Why I keep using \TeX
TUG 2009
After many years of doing certain things in similar ways, it makes sense to look back and reconsider certain choices. So, why am I still using TEX? I will reflect on this from three perspectives: fonts, graphics, and typesetting.

Among the reasons for using TEX are some that don't always relate that much to the above. For example:

- It’s fun to use TEX and MetaPost and play with document design and production.
- Some problems are challenging, and I’m always surprised by what users come up with.
- The community is kind of special, and I like meeting TEX friends at meetings.
- Of course, TEX equals Don Knuth, and that’s something in itself.
One starts out with the fonts that come with TEX: the Computer Modern Typefaces. In former times when you also needed support for math that more or less settled the issue. When you want to broaden your horizon, you quickly end up in a font mess (we're talking traditional TEX here): Fonts have encodings, and you have to make the right choice. Sometimes you're forced to use two sets for one language.

The number of slots in an encoding is limited to 256 and not all fonts have all characters that we want. Fonts relate to hyphenation and this adds an extra dimension to a proper font system. We have more encoding than needed because for some reason funny symbols that aren't part of the regular input happen to be part of them. Math has its own demands and adds another layer of complexity.

As font support is a prerequisite it is no surprise that only a few macro packages showed up. Designers like to mix font sets in one document. The systematic approach of TEX-based font systems often does not match designs.
In doesn't matter how many handy macro and tools you make, fonts are always somewhat complex. And then we don't even start mentioning support for non latin scripts. The last few years things got more complicated because wide fonts (TrueType and OpenType) started showing up. At the same time Unicode became the fashion. If nothing had happened with regards to fonts, after a few years TEX would have become pretty unusable. Writing macros that support all well has become a nightmare: there is always a new exception. So fonts are a pretty good reason to quit using TEX.
First pdfTEX started providing support for at least using TrueType fonts but still the 256 limitations applies. The Latin Modern as well as the TEX Gyre project started upgrading the fonts that normally come with distributions and that are used by most users. Then X\textTeX showed up and suddenly we could go forward and be part of the font scenery again. However, driven by third party libraries the level of control is limited. A while later Lua\textTeX showed up and that engine provides full access to the font machinery and permits \textTeXies to go beyond regular font support. We can keep filling up the typesetting niches too. Without the Unicode and OpenType aware engines \textTeX would become rather unusable pretty soon, at least outside the ‘Writing scientific articles in English’ market. So, the perspective of modern font technologies (that of course bring their own problems) kept me going.
Traditional TEX can only do straight lines and areas (rectangles). It is only thanks to \special and \write that TEX has survived. These mechanisms permit backends to add graphic capabilities to TEX. Some basic specials showed up for drawing curves but these are not that much in fashion. Some powerful drawing packages have been written in TEX. These use PostScript or pdf and can give impressive results. In ConTEXt we need a rather tight integration between the typesetting machinery and the graphics engine. Personally I use MetaPost a lot. This is an easy language. Processing is fast and the output is quite simple and can be parsed easily.
As most modern designs demand some graphics using MetaPost had become a bottleneck in traditional engines. More complex subsystems (like nested backgrounds that span pages) are quite demanding in terms of resources and runtime. All can be done, but you don't want to know the gory details. In automated workflows it makes a big difference if processing a moderately complex document takes 10 minutes or 60 seconds. In order to survive that situation had to be improved.
Embedding MetaPost in LuaTEX removed a significant bottleneck. It also made support for text in MetaPost more convenient as all happens in the same program. It also permits me to rewrite some bits and pieces in cleaner code. This tight integration of MetaPost in LuaTEX will keep me around for a while.
There is no doubt that \TeX{} does a good job on typesetting and I hope that Con\TeX{}t is a demonstration of this.

In most cases there is a solution for a problem, but sometimes it demands some detailed \TeX{} knowledge.

Not all problems have clean solutions or have multiple solutions. Something that works out okay here fails there which is no fun in a general purpose macro package.

So, although \TeX{} is pretty powerful, one keeps hitting the same barriers again and again:

- grid snapping
- vertical spacing
- manipulations
- tricky language depend spacing issues
- script related issues
- multiple columns with bells and whistles
- tricky graphic (and other) placements
Typesetting: so what

We need to find a way to keep up with demands of otherwise we will become a rare species.

As there are no universal solutions extending TEX itself with more tricks is no solution.

Opening up the machinery while keeping the virtues is the best way to go.
As LuaTEX opens up TEX things can be done easier and we can move forward. We can reimplement potentially fragile currents and reconsider existing limitations in macros. We can explore solutions that demand lots of calculations and use heuristics that are not easy to implement in the TEX language. Development time (if we forget about the many man-years involved in LuaTEX and MkIV) can be brought back to normal proportions.

Yes, LuaTEX will keep me going.
So, there are three reasons why I'll stick to TEX for a while:

The Latin Modern and TEX Gyre projects: these provide a proper base set of modern fonts.

The `themplib` project: `instantMetaPost` brings down processing time to what can be considered reasonable.

The LuaTEX (and Oriental TEX) project: without a flexible and extensible engine, writing macros that match today's demands would have become close to impossible.

Of course, if I had foreseen how much time would be involved in adapting ConTeXt to this new situation, my opinion could have been different.