TEX: A Branch in Desktop Publishing Evolution, Part 2

Barbara Beeton
American Mathematical Society
Karl Berry
TEX Users Group
David Walden

Donald Knuth began the development of TEX in 1977 and had an initial version running in 1978, with the aim of typesetting mathematical documents with the highest quality, and with archival reproducibility far into the future. Its usage spread, and today the TEX system remains in use by a vibrant user community. However, the world of TEX has always been somewhat out of sync with the commercial desktop publishing systems that began to come out in the mid-1980s and are now ubiquitous in the worlds of publishing and printing.

Part 1 of this history (*Annals* vol. 40, no. 3) was about the creation of TEX at Stanford and how it began to spread. This part is about (a) the expansion of TEX-based and TEX-related technology, and development of a worldwide community of TEX users and developers following the lead of Knuth's original collaboration model, and (b) the impact TEX has had on the broader world.

In Part 1 we were primarily talking about TEX as developed by Knuth. In Part 2 we sometimes speak of (LA)TEX, meaning TEX or LATEX but mostly LATEX. We also often say TEX when we mean TEX and everything that has been built on top of and around TEX; we hope the distinction between the TEX program itself and these extended meanings is clear from the context.

To save on the page count for *Annals* publication, a majority of this history's references and all of its notes are posted on the Web (`tug.org/pubs/annals-18-19`) with back pointers to pages in the published paper. Take a look now or anytime as you are reading the article.

EVOLUTION OF THE USER COMMUNITY

In Part 1 of this history, we sketched the development of TEX and Metafont, starting in 1977, by Knuth, his graduate students, and others, through Knuth's definitive releases of TEX82 and Metafont84, and his final significant addition to TEX, extension to 8-bit characters, in 1989.

In the approximately 35 years since TEX82 and Metafont84, TEX and its extensions have been used by millions of users. We speculate that a reason for its longevity is the Knuth-TEX "business model" of keeping strong control of TEX itself (providing long-term stability and consistency[1]) while completely revealing its internals for the edification of all, and providing hooks such that practically anything could be (and has been) built on top of basic TEX. This "business model" provided the foundation for a worldwide and enduring community of developers, "resellers," and users.

Another aspect of TEX is that it had an endpoint. Knuth finished his software development and went back to writing TAOCP. Arguably, this has benefited future TEX developments by being a stable reference point, as Knuth intended. Much of the software built on top of TEX is also highly stable. Unlike commercial companies that need to change things regularly to keep selling new versions, the TEX community of users and developers try to keep things working as is, with new capabilities not obsoleting prior capabilities.

There is no single reason why many developers have worked together and individually on TEX-related projects for years or decades, adapting TEX to the ever-changing world. Some need a new typesetting capability for themselves; some get gratification from serving the community; for some "hacking TEX" is a fascinating hobby; some are on a mission to keep a beautiful system alive. Whatever the reason, the openness and stability of TEX has allowed those attracted to working on TEX and the TEX infrastructure to dig in and make their contributions.

The rest of this section sketches the continuing evolution of TEX and the TEX community—the larger part of the history of the development of TEX and TEX's role in desktop publishing.

Continued Expansion

By 1983–84, TEX was fully operational and running on many different computers with different operating systems. Also, LATEX had come on the scene in 1983, making TEX easier to use, as had BIBTEX for systematic handling of bibliographies and references for documents. As with TEX, these were available for free, with open source code, which surely contributed to their popularity.

Throughout the rest of the 1980s, use of TEX continued to grow. A variety of distributions, including commercial distributions for personal computers, became available to make TEX easier to install and begin to use, and were supplying other capabilities such as alternative fonts.

By this time the cultural move of authors being willing or having to do their own composition rather than having secretaries type manuscripts was underway, although some secretaries also learned to use (LA)TEX. Various technical publishers liked receiving manuscripts in (LA)TEX and created macro packages to format a (LA)TEX document in the publisher's print style for a journal or book. TEX produced higher-quality output than many publishers' systems, and saved some effort in converting manuscripts into the publisher's style. University departments also created macro packages for thesis styles.

By the end of the 1980s, TUG membership was approaching 4,000 members and additional TEX user groups, primarily focused on languages other than English, began to form—for ex-

ample, in 1988 a Dutch group (De Nederlandstalige TEX Gebruikersgroep, NTG) and a French group (le Groupe francophone des Utilisateurs de TEX, GUT), and a German group in 1989 (Deutschsprachige Anwendervereinigung TEX e.V., DANTE). The number of local user groups continued to expand throughout the 1990s. These groups typically formed to accomplish the tasks necessary for having good support in TEX for their languages: creating hyphenation patterns, creating special macros to make input convenient, extending the Computer Modern fonts, translating embedded English words (such as "Chapter") in the LATEX macros, and writing documentation in their own language. Eventually there would be about 25 such "local" user groups; nowadays, half a dozen or so actively carry on more typical user group activities—newsletters or journals, semi-yearly or yearly conferences, and so on. (See `tug.org/usergroups.html`.)

The user groups supported the task of information dissemination and help requests primarily through various mailing lists and the Usenet newsgroup `comp.text.tex` (still active today). As the 1980s ended, CTAN (Comprehensive TEX Archive Network) was created as a central repository of TEX-related software—packages, programs, fonts, documentation, and more. It was mirrored by dozens of sites around the world, providing reasonable access to most TEX users. CTAN remains the clearinghouse for the TEX world today.

Competition

In the second half of the 1980s, Microsoft Word (Word for Windows by 1990) and the graphically oriented desktop publishing systems had built momentum and undoubtedly began converting users from (LA)TEX. The paradigm for word processing and desktop publishing was strongly moving to WYSIWYG; markup of plain text files, which had been in widespread use in word processing, came to be popularly viewed as hopelessly complex. Moreover, Word was beginning to become a *de facto* standard for submissions to book and journal publishers, preferred by many authors and publishers.

In addition to Word impacting TEX use, a variety of output formats that were not natural to TEX had perhaps even greater effect, such as PDF, HTML, XML, and EPUB. While there had been a path from TEX's DVI output to PostScript since 1985–86, by 1993 PDF was also on the scene and getting from DVI to PDF required another post-processor (for instance, ps2pdf or Acrobat Distiller). In other words TEX was a further step more awkward to use compared with text processors that could go directly to PDF.

