LaTeX 2.09 ↪ LaTeX3

Frank Mittelbach and Chris Rowley

Abstract

This is a brief sketch of the LaTeX3 Project: its history, its present state and its future, as at the end of 1991. It is based on a talk given by Frank Mittelbach, technical director of the LaTeX3 Project, in November 1991 to a meeting of the Nordic TeX Users Group; the handout from this talk was recently published in TeXline [Mit92].

1 The LaTeX3 Project

The LaTeX3 Project was initiated at the Stanford annual meeting in August 1989 but the first, somewhat vague, plans had already been formulated in 1988 when Rainer Schöpf and Frank, after sending several pages of bug fixes for LaTeX 2.09 to Leslie Lamport, received a positive answer.

1.1 History

Whenever the future is somewhat unpredictable it seems wise to take a look into history — to find out what has already been achieved and what remains to be tackled.

From the now quite long history of the LaTeX3 Project we can observe a growing bulk of syntax descriptions and (partial) implementations that are the results of three years work.

Looking at the original goals for a reimplementation, described in the Stanford paper [MS89], it would appear that nearly everything has now been achieved.

- The NFSS, which is now in widespread use, provides a very general font selection method. The extended syntax, on which we are now working, also provides for scaled fonts.
- With `amstex.sty` the mathematical capabilities of LaTeX have reached (at least) the standard of $\mathcal{A}\mathcal{M}\mathcal{S}$TeX.
- With the new implementation by Denys Duchier (this supersedes `array.sty`), together with valuable suggestions by several others, tabular processing will have reached a very high standard.
- A new error recovery and help system is being developed. This should provide a safe and easy-to-learn environment for novice users.
- A more flexible input language is being developed, in which environments and commands can have attributes specified. This will also allow easy conversion from SGML to LaTeX3, for suitable DTDs.

Milestones:
Syntax and implementation

1988 – Some bug fixes sent to Leslie Lamport
– Four page sketch of NFSS
1989 – First implementation of NFSS
– Implementation of `amstex.sty`
1990 – New tabular implementation by Denys Duchier
– First attribute prototype
– First kernel prototype
– First recovery/help prototype
1991 – Second kernel prototype
– Sketches for style designer interface
– Second description of the attribute concept
– Extended description of the help facility
– Syntax for extended NFSS
– Third kernel prototype
– Release of LaTeX 2.09 international with NFSS support

How did all this happen?

All the work has been carried out in the free time of several individuals and involves, as you can imagine, a lot of enthusiasm to keep the project alive. So far, more than thirty people have contributed in one way or another to the effort.

One of the major, and growing, problems is how to bring people from all over the world together to discuss the open questions and find new solutions. It is important that these meetings involve people from outside the project since we very much need the views and experience of typesetters, designers, publishers, etc. to help eliminate the flaws in the system and find new and better solutions.

In this regard both the London and the Dedham workshops were hugely successful, but further workshops of this kind are essential if we are to provide LaTeX3 with a suitable designer interface.

1.2 Current state

So why are we now saying that this project is still only at its start? Because we have learnt that our

Milestones:
Meetings, Workshops and Correspondence

1988 – Non-flame answer from Leslie Lamport
1989 – Talk in Stanford
– Meetings with Leslie in Stanford
– Talks in Karlsruhe
1990 – Mega-bytes of e-mail correspondence
– Working week in Mainz with Leslie
– Talk in Cork
1991 – More mega-bytes of e-mail correspondence
– Workshop in London
– Meeting with Leslie in London
– Workshop in Dedham
– Working week in Providence with Michael Downes
– Working week in Mainz

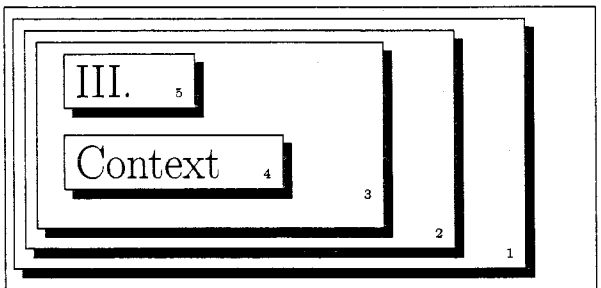
original goals did not touch many of the real problems as we see them today — there has thus been a major change of focus.

Working on the project has led us to the conclusion that one gains very little by just providing more and more specialized style files to solve this or that special problem. Instead, we think that, in order to provide a fully flexible and extensible system, the major effort in the future must go into the design of the right style interface — one that allows easy implementation of various layouts. (Easy, of course, is relative: easy compared to the complexity of the task.)

The project plan which emerged from this change of focus can be summarized as follows.

