Third Wave Writing and Publishing

Gary F. Simons
SIL International and Graduate Institute of Applied Linguistics

H. Andrew Black
SIL International and University of North Dakota

SIL International
2009

SIL Forum for Language Fieldwork 2009-005, December 2009
Copyright © 2008 Gary F. Simons, H. Andrew Black and SIL International
All rights reserved

Contents

Abstract 3

1 Introduction 3

2 Beware of Second Wave Tools in Third Wave Clothing 3

3 XML: the *lingua franca* of the Third Wave 4
- 3.1 Basic Concepts 5
- 3.2 Other Features of XML 6
- 3.3 Why is XML Strategic? 6
  - 3.3.1 Information Interchange 6
  - 3.3.2 A Huge Tool Chest 6
  - 3.3.3 Ideal Archiving Format 7

4 Third Wave Writing: Working Smarter 7

5 Third Wave Publishing: Incredible Impact 10

6 Conclusion 11

A Features of XML Lacking in Standard Format 11
- A.1 Validation with a DTD 11
- A.2 Character Sets 12
- A.3 Special Characters 12
- A.4 External (File) Entities 13
- A.5 Linking 13

References 14

Abstract

Now that we live in the period of the Third Wave, we need to reconsider how we do writing and publishing in the "Information Age." Our mental models of publishing tend to align with the Second Wave (or "Industrial Age") approach in which the product of writing is formatted pages that are mass-produced in printed books and journals. But the product of Third Wave writing is actionable information that may be published in print or published in multiple digital forms with just-in-time customization. We show that a Third Wave approach has several other advantages, including supporting archiving in a form that offers long term repurposability, allowing us to "work smarter" to increase quality and productivity, and making an incredible publishing impact possible. We also give examples which show that it is possible to write and publish in a Third Wave style today.

1 Introduction

In his insightful book, *The Third Wave*, Toffler (1980) describes the three major waves of development in the progress of human civilization. In the First Wave, known as the "Agricultural Age," people left behind the nomadic life style of hunting and gathering as they settled in sedentary communities supported by agriculture and animal husbandry. During this age, the economy was based primarily on access to land and labor.

The Second Wave, known as the "Industrial Age," broke in the nineteenth century. Society experienced a quantum change as the development of machines enabled an order of magnitude increase in levels of production and speeds of travel and communication. The economy was based on access to energy. A major theme of the Second Wave was the mass production of uniform goods in factories.

We now live in the period of the Third Wave, also known as the "Information Age." Society has experienced another quantum change as information processing machines have empowered another order of magnitude increase in production. Just as Second Wave machines amplified the muscle power of people, Third Wave machines are amplifying their brain power. The Third Wave economy is based on *actionable* knowledge. A major theme of the Third Wave is diversity and the ability to put actionable knowledge to work to create just-in-time customization of products to meet individual needs.

This paper discusses how this Third Wave is changing the areas of writing and publishing. We begin by giving a warning in section 2. We then introduce XML in section 3 and discuss its role as the *lingua franca* of the Third Wave. Section 4 follows with an illustration of how XML and the actionable knowledge of the Third Wave allow us to work smarter. Finally, in section 5, we address the incredible impact that Third Wave publishing can have.

2 Beware of Second Wave Tools in Third Wave Clothing

We can summarize the development of writing and publishing technologies through the ages as shown in (1).

(1)

	Process	Output	Publication
1st Wave	By hand	Hand-written page	Hand-made product
2nd Wave	By machine	Printed page	Manufactured product
3rd Wave	By computer	Actionable information	Digital download

In the First Wave, all writing and publishing was done by hand with quill or stylus or chisel; every product, from an individual word to an entire book, was unique. The Second Wave introduced manufactured uniform type. Typewriters increased both the speed and uniformity of writing for the individual. The printing press made it possible to mass produce thousands of identical products, which then had to be warehoused and distributed.

The Third Wave is not just about using computers as a more advanced machine; it also produces a different kind of product. The product is not the printed page, but it is the information that lies behind such pages made available in an actionable format. That is, the true product is the information itself which can be processed to yield, among other things, a printed page; it can also be used for other purposes. The primary publishing mechanism of the Third Wave is not to manufacture products that have a high overhead cost to produce, store, and ship. Rather, the primary publishing mechanism is on-demand digital download from a web site.

