaċ, craip-f
Táclaċ, snamaḋṫaċ, bár-ċas, daite, li
Air cló 'n óir, gan sgamal, gan ċe

Christie's type, 1815.

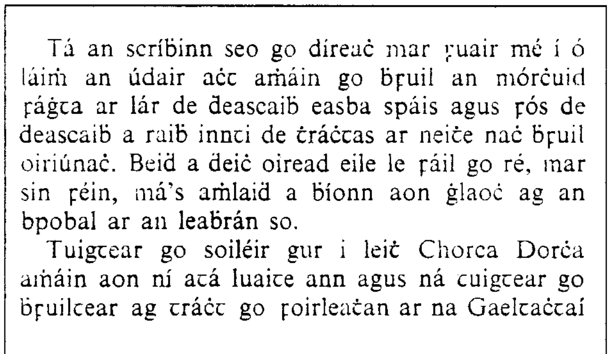

an cheaD Rann.
an cheaD cheacht:

De Chruṫúġaḋ, agus ḋé ċríċ an duine.
CEIST. Cé ċruṫuiġ, agus do ċuir air an saoġal ṫu?
FREAGAR. Dia.
C. Cad fáṫ ar ċuir Dia air an saoġal ṫu?
F. Chum aiṫne do ḃeiṫ air, ċum e do ġra-ḋúġaḋ, ċum a ṡeirḃís do ḋeanaḋ, agus ċum an ḃeaṫa ṡíorruiḋe do ṡaorṫúġaḋ tríd.
C. Créad is éigean do ḋéanaḋ ċuige sin?
F. Cheiṫre neiṫe.
C. Cad iad san?
F. An ċéad níḋ, gaċ níḋ d'ḟoillsiġ Dia,

The seminal Keating Society type, 1863.

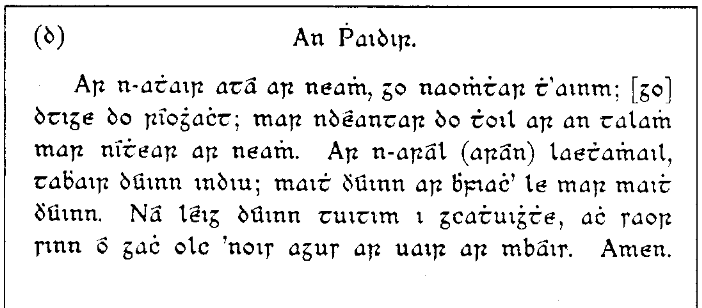

follnıġte ó ċeann go coıs le luḃra an ṗeaċaıd, a dul
ıarraıd na ngrása so aır an Te a m-bıdmuıd cur feırge
laeṫaṁaıl? Oċ! cad eıle, cıa aır a n-ıarfamúıd ṡe n
sın? Cıa an carad, no an duıne muıntırda aır a d-
ḃarfamuıd aġaıd ó rınneamar naṁaıd d' ar g-carad ı
ṁuın,-Críost? A tá, a ċaırde, aır an Maıġdean Muır

An attempt by Canon Burke to produce a hybrid Gaelic-Roman type, 1877. Note the ridiculous *séimhiú* over the lower case 't'.


Tá an scríḃinn seo go díreaċ mar ḟuair mé í ó láiṁ an údair aċt aṁáin go bḟuil an mórċuid ḟágṫa ar lár de ḋeascaiḃ easba spáis agus ḟós de ḋeascaiḃ a raiḃ innti de ṫráċtas ar neiṫe naċ bḟuil oiriúnaċ. Beiḋ a ḋeiċ oiread eile le ḟáil go ré, mar sin ḟéin, má's aṁlaiḋ a ḃíonn aon ġlaoċ ag an bpobal ar an leaḃrán so.

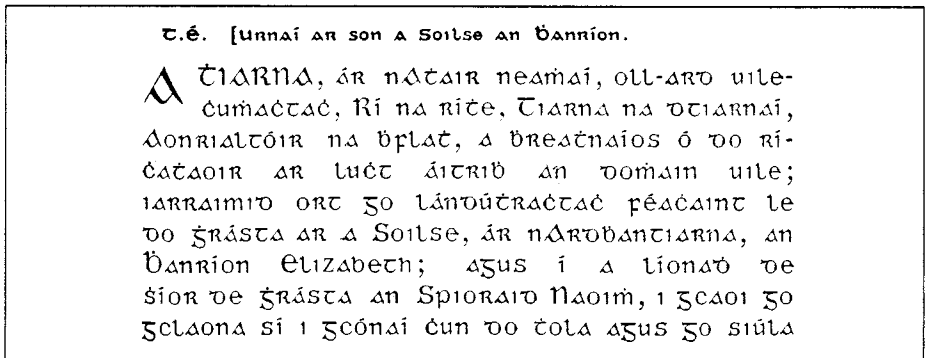
Tuigtear go soiléir gur i leiṫ Chorca Dorċa aṁáin aon ní atá luaite ann agus ná tuigtear go bḟuiltear ag tráċt go ḟoirleaṫan ar na Gaeltaċtaí

A rather better attempt at hybrid type, Liam Miller's *Times New Roman* was used in one edition of *An Béal Bocht*.


(d) An Ṗaidir.

Ar n-aṫair atá ar neaṁ, go naoṁṫar ṫ'ainm; [go] dtige do ríoġaċt; mar ndéantar do ṫoil ar an talaṁ mar níṫear ar neaṁ. Ar n-arán (arán) laeṫaṁail, taḃair dúinn indiu; maiṫ dúinn ar bḟiaċ' le mar maiṫ dúinn. Ná léig dúinn tuitim i gcaṫuiġṫe, aċ saor sinn ó gaċ olc 'nois agus ar uair ar mbáis. Amen.

A rather stylish Gaelic type cut about 1925, possibly in Germany.


T.É. [Urnaí ar son a Soilse an Ḃanríon.

A Tiarna, ár nAṫair neaṁai, oll-ard uile-ċuṁaċtaċ, Rí na ríṫe, Tiarna na dTiarnaí, Aonrialtóir na bḟlaṫ, a ḃreaṫnaios ó do rí-ċaṫaoir ar luċt áitriḃ an doṁain uile; iarraimid ort go lándúṫraċtaċ féaċaint le do ġrásta ar a Soilse, ár nArdḃantiarna, an Ḃanríon Elizabeth; agus í a líonaḋ de ṡíor de ġrásta an Spioraid Naoiṁ, i gcaoi go gclaona sí i gcónaí ċun do ṫola agus go siúla

Standard twentieth century Gaelic type. Excerpt from the *Book of Common Prayer* of the Church of Ireland. Here Irish-speaking Protestants in Northern Ireland pray for their Queen. Note the foreign 'z' in her name.

20. Agus duairt Dia: "Beireaḋ na h-uisgí go líonṁar an créatúir corruíoċ 'na ḃfuil beaṫa, agus éanlaiṫ eiteallas ós cionn na talṁan in iormailt ġil neiṁe". Ċruṫuiġ Dia freisin míolta móra agus gaċ créatúir beó ċorruíos; rug na h-uisgí iad go líonṁar agus na h-éanlaiṫ sgiaṫánaċa do réir a gcinéil. Agus ba léir do Ḋia goma ṁaiṫ é a ḟaoċar. Ḃeannuiġ Dia annsin iad: "Bíḋiḋ torraċ", deir sé, "agus foirlíonaiḋ, agus líonaiḋ na h-uisgí atá sna fairrgí, agus foirlíonaḋ na h-éanlaiṫ ar an talaṁ". Idir nóin agus maidin ḃí an cúigeaḋ lá istiġ.

The truly artistic Colm Cille Gaelic type of Colm Ó Lochlainn.

Ciotóg Colm Cille

Feaċt n-aon do ċuaiġ Colm Cille i gceann Ḟinnéin Ḋroma Ḟinn agus d'iarr iasaċt leaḃair air, agus fuair sé sin ó Ḟinnian. Agus d'ḟanaḋ sé i ndiaiḋ ċáiċ tar éis na dtráṫ agus na nAifreann sa teampall do ḃí sa ḃaile sin féin agus do ḃí sé ag scríoḃ an leaḃair ann gan ḟios d'Ḟinnian. Agus an uair ṫigeaḋ an oíċe ċuige, is iad ba ċoinnle dó ag déanaṁ na scríḃneoireaċta sin, .i. cúig meoir a láiṁe deise do lasaḋ aṁail cúig lóċranna rólasúnaċa ionas go gcuiridís dealraṁ agus solas fán teampall uile.

Agus an oíċe déanaċ do Ċolm Cille ag scríoḃ deiriḋ an leaḃair sin, do ċuir Finnian duine d'iarraiḋ a leaḃair air. Agus ar ndul go doras an teampaill a raiḃ Colm Cille dó, do b'iontaċ leis méad na soilse do ċonaic sé istiġ agus do ġaḃ eagla ṁór é, agus d'ḟéaċ sé go faiteaċ trí ṗoll do ḃí ar ċoṁla dorais an teampaill, agus ar ḃfeiscint Ċolm Cille dó ar an innealL sin, aṁail a dúramar roṁainn, níor lig an eagla dó labairt ris ná an leaḃar d'iarraiḋ air. Do foilsioḃ, iomorro, do Ċolm Cille an t-óglaċ do ḃeiṫ 'gá ḟeiṫeaṁ aṁlaiḋ sin, agus do ġaḃ fearg ṁór é fán ní sin, agus do laḃair sé re peata coirre do ḃí aige, agus is ea dúirt ria:

"Is cead liomsa, más cead le Dia, tusa do ḃuain a ṡúl as an óglaċ úd do ṫáinig dom ḟéaċaint gan ḟios dom féin."

D'éiriġ an ċorr i gcéadóir le briaṫar Ċolm Cille, agus ṫug buille dá gob trí ṗoll na coṁla i súil an óglaiġ gur ḃain a ṡúil as a ċeann, gur ḟág ar a ġrua amuiġ í. D'imiġ an t-óglaċ iar sin mar a raiḃ Finnian agus d'inis dó mar a d'imiġ air ó ṫús go deireaḋ. Níor ṁaiṫ le Finnian an ní sin, agus do ḃeannaiġ agus do ċoisric sé súil an óglaiġ, agus do ċuir ina hionad féin arís í, gan dioḃáil, gan uireasa do ḃeiṫ uirṫi, aṁail do ḃí sí ó ṫús.

Agus mar do ċuala Finnian a leaḃar do scrioḃ gan ċead dó féin, do ċuaiġ sé d'agairt Ċolm Cille ann, agus dúirt nár ċóir a leaḃar do scríoḃ gan ċead dó.

"Do ḃéarsa breiṫ Rí Éireann dinn," ar Colm Cille, .i. breiṫ Ḋiarmada ṁic Cearḃaill.

"Gaḃfadsa sin," ar Finnian.

Do ċuadar re ċéile ina ḋiaiḋ sin go Teaṁair na Rí, mar a raiḃ Diarmaid mac Cearḃaill, agus d'inis Finnian a scéala ar dtús don rí agus is ea a dúirt:

"Do scríoḃ Colm Cille mo leaḃar gan ḟios dom féin," ar sé, "agus deirim gur liom féin mac mo leaḃair."

"Deirimse," arsa Colm Cille, "naċ misce leaḃar Ḟinnéin ar scríoḃ mé as, agus naċ cóir na niṫe diaga do ḃí sa leaḃar úd do ṁúċaḋ nó a ḃacaḋ dom féin ná do ḋuine eile a scríoḃ ná a léaṁ ná a síolaḋ fá na ciníoċaiḃ; agus fós deirim, má do ḃí tairḃe do na poiḃleaċaiḃ, agus gan dioḃáil d'Ḟinnéin nó a leaḃair do ṫeaċt as, gur ceadaiṫe dom a scríoḃ."

Is ansin rug Diarmaid an ḃreiṫ oirirc, .i. "Le gaċ boin a boinín, agus le gaċ leaḃar a leaḃrán."

Eleven point Cló Naithí, 1990. This typeface is still in the experimental stage, and will be improved. The sample is printed at 300dpi using an HP LaserJet III.

The least Irish and least attractive of all Gaelic typefaces was that of Vincent Figgins, the first typefounder of that name. His Irish type of 1825 was apparently used in only two books. This is the same Figgins who, in 1832, made the first major promotion of sans serif types and popularized the name by which they are known today. By the beginning of the twentieth century, there was a considerable demand for printed material in Irish, and the Irish fonts necessary for mass production were not available. It was the London firm of Figgins which satisfied the demand. The same firm, under the name of Stevens, was responsible for the 1922 monotype which became familiar to succeeding generations of Irish school children as the standard form of the Irish letter before it was gradually phased out in the 1960s by governmental decree.

Though, in the Irish language revival, there had always been a faction which wanted to adopt the roman letter forms, it is probable that the main reason for its eventual success was a purely practical one. The variety of sizes and weights in which Irish type was available was extremely limited; moreover, Irish typewriters were old-fashioned and clumsy. The written language was apparently being strangled by its dependence on materials which did not belong to the mainstream of European trade and culture. In any case the change was made, and a whole generation has now grown up for the most part ignorant of a very distinctive part of its history.

TEX may well prove to be one of the most important innovations in the history of typesetting. If that is true, then it is even more certain that the invention of METAFONT will come to be regarded as revolutionary for type design and casting. If METAFONT had been available thirty years ago, it might well have saved the Irish typeface. As it is, Irish fonts in a range of weights and sizes unimaginable a generation ago are now being made using SBMF, a particularly fast PC version of METAFONT perfected by Dr Wayne Sullivan in the Mathematics Department of University College, Dublin.

The new typeface, known as Cló Naithí after one of the local saints of the area of County Dublin in which I live, already comprises the most extensive set of Gaelic fonts ever produced. It is, moreover, the only truly Gaelic typeface to include in a single design all twenty six letters of the modern English alphabet. It is intended primarily for everyday typesetting, not for fancy work; simplicity and readability have been the main considerations in its design. The point sizes of five, ten, eleven and seventeen are available in four types: unslanted and slanted, each at standard weight and in an extended bold form. There is yet a considerable amount of work to be done: parameters for other point sizes need to be worked out; the numerals and many special characters have to be designed; there are some questions about punctuation which must be settled; essential variant forms of letters and optional ligatures have to be created; sans-serif routines have to be finalised; large initial fancy letters, which were quite a common feature in Gaelic printing, should be included; work on kerning is yet to begin; moreover there are many imperfections in what has already been done, and these must be weeded out relentlessly. The Naithí project should also include mathematical fonts so that technical material can be typeset without difficulty; it ought also to include the production of a comprehensive hyphenation dictionary. How much of this can be achieved will probably depend on whether or not the workforce of one and the zero funding level can be increased.

There are more than sixty phonemes in the Irish language; it is thus phonetically one of the richest languages in Europe. To accommodate this splendid variety of sound, the Irish traditionally employ an alphabet of only seventeen letters; this number does not include aitch, which is used only to prevent hiatus; nor does it include the various letters which have crept in with some loan words in the past few years. The vowel sounds of the language are, for the most part, pure; they may be either short or long, the latter usually being indicated in the written language by a length mark, commonly called the *síneadh fada*; this is similar in appearance to the French acute accent.

Each of the twelve consonants is used to represent at least two distinct sounds, one broad and one slender; which of the two sounds is intended is usually indicated by adjacent vowels — if these are slender (e or i), then so is the consonant; otherwise the consonant is broad. Most of the consonantal sounds (of both varieties) are subject to a qualitative softening, indicated by the placing of a dot, called a *séimhiú*, over the appropriate letter. These sounds may also be subject to vocalization or nasalization, which is indicated by placing the appropriate consonant before the one to be inflected. There are, properly speaking, no written accents in Irish, and only the two diacritical marks noted here. The language is written more or less phonetically according to its own phonetic rules, which differ considerably from those of English.

The Irish language, in both spoken and written forms, is highly inflected. Approximately one in six vowels takes a length mark, and around one in five consonants a softening mark. Plain TEX's English language conventions for accessing diacritical marks are therefore clearly unsuitable for typesetting in Irish. In Cló Naithí all diacritical marks are treated as ligatures: for example, 'ch' is printed as 'ċ' and '/a' as 'á'; the ligatures can of course be broken if that is desired. On the other hand, the situation for the user of Computer Modern with standard conventions is considerably eased by noting that the correspondence between letters and diacritical marks is one-to-one. By making, say, the forward slash active and writing the appropriate macro, each desired inflection can be got by typing the slash before the letter to be inflected. There are few Latin alphabet languages in which such a proliferation of inflections can be dealt with so simply by TEX.

For many of us whose homes are Gaelic-speaking, there is little doubt that Gaelic fonts are more suitable for the printing of our language than are the Roman. One reason for this is the profusion of aitches with their ugly ascenders which take the place of the *séimhiú* in the roman fonts. Around twenty per cent of all other consonants are followed by the letter aitch; this percentage would have been even greater had not simplified spelling accompanied the change to roman fonts*. Historically, there is justification for the introduction of the aitch, at least in conjunction with the letters c, p and t, but that should hardly be used to support the widespread adoption of ugliness; the dot is less conspicuous than the ascending h, and it also helps to provide a sense of balance in a typeface with upper-case ascenders. Another reason for preferring the Gaelic typeface arises from the difficulty of placing diacritical marks over upper-case roman letters whose height already extends more or less to the extremities dictated by the typeface; this difficulty Irish has in common with many languages. In Gaelic typefaces it has never been a problem, since they have upper-case ascenders which determine that the natural height of fonts is sufficient to accommodate all such marks without any squeezing. The Gaelic typeface has tradition, beauty and practicality on its side, and it will give many of us great joy to see it being used more frequently in the future.

* Despite my interest in the re-emergence of Gaelic typefaces, I have no intention of promoting old spelling conventions.

Some Sources

Anderson, Donald M. "The Art of Written Forms." New York: Holt, Rinehart and Winston, inc., 1969.

Atkins, Gordon. "The Classification of Printing Types." Leicester, England: Apple Barrell Press, 1975.

Berry, W. Turner and A.F. Johnson. "Encyclopaedia of Type Faces." London: Blandsford, 1953.

Breatnach, Deasún. "The Best of the English. William Bedell and the Irish version of the Old Testament." Cavan, Ireland: Abbey Printers, 1971.

Dickins, Bruce. "The Irish Broadside of 1571 and Queen Elizabeth Types." *Transactions of the Cambridge Bibliographical Society*, 1, pages 48–60, 1949.

Henry, P.L. "Saoithiúlacht na Sean-Ghaeilge." Baile Átha Cliath, Éire: Oifig an tSoláthair, 1978.

Johnson, A.F. "Type Designs. Their History and Development." 3rd ed. London: Andre Deutsch, 1966.

Laistner, Max L.W. "Thought and Letters in Western Europe A.D. 500 to 900." London: Methuen, 1931.

Lynam, E.W. "The Irish Character in Print 1571–1923." Dublin: Irish University Press, 1968.

Mac Giolla Chomhaill, Anraí. "Beatha Cholm Cille." Baile Átha Cliath, Éire: Foilseacháin Náisiúnta Teoranta, 1981.

Martin, Judy. "The Complete Guide to Calligraphy." London: Quill, 1984.

Morison, Stanley . "On Type Designs Past and Present." Revised edition. London: Benn, 1962.

Ní Mhuiríosa, Máirín. "Gaeil agus Breatnaigh Anallód." Baile Átha Cliath, Éire: Clodhanna Teoranta, 1974.

Ó Fiaich, An Cairdinéal Tomás. "Gaelscrínte san Eoraip." Baile Átha Cliath, Éire: Foilseacháin Ábhair Spioradálta, 1986.

Ó Fiannachta, Pádraig. "Milis an Teanga." Corcaigh agus Baile Átha Cliath, Éire: Cló Mercier, 1974.

