Do Open Source Developers Respond to Competition?: The LaTeX Case Study[1]

Alex Gaudeul[2]

March 27, 2006

[1]I would like to thank William Adams, Jacques André, Nelson Beebe, Barbara Beeton, Karl Berry, Lance Carnes, Thomas Esser, David Fuchs, Bernard Gaulle, Hans Hagen, Yannis Haralambous, Jim Hefferon, David Kastrup, Donald Knuth, Leslie Lamport, Wendy McKay, Barry MacKichan, B. Mahesh, Frank Mittelbach, Oren Patashnik, Simon Pepping, John Plaice, Fabrice Popineau, Sebastian Rahtz, Denis Roegel, Chris Rowley, Joachim Schröd, Karel Skoupy, Hàn Thế Thành and all other participants in the (La)TeX project I met and interviewed. Special thanks are due to Hans Hagen for help in editing this paper. I would also like to thank Jacques Crémer, Bruno Jullien, Jean Tirole and Hal Varian for their support. All errors and omissions are mine.

[2]`a.gaudeul@uea.ac.uk`, University of East Anglia - Norwich and ESRC Centre for Competition Policy

Abstract

This paper traces the history of the development and use of the Open-Source (OS) TEX typesetting program. This software was developed over three decades and came into competition with a variety of open-source and proprietary alternatives. TEX was an early and very successful open-source project that imposed its standards in a particularly competitive environment and inspired many advances in the typesetting industry. While developers working under proprietary and open-source licenses responded to different sets of motivations, this did not mean they could abstract from each other's development decisions. The development of each type of software evolved in a symbiotic way. The strengths and weaknesses of both development methods were starkly revealed in the process. A pattern of semi-altruistic open-source development emerged, whereby developers considered non-developers' needs in order to make TEX more attractive to a broader audience and more competitive vs. proprietary alternatives. Self-use motivations for development were complemented by direct and indirect network effect considerations. This paper contributes to a better understanding of the long-term welfare effect of the emergence and growing importance of open-source development methods.

JEL Classifications: D23, H41, L13, L22, L31, L86, O34, O38.

Keywords: Open Source Software; Network Effects; Product Differentiation; Information Goods; Intellectual Property; Production Systems; Competition; Non-Profit; Volunteer Organizations.

Contents

List of Figures

This paper[1] offers an analysis of the competition between Open-Source (OS) and proprietary software systems through a study of the long term development dynamics of the OS TEX typesetting software.[2] TEX's development and its competitive environment will be analyzed on the basis of interviews with participants in the development of the TEX typesetting system, with administrators of the TEX Users' Group (TUG) and with maintainers of the project's organizational infrastructure. Proprietary versions of TEX and alternatives to TEX were released over time and came into direct competition with the original open-source project. Because the TEX project was launched in 1978, long before other better known OS projects such as Linux, this case study draws on a long history during which a wealth of OS and proprietary competitive strategies were observed. The (LA)TEX project provides an interesting natural experiment in which the drawbacks and advantages of public vs. private, open vs. closed, for-profit vs. not for profit development processes are starkly revealed. The paper will explain how the growth of TEX's user base came with a progressive diminution in the dynamism of the software's development; the weight of legacy users undermined attempts to adapt TEX to new uses, technologies and standards. As a degree of coherence was imposed onto the software's distribution[3] and attempts were made to structure and centralize development, centrifugal forces led to fragmentation in the development of TEX and in TEX's user base. There was 'forking' as developers chose to pursue their own projects separately. Coordination in the work of OS developers was difficult and the needs of users were not always congruent with the needs

[1]Previous versions of this paper include Gaudeul (2003a). That paper was extended in a Ph.D. thesis chapter (Gaudeul (2003b)). The case study gave rise to two theoretical papers, the first one analyzing the role of network effects in competition between proprietary and Open-Source Software (OSS) (Gaudeul (2005a)) and the other analyzing the Berkeley Software Distribution (BSD) license terms and comparing them with the General Public License (GPL) and the proprietary licenses (Gaudeul (2005b)). Current work includes a project to assess the positioning of OSS in the software industry.

[2]`http://www.tug.org`. TEX and (LA)TEX are used interchangeably to designate the whole of the project, although strictly speaking, TEX is the core of the typesetting program. Most users of TEX use the LATEX set of TEX macros, hence the term (LA)TEX.

[3]A software distribution is a collection of software, already compiled and configured, that when installed on a computer will constitute an entire self-contained software system.

of developers. Commercial ventures filled the resulting gaps in the TEX offering. Consumers bought proprietary products either for lack of an open-source alternative or because they needed support in their use of the software.

Proprietary products made TEX more accessible to the end-user and their existence motivated TEX developers into developing better user interfaces and additional functionalities. TEX developers maintained TEX's position in the marketplace by keeping up to date with changing technologies and standards and addressing a wider range of needs in an increasingly diversified market. The popularity of proprietary applications based on TEX increased its value as a standard and thus attracted additional developers to the program. A virtuous cycle was thus established in which commercial and open-source development reinforced each other.

Literature review This paper is a study of competition in the provision of a potentially excludable but non-rival good, software. When a good is non-excludable, then it is generally assumed it will have to be publicly provided. If it is made excludable, by instituting copyright law for example, then it is assumed it will be privately supplied in the same way as most standard goods. There is therefore little in the literature to understand the co-existence of 'public' and 'private' provision for the same type of good. This is however the situation in the software industry, where some software is provided under open-source licenses while some other software is provided under proprietary licenses.

One will see many similarities between the conclusions of this paper and those derived from the debate over the differences in the functioning of publicly funded and private institutions in health-care provision, cultural production or education.

However, the issues discussed in that debate do not translate immediately into the OS debate. While OSS is essentially a public good,[4] it is not usually pub-

[4]OS developer do keep the copyright over their contributions to an OS project, but are limited in its use; it can be asserted only to prevent others from using their contribution without acknowledgment (Berkeley licenses) or from using it as part of proprietary software (GNU Public License).

licly financed, does not rely to any significant extent on private donations for its development, and its provision is not directed by the State or any other formal institutions or interest group. Open-source projects are the result of the work of individually motivated developers, and it is difficult for any institution to direct their development.

This paper thus contributes like Schmidt and Schnitzer (2003) to the re-examination of some of the questions that were raised in the previous literature in the specific context of open-source provision: how do OS organizations react to competition, OS or otherwise? More generally, how does an open-source organization adapt to changes in its technological and commercial environment? How do commercial organizations adapt to the presence of OS competition? How and why does each type of organization specialize in specific areas of software development, or, in each development area, on serving different types of customers or working on different parts of software? Are there exchanges and synergies between each type of organization?

The existing literature on OS methods of production, for example Raymond (2001), Lerner and Tirole (2002) or Lerner and Tirole (2005), does not elaborate on those topics. Case studies do not generally go into analyzing the competitive dynamics of open source development – see for example Franke and von Hippel (2003) who studied the Apache webserver, Mockus, Fielding, and Herbsleb (2002) for the Mozilla web browser, Casadesus-Masanell and Ghemawat (2003) or Mustonen (2003) who, like many others, take the Linux operating system as a reference point. Existing work focused on other problems: Bonaccorsi and Rossi (2004) and Hann, Roberts, Slaughter, and Fielding (2004) examined OS developers' motivations. Kuan (2002) and Nichols and Twidale (2003) assessed the quality of the OS product and of its user interfaces. Kuan (2001) and von Krogh, Spaeth, and Lakhani (2003) explained the functioning of OS organizations.

Koenig (2004) does offer a typology of the strategies of proprietary software firms as they react to emergent competition from OSS. Wichmann (2002), Bitzer (2004), Mustonen (2005) or Bitzer and Schröder (2005) analyze different aspects of the coexistence and competition between OS and proprietary software. There is little work however that explains the dynamics of competition between OS and proprietary software. Most existing models are static, and even Gaudeul (2005a), which was inspired by the (LA)TEX case study, does not model the multi-period strategies by which OS and proprietary software borrow from each other, complement each other and react to advances on the competing side.

