PassiveTEX: from XML to PDF

Michel Goossens
CERN, IT Division, CH-1211 Genève 23, Switzerland
michel.goossens@cern.ch

Sebastian Rahtz
Oxford University Computing Services, Oxford, United Kingdom
sebastian.rahtz@oucs.ox.ac.uk

Abstract

This article introduces PassiveTEX, a library of TEX macros based on `xmltex`, that processes XML documents containing XSL formatting objects and generates PDF or DVI output. We show examples of typesetting XML sources marked up using the TEI, DocBook, and MathML DTDs.

Introduction

New information becomes available on the Internet every day, and increasingly material is becoming available in XML. However, when one wants to print the information on the screen one is faced with several shortcomings, the more important being the low typographic quality of the result or the problem of how to serialize a 'tree' of pages onto a linear output medium. In this article we explain how a source document marked up in XML can be typeset nicely by a two-step procedure that combines a transformation of the XML source into XSL formatting objects and the direct interpretation of these XML objects with Sebastian Rahtz' PassiveTEX, a variant of TEX based on David Carlisle's `xmltex` [4].

The choice of TEX as the typesetting engine is justified by the fact that TEX handles mathematics in a natural way, can manage several languages simultaneously (hyphenation, typographic rules, multiple encodings and alphabets, right-left typesetting, etc.), and has a good model for marking up complex tabular information. Moreover, TEX can easily handle very long documents, such as large software manuals, or bills for hundreds of thousands of telephone subscribers.

In the first part of this article we use as an example some literary texts, including poems by the French poet Paul Verlaine and the Russian poet Alexander Blok, marked up according to the TEI DTD (*Text Encoding Initiative* [3], Lite version). In the second part we show an example of a more technical document, including a couple of simple math formulae, using the DocBook [14] and MathML [17] DTDs.

The present article was prepared in XML using TEI markup. It was not actually typeset with PassiveTEX, but transformed into LATEX with an XSLT transformation stylesheet and typeset using the `ltugproc` LATEX class file.

XSL formatting objects

The *Extensible Stylesheet Language* (XSL) is a language for describing page designs. For any given class of arbitrarily structured XML documents or data files, an XSL stylesheet allows you to express how the content contained in the XML should be presented. The stylesheet indicates how source elements should be styled, laid out, and paginated in some presentation form. In this article we only address issues related to high quality *typesetting* (using TEX). Our XML examples can be transformed into other representations (see [7] for more discussion), in particular HTML, and XSL stylesheets targeting HTML are available for both the TEI [12] and DocBook [15] markup schemes.

An XSL processor reads an XML source document together with an XSL stylesheet, and produces the presentation of the given XML content according to the instructions in the stylesheet. The preparation of the presentation form proceeds in two steps. First, a *result tree* is constructed from the XML input source tree (*tree transformation* step). Second, the result tree is interpreted to produce the formatted results (*formatting* step), that can be presented on output media such as a computer screen, a WAP display, on paper, in speech, etc.

The result tree can be very different from the structure of the source tree, since parts from the input can be deleted from the result tree, while supplementary information (e.g., table of contents) can be generated. The tree transformation step also has to add the information necessary to format the result tree.

The nodes of the result tree are *formatting objects*, whose semantics are expressed in terms of a catalog of formatting object classes defined in the XSL Specification [24]. They denote typographic abstractions such as page, paragraph, table, list, etc. Fine control over the presentation of these abstractions is provided by a set of *formatting properties*, such as those controlling alignments, color, fonts, spacing, writing-mode, hyphenation etc. XSL offers a rich range of formatting objects as one of its aims is to cover the semantic functionality of both the *Document Style Semantics and Specification Language* (DSSSL) [10] and *Cascading Stylesheets* (CSS) [16] models as completely as possible.

PassiveTEX: implement XSL using TEX

In the previous section we explained how an input XML document can be transformed into a new XML document containing a result tree of XSL formatting objects. How do we go about rendering those objects with a real typesetting engine, and getting high-quality printout? One way is to use TEX itself. PassiveTEX is a library of TEX macros which can typeset the XSL formatting objects. In particular, PassiveTEX combined with the pdfTEX variant of TEX generates high-quality PDF files in a single operation. PassiveTEX allows one to choose TEX as the formatter of choice for XML, while hiding the details of its operation from the user.

PassiveTEX derives from and builds on `xmltex` [4], a TEX package by David Carlisle, providing the core XML parser and UTF8 handler. Ideas and TEX code are also inherited from earlier work by Sebastian Rahtz on his JadeTEX package, which implemented the output of the Jade DSSSL processor's TEX backend, and already used a catalogue of Unicode/TEX mappings.

Issues in using TEX Since PassiveTEX is based on TEX one can rapidly develop and test new high level code, knowing that TEX will take care of the typesetting details. Moreover, TEX is a well-understood, robust, and free page formatter, where good support for fonts, graphics inclusion, hyperlinks, etc., is already available. One can also profit from TEX's mature handling of language issues, including hyphenation, and its high-quality math rendering. Moreover, pdfTEX allows direct generation of high-quality PDF.

There are, however, some drawbacks associated with using TEX as typesetting engine. Firstly, we are constrained to use TEX's page makeup model, and have to force the XSL formatting objects to fit that model, which is not always straightforward. Moreover, since PassiveTEX is actually layered over LATEX it is too easy to allow things to fall through and take LATEX defaults.

On a more practical level, TEX macro writing is obscure and difficult and thus the system is not transparent for most programmers. And, last but not least, TEX is large and monolithic, and unsuited to embedding in other applications.

Users of the system with a TEX background can find it confusing to understand that TEX no longer does all the work, which is now split betwen the stylesheet and the formatter; the four important points to remember are:

- No explicit use is made of LATEX's high-level constructs, in particular there are no sections, lists, cross-references, bibliographies, etc.
- all vertical and horizontal space is explicit in the specification;
- page and line breaking is left to TEX: the rest is up to the stylesheet;
- XSL formatting objects use XML syntax, so that the underlying character set is Unicode. By default, entities are mapped to their Unicode position.

PassiveTEX will switch to using the Unicode-based TEX variant soon (Omega [8]), to handle non-Latin material more naturally.

Two extensions are needed for practical use. The first is support of MathML; this is largely in place (it is simply passed through as-is by an XSL stylesheet), but needs some tuning. In particular, the intricacies of equation numbers remain to be dealt with properly. Second, we need to support *Scalable Vector Graphics* (SVG) [21] XML code somehow (e.g., by direct interpretation, translation into MetaPost [9], or pre-processing). Moreover, SVG fragments need to be recognized directly to perform in-line graphical functions (e.g., setting text at an angle). No work at all has been done on this.

Support for XSL formatting objects Not all parts of the XSL specification are fully supported. The XSL formatting objects which are implemented fairly well cover the page specifications, blocks, inline sequences, lists, graphics inclusion, floats, font properties, and links. Not so well implemented at present are all the table properties, and object margins, borders, and padding. In order to properly implement these, we have to put every paragraph into an explicit TEX box ‘just in case’, and this becomes slow and clumsy. Tables, too, are treated rather differently in XSL, with many properties assigned at the cell level.

Parts of the specification that are not yet implemented at all include bidi handling, backgrounds, aural properties, and a large number of assorted properties.

A small example The set of XSL stylesheets for the TEI DTD (`tei-fo` [12]) specify how the various XML elements can be transformed into XSL formatting objects (see Section 7.6.6 of [7] for a more detailed discussion). Without further comments, the following code example shows the transformations for a paragraph (`p` element, lines 1 to 6) and emphasized material (`emph` element, lines 7 to 11).

```
1  <xsl:template match="p">
2    <fo:block indent-start="10pt"
3              space-before="12pt">
4      <xsl:apply-templates/>
5    </fo:block>
6  </xsl:template>
7  <xsl:template match="emph">
8    <fo:inline-sequence font-style="italic">
9      <xsl:apply-templates/>
10   </fo:inline-sequence>
11 </xsl:template>
```

A few words about `xmltex` As PassiveTEX uses `xmltex` to read and interpret the XML source it is appropriate to say a few words about it. `xmltex` is a non validating (and not 100% conforming) namespace-aware XML parser implemented in TEX. `xmltex` reads the encoding declaration and tries to implement it; in particular it handles UTF8. It reads the DTD subset and expands entities properly, it uses namespaces to load modules of TEX code to process elements (for instance for MathML, see the following example). `xmltex` offers limited access to child and parent elements to guide the interpretation process.

The workhorse of an `xmltex` package file is the `\XMLelement` command, whose four parameters indicate how TEX will handle a given element, its attributes and its content. The following example shows how a few simple elements of the MathML namespace are handled.

```
1  \DeclareNamespace{m}
2    {http://www.w3.org/1998/Math/MathML}
3
4  \XMLelement{m:math}
5    {}
6    {\begin{equation}}
7    {\end{equation}}
8
9  \XMLelement{m:mn}
10   {}
11   {\xmlgrab}
12   {\mathrm{#1}}
13
14 \XMLelement{m:msqrt}
15   {}
16   {\xmlgrab}
17   {\sqrt{#1}}
```

Lines 1 and 2 declare the prefix `m` to be used for referring to elements in the given namespace. Lines 4 to 7 declare that the start and end tag of a `math` element are transformed into the opening and closing of an `equation` environment. Lines 9 to 12 define how a MathML `mn` (number) element is typeset. Its contents are stored (grabbed, see line 11) and then injected as parameter of a `\mathrm` to be typeset in roman. Similarly, lines 14 to 17 show how a square root (`msqrt`) element and its content are transformed into LATEX’s `\sqrt` and its argument.

