

# **Writing Linguistic Papers in the Third Wave**

**H. Andrew Black**

**SIL International and University of North Dakota  
2008**

SIL Forum for Language Fieldwork 2008-YYY, November 2008  
Copyright © 2008 H. Andrew Black and SIL International  
All rights reserved

## Contents

Abstract .....	3
1 Introduction .....	3
2 Authoring vs. Formatting .....	4
3 Some Features of <b>XLingPaper</b> .....	5
4 One Source, Multiple Outputs .....	6
5 Archiving .....	7
6 Other tools that already use <b>XLingPaper</b> .....	7
7 The value of the <b>XMLmind XML Editor</b> .....	8
8 Future Plans .....	10
9 How to get the <b>XLingPaper</b> package .....	10
10 Conclusion .....	10
References .....	10

## Abstract

This article introduces a method to write linguistic articles and books that addresses most, if not all, of the frustrations authors often face when writing linguistic articles and books. This method follows a Third Wave approach as described in Simons and Black (2008). As Simons and Black note, we need to “work smarter” rather than work harder. One of the key notions is to produce “actionable” knowledge. They also note the role that XML technologies can play in enabling us to “work smarter.” In this article I explain a way we can “work smarter” when it comes to writing linguistic papers and books. The method uses several things:

- the mark-up language called XML
- an XML package designed for authoring and even archiving linguistic papers ([XLingPaper](#))
- an XML Editor program that makes it much easier to write XML documents ([XMLmind XML Editor](#)).

## 1 Introduction

Writing linguistic papers is seldom easy. Perhaps you've experienced one or more of the following issues:

- While writing a linguistic article or book, you have found yourself “fighting” with your editing program to get things to come out the way you really wanted.
- You've written an article and gotten it to look the way you wanted, only to find out that in order for it to be published, you or someone else will need to practically re-key the entire article because the publisher needs to use a different program than you used.
- You have found yourself saying something like, “I wish I could just write and not have to deal with all of these formatting issues.”
- You have tried to revise an article you wrote years ago and now find that the new version of your editing program either will not read your old document or it has lost some or all of its formatting.

If you can identify with any of these problems, you are definitely not alone.

This article introduces a method to write linguistic articles and books that addresses most, if not all, of these issues. This method follows a Third Wave approach as described in Simons and Black (2008). As Simons and Black note, we need to “work smarter” rather than work harder. One of the key notions is to produce “actionable” knowledge. They also note the role that XML technologies can play in enabling us to “work smarter.” In this article I explain a way we can “work smarter” when it comes to writing linguistic papers and books. The method uses several things:

- the mark-up language called XML (“Extensible Markup Language”)
- an XML package designed for authoring and even archiving linguistic papers ([XLingPaper](#))
- an XML Editor program that makes it much easier to write XML documents ([XMLmind XML Editor](#)).

I used these tools to write this article and we also used them to write Simons and Black (2008) and Black and Black (2008).

This article begins by highlighting a key distinction that exists between authoring and formatting in section 2. Section 3 then outlines some of the key features of the [XLingPaper](#) approach, followed

by a discussion of how to obtain formatted output in section 4. Then there are brief discussions on archiving (section 5) and other tools that already use **XLingPaper** (section 6). The key role that the **XMLmind XML Editor** plays in making all this technology be much more user-friendly is given in section 7. Since **XLingPaper** continues to be developed, section 8 outlines current development plans. Finally, section 9 explains how to obtain the **XLingPaper** package.

## 2 Authoring vs. Formatting

The first thing to consider is a distinction between authoring and formatting. WYSIWYG or “What you see is what you get” tools (like **Microsoft Word** that so many of us currently use) have the advantage of enabling us to immediately see how what we write will look on the printed page. This, however, actually forces us to do two tasks at once: both write and format. So rather than focusing on the authoring task, we find ourselves continually switching back and forth between formatting (making it come out right on the printed page) and writing (i.e. expressing our thoughts and presenting our data).

