The TEI/TEX Interface

Sebastian Rahtz

March 2005

Abstract

In the view of many people, the natural way to prepare a typeset document is to use LATEX or ConTEXt. It produces high-quality output, the source document is a clean mixture of text and markup, and it works on any computer. For another group of people, however, the natural way to proceed is to prepare a validated XML document which can be used to either make a web page or to make a printed document. One choice of an XML encoding for this group is the Text Encoding Initiative (TEI) scheme.

This paper is in two parts. The first part examines the arguments for and against authoring in XML, rather than TEX, and demonstrates how some common TEX situations are catered for in TEI XML.

The second part of the paper examines how, if we *do* choose XML, we can continue to harness the power of TEX. We examine the four main routes of

a) using a modified TEX to read XML directly;
b) translating XML direct to high-level TEX;
c) translating our XML to another XML which is functionally identical to LATEX and then translating that; and
d) translating XML to an XML-based page description language (XSL FO), and processing that with TEX.

None of these is completely satisfactory, and we end by considering what hope there is for the future.

1 The TEI / TEX interface

musings and reports

2 Personal background

I am Sebastian Rahtz:

- Information Manager for *Oxford University Computing* Services
- Manager of *OSS Watch*, the UK national Open Source Advisory Service
- Oxford representative on the Board of Directors of the *Text Encoding Initiative Consortium*; member of the TEI Technical Council, and convenor of its Meta Language working party
- Long-time (coming up to 20 years) TEX sorcerer (using the classification of Ursula Le Guin, not Don Knuth)
- Overall editor of TEXlive

3 TEI Background

The TEI

- an international and interdisciplinary standard that helps libraries, museums, publishers, and individual scholars represent all kinds of literary and linguistic texts for online research and teaching
- a comprehensive and well-documented markup language for all kinds of text material, from manuscripts to dictionaries, from film scripts to web pages.
- An XML vocabulary, coming up to a new release (P5) using XML schema languages

4 Example, part 1

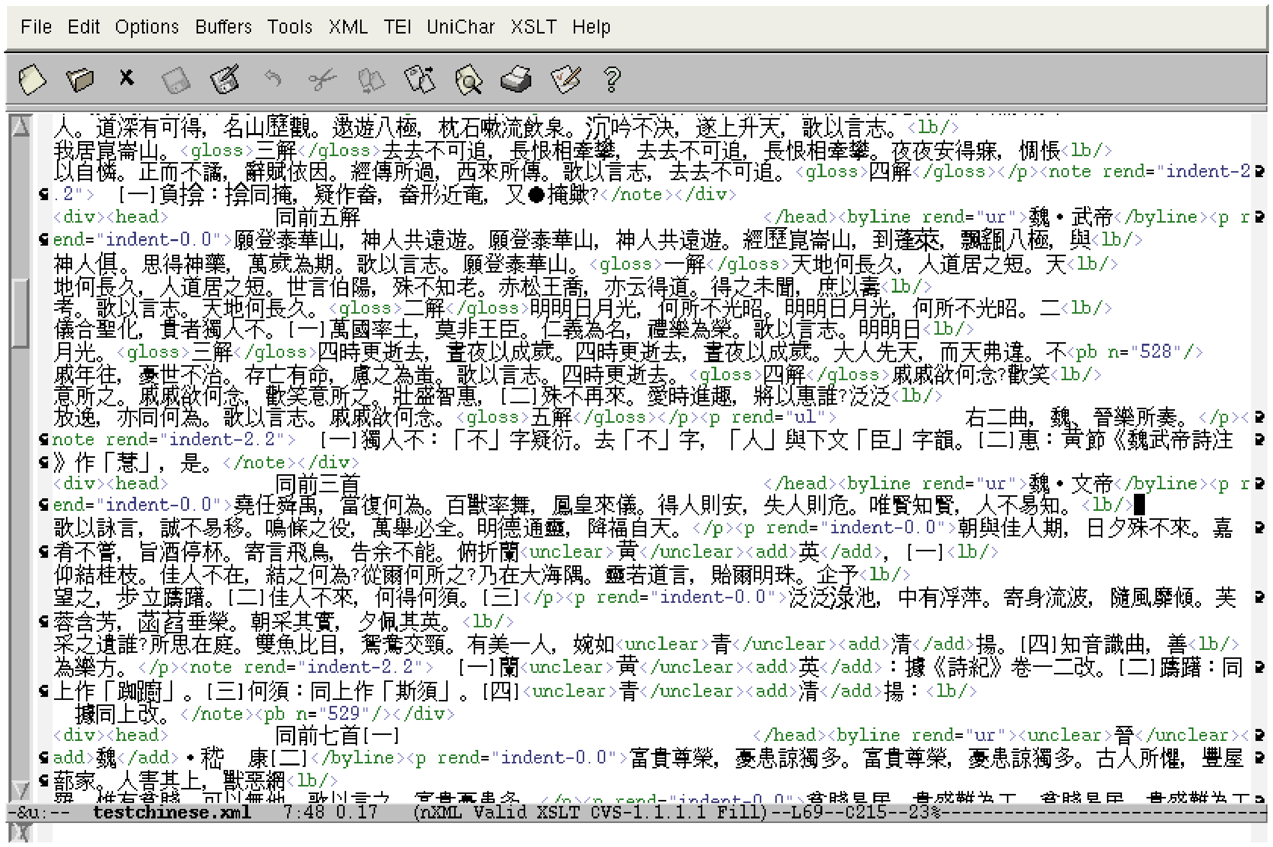
```
<TEI xmlns="http://www.tei-c.org/ns/1.0">
<teiHeader>
  <fileDesc>
    <titleStmt>
      <title>The TEI/TeX interface</title>
      <author>Sebastian Rahtz</author>
    </titleStmt>
    <editionStmt>
     <edition>
       <date>March 2005</date>
     </edition>
    </editionStmt>
  </fileDesc>
  <revisionDesc>
    <change>
      <date>$Date: 2005/03/10 $.</date>
       <respStmt>
          <name>$Author: rahtz $</name>
       </respStmt>
       <item>$Revision: #1 $</item>
    </change>
  </revisionDesc>
</teiHeader>
```

