

Escaping the Curse of the Wordprocessor

**Document Preparation with Generic
Mark-up Languages**

David Fisher

Escaping the Curse of the Wordprocessor: Document Preparation with Generic Mark-up Languages

by David Fisher

Table of Contents

Multi-format publication.....	5
1. Introduction.....	6
1.1. Objectives.....	6
1.2. Problems with Mainstream Technologies	6
1.3. Confusing Style, Structure and Content.....	7
1.4. Generic Mark-up Solutions.....	8
2. A Brief Overview of LaTeX.....	9
2.1. The Structure of a LaTeX Document.....	9
2.2. Marking-up Document Content With LaTeX	9
2.3. Processing LaTeX Sources	10
2.4. A Trivial Example of LaTeX Mark-up	12
2.5. Device Independent Output from LaTeX.....	12
2.6. Problems With LaTeX.....	13
3. LyX	15
3.1. LyX — A Half-way House	15
3.2. Problems With LyX	15
4. SGML/XML	17
4.1. The Jade-DocBook Text Processing Suite	17
4.2. Processing DocBook SGML Sources	18
4.3. Key DocBook Tools Commands.....	19
4.4. Example: A Trivial DocBook Document.....	20
4.5. Brief explanation.....	22
4.6. Editing With Psgml.....	22
4.7. Problems with the Jade-DocBook Tools Suite.....	23
4.8. SGML to XML/XSL.....	24
4.9. Conclusions and Prospects.....	25

List of Tables

List of Figures

2-1. Processing Marked-up LaTeX Sources	11
2-2. Processing Marked-up LaTeX Sources	13
4-1. Transforming a DocBook SGML File into other formats	19

List of Examples

4-1. A Trivial Document in DocBook SGML	21
---	----

Multi-format publication

This publication is available in many formats. All of them can be downloaded from <http://www.scs.leeds.ac.uk/wylug/>. If you are reading this in a web-enabled format, you should be able to download by clicking on the appropriate item:

- HTML (gzip-compressed tar file) (`textprocessing-wylug990913.html.tar.gz`)
- DVI - Device Independent File (gzip-compressed) (`textprocessing-wylug990913.dvi.gz`)
- PS - Postscript (gzip-compressed) (`textprocessing-wylug990913.ps.gz`)
- RTF - Rich Text File, for those still trapped in the proprietary prison (gzip-compressed) (`textprocessing-wylug990913.rtf.gz`)

Chapter 1. Introduction

1.1. Objectives

- Make the case for writing formal and business documents in generic mark-up
- Introduce the two most commonly used mark-up families used in Linux:
 - LaTeX and its relatives (TeX, LyX, etc)
 - SGML and its implementations (HTML, XML, etc)
- Explain their principles and potentials
- Demonstrate simple editing tools:
 - auctex – an emacs plugin for editing LaTeX
 - LyX – a pseudo-wysiwyg editor for pseudo-LaTeX
 - psgml – an emacs plug-in for editing SGML/XML
- Enough basic information on LaTeX and SGML to get going and learn more

1.2. Problems with Mainstream Technologies

- Wordprocessors are good for documents which are:
 - Short, unstructured and ephemeral
- DTP is basically the same, but usually better at:

- Complexity, quality graphics
- Things which neither WP, nor DTP, can do efficiently:
 - Handle large documents on commodity hardware
 - Convert to and from other formats
 - Re-use all or part of the data in other documents
 - Store, search and retrieve data contents
 - Multiple format publishing
 - Enforce good style (semantic, graphic or typesetting)

1.3. Confusing Style, Structure and Content

- WP and DTP are limited because they can't separate:
 - Semantic (meaningful) content
 - Logical structures
 - Rendering (graphic presentation)
- Formatting and processing instructions (mark-up) are mixed in with content, just like old-fashioned formatters troff and TeX
- It's practically impossible to edit text, style or structure independently, i.e. without changes in one type of data creating changes in the others
- Compounded by proprietorial binary file types which deny direct access to the data sources themselves
- N.B. Nothing to do with wysiwyg. HTML editors demonstrate wysiwyg editing of marked-up text sources

