
Book Reviews

HEIMUT KOPKA and PATRICK DALY

A Guide to L^AT_EX. Addison-Wesley, 1993. £24.95, 436 pp. softbound. ISBN 0 201 56889 6.

AANTONI DILLER

L^AT_EX Line by Line. Wiley, 1993. £19.95, 291 pp. softbound. ISBN 0 471 93471 2.

MALCOLM CLARK

Plain TeX Primer. Oxford University Press, 1992. £19.50, 481 pp. softbound. ISBN 0 19 853 7247.

Scientists seem to spend much of their lives writing documents. The tools for this activity, then, can be as important as their laboratories or computers. Document preparation systems broadly divide into WYSIWYG (what you see is what you get) and logical markup. In the latter, authors are encouraged to define the structure of a document (saying what elements of their text are lists, what section headings, and so on), leaving the actual rendering of these elements into specific typefaces to a document designer. The superiority of logical markup as a typographic technique is uncontested: indeed, 'desk top publishing' is a contradiction in terms, in that if a system will fit on your desk, it is unlikely to produce publishable quality output. Logical markup allows authors to concentrate on the *content* of their documents, leaving the details of typesetting to publishers (who tend to be rather better at it).

One of the most widely used markup systems is Leslie Lamport's L^AT_EX program. This allows a user to specify the structure of a document which is then realised by one of a number of *styles*. Thus the same user file can produce radically different pages depending on how the style translates logical information (such as section headings) into marks on the page (such as a piece of text set in Times Bold 14pt).

The original guide to L^AT_EX written by Lamport has not suited all users. On one hand, it offers quite detailed information on the various standard styles: on the other, it aims to nurse the naïve user through the elementary stages of document preparation. In trying to achieve both of these goals it succeeds in neither; hence the recent appearance of several explicitly introductory books on L^AT_EX.

One of the best of these is Kopka and Daly's *A Guide to L^AT_EX*. This is an English translation, clearly by a technically able translator, of a German introduction to L^AT_EX. The book claims to be suitable for those with little or no knowledge of computers. This I doubt, for there are many allusions that require a touch of geekery, but it does provide a reasonable introduction to markup in L^AT_EX.

The book has some notable flaws: one feels that the point of logical markup has sometimes been missed. There is too much encouragement for authors to experiment with page layout, a confusion between L^AT_EX-

level working and the underlying typesetting engine, and too little advice on the use of fonts. Perhaps this last is due to its origin; German typesetting has always demonstrated a fondness for sans serif fonts, something that, in the wrong hands, can lead to incredibly ugly layouts. The example letter format Kopka and Daly show is one of the worst examples of this, a hideous mix of an inappropriate sans serif heading font with a seriffed body font. (It is no coincidence that good German sans setting is based on Univers-like fonts, which read much more smoothly than the computer modern sans serif L^AT_EX provides.) There are advantages to the book's continental nascence, though: the material on multilingual working is particularly clear. In fact, minor blemishes aside, most of text is correct, readable, and informative. Thus Kopka and Daly's book would make a useful if slightly uninspiring addition to the naïve L^AT_EX user's library, although they will probably need another volume (such as the forthcoming *A L^AT_EX Companion*, by Goossens, Mittelbach and Samarin, or Lamport's book), for occasional reference.

Kopka and Daly have written an explicitly instructional book. Diller, in contrast, in *L^AT_EX Line by Line*, has written a book for those without enough time to learn L^AT_EX properly. It is perfect for picking up, when a typesetting problem occurs at the terminal, and finding a not-quite-right solution that will almost do. The book's style makes it impossible to read continuously—it seems as if there are more occurrences of the first person singular in the introduction than in the whole of *Daniel Deronda*—and the coverage is decidedly patchy. Diller clearly typesets mathematics regularly, as his coverage of this area is fairly sound. Other chapters, though, notably where control and macro features are tackled, are confusing, scant or erroneous. Even a naïve user would quickly need features not described in the book, so it cannot pretend to be a self-contained manual.

Various features do recommend Diller's book, notably the use of interesting and insightful examples throughout the text, but they are not enough for this reviewer to prefer it to *A Guide to L^AT_EX*.

L^AT_EX styles are written in TeX, an underlying language in which the actual mechanics of typesetting (line breaking, font selection and so on) are realised. TeX has the not entirely undeserved reputation of being a difficult and obscure language to program in. Until recently, just as Lamport's L^AT_EX book was the only reliable guide to L^AT_EX, so Donald Knuth's *The TeX Book* was the only guide to TeX. There are now a variety of replacements of which Clark's *A Plain TeX Primer* can be recommended.