5 Example, part 2

```
<text>
<body>
<div>
<head>Personal background</head>
<p>I am <hi>Sebastian Rahtz</hi>:
<list>
   <item>
      Information Manager for<emph>Oxford University Computing</emph>Services
   </item>
   <item>
      Manager of <emph>OSS Watch</emph>, the UK national Open Source Advisory Service
   </item>
   <item>
      Oxford representative on the Board of Directors of the
      <emph>Text Encoding Initiative Consortium</emph>
```

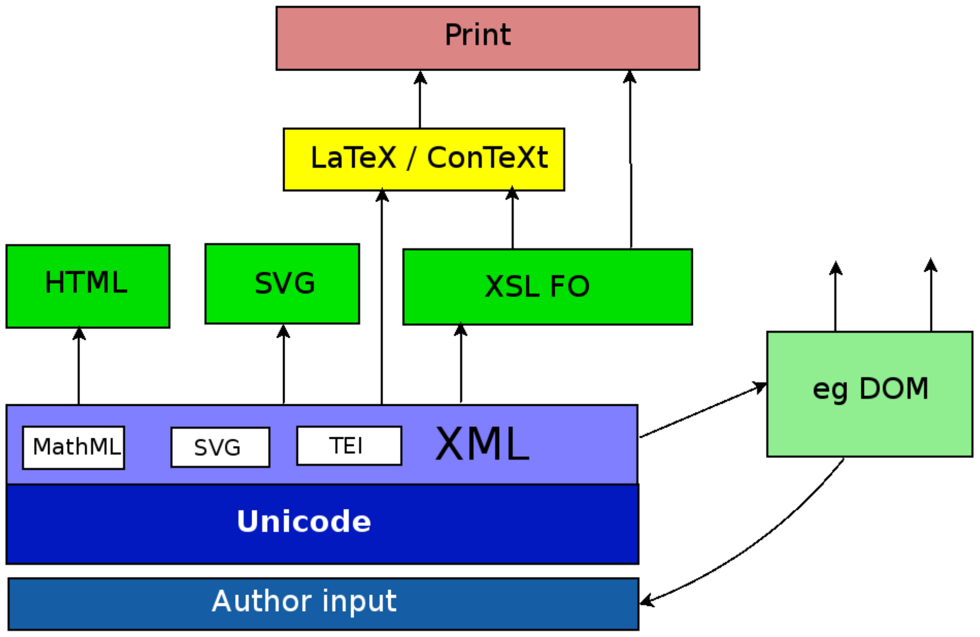
```
    </item>
    <item>
       Member of the TEI Technical Council, and convenor of its Meta Language
       working party
    </item>
    <item>
       Long-time (coming up to 20 years) TeX sorcerer
       <note>Using the classification of Ursula Le Guin,
             not Don Knuth or J K Rowling
       </note>
    </item>
 </list>
 <ref target="mailto:sebastian.rahtz@oucs.ox.ac.uk">sebastian.rahtz@oucs.ox.ac.uk</ref>
 </p>
 </div>
```

