XLingPaper's use of TEX Technologies

H. Andrew Black, Hugh J. Paterson III

Abstract

We discuss the use of TEX technologies by XLingPaper, an authoring tool for producing academically oriented publications with features required for linguistic publishing. We present the TEX modules used and the rationale for the history of its development.

1 Introduction

Within the publishing industry there are several notable products for producing complex documents in beautiful formats. TEX [11] [12] is one of the well known publishing technologies used to meet these needs. Since 2000, XML-based technologies such as XSL-FO[1] or the TEXML project [14][2] have also seen use to integrate content and compose complex documents such as textbooks and maintenance manuals. Requirements for composing these large, interlinked documents birthed the development of tools like XMLmind[3] and Xpublisher.[4] These can be used to compose the content within predefined XML structures. XLingPaper [3] [4] [5] seeks to provide a constrained environment in which authors of complex works dealing with language descriptions and linguistic analyses can focus on content structure independently from the styling requirements of publishers. The software has a growing number of users who have successfully typeset complex documents including:

- master theses [20] [13] [16],
- doctoral dissertations [9] [17],
- textbooks [15],
- linguistic grammars [7],
- journal articles [6], and
- bilingual software documentation [1] [2].

XLingpaper[5] is a plug-in to the XMLmind XML Editor. XLingPaper benefits from the XMLmind XML Editor's Java-based implementation which allows it to be used on MacOS, Windows, and Linux. XLingPaper, via a DTD, defines several document classes (articles, books, chapters, etc., as illustrated in Figure 1), in each case providing document layout sections (paragraphs, examples, endnotes, etc.). By working within the user-interface of the XMLmind XML Editor, formatting errors are reduced because users are constrained on where in the document flow they can introduce block and line level document elements. That is, first, authors cannot input XELATEX code directly into the document and second, the introduction of layout sections within the document flow is constrained via the DTD.

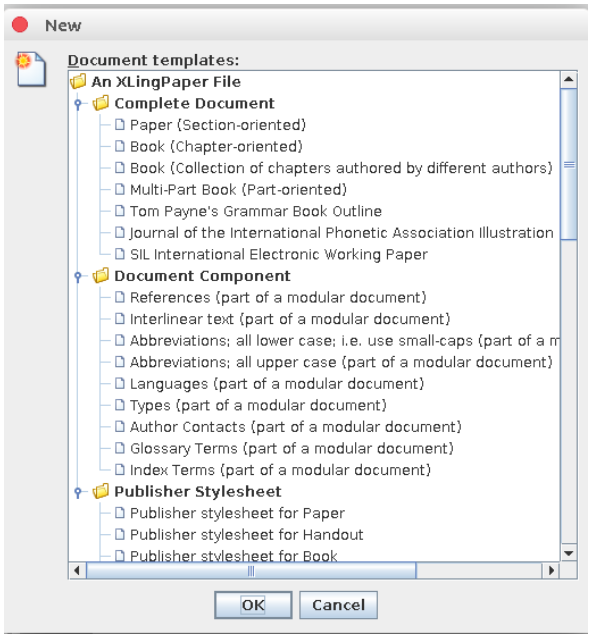

Figure 1: XLingPaper predefined document types via DTD

2 What is XLingPaper?

As mentioned above, XLingPaper is an XML- and Java-based computer plug-in for the XMLmind XML Editor. It is designed to make writing, reading, and publishing linguistic papers, grammars, and books better and more consistent. A full list of benefits to all parties in the publishing work flow is available [5]. XLingPaper can produce linguistic documents with at least five outputs, all from the same source document:

- PDF (version 1.5),
- Web pages (HTML 4),
- Microsoft Word (.doc),
- Open Office Writer Document (.odt), and
- ePUB.

It automatically numbers sections, figures, tables, and examples. It keeps track of internal references to these entities along with citation references and

[1] https://www.w3.org/TR/xsl11
[2] http://getfo.org/texml
[3] https://www.xmlmind.com/xmleditor
[4] https://www.xpublisher.com/products
[5] https://software.sil.org/xlingpaper

gloss abbreviations. This keeps numbering and reference links dependable and automated. It also automatically generates indexes, a table or list of abbreviations used, and a section for references cited (using a custom references implementation).

