

Why T_EX?

1 Some Remarks

Although ‘T_EX’ is a household word in fields such as computer science, mathematics, and physics, it is not so well-known in the humanities where, unfortunately, the word processor and WP paradigm (and its high-end sibling the *layout processor* or LP) is more popular. For this reason we should say a few words about T_EX and the superior advantages of its paradigm for the purposes of high-quality scholarly typesetting. We will discuss the following issues:

1. *Typesetting* and *structural typography* versus *word processing*;
2. The virtues of T_EX;
3. The defects of _____ (Place your favorite general WP or LP here);

2 The Woes of Word Processing

The preparation of a given text for publication in print or for on-line display involves two distinct aspects:

1. Matter.

This aspect involves the expression of thoughts and ideas to be communicated to the reader, viz., the composition of the actual words and sentences. This is entirely the responsibility of the author of the text;

2. Form.

The form of the text in turn involves three elements:

- A. Logical structure.

This involves the organization of the textual material into various logical divisions, from

- I. the more global level of divisions such as parts, chapters, appendices, preface, index, and bibliography; to
- II. the more mid-level divisions such as sections and subsections; to
- III. the lower level paragraphs, quoted paragraphs, itemized lists, parenthetical text, notes, right down to which expressions to emphasize.

In other words, each expression in a given text has a role to play in communicating the ideas of the author and it is generally the responsibility of the author to determine the role of each such textual element;

B. Typographical structure.

Given the document matter, the types of logical division to be employed, and the identification of each textual element with a particular logical division, issues such as the following must be considered:

- I. page design e.g., the dimensions of the page;
- II. layout e.g., amount of paragraph indentation, interline/interword spacing, placement of headings, choice of marginal notes, footnotes or endnotes and how they are formatted, setting of columns and tables, sophisticated and aesthetically pleasing vertical and horizontal spacing between each of the textual elements on a given page, use of color and figures;
- III. typefaces e.g., given a logical division of textual element, what font is most appropriate and at what size? This issue involves a lot of subtleties with respect to ligatures, kerning, metrics, and word spacing.

Typography especially is a highly developed art and expertise traditionally requiring years of apprenticeship with a qualified master. Until recent times it had been the sole responsibility of a specialist, namely, the typographer;

C. Typesetting.

It is the job of the typesetting machine to accurately and precisely implement the chosen typographical structure. Textual material was formerly entered on a specialized typing machine and put to paper using

hot lead presses. Today in the age of *digital typography*, a computer keyboard, special program(s), and laser printer do the job of actual typesetting.

Given a text to be printed, traditional typesetting involves a *conceptual* as well as *practical* distinction between

1. The textual matter and its logical structure on the one hand (viz., *composition*); and
2. The typographic and typesetting *implementation* of the textual matter and its logical structure on the other (viz. *structural typography*).

Ideally, even within the digital typography paradigm, we still want these tasks to remain conceptually and practically distinct. Not doing so leads to confusion and to output that is unprofessional at best and ugly at worst.

Now a WYSIWYG (What-You-See-Is-What-You-Get) WP, by its very nature, takes each of the afore-mentioned elements and wraps them into a single application. On the surface this paradigm appears to offer much in the way of convenience and ease of composition of a given text. This is to some degree true for writing short letters, memos, and posters or flyers. However, for preparing longer and/or professional monographs, reports, and books it is neither convenient nor easy but rather extremely tedious and many layouts are virtually or actually impossible to implement. From a typographical and typesetting quality point of view, this paradigm has been little short of total disaster.

For one thing, a WYSIWYG user generally makes typographical decisions in the process of composing his text. Digital typography in general allows, and almost forces, an author to become a typographer, but the WYSIWYG paradigm

1. Implements advanced typography in general very poorly;
2. And therefore does not encourage typographical sensitivity;

The problem with most implementations of the WYSIWYG paradigm is that the user is first exposed to (through the toolbar and menus) to *typographical structure* options as opposed to *logical* options. And it is the latter options which are much more important to the task of *composition*.

Consider the following example: You are typing in your WP and decide to emphasize something. There is not an immediate “Emphasize” button but rather buttons for italic, bold, and underlining. You make a decision, push the italic button, and get your emphasized expression. Somewhere in a later section you need an emphasis and use the bold button. At another point you slant something for emphasis. Soon your document has an inconsistent feel and gives a headache to a serious reader.

Let’s say you’re disciplined enough to only use italic for emphasis. Well, what happens when you have to emphasize within emphasized text? Having to continually make these kinds of decisions naturally breeds inconsistency even in the most disciplined composer. And you have the added mental strain of having to always keep track of all this.

Even supposing that you are super-disciplined, what happens when you decide that the document will look better if you use slant for emphasis instead of italic? Now you have to go back and change each individual instance of emphasis, as well as make sure you don’t change the italic in those places you used it, not for emphasis, but for e.g., special concepts.