6 Example 3

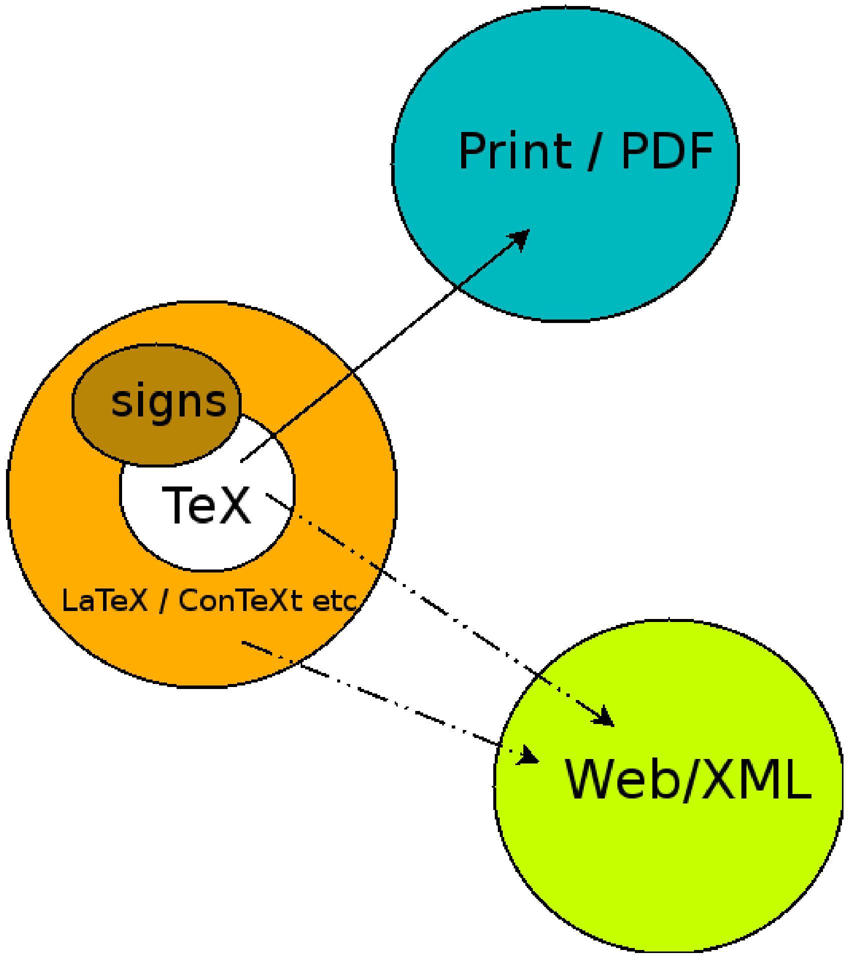


```
人。道深有可得，名山歷觀。遨遊八極，枕石嗽流飲泉。沉吟不決，遂上升天，歌以言志。<lb/>
我居崑崙山。<gloss>三解</gloss>去去不可追，長恨相牽攀，去去不可追，長恨相牽攀。夜夜安得寐，惆悵<lb/>
以自憐。正而不譎，辭賦依因。經傳所過，西來所傳。歌以言志，去去不可追。<gloss>四解</gloss></p><note rend="indent-2
.2">  [一]負揜：揜同掩，疑作畚，畚形近奄，又●掩歟?</note></div>
<div><head>          同前五解                    </head><byline rend="ur">魏・武帝</byline><p r
end="indent-0.0">願登泰華山，神人共遠遊。願登泰華山，神人共遠遊。經歷崑崙山，到蓬萊，飄颻八極，與<lb/>
神人俱。思得神藥，萬歲為期。歌以言志。願登泰華山。<gloss>一解</gloss>天地何長久，人道居之短。天<lb/>
地何長久，人道居之短。世言伯陽，殊不知老。赤松王喬，亦云得道。得之未聞，庶以壽<lb/>
考。歌以言志。天地何長久。<gloss>二解</gloss>明明日月光，何所不光昭。明明日月光，何所不光昭。二<lb/>
儀合聖化，貴者獨人不。[一]萬國率土，莫非王臣。仁義為名，禮樂為榮。歌以言志。明明日<lb/>
月光。<gloss>三解</gloss>四時更逝去，晝夜以成歲。四時更逝去，晝夜以成歲。大人先天，而天弗違。不<pb n="528"/>
戚年往，憂世不治。存亡有命，慮之為蚩。歌以言志。四時更逝去。<gloss>四解</gloss>戚戚欲何念?歡笑<lb/>
意所之。戚戚欲何念，歡笑意所之。壯盛智惠，[二]殊不再來。愛時進趣，將以惠誰?泛泛<lb/>
放逸，亦同何為。歌以言志。戚戚欲何念。<gloss>五解</gloss></p><p rend="ul">          右二曲，魏、晉樂所奏。</p><
note rend="indent-2.2">  [一]獨人不：「不」字疑衍。去「不」字，「人」與下文「臣」字韻。[二]惠：黃節《魏武帝詩注
》作「慧」，是。</note></div>
<div><head>          同前三首                    </head><byline rend="ur">魏・文帝</byline><p r
end="indent-0.0">堯任舜禹，當復何為。百獸率舞，鳳皇來儀。得人則安，失人則危。唯賢知賢，人不易知。<lb/>
歌以詠言，誠不易移。鳴條之役，萬舉必全。明德通靈，降福自天。</p><p rend="indent-0.0">朝與佳人期，日夕殊不來。嘉
肴不嘗，旨酒停杯。寄言飛鳥，告余不能。俯折蘭<unclear>黃</unclear><add>英</add>，[一]<lb/>
仰結桂枝。佳人不在，結之何為?從爾何所之?乃在大海隅。靈若道言，貽爾明珠。企予<lb/>
望之，步立躊躇。[二]佳人不來，何得何須。[三]</p><p rend="indent-0.0">泛泛淥池，中有浮萍。寄身流波，隨風靡傾。芙
蓉含芳，菡萏垂榮。朝采其實，夕佩其英。<lb/>
采之遺誰?所思在庭。雙魚比目，鴛鴦交頸。有美一人，婉如<unclear>青</unclear><add>清</add>揚。[四]知音識曲，善<lb/>
為樂方。</p><note rend="indent-2.2">  [一]蘭<unclear>黃</unclear><add>英</add>：據《詩紀》卷一二改。[二]躊躇：同
上作「踟躕」。[三]何須：同上作「斯須」。[四]<unclear>青</unclear><add>清</add>揚：<lb/>
  據同上改。</note><pb n="529"/></div>
<div><head>          同前七首[一]                </head><byline rend="ur"><unclear>晉</unclear><
add>魏</add>・嵇  康[二]</byline><p rend="indent-0.0">富貴尊榮，憂患諒獨多。富貴尊榮，憂患諒獨多。古人所懼，豐屋
蔀家。人害其上，獸惡網<lb/>
-&u:--  testchinese.xml    7:48 0.17    (nXML Valid XSLT CVS-1.1.1.1 Fill)--L69--C215--23%-----------------------------
```