To fine-tune the output, `xmltex` supports processing instructions to manipulate TEX formatting directly.

The biggest problem at present in using `xmltex` to write PassiveTEX is that it uses TEX grouping to make sure each element is handled in the right namespace. Because TEX does not allow for nested grouping, each element is caught in a group; while `\aftergroup` can help a bit, this is the first hurdle for anyone trying to extend the package.

An example of TEI markup typeset with PassiveTEX

In this section we use an XML source file that contains a collection of literature of the French poet Paul Verlaine, the Russian poet Alexander Blok, the English novelist Thomas Hardy, and the Finnish author Aleksis Kivi. It is marked up according to the TEI Lite DTD. The master file `poemstei-utf8` follows:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE TEI.2 SYSTEM "teixlite.dtd" [
3  <!ENTITY dash "&#x2010;">
4  <!ENTITY mdash "&#x2014;">
5  <!ENTITY oelig "&#x0153;">
6  <!ENTITY Verlaine SYSTEM "Verlainetei-utf8.xml">
7  <!ENTITY Blok     SYSTEM "Bloktei-utf8.xml">
8  <!ENTITY Hardy    SYSTEM "beyes.xml">
9  <!ENTITY Kihlaus  SYSTEM "kihlaus.xml">
10 ]>
11 <TEI.2>
12 <teiHeader type="text" status="new">
13 <fileDesc>
14 <titleStmt>
15 <title>Various languages and scripts</title>
16 <author>Collected by Michel Goossens</author>
17 <respStmt>
18 <resp>Transcription in TEI.2 markup</resp>
19 <name>Michel Goossens</name></respStmt>
20 </titleStmt>
21 <publicationStmt>
22 <distributor>Distributed by mg</distributor>
```

```
23 </publicationStmt>
24 </fileDesc>
25 <profileDesc>
26 <langUsage default="NO">
27 <language id="EN">English</language>
28 <language id="FI">Finnish</language>
29 <language id="FR">French</language>
30 <language id="RU">Russian</language>
31 </langUsage>
32 </profileDesc>
33 </teiHeader>
34   <text>
35 <front>
36 <titlePage>
37 <docTitle>
38 <titlePart>Languages and scripts</titlePart>
39 </docTitle>
40 <docAuthor>Collected by Michel Goossens</docAuthor>
41 <docDate>July 2000</docDate>
42 </titlePage>
43 </front>
44 <body>
45 &Verlaine;
46 &Blok;
47 &Hardy;
48 &Kihlaus;
49 </body>
50 </text>
51 </TEI.2>
```

This XML source document uses the UTF8 encoding, since we want to mix various alphabets (in this case Latin and Cyrillic). UTF8 is a Unicode-based encoding [13] that can encode each character in Unicode's base plane and its sixteen extended planes (allowing for more than one million characters) with at most three bytes. Documents that only contain ASCII can be coded using one byte per character. Parts of the XML source of the foreign language documents (`Verlainetei-utf8.xml`, defined on line 6 and input on line 45; and also `Bloktei-utf8.xml` defined on line 7 and input on line 46) are shown in Figure 1.

To typeset this document we first transform the XML into XSL formatting objects using an XSLT stylesheet, as explained previously. Then, this intermediate XML file is handled by `xmltex` and PassiveTEX. In what follows we use James Clark's `xt` XSLT processor ([5], see also Section 7.6.4 of [7]), but any conforming XSLT processor should be able to handle these transformations.

```
xt infile.xml style.xsl fotex.xml
latex fotex.tex
```

The file `style.xsl` is an XSL stylesheet that contains the XSLT transformations needed to transform the input XML file `infile.xml` into XSL formatting objects. The resulting intermediate XML document `fotex.xml` can be interpreted by applications that can render XSL formatting objects. In our case we use PassiveTEX.

Below we show part of the beginning of the XML file `fotex.xml` that corresponds to the first page of the collection of poems of Verlaine transformed using Sebastian Rahtz' TEI XSL stylesheets [12].

```
1    ...
2 <fo:flow flow-name="xsl-region-body"
3          font-family="Times Roman"
4          font-size="10pt">
5
6   <fo:block text-align="center" space-after="8pt">
7     <fo:block font-size="16pt">
8       <fo:inline font-weight="bold">
9         Poems in various languages and scripts
10      </fo:inline>
11    </fo:block>
12    <fo:block font-size="14pt">
13      <fo:inline font-style="italic">
14        Collected by Michel Goossens</fo:inline>
15    </fo:block>
16    <fo:block font-size="14pt">
17      July 2000
18    </fo:block>
19  </fo:block>
20
21  <fo:block keep-with-next.within-page="always"
22            id="N115"
23            text-align="start"
24            font-size="14pt"
25            text-indent="-3em"
26            font-weight="bold"
27            space-after="3pt"
28            space-before.optimum="9pt">
29    1. Paul Verlaine, F\^etes galantes (1869)
30  </fo:block>
31
32  <fo:block keep-with-next.within-page="always"
33            id="N128"
34            text-align="start"
35            font-size="12pt"
36            font-weight="bold"
37            space-after="2pt"
38            space-before.optimum="4pt"
39            text-indent="-3em">
40  1.1. <fo:inline font-style="italic">
41         Clair de lune</fo:inline>
42  </fo:block>
43
44  <fo:block text-align="start"
45            space-before.optimum="4pt"
46            space-after.optimum="4pt">
47    <fo:block space-before.optimum="0pt"
48              space-after.optimum="0pt">
49      Votre \^ame est un paysage choisi
50    </fo:block>
51      ...
52  </fo:block>
53  ...
54 </fo flow>
```

At the beginning of the file (not shown) the various page masters are defined (margins, height, width, etc.), and the static page information (running headers, footers) is initialized. Then, the construction of formatted output pages starts. We show what happens at the start of page 1 (compare with

```
 1  <?xml version="1.0" encoding="UTF-8"?>
 2  <div1 lang="FR">
 3  <head>Paul Verlaine, Fêtes galantes (1869)</head>
 4  <!- ++++++++++++++++++++++++++++++++++++++++++++++++++++ ->
 5  <div2>
 6  <head><hi rend="ital">Clair de lune</hi></head>
 7  <!- ++++++++++++++++++++++++++++++++++++++++++++++++++++ ->
 8  <lg type="stanza" org="uniform" sample="complete" part="N">
 9  <l>Votre âme est un paysage choisi</l>
10  <l>Que vont charmant masques et bergamasques,</l>
11  <l>Jouant du luth, et dansant, et quasi</l>
12  <l>Tristes sous leurs déguisements fantasques. </l>
13  </lg>
14    ...
15  </div2>
16  <!- ++++++++++++++++++++++++++++++++++++++++++++++++++++ ->
17  <div2>
18  <head><hi rend="ital">Les indolents</hi></head>
19  <lg type="stanza" org="uniform" sample="complete" part="N">
20  <l>Bah ! malgré les destins jaloux,</l>
21  <l>Mourons ensemble, voulez-vous ?</l>
22  <l>&dash; La proposition est rare.</l>
23  </lg>
24    ...
25  </div2>
26  <!- ++++++++++++++++++++++++++++++++++++++++++++++++++++ ->
27  <div2>
28  <head><hi rend="ital">Colloque sentimental</hi></head>
29  <lg type="stanza" org="uniform" sample="complete" part="N">
30  <l>Dans le vieux parc solitaire et glacé,</l>
31  <l>Deux formes ont tout à l'heure passé.</l>
32  </lg>
33  <lg>
34  <l>Leurs yeux sont morts et leurs lèvres sont molles,</l>
35  <l>Et l'on entend à peine leurs paroles.</l>
36  </lg>
37    ...
38  </div2>
39  </div1>
```

