Jonathan Kew

Jonathan Kew is best known in the TEX community as the developer of XƎTEX.

[Interview completed 3 April 2007.]

Dave Walden, interviewer: Please tell me a bit about your personal history independent of TEX.

Jonathan Kew, interviewee: I had a fairly conventional upbringing, I suppose, living in the south of England (except for a year spent with relatives in Sweden, which I still remember as one of the best parts of my childhood). I was the middle of three boys, though whether that explains anything about how I turned out is debatable at best. Growing up, I was always the small, quiet kid, more likely to be found in my room with a book than out with a crowd. (Those who know me will realize that this hasn't changed too much over the years!) My interests tended towards the academic and scientific, with a touch of the practical thrown in; at various times I was fascinated by subjects like early Antarctic exploration, butterflies and moths, aviation, electronics and ham radio, mountain travel, and more.

A key event came when my typical teenage wanderings and wonderings led eventually to a commitment to the Christian faith. My family had always been clearly non-religious, so this was a significant decision, and not one I took lightly, knowing that it represented a break with my background. I think my father felt at the time that it was just another phase I would pass through, but as the years have turned into decades it has remained a defining part of my life, and provides a framework and direction into which all other activities fit.

Most of my adult life — the only career I have ever had, really... has been spent working with SIL International, first in South Asia and more recently based in England. SIL is a Christian non-profit organization serving minority communities around the world with linguistic research, education, development, translation, and more, and has provided me with the ideal opportunity to combine my computing work with an interest in languages and writing systems. After starting out as a language surveyor, "roughing it" around Asia researching local languages, I soon found my real niche working on text processing and publishing systems for non-Western languages.

DW: Please sketch what a "language surveyor" does.

JK: As a language surveyor, I was involved in studying the languages and dialects of an area, and the patterns of language use among the communities there. For example, we collected word lists in different localities, to measure the percentage of shared vocabulary between communities; and tested comprehension of stories that we had recorded in other locations, to see how well people understood nearby dialects. Another aspect was studying bilingualism and multilingualism, to learn how national and regional languages were used in addition to the local mother tongue.

In the language survey I did in South Asia, I was always working together with local colleagues who were familiar with the area and fluent in the regional language. Each project involved both SIL personnel and local partners; we could not have carried out the work alone. While I learned enough of the regional language to get along in daily living situations, I was certainly not fluent enough to have worked effectively on my own.

DW: I know some churches (e.g., the Christian Science church whose publishing building is a block from my apartment in Boston) publish the Bible, daily devotional readings, the founder's books, etc., in many languages. But SIL is not a church, correct? Is the point that SIL's Christian mission is to help people with languages just as someone else's Christian mission is to help people in some other, not necessarily religious, area of need?

JK: It's tempting to just copy and paste some paragraphs from the SIL web site (`http://www.sil.org`), but I suppose that would be cheating!

Anyhow, you're right; SIL is not a church, though its members share a Christian motivation. We seek to serve minority language communities — regardless of faith or other factors — in a variety of language-related ways, as this is where our expertise lies. Linguistic and anthropological research contributes both to scientific knowledge in general and to the understanding and development of the particular communities where we work. It is a necessary basis for basic literacy and mother-tongue education, as well as supporting the transition into major-language education. This can help people interact on a more equal footing with their neighbors and with government and other agencies. SIL personnel, in partnership with local people, churches, and other agencies (both religious and secular) may be involved in almost any kind of development or educational work where linguistic expertise is important, from Bible translation to support an indigenous church to translation of basic health booklets or primary schoolbooks, or management of a community-wide literacy programme.

DW: Scouting around the SIL web site you mentioned, I find the following statement: "SIL members are volunteers who raise their own financial support." Should I assume this is also true in your case, and that you have to raise your own financial support and don't have another job?

JK: All my time serving with SIL has been on this basis, funded by contributions from several churches and a variety of individual friends and sponsors who are interested in supporting our work. I've worked full-time with SIL for over 20 years now, without any other job or means of support. And naturally the level of income received has varied, sometimes greatly, over the years, but there has at least been a roof overhead and food on the table throughout! I suppose I decided fairly early that it was more important to me to use the abilities I have in some way that could make a worthwhile contribution in the world, a significant difference in people's lives, than to concentrate on building a bank balance (or lavish lifestyle). And I think God honors that decision, through the faithful support of many individuals and churches. It's true that it takes a certain amount of time and energy — sometimes a substantial proportion — to find and maintain support in this way, which can be a frustration and distraction when there are other more "productive" things I'd like to be doing, but on the other hand I have been privileged to get to know some wonderful friends through the process.