Williams, N.J.A. "I bPrionta i Leabhar." An Clóchomhar, Baile Átha Cliath, Éire: 1986.

An International Phonetic Alphabet

Dean R. Guenther
Washington State University, Computing Service Center, Pullman, WA. 99164-1220, USA
Bitnet: `GUENTHER@WSUVM1` Internet: `GUENTHER@WSUVM1.CSC.WSU.EDU`

Janene K. Winter
American Mathematical Society, P.O. Box 6248, Providence RI. 02940, USA
Internet: `JXW@MATH.AMS.COM`

Abstract

This paper discusses the development of an International Phonetic Alphabet (IPA) font for use with TEX and the Computer Modern fonts primarily from the viewpoint of a manager. Included will be discussion on how to use this font and the direction we can go from here with it.

The Need for an IPA Font

Since 1983, TEX has found extensive use at Washington State University (WSU). Although at many sites, mathematics typesetting is a primary reason for using TEX, at WSU, the area of textual typesetting has always been the main thrust of TEX usage. In fact, the pioneer project for TEX at WSU was a critical edition of Robert Burton's seventeenth-century *The Anatomy of Melancholy* by Professors Thomas Faulkner and Nicolas Kiessling.

TEX's popularity as a text processor at WSU grew each semester and, as a whole, TEX earned rave reviews from its users. However, one complaint, or limitation, that was frequently echoed throughout the halls of the WSU Computing Service Center (especially from the liberal arts departments), was the lack of fonts available for use with TEX which contained certain special characters that were often needed (such as the Old English "thorn" (þ) or any unusually accented character of a native American or some other dialect). These "complaints" provided the seed of thought for a future METAFONT font that would contain some of these special characters.

Around 1986, this "future font" was escorted from the recesses of the "some-day-it-would-be-nice" projects, to front-row status as several linguists brought to our attention the need for an IPA font to work with TEX and Computer Modern fonts. Up to that time, for most linguists who wanted to use a diacritic or special character, it was frequently necessary to have the character hand drawn in, as illustrated in Figure 1. Early word processors allowed for the ability to superscript/subscript and overlay of some characters, and the introduction of TEX gave the phonetician much more flexibility as pointed out by Christina Thiele (Carleton University, Canada) in *TEX, Linguistics, and Journal Production.*

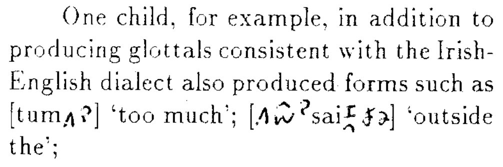

One child, for example, in addition to producing glottals consistent with the Irish-English dialect also produced forms such as [tumʌʔ] 'too much'; [ʌw̃ʔsaiʃ ʃə] 'outside the';

Figure 1: Hand drawn phonetic characters

Still, there were frequently used characters that could not be created by combining existing characters or accents. In fact, some characters, such as the ejective (ʔ) required the removal of bits, rather than the addition. Certainly we could have equipped each linguist on campus with a bottle of White Out, or perhaps we could have created a White Out font, but neither seemed as effective as a new IPA font.

There were already some IPA fonts available. However, they were not designed with METAFONT, and they did not always work well with Computer Modern. Figure 11 shows the `ph10` font designed by Jean-Pierre Paillet in 1983 in consultation with William Cowan, editor of the *Canadian Journal of Linguistics*. It was a bitmapped font created by hand, designed to work with `amr10`. Figure 2, a sample from Richard Rhodes' article in the

(5) Vowel harmony

Orthography	Standard	Métchif
pouilleux	[pujö]	[pʊjü] ~[pʊ̈jü]
mesure	[mɛzür]	[mɪzür] ~[mʊ̈zür]
chevreuil	[šəvrɔ̈j]	[šʊvrü]
fusil	[füzij]	[fɪzi]
musique	[müzik]	[mʊ̈zük]

Figure 2: ph10 with amr10

Actes du dix-septième congrès des algonquinistes, shows that this marriage worked well. When the crisper Computer Modern became available, ph10 proved to be too heavy. Figure 9 shows the use of ph10 with cmr10. This illustration comes from Terry Piper's article "On the Difference Between L1 and L2 Acquisition in Phonology" in the *Canadian Journal of Linguistics*. The only way to enhance the use of ph10 with Computer Modern would have been to tweak the pixels for each character by hand. One of these character bitmaps is illustrated in Figure 12.

The Development

To begin with, Janene Winter began experimenting with various characters, writing the code from scratch as she was attempting to learn METAFONT. These attempts created some rather unusual (and very non-IPA) characters during the learning process. This also created no small amount of frustration as Janene attempted to learn METAFONT on her own, for nobody else on campus knew or used METAFONT. Adding to this already less than ideal situation was the fact that there was no good graphics previewer available on VM/CMS so she could preview her work on the terminal. Each iteration of a character had to be printed on one of the University's laser printers, which was shared with about 2,000 other users. As can be imagined, this became a painstakingly slow process, not to mention all of the trees that went through the laser printer.

The first major breakthrough in the development of the WSU IPA was the discovery that the Computer Modern METAFONT code really was decipherable and could be used as a foundation on which to build from existing METAFONT characters, and whenever possible simply modify them to fit the required character. Shortly after this, a graphics terminal was obtained which allowed on-screen previewing of the characters via two preview programs. One from Malki Cymbalista (Israel) and the other from Georg Bayer (Germany).

With these two monumental accomplishments, it became obvious that a METAFONT IPA font could indeed be more than a "wouldn't-it-be-nice-if-we-could" project. Now that we were much more clear on the "how" were we going to do this, the next step was to specifically define "what" were we going to do—which characters, and which style or design of the characters would be used. Finding decent typographical examples of the IPA characters was more difficult than anticipated. Most books found in the campus library that showed the characters at all were at least 40 years old, and the largest character samples were printed in 10pt. Finally, Pullam and Ladusaw's *Phonetic Symbol Guide* was discovered. This proved to be a major find, as it had excellent IPA examples in a larger 19pt size. And more importantly, the equivalent of an x-height and baseline was also traced in for all characters illustrated.

The base design for the IPA was 10pt. Making additional sizes came with with their own sets of problems. For example, going from 10pt to 9pt was usually fine, but going to 8pt would sometimes cause problems in some of the point and stroke definitions. Additional METAFONT errors were encountered when slanted faces were generated. METAFONT's "strange turning path" message became a rather familiar sight at that time. In most cases, even though the error messages were generated, the characters themselves looked fine. Eventually, the magic combination of correct parameter values and point definitions was found and all error messages disappeared.

Another pitfall that was not obvious at first was the misplacement within the character box of a few of the characters. These were simply Computer Modern characters rotated or flipped, or sometimes both. The first few attempts at transforming these characters often gave the correct visual result, but later examination would show that the baseline or height was wrong. The baseline problems would usually be easily detected by placing the character in a line of Computer Modern text. The height problems would not show up until the character was used with an accent.

These placement problems occurred because the Computer Modern METAFONT code was not changed. It was simply "transformed," and therefore, the original Computer Modern character's height, depth and x-height were being used. This problem was solved simply by changing all of METAFONT's point definitions that were relative to the character's height, depth or x-height, from the

A shibilant is a term occasionally found for a fricative corresponding to a "hushing" sound, e.g., IPA [ʃ] (more technically, a grooved laminal fricative).

Figure 3: Using the IPA Characters

original Computer Modern character, to the new transformed IPA character.

Transforms for the slanted fonts brought further difficulties because METAFONT slants a character before it executes any of its transformations. This caused the IPA characters that were created by rotating existing Computer Modern characters that were then slanted, to be slightly off center in their character boxes. It took a while to master all the various transforms.

Karen Mullen (University of Louisville, Kentucky, USA), Christina, and many others helped Janene in debugging the various characters.

Finally, Figure 10 illustrates the use of the WSU IPA with Computer Modern. This example comes from Claire Lefebvre's article "Instrumental Take-Serial Constructions in Haitian and in Fon".

How to use the IPA

To use the IPA you must first embed a collection of macros by entering

```
\input ipamacs
```

Each of the IPA characters has a control sequence defined in `IPAMACS`. The names chosen for these control sequences were based in large part on the standard names used in Pullam and Ladusaw. For example,

```
A shibilant is a term occasionally
found for a fricative
corresponding to
a ``hushing'' sound, e.g., IPA
[\esh] (more technically,
a grooved laminal fricative).
```

yields the illustration in Figure 3.

The IPA character macros are defined so they can be used with the Computer Modern characters and accents without the need for delimiting curly braces. By using the IPA definitions, you can use the Computer Modern accents with an IPA character in the same way you would accent a Computer Modern character. For example, Figure 4 was created by entering:

```
The superscript tilde is a
nasalization marker for vowels,
thus [\~\scripta] is a
nasalized [\scripta].
```

The superscript tilde is a nasalization marker for vowels, thus [ɑ̃] is a nasalized [ɑ].

Figure 4: Accenting IPA characters

To use the IPA accents with Computer Modern characters or with another IPA character, you can define the IPA accent in the same way that the Computer Modern accents are defined. For example, an "over-ring" accent could be defined as,

```
\def\or#1{{\edef\next{\the\font}%
        \ipatenrm\accent"78\next#1}}
```

and may be used like,

```
The over-ring may be used over
letters with descenders as an
alternative to under-ring to
indicate devoicing, e.g. [\or g].
```

which will print as illustrated in Figure 5.

If the spacing or placement of the diacritic is not exactly what you desire with the basic definition, you can add kerns where needed. Or in some cases, you may want to space differently depending on what character is being accented. Perhaps your scheme for accenting characters with an "undercircle" requires less space between the undercircle and the character, and if the character to be accented is an "r" a completely different scheme would be used. For example, Figure 6 was created by entering:

```
\def\undercirc#1{\ifx#1r
      \oalign{#1\crcr\hidewidth
            \kern.24em\underring
            \hidewidth\crcr}
      \else\oalign{#1\crcr
            \hidewidth
            \raise.1ex\hbox{\underring}
            \hidewidth}}\fi}
A voiceless trilled r [\undercirc r] in
certain Scottish dialects...
```

The over-ring may be used over letters with descenders as an alternative to underring to indicate devoicing, e.g. [g̊].

Figure 5: Using IPA accents

A voiceless trilled r [r̥] in certain Scottish dialects

Figure 6: Using IPA accents

Finally, Figure 7 illustrates using the IPA. The native American Makah script at the top of the figure appears on a plaque at an archeological site in the northwest corner of the United States. The translation for the Makah script is at the bottom of the figure.

Where to go from here?

The fonts are in roman, bold extended and slanted faces. And they come in 8, 9, 10, 11, 12 and 17 point sizes. The project is completed and stable. Figure 8 illustrates the full character set.

The question remains, where do we go from here? Certainly, the WSU IPA is not an exhaustive collection of phonetic IPA characters. Pullam's *Guide* lists 78 "major" characters, all of which are contained in the WSU IPA. The *Guide* also contains many "minor" characters, 50 of which have been included in the IPA.

With Janene no longer at at WSU, and no one else willing to carry on the work, no further development on the IPA will be done here. There are more characters that can be added. Anyone willing to do so is welcome to use the existing base characters as examples. The arrival of TEX 3.0 gives us the ability to assign more than 128 characters is available. It might be reasonable to finish the rest of Pullam and Ladusaw's character set. Also, with the addition of Professor Knuth's virtual font, new ways of mixing the IPA characters into existing fonts is possible. If you choose to modify the WSU IPA, I do ask that you call the new font something other than WSU IPA.

How to get the IPA

You can get the IPA on diskette or by anonymous FTP. Also, the IPA will be distributed to the various site coordinators who can determine whether to place it on their distribution.

To obtain the IPA on diskette from Jon Radel, send him a note and ask for the WSU IPA fonts. Jon's address is:

Jon Radel
P.O. Box 2276
Reston, VA
22090-0276

To get the IPA by anonymous FTP, you can connect to the machine BOBCAT.CSC.WSU.EDU. The address is 134.121.1.1. Log on as `ANONYMOUS` and change to the `TEXT1` directory. You will see ten subdirectories which you can GET files from. The one you want is `WSUIPA` which contains the METAFONT source, `tfm` and `300pk` files.

Bibliography

Guenther, Dean, Alan Wittbecker, Janene Winter. "TEX at WSU" *TUGboat*, **5**(1), pages 24–25, 1984.

Lefebvre, Claire. "Instrumental *Take*-Serial Constructions in Haitian and in Fon." *Canadian Journal of Linguistics* 34:319–337. [Example from p. 330.]

Piper, Terry. "On the Difference Between L1 and L2 Acquisition in Phonology." *Canadian Journal of Linguistics* 32:245–259. [Example from p. 255.]

Pullam, Geoffrey K. and William A. Ladusaw. *Phonetic Symbol Guide.* Chicago: The University of Chicago Press, 1986.

Rhodes, Richard. "Métchif — A Second Look." Pp. 287–296 in *Actes du dix-septième congrès des algonquinistes.* William Cowan, ed. Ottawa: Carleton University. [Examples from p. 293.]

Thiele, Christina. "TEX, Linguistics, and Journal Production" *TEXniques* **5**, pages 5–26, 1987.

čabi·yʔakʔu kʷiči·ya· ʔuse·ʔił yeʔitx̌pi·t
ʔakyi·q qʷidiččaʔa·tx̌ yaqa·qey ʔadic̓aƛo·wis
ƛ̓icu·xʷadi· ʔiyax̌ q̓ʷiƛi·q̓ʷiƛš ʔuwadił tuk̓ʷitap
hubaqƛ kʷiči·ye·ʔiq ʔu·du·ʔaƛ t̓aččiƛšiƛqa·
tu·k̓ʷi·ƛ qʷisi·ʔaƛ qʷax̌ax̌iƛqey ƛa·dʔok̓ʷap
wik̓aƛ hita·kʷačiƛ huʔaq̓itwi·ʔiq wik̓i·d
ƛaʔu· hidawaƛ̓id yaqa·qey ʔu·ksda·q̓a·ł
huʔaq̓idiq yałax̌ batba·ʔas tu·k̓ʷiƛcki·
du·baqƛił qʷiqʷic̓aqałqey hu·ʔas qʷabitqey
qʷiyowis q̓ʷiƛšiƛ tu·k̓ʷitap batba·ʔasiq.

A major village of the Makah Indian Nation, the Ozette village was, over the past millennium, periodically covered by mud flows. The rapid burial created conditions that greatly retarded the deterioration of perishable artifacts. These conditions created a time capsule, unique in archaeology, in which the buried houses and their contents have survived in an almost perfect state of preservation.

Figure 7: American Macah script and translation

	´0	´1	´2	´3	´4	´5	´6	´7	
´00x	ɐ	ɑ	α	ɒ	ʌ	ƀ	b̵	b̸	"0x
´01x	ɓ	β	ȼ	ɕ	ʗ	đ	d̵	d̸	
´02x	ɗ	ɖ	ʤ	ð	D	ə	ɚ	ɘ	"1x
´03x	ɛ	ɜ	ɝ	ɞ	ɡ	ɠ	ɢ	ɣ	
´04x	ɤ	ɤ	hʋ	ħ	ɦ	ɧ	ɥ	ɨ	"2x
´05x	ɨ	ɩ	ɪ	ᵻ	ʝ	ƚ	ł	ɫ	
´06x	ɭ	ɮ	λ	ƛ	ɱ	ɯ	ɰ	ɲ	"3x
´07x	ŋ	ɳ	ɴ	ʘ	ɵ	ɔ	ω	ɷ	
´10x	∞	ᵽ	þ	ɸ	ɾ	ɼ	ɽ	ɹ	"4x
´11x	ɻ	ɺ	ʀ	ʁ	ʂ	ʃ	ʆ	σ	
´12x	ʈ	ʧ	ɟ	θ	ʉ	ɥ̸	ʊ	U	"5x
´13x	ᵿ	ʋ	ʍ	χ	ʎ	ʏ	ʑ	ʐ	
´14x	ʒ	ʓ	ʔ	ʕ	ʖ	ˀ	ˁ	˽	"6x
´15x	ˈ	ˌ	ˌ	˥	⊥	⊤	⊣	⊢	
´16x	ˑ	ː	¨	ʿ	˄	˅	<	>	"7x
´17x	˚	˳	˓	˜	˷	ˬ	˛	˰	
	"8	"9	"A	"B	"C	"D	"E	"F	

Figure 8: WSU International Phonetic Alphabet 12pt Roman

children had contact spoke a dialect of Irish-English which permits glottal stop in words such as *little* and *bottle*. Moreover, in connected discourse, word final /t/ is frequently glottalized, particularly if the next word begins with a vowel. It was likely, therefore, that the children were following, in varying degrees, the example of their teacher. All instances of glottal substitution which were acceptable in this teacher's dialect were, therefore, dropped from the analysis. Nevertheless, there remained in the data from these nine Ss, persistent if not widespread glottal replacement which may have resulted from various kinds of overgeneralization. One child, for example, in addition to producing glottals consistent with the Irish-English dialect also produced forms such as [tumʌʔ] 'too much'; [ʌɷʔsaɪ̭ðə] 'outside the'; [wɪʔčəz] 'witches'.

Figure 9: Sample combining `ph10` and `cmr10`

(41) a. Derived LCS resulting from the association of *sɔ́* 'Take' and *ɗó* 'Put':
LCS: [x cause [y undergo a change of location to z]/PUT

b. Kɔ̀kú sɔ́ àsɔ́n ɗó hàsùn mɛ̀
Koku take crab put basket in
'Koku put the crab in the basket.'

Figure 10: Sample combining `wsuipa9`, `wslipa9` and `cmr10`

	0	1	2	3	4	5	6	7
000	ɤ	ɗ	ɟ	ʎ	ʗ	ɓ	ʖ	ɯ
010	Φ	Ψ	Ω	α	β	γ	δ	ε
020	ζ	η	θ	ι	κ	λ	μ	ν
030	ξ	π	ρ	σ	τ	υ	φ	χ
040	ψ	ω	ɜ	ɵ	ϖ	ɹ	ç	ʾ
050	(	)	∗	+	ˏ	−	.	/
060	0	1	2	3	4	5	6	7
070	8	9	:	;	¡	=	ʕ	ʔ
100	ð	A	B	C	D	E	F	G
110	H	I	J	K	L	M	N	O
120	P	Q	R	S	T	U	V	W
130	X	Y	Z	[	“	]	ˆ	˙
140	ɨ	ɐ	ɲ	ɔ	ɖ	ə	ɾ	ŋ
150	ɦ	ɨ	ʄ	ʰ	ɭ	ɱ	ɳ	∞
160	ɒ	ʁ	ɽ	ʂ	ʈ	ʉ	ʌ	ʍ
170	ħ	ɥ	ʐ	ʃ	ʒ	ƶ	⌐	⁀

Figure 11: The ph10 font

```
+
 ...****...****..
 *******.*******.
 ****************
 ...******..*****
 ...*****...*****
 ...*****....***.
 ...*****........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.........
:...****.........
 ...****.........
 ...****.........
 ...****.........
 ...****.....***.
 ...*****...*****
 ....*****..*****
 .....****..*****
 ......*********.
 .......******...
```

Figure 12: The pixels for the "r with right tail"

Practical Halftoning with TEX

Adrian F. Clark
University of Essex
Janet: `alien@essex.ac.uk`

Abstract

The inclusion of photographic material in a book or similar publication is an expensive process and, unless care is taken, artefacts can be introduced into the reproduction. For publications in subjects like image processing, it is essential that the pictures are reproduced both cheaply and accurately. The principle of using TEX to typeset halftoned pictures is now quite familiar, but there are a number of practical problems involved. This paper discusses these problems and compares halftoning with TEX to other techniques such as direct inclusion of PostScript. Some of the advantages and disadvantages of both approaches are described, taking account of the need for low-resolution drafts from laser printers and high-resolution final output from typesetters.