7 The TEI world view



8 The LATEX world view



9 Core differences between TEI XML and (eg) LATEX

Markup using (most of) Unicode	Markup using ASCII (extensible with difficulty)
Verbose but consistent	Concise but arbitrary
International standard for markup	Private extensible markup
Unicode character encoding	Variable character encoding
Single syntax	Syntax determined by application
Vocabulary choice constrained by schema	Vocabulary dynamically extensible and changeable
Vocabulary checkable	Vocabulary only constrained by syntax
Language separate from processing	Processor and language intermixed
—	Builtin math engine
—	Builtin tabular engine
Multiple processors	One reliable processor

10 Why markup schemas?

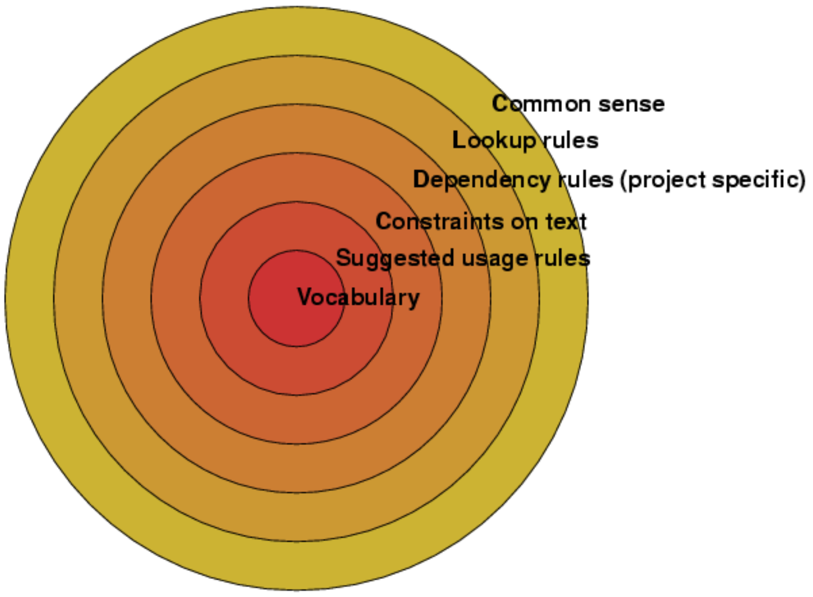
So we want machine readable text:

- We need a reasonable notation (XML or TEX)
- We need a character encoding system (Unicode)
- We need to make up vocabularies
- We need to be able to process our texts

What influences our choice? We

- want to interchange texts and tools with others
- need to have a formal way to express conditions about our markup
- should find a place to document our vocabulary

11 What we might do with a schema



12 Layers, using an XML schema

1. vocabulary

```
<list>, <item>, <label>
```

2. suggested usage rules

```
element list { item+ }
```

3. constraints on text

```
figure.attributes.url.content = xsd:anyURI
```

4. dependency rules (project specific)

```
<if test="self::list[@type='gloss'] and not(child::label)">
    <message>gloss lists must have <label> children</message>
</if>
```

5. lookup rules

```
<if test="document('lookup.xml')/people/person[@id=current::@ref]">
    <message>this person does not exist in the database</message>
</if>
```