DW: How did you first get involved with TeX?

JK: I first learned of TeX while at university in Cambridge. We didn't have TeX on the university mainframe, but I happened to find *The TeXbook* while browsing in the bookstore — this must have been in 1984, when it was quite new — and was fascinated by the system it described. It seemed so far ahead of the primitive text formatting software we had at the time. Mind you, even if we'd had TeX, it would have been difficult to get useful output on the line printers that were the main output devices of the day! There was a special Versatec printer that could do something like 200 dpi on special paper, I think, though as a lowly undergrad (not even in the computing department), I'm not sure if I would have had access to it.

Anyway, browsing *The TeXbook* in the store was the inspiration for the first "typesetting" software I wrote — a little program that took TeX-like input for equations and simple paragraphs of text, formatted them, and "drew" the resulting page on a drum plotter. An engineer friend and I used this successfully to produce some equation-rich pages to insert into his final-year thesis; I was quite proud of the results we achieved, with proper integrals, fractions, super- and subscripts, etc., considering that the normal output device for final copy was a daisy-wheel printer! This was not really TeX, of course, but it wouldn't have happened without my bookstore encounter with *The TeXbook*.

I think the first time I used TeX itself was in 1988, having managed to shoehorn Pat Monardo's Common TeX implementation into an old DOS machine, along with a LaserJet driver from Nelson Beebe's collection. By this time, I had spent a couple of years in South Asia working on text processing and printing solutions for several interesting scripts, and the potential of TeX and Metafont was clearly of interest. I'd been doing some Devanagari typesetting on home-built CP/M machines with 24-pin printers (we printed the body text in three passes, giving a 72-high glyph matrix, and then photographically reduced the output). So TeX and a 300 dpi laser printer was a great step forward, though the equipment I had was just barely adequate to run it. I spent many hours trying to coax Metafont to draw Urdu characters, and even printed at least one small book with them, though I must confess that it wasn't long before these were abandoned in favor of PostScript fonts.

DW: As a relative newcomer to TeX, I have not heard of "Pat Monardo's Common TeX implementation". Please tell me what this was or is and how it fits into the history of TeX engines.

JK: Common TeX was a version of the TeX program rewritten in C, primarily targeted at Unix systems, I think. As I understand the history, it was a hand translation of the Pascal/WEB program, done before the current Web2C system had been developed. Common TeX made it possible to get TeX running on machines that had a decent C compiler, but no reasonable Pascal system available.

I'm not sure whether Common TeX was 100% TRIP-test compatible, but for all practical purposes it was a standard TeX engine, it just avoided the need for a Pascal compiler and runtime system. I think as the whole Web2C system matured, the need for Common TeX eventually disappeared, as compilation via C became the most common implementation for all the TeX/MF family of programs.

DW: What sort of systems do you mean when you say you have worked "text processing and publishing systems for non-Western languages" for SIL? I see two papers by you listed on the Ethnologue.com (`http://www.ethnologue.com`) web site: "A computerized assistant for translators", 1992, and "Formatting interlinear text", 1990, with Stephen R. McConnel. For how long has TEX been part of your work in this area or how did it begin to become a part of this work?

JK: Yes, I haven't generally written much for publication! (In more recent years, I've had papers published in the proceedings of a few Unicode and TUG conferences, but those are probably hard to find cited.) But I've worked on several projects over the years, including a text editing and data management environment for scripture translation; software for linguistic analysis and dialect adaptation; and of course text formatting/typesetting. In all cases, ensuring that the tools were usable with complex writing systems has been a key focus for me, partly because of my early experience in South Asia which brought me into contact with several interesting (Indic and Arabic-based) scripts and made me aware of the challenges involved.

My first use of TEX in this field must have been around 1988–89, I think, when I worked on a package for handling interlinearized texts (such as linguists love to work with). While studying and documenting languages, we often annotate the words of a text with multiple levels of information; for example, the division of the word into morphemes, various types of grammatical information, glosses in one or more other languages, and so on. The result is a stack of information, aligned below the original word; this can of course be handled very naturally in TEX as a `\vbox`. And the beauty of TEX, for publishing lengthy texts of this nature, is that it can turn a list of such stacked boxes into paragraphs and pages in just the same way as it does with normal words. Try doing that with a typical word processor!

Then, around the end of 1989, I began to look seriously at providing publishing solutions for minority languages using the Arabic script, and started some font work in Metafont. With the enhanced ligature capabilities and 8-bit support in TEX 3.0, I was able to get some useful results, though the contextual behavior of the script remained a challenge. Within a year or two, frustrated by the limitations of TEX's font and glyph handling, and looking for easier integration with standard text editing environments, I had begun to experiment with extended TEX engines that interfaced with external text rendering systems to handle the complexities of non-Latin scripts. Today's XƎTEX, my main TEX project in recent years, has roots in that work back in the 1990s.