Given the ability to generate halftoned output, it is but a small step to consider including colour pictures in TEX documents. Some attempts at producing colour separations using the above approaches are described.

Introduction

Even with the advent of computer typesetting, one of the most costly aspects of book production is the inclusion of continuous-tone (grey-scale or colour) material. This is because the preparation of such copy is done essentially manually, the figure usually being merged with the text only just before the production of the offset printing masters. However, with a suitably high-resolution output device such as a phototypesetter, it is possible to produce representations of continuous-tone material which are of printable quality. This paper looks at how continuous-tone copy can be inserted into a TEX document (albeit plain TEX or LATEX).

The conventional approach to the reproduction of continuous-tone material, *halftoning*, is briefly described and the three main ways of producing halftoned output from TEX are then discussed and compared. An extension to the basic techniques to prepare colour separations is then described and examples of the output which can be obtained are given. Directions for future work are then considered.

Readers who wish to learn more about conventional and digital halftoning are referred to Southworth and Ulichney-book; the latter requires some familiarity with Fourier analysis. Halftoning with TEX is not a new idea, aspects having been discussed by Simpson, Knuth and Clark. Colour and its reproduction are discussed in great detail by Hunt. The problems in printing computer-originated imagery are discussed by Stone.

Reproducing Continuous-Tone Material

Conventional Halftoning. Printing is an all-or-nothing operation: one either puts ink at a specific position on a page or one doesn't. Hence, to reproduce material with a continuous range of tones (grey levels), one must produce a region of black and white (ink or no ink) dots which the eye blends together to give the appearance of being the appropriate greys. The process by which this is done is known as *halftoning*. This is usually achieved by photographing the original through a *screen,* which consists of a pair of glass plates, on which fine opaque lines have been ruled, cemented together at right angles to each other, leaving a structure of square interstices. When photographed, the interstices act as crude pinhole "lenses," forming out-of-focus "images" of the camera lens on the plate. A high-contrast film is used in the photographic process, so that the "screened" picture consists of only light and dark dots. Light areas in the original produce larger areas in the (negative) screened image and hence smaller black dots when printed; the converse is true for dark areas of the original image.

The quality of reproduction is governed primarily by the fineness of the screen, which is normally measured in terms of the number of lines ruled per inch — newspaper photographs typically have screens of about 70 lines/inch, while magazines have screens of about 130 lines/inch. For top-quality printing, screens of up to 300 lines/inch are used.

Digital Halftoning. It will be apparent from the above that the binary nature of halftoning lends itself well to a digital approach, and there has been a fair amount of work devoted to the topic. At the simplest level, one can assign a little region of dots to generate a single pixel of the image — the more dots set to black, the darker the pixel; but this leads to a very objectionable patterned effect in regions of constant grey value. Practical techniques do not require several binary dots to represent a single pixel, and attempt to remove the patterning effects; these are known as *dithering* algorithms.

The algorithm known as *ordered dither* is closest to conventional, photographic halftoning since it passes high-contrast, high-frequency detail into the halftoned image. A mask (typically 8×8 pixels in size) of fixed thresholds is tesselated over the image, with the output at each pixel being white if the pixel holds a greater value than the corresponding threshold in the mask and black otherwise. Unfortunately, ordered dither also produces objectionable artifacts in regions of constant grey level; this is typically reduced by adding a little high-frequency noise. As an aside, if one uses a randomly-chosen threshold instead of a mask of fixed values, one obtains an image which resembles a mezzotint.

The most popular dithering technique was developed by Floyd and Steinberg, and is known as *error diffusion.* This compares each pixel with a fixed threshold as for ordered dither. However, the threshold value is then compared with the value of the pixel and the difference (the "error") used to generate a correction factor to be added to future pixels. (In signal processing jargon, this is a "recursive filter.") The correction factor is calculated from the error according to the weighted values of neighbouring pixels, and some interest has centred around the best weights to use — see Ulichney for details.

Halftoning and TeX

There are three basic approaches to the inclusion of halftoned material in TeX documents:

1. generating a "picture" font,
2. building a halftone font, then typesetting the picture with TeX,
3. including the halftone material at the dvi-processing stage via \specials.

We shall look at each of these methods in turn.

Generating a "Picture" Font. In this approach, the entire picture is halftoned and converted to a pk file, which allows TeX to typeset it as a single (though large) character. The conversion may be direct from image to pk file, in which case a tfm file must also be generated, or via METAFONT. However, the run-time overhead of using METAFONT in this way is a significant one.

Having generated such a "picture" font (picture.tfm and picture.pk), one would insert the image into a floating figure in a plain TeX document with a series of commands like:

```
\font\pix=picture

\midinsert
   \centerline{{\pix\char0}}
   \smallskip
   \centerline{Picture inclusion using
               a picture font}
\endinsert
```

For LaTeX, the approach is similar but the syntax a little different:

```
\newfont{\pix}{picture}

\begin{figure}
   \centering
   {\pix\char0}
   \caption{Picture inclusion using a
               picture font}
\end{figure}
```

One problem with this approach is that some output devices — notably Digital's LN03 laser printer — have a limit to the height of a glyph which is smaller than the typical size of a complete picture. In this case, one must ensure that the picture is partitioned into a set of smaller regions.

Using a Halftone Font. An alternative approach is to devise an entire font which maps input characters onto output greys, then use TeX's boxes-and-glue approach to typesetting to construct the picture from the characters. This approach was also outlined by Knuth and popularised by this author.

```
\hbox{\vbox{\halftone\offinterlineskip % machine-generated by TEXPIC.
\def\BHT{\hbox\bgroup\ignorespaces}
\catcode`\^=12 \catcode`\_=12 \catcode`\.=\active \let.=\egroup
\catcode`\,=\active \let,=\BHT \catcode`\/=0 \catcode`\\=12
,abe___cbaadgljkigbeehe``ae_FB?=D.
,abe_^`caaddfkjic[JSbiea_]iNFA@?E.
,ace_^`baaa`dkj?DFF?60Qc_`gEH@A?F.
,bcd_^`bba`_ccAEDHOFMIBB^f[EJ@CDJ.
,acd___baa``UB>FCEMPTUQKMiMMQIOVZ.
,acd__`ba``ZUTGIQKX]^_ZLAd?CHCUUW.
,bcd_^`aa`_`YSR\deghojgTFO?CHBRPU.
,bcd_^`ba`^]_dbfjmhhmnifNH?DJNONS.
,bcd_^`ba`]]djikkmkid[aXZGAHWMQRT.
,bcd__aba`_Z[f^\[djih[[U]FDTWNVSU.
,bcd__aba`_hib\affddf]f[UFKTTR[UT.
,bdc__abba`jmkgjhillh`fafQSXXXXVU.
,bed`_abba`f[QRVccci jjgfdELUWVVXZ.
,bdd``bcbbaakiXXcjiidjigVCCWVVXVX.
,bdd``bccbbaV_LZS_c^aefaECBNWUWUX.
,bedaabcccbbJDL\LKQ\eidJDDEPWXWYV.
,cedaacccbcbP[UbUNTa[\SKFCBKTWVVX.
,cedbacccccc^aQTX]_dZTPRSHCMRXSUW.
,ceeaacddcbeeVXMdXad[RUVQLKUU[\VT.
,cgfbbddddcXe]UXZ\cceOKMQBCOW\^WZ.
,dhgddffefdiinVT\chhfOLMOBIMNNNLJ.
,ejifeefffemhlfYmkhggWRPNDRSRQOLN.
,\dihggkkkilk`ceUW_^[jfc`LSTSQPMM.
,aZ`cecgijaaaadhkhhhhifdfhkhRTVWT.
,gii]deiW]baabhdgmffeefefgeVRUXSU.
,cflg`fV`bcbcdhfgeheefhiknj_SBXQR.
,bclidb]`gdbfgfjiihhkijloggfd]R^].
,kkloo]^faffbeghkklmenmmgghhhj_aW.
,pnkop_adkgcdb]^fll^WA9PgghgijiWU.
,pmklpa_dkkjgba_dmj`QE1>?dhgnikbS.
,mllljb`bkkigijhehlORER??>kkokmjV.
,nmlmfcabkmnhiebeFJKMcY>:B:54onlc.
}}%
```

Figure 1: Machine-Generated TEX for Use with a Halftone Font

The halftone font in most widespread use is a variation on the "double-dot" font devised by Knuth and has some 65 grey levels. This maps the ASCII character "0" onto white, "1" onto near-white — and so on, up to "p" which represents black.

With such a font available, it is not difficult to write software which converts an image to a file which is compatible with both plain TEX and LATEX. The only point at which care is needed is in the handling of those characters which lie between "0" and "p" and have a special meaning to TEX, namely "\", "^" and "_". This is illustrated in Figure 1, which is a 32 × 32 representation of the "girl" image ubiquitous in image processing circles.

Including the figure in a plain TEX document is as easy as for the picture font:

```
\font\halftone=htfont

\midinsert
   \centerline{\input girl32}
   \smallskip
   \centerline{Picture inclusion using a
                  halftone font}
\endinsert
```

while, for LATEX, we would use

```
\newfont{\halftone}{htfont}
```

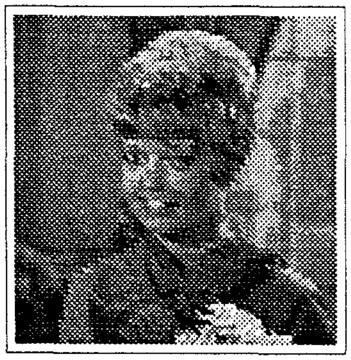


Figure 2: Picture Produced using a Halftone Font on a LaserWriter

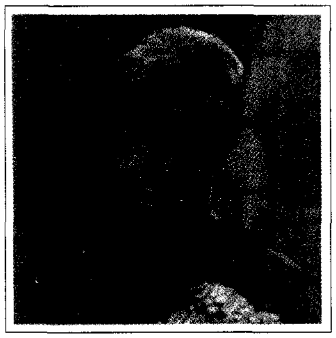


Figure 3: Picture Produced using Halftone Font on a Linotronic 300

```
\begin{figure}
   \centering
   \mbox{\input picture\relax}
   \caption{Picture inclusion using a
                 halftone font}
\end{figure}
```

Note the use of `\relax` here to stop LaTeX from treating the closing brace as part of the filename. In these examples, `girl32.tex` contains the machine-generated picture, while `htfont` is the name of the `tfm` file which contains the halftone font for the chosen output device. The machine-generated TeX input assumes that the halftone font is selected by the command `\halftone`.

To obtain a reasonable number of greys in the font, the size of the printed image must depend on the resolution (number of dots/inch) of the output device. This effect is illustrated in Figure 2 and Figure 3, which are output from an Apple LaserWriter (300 dpi) and a Linotronic 300 typesetter (1270 dpi) respectively: although these are the same physical size, the LaserWriter image contains 64 × 64 pixels and the Linotronic one 256 × 256.

Some care is needed in optimising the halftone font to the output device: the author has found that existing `mode_defs` for printers invariably cause the halftoned image to be too dark, and one must adjust the `blackness` and `fillin` parameters while printing off a test grey scale.

TeX, as originally devised, had a memory limit which is too restrictive for typesetting pictures of 256 × 256 or more pixels. This was one of the factors which led to the production of expanded-memory versions of TeX. Of course, the `dvi` driver (and, indeed, the output device) must also be able to handle this many characters on a single page. Furthermore, the time taken for TeX to typeset a 512 × 512 picture (about quarter of a million characters!) is not insignificant. The great advantage of this method, however, is that pictures are text files (and hence can be moved around by electronic mail) and produce essentially identical output on a variety of output devices.

Incorporation at the DVI-Processing Stage. In this approach, one simply leaves a gap in the TeX output and tells the `dvi` driver to insert a file of printer-specific instructions by means of a `\special`. Until the time comes when `\specials` are standardised, this approach is also specific to the chosen `dvi` driver: for example, the syntax of the `\specials` accepted by one `dvi`-to-PostScript driver are rarely accepted by another.

The most widely-used halftoning device available to most TeX users is probably a PostScript laser printer. Since it is relevant to the following discussion of colour output, a brief summary of the facilities provided by PostScript for image display is called for (see Roth for many practical insights into using PostScript devices for halftoning).

Image data is displayed in PostScript via the `image` operator. This requires information as to the sizes of the image (in pixels) and size (in units of 1/72 inch) of the resulting display. This is followed by the pixels themselves, with each pixel normally being coded as two hexadecimal digits. The pixels are halftoned within the printer according to a notional halftone screen which is defined by three parameters:

1. the screen frequency in lines/inch,
2. the rotation of the screen in degrees from the horizontal,
3. the *spot function*.

The first of these determines the fineness of the screen (within the limitations of the device), while

```
%!PS-Adobe-2.0 EPSF-1.2
%%Creator: PSPIC 0.1
%%CreationDate: 10-AUG-90 11:15:16
%%BoundingBox:     72     72    360    360
%%EndComments
/paperxorigin   72 def        /paperyorigin    72 def
/paperwidth    288 def        /paperheight    288 def
/imagewidth     32 def        /imageheight     32 def
/picstr imagewidth string def
/plotpic { gsave
  imagewidth imageheight 8
  [imagewidth 0 0 imageheight neg 0 imageheight]
  {currentfile picstr readhexstring pop} image grestore } def
paperxorigin paperyorigin translate
paperwidth paperheight scale
plotpic
3A382D43454235383A3C2E241218131E25392B2E1F2C3E3F3D2B44A6B8C2CAAE
3B372D424741363A3A31312613181B32549872371C2B3C434D1B88A[illegible]
3B352D43463E373A3C3C4130131AC4AFA8A9C4E8FF7[illegible]
3A342F43464036393C3F423634BCAA[illegible]886626A646A5E
3B342F444542363A[illegible]2B1D2E97AFAFAD7E6361655E68
3B34[illegible]6B396C876F3C554F7494A6B3B5946E65696661
[illegible]2323235354A3B7B6E5F4C442F57727F77749EB58D795E756A62
342C2E3A3C33313234362D2D695F8C325E3D3055796A687A8F936B6D5350696E
32252A363A302F3130345E2A4C6D5F575034322D82928C7CB6B384634E476359
3120233131292A2C2A311C1E09667050321E1F26838E8B83B99D8D8988888E96
2D191C292C2B2728262D0D2111285D0D151F232462777E86AF7675797A839189
512F1A2025221416151C111341342D6D6245495316283340 8F7471757B808B8D
3D5840322C32231D193A3E3D3B311F1621211E221C262E2720141F776F66646F
241C1A4C312C1D644C363A3B391F31230B26292B2D292B28222A67796B60756A
352611233F266641383337342E1E29242B222A2D27201E1408194275B9617B77
3932101B2F374C40232F38292427171A1D1F20151B1A1205242328304C78494A
13120F04064C47283D2628372C22201316110A2D070C0B22232221211A463C63
0008130501453D301625332F384A47290E124865BCDB7E232220221A181C656A
010D1311013B443012151925383D43310D163E7BADFAC7C5322123071D143974
0A0F111219393E3612121A221E1A1E2D200E8579AD77C2C1C6131505150B1768
0A0C110D29353D39140D09221D2B372CA798928C325BC7D8B8D6EBEE06070E32
showpage
%End of Picture
```

Figure 4: A Machine-Generated PostScript Image

the second is normally set to 45°for monochrome pictures. The spot function determines the shape of the halftone cell; for a modern LaserWriter, this defaults to,

```
{ abs exch 2 copy add 1 gt
   { 1 sub dup mul exch  1 sub dup mul
      add 1 sub }
   { dup mul exch dup mul add 1 exch sub }
ifelse }
```

which, for grey levels less than 128, builds a circular white cell in a black background and, for grey levels greater than 128, black cells in a white background. We can see that this is essentially a direct digital implementation of the photographic technique.

Generating a file of PostScript (or, more precisely, *encapsulated* PostScript) is as easy as generating the TEX input for use with a halftone font (Figure 4). As we can see, the major difference between TEX and PostScript is in the representation of the image data. Each pixel is represented as a character in TEX, while in PostScript it is encoded in hexadecimal. This reflects PostScript's ability (in principle, if not in practice) to display 256

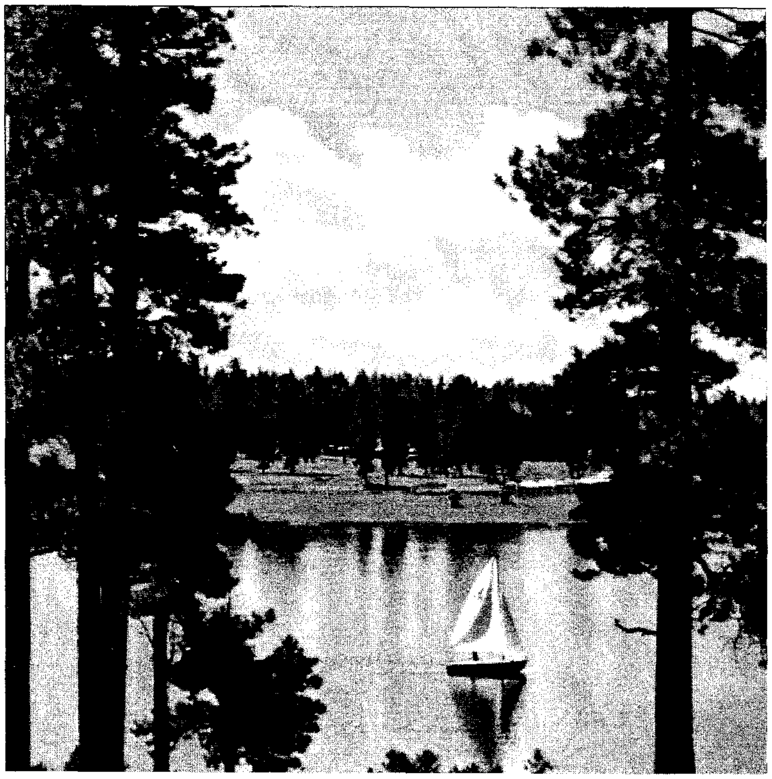


Figure 5: 512 × 512 Image Output via PostScript

different grey values. The output from a typesetter of such an image is certainly good enough to be used for camera-ready copy (Figure 5). The means by which the PostScript picture is inserted into a TeX document depends on the `\specials` offered by the `dvi` driver; for James Clark's `DVItoPS`, the appropriate plain TeX syntax is,

```
\midinsert
  \def\picht{4in}
  \special{dvitops: import boat.ps
      \the\hsize \picht}
  \smallskip
  \centerline{$512 \times 512$ Image
      Output via PostScript}
\endinsert
```

where `\picht` gives the height of the desired figure and `\the\hsize` its width. For LaTeX, the syntax is:

```
\begin{figure}
   \def\picht{4in}
   \vspace{\picht}
   \special{dvitops: import boat.ps
           \the\textwidth \picht}
   \caption{$512 \times 512$
         Image Output via PostScript}
\end{figure}
```

where `\the\textwidth` is the width of the figure. In both cases, `DVItoPS` ensures that the aspect ratio of the image is retained, centring the image in the text as necessary. For other `dvi`-to-PostScript drivers, the `psfig` macros can be used to achieve equivalent results.