And there are so many textual elements to keep track of. Maintaining the accuracy of a bibliography’s references, accurately placing headings and footnotes, making sure that a subsection of a subsection is always formatted consistently; the list goes on and on. If you are trying to set a critical edition with all sorts of numbering and footnote conventions you may well nigh have a nervous breakdown.

For very short documents these kinds of things are not so problematic but in a longer one they become a real nightmare. Although some WP’s do have features to identify logical divisions of text, most do not and in those that do those features are easily misused because the very paradigm itself in practice generally does not encourage typographical over logical structure. By its very nature it confuses the tasks of composition and structural typography which nearly inevitably leads to unprofessional output.

Even for the professional typographer disciplined enough to manage all this, his own expertise very quickly comes up against the technical limitations of the WP and LP paradigms. In the next section I summarize a few of the difficulties with some of the WP and LP software on the market.

3 Why not WP/LP X?

Microsoft Word, the most well known of the WP paradigm:

1. Has never been really stable or reliable;
2. A given DOC file produces different results on different versions of Word, so all the painstaking effort you put in getting a perfect document in version 7 is broken in version 8. A user on the comp.text.tex newsgroup reports: *I used Word for version 1 and 2. But had to deal with many bugs. What made me fed up with Word was the printing of the last version. I had to insert manual page breaks (!) to make the TOC [table of contents] and Index corresponding [sic.] to the page numbers! Incredible!;*
3. Is not available for all operating systems;
4. Limited structural typographical capabilities;
5. Proprietary software and format; can't make any changes or improvements. Expensive add-ons do exist for specialized purposes, but these extensions are generally proprietary and as well, and may only work for certain versions of Word or operating systems;
6. Monopoly control continuously used to gouge users and force unneeded upgrades.

Word Perfect, arguably the best of the WP paradigm:

1. Unstable and unreliable, especially recent versions; frequent crashes
2. Hardware-dependent: frequent crashes for some display drivers; printer-dependent font-metrics;
3. Limited structural typographical capabilities; no automatic ligatures, for example;
4. Is not available for all operating systems;
5. Proprietary software and format; can't make any changes or improvements;

QuarkXPress LP:

1. Limited structural typographical capabilities; hyphenation & justification controls rather primitive; requires too much manual intervention for tasks that could and should be automated;
2. Plugins for extra functionality are frequently buggy and unstable;

3. Poor default fonts (e.g., fake small capitals);
4. Platform dependent;
5. Proprietary and expensive; user at the mercy of the vendor. One former user (comp.text.tex) writes: *An example of how futile/frustrating this can be is that the one XTension [Quark plugin] which could hang punctuation [very important typographical operation] in Quark XPress 3, ceased to be able to do that in v4, and there was no explanation from Quark on why the API [application programming interface] which enabled it was taken away.*

Adobe Indesign LP:

1. Limited typographical capabilities; requires too much manual intervention for tasks that could and should be automated. A former user (comp.text.tex) writes: *I tried to use InDesign for one of my reports the other day. I switched back to T_EX when I realized that InDesign can't do bulleted and numbered lists. Yes, you can type them into InDesign and make them appear on paper. [But]it's a three-[part] operation. You can't refer to them, you can't have automatic renumbering of them, etc. I don't know why people want to use the stuff.*

According to another: *Doing a long document with footnotes, lists, and bibliography entries in InDesign would be torture. There are plugins available (<http://www.virginiashsystems.com/InDesign.html>, for example) to help with this, but it's an extra cost and I can't speak for the quality. With LaT_EX/BibT_EX [part of the T_EX family], these things simply work.;*

2. Platform and hardware dependent (needs very powerful hardware);
3. Proprietary and expensive;

Framemaker LP:

1. Limited structural typographical capabilities; very primitive kerning and hyphenation & justification controls; very poor measurement handling; only one undo level (!);
2. Poor color and PDF output;
3. Proprietary and expensive;

4 A Caveat

The better WP's and LP's do have their rightful place in the world of digital typography, however. For projects with a greater emphasis on *design* as opposed to *structure*, they may even be more efficient than T_EX-based solutions (though there is little in this regard that cannot in principle be done in T_EX; see also the leaflet on L^AT_EX accompanying this article). A National Geographic oversize coffee table book on the beauty of the Rocky Mountains, with more figures than text, is just the kind of thing, e.g., InDesign does well. An opera poster in fancy calligraphy is perfect for a specialized calligraphy application.