DW: Of course, other readers and I could look at your XƎTEX web site (`http://scripts.sil.org/xetex`), but perhaps you can also tell us something about it here. For starters, how should I pronounce XƎTEX — like Zee TEX? And why did you call it that?

JK: I don't have a strong opinion on the "correct" pronunciation, though personally I say *zee-TEX*. There was some discussion of this on the mailing list recently, and it was clear that the "natural" pronunciation depends on people's native language — which is fine with me.

The name was chosen to imply an eXtended version of ε-TEX, along with an association with Mac OS X (which was initially the only target platform). As one of the intended uses was for typesetting right-to-left scripts, a palindromic name seemed like fun; and the properly-typeset version is supposed to use the Unicode character U+018A LATIN CAPITAL LETTER REVERSED E for the first lowered "E", hinting at support of much more than the basic Western character set.

DW: Please sketch the technical approach you took to modifying or augmenting TEX to enable it to use fonts available on the operating system, as I saw you demonstrate at the

PracTEX 2006 conference in New Jersey.

JK: The first point is that the `\font` command is augmented so that it asks the host operating system to find fonts (by their real names, not cryptic filenames) from whatever collection of fonts the user has installed. So all the same fonts should be usable for typesetting in XƎTEX as in any standard GUI application, and known by the same names. There is no need for a TEX-specific font preparation or installation procedure any more.

When XƎTEX finds a font in this way, what it finds is the actual font itself (whether PostScript, TrueType, or OpenType), not a TFM file; and therefore the paragraphing routine needs to measure text by referring to the real font, not by using the TEX-style character metrics. In addition, it needs to take account of the complexities of mapping characters to glyphs, particularly in cursive and non-Latin scripts. So rather than building paragraphs from lists of characters, each of which has fixed metrics, XƎTEX builds paragraphs from "words", each of which is a whole run of consecutive characters in a given font. It can then call on a "layout engine" such as ATSUI (on Mac OS X only), ICU Layout (on any platform), or SIL's Graphite to carry out linguistically and typographically required transformations and effects, resulting in an array of glyphs and their positions that represent the word as laid out using the current font. The paragraph then consists of a list of such words (or chunks of text), interleaved with glue, penalties, etc.

There are further complications, of course; for example, hyphenation may require such "word nodes" to be taken apart and reassembled after finding possible break positions. But the fundamental idea is to collect runs of characters and hand them as complete units to a font rendering library that understands how to handle layout at the level of the individual glyphs.

DW: I assume you did this work on a Mac first because it was easier there. What made that so, and how have you now expanded things so XƎTEX also works on Linux and Windows systems?

JK: It was easier for me, at least, partly because I was most familiar with the Mac OS X platform, particularly in the area of text, fonts, and internationalization. This familiarity, and even parts of the XƎTEX code, dates back to work on earlier versions of Mac OS back in the mid-1990s; the first such "extended TEX" I implemented was based on Apple's WorldScript technology, which offered support for a number of non-Latin scripts, though with quite limited typographic features. This was followed by TEXGX, a reimplementation that used Apple's QuickDraw GX graphics system. This supported higher precision in text measurement and layout, and a more powerful implementation of complex script behaviors. Although TEXGX was never very widely used (to my knowledge), it did become known to at least a few people in the TEX community, and served me as a key tool for a number of years.

Sadly, QuickDraw GX never really succeeded in the marketplace, and Apple dropped it from later versions of the operating system. However, the advanced typographic features were reborn in ATSUI, the Unicode text layout library that is part of Mac OS X (actually, it was available on pre-OS X systems, though not widely adopted). With Apple's move to a Unix-based operating system, in conjunction with improved support for Unicode and international typography, it was natural to begin work on a successor to TEXGX that would work in this new environment; and so XƎTEX was born.

When I began these experiments in integrating TEX as a page formatting system with existing libraries for font handling, international text, and glyph-level typography, the facilities available in the Mac operating system were significantly ahead of those on Windows — and Linux wasn't even in the picture as a desktop environment. I was aiming to leverage the available tools to provide maximum functionality for a minimum amount of effort, and so I used Apple's text and graphics technologies wherever possible.