Also from 1993 to 1997, Mosaic and then Netscape had become serious rivals to email lists as a source of information and providing free searchable information immediately; these browsers also helped make the World Wide Web popular. This was the beginning of the "information wants to be free" movement, the decline in interest in print publications which had long provided support to professional societies (as had *TUGboat* for TUG), and the demand for open access journals.

The rise of WYSIWYG word processors and typesetting systems, the ubiquity of Word, and so many things being available on the Web meant that membership in the TEX user groups provided less and less evident benefit (and the TEX programs themselves had already been available for free). Thus, generally speaking, memberships in the various user groups began a slow decline over time, and the much larger group of non-user-group-member TEX users also declined.

Adapting to the Digital World

Regardless of the decline in TEX user group membership (and the concerns it caused), members of the TEX community continued on with new developments. A few major examples (among hundreds or thousands of other more or less important developments) include the following:

- The Web2C program was developed (1985) to maintain TEX's portability as the Pascal language went out of style.
- The TEX engine was extended by the Japanese company ASCII Media Works (ca. 1987) to support Japanese typesetting, first as jTEX, later pTEX and other derivatives.
- An improved version of LATEX, LATEX 2_ε, was created (1989).
- The ConTEXt format was developed (ca. 1993) from a need for sophisticated educational typesetting. It supports highly structured documents, integrated graphics with MetaPost and extensions with LuaTEX, and has an XML interface.
- Large TEX distributions oriented to specific operating systems were developed—teTEX for Unix (1994), MiKTEX for Windows (1996), and gwTEX for Mac (2001).
- The pdfTEX program was released (1996), supporting direct output to PDF from TEX and thus bringing TEX into the contemporary digital world.
- The PSTricks package was created (1997) that allowed specification of PostScript programs from LATEX and provided an alternative to drawing programs from outside the TEX world (for example, Adobe Illustrator).
- The Beamer class was written for LATEX (2003), providing another, especially powerful, slide presentation capability to the TEX community.
- The XETEX (2004) and LuaTEX (2007) programs were created; these are extensions of TEX, directly supporting Unicode input and use of OpenType, TrueType, and related font formats.

Also, in 1996, the first TEX Live software CD was produced, providing a large set of (LA)TEX macros, packages, and fonts, and in time adding other formats. These surely helped sustain TEX use and motivated some users and developers to continue to support the user groups. With ever-increasing Internet bandwidth, the user group journals got on the Web, but typically didn't substantively become more graphical or interactive as allowed by non-print publication. (An example of a non-LATEX format of TEX in the TEX Live collection is ConTEXt, an advanced TEX-based typesetting system developed by Hans Hagen for use in his own business and distributed freely to the TEX community—it is especially oriented to interfacing between TEX and XML.)

For a few years around the year 2000, the TEX Live software CDs (later DVDs) became at least as important a benefit as journals as a reason for people to join the various user groups. However, increasingly the TEX world has used the availability of the Internet to make more prior benefits of membership freely available, for example, free downloads of TEX Live (`tug.org/texlive`) and open access to *TUGboat* issues one year after publication. (For some time the CD/DVD has been known as the TEX Collection and has included: MacTEX, for Mac OSX including TEX Live; proTEXt, for Windows based on MiKTEX; TEX Live, for GNU/Linux, Unix, and Windows; and a snapshot of the CTAN archive. The project has always been carried out by a group of contributors, and there is a yearly poll of how many copies each user group needs to send to their members.)

The Internet, email, web servers, software version control systems, and other capabilities of the digital age also offer benefits. They allow better coordination of development efforts across TEX user groups and individuals working on various development, infrastructure maintenance, and user support activities. CTAN, TEX Live, the `tug.org` website itself, and many other development projects are examples of this. They also allow for different styles of membership administration, for instance, online voting for officers, paying for membership and user group store items online, and providing electronic-only subscriptions to user group publications (eliminating some printing and mailing costs).

The Future of the User Groups

The user groups continue to provide places (physical and virtual) to meet and exchange ideas for people who have interests in or needs for typography, font design, and a powerful typesetting system. In this sense, the TEX user groups are more like professional societies than computer product user groups. In another sense, the user groups and community of TEX developers are rather analogous to a small multi-national company, with multiple product lines and branch offices for various national markets, and worldwide user support capabilities—except this "company" is primarily made up of volunteers and there is not much possibility of top-down planning and assignment of resources.

TEX is a successful and continuing example of what is now generally known as open source software development. (Alexia Gaudeul extensively studied the TEX world through 2003 as an example of an open source software community.[2]) Supporting a large community of users are many volunteers (slowly changing over time) who put in time developing new capabilities and maintaining the existing capabilities of the TEX world. The (more or less ad hoc or informal) communication that goes on among various subsets of volunteers is critical.

However, membership in a TEX user group continues to provide less and less tangible benefit. Many people, especially younger ones, are no longer interested in paper journals. Forums such as `tex.stackexchange.com` and others not associated with user groups have become the principal electronic help mechanism, largely supplanting mailing lists and `comp.text.tex`. Computing historian David Grier has said, "... the Internet has become the new user group, and that's where you go for a lot of your support." Probably significant numbers of people are now joining the user groups or renewing memberships only to be part of the club and support the organizations and TEX more generally rather than to get particular benefits from their membership.

The longstanding TEX user groups probably have another decade or two of life until their memberships decline too far, and even when one (for instance, TUG) eventually becomes financially unviable (for instance, TUG being unable to support a print journal and a paid employee), it could continue as a volunteer organization: the current volunteer labor (or replacements for it) could put out an electronic-only journal, organize meetings, operate CTAN, and release the TEX software collection. In fact, given the state of online support and open source cooperation, the TEX community could survive without the formal user groups, while the various volunteer-staffed activities continue more or less unchanged. The real end for TEX users will come when appropriately capable people are no longer interested in providing the volunteer labor to continue TEX development and maintenance.

REFLECTIONS ON THE IMPACT OF TEX

The early and long-term impact of TEX was highly dependent on two things: the state of the art of computer technology at the time Knuth began his work, and Knuth's location at Stanford and his independence in what he did.

At the time of TEX's initial development, phototypesetting had substantially replaced typesetting with metal type, and the quality of published books and journals had decreased. The commercial hardware technology of desktop publishing (personal computers, laser printers) was still in the future. The software used at the time for commercial typesetting (for instance, from Atex for newspapers or from Science Typographers, Inc. used by the AMS) and for technical reports (for instance, Pub, nroff/troff) was not yet WYSIWYG—which was still in its initial stages of creation at the Xerox PARC laboratory. Digital type and type design (for example, Ikarus developed by Peter Karow of the URW type foundry) were breaking new ground but not widespread. While

word processing products such as WordPerfect and Word were still in the future, the popularity of word processing on a dedicated machine (for example, Wang) and then the WordStar software package indicated a building market demand for people other than professional typesetters to be able to do their own document formatting.