6. common sense rules

```
Don't use table markup to force layout
```

13 Does the TEI cover all these?

As of today, the TEI Guidelines contain:

vocabulary 362 elements, 95 attributes, 88 classes

suggested usage rules 24 modules with 7185 lines of rules in compact Relax NG

constraints on text W3C Schema datatyping

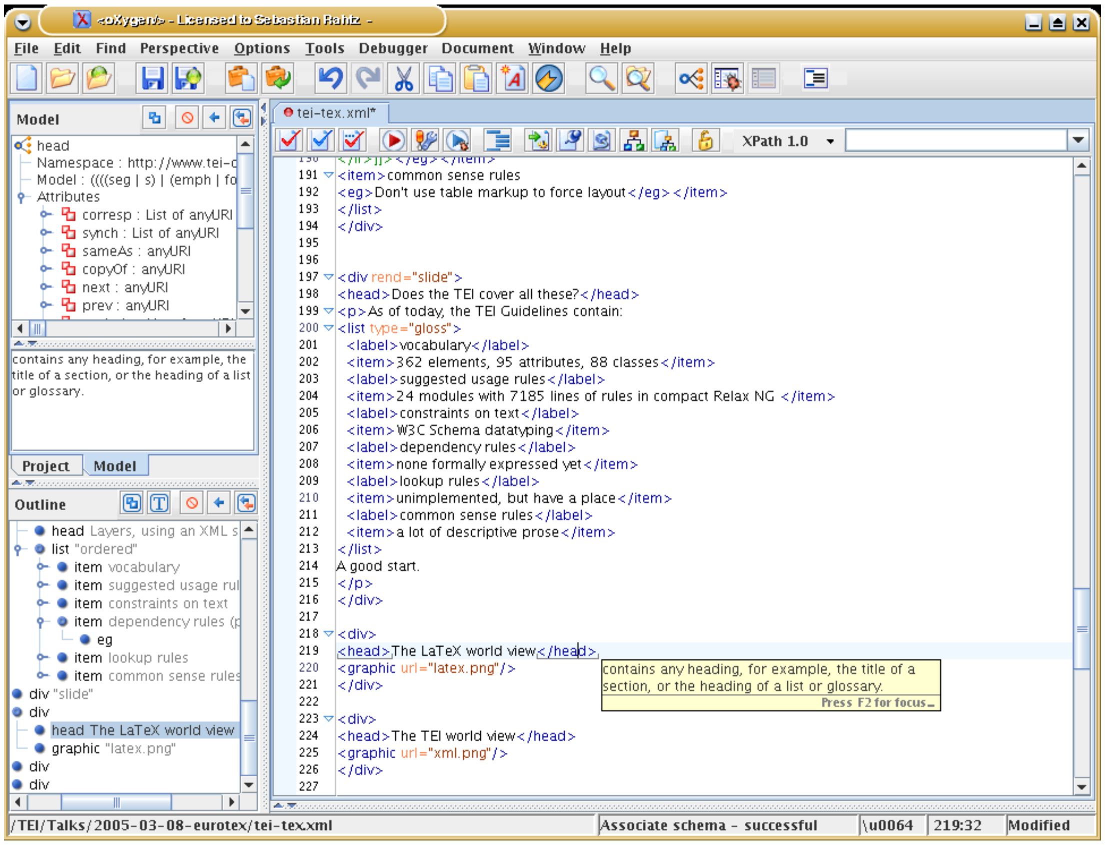
dependency rules none formally expressed yet

lookup rules unimplemented, but have a place

common sense rules a lot of descriptive prose

A good start.

14 An editors view



<oXygen/> - Licensed to Sebastian Rahtz -
File Edit Find Perspective Options Tools Debugger Document Window Help
Model
Project Model
Outline
tei-tex.xml*
XPath 1.0
/TEI/Talks/2005-03-08-eurotex/tei-tex.xml
Associate schema - successful
\u0064
219:32
Modified

Model

```
head
  Namespace : http://www.tei-c
  Model : ((((seg | s) | (emph | fo
  Attributes
    corresp : List of anyURI
    synch : List of anyURI
    sameAs : anyURI
    copyOf : anyURI
    next : anyURI
    prev : anyURI
```

contains any heading, for example, the title of a section, or the heading of a list or glossary.