For more mundane documents like letters and short reports, a multilingual WP like Universal Word is sufficient for e.g., Arabic, Hebrew and Latin script. But the structural typography demanded by professional scholarship is ill-served by this paradigm of typesetting. Donald K. Knuth, the author of T_EX, puts it best:

I never expected T_EX to be the universal thing that people would turn to for the quick-and-dirty stuff. I always thought of it as something that you turned to if you cared enough to send the very best. (<http://www.advogato.org/article/28.html>)

5 The Virtues of T_EX

T_EX is a full-featured typesetting system including a formatting language, a primary typesetting engine (T_EX proper) as well as subsidiary ones (like Omega and pdf_ET_EX), a huge expert font family and font-creation program, a number of integrated drawing engines, a number of high-level user interfaces or *formats*, and numerous other utilities.

The T_EX typesetting system works under a paradigm very different from that of WP's and LP's. It allows for a much cleaner separation of composition and logical structure from typography and typesetting. One can create a T_EX document on virtually any text processor, even a WP or LP. Typically, however, the user of T_EX composes his document from within a *text editor* and saves it as a simple text file that can be opened by nearly any other text-processing application or utility.

Analogous to HTML/XML, the author composes his text and marks up the logical structure as he goes along according to the conventions of a *macropackage* such as L^AT_EX or C_ON_TE_XT. L^AT_EX is like a built-in typographer; you give it the logical structure, it will build a beautiful document to

match that structure. Much less worrying about bold, italic, font size, and the like. `CONTEXT` is for total typographic control; one can consistently setup virtually any needed typographical scheme within which to implement your logical structure.

The typographic and typesetting implementation of the textual matter is setup either in a separate text file or, for shorter documents, in the *preamble* of the main document before the matter. An example of this in action: You have many expressions to emphasize. Instead of making explicit font assignments you mark each such expression with a tag, say, `\em`. In the setup area `\em` is set to italic. But after doing hundreds of emphasized expressions you decide to implement emphasis in a different font. Just make an adjustment to the definition of `\em` and you're done; `TEX` will now typeset the document in accordance with the new definition. So the author can focus on composition and logical structure as one task, and structural typography as another. As in traditional typesetting, the two tasks are kept conceptually and practically distinct; the `TEX` paradigm naturally encourages authors to comply with this separation. Doing it any other way in `TEX` paradigm is actually *difficult and painful*, in contrast to the WP paradigm where ignoring the separation of these tasks comes naturally and easily.

5.1 Typography & Typesetting

`TEX`'s structural typography capabilities are unmatched by any other publically available commercial product. This is especially the case for mathematical typesetting, where `TEX` is *the* standard. It can handle the most complex tables, indices, hyperlinks for interactive display, and cross-references (notice how few Word documents have an index or many cross-references). `TEX` automates the typesetting of lists descriptions, labels, tables of contents, bibliographies, and other tasks.

One of `TEX`'s greatest achievements is its *line-break algorithm*. This involves making decisions about line justification and word hyphenation with respect to, not the individual line, but the entire paragraph. This gives paragraphs a very even and professional appearance. The behavior of this algorithm can be fine tuned by the user as well for special purposes. Very few other programs have this feature; nearly every WP justifies only line by line. `TEX`'s quality is thus manifest even on very short documents. `TEX` also loads different hyphenation patterns depending on language: US English, UK English, French, Romanian, Turkish; virtually every hyphenation pattern is supported.

TeX also recognizes that the space after a period at the end of a sentence in the middle of a paragraph is different from the space after a period in an abbreviation like “U.S.A.”. This kind of attention to detail contributes towards making TeX output generally much more professional and pleasing than those created by WP’s.

TeX also includes some excellent facilities for drawing illustrations that fit in nicely with the rest of the text. Too often diagrams done in a drawing program do not match the text of the text processing program. TeX programs like METAPOST and PSTricks facilitate the creation of figures that are most harmonious with the rest of your document.

5.2 Fonts

TeX’s capabilities for font handling are among the best. Almost any *expert font* [font with extra ligatures and special/exotic characters] can be installed and manipulated in creative ways (Alan Hoenig has written a nearly 600–page book *TeX Unbound* nearly half of which is devoted to all the fancy things TeX can do with fonts).

TeX’s own very complete and beautiful font family, *Computer Modern*, contains virtually every needed font needed for typesetting modern texts; 32 (75 if we include each optically scaled version) well-matched and mutually harmonious fonts with ligatures (as in ‘fluffier’ and ‘muffle’). Another feature of TeX is its support of *optical scaling*. That is, each font size is separately designed to look best at that size. This is very important for a professional visual appearance. Consider the following two examples:

This is 17-point Computer Modern at 17

This is 10-point Computer Modern magnified to 17

Notice how the real 17-point font matches the rest of this document in an economical way. METAFONT, a font creation program that is an integral part of the TeX family, can generate optically scaled fonts from a single design. The Computer Modern fonts are *parameterized*; by changing the parameters in the original designs you can generate your own custom fonts that will match the rest of the Computer Modern family.