Stanford had state-of-the-art computer and display technology (and were close to Xerox PARC with its own state-of-the-art technology, and also close to the stream of Silicon Valley innovations). Knuth had students who could help him, and he could take a sabbatical year or (as he did later) reduce teaching responsibilities to concentrate on computer typesetting. He could take the time to study traditional typesetting methods. Finally, and at the foundation, he had deep mathematics *and* computer science understanding that could be applied to digital typesetting. He cared about typographical beauty and wanted to create a system able to do the "best" typesetting (not just "good enough" typesetting), and he also wanted to build a system that was stable and archival ("to create systems that would be independent of changes in printing technology as much as possible"[3]). Knuth was uninterested in personal financial gain from his typesetting work. He believed in collaboration and that advances in computer-based typesetting had previously been held back because of competing commercial interests. He had the connections and reputation to successfully get help from typography experts and attract potential collaborators and users in (especially) the mathematics and computer science communities. Knuth's reputation allowed him to get funding for the Stanford digital typesetting activities while keeping the technology essentially in the public domain.

While Knuth was an academic, he also knew how to program complex computer software systems that worked efficiently in practical use. The then state-of-the-art computer had a memory address space about one megabyte and could execute only a couple of million instructions per second. These were significant limitations in the face of the need for an implementation that could be used practically for typesetting large books, and the limitations affected the TEX language design and system implementation.

The rest of this section considers the impact of TEX in terms of first its strengths (and areas of influence), and then its weakness (including misconceptions). There is considerable overlap throughout, so items may fit in more than one category.

Areas of Strength or Influence

Breakthrough application. Knuth sought and, to a considerable extent, achieved a level of typesetting quality that was driven by researching and matching the best hot type and engraving typesetting, rather than just doing the (supposed) "best" a computer could do at the time, as was common for companies needing returns. TEX was an early example of good digital typesetting that non-TEX people knew about. That TEX had influence is clear from its fast adoption within the mathematics community and beyond. Knuth significantly revised TEX more than once; documented it comprehensively, including publishing the source code and detailed logs of his rigorous debugging efforts; and, from early on, paid small rewards for people finding new bugs in the program. Not only was TEX a breakthrough in its application area, it was also an example of a highly perfected instance of computer programming.

Math markup. TEX was originally developed by Knuth to produce a "beautiful" book with a lot of mathematics in it.[4] Perhaps the most important legacy of TEX is the language used for its math input. It's the language that mathematicians became used to, and it continues to be widely used today, both in TEX and in various other systems (for instance, MathJax, wikis, and even somewhat in the Rich Text Format, RTF, used by some versions of Microsoft Word).

Use in publishing. Over the decades since (LA)TEX initially came on the scene, many books and journals, especially for math and science, have been published using TEX or a TEX-based system, especially LATEX. Many publishers still use TEX although perhaps not as many as once did. Because most distributions of TEX are free, TEX can be adapted to and disproportionately used in less-developed countries with their smaller markets.

Innovative algorithms. Two of the algorithms developed for TEX have been used in other word processing and desktop publishing systems, for example, those for line breaking and hyphenation.

The line-breaking algorithm uses the dynamic-programming ("total fit") mathematical optimization method to improve the look of pages by avoiding ugly line breaks.

The hyphenation algorithm is driven by tables of statistically more likely patterns for valid breaks within words, and the algorithms and tables do a surprisingly good job of finding good hyphenation points and typically require only small tables of exceptions. Maintaining and using the program to process language dictionaries to produce the tables has become an international collaboration that spans many languages and applications in many text processing systems.

Although we don't know of its use in other systems as the line-breaking and hyphenation algorithms have been, Knuth's boxes-and-glue model is a surprisingly simple but powerful idea for dealing with words, paragraphs, pictures, and pages in a more or less common way, considering each type of element to be a box with glue being a stretchable thing (distance) between adjacent boxes.

The so-called *hz* micro-typesetting method of Hermann Zapf improves line-breaking further, lessening the need for end-of-line hyphens and making the right margin of justified text look better (`en.wikipedia.org/wiki/Microtypography`). The method has been fully implemented in pdfTEX and LuaTEX (discussed below). The *hz* approach also exists in some form in Adobe's InDesign.

The combination of the above four approaches (three of them from the 1970s) still appears to be at the state of the art for creating good looking columns of text.

Open source community advancing typography. TEX was created in an era when much software was freely shared, and this has remained the primary approach in the TEX world over the past 40 years. In this sense, TEX was an "open source" project (years before the term was coined) and continues to be a major open source project. The worldwide TEX community with its publications, and later websites, became and remains a center of digital typography R&D and technology exchange.

The TEX community's vast collection of materials related to TEX, the Comprehensive TEX Archive Network (CTAN—`ctan.org`), had at least naming impact on the equally vast sites of two freely available major programming languages—the Comprehensive Perl Archive Network (CPAN) and the Comprehensive R Archive Network (CRAN).

Impact of students. The Stanford students of Knuth and Charles Bigelow, beginning in the late 1970s and continuing well into the 1980s, provided a significant addition to the burgeoning desktop publishing and greater typography industry, for example, at Adobe, in the Word group at Microsoft, Frame Technology, and others.

Overall, Knuth's digital typesetting work may have had more impact on the world than his work with analysis of algorithms (TAOCP)—the latter being more theoretical and the former being more practical.

Weaknesses and Misconceptions

While TEX and Metafont were significant breakthroughs in computer-based typesetting created in the late 1970s and still relevant today, TEX had and has some clear areas of weakness.

For the word processing and desktop-publishing markets that developed after TEX, TEX clearly has the disadvantages of not being WYSIWYG, not directly handling popular type formats, and being essentially unknown to the consumer markets and largely unknown to the professional typography market.

Because TEX was created long before today's popular font formats and because TEX's own font formats were deeply embedded in the TEX implementation, TEX was always playing catchup as other font formats became popular. TEX originally had only its own font formats; when PostScript came on the scene, a post-processor to TEX (called dvips) was written to convert TEX's device-independent DVI output to PostScript, and work had to be done to connect TEX's fonts to the PostScript font system. After PDF was invented, eventually the TEX program was modified to produce PDF output directly (pdfTEX), with the advanced line-breaking features of so-called micro-typography also added, as mentioned above.