1.4. Generic Mark-up Solutions

- Serious electronic document preparation and management must use some form of generic mark-up:
 1. Mark up the structural position of every element in the document (e.g. title, paragraph, bullet-point, graphic, multi-media object, etc)
 2. Attach styles to these structures, not to the content itself
- Two partial solutions from the mid-1980s:
 1. LaTeX – mark-up structures in the text, then use styling instructions (in .sty files) to format the structures using TeX
 2. SGML – design a set of rules for validating any generic mark-up language
- Both LaTeX and SGML are:
 1. Considerably more efficient than WP/DTP
 2. Available, tried-and-tested (unlike XML/XSL)
- Both LaTeX and SGML have their limits:
 1. LaTeX is not fully generic
 2. SGML has proved too open and, therefore, too complex for some

Chapter 2. A Brief Overview of LaTeX

2.1. The Structure of a LaTeX Document

- Two fundamental document structures:
 1. Preamble — defines and customizes a basic document class
 2. Document — contains the substantive content of the document and its mark-up instructions (like "body" in HTML)
- The first line of the preamble (and, hence, of the document) defines the document class and any options which qualify that class, e.g.
 - Classes — book, report, article, foil, letter, memo, etc
 - Options — papersize, language, layout, etc
- The document class is defined by a file (classname.cls) in the TeX Directory Structure (TDS)
- Subsequent preamble lines specify:
 - Style packages to use (stylename.sty)
 - New (user-defined) commands
 - Modifications to commands previously defined in .cls or .sty files
 - Document-wide values, e.g. authorship, attributions, margin sizes, etc

2.2. Marking-up Document Content With LaTeX

- Commands

```
\commandname[options]{arguments}
```

- Environments

```
\begin[options]{commandname}
```

Textual content effected by the environment mark-up

```
\begin{commandname}
```

- Modes

- Math

- Paragraph

- Line

- Structural elements (depending on class), e.g.

- Sections

- Lists

- Counters

- Tables

- External objects (pics, sounds, movies, docs, etc)