```
 1  <?xml version="1.0" encoding="UTF-8"?>
 2  <div1 lang="RU">
 3  <head>Александр Блок, Стихи о прекрасной даме (1901-1902)</head>
 4  <!- ++++++++++++++++++++++++++++++++++++++++++++++++++++ ->
 5  <div2>
 6  <head>«Отдых напрасен. Дорога крута...» (28 декабря 1901)</head>
 7  <lg type="stanza" org="uniform" sample="complete" part="M">
 8  <l>Отдых напрасен. Дорога крута.</l>
 9  <l>Вечер прекрасен. Стучу в ворота.</l>
10  </lg><lg>
11  <l> Дольнему стуку чужда и строга,</l>
12  <l>Ты рассыпаешь кругом жемчуга.</l>
13  </lg>
14    ...
15  </div2>
16  <!- ++++++++++++++++++++++++++++++++++++++++++++++++++++ ->
17  <div2>
18  <head>«Я вышел. Медленно сходили...»
19    (С.-Петербург, 25 января 1901)</head>
20  <lg type="stanza" org="uniform" sample="complete" part="M">
21  <l>Я вышел. Медленно сходили</l>
22  <l>На землю сумерки зимы.</l>
23  <l>Минувших дней младые были</l>
24  <l>Пришли доверчиво из тьмы...</l>
25  </lg>
26    ...
27  </div2>
28  <!- ++++++++++++++++++++++++++++++++++++++++++++++++++++ ->
29  <div2>
30  <head>«Ветер принес издалка...» (29 января 1901)</head>
31  <lg type="stanza" org="uniform" sample="complete" part="M">
32  <l>Ветер принес издалка</l>
33  <l>Песни весенней намек,</l>
34  <l>Где-то светло и глубоко</l>
35  <l>Неба открылся клочок.</l>
36  </lg>
37    ...
38  </div2>
39  </div1>
```

Figure 1: Partial input source of French (left) and Russian (right) documents

the output shown in the upper left corner of Figure 3). The `fo:flow` element contains the material that has to be typeset and cut into individual pages by the typesetting engine. We see that the default document font is Times Roman at 10 pt.

Lines 6 to 19 typeset a block of centered material. In particular, lines 7–11 take care of the document title that is typeset in 16 point bold, lines 12–15 correspond to the author's name typeset in 14 point italic, while lines 16–18 typeset the date in 14 point roman. All this information is obtained from the content of the `titlepage` element of the XML master source file `poemstei-utf8`, shown previously (lines 36–42).

The remaining text is in the external files `Verlainetei-utf8.xml` (entity reference on line 46 of `poemstei-utf8`, which contains the poems of Paul Verlaine in French, a fragment of which is seen in the left half of Figure 1), and `Bloktei-utf8.xml` (entity reference on line 47 of `poemstei-utf8`, which contains the poems of Alexander Blok in Russian, a fragment of which is seen in the right half of Figure 1).

The author and the title of the collection of poems (specified as the content of the `head` element of a `div1` element on line 3 in Figure 1) are handled on lines 21 to 30 in the formatting objects output, where a block for the first level title is created. Note that the section number (1. on line 29) is generated by the XSLT application. Similarly, the title of an individual poem (specified as the content of the `head` element of a `div2` element in Figure 1, e.g., on line 6) is handled on lines 32 to 42. Once more, the number (1.1 on line 40) is generated automatically. Then, the stanza and the lines inside the stanza of the poem (the `lg` and `l` elements in Figure 1) are treated. For instance, the stanza on lines 8–13 in Figure 1 corresponds to the block on lines 44–52 in the formatting objects output, in particular the input line 8 becomes the block 47–50. More details about the various XSL formatting objects and their attributes can be found in the XSLT Recommendation [24].

The second stage of the process (typesetting the XSL formatting objects with LaTeX) uses `xmltex` that is called by running LaTeX on the intermediate file `fotex.tex` containing the following three lines:

```
1 \def\xmlfile{fotex.xml}
2   \input xmltex.tex
3   \end{document}
```

Line 1 specifies the file that has to be interpreted by `xmltex`, before it takes control on line 2. The result of treating the file `poemstei-utf8.xml` with the two-step procedure outlined is shown in Figure 2. Figure 3 shows, for comparison, the result of typesetting the same XML source file directly with `xmltex`, where we had to define for each XML element and its attributes how it should be rendered by LATEX.

Besides PassiveTEX, FOP [2], originally developped by James Tauber, and presently supported by the Apache Project, is another application, written in Java, that transforms XSL formatting objects into PDF. Because it uses Java, it is very portable, but it does not yet handle many aspects of fine typography as well as TEX. Commercial products targeting the same task of producing excellent typographic output from XSL formatting objects have also been announced (*RenderX*, ArborText's *Epic*), but are not yet available.

DocBook markup and some words about mathematics

Up to now we have discussed a document marked up according to the Text Encoding Initiative (TEI) schema. In this section we show that other XML markup schemes can also be used, in particular, DocBook [14] which we will combine with MathML [17], W3C's XML DTD for mathematics.

DocBook is an XML (originally SGML) DTD which is especially well-suited for marking up technical documents. It is used extensively for preparing software reference guides and computer equipment manuals.

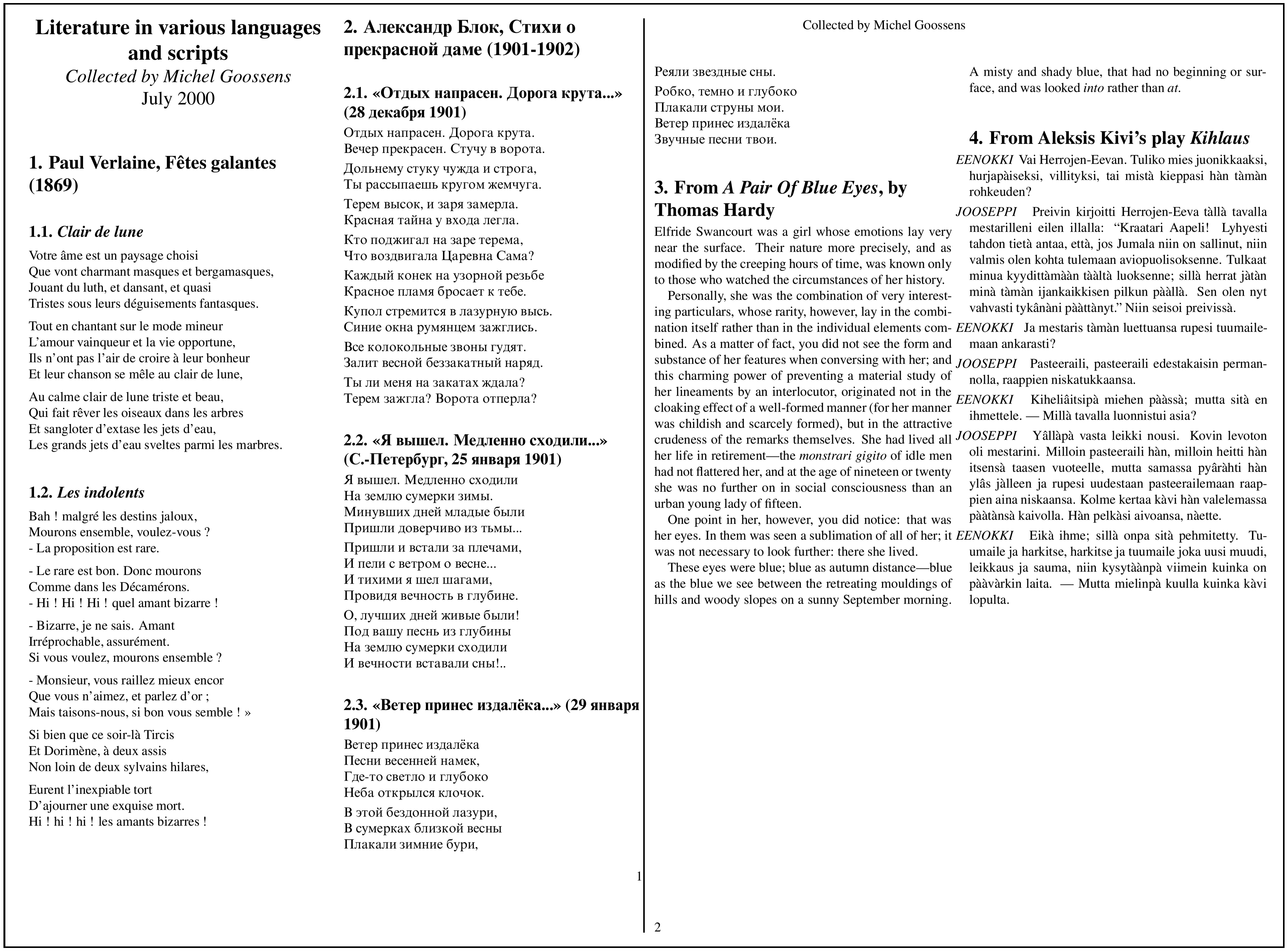
The DocBook DTD or schema contains hundreds of elements to mark up clearly and explicitly the different components of an electronic document (book, article, reference guide, etc.), not only displaying its hierarchical structure but also indicating the semantic meaning of the various elements. The structure of the DTD is optimized to allow for customization, thus making it relatively straightforward to add or eliminate certain elements or attributes, to change the content model for certain structural groups, or to restrict the value that given attributes can take.

An example of DocBook markup We are not going to cover the DocBook vocabulary in any detail (see Walsh's book [14] or his DocBook Web page). We shall only consider an example where we used the DocBook markup schema. A file `dbxmml.xml`, whose first part is reproduced below, contains various elements of the DocBook vocabulary. It will allow us to test the typesetting paradigm based on PassiveTEX outlined for the TEI schema previously.