Outline This paper is introduced by a presentation of the TEX software and of the actors and institutions that influenced its development. The development of the software is analyzed over time, which leads to a discussion of how TEX positioned itself in its environment. The paper concludes with the consideration of the strategic motives in the development of OS and proprietary software.

1 TEX, the software and its history

This part presents the TEX project. A brief chronology of the development of TEX and of its competitive environment will be offered (See also appendix D p. 33). The evolution of the organizational structure of the project will also be analyzed.

TEX is a computerized typesetting system, meaning that a TEX file contains both the text to be published and instructions on how to format it for an output (dvi, ps, pdf, html, or xml files) that may be distributed in digital or paper form. The TEX project was started by Donald E. Knuth in the late 70s at Stanford University. As part of this project, TEX was developed to typeset mathematical and scientific articles while Metafont was developed to design fonts. I will focus in this paper on the development of TEX and will only mention in passing the parallel development of Metafont and other associated programs.

TEX was developed by D.E. Knuth in order to satisfy his need for software that would provide good quality computer typesetting on any computer system and would remain stable over time so documents typeset with TEX could keep being read and compiled in the future. That software thus had to be sufficiently well documented that ulterior developers would be able to understand it, and sufficiently flexible that programmers could fit it to their own needs by using simple change files rather than by changing the existing source code.

TEX was meant to be easy to use for people without any programming background and many academics indeed typeset their papers with TEX. TEX is being used for a wide variety of tasks; the TEXinfo format is the official documentation format of the Free Software Foundation (FSF); TEX became the standard submission format for most mathematical journals and is commonly used for scientific papers; TEX is widely used in the publishing industry in the communication of printed work between authors, editors, and printers; TEX is frequently used as a back-end application in publishing, taking as input documents typeset in various markup languages (HTML, XML, TEX) and outputting documents in various rendering formats for printers or for the web. Finally, those typesetters whose needs are not fulfilled by a proprietary offering, those who cannot afford proprietary tools and those with very specialized needs appreciate the flexibility, versatility and free availability of TEX systems.

The concept of 'open-source' did not exist when TEX was developed so the program written by Prof. D.E. Knuth does not properly fit into any of the present categories. The concept of "free software" (1984, GNU Project by Richard Stallman) predated the concept of "open source software" by 14 years (1998, Open Source Initiative by Bruce Perens and Eric S. Raymond), and TEX predated "free software" by six years (1978). The core of TEX, tex.web is copyrighted by Knuth, and no changes are permitted, although of course concepts used in the program may be re-used. 'TEX: the program' is in the public domain, although the use of the name 'TEX' is restricted to exact copies of Knuths software systems. The license that is now used for most TEX programs – meaning most programs that have added up to 'TEX: the program' – is the LATEX Project Public License (LPPL). The LPPL can best be described as a Berkeley Software Distribution (BSD) license as closed-source proprietary versions of the software are allowed. However, any change to a file must be renamed and distributed with the original version, so as to guarantee the integrity of the system. This way, any user has access to the exact same program as any other user and can thus output the exact same result from a .tex file as that other user. To complicate matters further, binaries in TEX distributions are licensed under the General Public License (GPL).

TEX is a medium size software project; not as big as an operating system like GNU/Linux, but still a whole typesetting system with many interdependent modules. The initial version of TEX (1979) gained about 100 to 1,000 users, the second version (1982) had about 10,000 users while TEX was used by more than one million by the time its third version was released in 1990. TEX is widely available as it was ported to the Linux[5], Mac and Windows platforms, among others. The extent of the diffusion of TEX is reflected in the number of requests for support in TEX-related newsgroups, 4500 posts per month in 2000 across three newsgroups in English, German and French.[6] The creation of TEX newsgroups into a wider variety of languages is another testimony to the international diffusion of the software, as is the creation of Local Users' Group (LUG)s across the world.[7] There were about 6500 active members in a dozen national TEX users groups across the world in the early 2000s.

[5]It is estimated there are about 30 million users of the Linux operating system (`http://counter.li.org/estimates.php`).

[6]Data collected from the Google newsgroups search engine. From its creation in early 1990, posts to `comp.text.tex` gradually grew to 2000 per month in 1994 and stabilized since. As postings to the English-speaking newsgroup reached a plateau, posts to newsgroups in other languages grew. The German `de.comp.text.tex` started in 1996 and equalled the English group by 2000, while the French `fr.comp.text.tex` which started earlier (1994) received 500 messages per month by 2000. TEX newsgroups are primarily user and not developer oriented, while developers primarily participate in mailing lists. By way of comparison, the most active user oriented Linux newsgroup receive about 10,000 message per month, while ones that are developer oriented receive about 500.

[7]The first TEX users group was established in the US in 1980 and now counts 1800 members. It was followed by several LUG first in Western Europe and then in Eastern Europe and Asia: Netherlands (1988, 300 members), Germany (1989, 2000 members), France (1989, 500 members), the UK (1992, 150 members), Poland (1993, 300 members), the Czech Republic (1995, 400 members), India (1997, 100 members) and China (1999, 700 members), along with many smaller groups

Chronology of the development of TEX The development of the TEX system[8] can be divided in three eras/areas: an emphasis on the development of the core (early 80s)[9] was followed by an emphasis on the development of TEX sub-routines, i.e. sequences of TEX commands that eased access to TEX's functionalities (late 80s).[10] That period was followed by a priority given to the development of tools and systems that made the TEX system easier to use and develop (early 90s).[11] Usability was improved by making TEX systems available as a complete set of interrelated TEX packages ('distributions'), and by developing programs that allowed easy access to the various TEX programs and commands ('interfaces'). Development became better coordinated thanks to the creation of coding banks ('software packages repositories') and by the creation of a standard way of classifying those packages into the repository ('directory structure').

[8]Appendix C p.32 lists the most important TEX sub-projects, ranked by their date of first release.

[9]The core of TEX went through three versions, TEX1.0 in 1978, TEX2.0 in 1982 and TEX3.0 in 1989, each a progressively smaller marginal improvement on the other. By 1990 then, the development of TEX was essentially stopped and by the mid-90s, limitations in TEX became more apparent. Various projects to extend the capabilities of TEX emerged: Ω (1994) for multilingual typesetting, pdfTEX (1996) for pdf rendering, ϵ-TEX (1997) and the 'New Typesetting System' ($\mathcal{NTS}$) (1998) that made TEX more flexible. ϵ-TEX is now the standard TEX extension while pdfTEX is currently the most popular TEX package.

[10]Early sets of macros such as plain-TEX (1978) and AMS-TEX (1981) were soon replaced by LATEX (1983-1985). LATEX's development was taken over by the LATEX3 team (1989), a group development effort that led to the release of LATEX 2_ε (1994), while a set of macros for educational publishing was developed (ConTEXt, 1994).

[11]In that third phase, the TUG (1980), along with LUGs, set up a common repository for TEX packages (Comprehensive TEX Archive Network (CTAN), 1990) and a standard way of organising all the TeX-related files on a computer system (TEX Directory Structure (TDS), 1994). Several other initiatives gave coherence to TEX systems, such as the New Font Selection Scheme (NFSS) (1989) that allowed easier access to the wide variety of fonts that had been developed over time. Complete distributions were made available. TEX Live (1996), the OS LATEX distribution based on teTEX for UNIX distribution (1994), was translated into fpTEX for Windows Distributions (1998), a distribution that came to replace the independently developed MikTEX Windows distribution (1996). Interfaces were also developed, the most popular being now the freeware WinShell (1998) which competed with the earlier shareware WinEdt (1993). Finally, the LPPL, a formalization of existing TEX license agreements, was imposed on most packages in a LATEX distribution (1999). That license was adapted to the Debian Linux OS guidelines (2002) so as to facilitate the inclusion of the TEX system into Linux distributions.