- The development of a new internal language that is more suited to the expression of visual components of the layout process.
- The development of high-level generic functions that allow the straightforward expression of most commonly used layout components.
- The development of a model for specifying and modifying all of the parameters that influence the layout.

Since the syntax for this internal language is still changing on a daily basis, and the generic functions depend on it, in this article we shall concentrate on the *model* for parameter setting rather than its detailed syntax.



1. document context
2. section context
3. heading context
4. title context
5. number context

2 The model for parameter setting

The new system will implement a model of document formatting based on the fact that the formatting required for any particular part of the document depends on its context. Therefore the setting of layout parameters must be context-sensitive.

2.1 The concept of a context

What do we mean by a 'context'?

The nesting of the entities in a document is clearly important for the specification of its layout. For example, in LaTeX2.09 a second-level list (i.e. a list nested within another list) will be treated, by the standard styles, in a different way from a first-level list.

The order (or sequencing) of entities is also important. For example, the space after a section head will probably depend on what comes next: ordinary text or another, lower-level, head.

However, the context of an entity in a document is not given simply by the nesting and sequencing of its surrounding entities. Take, for example, what happens when some footnote or float that appears in a list itself contains a list. Because of the incorrect handling of contexts in the current LaTeX the inner list is typeset as a second-level list. In other words, an entity must be able to (partially) forget about its context or, more generally, must be able to manipulate its context.

Moreover, the context of an entity also has a 'visual' component — this is its relationship to the rest of the document *after* it is formatted. This 'visual context' usually depends on the formatting of many other entities in the document.

For example, the optional argument to `\marginpar` enables the author to specify that the

text of a marginal should depend on its visual context, i.e. on whether it finally appears on a recto or a verso. Thus its formatting depends, at least, on the formatting of all the entities which appear on nearby pages.

To summarize, at every point in the document we are in a logical context given by the nesting and sequencing of all the entities in the document; and we are also in a visual context determined by decisions made about the formatting of all the entities in the document.

2.2 Contexts in LaTeX3

Therefore the new system will, as far as possible, allow the settings of all the parameters (needed to specify the formatting of an entity) to vary according to the current context, both logical and visual. Here the idea of 'parameter' is used in a very general way — e.g., the particular generic function used to format some entity is considered to be a parameter so that, via this concept, different generic functions can be used in different contexts.

For example, it will be possible to change the layout of certain entities within floats by specifying the values of their layout-parameters in the context of "floats" differently from those applied in the main text.

- In this article, lists within floats are indented at both the left and the right margin (we had to do this the hard way for LaTeX2.09).
- A table entity in a float can be set in a smaller typesize than one in the main text.

Another case where the nesting of entities may affect the formatting is a list within a tabular p-entry: such a list may well be typeset 'in-line', i.e. a new item does not start a new line.

It will also be possible to specify formatting parameters dependent on the sequencing of entities. This is needed for design specifications such as: "The vertical space after a theorem is 3pt if it is immediately followed by a proof, otherwise it is 6pt." Another example is the specification of no indentation when a paragraph immediately follows a certain entity, e.g. an article title.

Visual-context-dependent specifications which will be implementable in LaTeX3 include cases where the formatting depends on page-breaks or column-breaks; where it depends on the type of page; and where it depends on the types of objects appearing on a page. For example: "A section head should be preceded by a text-width rule and have 5 lines of text after it before a page-break. If it appears at the top of a page then the rule is omitted."

2.3 Some problems

Implementing such a model within the TeX framework poses some interesting challenges.

1. The user input is not (in SGML jargon) a 'normalized document' — i.e. it may contain entities which are hidden inside user-defined shorthands so that they cannot be easily prescanned.
2. The specification of contexts in terms of the sequencing of entities is important but its efficient implementation is restricted somewhat by the underlying TeX engine.
3. Taking the visual component of contexts into account requires the use of a multi-pass system, and even then it is, in practice, restricted by the formatter's capabilities.

Let us look at the first of these problems in more detail. The problem here is that the document (`.tex`) file is not a normalized form of the document (one in which every entity is explicitly tagged) thus we cannot, before we start typesetting an entity, scan forward using TeX's native scanner to see what 'begin' or 'end' tags are coming up. This means that certain formatting decisions have to be taken without knowing what follows. Possible solutions to this problem are

- forbid the use of user-defined shorthands
- do the scanning 'by hand' using TeX's macro language
- use a two-pass system that normalizes the document in the first pass
- use a multi-pass system in which each pass uses information on sequencing gathered during the previous pass.

The first of these would be unacceptable as it would remove one of the strengths of LaTeX, its flexibility. The second would vastly increase processing time since TeX's native scanner is highly optimized. The latter two do not seem to be feasible for a system that uses TeX as the input language, but this should be explored further.