On the surface of things, the word processor would appear to be a Third Wave tool. After all, the process is by computer and the product is something that can be disseminated by digital download. However, a word processor is really just a Third Wave implementation of a Second Wave tool that produces nicely formatted pages. A WYSIWYG ("What you see is what you get") word processor does not produce actionable knowledge, which is the true product of the Third Wave. A word processor produces pages that people can read, but not information that computers can put into action. That is why the flip side of WYSIWYG is "What you see is all you get."

Third Wave products can generate Second Wave products, but the reverse is not true. That is, when a true Third Wave tool produces actionable information, a computer can automatically transform it into nicely formatted pages (among other uses), but formatted pages cannot be automatically transformed to actionable information. The notion of actionable information is illustrated in section 4, but first we describe XML which is the universal language for the interchange of actionable information in today's world.

3 XML: the *lingua franca* of the Third Wave

What does this actionable information of the Third Wave look like? Basically, it is data that are explicitly structured in terms of what the data mean (as opposed to being structured in terms of what they look like in a display). This happens inside computers in databases, for instance. But to become fully actionable, information needs to be transmitted from computer to computer so that it can be put into action wherever needed. It also needs to be archived in a form that is independent of any single software product, so that it can be called back into action at some point in the future when the software that created it no longer exists.

For these purposes of interchanging and archiving structured information we need a descriptive markup language. The Standard Format Markers that are used with SIL tools like Shoebox (Buseman et al. 2000) and Toolbox (see http://www.sil.org/computing/toolbox/) are an example of descriptive

markup, but they are little used outside these programs. The universal standard powering the Third Wave economy, on the other hand, is XML, for "Extensible Markup Language." While XML fills the same niche as Standard Format, it has some important differences:

- Its information structure is fully explicit so that nothing is left to interpretation.
- Its information structure is formally defined in a grammar so that documents can be automatically validated.
- It is a standard of the World Wide Web Consortium (see http://www.w3.org/XML/) so that it has become the *lingua franca* for information interchange on the Internet.
- It has been embraced by all the big software companies so that there are hundreds of tools that support it.

3.1 Basic Concepts

The basic unit of document structure in XML is the *element*. An element has a name that identifies the type of information the element contains. Enclosing the name in angle brackets forms a *start tag*. The *end tag* adds a slash before the name. Between the two tags is the *content* of the element. For instance, a `headword` element with the content of *fanga* is illustrated in (2).

(2)
```
<headword>fanga</headword>
```

A start tag may include *attributes*. For instance, the start tag for a `sense` element with an `n` attribute indicating that it is the second sense is illustrated in (3).

(3)
```
<sense n="2">
```

The content of an element may have other elements embedded within it. For instance, consider the following example:

(4)
```
<entry>
    <headword>fanga</headword>
    <sense n="1">
        <pos>vi</pos>
        <def>to eat (of humans)</def>
    </sense>
    <sense n="2">
        <pos>n</pos>
        <def>food</def>
    </sense>
</entry>
```

The XML in (4) is for a dictionary entry. It gives the headword and two senses. Each sense has a part of speech and a definition. The senses also have their sense number given as an attribute. When, as in this case, the element names actually document what the data are, it is called *descriptive markup*. On the other hand, when the element names indicate how the data are to be formatted, it is called *presentational markup* (Coombs, Allen & DeRose 1987).

3.2 Other Features of XML

The basic features of XML shown in the preceding section are similar in power to what can be expressed with Standard Format. This section briefly mentions a number of ways in which XML goes well beyond Standard Format. These ways are exemplified in more technical detail in appendix A for those who are interested in such specifics.

First, one can validate the content of an XML file via a Document Type Definition or DTD. Such a DTD can state which sequences of elements are licit or even what the hierarchy of elements may be. Standard Format has no built-in equivalent.

Second, one can overtly state the encoding used for the characters. This is often Unicode. With Standard Format, any such encoding is implicit at best.