Comparing the Approaches. The main features of the methods considered here are compared in Table 1. In terms of the quality of the rendition, there is little to choose between the methods, at least on high-resolution devices. The generation of a picture font definitely requires the largest programming effort, particularly since the fine details of the format of `pk` files differ slightly between implementations (*e.g.,* between VMS and UNIX). The compactness and portability of the halftone font perhaps wins, but the flexibility in being able to print a PostScript image at any size is an important feature. When it is important to avoid patterning in regions of constant grey tone, the generation of a picture font, since it is under program control, probably provides the simplest implementation—although it is possible to devise PostScript spot functions which dither.

picture font	halftone font	PostScript
picture description is specific to output device	picture description is portable	picture description is PostScript-specific
picture font is device-specific	halftone font is device-specific	image description is device-independent
one (TEX) character for the entire picture	one byte/pixel, 64 levels	two bytes/pixel, "256" levels
cannot easily perform geometric effects (*e.g.*, rotation of image)	cannot perform geometric effects	easy to perform geometric effects
easy to use dithering techniques	difficult to use dithering techniques	possible to use dithering techniques
very difficult to optimise rendition of picture	grey-level font must be optimised by tweaking `mode_def`	grey-level rendition may have been optimised by manufacturer
vertical spacing in document automatically taken into account	vertical spacing in document automatically taken into account	vertical spacing in document must be explicitly left
big-memory TEX not required	big-memory TEX required for pictures of reasonable size	big-memory TEX not required

Table 1: Comparison of Halftoning Approaches

Reproducing Colour

We have seen that there are several approaches to including monochrome pictures in TEX documents. This provides the basic means for incorporating *colour* pictures, since colour images are conventionally printed from a set of *colour separations* or layers using different coloured inks. Four coloured inks are conventionally used: the subtractive primaries (cyan, magenta, yellow) and black. There are two reasons for including a black separation: since printing inks are never pure colours, mixing the three primaries produces a dirty brown rather than a true black, so dark regions require "added blackness"; and black ink is substantially cheaper than coloured inks.

Given an image in terms of red, green and blue values for each pixel, which is the most common representation of a colour image, a simple-minded way to derive the separations would be as follows. First, we convert the red, green and blue values for each pixel to the corresponding values in terms of the subtractive primaries:

$$\begin{aligned}\text{cyan} &= 1 - \text{red}\\ \text{magenta} &= 1 - \text{green}\\ \text{yellow} &= 1 - \text{blue}\end{aligned}$$

where the value 1 represents a fully-saturated colour. The black value can then be determined by finding the minimum of the three colours:

$$\text{black} = \min\{\text{cyan}, \text{magenta}, \text{yellow}\}\,.$$

We can then reduce the coloured values by the amount of common blackness:

$$\begin{aligned}\text{cyan} &= \text{cyan} - \text{black}\\ \text{magenta} &= \text{magenta} - \text{black}\\ \text{yellow} &= \text{yellow} - \text{black}\end{aligned}$$

Finally, when the pixel is known to be close to black (*i.e.*, the red, green and blue values are all small), the amount of blackness is increased. (In practice, it is rare for all the black to be removed in this way, since it can cause problems at colour boundaries; a value of about 50% black removal seems to be the most common.)

Having generated the colour separations in this way, it is simple to print them using one of the above techniques. To produce convincing results, however, one must take account of the *orientation* of the halftone screen. To avoid moiré effects when the screens are superimposed, the conventional approach to halftoning employs screens at angles of 0°, 15°, 45°and 75°for yellow, cyan, black and magenta respectively. This is most conveniently implemented via the PostScript screen (see above).

The final problem in terms of colourimetry concerns the purity of the colours of the inks. Magenta ink, for example, is invariably "contaminated" with yellow (*i.e.*, it removes some blue as well as green), and cyan ink is contaminated with magenta. To obtain reasonable colour separations, one must take account of the impurity of colour, a process known as *under colour removal.* We can do this as follows:

$$\begin{aligned}\text{magenta} &= \text{magenta} - f_{\text{Y}}\ \text{yellow}\\ \text{cyan} &= \text{cyan} - f_{\text{m}}\ \text{magenta}\end{aligned}$$

where f_{Y} is the amount of yellow to remove under magenta and f_{m} the amount of magenta to remove under cyan. Typical values for these are 0.5 and 0.3 respectively. This step should, of course, be performed before determining the black separation as above.

Having generated the separations in this way, one can produce print them using one of the techniques outlined above. A set of separations are shown in Figure 6: these are PostScript output, plotted at 300 dots/inch, so that the directions of the halftone screens can clearly be distinguished.

Discussion

The techniques for preparing monochrome pictures are, as the figures show, impressive in result. With access to an output device of suitable resolution (say, 1200 dpi or better), it is possible to achieve halftoned output of publishable quality. However, there is significant variation between output devices, and it is necessary to tune the font or adjust the digital image accordingly. For example, with an identical input file, the Linotronic 300 typesetter at Aston University produces much darker results than an identical machine at the University of London Computer Centre, reflecting a difference in the adjustment of the machines rather than a fundamental disparity.

The colour separations presented above demonstrate the success of the general approach. However, the colour separation algorithm adopted here is too crude: for example, it should really ensure that there is some overlap of the separations in the region of sharp colour boundaries, to avoid unsightly white gaps should there be minor mis-registration when printing. (Making good colour separations digitally is one of the reasons why electronic prepress equipment is rather expensive!) One must also perform a significant degree of tuning of the colour conversion process to the target output device, much more so than for monochrome output.

With regard to the incorporation of halftoned output into TEX documents, we have already seen that there is a set of solutions to the problem for monochrome output. However, it does not seem possible to provide as neat a solution to the plotting of colour separations. One would like to provide a simple command, let us call it `\colourpic`, which defines the four files required to plot the separations. The obvious way in which `\colourpic` would work would be to plot the black separation in the main document and to generate auxiliary files which gave positioning information for the other three colours. Unfortunately, such information is only available within an `output` routine. (On a PostScript output device, one could alternately make the printer output this information.) This prohibits a solution of the simplicity and generality of that for monochrome pictures. However, easy-to-use macros for incorporating colour separations are under investigation.

Acknowledgements

The high-resolution figures for this paper were plotted on a Linotronic 300 made available to the author by Peter Abbott of the Aston University Computing Service. The author would like to thank Peter Wade of the Essex University Printing Service and Richard Kirk of Crosfield Electronics for helpful discussions.

References

[1] Adrian F. Clark. Halftone output from TEX. *TUGboat*, 8(3):270–275, November 1987.

[2] Malcolm Clark, editor. *TEX Applications, Uses, Methods*, chapter 28: Nontraditional Uses of METAFONT (by Richard O. Simpson). Ellis Horwood, Chichester, Sussex, UK, 1990 (*Proceedings of the TEXeter88 Conference*).

[3] R. W. Floyd and L. Steinberg. An adaptive algorithm for spatial greyscale. *Proc. SID*, 17(2):75–77.

[4] R. W. G. Hunt. *The Reproduction of Colour.* Fountain Press, London, UK, 3rd edition, 1975.

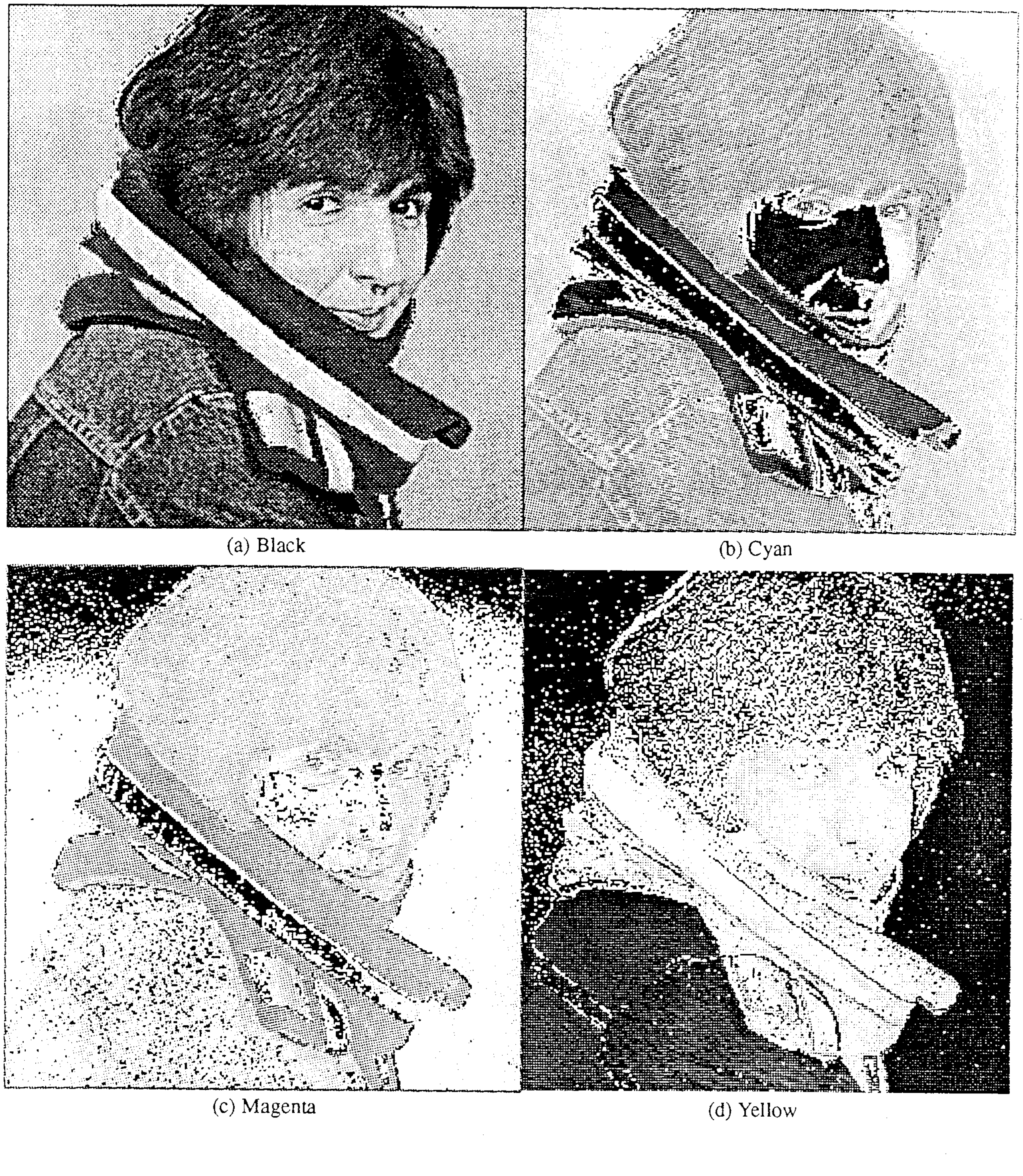
[5] Donald E. Knuth. Fonts for digital halftones. *TUGboat*, 8(2):135–160, July 1987.

[6] Stephen F. Roth, editor. *Real World PostScript.* Addison-Wesley, Reading, Massachusetts, USA, 1988.

[7] M. Southworth. *Colour Separation Techniques.* Graphic Arts Press, Livonia, Michigan, USA, 2nd edition, 1985.

[8] Maureen C. Stone, William B. Cowan, and John C. Beatty. Color gamut mapping and the printing of digital color images. *ACM Transactions on Graphics*, 7(44):249–292, October 1988.

[9] Robert Ulichney. *Digital Halftoning.* MIT Press, Cambridge, Massachusetts, USA, 1987.



(a) Black (b) Cyan

(c) Magenta (d) Yellow

Figure 6: Colour Separations Plotted via PostScript

Some TEX Manuals

Angela Barden
City of Cork, Vocational Education Committee, Ireland
email: stph8002@vax1.ucc.ie

Abstract

Looking for the perfect TEX manual. How to reduce the frustration of manual readers and make manuals user friendly. This paper reviews a number of the manuals available to teach TEX. Weaknesses of current manuals and recommendations for future or budding manual writers are included.

Introduction

In this first section, I would like to outline my credentials in relation to presenting a paper such as this to such a gathering of TEXperts. I am too old to have been educated during the latest technological revolution, but I do have one of the pre-revolution skills, that is, an ability to type. While not being antagonistic to the new technology, I am cautious enough in my approach to take a fairly accurate count of the personal cost required to acquire the relevant skills. The obvious major cost is that of time, but that can be balanced against the three reasons which teachers in Ireland give for belonging to the teaching professions — June, July and August. Last summer, through helping a friend to type a manuscript, I learnt the rudiments of word processing using PC-Write . By October, I had forgotten most of what I had learnt. So I had no problem with attempting to learn a new forgettable skill this summer. But I do remember clearly the exasperation I experienced when using the PC-Write Manual. I fully expected more of the same this summer.

During the past year, I had to write a paper on a number of new textbooks for a new syllabus for English which has been introduced to Irish schools. As the TEX syllabus is not as familiar to me as the Junior Certificate English syllabus, I read "Mathematical Writing" by Donald E. Knuth, Tracy Larrabee and Paul M. Roberts, published by the Mathematical Association of America so that I might gain some familiarity with the concerns of technical writers. Generally speaking, the writers were urged to do the following: Keep the reader uppermost in mind; write for the novice; avoid jokes and jargon; watch out for grammar, etc. I was interested to note that user manuals were called "a blight on the industry," but in general the emphasis was placed on producing clear, comprehensible and, if possible, stylish papers.

In the poem "The Rape of the Lock," the eighteenth century English poet, Alexander Pope, considers the possibility of the lock of hair having landed on the moon, as this is the place where was treasured everything wasted on earth, such as mis-spent time and wealth, broken vows, unanswered prayers, fruitless tears, unfulfilled desires and intentions, etc. Among the treasured items mentioned by Pope are

> The courtier's promises,
>   and sick men's prayers,
> The smiles of harlots,
>   and the tears of heirs,
> Cages for *gnats*,
>   and chains to yoke a flea,
> Dried butterflies,
>   and tomes of casuistry.

My method of alleviating the tedium of re-reading my own prose because of a surfeit of "undefined control sequences" has been to base my essay on Pope's "Essay on Criticism." Linked with this is the question: How many of the manuals should be consigned to the "lunar sphere" to join the gnats' cages, the tomes of casuistry and the time of June, July and August?

Judging Ill?

> 'Tis hard to say, if greater want of skill
> Appear in writing or in judging ill;

Writing well or ill with the reader uppermost in mind is the topic in "Mathematical Writing." Reading well or ill is a professional concern of mine. As a teacher of reading, I have some knowledge of the skills which are taught in order to improve one's reading.

When I am teaching reading skills, I will introduce techniques such as Survey, Question, Read (known as SQR), skimming to sample a complete text, scanning to find specific information, the importance of anticipation, active rather than passive involvement with a text. Obviously the skills used in reading a novel are different from those used in reading a textbook. With a novel, the reader must start at the beginning when the secondary universe is being established, become involved with storyline, and through a process of anticipation, detection and plot revelation proceed to the resolution at the end. With a text book, a range of study skills are brought into play. The good reader is actively involved in the text, surveying and questioning, checking backwards and forwards, comprehending and executing tasks on the basis of that comprehension. Text books are structured so that each insight of the reader can be built on to deal with the next problem. A manual, according to the Shorter Oxford English Dictionary, should be a "a small book for handy use." So what does a reader expect from a manual? I have no general answer, but this reader will look first of all at the index, then read the preface and the introduction, speculate about the dedication, and sample the first chapter carefully. Both surveying and questioning are involved here. I have surveyed the structure of the book, and I have questioned the personality of the writer whose style will be an important factor in my motivation throughout the book.

Doubting My Sense

To teach vain wits a science little known,
T' admire superior sense, and doubt their own.

The amateur word processor, not to mention the amateur typesetter, is governed by a sense of insecurity amounting to paranoia. One relies totally on the good temper of the final line of defense — the computer expert to whom one can speak. In my case, this person was Peter Flynn of the Computer Bureau at the University College Cork. On one occasion, it took me forty five minutes to move from the directory to my file. Eventually, I had to get help. This is the kind of problem that one encounters frequently, yet even the faintest memory of this problem seems to be obliterated entirely by expertise. My apologies to Peter — a "gnat" in an office has a high irritant value to those who work in that office, never mind how kind and patient they all are.

The TEXbook by Donald E. Knuth

Those Rules of old discover'd, not devis'd,
Are Nature still, but Nature methodiz'd;

This book generates a wary approach. The first thing that struck me in the preface was that the author is going to tell lies and that "the later chapters contain more reliable information than the earlier ones do." Whereas, I presume the author is not going to give me straightforwardly incorrect information, perhaps my approach should be to blame him for my typesetting mistakes. Then, we have the issue of *jokes* to enliven the dullness of the material. The problem with the jokes is that I may not appreciate them properly unless I understand the technical point that is being made. Already I feel as thick as two short planks. Then, there is the issue of the dangerous bends signs. These signs begin on page five. Unfortunately, I have been warned against them — "don't read the paragraph unless you need to" is what the Preface says. Am I to understand that the dangerous bends on page five are as much to be skipped as the dangerous bends on page 212? No, I presume not, but the fact is that I have been warned about trouble and being naturally lazy, I will skip the whole lot. Knuth has given me a variety of conflicting instructions — skip this lot, read that lot carefully, and I haven't yet got past the preface.

I found the notion of exercises in this manual (or in any other manual) unbearable. The first exercise is the one to ensure that one knows the correct pronunciation of the word TEX. However, when the answer is consulted, one is thrown into more confusion — what's a TEXacker? I know what a hacker is, but I do not know what an acker is. This may seem like petty quibling, but as the only solid information I have about computers is the necessity of great precision in following instructions, then being baffled by the first exercise is, in computerese, seriously unfriendly. Would the author not have been better employed teaching me how to typeset the name? A quick lesson and, if possible, guaranteed success and I would have been on my way as a well-motivated student of TEX.

What was helpful in this book? The comprehensive index was helpful, as was page 340. On this page are the commands for producing the format on page 341. I think that I would have handwritten this paper had I not found these pages. This book is for the serious student and not for the person who wishes to typeset as quickly as possible and have done with it. It must also be said that when I ran into a problem, I was most hopeful that Knuth would provide a solution.

The Joy of TEX by M.D. Spivak

But most by *Numbers* judge a Poet's Song

Pope had rhythm in mind, but I'll apply the line to Dr. Spivak's book. What interesting imagery! The same as that of "The Joy of Sex." My perusal of the index makes me look at the rectangular box that is a computer and wonder if copulation is rioting under that dull beige exterior. As there is only the one box, maybe it should be masturbation. Does this explain the failure of my computer to follow instructions — masturbation has finally blinded something. Really, it is hardly surprising that I am getting no results. Not only is the thing blind, but it is also stuffed to the gills with all these starters, main courses, sauces and pickles eaten in "sophisticated positions." In this context, I refrain from mentioning the dedication and hasten to do the first exercise, which I am told I skip at my peril.

The first exercise seemed somewhat pointless, but I did it in accordance with my sampling procedure. The exercise was: How many possible meanings does E loves Em only for Eir body? 1. He loves him only for his body. 2. He loves him only for her body. 3. He loves her only for his body. 4. He loves her only for her body. 5. She loves him only for his body. 6. She loves her only for his body. 7. She loves her only for her body. 8. She loves her only for his body. 9. A person of unidentified sex loves him for his body. 10. A person of unidentified sex loves another person of unidentified sex only for the body of that person and so on. It seems to me there are twenty seven possible meanings for this sentence. The answer in the back of the book is one, two, four or eight, depending. At this stage, I became somewhat irritated, and I won't waste any more of my time doing pointless exercises. Furthermore, I feel doubtful about continuing to expose my obvious incompetence as

A fool might once himself alone expose,
Now one in verse makes many more in prose.

A criticism of this book is that I was unable to find a definition of the word "macro." This word is used on the front cover, so I looked it up in the index. There is no entry for macro in the index, and it is not explained either in the preface or the introduction. Yet, I learn from the acknowledgements that the author was less than kindly disposed towards computers some years previously. I was able to find a definition in Knuth, and Leslie Lamport very kindly explained what a macro is on the back of his book. Apart altogether from the problems outlined above, my sampling of this book is extremely dubious. The reason for this is that I can hardly count, and this book is designed to help mathematicians. Maybe mathematicians will be helped, and I sincerely hope so.