The TrueType and OpenType formats, which have essentially replaced Type 1 fonts in the rest of the digital world, are usually still converted to Type 1 fonts for use with TEX and pdfTEX. Considerably later (ca. 2004), another modification of the TEX engine, XƎTEX, was created that reads these font formats directly, but XƎTEX lacks the full micro-typesetting capability that was implemented in pdfTEX. LuaTEX, another extended TEX engine created in 2007, can also handle TrueType and OpenType directly, and does keep the full micro-typesetting capability. Besides supporting the modern font formats, another principal reason for the creation of XƎTEX and LuaTEX was to natively support Unicode input, notably UTF-8. LuaTEX also has great extensibility capabilities via an embedded Lua language interpreter.

Metafont, Knuth's companion to TEX for creating fonts, never became particularly popular; Charles Bigelow has given a lengthy discussion of this.[5] The early fonts Knuth produced were not very good but soon got to be adequate, and were ultimately as aesthetically sophisticated as any other digital type. (Knuth's Computer Modern is not especially pleasing to many people, nor has the Euler typeface developed for the AMS by Zapf and Knuth become a popular font.) Adobe's John Warnock also took issue with Knuth's approach to rasterizing fonts in a "Knowledge@Wharton" interview. An advantage of the Computer Modern fonts, which were openly published and freely available, was the absence of the copyright and intellectual property issues involved in using some commercial fonts; users of TEX using Computer Modern fonts could post their documents on the Web and academic publishers could accept submissions without worry.

In any case, as TEX's default typeface, Computer Modern has been used by millions of TEX users. The Computer Modern typeface *is* appreciated for being an unusually large set of fonts and for having a relatively complete set of math symbols. Also, over the years, members of the TEX community have designed and coded significant extensions to Computer Modern—font support for eastern European languages, Cyrillic, African, Vietnamese, and others (`ctan.org/topic/font-mf`). For a time, TEX likely supported a much wider array of languages and scripts considerably earlier than any other system did, with generally excellent quality. Members of the TEX community have also worked on making a vast number of popular and interesting fonts, created outside the TEX world, available for use with TEX.

Knuth's `WEB` system for literate programming of software development and documentation also never caught on very much in the larger computer world. `WEB` remains mainly today as the implementation source for TEX, Metafont, and related early programs.

There are other qualities that some people see as weaknesses of TEX but others don't see as disadvantageous, and there are some areas where most people in the desktop publishing world are unaware of TEX's breadth of use.

Hard to use. That TEX is not WYSIWYG is part of the reason TEX is viewed as hard to use. Not only does TEX markup look cryptic and hard to the uninitiated, using TEX markup requires learning a separate editor (for example, Emacs, WinEdt) or another kind of front end (for example, LyX, Scientific Word). However, for people who do use TEX and friends (for example, LATEX, XETEX, LuaTEX, and so on), the TEX systems can be easier to use and more productive than WYSIWYG systems. For instance, having figures appear at sensible places within pages of text can be easier in LATEX than in Word; you can have powerful search and editing capabilities in a separate editor; TEX naturally and automatically updates numbering and maintains cross-references when parts of documents are moved around in the drafting process; and so forth.

When difficulty is encountered with TEX (either in trying to do something quite normal or to do something quite unusual and sophisticated), the TEX community is more easily tapped for help (for example, from `tex.stackexchange.com`, online repositories of decades of user group journals, and so on) than one finds with some of the commercial word processing and desktop publishing systems.

Users of many TEX systems also have to deal with many fewer user interface changes, instances of new operating system releases obsoleting old versions of the text processing system, and new versions of source file formats. (Source files, at least for TEX and LATEX, are essentially totally portable among different users of each system, and what is compiled once can be and has been compiled successfully again years or decades later with the now-current TEX distribution.)

The TEX systems have great capacity for user modification or extension to gain efficiency in specifying the markup. In particular, TEX's markup approach is also useful for being embedded in larger workflows, and (LA)TEX is seen as an adjunct to various non-TEX systems (for instance, R, SAS, WordPress) or markup languages (for instance, XML, Markdown). On the negative side, many users who appreciate these capabilities for modification or extension view its macro-language approach to such changes as a "disaster." However, that hasn't prevented massive numbers of augmentations to TEX from being created.

Math only. Many people believe that the TEX systems are used only for math and perhaps "math adjacent" fields (for instance, economics). However, the TEX systems are in fact used by anyone who wants the advantages sketched above (such as power, beauty, stability, low cost) of TEX over word processors. A few areas include the rest of the sciences, the humanities, linguistics, law, for business back ends, catalogs and schedules. Such "unexpected" uses reported in *TUGboat* are stunning in their breadth.

Non-commercial. Another misconception is that TEX is not commercial. Knuth put TEX in the public domain in a way that allowed commercial companies to package it for particular markets or subsets of users, and a variety of (typically rather small) companies have sold TEX commercially over the years, starting in the mid-1980s. The commercial systems contributed significantly to expansion of TEX use by providing systems that installed easily on particular platforms and had additional features, fonts, graphics, and support that appealed to users. These days TEX is so well-packaged by a collaboration of the user groups that commercial versions are less and less differentiated from free versions. On the other hand, a relatively new commercial use of TEX is from the companies Overleaf and ShareLaTeX (which announced they were merging in mid-2017), which run web servers to process LATEX files, thus relieving users (or perhaps university departments) of maintaining local TEX distributions and perhaps allowing users to create LATEX-based documents from phones and tablets that don't have TEX running on them. The Overleaf

system includes a WYSIWYG front end to LaTeX. (The companies claim that many hundreds of thousands of users and thousands of institutions use their systems for millions of projects. It is perhaps doubtful that they have that much ongoing use beyond initial trials, but it is an indicator of a new area of TeX expansion.)

Front ends and editors for LaTeX are a place where a few small companies are making money by supplying a product supporting the TeX world. Some fonts have been sold that specifically augment TeX systems. There are typesetting services that turn manuscripts into (perhaps ready-to-print) TeX for publishers or individuals. (Many more examples can be seen in the advertising section of *TUGboat* throughout the years.)