Unlike most editing programs which are based on either the WYSIWYG paradigm or as text editors used to code or produce Markdown, XLingPaper (via the XMLmind XML Editor) is a structured editor. Rather than visually structuring the document to look the way it is to be formatted, the author "marks up" the items in the document according to their kind. One of the many benefits that using a DTD provides is that there is a "grammar" of what a well-formed linguistic document looks like. This makes moving, replacing, switching, or reordering sections, chapters, and examples less error prone because it prevents users from inadvertently creating ill-formed documents.

The following sections of this paper discuss the TEX technologies used.

3 XLingPaper and TEX

Due to the heavy reliance on Unicode in modern language documentation and linguistic work, XLingPaper specifically uses XELATEX and compatible packages to produce PDF outputs. The following sections provide more detail on the design requirements and packages used.

3.1 Design desiderata for TEX with XLingPaper

From the outset, XLingPaper was designed to be free. The XMLmind XML Editor had a Personal Use License that met this requirement for the vast majority of the target audience of XLingPaper. The few who did not meet the terms of that license most likely would be able to afford to purchase (or have their organization purchase) a professional version of the XMLmind XML Editor. The actual XLingPaper plug-in has always been free.

XLingPaper development started in 2001. In 2006 it adopted XSL-FO for PDF production. Prior to 2009 XLingPaper used RenderX[6] to produce PDF documents. However, in 2009 plans were made to add XELATEX-based output to XLingPaper because, while there was a free version of RenderX, the output contained a watermark. By implementing the ability to export to PDF via XELATEX, there would be no water marks in PDF documents. The XELATEX method of PDF production is now the default method to produce PDF documents.

[6] http://www.renderx.com

When the XELATEX method of PDF production was introduced, XLingPaper had a way to format output per a user-created publisher style sheet. This meant the developer (Andrew Black) needed to be able to map from an XLingPaper publisher style sheet to XELATEX. Mapping style sheet information contained in the XML was the second criterion. It was known that LATEX was the ideal TEX implementation to target. However, pure LATEX came with predefined output formatting for front matter, chapters, sections, back matter, etc. Pure LATEX, then, would not allow direct control of formatting of all of these per an XLingPaper user-defined publisher style sheet. This meant overriding these standard features of LATEX with a custom implementation of the TEX commands needed to control formatting. However, `memoir` [18] [19], a recent discovery for the programmer of XLingPaper, accomplishes many of the same tasks and could be considered to replace some of the custom code if it were shown to be easy to implement and that the size of the total code base would be reduced.

The third criterion concerned some of the target audience for XLingPaper. Many of the expected users of XLingPaper live and work in places around the world where Internet connections are characterized by high costs, low bandwidth capacity, and general unavailability. Therefore, the download required to install XLingPaper needed to be as small as possible. The size constraint impacts XLingPaper because its distribution must be independent of larger mainstream TEX distribution solutions such as TEXLive which have a large footprint. Therefore the developer determined which LATEX packages and binaries were needed and created a custom installation package for just those items (a list of these packages is included in section 6.1).

The twenty LATEX packages that are part of the custom XLingPaper distribution are still rather large for someone for whom Internet bandwidth is an expensive and inconsistent commodity. To reduce bandwidth requirements two assumptions were made which have more or less proven to obtain. The first assumption that the developer made was that the twenty packages and binaries would not need to change over time; in contrast, the second assumption was that XLingPaper would acquire new features and need bug fixes. These assumptions resulted in an architecture where page layout information expressed in XML is translated via custom TEX commands to either TEX directly or to commands understood by LATEX packages distributed with XLingPaper. This abstraction layer was then executed when the XELATEX file was processed. This

middle layer has granted XLingPaper flexibility in adding new code and capabilities while keeping the "heavy" LaTeX packages stable. The net result is a "heavy" first install package, but a light-weight upgrade package. In the thirteen year history of development, there have been occasions where a new "heavy" package has been required. One such case was when framed units where added as an element. These elements depend on the `mdframed` package [8][7]. However, the split architecure has generally worked out well and kept update sizes low.