Third, special characters can be handled cleanly via what are called character entities. In particular, one can use a kind of mnemonic symbol for the character so that someone reading the file at a later date will quickly know what the character is. Standard Format has no equivalent mechanism.

Fourth, one may also include other XML files within an XML file which can make file management much easier. There is no equivalent in Standard Format.

Finally, one can make links from one part of an XML document to another without having to repeat any information like one typically does with Standard Format.

3.3 Why is XML Strategic?

One may wonder why all of this is so strategic. We have identified five reasons.

- It is the *lingua franca* for information interchange in the global information community.
- It gives us an ever growing tool chest.
- It is the ideal format for long-term archiving.
- It allows us to "work smarter" as creators of information.
- It allows us to ride the Third Wave as publishers of information.

We will discuss the first three here. The other two are so important that they deserve to be in main sections (see sections 4 and 5).

3.3.1 Information Interchange

As we mentioned earlier, XML has become the *lingua franca* for information interchange in the global information community. When machines want to share information, it is all about information structure, not about presentation format. That is, machines cannot reliably apprehend the meaning of information by interpreting display formats. They need to know exactly what the elements of information are and how they are structured. For this reason, XML is powering the new paradigm of *B2B* ("business-to-business") communication on the Web.

3.3.2 A Huge Tool Chest

The second reason that XML is strategic is that there is an ever-growing tool chest of programs and utilities that support XML and associated standards. Some other XML-based standards include:

- *XSLT*: A tree transformation language. This is used to transform XML information into other XML-based formats or into web pages or into plain text formats (like Standard Format). It is

built into web browsers like Internet Explorer and Firefox to support on-the-fly display of XML information as web pages.

- *XSL-FO*: A style language for page formatting. This is used with XSLT to convert XML information into nicely formatted documents or books in a page description format like PDF.[1]
- *SVG*: Scalable Vector Graphics language. This can be used with XSLT to convert XML information into complex graphic displays.
- *XPath* and *XQuery*: Query languages. The former is used to find elements that match specified patterns in XML information. The latter is used to search for patterns in XML information and construct new documents containing the results.
- *ISO Schematron*: Validation language. This goes beyond the context-free power of a DTD to provide a means of specifying context-sensitive constraints on what it means to be an instance of a valid document.

Given all this, it is not surprising that hundreds of vendors are supporting XML, including "big name" companies like Adobe, Google, IBM, Lotus, Microsoft, Netscape, and Sun. Besides this, there are hundreds of free XML-related tools. See http://www.garshol.priv.no/download/xmltools for a list of many of them.

3.3.3 Ideal Archiving Format

The third reason that XML is strategic is that XML is the ideal format for long-term archiving. This point is fully developed in Simons (2006).

XML is ideal for archiving because it is just plain text that has been marked-up in such a way as to preserve all the structure information. Since it is plain text, it will remain readable for generations into the future. We do not have to keep old versions of programs around just so we can read a binary format that some particular program happened to use.

This achieves the property of "data independence." When the data are stored in XML, we are not at the mercy of any single vendor and its proprietary format. Users in the future who find material archived in XML will be able to make use of it because it will be straightforward to interpret the structure of the information. It will then be possible to use XML tools to transform the information to a format that their future favorite software will be able to make use of.

XML separates information content from presentation format. What is archived is structured content. In the future, when we need user-friendly presentation formats, we can create them on-the-fly via XSLT stylesheets. We can even have a set of stylesheets to transform the information into several different formats. This is illustrated with three presentation forms of the same archival form in section 5 of Simons (2006). By contrast, when all we archive is a presentation form, then it is not possible to repurpose the information.