The Future of TeX

TeX has now existed for 40 years. During this period many commercial word processing and desktop publishing systems have come and gone. While the TeX systems have never been important to users of the well-known commercial word processing and desktop publishing programs and the number of TeX users is surely minuscule compared to the couple of billion users of Word, it would not be surprising if the number of TeX users, mostly individuals doing their own typesetting, can be measured on the same scale as the number of users of a commercial desktop publishing system such as InDesign, used by many professional typesetters. This is despite (or because of) TeX still being a page layout system where the user explicitly types markup into a plain text editor.

We foresee continuing demand for TeX for many years.[6] The TeX world has powerful free tools producing high-quality typographic output. This will long be attractive to some people. For instance, the ArXiv system is helping increase popularity of TeX; many researchers from a variety of fields publish their results at `arxiv.org` (approaching 100,000 articles per year) and thus most get accustomed to using LaTeX.

Because TeX documents are in plain text files that are compiled (in the sense a C language program is compiled), it is relatively easy for TeX-based typesetting to be part of automated workflows—perhaps more easily than Word or InDesign document formatting can be. This capability will remain useful to various publishers and individuals.

Another TeX niche is among users who like to be able to open the box of products and systems to adapt things to their own needs or to make contributions that help a wider community.

The TeX systems range from exceptionally stable ((La)TeX) to being regularly updated with significant new developments (LuaTeX). Throughout that spectrum, users are constantly extending the systems to their own more or less sophisticated needs, as has always been the practice in the TeX world. The support community and capabilities remain extensive and vibrant. And most TeX distributions continue to be available for free.

LuaTeX is perhaps the most recent fundamentally new development in the TeX world. It has addressed several traditional complaints: along with XeTeX, it natively supports Unicode (UTF-8) input and TrueType and OpenType fonts; LuaTeX also provides embedded scripting closely integrated with TeX in the Lua programming language (traditional TeX macros are still available); and many of TeX's internal variables and algorithms are available to the user to inspect, override, and modify. The LuaTeX project is a good example of a long-term project, addressing perceived needs, driven by a few individuals who have dedicated much time over the years for personal reasons and for an interest in serving the community (and occasionally funded by user groups or other charitable sources of money to keep development moving).

Users and developers of TeX and related technologies are a self-selecting community, and the user groups and community development and maintenance projects are aimed at serving this

community. We foresee the use and development of the TEX systems continuing for many years for these users for whom TEX is the "right" system and for developers who enjoy working in the TEX world. The international contributions and collaboration started by Knuth 40 years ago will continue within this TEX community.

Renewed Importance?

In recent decades, TEX's original domain of math has taken a new step in importance to the world at large. Math is used "for everything" these days: analytics in sports, polling in politics, evidence-based medicine, "big data" in marketing and business planning, and so on. Today, 40 years after the creation of TEX and with TEX continuing to function as an outstanding way to produce documents including math, perhaps TEX has new potential outside its current community.

APPENDIX: HOW TEX IS EXTENDED

Plain TEX consists of hundreds of primitive commands, implemented in the TEX program itself, and hundreds more macro definitions. The primitive commands alone provide the fundamental input, output, and typesetting functionality but at too low a level to be convenient for a user to specify the typesetting of a document. The macros turn that base engine into a system that knows the conventions of typesetting. An important group of primitive TEX commands is for defining macros and the mechanisms for calling the new macros. These allow users to augment TEX.

There are many ways augmentation can be done. For instance, one can define new macros to augment plain TEX or redefine existing macros to change the operation of plain TEX. (Eplain is an example of a collection of macros that augment plain TEX, for instance, to add cross-referencing by labels.) In addition, Knuth put hooks into TEX to allow the system to work with other programs that TEX doesn't know about (and, largely, that came into existence many years after TEX was finished); for instance, images in various formats can pass through TEX, and TEX can read and write external files (other than the normal input and output).

LATEX, created originally by Leslie Lamport, is a set of macro definitions that reside on top of TEX's primitive commands, while reproducing many of the macro definitions available in plain TEX. LATEX is probably the most common form in which people use TEX, and thousands of macro packages have been written to reside on top of it or to modify its operation, for example, the `url` package that handles the unusual characters in URLs and tries to do sensible line breaking of them, and the `fancyhdr` package that supports nearly arbitrary page header and footer conventions. The AMS-TEX version of TEX is also implemented with macros (on top of plain TEX), as was the later-created AMS-LATEX (on top of LATEX, later merged into LATEX). A user can also add his or her own macros on top of everything mentioned—including making changes to the existing macros.

The TEX ecosystem includes various other levels: `tug.org/levels.html` identifies large collections of TEX-related software; front ends or editors that provide a development environment, provide sophisticated editing of TEX markup, or provide a graphical user interface; and extended TEX engines that provide new basic functionality at the primitive level. Various packages and engines also allow TEX to be connected to various high-level languages instead of the user being forced to program with macro definitions and calls—although some interaction with macros is all but inescapable.

The sustained effort of Hàn Thế Thành to directly produce PDF files from TEX (rather than by conversions from DVI to PostScript to PDF) is an example of how one project dealt with the various levels of TEX. It is also a notable example of how one person became motivated to undertake

a significant project and how other people collaborated on it over time, rather analogous to the original TEX project, albeit on a much smaller scale.

- - -

The TEX macro processor used in extending TEX is enormously powerful and flexible (and is a comprehensively documented piece of software).[7,8,9] There are explicit commands in TEX for creating local or global definitions, as well as various other definition variations, such as delayed definition and delayed execution of macro calls. TEX has a rich rather than minimal set of conditional and arithmetic capabilities (some related only to position in typesetting a page). There are also ways to pass information between macros and, more generally, to hold things to be used later during long, complicated sequences of evaluation and computation. These capabilities allow unlimited amounts of new code (programs) for extending or changing TEX to be written in the macro language.

- - -

Historically, there has been an interesting set of pressures around TEX's macro capability. Originally, Donald Knuth included only enough macro capability to implement his typesetting interface.

Knuth has made the point that he was designing a typesetting system that he didn't want to make too fancy, that is, by including a high-level language. He has also noted that when he was designing TEX he created some primitive typesetting operations and then created a set of macros for the more complete typesetting environment he wanted. He expanded the original macro capability ("kicking and screaming") when early users, particularly fellow Stanford professor Terry Winograd, wanted to do some fancier things with macros. Knuth's idea was that TEX and its macro capability provided a facility with which different people could develop their own typesetting user interfaces, and this has happened to a large extent, for example with LATEX, ConTEXt, and so on.