Using such WYSIWYG editors can also potentially run into two other problems. If the computer file produced by the editor uses a proprietary, binary format, then one has to have a compatible version of the editor in order to read it. If years go by after writing something and you now try to read that draft or version, you may well discover that it comes out very poorly, if at all, in the new version of the editor. Similarly, if you submit your carefully written and carefully formatted work to a publisher, they may not use the same editing program and so they may actually end up re-keying and re-formatting the whole thing. You then find yourself in the position of having to very carefully proofread what they give you (the proofs), even though what you submitted to them was exactly what you wanted.

Another issue here is that when we are writing, we are actually dealing with various conceptual entities: titles, chapters, sections, paragraphs, examples, references to examples, references to sections, interlinear text, language data, glosses, tables, charts, trees, citations to other works, endnotes (or footnotes), etc. Unless we are extremely careful and self-disciplined, when using WYSIWYG editors like **Microsoft Word**, we do not actually tag all that we write according to these entities. We may well use styles to tag certain parts, but there is really nothing forcing us to do that. As long as it comes out looking the way it should on paper, we tend to be satisfied. Furthermore, when you do go to tag a certain portion with a style, you are typically presented with the entire list of styles, even though many of them may be totally inappropriate in the context.

So how can we improve on these problems? XML-based technologies enable us not only to tag each entity with its appropriate tag, but also allow us to write a grammar of how these tags relate to each other. The latter means that given an appropriate XML-aware tool, we will always be tagging what we write and we will be able to only insert new entities that are appropriate in the context. See Simons and Black (2008) for more on this.

In this article, I introduce such an XML-based approach (**XLingPaper**) and such an XML-aware tool (**XMLmind XML Editor**). This **XLingPaper** approach is designed to free you from trying

to both write and format at the same time. It, in conjunction with the **XMLmind XML Editor**, enables you to focus solely on the authoring task. This can be very freeing.<sup>1</sup>

### 3 Some Features of **XLingPaper**

Besides aiding the author by only allowing appropriate elements in context, **XLingPaper** provides several other key features. This section and the next delineate some of the most important. See the **XLingPaper** user documentation for a full listing.

Linguists commonly face three obstacles in producing papers. First, all examples are numbered in a paper. If during the writing process the author discovers a need to insert an example, then the numbering of all following examples and all references to those examples within the text need to be re-adjusted. This mechanical change can be both time-consuming and prone to error. Similarly, if the author decides to reorder some examples, then the numbering needs to be adjusted appropriately. **XLingPaper** provides an automatic way to facilitate such numbering and renumbering. You never actually number the examples. Rather, you give a unique, abbreviated description for each example (usually something appropriate to the content or purpose of the example). Whenever you make a reference to that example (from anywhere in the paper or book), you make use of this unique description. **XLingPaper** will also produce a hyperlink to the example at the place of the reference.

Secondly, linguists cite the work of other researchers using a standard citation format. This format functions essentially as an abbreviation or reference to the full citation entry which appears in the references section of the paper. The burden of maintaining consistency between citation and reference typically falls totally on the author. Many a reader has been disappointed to find a citation to a paper in the body of a paper for which there is no entry in the references section. **XLingPaper** provides an automatic means for a writer to maintain consistency; all citations in the text must have a corresponding entry in the references. Conversely, **XLingPaper** will include only those entries in the references section which are cited in the text. This latter characteristic implies that one can maintain one master list of references and merely include it in any given paper. Only those references actually cited in the given paper will appear in the references section.

Thirdly, linguists commonly use a set of abbreviations while glossing examples. They usually include either a list of the abbreviations and their definitions in a footnote, in a special front-matter page, or in a back-matter page. As for citations and references, the burden of maintaining consistency between the abbreviations used in the text and the abbreviations defined in the list typically falls totally on the author. Therefore, it is not unusual to discover that an abbreviation in a gloss has no corresponding entry in the list of abbreviations. Just like it does for citations, **XLingPaper** provides an automatic means for a writer to maintain consistency; the author can make it so all abbreviations in the text must have a corresponding entry in the list of abbreviations. Conversely, **XLingPaper** will include only those abbreviations in the list of abbreviations which are actually used in the text. Like for references, this implies that one can maintain one master list of abbreviations and merely include it in any given paper. Only those abbreviations actually cited in the given paper will appear in the list of abbreviations. By the way, **XLingPaper** also creates a hyperlink between the

---

<sup>1</sup>In all honesty, I must admit that there are occasionally times while writing with the **XLingPaper** package that I do think about formatting. These tend to be rare and are often occur while setting up the special language or type elements.

abbreviation in the text and the abbreviation in the list of abbreviations. Thus, a reader can click on the abbreviation and see what it means and then return to where they were reading.