With respect to the second problem, TeX has no built-in mechanism to detect whether simple character material is about to be contributed to some horizontal list immediately after a given tag. Only after the material has been contributed to the horizontal list can one deduce this fact by 'dirty tricks' involving special kerns. Thus this can be used only for interrupting a context sequence — the already contributed material cannot be manipulated further.

The last problem is of a more theoretical nature. All of the currently available automated-typesetting

systems format each document entity in a predetermined visual context, i.e. they assume that the visual context can be determined completely by the logical nesting and sequencing of entities. To a certain extent TEX is an exception as it applies dynamic programming to the process of paragraph formatting, and this involves the recomputation of contexts for ligatures etc.; the use of this kind of process needs to be extended much further in order to produce automated typesetting of high quality.

For example, in TEX there is very little interaction between the paragraph-building mechanism and the page-building mechanism. Thus the way in which TEX avoids a hyphen at the end of a page (a good example of the visual context affecting the formatting of a word) is by moving the whole line instead of by rejustifying the paragraph to avoid a hyphen at that point. These matters are discussed further by the authors in [MR92a].

This type of consideration draws the boundary between the ideal model and the real world. It also reveals that certain optimizations in the TEX language necessarily restrict the flexibility of high-level front-ends built on it. (Since TEX is a Turing machine, it is possible to move this boundary but not without an unacceptable increase in processing time.)

So let us now turn to an overview of the major components of the LATEX3 system.

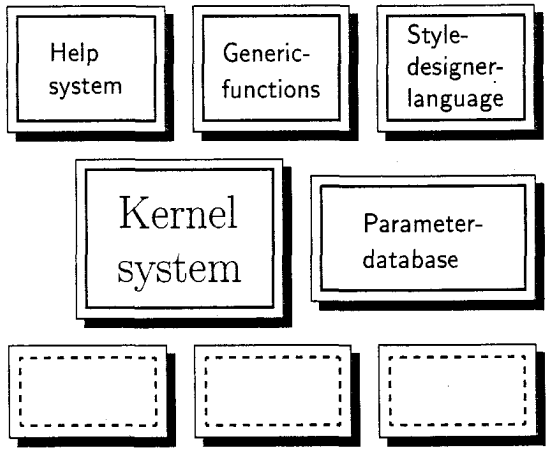
3 The structure of the system

The LATEX3 system contains many interwoven structures each of which could be the subject of a separate article. The kernel of the system provides the basic data structures such as lists, stacks, etc., used to program higher-level modules. It will also contain arithmetic functions for integers and dimensions, e.g., it will be possible to express relationships between individual parameters by specifying assignments that contain expressions.

On top of it are built the generic formatting functions and manipulation functions for the parameter database allowing context-sensitive parameter specification. Together these will form the basis for the style designer language.

One other important component of the system will be an interactive help system allowing extensive help texts as well as the possibility of defining system reactions dependent on user actions. Help messages and such additional error-correcting code will be held in external files that are read in when an error is detected by the system. In this way an elaborate, interactive help and error-handling mechanism will be possible whilst keeping the LATEX3 kernel compact.

The Structure of the LATEX3 System Modules



It is important to distinguish between document styles — these are written in the style designer language (and contain no, or nearly no, TEX code in the traditional sense) — and additional modules, some of which will provide additional functionality for use in a range of document styles (e.g., facilities for version control and change bars, enhanced graphical features, etc.) whilst others will define extra entities for typesetting specialized classes of documents (e.g., functions for critical text editions, for specialized math constructions, chemical formulas).

The new kernel will provide a programmer interface that makes the development of further such modules a reasonable task.

4 The new generation of LATEX users?

Over the last six years, the way in which TEX and its front-ends are commonly used has changed dramatically. Here is a quotation from some unknown novice of LATEX.

> "Dear Sir,
>
> I have successfully installed LATEX from the distribution — the file `sample` was just printed. However, somewhere in the README files a similar program called TEX is mentioned. Could you please explain to me how to install this program? ..."
>
> From a phone call.

This might sound funny at first, but this extreme is not far from the reality for many users (and even support personnel) nowadays.

- Today the overwhelming majority of users use LaTeX only.
- Today there are very few users who have more than a minor understanding of the underlying system, and they usually have no knowledge of the TeXbook — even of its existence.
- Today most users can be nicely classified as "has heard of 'macros', but has never seen one".

This does not mean that today's users are less intelligent than past users, but simply that they are using a plug-and-play TeX system.

Nowadays TeX, and LaTeX as its major front-end, have to compete with the so-called Desktop Publishing Systems. To keep them alive we have to bridge the gap between the 'implementor/wizard' type of user of the '80s and the new type who uses the system just as one tool out of many, with no understanding of its internals.

With the LaTeX3 Project we hope to achieve this goal.