That expanded capability allowed users to construct nearly any logic they wanted on top of TEX (although often such add-on logic was awkward to code using macro-type string manipulations). On the one hand, TEX and its macro-implemented derivatives have always been very popular and there have been nonstop macro-based additions for over 30 years. On the other hand, users then and now despair at how annoying coding using macros is, moan about "why Knuth couldn't have included a real programming language within TEX," and otherwise cast aspersions on TEX's macro capability.

It is natural that Knuth used (unfancy) macros to extend TEX rather than high-level language constructions. Macros have been and still are traditionally used for user extension of editors and other text processing systems. Anyone can understand abbreviations substituting for common phrases of text or sequences of commands. We suggest that TEX's "problem" of not having a programming language to allow user extensions is a primary reason that TEX became so popular and has lasted so long and been built on top of by so many people (and developers, not just users) in so many major ways that in retrospect the decision to have macros and not a programming language can seem unfortunate. RUNOFF, Pub, and Script are gone so people don't complain about extensions to them using macros instead of a programming language.

PRIOR LITERATURE

We see the topics of this history as being in two worlds of literature: the world of digital typesetting and desktop publishing, where TEX resides, and the world of user groups and/or open source communities, where the TEX user groups and community of users reside.

An interesting place to start studying the history of digital typography is a recent book chapter by Jacques André.[10]

As with many popular applications, many books have been written about using (LA)TEX for various levels of users—from those wanting only an introductory tutorial, to those who want a comprehensive reference book, to those who want to know how TEX works on the inside. TEX is not exceptional in this domain, but as such books have been published for 40 years, they provide an interesting historical record. Anyone wanting more information about TEX books and journals might start by clicking on links at `tug.org`.

Regarding the literature on the history of word processing and desktop publishing, interesting starting points are the *Annals* special issue on word processing[11] and Pamela Pfiffner's *Inside the Publishing Revolution*.[12] The other articles in these desktop publishing special issues are another good source. James Cortada's *The Digital Hand* places desktop publishing in the context of computing in the print media industries.[13]

The literature of computer user groups is not particularly extensive. An early famous example of a computer user group is IBM SHARE.[14] Other mainframe and minicomputer vendors also had user groups.[15] Digital Equipment's DECUS user group meetings included presentations on LATEX and there was a style file creating the meeting proceedings using LATEX.

The TEX user groups, which deal with TEX and the systems derived from or built on TEX, are somewhat different from the user groups of the commercial desktop publishing systems (for instance, for InDesign). Those commercial user groups are typically about how to use a system. While the TEX user groups and community provide user help as the commercial groups do, they are also "open source" development projects, expanding TEX in many ways (although the open source term came into use more than 15 years after the first TEX user group was founded). With regard to TEX specifically, Gaudeul did an extensive study on TEX as an open source effort. The papers by Gaudeul include many references to the academic literature of open source software; for example, see section 6.2 of Gaudeul's 2003 article in *TUGboat*.[2] In this sense, the TEX world is also a lot like the Unix/Linux community.[16]

Of course, there is the meticulous documentation by Knuth of TEX, his development of it, and what he learned while developing it, and the related computer files. There are more rich veins to be mined by future historians.

See `tug.org/pubs/annals-18-19` for notes and additional reference.

REFERENCES

1. D. E. Knuth, "The Future of TEX and Metafont," *TUGboat*, vol. 11, no. 1, 1990, p. 489, `tug.org/TUGboat/tb11-4/tb30knut.pdf`.
2. A. Gaudeul, "The (LA)TEX project: A case study of open source software," *TUGboat*, vol. 24, no. 1, 2003, pp. 132–145, `tug.org/TUGboat/tb24-1/gaudeul.pdf`.
3. D. E. Knuth, "Remarks to Celebrate the Publication of *Computers & Typesetting*," *TUGboat*, vol. 7, no. 2, 1986, pp. 95–98, `tug.org/TUGboat/tb07-2/tb15knut.pdf`.
4. B. Beeton and R. Palais, "Communication of Mathematics with TEX," *Visible Language*, vol. 50, no. 2, August 2016, pp. 40–51, `tug.org/pubs/vislang-16/article.pdf`.
5. Y. Wang, "Interview with Charles Bigelow," *TUGboat*, vol. 34.
6. J. Hefferon and K. Berry, "The TEX Family in 2009," *Notices of the AMS*, vol. 56, no. 3, March 2009, pp. 348–354, `tug.org/pubs/notices-09`.
7. D. E. Knuth, *The TEXbook*, Addison-Wesley, 1986, particularly chapter 20.

8. V. Eijkhout, *TEX by Topic*, Addison-Wesley, 1991, particularly chapters 11–14, ctan.org/pkg/texbytopic.
9. D. Salomon, *The Advanced TEXbook*, 1995.
10. Histoire technique des fontes numériques, chapter 5, volume 2 of *Histoire de l'Ecriture Typographique—le XXième siècle*, tome II/II, de 1950 à 2000, editorial direction by Jacques André; Charles Bigelow has written an English summary of this chapter—Review and summaries: *The History of Typographic Writing—The 20th century*, Volume 2 (ch. 1–5), *TUGboat*, vol. 38, no. 2, 2017, pp. 274–279, tug.org/TUGboat/tb38-2/tb119bigelow.pdf.
11. *IEEE Annals of the History of Computing*, special issue on word processing, vol. 28, no. 4, October–December, 2006 (see also computer.org/web/computingnow/annals/extras/wordvol28n4.
12. P. Pfiffner, "Inside the Publishing Revolution: The Adobe Story," 2003.
13. J. W. Cortada, *The Digital Hand, Volume 2: How Computers Changed the World of American Financial, Telecommunications, Media, and Entertainment Industries*, Oxford University Press, 2006.
14. P. Armer, "SHARE—A Eulogy to Cooperative Effort," *IEEE Annals of the History of Computing*, vol. 2, no. 2, 1980, pp. 122–129.
15. H. S. Bright, "Computer User Groups," *IEEE Annals of the History of Computing*, vol. 10, no. 3, 1990, pp. 56–61.
16. P. Salus, *A Quarter Century of UNIX*, Addison-Wesley, 1994.