In addition, in some formatting systems such as HTML, header elements are not automatically numbered. A web page produced via **XLingPaper**, however, will automatically number all the header elements, including parts, chapters, and sections. In addition, one may create references to parts, chapters, or sections and these will display as hyperlinks to them, using the appropriate number.<sup>2</sup>

## 4 One Source, Multiple Outputs

The key notion to keep in mind, however, when using **XLingPaper** is that we are not trying to format pages as we write. That would be a Second Wave approach. Rather, we are using a Third Wave approach where we create “actionable data” which can then be formatted in various ways, including being printed on paper.

The **XLingPaper** package comes with two sets of auxiliary files that produce either a web page or something that can be made into a PDF file.<sup>3</sup> When you use the **XLingPaper** configuration for the **XMLmind XML Editor**, you can create the web page or PDF file literally at the press of a button.

That “something that can be made into a PDF file” mentioned above is called XSL-FO.<sup>4</sup> This is a standard way to convert XML files into print. One potential implication of this relates to the problem mentioned above of submitting a finished document to a publisher, only to have them re-key it due to using a different formatting program. If the publisher uses XSL-FO, then you will be all set. You merely need to give them the .fo version of your **XLingPaper** file. I need to hasten to add, however, that it may be still some time before many publishers begin using this technology. Thus, while the **XLingPaper** package solves the other problems mentioned in section 1, the publishing problem may still be an issue until the publisher starts using this technology, too. We did do an experiment along these lines by submitting an article Cheryl Black wrote using **XLingPaper** to the SIL Electronic Working Papers. It has now been published at <http://www.sil.org/silewp/abstract.asp?ref=2008-002>. The compositor that worked with me on getting this paper done acknowledged that there was a learning curve involved, but did say she would welcome other papers written using **XLingPaper**.

In addition, the **XLingPaper** package now includes ways to define publisher style sheets. Such style sheets delineate the layout information for a particular publisher. The layout information includes such things as page size, header and footer information, chapter and section formatting and number styles, reference formatting and order, etc. The **XLingPaper** package comes with several such style sheets, including ones for SIL, [Language](#), and [IJAL](#). Since the **XLingPaper** data are

---

<sup>2</sup>These same elements are rendered as hyperlinks in the PDF output as well.

<sup>3</sup>One can also produce output in Word 2003 format using the XMLmind **XSL-FO Converter** program (<http://www.xmlmind.com/foconverter/>). Unfortunately, however, some tables and examples do not come out formatted well (the **XSL-FO Converter** program does not pay attention to font metrics when calculating column widths). One has to manually adjust them. Also, there is no way to automatically insert Word styles with this tool. Finally, any tables in footnotes are thrown away.

It is certainly possible for someone to write an XSLT transform for **XLingPaper** that would produce a Word file, by the way.

<sup>4</sup>See <http://www.w3.org/TR/xsl/> and <http://www.w3schools.com/xslfo/default.asp> for more on XSL-FO.

actionable, one merely associates a style sheet to one's **XLingPaper** document and then produces the appropriate PDF, formatted per the stylesheet. Ideally, editors would create or help to create these style sheets.

By the way, I am assuming that editors will continue to play their crucial roles in the publication process. Even though an author is writing using “actionable data,” the editing process should definitely still happen.

## 5 Archiving

Another advantage is that since the **XLingPaper** file is in XML, it also can serve as an archiving format. XML is “plain text” (as opposed to some proprietary binary format), so it will always be readable. Since each entity is tagged according to the **XLingPaper** Document Type Definition (DTD), if one archives the DTD along with the file, then one has kept the essential information about the content.<sup>5</sup> The data remain “actionable” even in this archived form.