PCTEX by Michael Spivak

But of the two, less dang'rous is th' offence
To tire our patience than mislead our sense.

There is neither preface nor introduction, nothing written on the back cover of the book, not even a dedication. There is a title page. The pagination begins again for each new chapter. There is an Appendix A and an Appendix F. In Appendix F, there is a Fabulous Festive Font Finale which contains More Marks, Magically Magnified and Methodically Marshalled. To this feast of alliteration, there is an F word missing which I would dearly like to add. The index is on the page after 7-14, but Appendix A also has an index which begins on the page after A-58. Therefore, there are endless possibilities for confusion. The table of contents is on page iii from which one can deduce the structure of the manual. Black tabs mark the beginning of each new chapter. Each chapter also has its own separate table of contents, but one has to turn back a page because these pages have been excluded from the system of tabs. My patience was so tried that I failed to follow the correct instructions for the vanilla style and abandoned the book.

LATEX by Leslie Lamport

Survey the *whole*,
nor seek slight faults to find
Where nature moves,
and rapture warms the mind.

I am so grateful to Leslie Lamport for the straightforward way in which he gives information. I looked at the table of contents, and every nerve ending began to jump when I saw a section titled "How to Avoid Reading This Book." I was charmed when I discovered that that was precisely what the author meant. Nor did he indulge in silly jokes in "The Game of the Name." He told me what the normal pronunciation of TEX is, and accepted that pronunciation is best determined by usage, not fiat. As there were no tedious exercises, one did not have the feeling that vital information was being concealed in the answer to an undone exercise. I liked the no-nonsense way that error messages were deciphered, both for TEX and LATEX. Dr. Lamport seems to understand that clear, concise prose is most helpful to the person whose main concern is other than that of typesetting. His style is such that he is the only writer who could persuade me to read a paragraph of explanation for some of the more arcane areas in TEX.

TEX for the Impatient by Abrahams, Berry and Hargreaves

Avoid Extremes; and shun the fault of such,
Who *still* are pleas'd *too* little or *too* much.

There was not much time available to survey this book, but my overall impression is that this book is not radically different from many of the other books that I looked at. The layout of the book is rather dull and reinforces my notion of manual writers as being an overwhelmingly verbal lot. I was very surprised to read in the biography of Kathryn A. Hargreaves that she is interested in visual arts. Apart from her charming, but too infrequent illustrations, there is no evidence of a visual imagination in this manual. As in the other manuals, the volume of print to be read is enormous. The prose style, though turgid, was not as irritating as in some other manuals. There are a number of examples which are of great assistance. However, the authors have not gone the full distance and made a visual differentiation between the commands and the textual material to be printed. When I was skimming through the book, I felt that I might just as well have been using Knuth to whom they refer frequently for further explanation.

Miscellaneous

Some foreign writers, some our own despise;

The five pages on TEX Basics, produced by Peter Flynn, were as useful to me as any of the manuals. An added bonus was that there was no "intellectual whimsy" to deal with. The command words were laid out clearly and the instructions were comprehensible. But for the fact that I needed a few extra commands, I need never have looked at a manual. I was also quite happy with the tone of *A Gentle Introduction to TEX* by Michael Doob, but there were exercises in it!

Conclusion

A perfect Judge will read each work of Wit
With the same spirit that its author writ:

I am not a perfect judge, and I have been very unkind about many of the manuals that I reviewed. Such criticism is foreign to my training as a remedial teacher where positive feedback is essential for a student in difficulties. I hope that your reaction to my criticism is not so negative that my requests for specific improvements are consigned to the dustbin. The consolation I offer is Pope's

Turn'd Critics next, and
prov'd plain Fools at last.

You may apply that judgment to me.

My first request to the writers of computer manuals is that they would consider avoiding jokes at all costs. Possibly in the body of the text they might be permitted, but in the table of contents they are unforgivable. The earnest amateur will read that joke fifty times in a day and will end up detesting the perpetrator and all "eir" works and pomps. The same applies to alliteration.

Secondly, I would ask that manual writers be more careful about the imagery used. There is something vaguely revolting about the idea of these tokens being read by the eyes, fed through the mouth, down the gullet, to the stomach, into the intestine and then being sicked up again on receiving a new set of instructions. Even more revolting is a consideration of the route after the tokens have reached the intestines.

The third request I have is for a reduction in the amount of print. Show me how, do not tell me how, in laborious detail. In this area, an increased use of graphics would be pleasant. It seems to me that the concentration on print has been so total that the visual appearance of the page has been forgotten. Not one of the books has any colour, no visual symbols have been used, there is not even some shading to indicate the difference between the commands and the textual matter.

Fourthly, I think it would be a good idea if writers decided beforehand whether they are writing a text book or a manual. A combination of the two satisfies neither market. The manual that has not yet been written is the one with a limited number of commands, attractively presented using colour and graphics, and with a severely limited amount of explanation. Such a manual would be adequate for those who have occasional need of the TEX programme and would give the potential TEXnician sufficient confidence to buy another book.

Finally, I think that the most user-friendly approach that manual writers could adopt would be to include many more sample formats. These sample formats should show the computer commands on one side of the page and the end result on the other. Do not use these sample formats for TEX explanations as that makes them unclear. If the manual writer wanted to be really nice about the whole thing, she/he would choose a decent text to typeset. I would far rather read a Shakespearean sonnet over and over again than some piece of futuristic garbage written specifically to show the application of a simple command.

The question at the beginning of this paper was: How many of the manuals should be consigned to the lunar sphere to join the gnats' cages and the tomes of casuistry and the time of June, July and August? In relation to the manuals, I leave that decision to those more expert with computers than myself. In relation to the time, the sooner this paper is submitted, the lighter the load on the moon.

simply not a very didactic tool. There are better programming languages to show basic concepts such as recursion. There is even a language developed for teaching children: LOGO. It enables the teacher to use recursion, or all the other major concepts like functions or procedures, in a very intuitive manner. In the appendix, you will find two programs which accomplish exactly the same thing, one in LOGO and the other one in TEX. It is needless to say which is more useful in a classroom situation.

In the TEX world we have a tool which could be used in that context and which has not yet earned a lot of respect for it's potential didactic use: Literate Programming. I know of one university lecture where the lecture notes are available as `WEB` files to the students, but there is no such thing in schools. And, for me, the important thing behind Literate Programming is that it allows the reader to watch the program designer originate ideas and develop them into working programs (if the programmer does use `WEB` accordingly, of course). As unreadable as Knuth's books might be, his programs in `WEB` are a pleasure to read. The didactic value of *TEX: The Program* must not be underestimated. It is one of the best things given to a reasonably experienced Pascal programmer. And with `Spidery WEB`, we have the tools to use Literate Programming with other languages, too.

The difficulty in programming TEX to do as you want it to also creates another problem: teacher education. Although this should definitely not be one of the prime criterion in selecting educational tools, it still is a problem. To really program TEX is difficult enough. But to be good enough to further teach how to program in TEX is even harder. So we see that TEX is not really a good tool to teach computer science.

TEX in "Classic" Subjects

Maybe we can find a way to use TEX as an application? There are two more places we could try to let TEX sneak into schools: language courses and arts courses.

Language courses. Teachers of language courses teach the students how to represent their ideas on paper, although their main goal still should be to teach them how to verbally express ideas. For this, TEX is overkill. Students should be taught the basic principles of document design, but the emphasis still should be on the content, not on the presentation. If we let them use TEX, they would spend a lot of time on visual presentation and not on the content. This is because of the perfectionism TEX provokes. We already can produce very, very good output with TEX so we also have to clean up the minor glitches and re-edit the input file at least two or three times until the paper looks almost perfect. This takes a lot of time. A researcher might do that, because nobody minds if his paper is finished half an hour later, but in school, half an hour is a lot of time, especially with the tight schedules the schools usually have (in Austria, the units of subjects taught are 50 minutes). So in the time the student spends TEXing and previewing his paper, he might instead be polishing up its content.

In schools, word processors which offer some basic functions to polish up the image of a text are sufficient, even if the lines are not broken as perfectly as if done by TEX. Especially for younger students, it is not so important to have a perfect representation of what the output will finally look like on paper, but it is important that the visual representation can be changed interactively. Even tools like Microsoft Word or WordPerfect are too much because their large number of functions are not really necessary for educational use.

The basic idea behind didactic software should always be: how can I easily show what is important? If we use WordPerfect, students spend most of their time memorizing key combinations. If we use TEX, they would have to use an editor (which takes time to learn) and still remember all the cryptic tokens that TEX uses. A lot of things can be said about TEX, but not that it is either didactic or intuitive. Small word processors like MicroStar from the Borland Turbo Pascal Editor Toolbox are sufficient. (Well, not quite. There are some things it lacks too.) There are pull-down menus, so there is no need to remember the control-alt-cokebottle combinations, and all the functions are easily accessible.

Art courses. Art education does not consider typography and typesetting worthy topics of art education because there are more important things to learn. In my career as a student, I never heard how a good-looking document is produced. I learned how to paint a surreal city; I learned how to see colors, I learned about the ideas of different schools of painters. But I did not learn how to make a document strikingly different, be it due to its very special font or markup. Apart from that, the art teachers I know have some knowledge about calligraphy, which is considered an art-form, but typography is a trade.

TEX in Schools: Just Say No

Konrad Neuwirth
Postfach 646, A-1100 Wien, Austria (Europe)
EARN/Bitnet: `a4422dae@awiuni11`

Abstract

TEX is a very good tool for typesetting. But does it offer anything for schools? The author explains in detail why he thinks that TEX should not go into the schools.

Introduction

After some years of fooling around with TEX and using TEX professionally, the author has started to think about the broader use of TEX. Due to his current affiliation with the Austrian school system (the author is still a student), his main focus in this paper will be his thoughts about using TEX in schools, especially in computer science didactics. This paper was written with the Austrian schools in mind and the state of computer science introduction here. Although Austria is not the only country undergoing the described problems, the author is not trying to set up any globally valid rules.

Current Situation

Before we start to discuss where TEX could come into schools, we might want to consider what should be the goals of the basic introductory courses to computers (and this is all the schools can offer to all of the students). Currently, two main approaches are taken.

Approach One: use the computer only in a special subject. Here, we still have to distinguish between the approaches:

1) Teaching the basic ideas behind applications with didactically meaningful examples or
2) teaching the basic principles of programming.

If the computer is only used in one subject (which will probably be called something like "computer science"), I find the first approach the more desirable one. The students should see the computer as a tool rather than as a toy for some freaks who just sit in their back room and hack up new programs. Here, I see no space for fitting in TEX, which I take as a programming language that pretends to be an application.

The second approach, although widely practiced, is the best way to scare people. I think having knowledge about only one programming language (which could even be Basic for that matter) is worse than knowing nothing about computers. If the first thing a user ever sees are some cryptic symbols which help with almost no work but only create more work, it is a very traumatic experience. It is hard work to convince such a user of the fact that computers can help with his or her standard work. What we will be faced with in this case won't be computer illiteracy but computer hatred. I should point out that TEX could be used in such a situation, but I suggest this is not what a teacher should do.

Approach Two: Integrate the computer into the "classic" subjects. This means that the language course teacher teaches how to use word processors, on-line spelling-checkers and thesauruses (a triple that will revolutionize language courses); math teachers will teach the use of spreadsheets and geometry teachers will teach what CAD systems can do for technical drawing.

This approach is probably the best as it makes students use computers for their real papers in the language courses and not for some texts which are just typed for the sake of learning how to word-process them in a computer course. This approach can, but need not, be combined with an introduction to computing which should start at least one or two years after the students start using computers. They first have to think of the computer as a tool like pen and paper before they can be confronted with the inner workings of computers or programming.

TEX in Computer Science Courses

As noted earlier, I think that TEX can only be presented as a programming language and not as an application program. So, in my mind, this rules out a few things already. But can we use it in the other approaches? The answer is a simple "no". TEX is

Other courses. I see no place where TEX could be used sensibly. For instance, with current math curriculums, I don't know any place where TEX could be used. Neither can it be used as a tool nor an example to illustrate a specific mathematical concept. It might be shown as an example where different "classical" subjects can be integrated (mathematics, computer science and arts). However, this is a threatening idea to current school systems, because then they could not keep up with the very strict separation of the different subjects and would have to go to a more integrated and overall different way of teaching. Although this is done in a very experimental way, in so called project-weeks, this is not yet an accepted way of teaching in a school.

A place where TEX could come in handy. Perhaps some teachers could use TEX to produce material for students, but I think this option would be used by a very small minority. No, there is no place were TEX fits into schools. It is too big, too powerful.

Conclusion

TEX is definitely a good tool for typesetting. I don't want to stop anybody from using it. I like to use it myself, in my spare time. But I think that TEX has nothing to do in schools. Let's keep it in the academic and commercial world.

TEX in schools: just say no.

Bibliography

BYTE 7#8 (August 1982) An all LOGO issue

Harvey, Brian *Computer Science LOGO Style* 3 Volumes. Cambridge, Mass.: MIT Press. (typeset with TEX)

Knuth, Donald E. *The TEXbook*. Reading, Mass.: Addison-Wesley, 1984.

Knuth, Donald E. *TEX: The Program*. Reading, Mass.: Addison-Wesley, 1984

Lovis, D. and Tagg, E.D. *Computers in Education* Proceedings of the IFIP TC 3 Conference ECCE Lausanne 88. Amsterdam 1988: Elsevier Science Publishers

Papert, Seymour *Mindstorms: Children, Computers and Powerful Ideas*, Harvester Press

Appendix

Programming Examples

Compare the following two listings which compute prime numbers. The second one is taken out of *The TEXbook*, the first one was written in LOGO in maybe 5 minutes[1].

The only thing to understand in the LOGO program is the concept of lists: they are like groups in TEX or if you want, like LISP lists, but with brackets instead of parentheses. And FPUT puts a given element into a list at the first position. With this, you should be able to read the program.

```
TO PRIMES :MAX
 PRINT PRIMELIST MAKELIST 2 :MAX
END

TO PRIMELIST :LIST
 IF EMPTYP :LIST [OUTPUT []]
 OUTPUT FPUT FIRST :LIST PRIMELIST REMOVEMULTIPLE (FIRST :LIST) :LIST
END

TO REMOVEMULTIPLE :BASE :LIST
 IF EMPTYP :LIST [OUTPUT []]
 IF (REMAINDER (FIRST :LIST) :BASE) = 0
        [OUTPUT REMOVEMULTIPLE :BASE (BUTFIRST :LIST)]
 OUTPUT FPUT FIRST :LIST REMOVEMULTIPLE :BASE BUTFIRST :LIST
END

TO MAKELIST :START :STOP
 IF :START > :STOP [OUTPUT []]
 OUTPUT FPUT :START MAKELIST (:START + 1) :STOP
END
```

Now, to print the primes up to a given number, you just say PRIMES *number*.
And now, for the same thing in TEX:

```
\newif\ifprime \newif\ifunknown
\newcount\n \newcount\p \newcount\d \newcount\a
\def\primes#1{2,~3% assume that #1 is at least 3
  \n=#1 \advance\n by-2 % n more to go
  \p=5 % odd primes starting with p
  \loop\ifnum\n>0 \printifprime\advance\p by2 \repeat}
\def\printp{, % we will invoke \printp if p is prime
  \ifnum\n=1 and~\fi % this precedes the last value
  \number\p \advance\n by -1 }
\def\printifprime{\testprimality \ifprime\printp\fi}
\def\testprimality{{\d=3 \global\primetrue
  \loop\trialdivision \ifunknown\advance\d by2 \repeat}}
\def\trialdivision{\a=\p \divide\a by\d
  \ifnum\a>\d \unknowntrue\else\unknownfalse\fi
  \multiply\a by\d
  \ifnum\a=\p \global\primefalse\unknownfalse\fi}
```

Called by \primes *number*. Talk about legible programs!

[1] Thanks to Erich Neuwirth for the program, he cooked it up in that amount of time.

Calendar

1991

Feb	19	**TUGboat Volume 12, 1st regular issue:** Deadline for receipt of *technical* manuscripts.
Feb	20 – 22	10th annual meeting, "Deutschsprachige TeX-Interessenten"; DANTE e.V.: 4th meeting, Technical University of Vienna. For information, contact Dr. Hubert Partl (Bitnet: `Z3000PA@AWITUW01`) or DANTE e.V. (Bitnet: `DANTE@DHDURZ1`). (See also page ???.)
Mar	4 – 7	Seybold Seminars '91. Marriott Copley Place Hotel, Boston, Massachusetts. For information, contact Seybold Publications, (213-457-5850).
Mar	19	**TUGboat Volume 12, 1st regular issue:** Deadline for receipt of news items, reports, etc.
Mar	25 – 29	Intensive LaTeX, Northeastern University, Boston, Massachusetts

University of Hawaii at Manoa

Mar	25 – 29	Intensive Beginning/Intermed. TeX
Mar	25 – 29	Advanced TeX/Macro Writing

Apr	2 – 5	RIAO Conference on Intelligent Text and Image Handling, Universidad Autónoma de Barcelona, Spain. For information, contact (in the U.S.) RIAO '91, Center for the Advanced Study of Information Systems, Inc. (CASIS), Ms. M.-T. Maurice, 220 East 72nd Street #10F, New York, NY 10021; (in Europe) CID, 36 bis rue Ballu, F-75009 Paris, France.
May	2	7th NTG meeting, Amsterdam, The Netherlands. For information, contact Gerard van Nes (Bitnet: `vannes@ECN.NL`)

TUG91 Conference
Dedham, Massachusetts (suburban Boston)

Jul	15 – 18	**TUG's 12th Annual Meeting**

Jul	25	**TUGboat Volume 12, Proceedings issue:** Deadline for receipt of news items, reports (tentative).
Aug	11	**TUGboat Volume 12, 2nd regular issue:** Deadline for receipt of *technical* manuscripts (tentative).
Sep	10	**TUGboat Volume 12, 2nd regular issue:** Deadline for receipt of news items, reports (tentative).
Sep	23 – 25	**6th European TeX Conference** Paris, France. For information, contact GUTenberg, 6th European TeX Conference, B.P. 21, 78354 JOUY EN JOSAS cedex, France. Phone: +33 1 34 65 22 32; Fax: +33 1 34 65 20 51; E-mail: `gut@irisa.irisa.fr`. (See also *TUGboat* 11, no. 4, page 667.)
Sep	26	GUTenberg'91 Congress, Paris, France. "Technical and scientific edition". For information, same as 6th European TeX Conference. (See also *TUGboat* 11, no. 4, page 667.)
Sep	30 – Oct 3	Computer Publishing Conference. San Jose Convention Center and Fairmont Hotel, San Jose, California. For information, contact Seybold Publications, (213-457-5850).
Oct	15 – 16	RIDT 91, The second international workshop on raster imaging and digital typography, Boston, Massachusetts. For information, contact Robert A. Morris (`ridt91-request@cs.umb.edu`, or 617-287-6466). (See also *TUGboat* 11, no. 4, page 668.)

For additional information on the events listed above, contact the TUG office (401-751-7760) unless otherwise noted.

Status as of 20 December 1990

1991 TUG Elections Procedure for nominations

Pierre MacKay
Chair, TUG Nominating Committee

The 1991 elections will take place at the annual meeting, to be held in Dedham, Massachusetts, July 15-18, 1991. Elections will be required for all four officers: President, Vice-President, Secretary, and Treasurer.

It is the intention of the Nominating Committee to provide a slate of two candidates for each office. The bylaws provide for additional candidates to be nominated by petition to the Nominating Committee at least 30 days prior to the election, signed by the candidate and two other members in good standing.

In order to permit the slate to be published in the spring issue of *TUGboat*, the Nominating Committee hereby solicits suggestions and nominations for candidates, to be returned as soon as possible, but no later than April 2.