ACKNOWLEDGMENTS

We appreciate the great efforts and enthusiasm for over 40 years of the developers and users of TEX and its derivatives, starting with Donald Knuth. The following people answered questions over the past several years as we researched this history: Atsushi Akera, Gordon Bell, Charles Bigelow, David Brock, James Cortada, Dimitrios Filippou, David Fuchs, Burt Grad, David Grier, Hàn Thế Thành, Ferdy Hanssen, David Hemmendinger, John Hobby, Jerzy Ludwichowski, James Mason, Paul McJones, Doug McKenna, Richard Palais, Massimo Petrozzi, Norbert Preining, Lynne Price, Arthur Reutenauer, Jonathan Seybold, Christian Schenk, Petr Sojka, Guy Steele, Tommie Usdin, Boris Veytsman, Herbert Voß, Gerben Wierda, Joseph Wright, and Jeffrey Yost. Our apologies to anyone we have forgotten.

We are particularly grateful to the special issue editors for their encouragement and guidance and to the *Annals* editors and anonymous reviewers for their thorough reviews and helpful suggestions for improvement. We also appreciate the work of the Computer Society staff in converting our manuscript to its published form.

ABOUT THE AUTHORS

Barbara Beeton is employed by the AMS and has been involved with TEX since 1978. She has been a TUG board member since TUG's founding in 1980 and has been editor of *TUGboat* since 1983.

Karl Berry has worked with TEX since 1982, is production editor of *TUGboat*, a past president of TUG, and has been a key person in a variety of aspects of the TEX and TEX community infrastruc-

ture.

David Walden studies and writes about computing and digital typography history, including doing oral history interviews (75 from the TEX world and the others from the more general computing world).

Additional references and notes for TEX: A branch in desktop publishing evolution, Part 2

In the following, the number at the beginning of a note is a page number; words or a topic from that printed text page then appear indicating the position on the page to which the note or reference relates; then comes the note or reference itself.

2 "LATEX had come on the scene in 1983" One brief overview of LATEX is Open Source Documentation Software: An Overview, Shubhashree Savant and Sonal Sarnaik, International Conference on Advances in Information Technology and Management ICAIM, 2016, research.ijcaonline.org/icaim2016/number1/icaim201635.pdf

2 "surely contributed to their popularity" In *TUGboat* LATEX creator Leslie Lamport said, "I don't think TEX and LATEX would have become popular had they not been free. Indeed, I think most users would have been happier with Scribe. Had Scribe been free and had it continued to be supported, I suspect it would have won out over TEX. On the other hand, I think it would have been supplanted more quickly by Word than TEX has been."; *TUGboat*, vol. 22, no. 1/2, 2001, pp. 20–22, tug.org/TUGboat/tb22-1-2/tb70lamp.pdf

2 "local user groups continued to expand" Eric Frambach, TEX user groups worldwide—what's cooking?, *MAPS*, Autumn 2003, pp. 6–9, ntg.nl/maps/29/03.pdf

2 "local user groups continued to expand" Christina A. L. Thiele, The Future of TEX and TUG, *TUGboat*, vol. 14, no. 3, 1993, pp. 162–166, tug.org/TUGboat/tb14-3/tb40thiele-future.pdf

3 "DVI output to PostScript" Tomas Rokicki's dvips DVI post-processor, tug.org/dvips.

4 jTEX and pTEX ctan.org/pkg/ptex, https://ctan.org/pkg/uptex; also Haruhiko Okumura, pTEX and Japanese Typesetting, *The Asian Journal of TEX*, vol. 2, no. 1, April 2008, pp. 43–51, ajt.ktug.org/2008/0201okumura.pdf

4 ConTEXt format pragma-ade.com; contextgarden.net

4 "specification of PostScript programs" While PostScript is often thought of in terms of text fonts, it can specify any sort of drawing, and indeed, is a general-purpose programming language.

4 "in the TEX Live collection" ConTEXt is also widely distributed independent of the TEX Live collection.

5 "David Grier has said" Page 11 in the transcript of the Computer History Museum "PC Software Workshop: Marketing and Sales," recorded May 6, 2004, CHM reference number X4621.2008.

5 "TEX's initial development" Donald E. Knuth, *Digital Typography*, CSLI Publications, Stanford, CA, 1999.

5 "Ikarus developed by Peter Karow" Peter Karow, *Digital Formats for Typefaces*, second edition, URW Verlag, 1987.

6 "in various other systems" tinyurl.com/ms-blog-use-of-tex

6 "not as many as once did" Word has become a preferred manuscript submission format, and other systems (e.g., InDesign) are now good at fine typesetting.

6 "disproportionately used in less-developed countries" Glyn Moody, *Rebel Code: The Inside Story of Linux and the Open Source Revolution*, Perseus Publishing, 2001, p 317.

7 "algorithms developed for TEX" TEX has so much computer science in it that Victor Eijkhout built a university computer science course around TEX: Victor Eijkhout, *The Computer Science of TEX and LATEX*, based on CS 594, fall 2004, University of Tennessee, pages.tacc.utexas.edu/~eijkhout/Articles/TeXLaTeXcourse.pdf; no doubt other word processors and desktop publishing systems also had lots of embedded computer science—lexing, parsing, semantic interpretation, optimization of searches, etc.—but their source files may not have been so thoroughly documented or may not be available for study.

7 "line-break algorithm uses" Donald E. Knuth and Michael F. Plass, Breaking Paragraphs into Lines, reprinted in Knuth's *Digital Typography*, CSLI Publications, Stanford, CA, pp. 67–155; `TEXDR.AFT`, Chapter 24 of *Digital Typography.*

7 dynamic-programming TEX's line-breaking algorithm is routinely used as an example in computer science algorithms courses in explaining dynamic programming.