- Cross-references

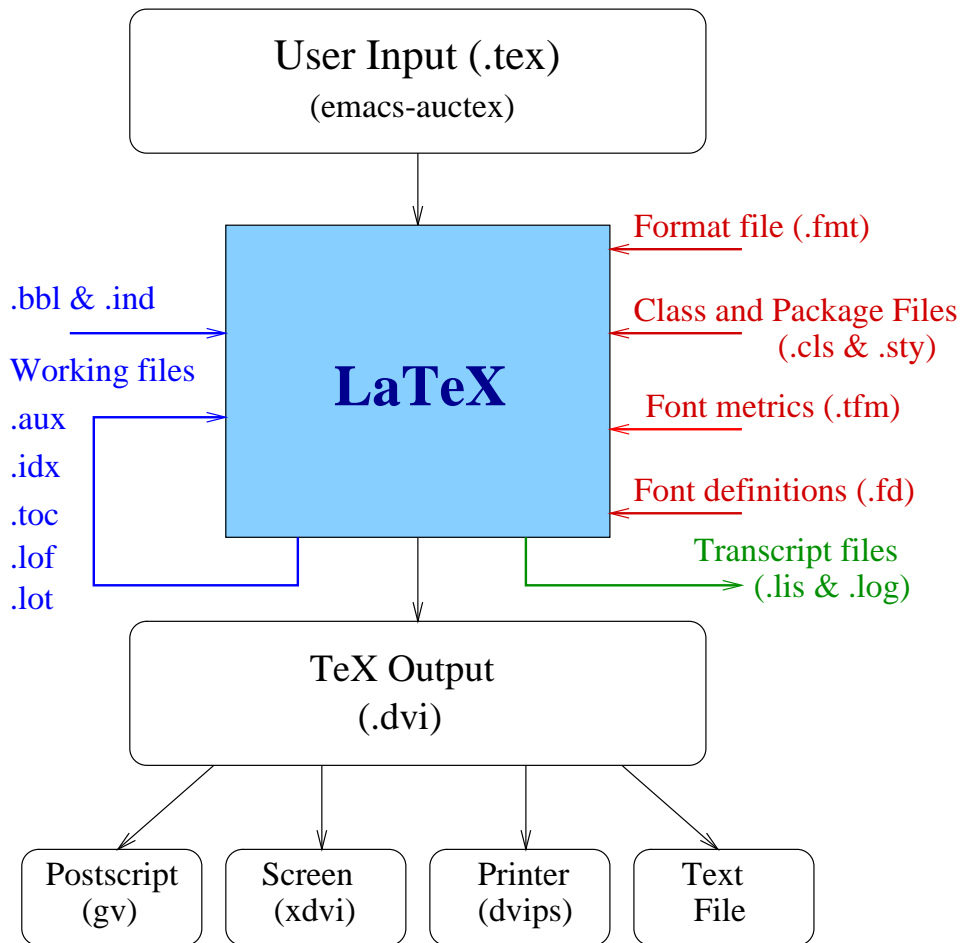
- Indices

- Bibliographies

- Headers, footers, etc

2.3. Processing LaTeX Sources

Figure 2-1. Processing Marked-up LaTeX Sources



- For simple files:

```
$ latex filename.tex
```

- Complex files (containing tables of contents/figures/tables, citations, bibliographies, cross-references, etc) need to be processed twice to update then include the indexing information kept in separate files

```
$ latex filename.tex; latex filename.tex
```

2.4. A Trivial Example of LaTeX Mark-up

```
\documentclass[12pt]{article}
\usepackage{memo}
\begin{document}
\begin{Front}
\To{Mike, Lee, Julie}
\Subject{My WYLUG Talk}
\Date{\today}
\Time{12.30}
\end{Front}
\begin{Body}
```

If anybody needs to talk to me today, please give them my home phone number.

I shall be at home all day, preparing the foils for my WYLUG talk on \LaTeX{} and SGML.

The meeting is at 19.00. If you fancy going, why not give me a ring?

```
\end{Body}
\begin{Back}
  \Signature{Dave}
\end{Back}
\end{document}
```

2.5. Device Independent Output from LaTeX

Figure 2-2. Processing Marked-up LaTeX Sources

MEMORANDUM

To whom: Mike, Lee, Julie
Subject: My WYLUG Talk
Date: September 13, 1999
Time: 12.30

If anybody needs to talk to me today, please give them my home phone number.

I shall be at home all day, preparing the foils for my WYLUG talk on \LaTeX and SGML.

The meeting is at 19.00. If you fancy going, why not give me a ring?

Dave

2.6. Problems With LaTeX

- Allows non-generic mark-up to be included (e.g. plain TeX)
- Open-ended declarations, i.e. no indication of where an element of text should end

- End-marker (the closing brace) is not explicitly linked to the specific command which opened it
- The need to test for context in order to identify linked pairs of braces makes scripted transformations or conversions difficult
- Closely tied to ink and paper metaphors
- Almost all conversions lose significant presentational features
- Marking-up can be tiresome without auctex

Chapter 3. LyX

3.1. LyX — A Half-way House

- Two versions
 - LyX (www.lyx.org)
 - KLyX (www.kde.org)
- Benefits:
 - Can help bridge the conceptual gap between wordprocessing and mark-up
 - Pretty good documentation (by Linux standards)
 - Point-and-click
 - WYSIWYG-style rendering of basic text
 - Tricky stuff (notes, referencing, etc) is marked by icons
 - Enforces good practice on style and structure
 - Outputs beautiful documents

3.2. Problems With LyX

- Dodgy install scripts
- Dodgy xforms libraries
- Non-standard LaTeX

Chapter 3. LyX

- Limited range of classes and styles
- Pampers to the point-and-click mentality

Chapter 4. SGML/XML

4.1. The Jade-DocBook Text Processing Suite

- In 1997-8 the Linux Documentation Project (LDP) endorsed SGML standard changed:
 - Name — From Linuxdoc to Sgmltools
 - Document Type Definition (DTD) — From Qwertz.dtd to DocBook.dtd
- In Spring 1999 the Sgmltools project was suspended for want of a maintainer, but:
 - Its key components are independent and remain available
 - Gnome documentation project members maintain packaged versions of these components for different Linux distributions
- RPMs are available from Mark Galassi (of Cygnus) at <http://sourceware.cygnus.com/pub/docbook-tools>