```
 1 <?xml version="1.0" encoding="ISO-8859-1"?>
 2 <!DOCTYPE article SYSTEM
 3 "/usr/local/share/docbookxml/4.11/docbookx.dtd"
 4 [
 5  <!ENTITY LaTeX "LaTeX">
 6  <!ENTITY TeX   "TeX">
 7  <!ENTITY PTeX  "PassiveTeX">
 8  <!ENTITY xmltex
 9           "<application>xmltex</application>">
10  <!ENTITY % equation.content "(math+)">
11  <!ENTITY % inlineequation.content "(math+)">
12  <!ENTITY % mathml
13           SYSTEM "mathmldtd/mathml2.dtd">
14  <!ATTLIST math xmlns CDATA #FIXED
15           "http://www.w3.org/1998/Math/MathML">
16 <!-- load MathML -->
17 %mathml;
18 ]>
19 <article>
20 <articleinfo>
21 <title>A Docbook document featuring a
22        few formulae</title>
23 <author><forename>Michel</forename>
24         <surname>Goossens</surname>
25 </author>
26 <pubdate>Wednesday, 18 July 2000</pubdate>
27 <abstract>
28 <para>
29 This XML document is marked up according the
30 DocBook schema It shows a few elements of the
31 DocBook vocabulary, as well as a couple of
32 examples of mathematical expressions where
33 we used MathML markup.
34 </para>
35 </abstract>
36 </articleinfo>
37 <section>
38 <title>The Docbook model</title>
39 <para>
40 DocBook <xref role="bib" linkend="docbook"
41 endterm="docbookab"/> is an XML model
42 <xref role="bib" linkend="rec-xml"
43 endterm="rec-xmlab"/> for marking up technical
44 documents.  It is particularly well-suited for
45 software reference guides and computer
46 equipment manuals.
47 </para>
```

The DTD internal subset (lines 4–18) contain the general entity definitions of `LaTeX`, `PassiveTeX`, `TeX`, and `xmltex` (lines 5 to 9), and on lines 10 and 11 parameter entity definitions specifying the content models for the `equation.content` and `inlineequation.content` elements, that can contain one or more `math` elements. Lines 12–13 define the `mathml` parameter entity that defines the system file containing the MathML DTD, which is loaded on line 17. Line 14 specifies the namespace of the `math` element. The remaining lines have markup using the DocBook vocabulary, and their meaning should be rather straightforward to understand.

An example of MathML presentation markup [17] follows.



Literature in various languages and scripts

Collected by Michel Goossens

July 2000

1. Paul Verlaine, Fêtes galantes (1869)

1.1. *Clair de lune*

Votre âme est un paysage choisi
Que vont charmant masques et bergamasques,
Jouant du luth, et dansant, et quasi
Tristes sous leurs déguisements fantasques.

Tout en chantant sur le mode mineur
L'amour vainqueur et la vie opportune,
Ils n'ont pas l'air de croire à leur bonheur
Et leur chanson se mêle au clair de lune,

Au calme clair de lune triste et beau,
Qui fait rêver les oiseaux dans les arbres
Et sangloter d'extase les jets d'eau,
Les grands jets d'eau sveltes parmi les marbres.

1.2. *Les indolents*

Bah ! malgré les destins jaloux,
Mourons ensemble, voulez-vous ?
- La proposition est rare.

- Le rare est bon. Donc mourons
Comme dans les Décamérons.
- Hi ! Hi ! Hi ! quel amant bizarre !

- Bizarre, je ne sais. Amant
Irréprochable, assurément.
Si vous voulez, mourons ensemble ?

- Monsieur, vous raillez mieux encor
Que vous n'aimez, et parlez d'or ;
Mais taisons-nous, si bon vous semble ! »

Si bien que ce soir-là Tircis
Et Dorimène, à deux assis
Non loin de deux sylvains hilares,

Eurent l'inexpiable tort
D'ajourner une exquise mort.
Hi ! hi ! hi ! les amants bizarres !

2. Александр Блок, Стихи о прекрасной даме (1901-1902)

2.1. «Отдых напрасен. Дорога крута...» (28 декабря 1901)

Отдых напрасен. Дорога крута.
Вечер прекрасен. Стучу в ворота.

Дольнему стуку чужда и строга,
Ты рассыпаешь кругом жемчуга.

Терем высок, и заря замерла.
Красная тайна у входа легла.

Кто поджигал на заре терема,
Что воздвигала Царевна Сама?

Каждый конек на узорной резьбе
Красное пламя бросает к тебе.

Купол стремится в лазурную высь.
Синие окна румянцем зажглись.

Все колокольные звоны гудят.
Залит весной беззакатный наряд.

Ты ли меня на закатах ждала?
Терем зажгла? Ворота отперла?

2.2. «Я вышел. Медленно сходили...» (С.-Петербург, 25 января 1901)

Я вышел. Медленно сходили
На землю сумерки зимы.
Минувших дней младые были
Пришли доверчиво из тьмы...

Пришли и встали за плечами,
И пели с ветром о весне...
И тихими я шел шагами,
Провидя вечность в глубине.

О, лучших дней живые были!
Под вашу песнь из глубины
На землю сумерки сходили
И вечности вставали сны!..

2.3. «Ветер принес издалёка...» (29 января 1901)

Ветер принес издалёка
Песни весенней намек,
Где-то светло и глубоко
Неба открылся клочок.

В этой бездонной лазури,
В сумерках близкой весны
Плакали зимние бури,

1

Collected by Michel Goossens

Реяли звездные сны.

Робко, темно и глубоко
Плакали струны мои.
Ветер принес издалёка
Звучные песни твои.

3. From *A Pair Of Blue Eyes*, by Thomas Hardy

Elfride Swancourt was a girl whose emotions lay very near the surface. Their nature more precisely, and as modified by the creeping hours of time, was known only to those who watched the circumstances of her history.

Personally, she was the combination of very interesting particulars, whose rarity, however, lay in the combination itself rather than in the individual elements combined. As a matter of fact, you did not see the form and substance of her features when conversing with her; and this charming power of preventing a material study of her lineaments by an interlocutor, originated not in the cloaking effect of a well-formed manner (for her manner was childish and scarcely formed), but in the attractive crudeness of the remarks themselves. She had lived all her life in retirement—the *monstrari gigito* of idle men had not flattered her, and at the age of nineteen or twenty she was no further on in social consciousness than an urban young lady of fifteen.

One point in her, however, you did notice: that was her eyes. In them was seen a sublimation of all of her; it was not necessary to look further: there she lived.

These eyes were blue; blue as autumn distance—blue as the blue we see between the retreating mouldings of hills and woody slopes on a sunny September morning. A misty and shady blue, that had no beginning or surface, and was looked *into* rather than *at*.

4. From Aleksis Kivi's play *Kihlaus*

EENOKKI Vai Herrojen-Eevan. Tuliko mies juonikkaaksi, hurjapàiseksi, villityksi, tai mistà kieppasi hàn tàmàn rohkeuden?

JOOSEPPI Preivin kirjoitti Herrojen-Eeva tàllà tavalla mestarilleni eilen illalla: "Kraatari Aapeli! Lyhyesti tahdon tietà antaa,ettà, jos Jumala niin on sallinut, niin valmis olen kohta tulemaan aviopuolisoksenne. Tulkaat minua kyydittàmààn tààltà luoksenne; sillà herrat jàtàn minà tàmàn ijankaikkisen pilkun pààllà. Sen olen nyt vahvasti tykânàni pààttànyt." Niin seisoi preivissà.