In the Mexico branch of SIL we are considering archiving **XLingPaper** documents by enclosing them in archive-oriented XML. That is, the archivist can mark up all of the pertinent archiving information in XML (such as OLAC <http://www.language-archives.org/>) and then merely insert the **XLingPaper** XML within the archive XML. Thus, the author of the paper is responsible for the paper-oriented XML while the archivist is responsible for the archive-oriented XML.

## 6 Other tools that already use **XLingPaper**

The **XLingPaper** format is already being used by at least two computational tools.

The first such tool is the **FieldWorks Language Explorer** program (<http://www.sil.org/computing/fieldworks/index.html>). It uses **XLingPaper** in two places: the automatically generated morphological grammar sketch and in exporting interlinear texts (the latter can be done in one of several formats).<sup>6</sup> One use of this is to include interlinearized text as an appendix. One can even copy an interlinear portion and use it to create an example in the body of the paper. **XLingPaper** will automatically create a reference from that example to the text in the appendix.

The second tool is the **PAWS Starter Kit** (see Black and Black 2008). It produces a draft of a syntactic grammar write-up in **XLingPaper** format based on the answers one gives to questions about the sentence structure of a language. One can answer the questions, produce the draft of the write-up and then refine the resulting document. This is one way to “jump start” a grammar write-up.

---

<sup>5</sup>The **XLingPaper** DTD contains many comments explaining the purpose of most elements and attributes. Such additional documentation to the DTD improves the value of archiving the DTD along with the XML document.

<sup>6</sup>One can export interlinear from **FieldWorks Language Explorer** in several formats besides **XLingPaper**, too. These include HTML, Open Office Writer, Microsoft Word 2003, and an XML format proposed in Hughes, Bird and Bow (2003).

## 7 The value of the XMLmind XML Editor

Given that **XLingPaper** is used in these two tools, it should not be surprising to learn that **XLingPaper** is not new. The problem has been that editing XML has been a bit challenging. One had to manually key the XML mark-up oneself and all that mark-up could make it hard to see what one was actually writing.

For example, imagine having to key what is shown in (1).

```
(1) <example num="xYalalagZapotecVerbClass1">
    <listInterlinear letter="xYalalagZapotecVerbClass1Pass">
      <lineGroup>
        <line>
          <langData lang="IYalalag">utecho</langData>
        </line>
        <line>
          <langData lang="IYalalag">u-te-cho</langData>
        </line>
        <gloss lang="IGloss">Fut-to.pass(trans)-1PIIncl</gloss>
        </line>
      </lineGroup>
    </listInterlinear>
    <listInterlinear letter="xYalalagZapotecVerbClass1Limp">
      <lineGroup>
        <line>
          <langData lang="IYalalag">u-:ke'nia'cho</langData>
        </line>
        <line>
          <langData lang="IYalalag">u-:ke'nia'-cho</langData>
        </line>
        <gloss lang="IGloss">Fut-to.limp(intrans)-1PIIncl</gloss>
        </line>
      </lineGroup>
    </listInterlinear>
  </example>
```

Clearly this would be hard to key directly. In fact, it is difficult to imagine who would want to key such things all the time.

In the Fall of 2005, I learned about the **XMLmind XML Editor** which looked like it at last provided a way to edit XML without actually having to see the XML mark-up. So I have written some **XLingPaper**-specific configuration files for this editor. Compare (1) above with how it shows in the **XMLmind XML Editor** as given in (2).

(2)

```

(xYalagZapotecVe)
(xYalagZapotecVerbClass1)
  utecho
  u-te-cho
  Fut-to.pass(trans)-1PIIncl
(xYalagZapotecVerbClass1)
  u-:ke'nia'cho
  u-:ke'nia'-cho
  Fut-to.limp(intrans)-1PIIncl

```

All of the information, including the tagging that is in (1) is also present in (2). It is just that the **XMLmind XML Editor** hides the information that gets in the way. This is the value of the **XMLmind XML Editor**. As a structured editor, it only lets you insert elements that correspond to the “grammar” of the DTD. At the same time, you only see the pertinent information you need as an author. Rather than being WYSIWYG, it is WYSIWYN “What you see is what you need.”