The names of candidates and petitioners should be submitted in the form shown in the TUG membership list or on a *TUGboat* mailing label; for each name, a full mailing address should be provided, and a signature should accompany every name. For candidates, a phone number and e-mail address, if available, would be helpful.

For a candidate whose name is submitted as a suggestion and not in the form of a petition, the Chair of the Nominating Committee will ask the candidate whether s/he will be willing to serve in the office for which the nomination is presented. This procedure can be facilitated by inclusion of a signed statement of consent from the proposed candidate.

It is recognized that past election procedures have not permitted members to vote for officers unless they attend the annual meeting. The bylaws state only that elections must be by secret ballot. Alternate procedures are being examined to determine whether wider participation can be accommodated. Any change in procedures will be announced in *TUGboat* 12, no. 2.

◇ Pierre MacKay
Chair, TUG Nominating Committee
Department of Computer Science, FR-35
University of Washington
Seattle, WA 98195
`MacKay@June.cs.washington.edu`

TeX in Czechoslovakia

Jiří Veselý

When President Václav Havel returned from the United States he reported that Czechoslovakia had again become popular with the Americans. This was one of the reasons why I started to write a report about TeX in this country. I promised to do so about a year ago but at that time I felt that it would hardly be of interest to the (probably mostly American) readers of *TUGboat*. Since that time have we found out that TUG was much more interested in supporting local groups in Eastern and Central Europe. This was one of the motives that encouraged us in our effort to create an independent Czechoslovak organization of TeX fans.

On May 9, 1990, a branch of TUG was registered in Prague. It was the last formal step required to give the group an independent and legal status. What preceded? My account will not necessarily cover all activities, but I hope it will be relatively complete. As far as I know the first practical meeting of TeX by Czechs was realized by mathematicians and physicists during their stays abroad. For a long time it was just something rather interesting but not importable to Czechoslovakia (an implementation of TeX on an abacus is unknown even now). Even a year after opening borders the standard PC AT in Czechoslovakia costs approximately as much as the Czech car "Škoda" and the number of laserprinters in the biggest and oldest Czech University in Prague (Charles University was founded in 1348) is rather small.

When the first PCs appeared in Czechoslovakia about 4 or 5 years ago the way to using TeX was open. At that time a few people in the Mathematical Institute of Charles University in charge of publishing the mathematical journal "Commentationes Mathematicae Universitatis Carolinae"(CMUC) decided to improve the appearance of the journal. We found that the articles which appeared in the Notices of AMS were very helpful and after several consultations with friends from the western universities we decided to replace the old technology by TeXnology. It took a year to acquire two PCs and HP LaserJet Series II and then we could start. We bought PCTeX from Kettler (FRG) to begin the marvelous adventure known already to most of you.

The group of people interested in TeX has grown rapidly. Since mid-1988 seminars have been organized among mathematicians. Also physicists and some linguists started to be active in the TEX field. The endeavor of Jaroslav Nadrchal led to important international contacts and some lecturers of the seminar in Skalský Dvůr reported to TUG about the growing interest in TEX in Eastern European countries. Since the beginning of 1989 CMUC has been prepared exclusively with the help of TEX. Our first contacts with TUG were not quite successful, but that changed: we are receiving the *TUGboat* in exchange for CMUC.

Shortly before the Revolution (the so called "velvet", not "Great", which is used for the Great Red October Revolution) in November 1989 we established a group of TEX users within the Czechoslovak Union of Mathematicians and Physicists, which is our analogy of AMS. Since we want to be open also to people working in other fields, we decided to create an independent organization "Československé sdružení uživatelů TEXu" (CS TUG). It is also open to people and organizations abroad. The address of it is as follows:

CS TUG
c/o MÚ UK
Sokolovská 83
CS-186 00, PRAHA 8
Czechoslovakia

I would like to mention some people who helped us very much: Michael Doob (we translated his text "A Gentle Introduction to TEX" into Czech); Jon Radel, who kindly sent us a lot of public domain TEXware; Barbara Beeton, Malcolm Clark and Hubert Partl.

Since a lot of the Czech letters have accents, automatic hyphenation was a problem; on the other hand, all letters are printable with standard TEX with accents `\'`, `\v`, and `\accent23`. Hyphenation patterns for Czech had already been generated: at first ad hoc after grammar rules by Ladislav Lhotka and then with the help of modified `patgen` by Petr Novák. Don Knuth's decision to modify TEX was appreciated very much. The new 8-bit TEX has already reached Czechoslovakia. Now a new "bilingual version" (Czech — English, not Czecho-Slovak) is available for everybody willing to start on public domain software. Besides, our organization has no "hyphen problems", it is one for the whole Czechoslovak federation.

Two representatives of CS TUG visited the TEX90 conference in Cork and this again would not be possible without the help of TUG. The first general assembly of new independent CS TUG took place on November 10, 1990. We are going to keep contacts with TUG and national TEX groups from other countries. We would like to organize a meeting in Czechoslovakia as a part of the traditional meeting of physicists at Skalský Dvůr. Economic changes make it more difficult, but we hope it will take place in September 1991. In virtue of the agreement from Cork we hope to get several copies of *TUGboat* as a collective member of TUG. A lot of friends from European organizations promised to help us and some have already done so. We already have also a small booklet in Czech on LATEX and another booklet on AMS-TEX is forthcoming. We would be grateful for any public domain TEX software (we already developed some things existing for a long time abroad) and texts in the public domain on various aspects of TEX which we might translate into Czech. I would like to thank our TEX friends and supporters for their help and on behalf of Czech TEX fans I end with the wish to see many of you in Czechoslovakia in the near future.

◇ Jiří Veselý
Sokolovská 83
CS-186 00 Prague 8
Czechoslovakia

Upgrades of sbTEX and sbMF

Upgrades of sbTEX and sbMF for MS-DOS PCs are available by anonymous FTP from

`Yale.YCC.VENUS.EDU`

as `SB34TEX.ZIP` and `SBMF12.ZIP`. These are also available from SIMTEL20. The Aston repository keeps these files in BOO form. Also available from Aston is a provisional MS-DOS version of TEX–XET contained in `SB32XET.BOO`.

– Wayne Sullivan

Editor's note: The program announced below is still under construction. The author has expressed an interest in hearing from anyone who has done or is doing work in this area. The TUG office would also appreciate receiving such information, which will be kept on file as a resource for answering future inquiries.

TeX TALK

T. V. Raman

Abstract

Traditionally books have become accessible to the visually handicapped only after they have been either published in Braille or more recently after they have been recorded in Talking Book format. However not all publications can be made accessible in this manner, and this still means that the choice of reading material available, especially in the technical fields, continues to be severely restricted. The fact that more and more publications are first written by the author using a computer along with the development of voice synthesizer technology presents us with an excellent opportunity of overcoming this problem.

Several advances have been made during the past few years using voice synthesizer technology and optical character recognition culminating in the development of reading machines. These machines can read out printed matter, provided the quality of printing is extremely good. They also suffer the disadvantage that special characters such as those occurring in mathematical texts continue to present a problem. For material published before the time when books and papers were written using the computer, this approach is perhaps the only way out. However the fact that more and more publications are available in electronic form needs to be exploited as there is no point in trying to reconvert something that is already available in computer readable format using optical character recognition.

This of course leads us to the problem of reading TeX source directly using a talking computer. TeX has the advantage of representing all special symbols, using special names that are formed using the letters of the alphabet along with a few extra special characters. Thus the Greek letters, which are ubiquitous in any mathematical text and present severe problems to any reading machine, are already present in the TeX source in a format that can be read out by a voice synthesizer. However we also need to take account of the fact that a TeX source file is adorned with a lot of special characters that can sound extremely confusing if read out by the synthesizer.

We of course have the option of stripping away all these extra characters from the TeX source, but by doing this we would actually lose valuable information that is being given to us for free. There is enough information present in a TeX source file to enable a printer to lay out mathematical formulae, so that information should certainly be useful in transforming the text to a format that would be comparable to what a person reading out the printed text aloud would generate.

Consider the following example:

$\frac{1+\sqrt{5}}{2}$ `(\(\frac{1+\sqrt{5}}{2}\))`

This is what the above expression sounds like when read out by a voice synthesizer:

> backslash left paren backslash frac leftbrace one plus backslash sqrt leftbrace five rightbrace rightbrace leftbrace two rightbrace backslash right paren .

It certainly sounds extremely mysterious! Instead consider the following piece of transformed text:

> The fraction with numerator 1 + square root of 5 and denominator 2 end of fraction.

This certainly compares very well with what a person reading out the printed text resulting from the TeX source would say.

Similarly $A \subset A \cup B$ is transformed to "A is a subset of A union B".

TeX TALK is a program that looks at the TeX source and carries out transformations like the above. The resulting text is much easier to understand when read out by a voice synthesizer. As becomes evident from the introductory paragraphs, the potential presented by this kind of software is enormous. The program as it currently stands is eminently usable but there is certainly a lot more work to be done on it. When complete it will perhaps result in something I have always hoped to achieve, namely to have the same kind of access to published material as anyone else.

◇ T. V. Raman
Center for Applied Mathematics
Cornell University
Ithaca NY 14853–6201

Institutional Members

The Aerospace Corporation, *El Segundo, California*

Air Force Institute of Technology, *Wright-Patterson AFB, Ohio*

American Mathematical Society, *Providence, Rhode Island*

ArborText, Inc., *Ann Arbor, Michigan*

ASCII Corporation, *Tokyo, Japan*

Aston University, *Birmingham, England*

Belgrade University, Faculty of Mathematics, *Belgrade, Yugoslavia*

Brookhaven National Laboratory, *Upton, New York*

CERN, *Geneva, Switzerland*

Brown University, *Providence, Rhode Island*

California Institute of Technology, *Pasadena, California*

Calvin College, *Grand Rapids, Michigan*

Carleton University, *Ottawa, Ontario, Canada*

Carnegie Mellon University, *Pittsburgh, Pennsylvania*

Centre Inter-Régional de Calcul Électronique, CNRS, *Orsay, France*

College of William & Mary, Department of Computer Science, *Williamsburg, Virginia*

Communications Security Establishment, Department of National Defence, *Ottawa, Ontario, Canada*

Construcciones Aeronauticas, S.A., CAE-Division de Proyectos, *Madrid, Spain*

DECUS, Electronic Publishing Special Interest Group, *Marlboro, Massachusetts*

Department of National Defence, *Ottawa, Ontario, Canada*

Digital Equipment Corporation, *Nashua, New Hampshire*

Edinboro University of Pennsylvania, *Edinboro, Pennsylvania*

Elsevier Science Publishers B.V., *Amsterdam, The Netherlands*

Emerson Electric Company, *St. Louis, Missouri*

European Southern Observatory, *Garching bei München, Federal Republic of Germany*

Fermi National Accelerator Laboratory, *Batavia, Illinois*

Fordham University, *Bronx, New York*

General Motors Research Laboratories, *Warren, Michigan*

Geophysical Company of Norway A/S, *Stavanger, Norway*

GKSS, Forschungszentrum Geesthacht GmbH, *Geesthacht, Federal Republic of Germany*

Grinnell College, Computer Services, *Grinnell, Iowa*

GTE Laboratories, *Waltham, Massachusetts*

Harvard University, Computer Services, *Cambridge, Massachusetts*

Hatfield Polytechnic, Computer Centre, *Herts, England*

Hewlett-Packard Co., *Boise, Idaho*

Hughes Aircraft Company, Space Communications Division, *Los Angeles, California*

Hungarian Academy of Sciences, Computer and Automation Institute, *Budapest, Hungary*

IBM Corporation, Scientific Center, *Palo Alto, California*

Institute for Advanced Study, *Princeton, New Jersey*

Institute for Defense Analyses, Communications Research Division, *Princeton, New Jersey*

Intevep S. A., *Caracas, Venezuela*

Iowa State University, *Ames, Iowa*

The Library of Congress, *Washington D.C.*

Los Alamos National Laboratory, University of California, *Los Alamos, New Mexico*

Louisiana State University, *Baton Rouge, Louisiana*

MacroSoft, *Warsaw, Poland*

Marquette University, Department of Mathematics, Statistics and Computer Science, *Milwaukee, Wisconsin*

Massachusetts Institute of Technology, Artificial Intelligence Laboratory, *Cambridge, Massachusetts*

Mathematical Reviews, American Mathematical Society, *Ann Arbor, Michigan*

Max Planck Institut für Mathematik, *Bonn, Federal Republic of Germany*

Max Planck Institute Stuttgart, *Stuttgart, Federal Republic of Germany*

McGill University, *Montréal, Québec, Canada*

Michigan State University, Mathematics Department, *East Lansing, Michigan*

NASA Goddard Space Flight Center, *Greenbelt, Maryland*

National Cancer Institute, *Frederick, Maryland*

National Research Council Canada, Computation Centre, *Ottawa, Ontario, Canada*

Naval Postgraduate School,
Monterey, California

New York University,
Academic Computing Facility,
New York, New York

Nippon Telegraph &
Telephone Corporation,
Software Laboratories,
Tokyo, Japan

Northeastern University,
Academic Computing Services,
Boston, Massachusetts

Norwegian Pulp & Paper
Research Institute,
Oslo, Norway

The Open University,
Academic Computing Services,
Milton Keynes, England

Pennsylvania State University,
Computation Center,
University Park, Pennsylvania

Personal TEX, Incorporated,
Mill Valley, California

Princeton University,
Princeton, New Jersey

Promis Systems Corporation,
Toronto, Ontario, Canada

Peter Isaacson Publications,
Victoria, Australia

Purdue University,
West Lafayette, Indiana

Queens College,
Flushing, New York

RE/SPEC, Inc.,
Rapid City, South Dakota

Rice University,
Department of Computer Science,
Houston, Texas

Rogaland University,
Stavanger, Norway

Ruhr Universität Bochum,
Rechenzentrum,
Bochum, Federal Republic of Germany

Rutgers University, Hill Center,
Piscataway, New Jersey

St. Albans School,
Mount St. Alban, Washington, D.C.

Sandia National Laboratories,
Albuquerque, New Mexico

SAS Institute,
Cary, North Carolina

I. P. Sharp Associates,
Palo Alto, California

Smithsonian Astrophysical
Observatory, Computation Facility,
Cambridge, Massachusetts

Software Research Associates,
Tokyo, Japan

Sony Corporation,
Atsugi, Japan

Space Telescope Science Institute,
Baltimore, Maryland

Springer-Verlag,
Heidelberg, Federal Republic of Germany

Stanford Linear
Accelerator Center (SLAC),
Stanford, California

Stanford University,
Computer Science Department,
Stanford, California

Syracuse University,
Syracuse, New York

Talaris Systems, Inc.,
San Diego, California

TECOGRAF Software,
Milan, Italy

Texas A & M University,
Department of Computer Science,
College Station, Texas

Texcel, *Oslo, Norway*

TRW, Inc., *Redondo Beach, California*

Tufts University,
Medford, Massachusetts

TV Guide, *Radnor, Pennsylvania*

TYX Corporation,
Reston, Virginia

UNI-C, *Aarhus, Denmark*

Universidad Sevilla,
Sevilla, Spain

Universidade de Coimbra,
Coimbra, Portugal

Università degli Studi Milano,
Istituto di Cibernetica,
Milan, Italy

University College,
Cork, Ireland

University of Alabama,
Tuscaloosa, Alabama

University of British Columbia,
Computing Centre,
Vancouver, British Columbia, Canada

University of British Columbia,
Mathematics Department,
Vancouver, British Columbia, Canada

University of Calgary,
Calgary, Alberta, Canada

University of California,
Division of Library Automation,
Oakland, California

University of California, Berkeley,
Space Astrophysics Group,
Berkeley, California

University of California, Irvine,
Department of Mathematics,
Irvine, California

University of California, Irvine,
Information & Computer Science,
Irvine, California

University of California,
Los Angeles, Computer
Science Department Archives,
Los Angeles, California

University of California,
San Diego, *La Jolla, California*

University of Canterbury,
Christchurch, New Zealand

University of Chicago,
Computing Organizations,
Chicago, Illinois

University of Chicago,
Chicago, Illinois

University of Crete,
Institute of Computer Science,
Heraklio, Crete, Greece

University of Delaware,
Newark, Delaware

University of Exeter,
Computer Unit,
Exeter, Devon, England

University of Glasgow,
Department of Computing Science,
Glasgow, Scotland

University of Groningen,
Groningen, The Netherlands

University of Illinois at Chicago,
Computer Center,
Chicago, Illinois

University of Kansas,
Academic Computing Services,
Lawrence, Kansas

University of Maryland,
Department of Computer Science,
College Park, Maryland

University of Maryland
at College Park,
Computer Science Center,
College Park, Maryland

University of Massachusetts,
Amherst, Massachusetts

University of Oslo,
Institute of Informatics,
Blindern, Oslo, Norway

University of Oslo,
Institute of Mathematics,
Blindern, Oslo, Norway

University of Ottawa,
Ottawa, Ontario, Canada

University of Salford,
Salford, England

University of Southern California,
Information Sciences Institute,
Marina del Rey, California

University of Stockholm,
Department of Mathematics,
Stockholm, Sweden

University of Texas at Austin,
Austin, Texas

University of Vermont,
Burlington, Vermont

University of Washington,
Department of Computer Science,
Seattle, Washington

University of Western Australia,
Regional Computing Centre,
Nedlands, Australia

University of Wisconsin,
Academic Computing Center,
Madison, Wisconsin

Uppsala University,
Uppsala, Sweden

USDA Forest Service,
Washington, D.C.

Vereinigte Aluminium-Werke AG,
Bonn, Federal Republic of Germany

Villanova University,
Villanova, Pennsylvania

Vrije Universiteit,
Amsterdam, The Netherlands

Washington State University,
Pullman, Washington

Widener University,
Computing Services,
Chester, Pennsylvania

John Wiley & Sons, Incorporated,
New York, New York

Worcester Polytechnic Institute,
Worcester, Massachusetts

Yale University, Computer Center,
New Haven, Connecticut

Yale University,
Department of Computer Science,
New Haven, Connecticut

Each Institutional Member is entitled to:

- designate up to 7, 12 or 30 individuals to receive TUGboat subscriptions, depending on category of membership chosen; named individuals will be accorded full status as individual TUG members;
- reduced rates for TUG meetings/courses for *all* staff members, and for rental/purchase of videotapes;
- be acknowledged in every issue of TUGboat published during the membership year.

Instructions: Attach a list of the names and addresses of individuals to whom you would like TUGboat subscriptions mailed, to include answers to the questions on both side of this form–as approrpiate, in particular those regarding the status of TEX and the computer(s)/operating system(s) on which it runs or is being installed. (For IBM and VAX, especially, the operating system is more relevant than model.) It would be particularly useful if you could provide this information as it relates to each individual or group using the same hardware. Please make as many copies of this form as needed or contact the TUG office for additional copies.

- *Send completed form with remittance* (checks, money orders, UNESCO coupons) to:
  TEX Users Group
  P. O. Box 594
  Providence, Rhode Island 02901, U.S.A.
- *For foreign bank transfers* direct payment to the TEX Users Group, account #002-031375, at:
  Rhode Island Hospital Trust National Bank
  One Hospital Trust Plaza
  Providence, Rhode Island 02903-2449, U.S.A.
- *General correspondence* about TUG should be addressed to:
  TEX Users Group
  P. O. Box 9506
  Providence, Rhode Island 02940-9506, U.S.A.

Institution/Organization: ________________ **Principal contact:** ________________

Phone: ________

Qty	1991 Institutional Membership (Jan.–Dec.)												Amount
	Category A (incl. 7 subs.): educational $435; non-ed. $535; add'l subs. $40/ea.												
	Category B (incl. 12 subs.): educational $635; non-ed. $735; add'l subs. $40/ea.												
	Category C (incl. 30 subs.): educational $1260; non-ed. $1360; add'l subs. $35/ea.												
	TUGboat back volumes	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	
	Circle volume(s) desired:	v. 1	v. 2	v. 3	v. 4	v. 5	v. 6	v. 7	v. 8	v. 9	v. 10	v. 11	
	Indiv. issues $18.00 ea.	$18	$50	$35	$35	$35	$50	$50	$50	$50	$75	$75	

Issues of TUGboat will be shipped by air service outside North America.