EENOKKI Ja mestaris tàmàn luettuansa rupesi tuumailemaan ankarasti?

JOOSEPPI Pasteeraili, pasteeraili edestakaisin permannolla, raappien niskatukkaansa.

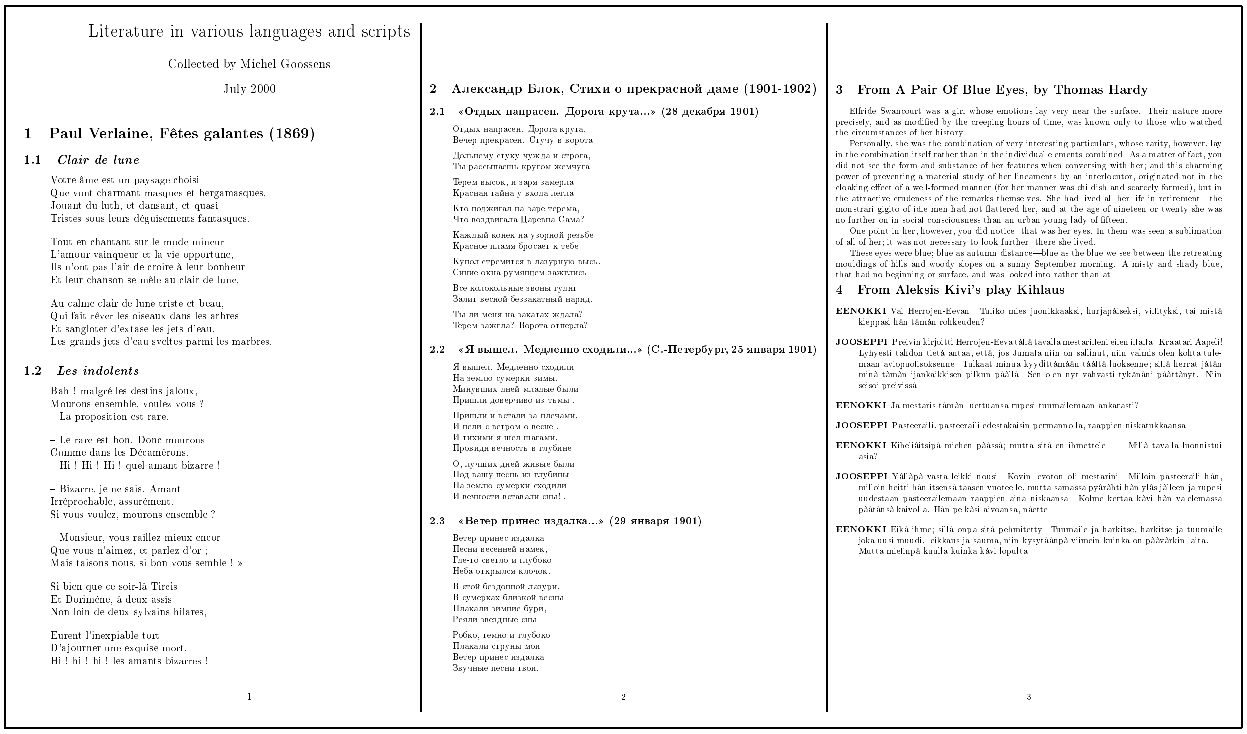
EENOKKI Kiheliâitsipà miehen pààssà; mutta sità en ihmettele. — Millà tavalla luonnistui asia?

JOOSEPPI Yâllàpà vasta leikki nousi. Kovin levoton oli mestarini. Milloin pasteeraili hàn, milloin heitti hàn itsensà taasen vuoteelle, mutta samassa pyâràhti hàn ylàs jàlleen ja rupesi uudestaan pasteerailemaan raappien aina niskaansa. Kolme kertaa kàvi hàn valelemassa pààtànsà kaivolla. Hàn pelkàsi aivoansa, nàette.

EENOKKI Eikà ihme; sillà onpa sità pehmitetty. Tuumaile ja harkitse, harkitse ja tuumaile joka uusi muudi, leikkaus ja sauma, niin kysytàànpà viimein kuinka on pààvàrkin laita. — Mutta mielinpà kuulla kuinka kàvi lopulta.

2

Figure 2: Typesetting a collection of literature marked up according to the TEI, using XSL formatting objects



Literature in various languages and scripts

Collected by Michel Goossens

July 2000

1 Paul Verlaine, Fêtes galantes (1869)

1.1 *Clair de lune*

Votre âme est un paysage choisi
Que vont charmant masques et bergamasques,
Jouant du luth, et dansant, et quasi
Tristes sous leurs déguisements fantasques.

Tout en chantant sur le mode mineur
L'amour vainqueur et la vie opportune,
Ils n'ont pas l'air de croire à leur bonheur
Et leur chanson se mêle au clair de lune,

Au calme clair de lune triste et beau,
Qui fait rêver les oiseaux dans les arbres
Et sangloter d'extase les jets d'eau,
Les grands jets d'eau sveltes parmi les marbres.

1.2 *Les indolents*

Bah ! malgré les destins jaloux,
Mourons ensemble, voulez-vous ?
– La proposition est rare.

– Le rare est bon. Donc mourons
Comme dans les Décamérons.
– Hi ! Hi ! Hi ! quel amant bizarre !

– Bizarre, je ne sais. Amant
Irréprochable, assurément.
Si vous voulez, mourons ensemble ?

– Monsieur, vous raillez mieux encor
Que vous n'aimez, et parlez d'or ;
Mais taisons-nous, si bon vous semble ! »

Si bien que ce soir-là Tircis
Et Dorimène, à deux assis
Non loin de deux sylvains hilares,

Eurent l'inexpiable tort
D'ajourner une exquise mort.
Hi ! hi ! hi ! les amants bizarres !

1

2 Александр Блок, Стихи о прекрасной даме (1901-1902)

2.1 «Отдых напрасен. Дорога крута...» (28 декабря 1901)

Отдых напрасен. Дорога крута.
Вечер прекрасен. Стучу в ворота.

Дольнему стуку чужда и строга,
Ты рассыпаешь кругом жемчуга.

Терем высок, и заря замерла.
Красная тайна у входа легла.

Кто поджигал на заре терема,
Что воздвигала Царевна Сама?

Каждый конек на узорной резьбе
Красное пламя бросает к тебе.

Купол стремится в лазурную высь.
Синие окна румянцем зажглись.

Все колокольные звоны гудят.
Залит весной беззакатный наряд.

Ты ли меня на закатах ждала?
Терем зажгла? Ворота отперла?

2.2 «Я вышел. Медленно сходили...» (С.-Петербург, 25 января 1901)

Я вышел. Медленно сходили
На землю сумерки зимы.
Минувших дней младые были
Пришли доверчиво из тьмы...

Пришли и встали за плечами,
И пели с ветром о весне...
И тихими я шел шагами,
Провидя вечность в глубине.

О, лучших дней живые были!
Под вашу песнь из глубины
На землю сумерки сходили
И вечности вставали сны!..

2.3 «Ветер принес издалка...» (29 января 1901)

Ветер принес издалка
Песни весенней намек,
Где-то светло и глубоко
Неба открылся клочок.

В єтой бездонной лазури,
В сумерках близкой весны
Плакали зимние бури,
Реяли звездные сны.

Робко, темно и глубоко
Плакали струны мои.
Ветер принес издалка
Звучные песни твои.

2

3 From A Pair Of Blue Eyes, by Thomas Hardy

Elfride Swancourt was a girl whose emotions lay very near the surface. Their nature more precisely, and as modified by the creeping hours of time, was known only to those who watched the circumstances of her history.

Personally, she was the combination of very interesting particulars, whose rarity, however, lay in the combination itself rather than in the individual elements combined. As a matter of fact, you did not see the form and substance of her features when conversing with her; and this charming power of preventing a material study of her lineaments by an interlocutor, originated not in the cloaking effect of a well-formed manner (for her manner was childish and scarcely formed), but in the attractive crudeness of the remarks themselves. She had lived all her life in retirement—the monstrari gigito of idle men had not flattered her, and at the age of nineteen or twenty she was no further on in social consciousness than an urban young lady of fifteen.

One point in her, however, you did notice: that was her eyes. In them was seen a sublimation of all of her; it was not necessary to look further: there she lived.

These eyes were blue; blue as autumn distance—blue as the blue we see between the retreating mouldings of hills and woody slopes on a sunny September morning. A misty and shady blue, that had no beginning or surface, and was looked into rather than at.

4 From Aleksis Kivi's play Kihlaus

EENOKKI Vai Herrojen-Eevan. Tuliko mies juonikkaaksi, hurjapàiseksi, villityksi, tai mistà kieppasi hàn tàmàn rohkeuden?

JOOSEPPI Preivin kirjoitti Herrojen-Eeva tàllà tavalla mestarilleni eilen illalla: Kraatari Aapeli! Lyhyesti tahdon tietà antaa, ettà, jos Jumala niin on sallinut, niin valmis olen kohta tulemaan aviopuolisoksenne. Tulkaat minua kyydittàmààn tààltà luoksenne; sillà herrat jàtàn minà tàmàn ijankaikkisen pilkun pààllà. Sen olen nyt vahvasti tykânàni pààttànyt. Niin seisoi preivissà.

EENOKKI Ja mestaris tàmàn luettuansa rupesi tuumailemaan ankarasti?

JOOSEPPI Pasteeraili, pasteeraili edestakaisin permannolla, raappien niskatukkaansa.

EENOKKI Kiheliâitsipà miehen pààssà; mutta sità en ihmettele. — Millà tavalla luonnistui asia?

JOOSEPPI Yâllàpà vasta leikki nousi. Kovin levoton oli mestarini. Milloin pasteeraili hàn, milloin heitti hàn itsensà taasen vuoteelle, mutta samassa pyâràhti hàn ylâs jàlleen ja rupesi uudestaan pasteerailemaan raappien aina niskaansa. Kolme kertaa kàvi hàn valelemassa pààtànsà kaivolla. Hàn pelkàsi aivoansa, nàette.

EENOKKI Eikà ihme; sillà onpa sità pehmitetty. Tuumaile ja harkitse, harkitse ja tuumaile joka uusi muudi, leikkaus ja sauma, niin kysytàànpà viimein kuinka on pààvàrkin laita. — Mutta mielinpà kuulla kuinka kàvi lopulta.

3

Figure 3: Typesetting a collection of literature marked up according to the TEI, using xmltex