Organizational phases in the development of TEX The way the development of TEX was organized changed over the years. In the first phase (1978-1985), development was heavily centred around Donald Knuth at Stanford University. He set the goals for the developers who worked with him and suggested directions of development to the independent developers who soon became interested in his project. He also welcomed suggestions for development, notably from the American Mathematical Society (AMS). In a second phase (mid-80s), core development was stopped. A number of packages were developed in a rather decentralized way, leading to many replications in the development of similar features. The suite of programs became more complex and the number of programs linked to it rose. The programs came to fit increasingly specialized needs: programs to generate bibliographies (BibTEX, 1984) and indexes (makeindex, 1987) or to draw figures (PSTricks, 1992) were developed. In Europe, local user groups initiated adaptations of macro packages to non-American typesetting traditions; TEX the program was adapted to deal with 8 bit input needed for non English languages (TEX 3.0, 1989)

Routines to translate TEX in various alternative typesetting markup languages (HTML, XML, SGML...) and convert its output into various document formats (BITMAP, POSTSCRIPT, PDF...) were developed. User interfaces became more sophisticated. Several different distributions of TEX competed on the market, each maintained by different persons and distributed under different licenses: freeware, OS, proprietary. Several macro packages were developed for TEX but the LATEX package rapidly became the most widely adopted. This stage of relative anarchy ended when users progressively came to coordinate on using the same distribution. That was the third stage (end-80s, beginning 90s) in which the TEX Users Groups worked to offer a distribution (TEX Live), structured along a standard (TDS) and available on a common repository (CTAN). This was a stage of rationalization and of integration. The fourth and present stage (90s to present) is characterized by a return to decentralization. The need for a central dedicated coordination system became less urgent as Web based tools made it possible to efficiently coordinate

OS projects in a decentralized way. The software came to be used in a wider variety of countries and by a wider variety of people. The software also came to be called upon for the fulfillment of needs that the original developers had not anticipated and that the standard TEX software couldn't respond to adequately. This was why there were several attempts at remodeling the core of TEX along new lines, with various levels of success.[12]

The next part explains further the functioning and limits of the OS organization so the reader will better be able to understand the patterns in the competition between OS and proprietary software in part 3.

2 The dynamic of TEX's users' and developers' organizations

The TEX Users' Group (TUG) was established in 1980, or only two years after the inception of the project. It first relied on contributions by US institutions (The AMS, universities, typesetting companies) and then came to rely on individual users participation in a broader array of countries.[13] The TUG and other LUGs

[12]One will later speak further about such projects as $\Omega(1994)$, pdfTEX (1996), ϵ-TEX (1997) or NTS (1998)

[13]The AMS was the TUG's originating sponsor; it provided a large part of the initial budget, notably providing personnel and office space. The AMS wanted an OS Document Preparation System because it didn't want the scientific community to use different and (necessarily?) incompatible proprietary systems. While the AMS didn't fund Knuth's research programme, it paid a developer to develop a set of macros for TEX, AMS-TEX. This sponsoring by the AMS had an impact on the development of TEX that was far beyond the means involved as it started a bandwagon effect and other sponsors (hardware companies and universities) came to join and finance the TUG. The AMS and other sponsors progressively lowered their financial contribution to the TUG after the release of the final version of TEX in 1990. The TUG's revenues and membership also declined because of the competition with LUGs for individual membership. Starting with revenues of $75,000 in 1984 and 700 members, the TUG revenues rose to $650,000 by 1990 and membership to 4,000, followed by as quick a decline to $100,000 and 1,500 members by 1997, after which revenue stabilized and membership rose to 1,800. After the AMS and other institutional sponsors became less heavily involved in supporting the development of TEX, lay' users came to have more influence on the dynamics of the development of TEX . LUGs led their own initiatives (nationalization of TEX, distribution of TEX packages, development of users' interfaces, etc…)

encouraged projects that were of interest to all users: extensions of the software's core, compatibility with alternative standards (PDF, HTML, XML), rationalization of the TEX distributions, standardization of the most popular packages. It did not get involved in specialized projects that were of interest only to a few or could lead to problems of compatibility between versions of TEX: radical restructuring of TEX, development of What You See Is What You Get (WYSIWYG) interfaces. This focus on compatibility and serving the existing users discouraged a widening of the user base, but solidified the existing user base. The TUGs preferred to improve the existing product so as not to scare away existing users. The preference of established developers for gradual improvements led to dissensions as some developers wanted to change TEX more radically in fear that it would eventually become obsolete. These two strategies, radical changes vs. gradual improvements, were alternative ways to guarantee the continued growth and renewal of TEX's user community.

This part will first present TEX's user organizations and their influence on TEX's development. It will then present the main issues that faced those organizations. The two main roles of the TEX user groups were to foster collaboration between developers and make TEX popular by providing ready to install distributions of TEX, training users and encouraging the development of users' and developers' interfaces.[14] Limited means meant only incremental improvements could be supported, while limited central authority meant it was difficult to impose the tight coordination between developers and agreement on objectives that were necessary for the building of user interfaces. It was also difficult to reconcile the needs of all components of TEX's large user base (students, academics in mathematics or in language departments, professional typesetters). Many developers programmed for their own idiosyncratic and punctual needs, and did not provide continued support to those who used their packages. The LATEX3 team, which took up the development of LATEX, thus had to define a core set of TEX

[14]Users' interfaces are meant to make the product easy to install, use, keep up to date and upgrade, developers' interface are meant to facilitate coordination and collaboration between developers (Conferences, code repositories, standard classification of packages, etc).

macros for which it guaranteed continued support. Authors of the most popular TEX packages were encouraged to designate a successor when they decided to quit active development.

2.1 Building an interface

There were three stages in making TEX practically available to different groups of users: in the first stage, the raw programs were developed: these were usable only by sophisticated users. In the second stage, the programs were integrated into distributions. These linked many programs together and required that each program be formatted along some guidelines. Not all programs were integrated into the TEX Live distribution for example, as not all developers were motivated to make the changes that were necessary for this to happen – conforming to the distribution guidelines, changing the package's license to conform to the LPPL, etc. . . In the third stage, user interfaces[15] were built on top of the distribution. This is also the stage when a quick way to install the distribution on mainstream operating systems was provided (TEX Live, 1996).[16]

[15]The main interfaces were the freeware WinEdt (1993) and the OS WinShell (1998) but they required knowledge of the TEX syntax and commands. Some developers tried to go further and develop WYSIWYG interfaces based on their own implementations of TEX (LYX (1999), TEXMacs (2002)). This disconnected them from the main branch in the development of TEX. There were therefore individual OS initiatives trying to adopt the middle ground (TEXshop (2000) for Mac Operating Systems, preview-LATEX (2001) for Unix OS or XEMTEX (2003) for Windows OS). Those quasi-WYSIWYG interfaces could be used on top of the standard TEX distributions.

[16]The European LUG were instrumental in making TEX more accessible to the users. A group of volunteers at Aston University in the UK established a central repository for TEX code from which it was possible to download the latest developments in the TEX system via the Net. This group inspired the development of a package classification system, the TEX Directory Structure, which served as the model for TEX distribution everywhere. The Aston initiative led to the creation of the CTAN archives in 1990. Another seminal initiative came from the Netherlands LUG which developed 4AllTEX in 1993. 4AllTEX was a TEX distribution on a CD that was intended for an end user with no programming background. It was a precursor and an inspiration for the development of TEX Live. Designing an user interface for TEX required a stable core with which other modules would have to be compatible. The LATEX3 team (1989) thus defined a 'core' LATEX and committed to preserve compatibility between every of its components. The development of the TEX Live distribution also helped to focus energies by defining what would be distributed more widely to the end-users.

Producing a user-friendly, mass-market program required a different turn of mind than that of most, technically oriented, TEX developers. It was difficult to coordinate developers efficiently enough to release a fully functional pre-packaged product, and there were limited ways to communicate efficiently with end-users and design good documentation for the software. Despite these difficulties, TEX managed to be relatively accessible to the common user, for three reasons:

The first is that it was developed in a domain that did not hold special interest for developers. This is not to say that the development of TEX was not interesting from a programming point of view, but TEX was not meant to be used by developers but by authors and typesetters. From the beginning, the main motivation for its development was for the software to be used as widely as possible so as to improve standards in the typesetting industry. Developers who took up the task of development did so out of a desire either to make use themselves of the software, or to provide a working version to colleagues.