3.2 PDF production

When an author has XLingPaper produce PDF output via XeLaTeX, XLingPaper produces a TEXML-like XML file. This is then converted into LaTeX format via a set of XSLT transforms and given to XeLaTeX which produces the PDF. Figure 2 contains a diagram of this process.

3.3 TEXML

When Andy began implementing the XeLaTeX-based output, he found TEXML, but understood it to have two infelicities:

1. TEXML required Python and he did not want to force XLingPaper users to have to install a version of Python for TEXML when that version may conflict with other versions of Python they might already have installed. Furthermore, this approach would make the installation package much larger because of needing to include Python.
2. He also needed some extensions for formatting white space (more finely as far as he could tell).

He did implement some Java code to deal with mapping certain characters used in TEX commands to their TEX equivalents. He used Java because the XMLmind XML Editor is written in Java and XLingPaper already used Java code to improve the user experience in the XMLmind XML Editor.

3.4 Ling-TeX

At the time Andy began implementing the XeLaTeX-based output, he discovered the Ling-TeX group. From what he could tell, the packages that help with interlinear texts did not allow for the larger number of capabilities XLingPaper already dealt with. So he rolled his own. Figure 3 contains an example output with some of the special capabilities XLingPaper offers.

[7] `https://ctan.org/pkg/mdframed`

4 Typesetting tasks XLingPaper users often encounter

Linguistic documents have several formatting needs that other kinds of documents do not. This section discusses some of them.

4.1 Numbered example layouts

Linguistic documents usually have many numbered examples. The prose often refers to examples near the material or to previous examples. XLingPaper automatically keeps track of the numbers. Besides table-like layouts, linguists also need lists of words along with their glosses (as shown in Figure 4), interlinear clauses (as shown in Figure 3), and even having headings in portions of the example.

4.2 Automatically wrapping interlinear texts

Many linguists want to include interlinear glossed text in their document. XLingPaper allows these to be wrapped automatically which makes the author's job much easier. Figure 5 shows one such text portion.

4.3 Gloss abbreviations

Linguists standardly use glosses for indicating the meaning of pieces of words (morphemes). XLingPaper allows the author to define a set of abbreviations and their definitions. When producing the output, XLingPaper creates hyperlinks between the abbreviation and its definition.

5 Outputs LaTeX allow that others do not

While XLingPaper has a large array of linguistically-oriented formatting capabilities, there are some that only the XeLaTeX output can produce. This is, of course, due to the formatting power of TEX and XeLaTeX.

5.1 Automatically wrapping interlinears

One of the most popular features of XLingPaper is its ability to automatically wrap long interlinear examples and lines in interlinear texts. It does so by formatting each aligned word in an hbox and then having XeLaTeX put them together in a hanging indent paragraph. This is based on the work of Kew & McConnel 1990 [10].

5.2 Font rendering

XeLaTeX renders fonts extremely well. It can even handle special features requiring Graphite[8] processing. For other outputs, some fonts (such as Charis