Table 4-1. The Jade-DocBook Suite

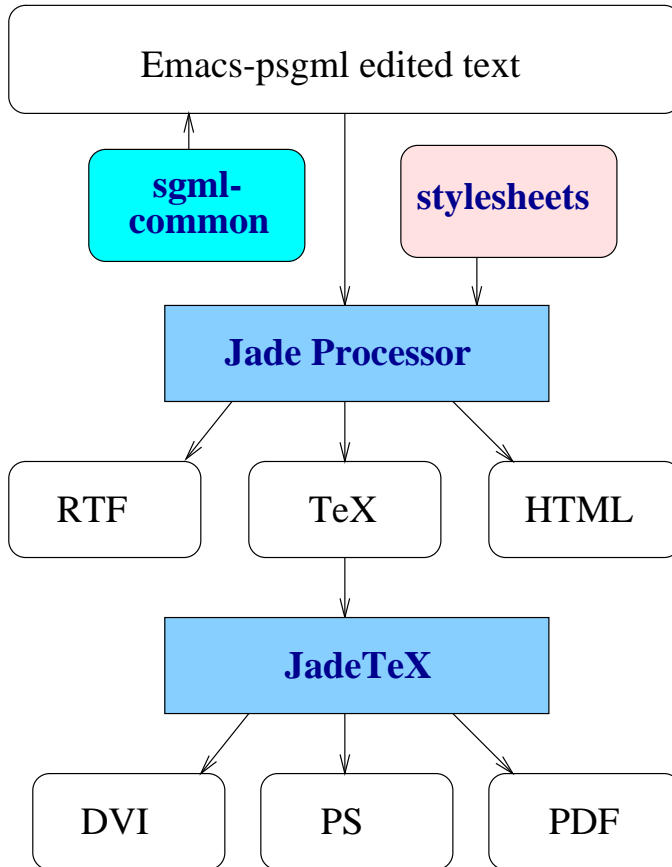
Name	Description
Jade	SGML/DSSSL processor and parser
Sgml-common	Character set definitions used by psgml
Stylesheets	DSSSL modular stylesheets used by Jade for rendering HTML, RTF, and TeX
Jadetex	Back-end which uses LaTeX to transform the TeX output from Jade into .dvi, postscript and .pdf formats

Name	Description
DocBook	Extremely feature-rich SGML DTD
Psgml	A plug-in which turns emacs into an fully-featured SGML/XML editor

- Debian users can:
 - Install these RPMs using alien
 - Install from Galassi's sources
 - Find the Debian maintainer

4.2. Processing DocBook SGML Sources

Figure 4-1. Transforming a DocBook SGML File into other formats



4.3. Key DocBook Tools Commands

- The basic DocBook tools commands are as simple and as obvious as possible:

```
$ db2html filename.sgml
```

Outputs the document as a complete html website

```
$ db2rtf filename.sgml
```

Outputs the document in Rich Text Format (RTF)

```
$ db2dvi filename.sgml
```

Outputs the document as device independent file (.dvi)

```
$ db2ps filename.sgml
```

Outputs the document in postscript format (.ps)

```
$ db2pdf filename.sgml
```

Outputs the document in portable document format (.pdf)

- Default appearance is determined by Norm Walsh's concerns and tastes
- Greater control and complexity is possible:
 - Modify or swap style sheets
 - Define specialist mark-up languages (DTDs)
 - Define LaTeX classes for customised output
 - Convert between different SGML and/or XML mark-up languages
 - Integrate into web-site or DB building scripts

4.4. Example: A Trivial DocBook Document

Example 4-1. A Trivial Document in DocBook SGML

```

<!doctype book PUBLIC "-//Davenport//DTD DocBook V3.0//EN" []>
<book>
  <bookinfo>
    <date>1999-09-13</date>
    <title>The Title of the Book</title>
    <subtitle>The Book's Subtitle</subtitle>
  </bookinfo>
  <toc></toc>
  <!-- The front matter ends here and we now begin the body
of the document -->
  <chapter>
    <title>Title of the first chapter</title>
    <para>In a book, it is stylistically poor to put a section
title immediately after the chapter title. I did so with these
foils, because they are not strictly part of a book.</para>
    <sect1>
      <title>First section in the first chapter</title>
      <para>I need a single paragraph of text here to make the
section valid.</para>
    </sect1>
  </chapter>
  <appendix>
    <title>Title of an Appendix</title>
    <para>The appendix is a handy place to put the stuff you
couldn't fit in elsewhere or merely supports the main text.
</para>
    <sect1>
      <title>A Section Title in an appendix</title>
      <para>This is an almost meaningless paragraph of text
in the appendix which I use to illustrate the use of the
<emphasis>emphasis</emphasis>tag.</para>
    </sect1>