4 Third Wave Writing: Working Smarter

The fourth reason that XML is strategic is that XML allows us to "work smarter" as creators of information. Toffler (1980) observes that in the Third Wave, brain force wins out over brute force (which was the *modus operandi* of the Second Wave). Or, as a First Wave sage put it, "If the iron

[1]This is how we created this PDF file you are currently reading. To be more specific, we used the approach outlined in Black (2009).

is blunt, and one does not whet the edge, he must put forth more strength; but wisdom helps one to succeed." (Ecclesiastes 10:10, RSV)

So why does XML actually help us "work smarter?" It is because the computer has a grammar for the content and structure of the data (e.g. the Document Type Definition; see section 3.2 as well as appendix A.1). Several things follow from this:

- Users are guided in the process of building documents. Only the appropriate elements are offered as possibilities.
- Users are prevented from ever creating invalid files.
- Users in the field have the same validation device as the publisher does.
- Stylesheets are guaranteed to give the expected results since the structured information will always match the format that was expected by the stylesheet designer.
- We can develop sophisticated integrity checks on the information in documents using tools like ISO Schematron or special purpose XSLT scripts.
- Programmers need never write extra code to deal with improperly formatted input.

All of the above are examples of working smarter in the process of creating a product that is itself smarter, since it is actionable. Whereas a Second Wave document is just a document, a Third Wave document is actionable knowledge that can be put to work.

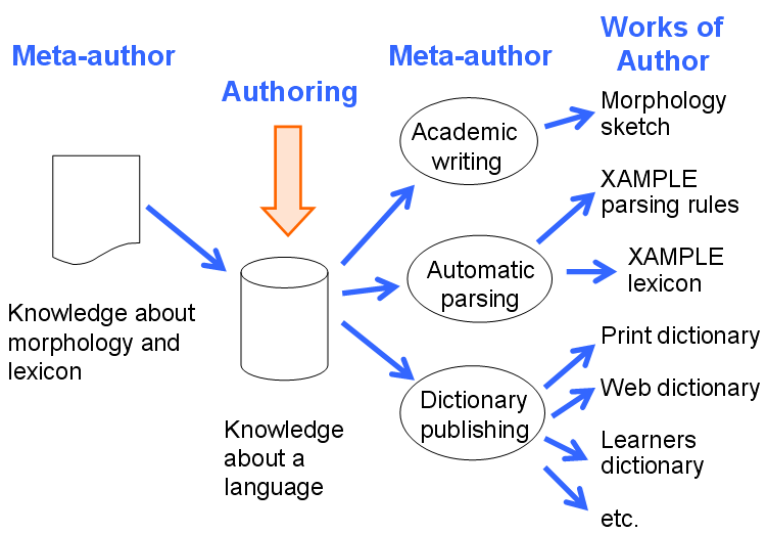
The thing that empowers working smarter with XML is the DTD which is actionable knowledge at a higher level. The author who is creating actionable knowledge as an XML document is being guided by the actionable knowledge of a meta-author who designed the document type and its integrity constraints (as well as any presentation formats that may have been made actionable in stylesheets). Had such knowledge been expressed as a conventional style manual in a Second Wave word processing document, it would just be a document, and every author would be required to read and learn the knowledge in the manual, and then to try to be consistent in applying it.

Third Wave writing is working smarter because the tool used for authoring implements the actionable knowledge of the meta-author to create an environment in which the user (as author) is guided and constrained by that meta-knowledge. As a result, novices perform at a much higher level than they would if given a word processor to perform the same writing task. What is more, the knowledge of the user is also being captured in a form that is actionable. As a consequence the publication process is streamlined since much of it can be automated. Even though the publication process is streamlined, this does not mean that humans like editors and graphic artists are no longer needed. To the contrary, they still play crucial roles in the process.

The FieldWorks Language Explorer program[2] is an excellent example of this Third Wave approach as illustrated in the following diagram:

[2]See http://www.sil.org/computing/fieldworks/flex/.

(5)



The diagram in (5) begins at the far left with people who have expert knowledge about morphology and lexicography; they make their knowledge actionable when they design the structure of the database and the user-interface of the program. The user who has knowledge about a particular language then uses the FieldWorks Language Explorer program to "write" what they know. When they write, however, they are being guided to enter structured information in the database. This actionable information is then transformed in various ways to produce multiple kinds of outputs. These include:

- A draft of a morphology sketch that organizes the information in the database into a well-structured document that gives a human-readable description of the morphology. (See Black and Simons (2006) for an example of this and the next output.)
- The rules and lexicon files for a morphological parser so that it can automatically analyze words in texts and thus tell the user what their current grammatical description predicts the analysis of a given word to be.
- A dictionary in various possible forms.