[8] `graphite.sil.org`

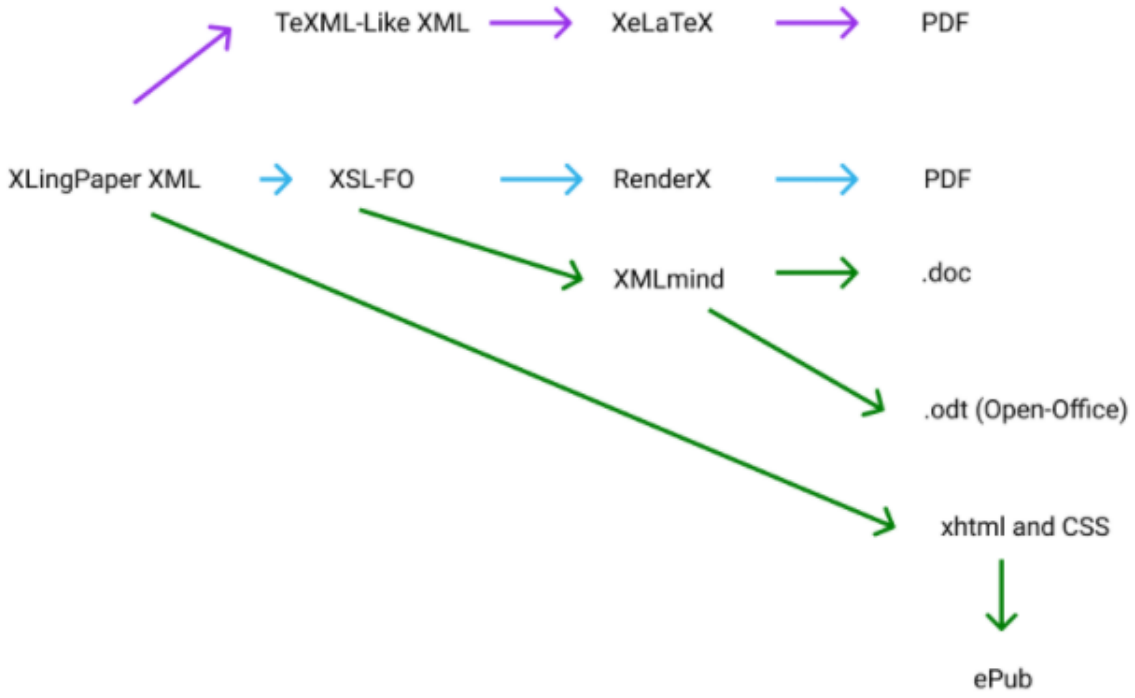

Figure 2: XLingPaper output process

Una frase cuantificadora puede acompañar al sustantivo (véanse Los Cuantificadores y Los Números Cardinales). Cuando se presenta esta frase, siempre va delante del núcleo de la frase nominal, como en los ejemplos en (2).

(2) a. [tcf-Zila]
Náa majñuu nákhu iduu iya'
náā māhjūù^{n} nákù īdūū ījā?
LOC entre TOT.cuatro ojo.3SG agua
'De entre los cuatro manantiales[5]' [Smajiin:6]

b. [tpl-Tlac]
Gí'doo witsu rakhóó mikhúdú
EST.tener.3SG cinco nariz.3SG (EST).picud@
'Tiene cinco esquinas picudas' [FC:5.1]

El cuantificador puede presentarse en construcciones donde no hay sustantivo expreso, como se explica en Los Cuantificadores. Un ejemplo se incluye aquí.

Figure 3: Interlinear example

Bantu D30 canonical infinitive verb pattern is exemplified in the Mbo data in (11):

(11)
a. [ex[ko-sis-o]ex] [ex[[– ¯ ¯]]ex] move forward
b. [ex[kɔ-kɨj-ɑ]ex] [ex[[– ¯ ¯]]ex] act
c. [ex[ko-ɓund-o]ex] [ex[[– ¯ ¯]]ex] break
d. [ex[kɔ-ɓʉt-ɑ]ex] [ex[[– ¯ ¯]]ex] become long
e. [ex[ko-ɓeŋ-o]ex] [ex[[– ¯ ¯]]ex] wink
f. [ex[kɔ-kɛk-ɑ]ex] [ex[[– ¯ ¯]]ex] decorate
g. [ex[ko-sok-o]ex] [ex[[– ¯ ¯]]ex] cackle
h. [ex[kɔ-mvɔɗ-ɑ]ex] [ex[[– ¯ ¯]]ex] suck
i. [ex[kɔ-bɑb-ɑ]ex] [ex[[¯ – ⁄]]ex] carry

Figure 4: List of words

Rikha[2]

FC:1

Rikha	rígi'	najmąą	náą	yúoo'	rą'khą ká',[3]	rą'khą suan'[4]
flor.de.calabaza	INAN:PROX	IMPF.producirse	LOC	guía.3SG	calabaza.especie	calabaza.especie

khamí	náą	yúoo'	rą'khą' májin'.[5]
y	LOC	guía.3SG	chilacayote

'La flor de calabaza se da en la guía de la calabaza de Castilla, de la "calabaza espina" y del chilacayote.'