The second is that TEX was meant to become a standard and forking was discouraged in several ways. Developers were disciplined into producing software that complied with TEX specifications as D.E. Knuth devised some tests (TRIP/TRAP) that a program had to go through in order to have the right to call itself a TEX implementation. TEX thus did not suffer from the coexistence of several competing versions from which users would not know which to choose.[17]

The third is that TEX's BSD-like license allowed the release of proprietary implementations of TEX. Developers were thus able to provide (proprietary) TEX engines, editing environments and/or graphical user interfaces (PCTEX and MicroTEX (1985), TEXtures (1989), Y&Y (1991) or Scientific Workplace (1992)). Many of these were/are compatible with the Windows and Mac systems; systems that the OS community generally shunned in favor of Unix or, later, Linux, but that were popular with the general community of computer users. These propri-

[17]The way TEX was coded also contributed to its stability. TEX was coded in WEB, a subset of Pascal. The final code was monolithic and makes extensive use of global variables. This makes writing extensions difficult, especially for todays' developers who are used to program in C in a modular way.

etary systems thus contributed to making TEX more popular, even if not under its OS version. It was only late (1996) that OS Windows distributions appeared: MikTEX is the most popular OS user-friendly Windows distribution. TEX Live's fpTEX (1998) was also a popular TEX distribution for the Windows system because it guaranteed compatibility with the leading Unix distribution, teTEX (1994).

2.2 Coordinating development

TEX and LATEX were the two systems that historically attracted the most collaborative work. They were subject to many potentially contradictory sets of requirements that had to be reconciled. Because TEX was meant to be standardized and easy to use by non-programmers, it was not desirable for different versions to be developed for all different sets of TEX users.[18] TEX documents had to be easy to share for it to be adopted widely and therefore each package in TEX had to follow some standard so as to fit well into TEX distributions. Good coordination in the development of TEX was thus very important . Developers had to make sure new coding was not going to clash with new coding by others. This was particularly important because most TEX users were not able to solve themselves problems of conflicts between packages when installing TEX.

In order to promote contacts between users and developers and thus ensure development responded to needs, the TUG led a process of consultation of the user base and provided sponsorship for the work of some developers. The TUG encouraged consultation between TEX developers and professionals from the publishing industry.[19]

[18]In that way, TEX differs from Apache, the other main BSD-licensed software in the open source literature, where such tensions were dealt with by developing several specialized versions of Apache.

[19]Members of the publishing section of the AMS spent time in Stanford developing TEX macros to test TEX capabilities. Their work led to a series of suggestions for improvements for the generation of indexes notably. The LATEX3 project members also consulted with the AMS, various LUGs, publishers such as Addison-Wesley or Elsevier, and got support from companies, some that sell/sold TEX-based software – Blue Sky Research, TCI Research, PCTEX – but also from the computer hardware and services industry – Digital Equipment Corporation, Electronic Data

The TUGboat, official journal of the TEX community, collected users' initiatives and outlined the problems they encountered. Several programmatic papers outlining goals and priorities for the development of TEX were published in the TUGboat and presented at TUG conferences. Donald Knuth (Knuth (1991)) and the LATEX3 team (LATEX3 project team (1997)) expressed the wish that TEX and LATEX be typesetting standards. This contributed to discourage independent development. It took time for example to accept that ConTEXt could act as a full featured replacement for LATEX. That wish was translated into the license terms for those programs. Mittelbach and Rowley (1997) set out the program for the reworking of LATEX by the LATEX3 project team. Their priority was the development of better user interfaces. The Netherland LUG (NTG TEX future working group (1998)) called for more cooperative development efforts in the reworking of TEX. Flynn (2001) set priorities for the broadening of TEX's user base. Beebe (2004) advocated the adoption of the new OS markup languages rather than TEX's native language, etc... The debate over whether TEX should develop Windows-like, WYSIWYG interfaces was particularly heated (Cottrell (1999)).

The OS organization met its limit when it became necessary to rewrite the core of TEX so as to make it more flexible and adapt it to new requirements. It soon became clear it was not possible to lead such a project with people linked only through electronic means; it was necessary to get developers to work full time on the project in one location and in constant communication, the same way TEX was originally developed. This task was taken up by the NTS team (1993) and sponsored the German LUG. Disagreements in the committee responsible for overseeing this project slowed down its progression and when the result was finally delivered, many users had lost interest in TEX or had adopted temporary fixes. Many preferred to base their future use of TEX on other, less ambitious, extensions of TEX such as pdfTEX (1996) which made TEX compatible with the new, Adobe-led, Portable Document Format (pdf) technology.

Systems... Users' wishes were gathered from the LATEX discussion lists and newsgroups and led to improvements in the page layout specifications and the user-interface design.

The next, final and most important section of this article outlines how elements introduced in the previous parts translated into an original pattern of competition between TEX and its OS and proprietary competitors.

3 Competition with proprietary software

3.1 A map of the industry

TEX had to face intense competition from an array of competitors, from simple word processing software to professional publishing software.[20] First aimed at professional typesetters, TEX's superior quality (better fonts, better hyphenations and justification algorithms, better rules for letter spacing) drove out less advanced proprietary competition from which it borrowed many ideas and features as well (Script (1970), Roff (1971), Scribe (1978)). It also replaced in-house proprietary typesetting systems such as the one that was used by the AMS for mathematical typesetting.

TEX's code was then integrated into proprietary offerings. Distributions like pcTEX (1985), MicroTEX (1985), TRUETEX and Y&Y (1991) were/are pre-packaged and easy to install. Environments such as TEXtures (1989) for the Mac or Scientific Workplace (1994) for Windows provide(d) an interface that is easy to interact with. Extensions such as vTEX added functionalities to open-source TEX systems. Some companies (Arbortext (1982), 3B2 (1986)) borrowed parts of TEX. TEX's conception principles, line breaking algorithm or syntax were widely copied. TEX's equation editor was integrated in MS's Word (1984) and TEX's hyphenation and justification algorithm was used in Adobe's InDesign (2000). Finally, some companies provided additional font packages and printer drivers for TEX systems. With the exception of Scientific Word and PCTEX, the proprietary competition did not maintain itself over the long term as OS versions caught up

[20]Appendix B p.30 categorizes the different types of software that came in competition with TEX, depending on whether they were proprietary systems based on TEX, independently developed proprietary systems, independently developed OSS or OSS inspired by TEX.

with it. TEX Live and MikTEX provided for free what many proprietary software sold. Graph 2 p. 32 illustrates that process. TEX was then itself under the threat of extinction as easier to learn proprietary software with superior interfaces took a large part of the market in a two pronged attack: some proprietary systems fulfilled the needs of professional typesetters who needed easy to learn standardized software (Framemaker (1986), Quark's QuarkXPress (1987)). Some others satisfied the needs of lay users who only wanted easy to use simple text processors (Corel's WordPerfect (1982), Microsoft's Word (1984)).

Independently developed proprietary software came to replace TEX in its less specialized, general typesetting functionalities. Adobe consolidated the typesetting software industry in the mid-90s on the strength of its proprietary Portable Document Format technology (1993). That technology replaced Adobe's Postscript format (1985) which itself had replaced TEX's DeVice Independent (DVI) format (1979). Publishers adopted new OS markup languages (SGML (1986), XML (1998), MathML (1999), XHTML(2000)) rather than using TEX markups that suffered from lack of standardization and can be processed only by TEX.

TEX kept those users who could not afford proprietary typesetting software (users in emerging countries, students), those people who needed its unique mathematical capabilities (academics) and those typesetters who valued the flexibility OS offers to devise their own specialized versions of TEX: Ω (1994) for multilingual typesetting was developed to typeset rare script and ancient documents with complex typesetting, ConTEXt (1994) was developed to produced interactive educational material. TEX was one of the first high end typesetting programs to be adapted to handle new markup languages such as XML, and many typesetters keep using it as a back-end to handle documents typeset in those new languages.