```
1  <section>
2  <title>A MathML example</title>
3  <para>
4  A MathML formula can be typeset inline, as here
5  <inlineequation>
6  <math><mi>E</mi><mo>=</mo><mi>m</mi>
7  <msup><mi>c</mi><mn>2</mn></msup></math>
8  </inlineequation>, Einstein's famous equation.
9  </para>
10
11 <para>
12 A mathematical equation can also be typeset in
13 display mode using DocBook's
14 <sgmltag class="element">informalequation</sgmltag>
15 element, as is shown in the following example
16 containing a matrix:
17 </para>
18
19 <informalequation>
20 <math>
21 <mrow>
22   <mi>A</mi>
23   <mo>=</mo>
24   <mfenced open="[" close="]">
25     <mtable><!-- table or matrix -->
26       <mtr> <!-- row in a table  -->
27          <mtd><mi>x</mi></mtd><!-- table -->
28          <mtd><mi>y</mi></mtd><!-- entry -->
29       </mtr>
30       <mtr>
31          <mtd><mi>z</mi></mtd>
32          <mtd><mi>w</mi></mtd>
33       </mtr>
34     </mtable>
35   </mfenced>
36 </mrow>
37 <mtext>.</mtext>
38 </math>
39 </informalequation>
```

On lines 5–8 we use the `inlineequation` element, which contains a mathematical equation to be displayed inline, whereas line 19 to 39 show an `informalequation` element, that contains mathematical material (a matrix equation) to be typeset in display mode. The typeset result (with PassiveTEX, see below) is shown following the heading *A MathML example* in the lower left hand image of Figure 4.

The DocBook example and PassiveTEX We use the same two-step procedure (transform the XML file into XSL formatting objects and then typeset with PassiveTEX as outlined previously in the context of the TEI) here with DocBook. For the transformation of the XML DocBook markup into XSL formatting objects we use XSL stylesheets developed by Norman Walsh [15]. We customized the stylesheet somewhat to handle the mathematics and add templates for a few elements. The customized stylesheet `foplus.xsl` follows.