TOTAL ENCLOSED: ________
(*Prepayment in U.S. dollars required*)

Membership List Information

Institution: ________________

Principal contact: ________________
Phone: ________
Specific applications or reason for interest in TEX:

This installation can offer the following software or technical support to TUG:

Please list high-level TEX users at your site who would not mind being contacted for information; give name, address, and telephone.

Date: ________
Status of TEX: [ ] Under consideration
[ ] Being installed
[ ] Up and running since: ____
Approximate number of users: ____
Version of TEX:
[ ] Pascal
[ ] C
[ ] other (describe)
From whom obtained: ________
Hardware on which TEX is used:

Computer(s)	Operating system(s)	Output device(s)

Request for Information

The TEX Users Group maintains a database and publishes a membership list containing information about the equipment on which TEX is (or will be) installed and about the applications for which TEX is used. This list is updated periodically and distributed to members with TUGboat, to permit them to identify others with similar interests. Thus, it is important that the information be complete and up-to-date.

Please answer the questions below, in particular those regarding the status of TEX and the hardware on which it runs. (Operating system information is particularly important in the case of IBM mainframes and VAX.) This hardware information is used to group members in the listings by computer and output device.

If accurate information has already been provided by another TUG member at your site, indicate that member's name and the same information will be repeated automatically under your name. If your current listing is correct, you need not answer these questions again. Your cooperation is appreciated.

- *Send completed form with remittance* (checks, money orders, UNESCO coupons) to:
  TEX Users Group
  P. O. Box 594
  Providence, Rhode Island 02901, U.S.A.
- *For foreign bank transfers* direct payment to the TEX Users Group, account #002-031375, at:
  Rhode Island Hospital Trust National Bank
  One Hospital Trust Plaza
  Providence, Rhode Island 02903-2449, U.S.A.
- *General correspondence* about TUG should be addressed to:
  TEX Users Group
  P. O. Box 9506
  Providence, Rhode Island 02940-9506, U.S.A.

Name: ______________________
Home [ ]
Bus. [ ] Address: ______________________
____
____
____

Qty	1991 Membership/TUGboat Subscription (Jan.-Dec.)											Amount
	Ordinary: [ ] $45.00 Students: [ ] $35.00 (photocopy of student ID required)											
	TUGboat back volumes	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	
	Circle volume(s) desired:	v. 1	v. 2	v. 3	v. 4	v. 5	v. 6	v. 7	v. 8	v. 9	v. 10	v. 11	
		$18	$50	$35	$35	$35	$50	$50	$50	$50	$75	$75	

Issues of TUGboat will be shipped via air service outside North America.
Quantity discounts available on request.

TOTAL ENCLOSED: __________
(*Prepayment in U.S. dollars required*)

Membership List Information

Institution (if not part of address): ______________________
____

Title: ______________________
Phone: ______________
Network address: ______________________
[ ] Arpanet [ ] BITnet
[ ] CSnet [ ] uucp
[ ] JANET [ ] other __________

Specific applications or reason for interest in TEX:

My installation can offer the following software or technical support to TUG:

Please list high-level TEX users at your site who would not mind being contacted for information; give name, address, and telephone.
____
____

Date: ______________

Status of TEX: [ ] Under consideration
[ ] Being installed
[ ] Up and running since: ______
Approximate number of users: ______

Version of TEX:
[ ] Pascal
[ ] C
[ ] other (describe)
From whom obtained: __________

Hardware on which TEX is used:

Computer(s)	Operating system(s)	Output device(s)

TEX FOR THE 90'S

Are you still struggling with PXL's, PK's or GF's? Move on to scalable fonts:

- Save megabytes of storage–entire VTEX fits on one floppy.
- Instantly generate any font in any size and in any variation from 5 to 100 points.
- Standard font effects include compression, slant, smallcaps, outline, shading and shadow.
  New: landscape. **New: scalable graphics.**
- Discover the universe of MicroPress Font Library professional typefaces: not available from any other TEX vender.

List price $399 **Introductory offer $299**

Includes the VTEX typesetter (superset of TEX), 10 scalable typefaces, VVIEW (arbitrary magnification on EGA, CGA, VGA, Hercules, AT&T), VLASER (HP LaserJet), VPOST (PostScript), VDOT (Epson, Panasonic, NEC, Toshiba, Proprinter, Star, DeskJet) and manuals.
Introductory offer expires on **June 1, 1991.** S/H add $5. COD add $5. WordPerfect Interface add $100. Site licenses available. Dealers' inquiries welcome. Professional typefaces available for older implementations of TEX.

MICRO
PRESS

MicroPress Inc.

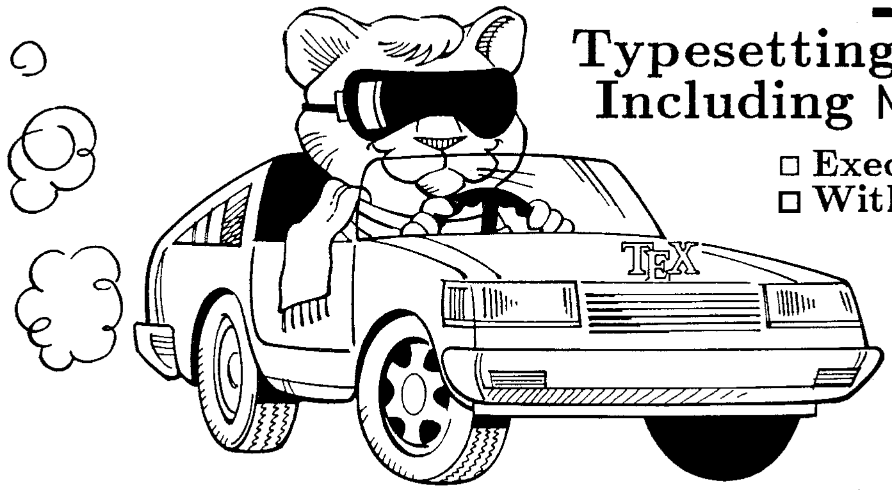
68-30 Harrow Street, Forest Hills, NY 11375
Tel: (718) 575-1816 Fax: (718) 575-8038

VTEX is a trademark of MicroPress Inc. Other Products mentioned are trademarks of their respective companies.

TurboTEX

Typesetting Software Including METAFONT

□ **Executables $150**
□ **With source $300**



TEX 3.0 Update! Now Includes EGA/VGA Previewer!

TurboTEX Release 3.0 software brings you the latest TEX 3.0 and METAFONT 2.0 standards: preloaded plain TEX, LATEX, AMS-TEX and AMS-LATEX, and plain METAFONT interfaced to CGA/EGA/VGA/Hercules graphics; TRIP and TRAP certification; Computer Modern and LATEX fonts, and printer drivers for HP LaserJet Plus/II/IIP, HP DeskJet, PostScript, and Epson LQ and FX dot-matrix printers. This wealth of software runs on your IBM PC (MS-DOS or OS/2), UNIX, or VAX/VMS system.

■ **Best-selling Value:** TurboTEX sets the standard for power and value among TEX implementations: one price buys a complete, commercially-hardened typesetting system. *Computer* magazine recommended it as "the version of TEX to have," *IEEE Software* called it "industrial strength," and thousands of satisfied users worldwide agree.

TurboTEX gets you started quickly, installing itself automatically under MS-DOS, and compiling itself automatically under UNIX. The 90-page User's Guide includes generous examples and a full index, and leads you step-by-step through installing and using TEX and METAFONT.

■ **Power Features:** TurboTEX breaks the 640K memory barrier under MS-DOS on any IBM-compatible PC with our virtual memory sub-system. Even without expanded memory hardware, you'll have the same sized TEX that runs on multi-megabyte mainframes, with plenty of memory for large documents, complicated formats, and demanding macro packages (like PICTEX and AMS-LATEX 2.0) that break other TEX implementations. On larger computers, TurboTEX runs up to 3 times faster in less memory than the Stanford Pascal distribution.

■ **Source code:** Order the TurboTEX source in portable C, and you will receive more disks with over 85,000 lines of generously commented TEX, TurboTEX, METAFONT, and printer driver source code, including: our WEB system in C; PASCHAL, our proprietary Pascal-to-C translator; and preloading, virtual memory, and graphics code. TurboTEX meets C portability standards like ANSI and K&R, and is robustly portable to a growing family of operating systems.

■ **Availability & Requirements:** TurboTEX executables for IBM PC's include the User's Guide and require 640K and hard disk. Order source code (includes Programmer's Guide) for other machines. Source compiles with Microsoft C 5.0 or later on the PC; other systems need 1 MB memory and a C compiler supporting UNIX standard I/O. Media is 360K 5-1/4" PC floppy disks; other formats at extra cost.

■ **Upgrades:** If you have TurboTEX Release 2.0, you can upgrade the executables for only $40. If you have the source distribution, upgrade both executables and source for $80. Or, get either applicable upgrade free when you buy the AP-TEX fonts (see facing page) for $200!

■ **No-risk trial offer:** Examine the documentation and run the PC TurboTEX for 10 days. If you are not satisfied, return it for a 100% refund or credit. (Offer applies to PC executables only.)

■ **Free Buyer's Guide:** Ask for the free, 70-page Buyer's Guide for more details on TurboTEX and dozens of TEX-related products: previewers, TEX-to-FAX and TEX-to-Ventura/Pagemaker translators, optional fonts, graphics editors, public domain TEX accessory software, books and reports.

Ordering TurboTEX

Ordering TurboTEX is easy and delivery is fast, by phone, FAX, or mail. Terms: Check with order (free media and ground shipping in US), VISA, Mastercard (free media, shipping extra); Net 30 to well-rated firms and public agencies (shipping and media extra). Discounts available for quantities or resale. International orders gladly expedited via Air or Express Mail.

The Kinch Computer Company
PUBLISHERS OF TURBOTEX
501 South Meadow Street
Ithaca, New York 14850 USA
Telephone (607) 273-0222
FAX (607) 273-0484

AP-TEX Fonts

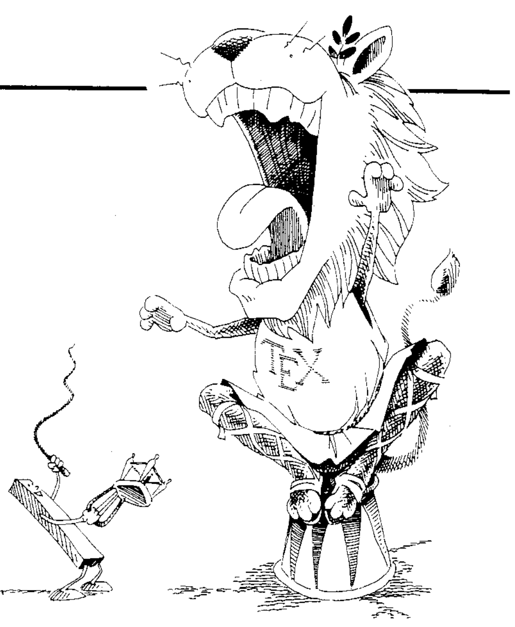
**TEX-compatible Bit-Mapped Fonts
Identical to
Adobe PostScript Typefaces**

If you are hungry for new TEX fonts, here is a feast guaranteed to satisfy the biggest appetite! The AP-TEX fonts serve you a banquet of gourmet delights: 438 fonts covering 18 sizes of 35 styles, at a total price of $200. The AP-TEX fonts consist of `PK` and `TFM` files which are exact TEX-compatible equivalents (including "hinted" pixels) to the popular PostScript name-brand fonts shown at the right. Since they are directly compatible with any standard TEX implementation (including kerning and ligatures), you don't have to be a TEX expert to install or use them.

When ordering, specify resolution of 300 dpi (for laser printers), 180 dpi (for 24-pin dot matrix printers), or 118 dpi (for previewers). Each set is on ten 360 KB 5-1/4" PC floppy disks. The $200 price applies to the first set you order; order additional sets at other resolutions for $60 each. A 30-page user's guide fully explains how to install and use the fonts. Sizes included are 5, 6, 7, 8, 9, 10, 11, 12, 14.4, 17.3, 20.7, and 24.9 points; headline styles (equivalent to Times Roman, Helvetica, and Palatino, all in bold) also include sizes 29.9, 35.8, 43.0, 51.6, 61.9, and 74.3 points.

The Kinch Computer Company
PUBLISHERS OF TURBOTEX
**501 South Meadow Street
Ithaca, New York 14850
Telephone (607) 273-0222
FAX (607) 273-0484**

Avant Garde	Bold
Avant Garde	Bold Oblique
Avant Garde	Demibold
Avant Garde	Demibold Oblique
Bookman	Light
Bookman	Light Italic
Bookman	Demibold
Bookman	Demibold Italic
Courier	
Courier	Oblique
Courier	Bold
Courier	Bold Oblique
Helvetica	
Helvetica	Oblique
Helvetica	Bold
Helvetica	Bold Oblique
Helvetica Narrow	
Helvetica Narrow	Oblique
Helvetica Narrow	Bold
Helvetica Narrow	Bold Oblique
Schoolbook	New Century Roman
Schoolbook	New Century Italic
Schoolbook	New Century Bold
Schoolbook	New Century Bold Italic
Palatino	Roman
Palatino	Italic
Palatino	Bold
Palatino	Bold Italic
Times	Roman
Times	Italic
Times	Bold
Times	Bold Italic
Zapf Chancery	Medium Italic
Symbol ΔΦΓϑΛΠΘ	
Zapf Dingbats ✂✁☞❑	

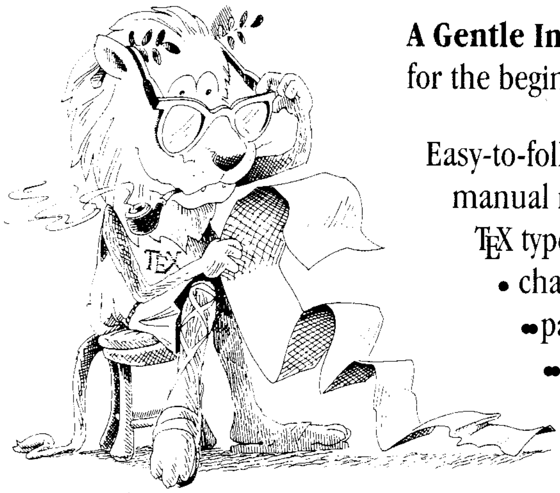


TAME THAT TEX LION!

with

A Gentle Introduction to TEX

A Manual for Self-study by Michael Doob



©Lions, *The TEXbook*, 1986; used by permission of Addison-Wesley Publishing Co.

A Gentle Introduction to TEX is perfect for the beginning TEX user.

Easy-to-follow and straightforward, this 89 page, softbound manual reveals the basics behind TEX typesetting:

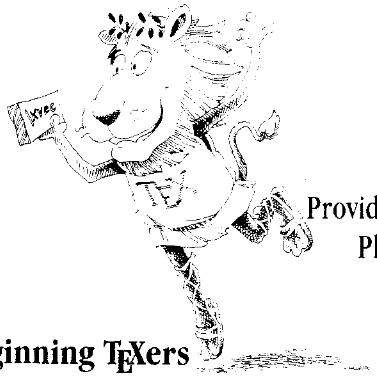
- characters and words
- paragraphs and pages
- mathematics
- simple tables
- simple macros

...with dozens of helpful examples and solutions in 11 friendly chapters.

Special offer for TUG members ...	**$10**
10 or more copies (members)	**$ 8**
Regular price	**$15**

Shipping:	U S:	$2 per copy
	Outside of U S:	$3 per copy surface; $4 air

Ask about TUG's other publications for beginning TEXers



Available now from TUG

TEX Users Group
P. O. Box 9506
Providence, RI 02940 U. S. A.
Phone: (401) 751–7760
Fax: (401) 751–1071

Mastercard/Visa, checks and money orders accepted

For $\TeX$ Users....

New Services and Prices from Computer Composition Corporation

We are pleased to announce the installation of several ***new output services*** now available to $\TeX$ users:

1. High Resolution Laser Imaging (1200 dpi) from *Postscript diskette files* created on either Mac- or PC-based systems.
2. High Resolution Laser Imaging (960 dpi) from *DVI magnetic tape or diskette files* using a variety of typefaces in addition to the Computer Modern typeface family.
3. High quality laser page proofs at 480 dpi.
4. **NEW PRICING** for high resolution laser imaging:

a.	From **Postscript text files** in volumes over 400 pages	**$2.00 per page**
b.	From **Postscript text files** in volumes between 100 & 400 pages	**$2.25 per page**
c.	From **Postscript text files** in volumes below 100 pages	**$2.40 per page**
d.	From **DVI files** in volumes over 400 pages	**$2.15 per page**
e.	From **DVI files** in volumes between 100 & 400 pages	**$2.30 per page**
f.	From **DVI files** in volumes below 100 pages	**$2.45 per page**

NOTE: DEDUCT $1.00 FROM THE ABOVE PRICES FOR HIGH QUALITY LASER PAGE PROOFS.

5. **All jobs shipped within 48 hours.**

Call or write for page samples or send us your file and we will image it on the output unit of your choice.

COMPUTER COMPOSITION CORPORATION
1401 West Girard Avenue • Madison Heights, MI 48071
(313) 545-4330 FAX (313) 544-1611
— *Since 1970* —

▼ ● ☆ ➡ □ ⇩ ★

$79

IS ALL IT WILL TAKE TO ADD A FEW OF THESE TO YOUR TEX DOCUMENTS

◇ ← ● ▲ ☆ ⬇ ○

With TEXPIC Graphics language* you will have the tools to make graphics for your TEX documents. TEXPIC is now available from Bob Harris at:

MICRO PROGRAMS INC
251 Jackson Avenue, Syosset NY 11791
Telephone: (516) 921 1351.

* TUG Boat Volume 10, No. 4, Page 627
1989 Stanford Conference Proceedings

A new and unique service from the Printing Division of the Oldest Press in the World

TEX-to-TYPE

The CAMBRIDGE service that lets you and your publisher decide how your mathematical or scientific text will appear.

Monotype output in Times and Helvetica as well as a complete range of Computer Modern faces from your TEX keystrokes

The right of the University of Cambridge to print and sell all manner of books was granted by Henry VIII in 1534. The University has printed and published continuously since 1584.

For details contact

TECHNICAL APPLICATIONS GROUP CAMBRIDGE UNIVERSITY PRESS
UNIVERSITY PRINTING HOUSE SHAFTESBURY ROAD CAMBRIDGE CB2 2BS ENGLAND

TELEPHONE (0223) 325070

Three truths about TeX

Beautiful mathematical typography.
Difficult to use. Unsupported.

True. False. False.

Northlake Software provides TeX and LaTeX products for VMS systems. We've organized the pieces so you can find them and use them. The device interfaces in our T2 family are compatible, functional. They're documented. We provide support.

These are tools you can pick up and use once a month. Or every day. And they're so cleanly designed, you'll enjoy using them.

The truth and beauty —
they're up to you.

TEX 3.1 and
METAFONT 2.7
now available –
virtual font utilities;
newest versions
of the best public
drivers, utilities,
and TEX and
LATEX macros;
AMS, Concrete,
other new fonts.

Northlake Software
812 SW Washington, Suite 1100
Portland, Oregon 97205-3215
USA
800-845-9111 503-228-3383
Fax 503-228-5662 tex@nls.com

TEX

THE ARBORTEXT WAY

Another first from ArborText!