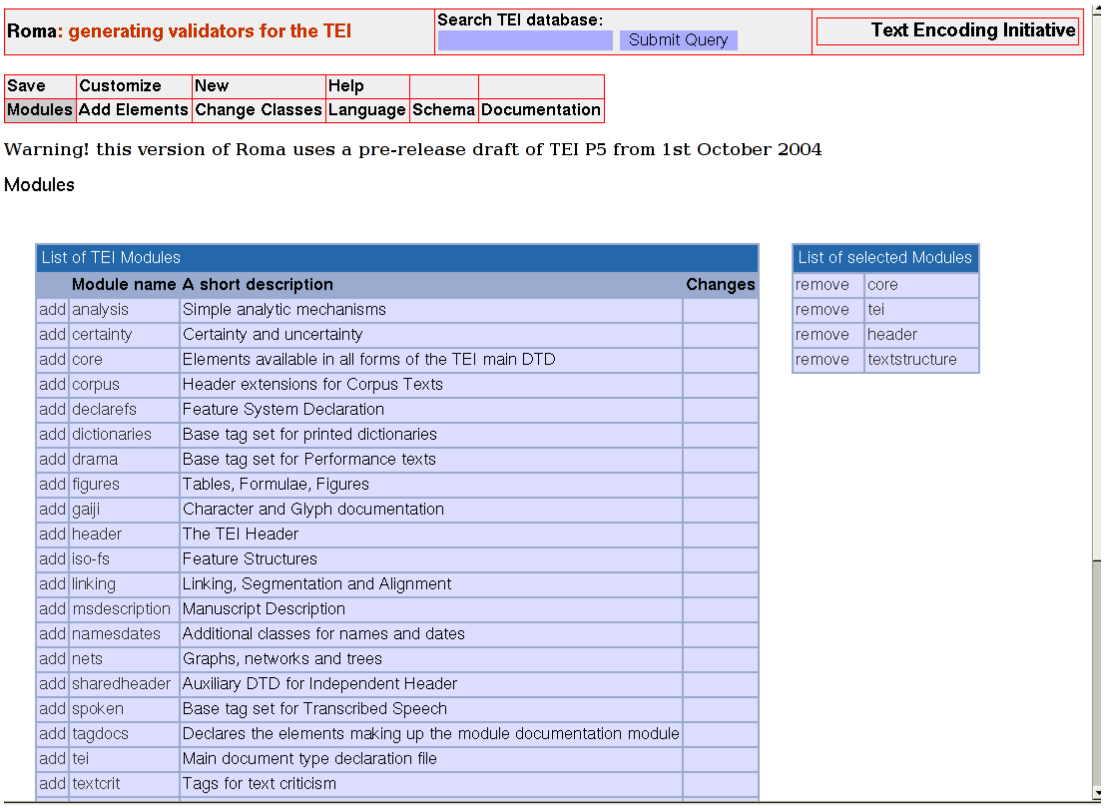
Outline

```
  head Layers, using an XML s
  list "ordered"
    item vocabulary
    item suggested usage rul
    item constraints on text
    item dependency rules (p
      eg
    item lookup rules
    item common sense rules
div "slide"
div
  head The LaTeX world view
  graphic "latex.png"
div
div
```

```
<item>common sense rules
<eg>Don't use table markup to force layout</eg></item>
</list>
</div>


<div rend="slide">
<head>Does the TEI cover all these?</head>
<p>As of today, the TEI Guidelines contain:
<list type="gloss">
  <label>vocabulary</label>
  <item>362 elements, 95 attributes, 88 classes</item>
  <label>suggested usage rules</label>
  <item>24 modules with 7185 lines of rules in compact Relax NG </item>
  <label>constraints on text</label>
  <item>W3C Schema datatyping</item>
  <label>dependency rules</label>
  <item>none formally expressed yet</item>
  <label>lookup rules</label>
  <item>unimplemented, but have a place</item>
  <label>common sense rules</label>
  <item>a lot of descriptive prose</item>
</list>
A good start.
</p>
</div>

<div>
<head>The LaTeX world view</head>
<graphic url="latex.png"/>
</div>

<div>
<head>The TEI world view</head>
<graphic url="xml.png"/>
</div>
```

contains any heading, for example, the title of a section, or the heading of a list or glossary.
Press F2 for focus

15 Example 3



Roma: generating validators for the TEI

Save	Customize	New	Help		
Modules	Add Elements	Change Classes	Language	Schema	Documentation

Warning! this version of Roma uses a pre-release draft of TEI P5 from 1st October 2004