Over the years, the picture has of course changed, and there are excellent libraries for international text layout on all the major platforms. And so once XeTeX was functional and stable, I began to look at how it could be made cross-platform by replacing the Apple-proprietary components used with open-source alternatives. This meant supporting OpenType fonts through the ICU layout library, and replacing the original PDF output driver (based heavily on the Mac OS X graphics system) with an extended version of the existing dvipdfmx, among other changes. But as far as possible, I have tried to maintain the simplicity of use that I think made XeTeX attractive to its early adopters when the first Mac-only version appeared.

DW: Do you believe other systems such as Dick Koch's TeXShop have hastened the popularity of XeTeX, or do you think its intrinsic benefits would have led to its great popularity in any case?

JK: I think TeXShop has done a huge amount to make TeX (in general) on Mac OS X more accessible and attractive, and thus given a larger potential audience for any new development in this area. And perhaps because TeXShop made TeX accessible to less technically-inclined users, it also provided more fertile ground for XeTeX, which allows such users to experiment with fonts that they would otherwise not even attempt to use.

DW: I suppose the reverse may also be true: XeTeX's popularity has helped increase the number of TeXShop users.

JK: Without such front-ends (on any platform), the TeX user community would be significantly more limited, and would consist mainly of people willing and able to deal with arcane command lines, configuration files, and so forth. This might have meant that XeTeX's approach to font access would have seemed less important to many. I suspect that its other major features — Unicode and complex-script support — would still have been highly attractive to some users, but perhaps to a much more limited, specialized community.

DW: Do you develop XeTeX essentially alone, or do you have some sort of "open source" team helping you?

JK: It's been essentially a one-person (and part-time) project over the years, driven primarily by my own need for a powerful and flexible typesetting system with Unicode and non-Latin script support for the publishing jobs I'm asked to do. However, once the source repository was made public (a couple of years ago — I forget exactly), I have received valuable patches from several contributors, particularly in the area of CJK support. Suggestions for ongoing development are always welcome, and those accompanied by code are even more so!

In particular, I should mention Will Robertson's outstanding work on LaTeX integration (primarily the fontspec package [discussed in his interview]), Ross Moore's work on the LaTeX graphics and color drivers, Jin-Hwan Cho's help in extending the dvipdfmx driver, and Miyata Shigeru's contributions to improve vertical text and CJK support in both XeTeX itself and the driver, and to provide support for PSTricks graphics. And with the integration of XeTeX into the TeX Live sources, many people have helped find and solve portability and build issues on the wide variety of supported platforms.

DW: I don't know if you have read what Taco Hoekwater said in his interview about the differences between what he and others are working on and what you are doing with

XeTeX. Do you have anything to add about connections between the two projects?

JK: Not really. The pdfTeX/LuaTeX team is taking quite a different approach to some issues, and so it's unclear whether there will come a time when merging the projects makes sense. But I am of course happy to share ideas (and code), and hope that wherever possible we can provide features in ways that make it easy for macro writers and users to work with either system. If LuaTeX proves successful and popular, and develops to the point where it offers users all the same capabilities as XeTeX (even if the underlying implementation is quite different), I'll be delighted, and may no longer feel a need to continue working on XeTeX. But for the time being, at least, I think the two projects each need the freedom and flexibility to explore their own ideas, and users are of course free to work with whichever serves their needs best.

DW: Has XeTeX been extensively used to publish documents in SIL's domain? Or has the TeX community been more of an "early adopter" of XeTeX?

JK: SIL has never had a large TeX-using community; the vast majority of publishing work is done using commercial word processing and DTP systems. So my TeX projects, including XeTeX, have only had a few active users within SIL. We have found them invaluable for a number of challenging publications, but I wouldn't call it anything like "extensive" use. We're seeing an increase in interest these days, especially with the complex-script support XeTeX offers, but I've never expected it to displace products like Word, OpenOffice, InDesign, etc. on a majority of desktops.

When I released the first public version of XeTeX (for Mac OS X only), I was somewhat surprised at the level of interest and even excitement that it seemed to generate. I was very much aware that XeTeX's extensions mean that in certain cases, it is not 100% compatible with all existing TeX/LaTeX/ConTeXt/etc. documents or macro packages, and for this reason I expected that it would be something of a specialist niche product, used by a few people who understood its approach and needed the Unicode and non-Latin features it provided. So it has been exciting to see it adopted with such enthusiasm by a range of people in the existing TeX community, and I'm very glad to have been able to provide something that many users seem to appreciate and enjoy. I hope my work can facilitate the production of beautiful books of all kinds, for the benefit of many communities worldwide.

DW: Thank you for taking the time to be interviewed. Your life story and its involvement with languages, fonts, and TeX is fascinating to me; you have my great admiration.