```
<?xml version='1.0'?>
<xsl:stylesheet
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
```

```
4   version="1.0"
5   xmlns:fo="http://www.w3.org/1999/XSL/Format"
6   xmlns:m="http://www.w3.org/1998/Math/MathML">
7
8  <xsl:import
9   href="/usr/local/share/xsl/1.15/docbook.xsl"/>
10
11 <!-- ****************************************
12   customisations of Norm's XSL FO stylesheets
13   **************************************** -->
14
15 <xsl:template match="m:math">
16  <xsl:copy>
17   <xsl:apply-templates mode="math"/>
18  </xsl:copy>
19 </xsl:template>
20
21 <xsl:template  mode="math"
22     match="* | @* | comment() |
23     processing-instruction()|text()">
24   <xsl:copy>
25     <xsl:apply-templates    mode="math"
26        select="* | @* |
27          processing-instruction() | text()"/>
28  </xsl:copy>
29 </xsl:template>
30
31 <xsl:template
32      match="informalequation|inlineequation">
33   <xsl:element name="{name(.)}">
34     <xsl:apply-templates/>
35   </xsl:element>
36 </xsl:template>
37
38 <xsl:template match="programlisting" priority="4">
39   <fo:block wrap-option="no-wrap"
40            whitespace-treatment="preserve"
41            font-family="monospace"
42            font-size="9pt"
43            space-before.optimum="4pt"
44            space-after.optimum="4pt"
45            text-align="start"
46   >
47     <xsl:apply-templates/>
48   </fo:block>
49 </xsl:template>
50
51 </xsl:stylesheet>
```

Lines 3, 5 and 6 define the prefixes of the various namespaces referenced: XSLT instructions (`xsl`), XSL formatting objects (`fo`), and MathML elements (`m`), respectively. On lines 8–9 we import Norman Walsh's XSL stylesheet `docbook.xsl`, which is part of his DocBook distribution. It transforms elements in the DocBook namespace into XSL formatting objects. The `xsl:import` XSL element must be placed at the beginning of the stylesheet if template definitions further down in the stylesheet are to take precedence over those defined in the imported stylesheet `docbook.xsl`. Lines 31 to 36 specify that the elements `informalequation` and `inlineequation` themselves and their contents are to be copied through. Similarly (lines 21–29), the MathML element `m:math` and the whole tree structure it supports have to be copied unchanged to the result tree. This means that the MathML elements must be dealt with by the application that interprets the XSL formatting objects (in our case PassiveTEX and `xmltex`). Finally, lines 38 to 49 show how we define the characteristics of the `programlisting` element that is used to represent verbatim material and is to be typeset *as-is* (similar to LATEX's `verbatim` environnement).

The two-step procedure to typeset the XML source file `dbxmml.xml` uses the following commands:

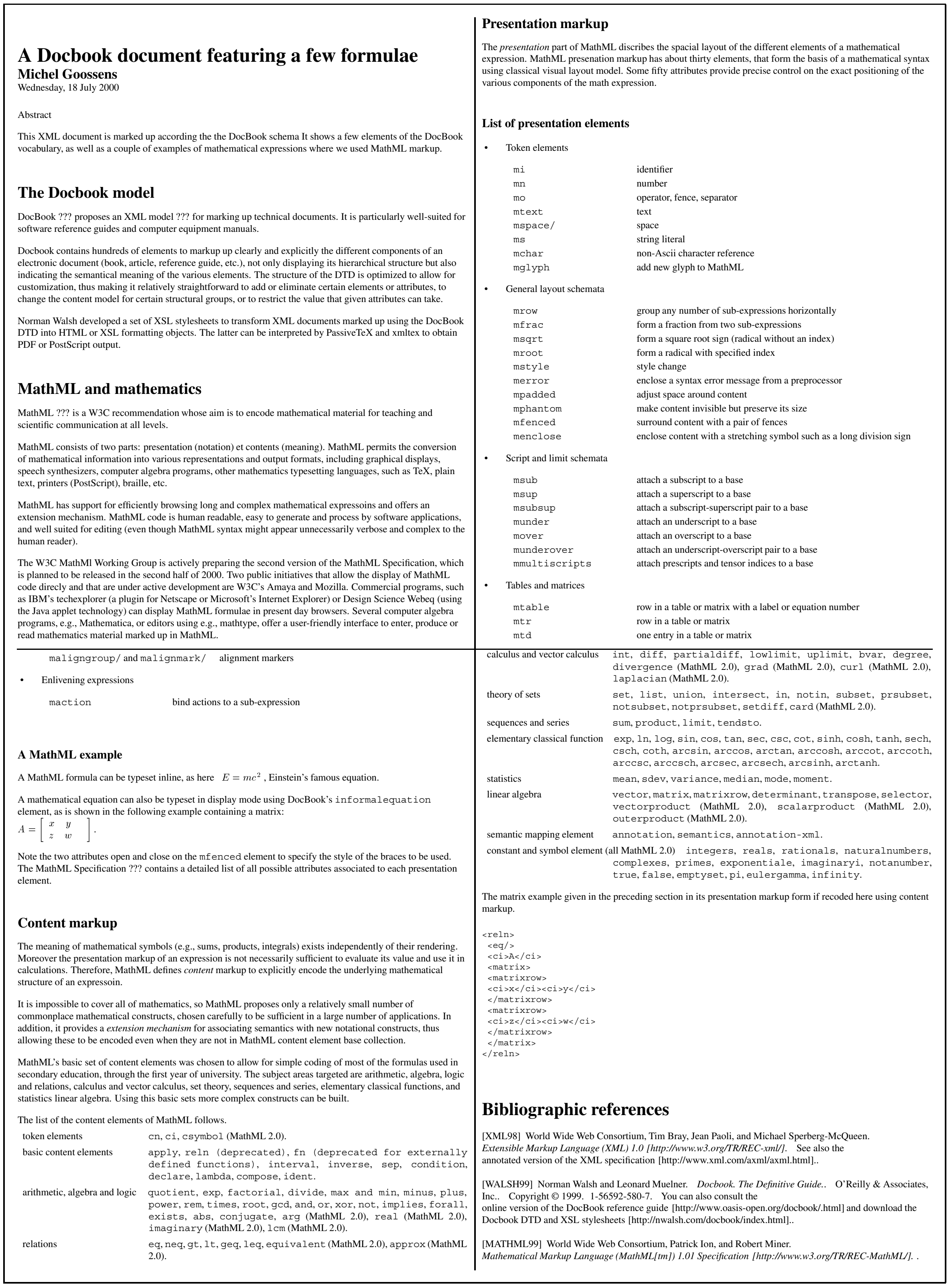
```
xt dbxmml.xml foplus.xsl fotex.xml
latex fotex.tex
```

The first line transforms the DocBook markup into XSL formatting objects, that are then handled (as well as passed through MathML markup) by PassiveTEX (and `xmltex`) referenced on the second line. Figure 4 shows the result of typesetting the complete DocBook example document with PassiveTEX.

XML and PassiveTEX at the heart of the Internet

Different ways of using an XML document in the context of an electronic document repository are shown in Figure 5. At the top right we represent the XML document with its defining vocabulary (DTD or XML Schema [18], [19] and [20]). This document, which is encoded in Unicode, can be viewed, searched, indexed, edited, validated without problems by any of a series of XML-aware applications anywhere on the Internet. The XML document can be typeset using TEX or its Unicode-aware variant Omega [8]. Three methods are shown: (A) uses direct interpretation by `xmltex`, an example of which was shown in Figure 2, (B) uses XSL formatting objects and PassiveTEX, as described in the present article, (C) transforms the XML source into a LATEX file. This procedure was used for typesetting the present article so that the resulting LATEX document could be included in proceedings of TUG 2000 Conference.

Alternatively the XML source can be transformed into HTML for viewing with present-day browsers (X2H via XSLT, see [7] for a detailed discussion). In the (near) future, once browsers can handle XML directly, we can probably skip the HTML intermediate format and let CSS [16] (possibly via XSLT) style the XML file directly for display on the Web. Figure 5 also contains arrows going from left (TEX) to right (XML/HTML,



A Docbook document featuring a few formulae

Michel Goossens
Wednesday, 18 July 2000

Abstract

This XML document is marked up according the the DocBook schema It shows a few elements of the DocBook vocabulary, as well as a couple of examples of mathematical expressions where we used MathML markup.

The Docbook model

DocBook ??? proposes an XML model ??? for marking up technical documents. It is particularly well-suited for software reference guides and computer equipment manuals.

Docbook contains hundreds of elements to markup up clearly and explicitly the different components of an electronic document (book, article, reference guide, etc.), not only displaying its hierarchical structure but also indicating the semantical meaning of the various elements. The structure of the DTD is optimized to allow for customization, thus making it relatively straightforward to add or eliminate certain elements or attributes, to change the content model for certain structural groups, or to restrict the value that given attributes can take.

Norman Walsh developed a set of XSL stylesheets to transform XML documents marked up using the DocBook DTD into HTML or XSL formatting objects. The latter can be interpreted by PassiveTeX and xmltex to obtain PDF or PostScript output.

MathML and mathematics

MathML ??? is a W3C recommendation whose aim is to encode mathematical material for teaching and scientific communication at all levels.

MathML consists of two parts: presentation (notation) et contents (meaning). MathML permits the conversion of mathematical information into various representations and output formats, including graphical displays, speech synthesizers, computer algebra programs, other mathematics typesetting languages, such as TeX, plain text, printers (PostScript), braille, etc.

MathML has support for efficiently browsing long and complex mathematical expressoins and offers an extension mechanism. MathML code is human readable, easy to generate and process by software applications, and well suited for editing (even though MathML syntax might appear unnecessarily verbose and complex to the human reader).

The W3C MathMl Working Group is actively preparing the second version of the MathML Specification, which is planned to be released in the second half of 2000. Two public initiatives that allow the display of MathML code direcly and that are under active development are W3C's Amaya and Mozilla. Commercial programs, such as IBM's techexplorer (a plugin for Netscape or Microsoft's Internet Explorer) or Design Science Webeq (using the Java applet technology) can display MathML formulae in present day browsers. Several computer algebra programs, e.g., Mathematica, or editors using e.g., mathtype, offer a user-friendly interface to enter, produce or read mathematics material marked up in MathML.

Presentation markup

The *presentation* part of MathML discribes the spacial layout of the different elements of a mathematical expression. MathML presenation markup has about thirty elements, that form the basis of a mathematical syntax using classical visual layout model. Some fifty attributes provide precise control on the exact positioning of the various components of the math expression.

List of presentation elements

- Token elements

`mi`	identifier
`mn`	number
`mo`	operator, fence, separator
`mtext`	text
`mspace/`	space
`ms`	string literal
`mchar`	non-Ascii character reference
`mglyph`	add new glyph to MathML

- General layout schemata

`mrow`	group any number of sub-expressions horizontally
`mfrac`	form a fraction from two sub-expressions
`msqrt`	form a square root sign (radical without an index)
`mroot`	form a radical with specified index
`mstyle`	style change
`merror`	enclose a syntax error message from a preprocessor
`mpadded`	adjust space around content
`mphantom`	make content invisible but preserve its size
`mfenced`	surround content with a pair of fences
`menclose`	enclose content with a stretching symbol such as a long division sign

- Script and limit schemata

`msub`	attach a subscript to a base
`msup`	attach a superscript to a base
`msubsup`	attach a subscript-superscript pair to a base
`munder`	attach an underscript to a base
`mover`	attach an overscript to a base
`munderover`	attach an underscript-overscript pair to a base
`mmultiscripts`	attach prescripts and tensor indices to a base

- Tables and matrices

`mtable`	row in a table or matrix with a label or equation number
`mtr`	row in a table or matrix
`mtd`	one entry in a table or matrix
`maligngroup/` and `malignmark/`	alignment markers

- Enlivening expressions

`maction`	bind actions to a sub-expression

A MathML example

A MathML formula can be typeset inline, as here $E = mc^2$, Einstein's famous equation.

A mathematical equation can also be typeset in display mode using DocBook's `informalequation` element, as is shown in the following example containing a matrix:
$A = \begin{bmatrix} x & y \\ z & w \end{bmatrix}$.

Note the two attributes open and close on the `mfenced` element to specify the style of the braces to be used. The MathML Specification ??? contains a detailed list of all possible attributes associated to each presentation element.

Content markup

The meaning of mathematical symbols (e.g., sums, products, integrals) exists independently of their rendering. Moreover the presentation markup of an expression is not necessarily sufficient to evaluate its value and use it in calculations. Therefore, MathML defines *content* markup to explicitly encode the underlying mathematical structure of an expressoin.

It is impossible to cover all of mathematics, so MathML proposes only a relatively small number of commonplace mathematical constructs, chosen carefully to be sufficient in a large number of applications. In addition, it provides a *extension mechanism* for associating semantics with new notational constructs, thus allowing these to be encoded even when they are not in MathML content element base collection.

MathML's basic set of content elements was chosen to allow for simple coding of most of the formulas used in secondary education, through the first year of university. The subject areas targeted are arithmetic, algebra, logic and relations, calculus and vector calculus, set theory, sequences and series, elementary classical functions, and statistics linear algebra. Using this basic sets more complex constructs can be built.

The list of the content elements of MathML follows.

token elements	`cn`, `ci`, `csymbol` (MathML 2.0).
basic content elements	`apply`, `reln (deprecated)`, `fn (deprecated for externally defined functions)`, `interval`, `inverse`, `sep`, `condition`, `declare`, `lambda`, `compose`, `ident`.
arithmetic, algebra and logic	`quotient`, `exp`, `factorial`, `divide`, `max and min`, `minus`, `plus`, `power`, `rem`, `times`, `root`, `gcd`, `and`, `or`, `xor`, `not`, `implies`, `forall`, `exists`, `abs`, `conjugate`, `arg` (MathML 2.0), `real` (MathML 2.0), `imaginary` (MathML 2.0), `lcm` (MathML 2.0).
relations	`eq`, `neq`, `gt`, `lt`, `geq`, `leq`, `equivalent` (MathML 2.0), `approx` (MathML 2.0).
calculus and vector calculus	`int`, `diff`, `partialdiff`, `lowlimit`, `uplimit`, `bvar`, `degree`, `divergence` (MathML 2.0), `grad` (MathML 2.0), `curl` (MathML 2.0), `laplacian` (MathML 2.0).
theory of sets	`set`, `list`, `union`, `intersect`, `in`, `notin`, `subset`, `prsubset`, `notsubset`, `notprsubset`, `setdiff`, `card` (MathML 2.0).
sequences and series	`sum`, `product`, `limit`, `tendsto`.
elementary classical function	`exp`, `ln`, `log`, `sin`, `cos`, `tan`, `sec`, `csc`, `cot`, `sinh`, `cosh`, `tanh`, `sech`, `csch`, `coth`, `arcsin`, `arccos`, `arctan`, `arccosh`, `arccot`, `arccoth`, `arccsc`, `arccsch`, `arcsec`, `arcsech`, `arcsinh`, `arctanh`.
statistics	`mean`, `sdev`, `variance`, `median`, `mode`, `moment`.
linear algebra	`vector`, `matrix`, `matrixrow`, `determinant`, `transpose`, `selector`, `vectorproduct` (MathML 2.0), `scalarproduct` (MathML 2.0), `outerproduct` (MathML 2.0).
semantic mapping element	`annotation`, `semantics`, `annotation-xml`.
constant and symbol element (all MathML 2.0)	`integers`, `reals`, `rationals`, `naturalnumbers`, `complexes`, `primes`, `exponentiale`, `imaginaryi`, `notanumber`, `true`, `false`, `emptyset`, `pi`, `eulergamma`, `infinity`.

The matrix example given in the preceding section in its presentation markup form if recoded here using content markup.