Modules

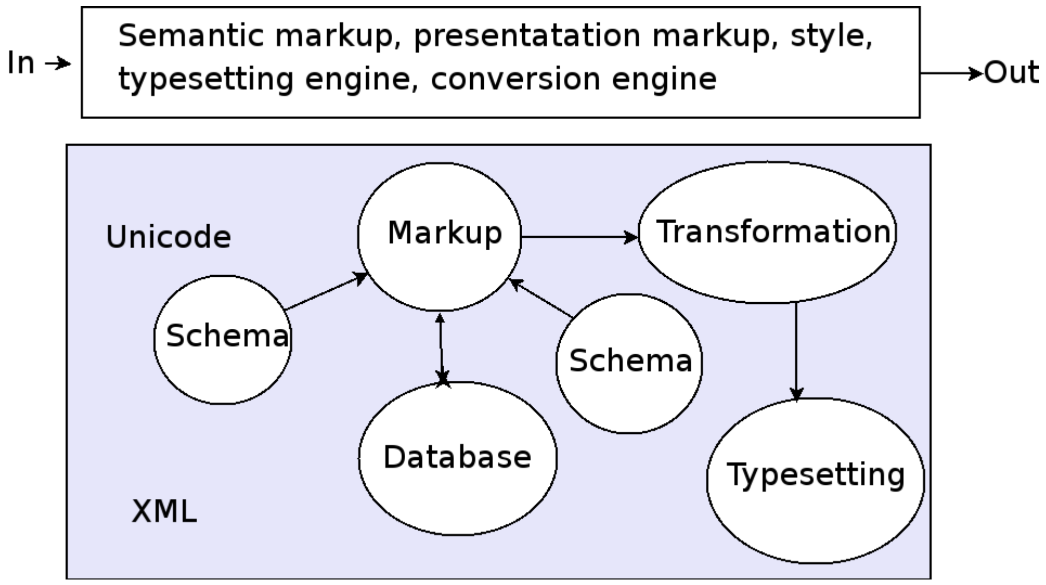
List of TEI Modules

	Module name	A short description	Changes
add	analysis	Simple analytic mechanisms	
add	certainty	Certainty and uncertainty	
add	core	Elements available in all forms of the TEI main DTD	
add	corpus	Header extensions for Corpus Texts	
add	declarefs	Feature System Declaration	
add	dictionaries	Base tag set for printed dictionaries	
add	drama	Base tag set for Performance texts	
add	figures	Tables, Formulae, Figures	
add	gaiji	Character and Glyph documentation	
add	header	The TEI Header	
add	iso-fs	Feature Structures	
add	linking	Linking, Segmentation and Alignment	
add	msdescription	Manuscript Description	
add	namesdates	Additional classes for names and dates	
add	nets	Graphs, networks and trees	
add	sharedheader	Auxiliary DTD for Independent Header	
add	spoken	Base tag set for Transcribed Speech	
add	tagdocs	Declares the elements making up the module documentation module	
add	tei	Main document type declaration file	
add	textcrit	Tags for text criticism	

List of selected Modules

remove	core
remove	tei
remove	header
remove	textstructure

16 Another graphical view of TEI and TEX


In
Semantic markup, presentatation markup, style, typesetting engine, conversion engine
Out
Unicode
Markup
Transformation
Schema
Schema
Database
Typesetting
XML

17 Using TEX behind XML

1. using a modified TEX to read XML directly;
2. translating XML direct to high-level TEX;
3. translating our XML to another XML which is functionally identical to LATEX and then translating that; and
4. translating XML to an XML-based page description language (XSL FO), and processing that with TEX (XSLFO).

17.1 [1] TEX reads XML directly: ConTEXt

Using mapping files:

```
\defineXMLenvironment[article][id=\undefined]
   {\XMLDBpushelement\currentXMLelement
    \XMLDBmaystartdocument
    \XMLDBmayensurebodymatter}
   {\XMLDBmayfinishdocument
    \XMLDBpopelement}

\defineXMLenvironment[itemizedlist]
    {\XMLDBpushelement\currentXMLelement \XMLDBmayensurebodymatter
     \doifsamestringelse{\XMLpar{itemizedlist}{spacing}{normal}}{compact}
       {\startitemize[packed]}
       {\startitemize}%
     \defineXMLignore[titleabbrev]%
     \defineXMLenvironment[listitem]
        {\item\XMLDBcontinuepartrue\ignorespaces}{}%
    }
    {\stopitemize\XMLDBpopelement}
```

17.2 [1] TEX reads XML directly: xmlTEX

```
\XMLelement{TEI.2}{}
  { \documentclass{article}
     \usepackage[bookmarks=false]{hyperref}
     \usepackage{teixml}
     \begin{document}  }
  {\end{document}}
...
\XMLelement{ref}
 {\XMLattribute{target}{\reftarget}{}}
 {\xmlgrab}
 {\hyperref[\reftarget]{#1}}
```

(used for PassiveTEX XSL FO processor)

17.3 [1] TEX reads XML directly: problems

1. gobbledygook, even by TEX standards: only experts need apply
2. limited access to document tree
3. (xmlTEX) forced grouping makes mapping some constructs almost impossible
4. catcode issues in auxiliary files

17.4 [2] Translate XML to high-level LATEX

```
<xsl:template match="tei:list">
<xsl:choose>
 <xsl:when test="@type='gloss'">
   \begin{description}
        <xsl:apply-templates mode="gloss" select="tei:item"/>
   \end{description}
 </xsl:when>
 <xsl:when test="@type='unordered'">
   \begin{itemize}<xsl:apply-templates/>
   \end{itemize}
 </xsl:when>
 <xsl:when test="@type='ordered'">
   \begin{enumerate}<xsl:apply-templates/>
    \end{enumerate}
 </xsl:when>
 <xsl:otherwise>
   \begin{itemize}<xsl:apply-templates/>
   \end{itemize}
 </xsl:otherwise>
</xsl:choose>
</xsl:template>
```