Graph 1 p. 19 illustrates the above paragraph. The three phases in the relation of TEX to its competition are outlined (dominance, squeeze and fragmentation) and the main actors in each phase are represented. At the top are represented high end applications for typesetting and publishing, at the bottom are common applications for the end user (word processors). The right hand side illustrates the fragmentation in the development of high-end TEX applications, as well as the development of integrated TEX systems and interfaces for the end-user. Development at that stage was mainly influenced by the emergence of open typesetting and document rendering standards and the drive to develop user friendly open source word processors.

The next two sections explain the patterns in the competition between TEX and its proprietary alternatives; the composition of their respective user base and their fluctuations over time. As developers in each type of organization were driven by different kind of motivations, there were differences in their respective organizational objectives and the way developers became aware and responded to users' needs.

3.2 Market responsiveness and interface development

While the TUG encouraged the diffusion of new TEX developments to a broad audience and thus prevented unnecessary replication in the development of new features, it cared mostly about existing users and improving the existing software, not necessarily finding new domains of application or increasing TEX's market share. It also lacked the means to conduct radical restructuring of TEX; the $\mathcal{NTS}$ project was not successful (ref. p.15). In contrast to the $\mathcal{NTS}$ approach, pdfTEX was developed through small extensions of TEX and focused on practical typesetting problems. This contributed to now making it the most widely used TEX engine.

This meant that proprietary developers were often first to spot new market opportunities and act on them. While proprietary developers were quick to the market, they found it difficult to adapt to its evolution: proprietary software could not be as versatile in its applications as OSS because of two effects: their sophisticated user-interface made them difficult to change to integrate new functions, and proprietary developers were somewhat shielded from external influences as users of their software couldn't, unlike OS users, choose to develop their own versions

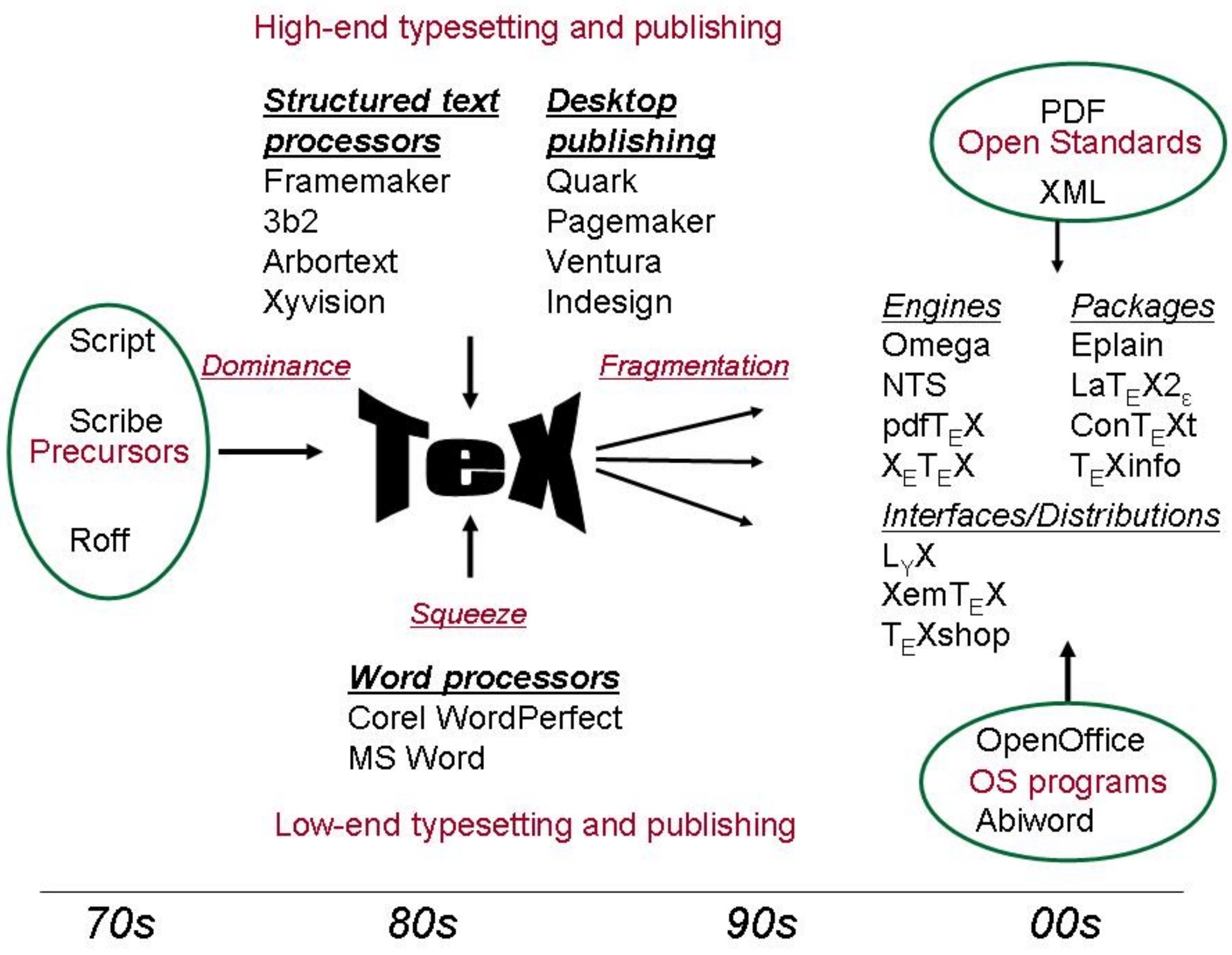

Figure 1: The development of TEX over time: Dominance, squeeze and fragmentation.

of their software if they were not satisfied. Proprietary development, in contrast to OS development, was directed toward providing new functionalities so as to attract new segments of the user population, at the expense of improving existing functionalities that were of use to their existing, partially locked in users.

This tendency led proprietary firms to develop user-interfaces to TEX rather than improving its core functionalities. OS developers on the other hand focused on developing new functionalities, and indeed frequently anticipated the need for some features; they acted as precursors. Quite often however their needs were too specialized and of no interest to the common user, while on the other hand, what would have been interesting for the common user was of no interest to them. They for example were not interested in a user interface as they already were comfortable with the TEX language. There was also considerable reluctance in considering the needs of Windows users; TEX Live was always considered to be a superset of the Unix teTEX distribution. Changes in that distribution always took precedence over changes in the fpTEX distribution for Windows, which led to this later distribution being abandoned.

This is how commercial software found a relative advantage in facilitating the use of TEX by adding a user interface and porting TEX to non-Unix platforms. Proprietary software based on TEX was also frequently able to offer new and original features that were in demand but outside of the realm of interest of existing TEX users. TEXtures catered to the Mac users, PCTEX catered (and still caters) to the PC users, Y&Y provided sets of proprietary fonts outline, Scientific Workplace integrates a computation software (Maple and then MuPAD). This relative advantage was only transitory though, and the OS TEX organization did come to integrate new font sets, offering a complete easy to install distribution and also pdf format output. TEX Live, the OS distribution of TEX, and graphical user interfaces such as the shareware WinEdt or the freeware WinShell spelled the end of many proprietary versions of TEX. [21] But there was clearly a difference in emphasis

[21]While OS interfaces improved over time, they never became as aesthetically pleasing, complete or ergonomically efficient as that of competing proprietary WYSIWYG text editors such as Microsoft's Word, of competing proprietary typesetting software such as Quark, or of some

and in financial means that resulted in these improvements coming later into standard OS use than in proprietary software based on TEX. Figure 2 p. 32 illustrates further the 'catching-up' dynamics of interface development. After TEX Live was released, most users of TEX came to use either TEX's OS version or Scientific Workplace. Only these proprietary software that included features that the OS community was not able to develop, such as the WYSIWYG equation editor of Scientific Workplace and its integration of the Maple computation program, were able to survive. The fate of independently developed software was less affected by improvements in OS TEX.