Virtual Fonts

ü Ü ö Ö ä Ä

We have added built-in support to

Preview and DVILASER

which makes it easy to use the new

multilingual features of TEX 3.0.

Future enhancements will provide

a standard set of virtual fonts based on

coding conventions adopted at TEX 90.

TEX 3.0 and support software is available now for DEC/Risc-Ultrix, Sun, Apollo, and HP9000 (300 & 400). *μTEX* 3.0 is a newly enhanced version for the IBM PC.

535 W. William St., Ann Arbor, MI 48103, (313) 996-3566, FAX (313) 996-3573

ARBORTEXT INC.

Publishing Companion® translates

WordPerfect

to

TEX or LATEX

IN ONE EASY STEP!

With **Publishing Companion**, you can publish documents using TEX or LATEX with **little or no TEX knowledge.** Your WordPerfect files are translated into TEX or LATEX files, so anyone using this simple word processor can immediately begin typesetting their own documents!

Publishing Companion translates EQUATIONS, FOOTNOTES, ENDNOTES, FONT STYLES, and much more!

Retail Price	$249.00
Academic Discount Price	$199.00

For more information or to place an order, call or write:

K-TALK® COMMUNICATIONS

30 West First Ave, Suite 100
Columbus, Ohio 43201
(614)294-3535
FAX (614)294-3704

TYPESET QUALITY WITH THE EASE OF WORD PROCESSING

9009-9

TEX EDITION of MathEdit

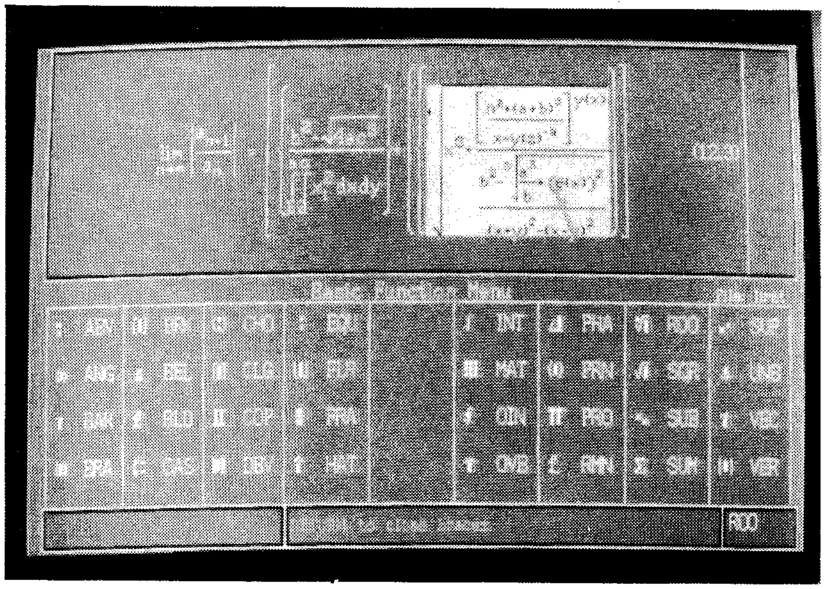
This powerful new equation editor can be used to create equations for TEX. For IBM PC

MathEdit

* Easy to Use Version 2.0

* Menu driven so no codes need to be learned

* High-quality TEX printing



WYSIWYG ->
View your equation as you create it. Then insert into your TEX document with one command.

K·TALK®
COMMUNICATIONS

30 West First Avenue
Columbus, Ohio 43201
(614) 294-3535
FAX (614) 294-3704

TEX Edition ONLY	$129.00
Professional Edition	$199.00

Shipping: $4 (U.S.A.), $25 (Canada), $35 (Overseas)
VISA, Mastercard and University and Government P.O.'s accepted.

Do more and do it better with new PCTeX.

PC TeX, PC TeX/386 & Big PC TeX/386, Versions 3.0

Feature/Specification		PC TeX 2.93	PC TeX 3.0	PC TeX/386 3.0	Big PC TeX/386 3.0
Page & Memory Capacity You won't see "TeX Capacity Exceeded"!	`mem_max`	65534 (1.00)	**131070** (Double!)	**131070** (Double!)	**262140** (Quadruple!)
Hyphenation Table Size Space for hyphenation patterns	`trie_size`	15000 (1.00)	**30000** (Double!)	**30000** (Double!)	**60000** (Quadruple!)
Trie Op Size Complexity of hyphenation patterns	`trie_op_size`	255 (1.00)	**1024** (4.02)	**1024** (4.02)	**2048** (8.03)
Maximum Trie Ops Per Language Especially important for Dutch and German hyphenation		N/A	**512**	**512**	**512**
Buffer Size Maximum # of characters on input lines	`buf_size`	1024 (1.00)	**1500** (1.46)	**1500** (1.46)	**3000** (2.93)
Stack Size Maximum # of simultaneous input sources	`stack_size`	200 (1.00)	**200** (1.00)	**200** (1.00)	**300** (1.50)
Maximum # of Strings	`max_strings`	4500 (1.00)	**5000** (1.11)	**5000** (1.11)	**10000** (2.22)
String Pool Maximum # of characters in strings	`pool_size`	50000 (1.00)	**50000** (1.00)	**60000** (1.20)	**60000** (1.20)
Save Size Space for saving values outside current group	`save_size`	600 (1.00)	**2000** (3.33)	**2000** (3.33)	**4000** (6.67)
Maximum # of TeX Commands	`hash_size`	3000 (1.00)	**5000** (1.66)	**5000** (1.66)	**10000** (3.33)
Minimum Free RAM Required		385K (1.00)	**385K** (1.00)	**1.3M** (3.38)	**1.3M** (3.38)
Minimum Free RAM Recommended Memory recommended for optimum performance		550K (1.00)	**550K** (1.00)	**1.3M** (2.36)	**4.0M** (7.27)
Font Memory For TFM data storage	`font_mem_size`	51199 (1.00)	**65534** (1.28)	**65534** (1.28)	**65534** (1.28)
Maximum Fonts Per Job	`font_max`	127 (1.00)	**127** (1.00)	**127** (1.00)	**255** (2.00)
List Price Order yours today!		$249 (1.00)	**$249** (1.00)	**$295** (1.18)	**$349** (1.40)
Upgrade Price From PC TeX 2.93 or earlier version		$50 (1.00)	**$50** (1.00)	**$99** (1.98)	**$149** (2.98)

This all adds up to...

More power, greater performance, and increased memory capacity for the latest versions of popular macro packages like LaTeX and AMS-TeX. And all three new PC TeX products feature the character sets and hyphenation tables to handle even the most complex European languages.

Order today. Call (415) 388-8853.

PERSONAL TeX INC
12 Madrona Avenue
Mill Valley, CA 94941

TYPESETTING: JUST

$2.50

PER PAGE!

Send us your TEX DVI files and we will typeset your material at 2000 dpi on quality photographic paper — $2.50 per page!

Choose from these available fonts: Computer Modern, Bitstream Fontware™, and any METAFONT fonts. (For each METAFONT font used other than Computer Modern, $15 setup is charged. This ad was composed with PCTEX® and Bitstream Dutch (Times Roman) fonts, and printed on RC paper at 2000 dpi with the Chelgraph IBX typesetter.)

And the good news is: just $2.50 per page, $2.25 each for 100+ pages, $2.00 each for 500+ pages! Laser proofs $.50 per page. ($25 minimum on all jobs.)

Call or write today for complete information, sample prints, and our order form. **TYPE 2000, 16 Madrona Avenue, Mill Valley, CA 94941. Phone 415/388-8873.**

TYPE
2000

DVIPSONE™

The converter that produces resolution-independent Post Script output using outline fonts

- Tired of reconverting DVI files to PS form every time you switch to a higher resolution device?
- Unhappy about the disk space chewed up by those high-resolution bitmapped fonts?
- Sick of running out of virtual memory on the printer when you use more than a handful of fonts?

DVIPSONE™ runs on IBM PC compatibles and can use any Adobe Type 1 outline font, including printer resident fonts, Computer Modern outline fonts and any of over 700 fonts in the Adobe font library.

For more information contact: Y&Y, 106 Indian Hill, Carlisle MA 01741
(508) 371-3286

All product names are trademarks of their respective manufacturers.

TEX Publishing Services

From the Basic

The American Mathematical Society can offer you a basic TEX publishing service. You provide the DVI file and we will produce typeset pages using an Autologic APS Micro-5 phototypesetter. The low cost is basic too: only $5 per page for the first 100 pages; $2.50 per page for additional pages, with a $30 minimum. Quick turnaround is important to you and us ... a manuscript up to 500 pages can be back in your hands in just one week or less.

To the Complex

As a full service TEX publisher, you can look to the American Mathematical Society as a single source for all your publishing needs.

Macro-Writing	**TEX Problem Solving**	**Autologic Fonts**	**Keyboarding**
Art and Pasteup	**Camera Work**	**Printing**	**Binding**

For more information or to schedule a job, please contact Regina Girouard, American Mathematical Society, P.O. Box 6248, Providence, RI 02940 or call 401-455-4060 or 800-321-4AMS in the continental U.S.

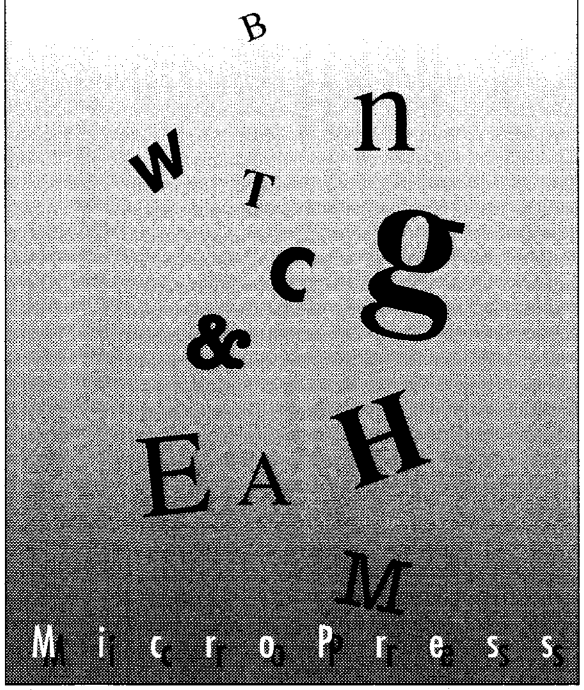
PXLGen™

Finally, you can use superior MicroPress scalable typefaces with your version of TEX.

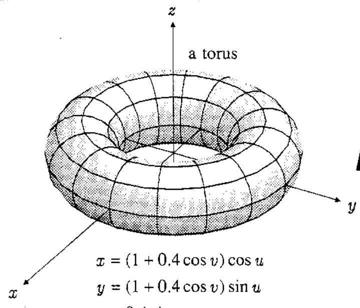
- Choose from 150+ quality typefaces
- Generate PXL/PK files in seconds
- Font effects: compressed/expanded fonts, shading, outline, smallcaps, and more
- Creates matching TFM files
- As low as $10 per font

Contact MicroPress for pricing and availability for your CPU.

MICROPRESS INC.
68-30 HARROW STREET, FOREST HILLS, NY 11375
TEL: 718-575-1816 FAX: 718-575-8038



PXLGen is a trademark of MicroPress Inc. Other Products mentioned are trademarks of their respective companies.



$x = (1 + 0.4 \cos v) \cos u$
$y = (1 + 0.4 \cos v) \sin u$
$z = 0.4 \sin v$

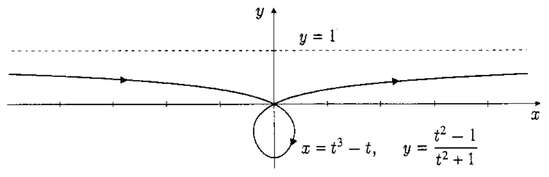
MG
Mathematical Graphics System

- an MS-DOS based system for generating high quality mathematical graphics on screen and for printing in a PostScript environment.
- generates both encapsulated PostScript and special PostScript for including in a TEX document where TEX will typeset the labels.

Write or phone for information.

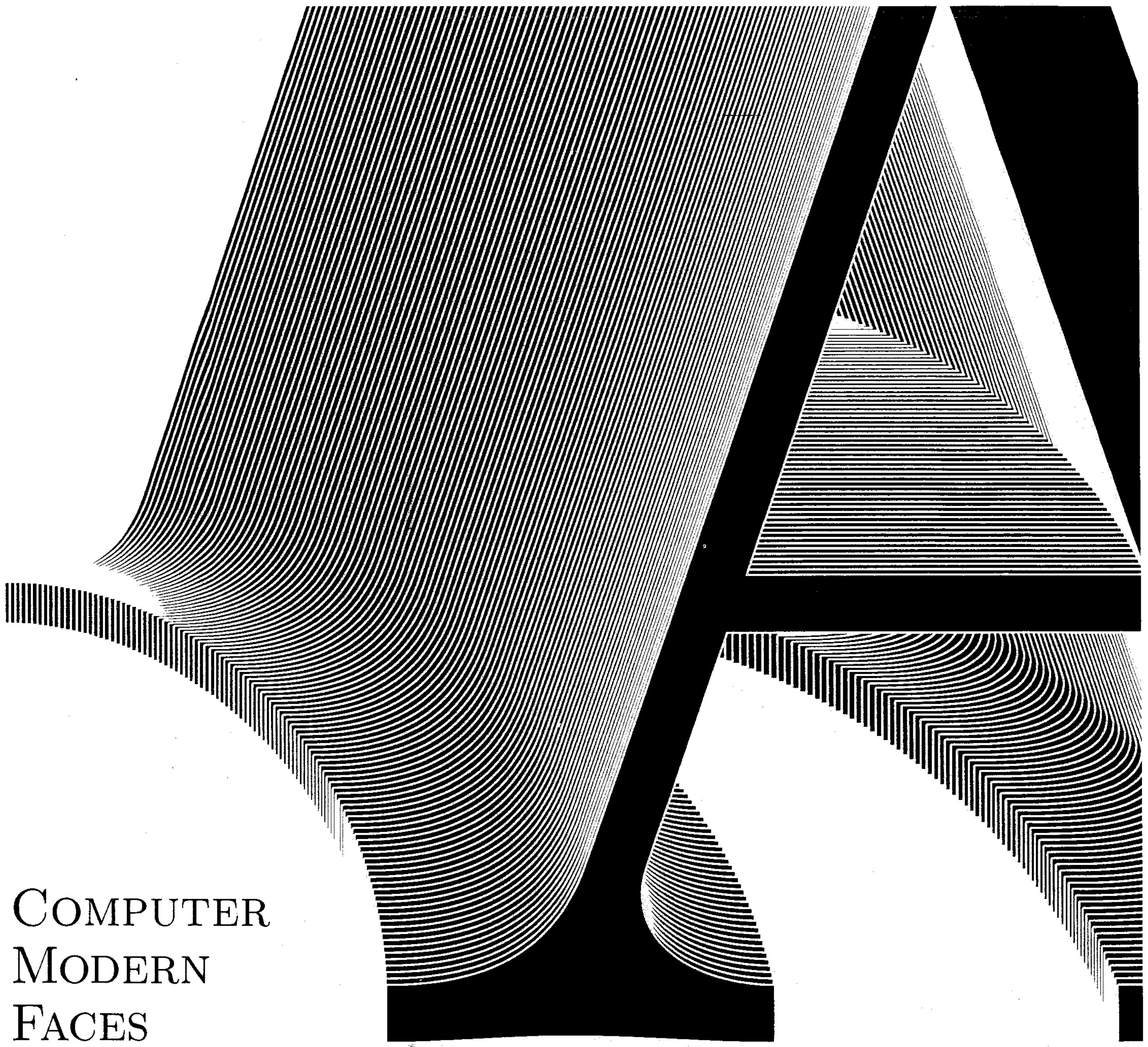
Single CPU: US$95.00
Shipping: $5 USA & Canada, $10 overseas
Demo diskette: $10 (credit towards purchase)

MG Software
4223 W. 9th Ave.
Vancouver, B.C.
Canada. V6R 2C6
(604)-228-8550



Index of Advertisers

192, 198 American Mathematical Society
193 ArborText
Cover 3 Blue Sky Research
190 Cambridge University Press
189 Computer Composition
194, 195 K-Talk Communications
186, 187 Kinch Computer Company
200 MG Software
185, 200 MicroPress, Inc.
190 Micro Programs, Inc.
191 Northlake Software
196 Personal TEX Inc.
184, 188, 199 TEX Users Group
197 Type 2000
198 Y&Y



Computer
Modern
Faces

Adobe
Type 1
PostScript
Fonts

Blue
Sky
Research

Forty faces of Computer Modern
designed by Donald Knuth
published in Adobe Type 1 format
compatible with
Adobe Type Manager
and all PostScript printers

$345.00 Educational $195.00
Macintosh or MS-DOS

Blue Sky Research
534 Southwest Third Avenue
Portland, Oregon 97204 USA
(800) 622-8398, (503) 222-9571
FAX (503) 222-1643

TUGBOAT

Volume 12, Number 1 / March 1991
TEX90 Conference Proceedings

Introduction
3 Introduction / *Barbara Beeton*

Databases and Hypertext and their Relationship with TEX
5 Erich Neuwirth / *Quick and dirty databases with nice output: AWK and TEX*
8 Christine Detig / *TEX & hypertext — The future of electronic publishing?*
13 Les Carr, Sebastian Rahtz and Wendy Hall / *Experiments with TEX and hyperactivity*

User Groups and Dissemination of Information
21 Johannes Braams, Victor Eijkhout and Nico Poppelier / *The Dutch national LATEX effort*
25 Adrian Clark / *Documenting a TEX archive*

Typographics: Æsthetics and Practicalities
28 Thomas Kneser / *LATEX-paragraphs floating around figures*
31 Victor Eijkhout / *The document style designer as a separate entity*
35 Philip Taylor / *Improving the æsthetics of mixed-font documents*
45 Alan Wittbecker / *ArchiTEX, a preliminary international page pattern maker*
58 Friedhelm Sowa / *Integration of graphics into TEX*

Expanding Horizons
64 Timothy Murphy / *PostScript, QuickDraw, TEX*
66 M.P. Maclenan and G.M. Burns / *An approach to drawing circuit diagrams for text books*
70 Olivier Nicole / *A graphic driver to interface statistical software S and PICTEX*

Late Breaking News
74 Frank Mittelbach and Rainer Schöpf / *Towards LATEX 3.0*

Public Domain TEX
80 Brian Hamilton Kelly / *Public-domain, documented implementations of TEX and METAFONT for VAX/VMS*

Documentation, Origination, Editing and Markup
84 Malcolm Clark / *Post congress tristesse*
90 Kees van der Laan / *SGML (, TEX and ...)*
105 N.A.F.M. Poppelier / *SGML and TEX in scientific publishing*
110 Amy Hendrickson / *Getting TEXnical: Insights into TEX macro writing techniques*
111 Reinhard Wonneberger and Frank Mittelbach / *BIBTEX reconsidered*

Font Design and New Fonts
125 Alan Hoenig / *Labelling figures in TEX documents*
129 Yannis Haralambous / *Typesetting old german: Fraktur, Schwabacher, Gotisch and initials*
139 Mícheál Ó Searcóid / *The Irish alphabet*
149 Dean Guenther and Janene Winter / *An international phonetic alphabet*
157 Adrian F. Clark / *Practical halftoning with TEX*

TEX Training and Education: The New User
166 Angela Barden / *Some TEX manuals*
171 Konrad Neuwirth / *TEX in schools: Just say no*

Announcements
175 Calendar
176 Nominations
176 Jiří Veselý / *TEX in Czechslovakia*
177 Wayne Sullivan / *sbTEX and sbMF*
178 T.V. Raman / *TEX TALK*

TUG Business
179 Institutional members

Forms
182 TUG membership applications

Advertisements
200 Index of advertisers