17.4.1 (dirty details)

```
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{ucs}
```

```
\catcode`\_=12\relax
\let\tabcellsep&
\catcode`\&=12\relax
\catcode`\$=12\relax
\catcode`\^=12\relax
\catcode`\~=12\relax
\catcode`\#=12\relax
\catcode`\%=12\relax
```

17.5 [2] Translate XML to high-level LATEX: problems

1. Remaining catcode problems (\, {, })
2. When LATEX signals an error, where is it in the source?
3. Where do you make style decisions?
   - `\tableofcontents` or `<divGen type="toc"/>`
   - `\section{Introduction}` or `\section{1.  Introduction}`
   - `\def{xxxxx \def{yyyyy}` or `\usepackage{fooo}`

17.6 [3] Transform XML to XML-ised LATEX

Transform to

```
<cmd name="documentclass">
   <opt>12pt</opt>
   <parm>letter</parm>
 </cmd>
 <env name="document">
    <cmd name="author" nl2="1">
      <parm>A. U. Thor</parm>
    </cmd>
    <cmd name="title" nl2="1">
      <parm>A SHORT STORY</parm>
    </cmd>
    <cmd name="maketitle" nl2="1" gr="0"/>
    <cmd name="section*" nl2="1">
      <parm>A SHORT STORY</parm>
    </cmd>
```

and thence to

```
  \documentclass[12pt]{letter}
  \begin{document}
    \author{A. U. Thor}
    \title{A SHORT STORY}
```

17.7 [3] Transform XML to XML-ised LATEX: ups and downs

✔ You don't have to worry about \ { and }

✘ It takes another processor

✘ It inserts yet another layer of obscurity between author and error on printout

✔ Allows for implementation using the first technique

17.8 [4] Transform XML to XML-based page description language

```
<fo:list-block margin-right="10pt" space-before="6pt"
          space-after="6pt" margin-left="15pt">
  <fo:list-item space-before.optimum="4pt">
    <fo:list-item-label>
      <fo:block margin-right="2.5pt" text-align="center">
          &#x2219;
      </fo:block>
    </fo:list-item-label>
    <fo:list-item-body>
      <fo:block font-weight="normal">Marley's ghost1</fo:block>
    </fo:list-item-body>
  </fo:list-item>
  <fo:list-item space-before.optimum="4pt">
    <fo:list-item-label>
      <fo:block margin-right="2.5pt" text-align="center">
          &#x2219;
      </fo:block>
    </fo:list-item-label>
    <fo:list-item-body>
      <fo:block font-weight="normal">
          The first of the three spirits 39
      </fo:block>
    </fo:list-item-body>
  </fo:list-item>
...
```

17.9 [4] Transform XML to XML-based page description language (creation)

```
<xsl:template match="tei:list">
 <fo:list-block   margin-right="{$listRightMargin}">
  <xsl:call-template name="setListIndents"/>
  <xsl:choose>
   <xsl:when test="@type='gloss'">
    <xsl:attribute name="margin-left">
     <xsl:choose>
     <xsl:when test="ancestor::tei:list">
         <xsl:value-of select="$listLeftGlossInnerIndent"/>
     </xsl:when>
     <xsl:otherwise>
         <xsl:value-of select="$listLeftGlossIndent"/>
     </xsl:otherwise>
     </xsl:choose>
    </xsl:attribute>
  </xsl:when>
   <xsl:otherwise>
     <xsl:attribute name="margin-left">
         <xsl:value-of select="$listLeftIndent"/></xsl:attribute>
   </xsl:otherwise>
  </xsl:choose>
  <xsl:apply-templates select="tei:item"/>
  </fo:list-block>
</xsl:template>
```

18 Implementations of XSL FO

Open source

- PassiveTEX: 4000 lines of incomprehensible TEX macros by an amateur, incomplete and stalled
- FoXeT: 4000 lines of TEX macros by a professional, getting closer
- FOP: free-standing Java program, in the doldrums for several years

Closed source

- Antenna House: excellent full implementation, Windows only
- XEP: excellent full implementation in Java

19 FO's good and bad points

✔ Simple to read and write, although very verbose

✔ 'Standardised' by a reputable body

✔ Multiple implementations

✔ Understands colour, backgrounds, fonts, URLs, Unicode etc

✘ Divorced from the typesetter

✘ Simplistic and limited page model (eg floats)

possibly "good enough" (anathema to TEXxies!)

20 Which direction?

- Forget direct TEX interpretation of arbitrary XML...
- ...embrace direct TEX reading of constrained XML
- Forget trying to teach people `\{}`...
- ...embrace semantically clean markup
- Forget trying to make TEX the centre of the universe
- ...develop TEX to keep being the best typesetting **engine**