The fate of entrepreneurs who chose to borrow from TEX and that of others who chose to innovate independently of OS development differed markedly. Entrepreneurs played a two-stage game. In the first stage, the advantage of borrowing was that it economized on development cost and, as TEX was already established, offered users of the proprietary product the benefit of being able to use an established standard. In the second stage, the drawback of borrowing was that TEX caught up with the improvements on the OS product the entrepreneur made in the first stage. TEX benefited from being licensed under the BSD. Proprietary products often served as a Trojan horse for open-source products. The enhanced diffusion of TEX thanks to its better marketed and more accessible proprietary versions allowed it to impose its standard on the industry. Users of the proprietary versions of TEX, which TEX would not have reached if it had been distributed only in its OS version, switched to TEX when the first stage advantage of proprietary versions weakened. Most firms that based their strategies on offering augmented versions of TEX were thus driven off the market when the OS TEX distribution caught up with them in terms of ease of use and functionalities. Commercial developers who did not differentiate from TEX (producing niche products, in areas which OS development neglected or easing access to the newest developments in TEX) only gained a fleeting competitive advantage while the more expensively independently developed software could better maintain its position on the market,

software based on TEX such as Scientific Word.

eschewed TEX's technological limitations and did not depend on the continued development of TEX for its own survival.

3.3 Organizational inertia and network dynamics

3.3.1 Organizational inertia

The main problems facing the TEX organization were the inertia in the development of TEX which led to the fragmentation of its user base and of its development groups. That fragmentation opened opportunities for commercial developers whose role was to make a coherent whole from OS development efforts or fill in its missing features.

TEX's development was dominated by the original set of users of the software, which meant new or potential new users found it difficult to influence the dynamics of the software's development. TEX's license forbade different versions of TEX from calling themselves TEX. This meant there was only one version of TEX which was controlled by its original developer, D.E. Knuth. The same pattern was repeated in the case of LATEX, the dominant set of TEX macros. First controlled by its original developer, Leslie Lamport, its control was then transferred to the LATEX3 team. This tight control made it difficult for outsiders to influence the development of new versions of TEX or LATEX, but ensured coherence in the development of (LA)TEX.

The TUGs had an ambiguous influence on development, both encouraging better coordination between developers and discouraging development into new areas which would have required adaptation by "legacy" users. One of the TUG's main sponsors was the AMS, which saw TEX as an archival format; backward compatibility was thus essential. The TUGs wanted to protect their constituency and a change attracting some users had to be balanced against the risk of alienating some established users. Development thus became ever more efficient – with the use of better coordination tools, the building of central code repositories and the writing of procedures to submit new codes – but yet ever more specialized and remote from the preoccupation of the new, potential users – who thus had to organize themselves in different ways, separately from the core development team. The inertia from having to keep on serving established users thus translated into tensions between TEX developers and a tendency for new developers to start their own projects, leading to a progressive fragmentation in the development effort. Fragmentation (‘forking’) was due to initiatives by users whose needs were not accommodated by the (LA)TEX development teams. While this forking weakened the position of TEX as a standard and diverted developers resources from the main strand of development, it also adapted TEX to the needs of a wider variety of users, thus contributing to the popularity of TEX.

In the same way, on the proprietary side, there was a switch from entrepreneurs borrowing from TEX to them developing independent features. While this made TEX increasingly isolated, it also triggered a reaction by TEX developers, encouraging them to address new needs.

3.3.2 Network dynamics

The effectiveness of the OS organization was largely dependent on the anticipated benefit of getting work coordinated, and this was linked in the minds of developers to whether better coordination would make (LA)TEX more competitive vs. its proprietary alternatives. Developers did want (LA)TEX to keep a high market share in some domains – mathematical typesetting foremost among those. Developers were thus quite aware that if TEX was to be of use to themselves, it had to keep on being developed, that this continued development depended on cultivating its user base (source of future developers), and that therefore, they had to consider interests other than their own in the choice of what features to develop and whether to participate in collaborative work instead of each working on their own.

The consideration of such network effects was important to TEX developers for three reasons: 1) they arose naturally from the way TEX was developed (the more developers fixing bugs the better), 2) TEX developers and maintainers derived prestige, backing and influence from how many developers they attracted to their project and how many people used their packages, and 3) standards used in the industry varied widely, and there were substitutes to TEX for a wide range of its capabilities, so that gains in market shares for proprietary versions or alternatives to TEX were losses for the OS TEX software and typesetting standard. Network effects thus encouraged TEX developers to take into account competitors; development couldn't be exclusively oriented toward fulfilling their own needs.

The OS organization's dynamism was thus highest when LATEX was under the highest threat of losing market share to its alternatives (1990s). This was when OS developers worked on interfaces, distributions and a rationalization of development so as to keep their user base. It can therefore be argued that competition from proprietary software acted as a stir for the development of features the existing developers would not otherwise have found profitable to develop.

4 Conclusion

This paper traced the history of the development and use of the OS TEX typesetting program. This software was developed over three decades and came in competition with a variety of open-source and proprietary alternatives. TEX was an early and very successful open-source project that imposed its standards in a particularly competitive environment and inspired many advances in the typesetting industry.

A pattern of semi-altruistic open-source development emerged, whereby developers considered non-developers' needs in order to make TEX more attractive to a broader audience and more competitive vs. proprietary alternatives. Self-use motivations for development were thus complemented by direct and indirect network effect considerations. The TEX Users' Group, staffed by volunteers, used members contributions to finance and encourage development of a more complete and accessible distribution as well as a more user-friendly interface. Since this had no clear direct benefit to developers, who anyway were able to make use of TEX without such improvements, this type of behavior would not have been observed if OS developers had been motivated only by the fulfillment of their idiosyncratic needs. It could be that TeX developers were uniquely altruistic and thus motivated to work for free for the benefit of others. This case study offers an alternative explanation for a pattern of semi-altruistic development whereby developers consider network effects in their development decisions: they develop more features than they would need for themselves in order to attract and retain a mass of users for their software. A strong user base increases the value of the software through direct and indirect network effects. Direct, demand side network effects were particularly strong for TeX, a standard format for document exchange, while indirect, production side network effects are always strong for any OSS as users become developers and contribute themselves to the improvements in the software.

While network effects motivated developers to consider users' needs, TeX developers kept preferring development that benefited their own constituency – be it defined as the group of early developers, as early users of TeX, as academics or more broadly, as specialized mathematical typesetters. This situation gave the opportunity for commercial software developers to offer typesetting software for other, more common types of users. Over time, as commercial software came to offer increasingly high typesetting standards, TeX lost a part of its user base. Compatibility with its standard thus became less valuable and those OS developers who previously had to abide by the TeX standards for fear of losing compatibility of their systems with the main strand of TeX development were no longer so reluctant to launch their own versions of TeX, extending its capabilities to, for example, multilingual typesetting or educational publishing. The consideration of network effects thus played an important role in the dynamics of TeX's development. The stronger they were, the more coherent TeX development was.

This case study helps assess the long-term welfare effect of the emergence and growing importance of open-source development methods. One has to consider the dynamics of the interaction between OS and proprietary development. The TeX case study shows that entrepreneurs often complemented the OS effort in areas in which it was less adapted and borrowed code and ideas from OS systems.

The TEX experience may inform other OS projects and similar community wide efforts in the development of a public good. One may attach special attention to the originality of the terms of the LATEX Project Public License (LPPL). It is permissive in the sense that proprietary exploitation of code licensed under the LPPL is allowed and restrictive in the sense that modifications must be clearly labeled as such and must be distributed with or refer to the original version. These restrictions ensure that TEX systems remain a stable whole that produces the same output for all users. These license terms are probably optimal for software which like TEX is meant for a wide public that does not have a programming background: they ensure the preservation of TEX as a standard and this in turn encourages its take-up by commercial companies that will make it available to a wider public. TEX's rather unique success in a domain, end-users applications, that is usually reserved to proprietary software, gives it a status as a role model for Open-Source projects that would want to attain success beyond the rarefied world of current users of Open-Source Software.

References

LATEX3 PROJECT TEAM (1997): "Modifying LATEX," *TUGboat*, 18(3), 127–130.