Here is how the example in (2) comes out in a web page (the example numbering happened to be 25 in this paper):

(3) (25) a. utecho  
 u-te-cho  
 Fut-to.pass(trans)-1PIIncl  
 b. u-:ke'nia'cho  
 u-:ke'nia'-cho  
 Fut-to.limp(intrans)-1PIIncl

The **XMLmind XML Editor**<sup>8</sup> is a downloadable editor that makes it quite easy to create not only **XLingPaper** documents, but also XHTML documents, among others. This editor is a structured editor and has at least the following very nice features:

- One never sees the XML element and attributes markers. These are all nicely hidden.
- It uses Unicode.
- There are templates which make it easy to start a new document and also to easily create various kinds of examples, lists, tables, etc.
- As a structured editor, one can only insert elements (e.g. citations, examples, lists) when it is appropriate to do so.
- There are ways to easily set references to examples, sections, etc.
- One can maintain a master file of references that one can then reuse for multiple documents.
- One can maintain a master file of abbreviations that one can then reuse for multiple documents.
- One can maintain a master file of index terms that one can then reuse for indexing multiple documents.
- It has a nice find and replace capability.
- It has the ability to easily cut and paste entire collections of elements.
- It has spelling checkers for English, French, Spanish, German, Czech, Polish, Russian, Slovak, and Swedish.
- The user interface can be in English, Czech, French, Italian, Spanish, or German.
- It works with **Keyman 6.0**.

<sup>8</sup>See <http://www.xmlmind.com/xmleditor/>. Be sure to choose the appropriate license for your situation.

- It has some ways to find and key Unicode and other special characters.
- Unlike some other XML Editors, performance does not seem to degrade for a large file, once the file is loaded.
- It is possible to create and maintain modular documents (ones which consist of multiple files).

For an unsolicited review of the XMLmind XML Editor, see

<http://www.writersua.com/articles/xmlmind/index.html>.

A known issue with the **XMLmind XML Editor** editor is with right-to-left scripts. One can indeed key data in a right-to-left script and it will appear correctly in the editor. The problem arises when one tries to edit the right-to-left material: the insertion point (where you start typing) is always calculated from the left. This can make editing right-to-left material a bit challenging.

## 8 Future Plans

**XLingPaper** continues to be developed. I welcome your feedback on any aspect of it. Current plans include the following.

- Publishing information: one would be able to tag document-specific publishing information such as copyright and title page publishing items. That is, one would associate both an **XLingPaper** document and a stylesheet as mentioned above with this additional publishing information. One could then produce a PDF for the published document.
- Produce formatted output via the typesetting system called TeX (Knuth 1984).<sup>9</sup>

## 9 How to get the **XLingPaper** package

You can obtain the latest information about this **XLingPaper** package by visiting this web site: <http://www.sil.org/%7Eblacka/xlingpap/index.htm>.

The package includes documentation illustrating how to use **XLingPaper** with the **XMLmind XML Editor**, including dozens of screen shots.

## 10 Conclusion

This article has introduced a Third Wave approach to writing linguistic papers and books. The method uses XML, including the **XLingPaper** package specially designed for authoring papers and the **XMLmind XML Editor** tool. This method not only makes writing linguistic articles and books more pleasant, it also provides “actionable data” which can be formatted into multiple outputs as well as serve as an archivable format.

## References

- Black, Cheryl A. and H. Andrew Black. 2008. “PAWS: Parser And Writer for Syntax: Drafting Syntactic Grammars in the Third Wave.” *SIL Forum for Language Fieldwork 2008-ZZZ*.
- Hughes, Baden, Steven Bird and Catherine Bow. 2003. <http://eprints.unimelb.edu.au/archive/00000455/>

---

<sup>9</sup>For more information on TeX, see <http://en.wikipedia.org/wiki/TeX#Community> and <http://www.tug.org/>.

- Knuth, Donald E. 1984. *The TeXBook (Computers and Typesetting, Volume A)*. Reading, Massachusetts: Addison-Wesley.
- Simons, Gary F. and H. Andrew Black. 2008. "Third Wave Writing and Publishing." *SIL Forum for Language Fieldwork* 2008-XXX.