Following the Second Wave approach, the user would be given a word processor and expected to create all the outputs shown at the far right of the diagram. Using the Third Wave approach, the user only enters actionable information into the database. The meta-authored actionable knowledge of experts in academic writing, automated parsing, and dictionary publishing is able to transform the user's actionable information into all the needed Second Wave documents. (In the implementation of the FieldWorks Language Explorer program, the actionable information in the database is output as XML so that XML-based transformation and presentation tools can be used to produce the final outputs.)

Note in all of this that the driving force is not speed. We are not trying to do the same old things faster. That is merely a quantitative brute force view that belies a Second Wave way of thinking. Rather, the Third Wave is about harnessing actionable knowledge to improve quality and productivity. It is about producing higher quality results from non-experts by using expert knowledge to guide their work. It is about being more productive by automating that expert knowledge to create results that we would not even have had time for in the past. This is the proper qualitative brain force view of what the Third Wave is about.

The secret here is finding the leverage point. With a word processor, we can only produce documents one at a time. By creating actionable knowledge instead of documents, we can actually produce *n* documents at a time. One novice input of language-specific data yields *n* quality products about that language. One expert input of language-general knowledge yields quality results from *m* novices.

See Black and Black (2009) for another instance of Third Wave writing in the area of syntactic description. In yet another example, Black (2009) describes a Third Wave authoring tool for linguistic papers.

5 Third Wave Publishing: Incredible Impact

The fifth and final reason that XML is strategic is that XML allows us to ride the Third Wave as publishers of information. This offers a completely new paradigm for publishing.

In Second Wave publishing, books are mass produced and warehoused until sold. There is a high cost associated with manufacturing and storing the goods, with the result that it is not cost-effective to publish for very small niche markets. In the transition from the Second Wave to the Third Wave, we are able to take advantage of Third Wave approaches to make the publishing of Second Wave products more cost effective. Books are electronically mastered in PDF. These can be offered as a product for self-service download from the web, thus obviating the overhead costs of manufacturing, warehousing, and shipping. For customers who still want a physical book, digital presses now make it cost effective to produce a single copy of a book from a PDF master when a customer orders one, thus removing the risk associated with mass production of products that might not sell.

Third Wave publishing, however, goes even further than this. While the theme of the Second Wave is uniformity and mass production (which is still achieved in a PDF master of a book), the major theme of the Third Wave is diversity and just-in-time customization. When the product of writing is structured information, the publisher is able to repurpose that information in multiple products, including interactive ones that allow customers to obtain customized results that match their criteria.

For example, as a Second Wave book, the *Ethnologue* sells about 500 volumes per year. As a Third Wave interactive repository on the Web, during the 2006 fiscal year, there were 50 million page requests on www.ethnologue.com in 10 million user sessions. The difference in impact of the two approaches to publishing is staggering.

Third Wave publications offer many advantages:

- They are accessible anytime from anywhere.
- Anyone can discover them through a web search.
- They are interactive (e.g. with mechanisms like links, menus, and queries), which makes it possible to provide just-in-time customization.
- They can hide or reveal detail as requested.
- The same content is presented in multiple ways for multiple audiences or purposes.
- Page limits (due to the high cost of manufacturing) are no longer an issue.
- They can easily grow over time in multiple editions.
- They can be changed at any time, even to correct a single error.

The high costs associated with the physical limits of the Second Wave approach created an economy of scarcity. The lower costs associated with the near limitless potential for reproduction

and storage with the Third Wave's digital approach is creating a new economy of abundance. See Simons (2007) for a discussion of the economy of abundance and how it could empower a new cyberinfrastructure for doing linguistics in the twenty-first century. The key to such an infrastructure will be the interoperation that is made possible when multiple publishers share actionable information that conforms to common standards.

6 Conclusion

We have shown that the Third Wave approach to writing and publishing has several advantages, including:

- It produces structured data that are actionable.
- It supports archiving in a form that offers long-term repurposability.
- It allows us to "work smarter" to increase quality and productivity.
- It makes possible an incredible publishing impact.