ADAM, A. E. (2004): "Hacking into Hacking: Gender and the Hacker Phenomenon," *ACM SIGCAS Computers and Society*, 32(7).

BEEBE, N. H. (2004): "25 years of TEX and METAFONT: Looking back and looking forward," *TUGboat*, 25(1), 7–30.

BITZER, J. (2004): "Commercial versus open source software: the role of product heterogeneity in competition," *Economic Systems*, 28, 369–381.

BITZER, J., AND P. SCHRÖDER (2005): "The impact of entry and competition by open source software on innovation activity," Working Paper 2005-12, Aarhus School of Business.

BONACCORSI, A., AND C. ROSSI (2004): "Altruistic individuals, selfish firms? The structure of motivation in open source software," *First Monday*, 9(1).

CASADESUS-MASANELL, R., AND P. GHEMAWAT (2003): "Dynamic Mixed Duopoly: A Model Motivated by Linux vs. Windows," Strategy Unit Working Paper 04-012, Graduate School of Business Administration, Harvard University.

COTTRELL, A. (1999): "Word Processors: Stupid and Inefficient," http://www.ecn.wfu.edu/ cottrell/wp.html.

FLYNN, P. (2001): "TEX — a mass market product? Or just an image in need of a makeover?," *TUGboat*, 22(3), 137–139.

FRANKE, N., AND E. VON HIPPEL (2003): "Satisfying heterogeneous user needs via innovation toolkits: the case of Apache security software," *Research Policy, Special Issue on Open Source Software*, 32(7), 1199–1215.

GAUDEUL, A. (2003a): "The (LA)TEX project: A case study of open-source software," *TUGBoat*, 24, 66–79.

——— (2003b): "The (LA)TEX project: A case study of open-source software," Ph.D. thesis, chapter 3.

——— (2005a): "Differentiation and network effects in open source production," Working Paper, University of East Anglia, Norwich.

——— (2005b): "Public provision of a private good: What is the point of the BSD license?," Working Paper, University of East Anglia, Norwich.

HANN, I.-H., J. ROBERTS, S. SLAUGHTER, AND R. FIELDING (2004): "An empirical analysis of economic returns to open source participation," Working Paper, Carnegie Mellon University.

KNUTH, D. E. (1991): "The future of TEX and Metafont," *TUGboat*, 11(489).

KOENIG, J. (2004): “Seven open source business strategies for competitive advantage,” IT Manager’s Journal, http://management.itmanagersjournal.com/.

KUAN, J. (2001): “Open source software as consumer integration into production,” Working Paper, Haas School of Business, University of California-Berkeley.

——— (2002): “Open Source Software as Lead Users Make or Buy Decision: A study of Open and Closed Source Quality,” 2002 OSS Conference, Toulouse.

LERNER, J., AND J. TIROLE (2002): “Some Simple Economics of Open Source,” *Journal of Industrial Economics*, 50(2), 197–234.

LERNER, J., AND J. TIROLE (2005): “The economics of technology sharing: open source and beyond,” *Journal of Economic Perspectives*, 19(2), 99–120.

LIN, Y. (2005): “Inclusion, diversity and gender equality: Gender Dimensions of the Free/Libre Open Source Software Development,” Working Paper.

MITTELBACH, F., AND C. ROWLEY (1997): “The LaTeX3 project,” *TUGboat*, 18(3), 195–198.

MOCKUS, A., R. T. FIELDING, AND J. D. HERBSLEB (2002): “Two case studies of open source software development: Apache and Mozilla,” *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 11(3), 309–346, Working Paper.

MUSTONEN, M. (2003): “Copyleft - the economics of Linux and other open source software,” *Information Economics and Policy*, 15, 99–121.

——— (2005): “When Does a Firm Support Substitute Open Source Programming?,” *Journal of Economics & Management Strategy*, 14(1), 121–139.

NICHOLS, D., AND M. TWIDALE (2003): “The usability of open source software,” *First Monday*, 8(1).

NTG TeX FUTURE WORKING GROUP (1998): "TeX in 2003," *TUGboat*, 19(3), 323–329.

RAYMOND, E. (2001): *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary.* O'Reilly and Associates.

SCHMIDT, K., AND M. SCHNITZER (2003): "Public Subsidies for Open-Source? Some Economic Policy Issues of the Software Market," *Harvard Journal of Law and Technology*, 26, 473–505.

VON KROGH, G., S. SPAETH, AND K. R. LAKHANI (2003): "Community, joining, and specialization in open source software innovation: a case study," *Research Policy*, 32, 1217–1241.

WICHMANN, T. (2002): "Firms' open source activities: Motivations and policy implications," Free/Libre and Open Source Software Final Report, Part II, International Institute of Infonomics.

A Acronyms

AMS American Mathematical Society

BSD Berkeley Software Distribution

CTAN Comprehensive TeX Archive Network

pdf Portable Document Format

GPL General Public License

LPPL LaTeX Project Public License

LUG Local Users' Group

NFSS New Font Selection Scheme

NTS New Typesetting System

OS Open-Source

OSS Open-Source Software

TDS TEX Directory Structure

TUG TEX Users' Group

WYSIWYG What You See Is What You Get

B Typology of competition

B.1 Competition with independently developed proprietary systems

There existed proprietary typesetting systems prior to TEX: IBM's Script (1970), The Bell group's ROFF (1971) for Unix systems or Reid's Scribe (1978). However, either this software was not affordable or not accessible to the type of users TEX was developed for (scientific community), or its standards were lower than those of TEX. Some features in TEX were inspired by ROFF and Script and LATEX commands were often directly inspired by Scribe commands. As TEX gained in popularity, this pre-existing software became less popular although a GNU/Linux version of ROFF, GROFF, is still being used. In the mid 80s, WordPerfect (1982) and Microsoft's Word (1984) took up TEX's less sophisticated users thanks to their WYSIWYG interfaces, while commercial typesetters adopted Framemaker (1986), Corel's Ventura (1986) or QuarkXpress (1987) and other more specialized proprietary systems such as 3B2 (1986) for mathematical typesetting. Those systems were relatively easy to learn and established themselves as standards in the industry. QuarkXpress became the equivalent of the QWERTY keyboard in the industry: not necessarily the system that was the most efficient or the most powerful but any typesetter knew how to use it. Adobe bought Pagemaker in 1994, Framemaker in 1995 and gradually imposed its standards (postscript, then pdf) to the industry. It launched its own desktop publishing application, Indesign, in 2000.

B.2 Competition with proprietary systems inspired by TEX

Another type of competition for OS TEX systems came from proprietary typesetting systems based on TEX. PCTEX (1985), MicroTEX (1985), TEXtures (1989), Y&Y (1990), Scientific Word (1992) were implementations of TEX that offered better and more fonts (Y&Y), more sophisticated interfaces (WYSIWYG interface of Scientific Word) or were adapted to platforms that were neglected by the OS community (TEXtures for Macintosh, PCTEX for personal computers). All these proprietary implementations were easier to install than the OS versions of TEX of the time and provided better user documentation and support.

However, while proprietary versions of TEX were often the first to integrate some user-oriented features, the OS developers frequently caught up with them over time. It took about 10 years for the first complete OS TEX distributions for the PC to be as user friendly as the first proprietary ones (MikTEX (1996), TEX Live (1996)), and the first OS and user friendly interfaces such as LYX (1995), TEXshop (2000), TEXMacs (2002) or XEMTEX (2003) appeared about 7 years after the first such proprietary interfaces. Graph 2 p. 32 relates the chronology of the development of proprietary and then OS distributions and interfaces for TEX.