FC:2

Rí	rikhoo	rą'khą suan',	nagí'dụụ	namịdị	rí
SBD:INAN	flor.de.calabaza.3SG	calabaza.especie	IMPF.empezar.3SG.FM ±	IMPF.florear	SBD:INAN

gụn'	agóstó.
luna	agosto*

'La flor de la "calabaza espina" empieza a abrir en el mes de agosto.'

FC:3

Mbą'jụ,	mụjmụ'	rị'jiụụ.
(EST).grande:PL	(EST).amarill@	flor.3SG

'Sus flores son grandes y amarillas.'

Figure 5: Wrapped interlinear text

SIL) may not line up vertically as expected due to them having different ascender and descender values. One has to add custom commands to deal with these. In the case of Graphite, they may not be able to be done at all. The RenderX way of producing PDF cannot handle stacked diacritics, but the XeLaTeX way does it very well.

5.3 Hyphenation for non-English languages

Since we use the `polyglossia` package, one can write an XLingPaper document in a non-English language and XeLaTeX will hyphenate according to that language's hyphenation rules.

5.4 Author contact information

XLingPaper allows one to define a set of contact information for authors. Only the XeLaTeX output is able to format them correctly.

5.5 Vertical fill

For title page material, only the XeLaTeX output allows using vertical fill between items. The other outputs require using overt, fixed spacing values.

5.6 Blank page

When one wants a totally blank even-numbered page between a final odd-numbered page and the next odd-numbered page which begins, say, a chapter or appendix, only the XeLaTeX approach is able to do this.

6 Features other outputs have that the LaTeX output does not

XeLaTeX does not allow for custom table cell padding and spacing. Having said that, Andy cannot remember any XLingPaper user ever asking for a way to do this for the XeLaTeX output. It just looks great.

Background color is not available for section titles.

Section 11.17.1.1 "Known limitations of using XeLaTeX" in the XLingPaper user documentation lists known problems.

6.1 List of LaTeX packages used

XLingPaper currently uses the following XeLaTeX packages (in alphabetical order):

`attachfile2`	`lineno`
`booktabs`	`longtable`
`calc`	`lscape`
`color`	`mdframed`
`colortbl`	`multirow`
`etoolbox`	`normalem`
`fancyhdr`	`polyglossia`
`fontspec`	`setspace`
`footmisc`	`tabularx`
`hyperref`	`xltxtra`

6.2 Custom TeX commands

XLingPaper has a number of custom commands that enable it to handle various tasks in a way that is consistent with our desired outcomes. The following lists some of them in a schematic way:

Command for	**Purpose**
Table of contents	Store and retrieve page numbers; format the contents.
Lists	Numbered and bulleted lists with control over indents, etc.
Examples	Example number and example content, where the content can be a line, a list of lines, a set of words, a list of a set of words, interlinear, a list of interlinears, etc.
Indexes	Handle keeping track of XLingPaper's indexing capability, including page numbers.
Interlinears	Handle lines in an interlinear text or example, including dealing with an ISO 639-3 code in an interlinear example.
Block quotes	Handle special cases needed for block quotes.
Table headers	Attempt to calculate a column's width via its contents.

7 Conclusion

While the XLingPaper approach to writing linguistic documents has great value in and of itself, the fact that it can produce great looking output via XeLaTeX makes it very worthwhile learning to use. We feel that being able to produce PDF via XeLaTeX has made XLingPaper a fantastic tool for linguists.

References

[1] Beadle, Jennie, and Matthew Lee. 2020a. *Paratext 9 Manual – in English*. SIL International. https://lingtran.net.

[2] Beadle, Jennie, and Matthew Lee. 2020b. *Paratext 9 Manual – in French*. SIL International. https://outilingua.net.