```
<reln>
 <eq/>
 <ci>A</ci>
 <matrix>
 <matrixrow>
 <ci>x</ci><ci>y</ci>
 </matrixrow>
 <matrixrow>
 <ci>z</ci><ci>w</ci>
 </matrixrow>
 </matrix>
</reln>
```

Bibliographic references

[XML98] World Wide Web Consortium, Tim Bray, Jean Paoli, and Michael Sperberg-McQueen. *Extensible Markup Language (XML) 1.0 [http://www.w3.org/TR/REC-xml/].* See also the annotated version of the XML specification [http://www.xml.com/axml/axml.html]..

[WALSH99] Norman Walsh and Leonard Muelner. *Docbook. The Definitive Guide.*. O'Reilly & Associates, Inc.. Copyright © 1999. 1-56592-580-7. You can also consult the online version of the DocBook reference guide [http://www.oasis-open.org/docbook/.html] and download the Docbook DTD and XSL stylesheets [http://nwalsh.com/docbook/index.html]..

[MATHML99] World Wide Web Consortium, Patrick Ion, and Robert Miner. *Mathematical Markup Language (MathML[tm]) 1.01 Specification [http://www.w3.org/TR/REC-MathML/].* .

Figure 4: Typeset result of an article marked up according to the DocBook schema

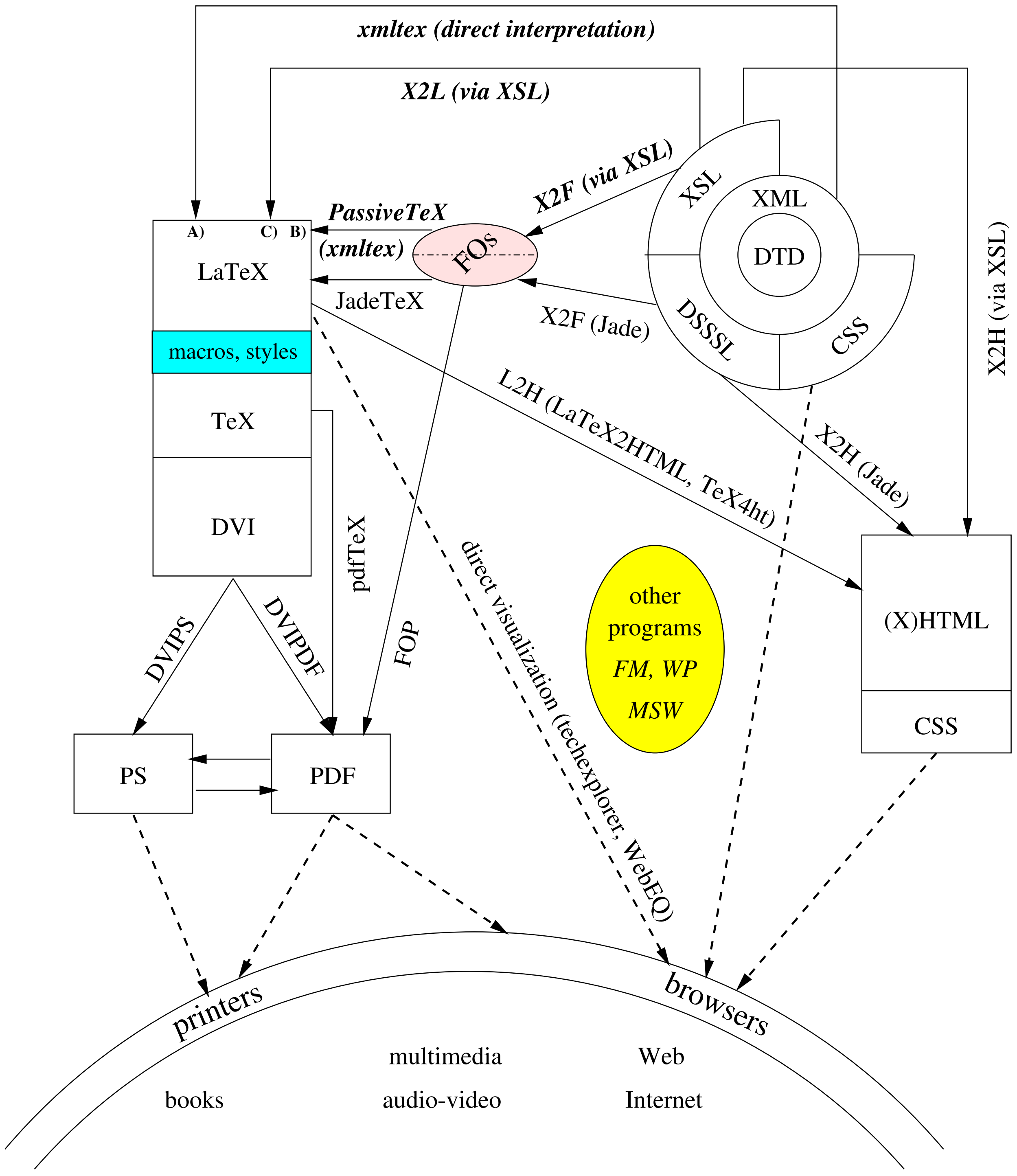


Figure 5: XML as the central part of a document strategy for the Web

browsers). They indicate programs to transform existing LATEX source documents into XML (using one or more standard DTDs) to store the information for archiving purposes. The vertical ellipse in the centre represents other editing tools, such as Adobe's *FrameMaker* [1] and Corel's *WordPerfect* [6], that allow or are expected to allow import/export of XML documents. Thus, XML genuinely becomes the central element in a global strategy for managing electronic documents by allowing information to be stored, saved, shared, and used by different applications on all computer platforms. PassiveTEX can be truly considered as a much-needed complement to XML for bringing typographic excellence to the Web, just as Knuth with TEX introduced the same excellence to the emerging electronic printing industry a generation ago.

References

[1] Adobe. *FrameMaker 6.0.* `http://www.adobe.com/products/framemaker`.

[2] Apache XML Project. *FOP, XSL Formatting Object Processor in Java.* `http://xml.apache.org/fop/`

[3] Lou Burnard and C.M. Sperberg-McQueen. TEI Guidelines for Electronic Text Encoding and Interchange. `http://etext.lib.virginia.edu/tei.html`

[4] David Carlisle. *`xmltex` A non validating (and not 100% conforming) namespace aware XML parser implemented in TEX.* Can be downloaded from CTAN in the directory `macros/xmltex/`.

[5] James Clark. *xt, an implementation in Java of XSL Transformations.* `http://www.jclark.com/xml/xt.html`

[6] Corel. *WordPerfect Office 2000.* `http://www.corel.com/Office2000` (Microsoft Windows) and `http://linux.corel.com/products/wpo2000_linux` (Linux).

[7] Michel Goossens and Sebastian Rahtz. *The LATEX Web Companion.* Addison-Wesley, Reading, 1999.

[8] Yannis Haralambous and John Plaice. The latest developments in Omega. *TUGBoat*, 17 (2), pages 181-183, June 1996. (See also `http://www.gutenberg.eu.org/omega/`).

[9] John Hobby. *A user's manual for MetaPost.* Computer Science Technical Report 162, AT&T Bell Laboratories, 1992.

[10] International Organization for Standardization. *Information Technology—Processing Languages—Document Style Semantics and Specification Language (DSSSL). First edition, 1996* International Standard ISO/IEC 10179:1996.

[11] Sebastian Rahtz. *Passive TEX* `http://users.ox.ac.uk/~rahtz/passivetex/`

[12] Sebastian Rahtz. *XSL stylesheets for TEI XML.* `http://users.ox.ac.uk/~rahtz/tei/`

[13] The Unicode Consortium. *The Unicode Standard, Version 3.0.* Addison-Wesley, Reading, 2000.

[14] Norman Walsh and Leonard Muelner. *DocBook. The Definitive Guide.* O'Reilly & Associates, Inc., Sebastopol, USA, 1999. See also `http://nwalsh.com/docbook/index.html`.

[15] Norman Walsh. *XSL DocBook Stylesheets.* `http://nwalsh.com/docbook/xsl/index.html`

[16] World Wide Web Consortium. Håkon Wium Lie, Bert Bos, Chris Lilley and Ian Jacobs (editors). *Cascading Style Sheets, level 2.* `http://www.w3.org/TR/REC-CSS2`.

[17] World Wide Web Consortium. Patrick Ion and Robert Miner (editors). *Mathematical Markup Language (MathML[tm]) 1.01 Specification.* `http://www.w3.org/TR/REC-MathML/`.

[18] World Wide Web Consortium, David C. Fallside (editor). *XML Schema Part 0: Primer (W3C Working Draft).* `http://www.w3.org/TR/xmlschema-0`.

[19] World Wide Web Consortium, Henry S. Thompson, David Beech, Murray Maloney, Noah Mendelsohn (editors). *XML Schema Part 1: Structures (W3C Working Draft).* `http://www.w3.org/TR/xmlschema-1`.

[20] World Wide Web Consortium, Paul V. Biron, Ashok Malhotra (editors). *XML Schema Part 2: Datatypes (W3C Working Draft).* `http://www.w3.org/TR/xmlschema-2`.

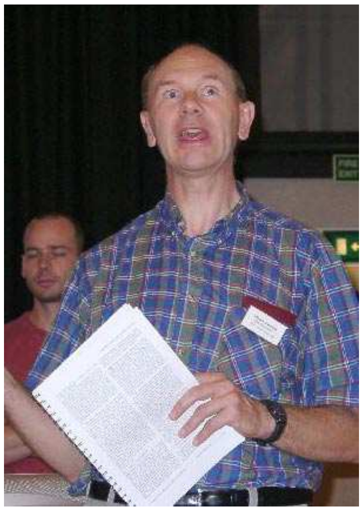
[21] World Wide Web Consortium, Jon Ferraiolo (editor). *Scalable Vector Graphics (SVG) 1.0 Specification (W3C Working Draft).* `http://www.w3.org/TR/SVG`.

[22] World Wide Web Consortium. Tim Bray, Jean Paoli, and C. M. Sperberg-McQueen

(editors). *Extensible Markup Language (XML) 1.0.* `http://www.w3.org/TR/REC-xml`. An annotated version of the specification is at `http://www.xml.com/axml/axml.html`.

[23] World Wide Web Consortium, James Clark (editor). *XSL Transformations (XSLT), Version 1.0 (W3C Recommendation 16 November 1999).* `http://www.w3.org/TR/xslt`.

[24] World Wide Web Consortium, Stephen Deach (editor). *Extensible Stylesheet Language (XSL), Version 1.0 (W3C Working Draft).* `http://www.w3.org/TR/WD-xsl`.



Michel Goossens

Sebastian Rahtz