B.3 Competition with open source alternatives to TEX

OS competition for TEX's user base came from software that was developed in a proprietary way and was then released in open source versions, such as the GNU/Linux version of the Bell Lab's ROFF, GROFF (1991), Corel's OS version of WordPerfect for Linux (1998) or Sun's StarOffice which became the OpenOffice

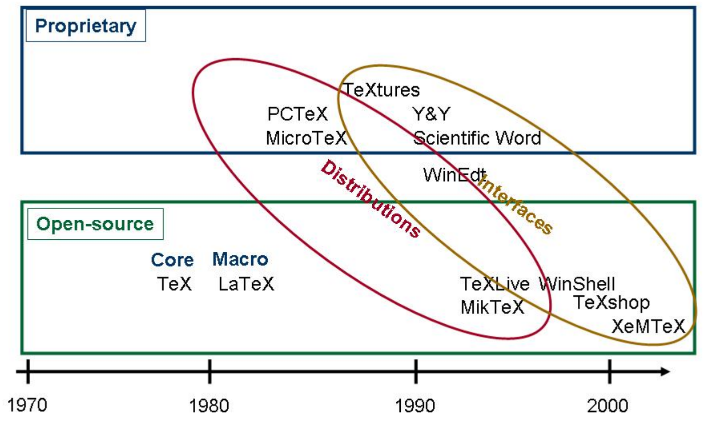

Figure 2: Proprietary versions of TEX: a game of catch-up.

OS project (2002). Very few word processing software developed in an OS way from scratch could establish themselves (Abiword (2000) is an exception). Another type of competition came from alternative standards for typesetting documents (SGML (1986), XML (1998)), notably mathematical documents (MathML (1999)). These new languages allowed more complex manipulation of mathematical formulas and/or more precise control over the layout and design of the pages. TEX developers had to develop converters to translate TEX documents into these more widely used and more sophisticated languages (XMLTEX (2000), JadeTEX (2001)).

C TEX sub-projects

See table 1. The list of projects is not comprehensive. A more complete list of contributors is given below. Categories include core, macros, distribution and interface development. Additional categories are systems development, which are improvements in the accessibility and structure of TEX systems, and extensions, which are programs extending the capabilities of TEX without changing its core.

ϵ-TEX: Peter Breitenlohner and Phil Taylor
CTAN team: Rainer Schöpf, Joachim Schrod, Sebastian Rahtz, George Greenwade, Robin Fairbairns, Jim Hefferon.
LATEX3 team: Johannes Braams, David Carlisle, Robin Fairbairns, Frank Mittelbach, Chris Rowley, Rainer Schöpf, Thomas Lotze, Morten Høgholm, and Javier Bezos. Previous members: Denys Duchier, Alan Jeffrey, Michael Downes
TEX Live/Collection coordinators: Sebastian Rahtz, Karl Berry, Staszek Wawrykiewicz, Manfred Lotz
ConTEXt: Hans Hagen, Taco Hoekwater, Patrick Gundlach, Adam Lindsay and others.
pdfTEX: Hàn Thế Thành, Martin Schröeder, Hartmut Henkel, Taco Hoekwater, Hans Hagen, Heiko Oberdiek.

D Chronology

See table 2 and 3. Development is divided in 3 areas: core, macro, others (organization, distributions and interfaces). Competitors are divided in 4 types: independently developed software, OS or proprietary, and software inspired by TEX, open source or proprietary. Freeware and shareware are classified as proprietary.

Name	**Object**	**Category**	**Lead Developer**	**Inception**
TEX	Mathematical typesetting	Core	Donald Knuth	1978
plainTEX	Basic macros	Macros	Donald Knuth	1978
AMS-TEX	Macros for AMS journals	Macros	Michael Spivak	1981
LATEX	General purpose TeX macros	Macros	Leslie Lamport	1983
BiBTEX	Bibliographies	Extension	Oren Patashnik	1984
MakeIndex	Indexes	Extension	Pehong Chen	1987
NFSS	Easy access to a complete selection of fonts	System	Frank Mittelbach and Rainer Schöpf	1989
CTAN	Online repository of TEX packages	System	CTAN team	1990
PSTricks	Figures and graphs	Extension	Timothy Van Zandt	1993
4AllTEX	Plug and Play TEX distribution for MSDOS PCs	Distribution	Netherland LUG	1993
WinEdt	Shareware Interface	Interface	Aleksander Simonic	1993
Ω	Multilingual typesetting	Core	Yannis Haralambous and John Plaice	1994
LATEX 2$_\varepsilon$	Extension of LATEX	Macros	LATEX3 team	1994
ConTEXt	Macros for educational publishing	Macros	Hans Hagen	1994
TDS	Directory structure	System	TUG Working Group	1994
teTEX	TeX distribution for Unix	Distribution	Thomas Esser	1994
pdfTEX	pdf output	Extension	Hàn Thế Thành	1996
TEXLive	Official distribution	Distribution	TUG and LUGs	1996
MikTEX	Distribution for Windows systems	Distribution	Christian Schenk	1996
ϵ-TEX	Updating TEX	Core	Peter Breitenlohner	1997
NTS	Rewriting TEX	Core	Karel Skoupý	1998
fpTEX	Windows version of teTEX	Distribution	Fabrice Popineau	1998
Winshell	Freeware interface	Interface	Ingo de Boer	1998
LYX	WYSIWY Mean interface	Interface	Matthias Ettrich	1999
TEXshop	Quasi-WYSIWYG interface for the Mac	Interface	Richard Koch	2000
preview-LATEX	Instant previewing of documents	Interface	David Kastrup	2001
XEMTEX	TEX system for the education sector and public administrations	Distribution	Marie-Louise Chaix and Fabrice Popineau	2003.
XeTEX	Mac based TEX	Core	Jonathan Kew	2004

Table 1: TEX sub-projects by year and category

	DEVELOPMENT OF TEX												
	1970-1977	**1978**	**1979**	**1980**	**1981**	**1982**	**1983**	**1984**	**1985**	**1986**	**1987**	**1988**	**1989**
Core		TEX 1.0 (Sail)	DVI format			TEX 2.0 (Pascal)					TEX2C		TEX3.0
Macro		PlainTEX			AMS-TEX		LATEX	BibTEX			makeindex		LATEX3 team
Organization (O), distributions (D) and Interface (I)				TUG									NFSS (O)
	EVOLUTION IN THE SOFTWARE ENVIRONMENT												
	1970-1977	**1978**	**1979**	**1980**	**1981**	**1982**	**1983**	**1984**	**1985**	**1986**	**1987**	**1988**	**1989**
Independent proprietary	IBM Script (70); Bell Labs Roff (71)					Corel Wordperfect		Microsoft Word	Pagemaker; Adobe Postscript	Framemaker; Ventura; 3B2	QuarkXPress	MS Word for Windows	
Independent OS										SGML			
Inspired proprietary									PCTEX, MicroTEX				TEXtures
Inspired OS													

Table 2: **Chronology of the development of TEX from 1970 to 1989.**

	DEVELOPMENT OF TEX													
	1990	**1991**	**1992**	**1993**	**1994**	**1995**	**1996**	**1997**	**1998**	**1999**	**2000**	**2001**	**2002**	**2003**
Core	Web2C				Ω		pdfTeX	ϵ-TeX	NTS					XeTeX
Macro			PSTricks (drawings)		LaTeX 2$_\varepsilon$; ConTeXt						XMLTeX	JadeTeX (SGML)		
Organization (O); distributions (D) and Interface (I)	CTAN (O)				TDS (O); teTeX (D)		MikTeX (D); TeXLive (D)		fpTeX (D)	LPPL (O)	TeXshop (MacOS) (I)	preview-LaTeX (Unix) (I)	Debian guidelines (O)	XEMTEX (Windows) (I)
	EVOLUTION IN THE SOFTWARE ENVIRONMENT													
	1990	**1991**	**1992**	**1993**	**1994**	**1995**	**1996**	**1997**	**1998**	**1999**	**2000**	**2001**	**2002**	**2003**
Independent proprietary				pdf by Adobe							InDesign by Adobe			
Independent OS		Groff (GNU roff)	HTML						XML; Wordperfect for Linux	MathML	XHTML; Abiword		OpenOffice	
Inspired proprietary		Y&Y	Scientific Word	WinEdt					WinShell					
Inspired OS						Lyrix				LyX for Unix			TeXMacs	

Table 3: **Chronology of the development of TeX from 1990 to 2003**.