[3] Black, Cheryl A., and H. Andrew Black. 2012. *Grammars for the People, by the People, Made Easier Using PAWS and XlingPaper*. In Electronic Grammaticography, edited by Sebastian Nordoff, 103–28. LD&C Special Publication 4. Honolulu, Hawai i: University of Hawai i Press. http://hdl.handle.net/10125/4532.

[4] Black, H. Andrew. 2009. Writing Linguistic Papers in the Third Wave. *SIL Forum for Language Fieldwork* 2009 (004): 11 pages. https://www.sil.org/resources/publications/entry/7790.

[5] Black, H Andrew. 2017. *Why Learn to Use XLingPaper*. Dallas, Texas: SIL International. http://software.sil.org/downloads/r/xlingpaper/resources/documentation/WhyUseXLingPaper.pdf.

[6] Brownie, John. 2013. Adverbs in the Mussau-Emira Verb Phrase. *Language & Linguistics in Melanesia* 31(1): 1–11. https://www.langlxmelanesia.com/LLM%20Vol.%2031%20Adverbs%20Mussau.pdf.

[7] Buck, Marjorie J. 2018. *Gramática del amuzgo Xochistlahuaca, Guerrero*. (Serie de gramáticas de lenguas indígenas de México No16.) Tlalpan, Ciudad de México, México: Instituto Lingüístico de Verano, A.C. [SIL International in Mexico]. https://www.sil.org/resources/archives/75518.

[8] Daniel, Marco, and Elke Schubert. 2013. *The `mdframed` Package: Auto-split Frame environment* version 1.9b.

[9] Ebarb, Kristopher J. 2014. *Tone and variation in Idakho and other Luhya varieties*. University of Indiana Ph.D. dissertation. https://pqdtopen.proquest.com/doc/1625743679.html?FMT=ABS.

[10] Kew, Jonathan and Stephen McConnel. 1990. *Formatting Interlinear Text*. Occasional Publications in Academic Computing, Number 17. Dallas, Texas: Summer Institute of Linguistics.

[11] Knuth, Donald Ervin. 1984. *The TEXbook*. A. Computers & typesetting. Reading, Massachusetts: American Mathematical Society; Addison-Wesley.

[12] Knuth, Donald Ervin. 1986. *TEX: The Program*. B. Computers & typesetting. Reading, Massachusetts: Addison-Wesley.

[13] Lamicela, Andrew Charles. 2020. *Distinguishing Passive from MP2-marked Middle in Koine Greek*. University of North Dakota M.A. thesis. https://commons.und.edu/theses/3277.

[14] Lovell, Douglas. 1999. TEXML: Typesetting XML with TEX. *TUGboat*. 20 (3): 176–183. https://tug.org/TUGboat/tb20-3/tb64love.pdf.

[15] Marlett, Stephen A. 2019. *Phonology From the Ground Up: The Basics*. Dallas, Texas: SIL International. https://www.sil.org/resources/archives/79207.

[16] Paterson III, Hugh J. 2021. *Language Archive Records: Interoperability of Referencing Practices and Metadata Models*. University of North Dakota M.A. thesis. https://commons.und.edu/theses/3937.

[17] Rasmussen, Kent. 2018. *A Comparative Tone Analysis of Several Bantu D30 Languages (DR Congo)*. University of Texas Arlington Ph.D. dissertation. http://hdl.handle.net/10106/27483.

[18] Wilson, Peter. 2007. The Memoir Class. *TUGboat* 28 (2): 243–46. https://www.tug.org/TUGboat/tb28-2/tb89wilson.pdf.

[19] Wilson, Peter. 2021. *The Memoir Class for Configurable Typesetting: User Guide*. version 3.70. Normandy Park, WA: The Herries Press. https://texdoc.org/serve/memoir/0.

[20] Wood, Joyce Kathleen. 2012. *Valence-Increasing Strategies in Urim Syntax*. Graduate Institute of Applied Linguistics M.A. thesis. https://www.diu.edu/documents/theses/Wood_Joyce-thesis.pdf.

◇ H. Andrew Black
blackhandrew (at) gmail dot com

◇ Hugh J. Paterson III
i (at) hp3 dot me
http://hp3.me