5 Establishing contact with the project

There is now a public list for discussion of the design of LaTeX3. This list is *not* for questions about, or discussion of, how to solve problems within the current LaTeX — so please refrain from asking questions such as "How do I avoid getting double-spaced tabulars whilst using `doublespace.sty`?" But, of course, it is OK to point out problems with the current LaTeX which should be addressed by the new version. We must stress this request not to misuse the list, otherwise there will be no new LaTeX before 2001!

We are now at the stage where we can identify sub-projects: these will vary in type and size, and will not all require advanced TeXnical expertise. Anyone seriously interested in finding out more about such (voluntary) work should contact one of the project managers.

The TUG office is administering a fund to help with the expenses of the project: further details of this can be found in [MR91], or [MR92b].

Contacts

As at February 29th 1992, the e-mail addresses of the LaTeX3 Project group managers are:

Frank Mittelbach
`Mittelbach@mzdmza.zdv.Uni-Mainz.de`
Chris Rowley
`C.A.Rowley@vax.acs.open.ac.uk`
Rainer Schöpf
`Schoepf@sc.ZIB-Berlin.de`

To subscribe to the LaTeX3 design list, send the following in an e-mail message to
`LISTSERV@dhdurz1.bitnet`:

```
SUB LATEX-L  Firstname Secondname
```

References

Further information about the project can be found in the articles listed below. This bibliography also contains entries concerning BibTeX, which will be reimplemented and enhanced by Oren Patashnik for use with LaTeX3.

[Gue91] Mary Guenther, editor. *TeX 90 Conference Proceedings*, March 1991. Published as *TUGboat* 12(1).

[Mit90] Frank Mittelbach. LaTeX 2.10. In Lincoln K. Durst, editor, *1990 Conference Proceedings*, page 444, September 1990. Published as *TUGboat* 11(3).

[Mit92] Frank Mittelbach. LaTeX2.09→LaTeX3. *TeXline*, (14):15–18, February 1992.

[MR91] Frank Mittelbach and Chris Rowley. The LaTeX3 project fund. *Die TeXnische Komödie*, 3(4):13–15, December 1991.

[MR92a] Frank Mittelbach and Chris Rowley. The pursuit of quality — How can automated typesetting achieve the highest standards of craft typography? In C. Vanoirbeek and G. Coray, editors, *Electronic Publishing '92*, Cambridge University Press, April 1992.

[MR92b] Frank Mittelbach and Chris Rowley. The LaTeX3 project fund. *TeX and TUG News*, 1(1):5–6, February 1992.

[MS89] Frank Mittelbach and Rainer Schöpf. With LaTeX into the nineties. In Christina Thiele, editor, *1989 Conference Proceedings*, pages 681–690, December 1989. Published as *TUGboat* 10(4).

[MS90a] Frank Mittelbach and Rainer Schöpf. LaTeX dans les années 90. *Cahiers GUTenberg*, (6):2–14, July 1990.

[MS90b] Frank Mittelbach and Rainer Schöpf. LaTeX limitations and how to get around them. In Brüggemann-Klein, editor. *1989 EuroTeX Conference Proceedings*. To appear.

[MS90c] Frank Mittelbach and Rainer Schöpf. Reprint (with corrections): The new font family selection — user interface to standard LaTeX. *TUGboat* 11(2):297–305, June 1990.

[MS90d] Frank Mittelbach and Rainer Schöpf. Towards LaTeX 3.0. *TEX Gebruikers Group*, (2):49–54, September 1990. Reprint of [MS91].

[MS91] Frank Mittelbach and Rainer Schöpf. Towards LaTeX 3.0. In Guenther [Gue91], pages 74–79. Published as *TUGboat* 12(1).

[Rhe90a] David Rhead. Could LaTeX do more for chemists? *TEXline*, (12):2–4, December 1990. Suggestions for LaTeX3.

[Rhe90b] David Rhead. Towards BibTeX style-files that implement principal standards. *TEXline*, (10):2–8, May 1990.

[Rhe91] David Rhead. How might LaTeX3 deal with citations and reference lists? *TEXline*, (13):13–20, September 1991. Suggestions for LaTeX3.

[WM91] Reinhard Wonneberger and Frank Mittelbach. BibTeX reconsidered. In Guenther [Gue91], pages 111–124. Published as *TUGboat* 12(1).

◇ Frank Mittelbach
EDS, Electronic Data Systems
(Deutschland) GmbH
Eisenstraße 56 (N15)
D-6090 Rüsselsheim
Federal Republic of Germany
Internet:
mittelbach@
mzdmza.zdv.uni-mainz.de

◇ Chris Rowley
Open University
Walton Hall
Milton Keynes MK7 6AA
United Kingdom
Internet:
c.a.rowley@vax.acs.open.ac.uk

Les Cahiers GUTenberg
Contents of Recent Issues

Numéro 9 - July 1991