In addition to this, we have given examples showing that it is actually possible to write and publish in a Third Wave style today. However, the Second Wave approach persists as the predominant mindset in our collective thinking. It is high time that we abandon word processing in favor of information structuring and thereby harness the new quality and productivity leverage of the Third Wave.

Appendix A: Features of XML Lacking in Standard Format

As noted in section 3.2, XML has several capabilities that simply are not directly available with Standard Format.[3] This appendix outlines those five capabilities.

A.1 Validation with a DTD

Earlier in section 3, we mentioned that XML can be validated. An XML document actually can be checked for two things: well-formedness and validity. An XML document is *well-formed* if its markup follows the basic rules of XML structure (for instance, every start tag must have a matching end tag). This is the minimum level of integrity for an XML document. As an even stronger test, an XML document is *valid* if its markup also conforms to a named Document Type Definition or DTD.[4] A DTD is simply a context-free phrase structure grammar that specifies the vocabulary (e.g. the names of all the elements and attributes used) and syntax (rules of where the elements and attributes are allowed or required) in the document type. For example, (7) gives a DTD for the XML in (6), repeated here from (4).

[3]Standard Format is a descriptive markup system that was developed by SIL iin 1979 and still in use today in products like Shoebox (Buseman et al. 2000) and Toolbox (see http://www.sil.org/computing/toolbox/).

[4]One may also use an XML Schema to validate an XML document (see http://www.w3.org/XML/Schema). While XML Schema provide better constraining mechanisms than DTDs, they are much harder for humans to read.

(6)
```
<entry>
    <headword>fanga</headword>
    <sense n="1">
        <pos>vi</pos>
        <def>to eat (of humans)</def>
    </sense>
    <sense n="2">
        <pos>n</pos>
        <def>food</def>
    </sense>
</entry>
```

(7)
```
<!DOCTYPE entry [
<!ELEMENT entry (headword, sense+)>
<!ELEMENT headword (#PCDATA) >
<!ELEMENT sense (pos, def) >
<!ATTLIST sense n CDATA #IMPLIED >
<!ELEMENT pos (#PCDATA) >
<!ELEMENT def (#PCDATA) >
]>
```

In this DTD, the document type is for `entry`. It says that an `entry` consists of a `headword` and one or more `sense` elements. A `headword` element contains text (where `#PCDATA` stands for "parseable character data") and a `sense` element contains a `pos` and a `def` element. In addition, a `sense` element can have an `n` attribute (which contains `#CDATA` or "character data" and is `#IMPLIED` if absent, i.e. it is optional). Both `pos` and `def` elements contain text.

A.2 Character Sets

Another feature of XML is support for different character sets. The default character set is Unicode which defines nearly 100,000 characters (Unicode 2007). It covers all the major writing systems of the world and has extension mechanisms for up to a million characters.

If a different character set is used, the XML declaration at the beginning of the document includes an *encoding declaration*. For example, the ISO standard for Latin characters used in Central and Eastern Europe would be declared as in (8).

(8)
```
<?xml encoding="ISO-8859-2" ?>
```

A.3 Special Characters

XML also has two ways to express special characters.

A *character reference* can be used to specify any character that cannot be typed on the keyboard. One does this by simply referring to its number in the declared character set. For example, ó is Unicode character code 243. The notation for expressing this as a character reference is `ó`. Using this character reference to encode the name Givón in an XML document would be as in (9).

(9) `Givón`

Entity references provide a way to make mnemonic (rather than numerical) references to special characters. For example, to define an entity reference for ó, one would add a declaration like the one in (10) to the DTD.

(10) `<!ENTITY oacute "ó">`

Using this entity reference to encode the name Givón in an XML document would then be as in (11).

(11) `Givón`

Some XML editing tools actually hide these details from the user. The user only sees the special character, not the character reference or the entity reference which is there "under the hood." The XML editor described in Black (2009) is such a tool that hides these details from the user.