7 international collaboration on hyphenation See `hyphenation.org`; also Mojca Miklavec and Arthur Reutenauer, Hyphenation in TEX and elsewhere, past and future, *TUGboat*, vol. 37, no. 2, 2016, pp. 209–213, `tug.org/TUGboat/tb37-2/tb116miklavec.pdf`

7 "Knuth's boxes-and-glue model" Nelson Beebe, Using boxes and glue in TEX and LATEX, `math.utah.edu/~beebe/reports/2009/boxes.pdf`

7 "*hz* micro-typesetting method" Peter Karow has described the development of the *hz* in Digital Typography with Hermann Zapf, *TUGboat*, vol. 36, no. 2, 2015, pp. 95–99, `tug.org/TUGboat/tb36-2/tb113zapf-karow.pdf`; Karow describes the approach as "a justification per paragraph system — as described by Donald Knuth". Hermann Zapf described his ideas on micro typography in his paper about micro-typography and the hz-program, *Electronic Publishing*, vol. 6 no. 3, September 1993, pp. 283–288, `cajun.cs.nott.ac.uk/compsci/epo/papers/volume6/issue3/zapf.pdf`. We know the details of Hàn Thế Thành's pdfTEX development, but not of the development in InDesign. Both implementations got help from Zapf or his work: `wikipedia.org/wiki/Hermann_Zapf`; Hàn Thế Thành email of 2017-09-10; Hàn Thế Thành, An Experience from a Digitization Project, `cahiers.gutenberg.eu.org/cg-bin/article/CG_1998___28-29_197_0.pdf`; Hàn Thế Thành, Micro-typographic extensions to the TEX typesetting system, dissertation, Masaryk University Brno, Faculty of Informatics, October 2000, reprinted in *TUGboat*, vol. 21, no. 4, December 2000, pp. 317-434, `tug.org/TUGboat/tb21-4/tb69thanh.pdf`

7 Math layout algorithm Knuth's math layout algorithm (Appendix G of *The TEXbook*[??]) was also an innovation, but it has not been as widely copied.

8 TEX and PostScript Knuth created his own font format before PostScript was available. He developed his fonts with Metafont which generated bitmaps of characters, and TEX used the fonts out of Metafont, the characters of which could then be passed to a printer driver. This was in keeping with Knuth's desire to control his digital typography work down to the level of the pixels. The (later dominant) PostScript approach from Adobe provided outlines of characters to its raster image processor that sits between the PostScript interpreter and a printer. The TEX and Adobe approaches had little in common, and people other than Knuth later provided ways for TEX to use PostScript and PDFs.

8 "embedded Lua language interpreter" In several instances, someone has developed a way to escape from TEX to a programming language (e.g., Perl, Scheme), but these have not been widely used. There was a DANTE e.V.-sponsored effort to reimplement TEX in Java, but it was not practically useful. Now the LuaTEX team has created a useful version of TEX that keeps TEX's macro capability and adds the ability to escape to the Lua programming language.

8 John Warnock interview Knowledge@Wharton, Adobe Co-founder, John Warnock on Competitive Advantages of Aesthetics and the "Right" Technology, January 20, 2010, `tinyurl.com/wharton-warnock`

9 "easier to use and more productive than WYSIWYG systems" This two-part paper, including the figures and the Webnotes, was composed and revised with LATEX before being converted to Word for submission to the journal's prepress process.

9 "variety of (typically rather small") companies Early instances of these were MicroTEX, developed by David Fuchs, and PCTEX, developed by Lance Carnes. David Kellerman and Barry Smith provided commercial support of TEX on VAX/VMS systems. Barry Smith later moved on to the commercial TEXtures distribution for the Mac. Berthold and Blenda Horn's Y&Y TEX was another small commercial distribution of TEX for Windows.

9 Overleaf and ShareLaTeX `overleaf.com`

10 "a few small companies are making money" The Wikipedia article on "comparison of TEX editors" (accessed in November 2017) lists 49 editors for (LA)TEX, with eight of them requiring payment. Most of the 49 editors work at the source code level; five claim a mix of source-code and WYSIWYG editing; two claim to be WYSIWYG; four claim What-You-See-Is-What-You-Mean editing. (LyX, for instance, lets one edit graphically with menu commands to declare what lines of text are, e.g., title line, in-line equation, etc.; and the program then turns the text into a LATEX document out of view of the user.)

10 "advertising section of *TUGboat*" `tug.org/TUGboat/Contents/listkeyword.html#CatTAGAdvertisements`

10 LuaTEX project luatex.org

11 "Eplain is an example" Originally created by Karl Berry, tug.org/eplain

11 "reproducing many of the macro definitions available in plain TEX" In answer to the question of why didn't Lamport implement LATEX on top of plain TEX, we are not aware of any explication by Lamport, and could merely speculate; and it is good that he did replicate plain TEX's macro definitions as it allows users to escape out of LATEX to do things not readily possible in LATEX.

11 "sustained effort of Hàn Thế Thành" Interview of Hàn Thế Thành, tug.org/interviews/thanh.html

11 "notable example of how" Hàn Thế Thành, The PDFTEX Program, *Cahiers GUTenberg*, no. 28–29, 1998, pp. 197–210, cahiers.gutenberg.eu.org/cg-bin/article/CG_1998___28-29_197_0.pdf

12 "by including a high-level language" Knuth clearly believes in creating tools appropriate to the job at hand. For instance, one way of thinking of the Metafont typeface design system is as a programming language. Yet for that Knuth also developed a macro capability—more powerful in some ways and significantly different, as people writing mathematical character definitions have vastly different needs than people typesetting documents. Also, when Knuth created his literate programming system WEB which uses Pascal for the code, he included a macro capability at the WEB level, partly to get around some of Pascal's shortcomings for systems programming; when literate programming systems targeting C were later developed, C's own macro processor could be used.

12 "primitive typesetting operations" Primitive TEX is really basic. In addition to macros for adding new commands, special characters used by lexical analysis and character sets can be specified by users to be different than Knuth specified with plain TEX.

12 "kicking and screaming" Peter Seibel, *Coders at Work*, Apress, 2009, p. 597.

12 "user extension of editors and other text processing systems" For example, later versions of RUNOFF, TECO, troff, Script, and Pub—a prevalent text formatting system in the Stanford AI Lab where Knuth initially developed TEX. Indeed the SAIL language in which TEX was originally written supported macro definitions, and Knuth used them for TEX.

12 "world of user groups" en.wikipedia.org/wiki/History_of_free_and_open-source_software

13 "Gaudeul did an extensive study on TEX"

- Alexandre Gaudeul, The (LA)TEX project: A case study in open-source software, working paper, September 17, 2004 (quite different text than in the 2003 *TUGboat* publication).
- Alexandre Gaudeul, Competition between open-source and proprietary software: the (LA)TEX case study, working paper, January 6, 2005.
- Alex Gaudeul, Do Open Source Developers Respond to Competition? The (LA)TEX Case Study, March 27, 2006 (a different paper than the next paper with the same name).
- Alexia Gaudeul, Do Open Source Developers Respond to Competition? The (LA)TEX Case Study, working paper, March 2007, perhaps a preprint of a paper of the same name in *Review of Network Economics*, vol. 6, no. 2, June 2007, pp. 239–263.