Before the digital typography age, hot metal typography in particular implemented optical scaling, but this died out as the printing industry moved to phototype. With few exceptions, the overwhelming majority of fonts came to

be based upon a single design size and then scaled as needed by WP's and LP's. Adobe through its Multiple Master technology (MM) has provided a solution for font designers to again implement optical scaling for digital typography. However, very few fonts take advantage of it, and there are virtually only 2 applications that can properly take advantage of MM technology, one of the being Adobe's own InDesign. The other, of course, is T_EX.

5.3 Flexibility

One of the strengths of T_EX is its flexibility. At its core is a Turing-complete, low-level formatting language that can be adapted to a variety of needs and paradigms. From the low-level formatting commands one can build high-level macros or packages for the average end-user to accomplish particular tasks. The two most general and complete packages for structural typography are L^AT_EX and CON_TE_XT. But the low-level commands are always available for the truly exotic or unforeseen cases.

Because the source code to T_EX is freely available, anyone can add low-level extensions to perform tasks not easily handled by the original. This is what e.g., the Omega project has done. It has added multiple typesetting direction capability to the formatting engine as well as support for fonts with up to 65,536 characters. Both features are needed to make T_EX do e.g., Arabic-, Hebrew-, and Japanese-script typesetting as seamlessly and natively as it does Latin-script.

6 Stability, Consistency & Portability

One of the unique features of T_EX is the rock-solid stability of the typesetting engine and formatting language. Through the combined efforts of the author, users, and developers around the world, there is hardly any large-scale software project that has been as thoroughly debugged. In fact, there has apparently only been a single bug found since 1996¹!

T_EX is not only stable but it's also consistent. All else being equal (e.g., font setup) and bug fixes aside, a document written 10 years ago will be typeset by T_EX today *exactly* as it would have done it then. Not only that, the output will be identical on every implementation of T_EX. On the other

¹ We say “apparently” because the bug has yet to be documented and verified.

hand, a document done in the latest version of Word on Windows will often look different from the same document opened in Word on the Macintosh!

T_EX has been ported to virtually every operating system and platform. Its users are not tied to any operating system or hardware. The availability of this software for virtually any computer is especially liberating for authors around the world and for the free flow of information without the interference of any corporate software or hardware monopoly.

The fact that a T_EX source file is just a plain text file has a number of advantages over the binary formats used by WP's and LP's, among them:

1. You can use any text editor on any platform to compose, edit, or view it, including a WP;
2. For archiving (e.g., in a zip file) it compresses very well;
3. Less prone to data corruption, and easier to fix when such corruption does occur;
4. One can save virtually any encoding in text format (important for multilingual work) for processing by T_EX;
5. While T_EX will output PDF for you one only needs to maintain and store is the text file;
6. Text files are also most easily manipulated by database utilities for conversion to other formats and so forth. As times change and text formatting methods progress, the handling of a text file can always be adjusted to fit the circumstances.

7 Output

From a T_EX file, one can generate a PostScript or PDF file for printing, be it on a home machine or on a high-resolution photo-typesetter. For viewing T_EX can produce the most sophisticated interactive documents, with color, cross-references, and weblinks. One can even compose a given file such that the same file produces one pdf file for print publication and another with a very different look for interactive screen display, complete with menus, hyperlinks, even multimedia features. The T_EX source file can be converted to HTML/XML as a basis for even further processing such as for voice recognition or translation software. The possibilities are limitless.

8 Standard

The above partly accounts for the fact that T_EX has become a major standard in the publication industry. Through books on T_EX, the comp.text.tex newsgroup, and other forums, support and answers to questions are easy to find. Macropackages and extensions based on this standard and their users will be able to tap into the huge well of T_EXnical knowledge for suggestions and solutions to difficulties.

9 Free versus Proprietary

One of the most important features of the T_EX paradigm is its *openness*. Anyone can inspect the source code, make fixes or improvements to it, and adapt it as needed. T_EX is copywrited and allows for commercial implementations as well. But the beauty of it is that a user need never be tied to any commercial implementation. Proprietary programs on the other hand restrict your freedom to chose your own operating system and hardware. You are completely at the mercy of the vendor for updates, and if the vendor goes out of business you're left in the cold with hundreds of pages of carefully tuned documents which can't be processed by other applications exactly as you intended. Mention was made earlier of how QuarkXpress changed its application programming interface so that a feature needed by a customer no longer worked.

9.1 Summary

In short, the T_EX paradigm is the most ideal for the implementation of a full-featured typesetting solution for most structural typography processing. It is superior to the WP and LP paradigms, can output both for high-resolution print and on-line interactive display, can be converted to other structured formatting standards, and is a stable, open system.

Idris S. Hamid: June 23 2002