A.4 External (File) Entities

In addition, an XML document can be assembled from multiple files by declaring *external entities*. For example, suppose one has a Document Type Definition called `book.dtd` and one has a particular book containing three chapters, each in its own file. Then a master XML file for the entire book could look like this:

(12)
```
<?xml standalone="no" ?>
<!DOCTYPE book SYSTEM "book.dtd" [
<!ENTITY chap1 SYSTEM "chap1.xml">
<!ENTITY chap2 SYSTEM "chap2.xml">
<!ENTITY chap3 SYSTEM "chap3.xml">
]>
<book>&chap1; &chap2; &chap3;</book>
```

This defines three external entities (`chap1`, `chap2`, and `chap3`) each of which is associated with a file (`chap1.xml`, `chap2.xml`, and `chap3.xml`). The content of the *book* element merely consists of references to the three external entities (`&chap1;`, `&chap2;`, and `&chap3;`). An XML processor will insert these files on-the-fly when the document in (12) is processed in order to instantiate the complete book.

A.5 Linking

The final feature of XML that we discuss is linking. Linking is a way for one part of a document to refer to another. Cross-referencing in documents is a typical application for linking.

Linking in XML involves two parts. First, the element that is being referred to must have a unique identifier; this is expressed in an attribute that is declared to be an *ID* attribute. Second, other elements can then point to it by using that unique identifier as the value of an attribute that is declared to be an *IDREF* attribute. For example, suppose the DTD for a dictionary contains attribute declarations like the following for its sense and synonym elements.

(13)
```
<!ATTLIST sense id ID #REQUIRED>
...
<!ATTLIST synonym ref IDREF #REQUIRED>
```

Then the XML document itself could contain something like this:

(14)
```
<entry>
    <headword>fanga</headword>
    <sense n="1" id="fanga.sense.1">
        <pos>vi</pos>
        <def>to eat (of humans)</def>
    </sense>
    <sense n="2" id="fanga.sense.2">
        <pos>n</pos>
        <def>food</def>
    </sense>
</entry>
...
<entry>
...
    <lexRelations>
        <synonym ref="fanga.sense.1"/>
    </lexRelations>
...
</entry>
```

That is, the latter entry includes a synonym reference to the first sense of the *fanga* entry.

References

Black, Cheryl A. and H. Andrew Black. 2009. "PAWS: Parser And Writer for Syntax: Drafting Syntactic Grammars in the Third Wave." *SIL Forum for Language Fieldwork* 2009-002. http://www.sil.org/silepubs/abstract.asp?id=51432

Black, H. Andrew. 2009. "Writing Linguistic Papers in the Third Wave." *SIL Forum for Language Fieldwork* 2009-004. http://www.sil.org/silepubs/abstract.asp?id=52286

Black, H. Andrew and Gary F. Simons. 2006. "The SIL FieldWorks Language Explorer Approach to Morphological Parsing." Computational Linguistics for Less-studied Languages: Texas Linguistics Society 10. 3–5 November 2006, Austin, TX. http://www.sil.org/~simonsg/preprint/FLExParser%20Preprint.pdf

Buseman, Alan, Karen Buseman, David Coward, Dean Jordan, Tom Bogle, Rod Early, Mark Pedrotti, Brian Yoder, and Bryan Wussow. 2000. *The Linguist's Shoebox: Tutorial and User's Guide.* Waxhaw, North Carolina: SIL International.

Coombs, James H., Allen H. Renear, and Steven J. DeRose. 1987. "Markup systems and the future of scholarly text processing." *Communications of the ACM* 30(11):933-947.

Simons, Gary F. 2006. "Ensuring that digital data last: The priority of archival form over working form and presentation form." *SIL Electronic Working Papers* 2006-003. http://www.sil.org/silewp/abstract.asp?ref=2006-003

Simons, Gary F. 2007. "Doing linguistics in the twenty-first century: Interoperation and the quest for the global riches of knowledge." E-MELD/DTS-L Workshop: Toward the Interoperability of Language Resources. 13–15 July 2007, Palo Alto, CA. http://linguistlist.org/tilr/papers/TILR%20Plenary.pdf

Toffler, Alvin. 1980. *The Third Wave.* New York: Morrow.
Unicode Consortium. 2007. *The Unicode Standard, Version 5.0.* Boston, MA: Addison-Wesley. http://www.unicode.org/versions/Unicode5.0.0/