```

```
</appendix>  
</book>
```

4.5. Brief explanation

- The first line can declare:
 - The type of document (its a book)
 - Document status (public vs private)
 - Document type definition and Version (DocBook.DTD 3.0)
 - Language used in the tags (automatically translatable)
 - A URL for the DTD (not used here)
- The bookinfo section defines document-wide values, like a LaTeX preamble
- Jade builds and inserts tables of contents, figures, etc. at the "toc" tag
- Stylesheets will only mark a new paragraph properly (e.g. linespacing, indenting, etc) if it is signalled by the "para" tag.
- Every section (e.g. chapter, appendix, sect1) should have a title element within it
- Documents are only valid if their DTD allows or requires elements in the positions they occur, e.g. elements may be invalid if empty or if they contain 'foreign' sub-elements.
- All the formatting of marked-up elements is done automatically by Jade, following instructions in the stylesheets

4.6. Editing With Psgml

Psgml can help improve the quantity and quality of SGML writing in hundreds of ways, these are just a few of them:

- Invoke an external parser to validate an SGML document (nsgmls by default)
- Use parsed DTD data to only offer valid tags from drop-down menus, pop-up menus and command completion
- Show all the valid tags at any position in the document as SGML comments
- Navigate DTD structures, not just textual elements (between tags, up and down nested tags, etc)
- Handle child-parent relationships between documents
- Indent all or part of the document according to structure
- Show and edit the available attributes for any element
- Show/hide and/or Fold/unfold mark-up tags for all or part of the document (e.g. edit in 'outline mode')
- Highlight mark-up using multiple typefaces and colours (font-locking)
- Tag selected 'regions'

4.7. Problems with the Jade-DocBook Tools Suite

Because these tools are designed to handle almost every conceivable type of SGML document and software/hardware platform, they can never be installed and configured by one press of a button. These are some of the problems many people will encounter:

- Some versions of sgmtools have problems with install script portability (the last version seems ok)

- RPM packages have to be installed in a precise order and are built against specific glibc libraries (update your libraries or find some old RPMs)
- Every version I've used barfs on TeX-dependent output. Usually, because the TeX config file (`/usr/share/texmf/web2c/texmf.cnf` in teTeX) does not provide enough resources (add a zero to every value which produces an error message)
- You want to produce more than just the DocBook books and article types (design a DTD or join a team)
- Norm Walsh's stylesheets don't suit your needs (modify them after backing up the originals)
- DSSSL stylesheets are very powerful, but they are complex and depend on scheme programming (CSS1, CSS2, and XSL may be substituted)
- Emacs is the only free editor with full support for SGML editing (not for long, WYSIWYG is on the way via abiword, etc)
- You've installed customised DTDs and stylesheets, but you can't get Jade to process them. This is, usually, because they are not listed properly in the right "catalog" (sic) file (need to research this one).

4.8. SGML to XML/XSL

- SGML only widely implemented by giant corporations, until HTML
- SGML made the modern Internet possible
- HTML also demonstrated weaknesses of SGML
 - Excessive complexity (too many options)
 - Excessive openness (to abuse/extension)
- XML docs are just valid SGML docs, written to a more tightly defined set of rules

- XML covers blushes of those who rejected open-standard docs before, but includes useful restrictions:
 - Should be viewable over Internet
 - Should support wide variety of apps
 - Optional features should be minimal, ideally zero
 - Docs should be human-readable and clear
 - XML specs should be formal and precise, hence easy to process automatically
 - Valid docs should be easy to write
 - Terse mark-up is *not* important, clarity always beats conciseness

4.9. Conclusions and Prospects

- XML specs yet to be proven, XSL still unpublished
- LaTeX works now for paper-orientated docs
- Jade-Docbook tools proven in technical and commercial publishing (Sun, O'Reilly, etc)
- Sgmltools could provide a viable basis for general-purpose document preparation and management system, but needs:
 - Really good introductory texts on how to use the DocBook, and other, DTDs
 - Install script options to reconfigure TeX resources
 - Companion DTDs for Foils, Reports, Letters, Memos, etc which can be styled by the same collection of modular stylesheets, according to context
 - Simple configuration tool to enable all DTDs to use the same stylesheets
 - Ability to use a simple stylesheet language (like CSS1, but ideally XSL)

Chapter 4. SGML/XML

- WYSIWYG editing with as much, or more, functionality as psgml
- One single RPM/deb package to simplify installation?
- A WYLUG development project?
- A reading group for Norm Walsh's definitive guide to DocBook?