This book is a careful and thorough introduction to TeX. In places it is difficult reading—one wonders if the average reader is ready for a discussion of the rather technical matter of tolerance as early as page 44—and

it tends towards the patronising, but there is little actually wrong with it, and the various chapters make generally much easier and clearer reading than material on the same subject in *The TeX Book*. It is also a very interesting book for something that is essentially a manual; I found myself reading well beyond a specific query, so beguiling is Clark's style. One may criticise his dedication to Knuth (a more distanced and objective attitude to the failures of TeX would strengthen the book), his rather *recherché* sense of humour, and much of the book's typesetting (one only hopes Clark did not actually choose to set his exercises as they appear in the final text), but there is much to praise here: the discussion of the notoriously difficult topic of output routines is very clear, for instance, and most of the control structure TeX offers are covered in a very well-judged amount of detail. Clark's book is not comprehensive enough to act as a definitive reference, (there is little material on dvi drivers, or on including graphics from other packages, or on sound book design) but it is a worthy attempt at explaining TeX to the uninitiated, and an adequate reference source. I welcome its appearance.

D. MURPHY

University of Birmingham

SAUL GREENBERG

The Computer User as Toolsmith: The Use, Reuse, and Organization of Computer-based Tools. Cambridge University Press, 1993. £27.95, 187 pp. hardbound. ISBN 0 521 40430 4.

The word 'toolsmith' conjures up a picture of a person wearing safety goggles and gloves, a pair of tongs holding a piece of incandescent steel in one hand and a hammer in the other. Transfer this image from the smithy to the software house and we see a software engineer concentrating in front of a workstation, using an editor, a compiler and a debugger to track down the bugs in a newly-crafted program. This book is not actually about the smithing of software tools, the creation of programs, but about their use and re-use, as a branch of Human Computer Interaction (HCI).

The kitchen provides an image of a well-organized user of tools: the cook at the stove, pans hanging from hooks, knives thrust into a block and small gadgets in a drawer. A thesis of this book is that the computer user is not similarly well organized: people have not transferred the strategies they use for arranging and selecting physical tools into interactive computer systems. Greenberg explores this thesis and the lack of knowledge on how people organize their on-line computing work. He presents a new study of computer usage and draws from it some general principles on re-use facilities. The last section of the book describes a computer 'workspace', based on grouping software tools by task and function.

The study of computer usage is of a UNIX system at

the University of Calgary, with staff and student users of the command shell *csh*. From this narrow but focussed study, Greenberg draws some interesting results on command usage. Experienced users do not share command sets. The Zipf distribution models the frequency distribution of command usage by groups of users but not necessarily by individuals. Commands cluster or follow each other in patterns, but these patterns vary between individuals.

Greenberg suggests that past studies have over-emphasized command usage as opposed to command lines, which also include options and arguments. His interest in the re-use and organization of command lines leads to a study of re-use facilities including history mechanisms, adaptive systems, and programming by example. Although the history facility provided by *csh* is found to be used very rarely, its potential is explored by developing different methods for 'conditioning' a set of command lines for re-use, with some success.

General principles on how users repeat their actions are derived from the empirical data on *csh* use. These are then reformulated as six guidelines for design of re-use facilities supporting the use of past history.

The final chapters draw together previous threads into the topic of computer workspaces, a means of organizing software tools. Greenberg discusses previous guidelines for workspace design and three new ones involving the use of symbols to represent activities, the ability of a user to customize his or her own workspace, and the use of past activities to construct a workspace. He relates these guidelines with five existing workspaces and concludes by presenting WORKBENCH, a graphical window-based front-end to *csh*, containing a work surface, a tool area with tool panel and re-use facility, and a tool cabinet.

For a close look at some of the problems associated with user interface design, *The Computer User as Toolsmith* is worth reading. It does not look in detail at Graphical User Interfaces or non-command based interfaces: see [1] on the next generation of GUIs for example. What its narrow focus on a traditional, command-based, general purpose computing environment does provide is an in-depth study with conclusions of general interest and applicability. In the foreword, Harold Thimbleby claims that the book is 'the starting point for evaluating and improving almost any existing or future system'. To users of computers and software tools who are not specialists in HCI, the book gives a most interesting insight into both a branch of HCI research and also their own everyday use of computer-based tools.

REFERENCES

- D. Mandelkern, (ed.), Special issue on Graphical User Interfaces: The Next Generation. *Communications of the ACM*, 36, pp. 39–109 (1993).

J. A. DAIN
Warwick University