Implementing Concurrent Markup in XML

Patrick Durusau (pdurusau@emory.edu)
Society of Biblical Literature

Matthew Brook O'Donnell (m.odonnell@roehampton.ac.uk)
OpenText.org and University of Surrey Roehampton
Why Concurrent Hierarchies?

- Different Interpretations of Text
- Structures that do not “properly” nest in the XML sense
- Complex textual traditions with multiple witnesses and variants
- Recording physical layout of text and other analysis
Matthew 3:8 Bear fruit that befits repentance,

Matthew 3:9 and to not presume to say to yourselves, ‘We have Abraham as our father’; for I tell you, God is able from these stones to raise up children of Abraham.
Matthew 3:8-9 – First Choice

<verse id="Matt.3.8">
Bear fruit that befits repentance,
</verse>

<verse="Matt.3.9">
and to not presume to say to yourselves, ‘We have Abraham as our father’; for I tell you, God is able from these stones to raise up children of Abraham.
</verse>
Bear fruit that befits repentance, and to not presume to say to yourselves, ‘We have Abraham as our father’; for I tell you, God is able from these stones to raise up children of Abraham.
Matthew 3:8-9 – Verboten!

<verse id="Matt.3.8">
  <sentence>
  Bear fruit that befits repentance,
  </sentence>
</verse>

<verse="Matt.3.9">
  <sentence>
  and to not presume to say to yourselves, ‘We have Abraham as our father’; for I tell you, God is able from these stones to raise up children of Abraham.
  </sentence>
</verse>
Design Principles: Part 1

- Formal simplicity
- Capacity to represent all occurring or imaginable kinds of structures
- Suitability for formal or mechanical validation
- Clear identity with the notations needed for simpler cases
- Allow for conditional indexing and processing
Design Principles: Part 2

- Allow for extraction of well-formed subtrees and documents
- Allow for query of the position of the element between two or more hierarchies
- Use standard XML syntax and mechanisms
- Validation and processing must be possible with standard XML software
- Can be used with existing documents encoded in XML markup
Bottom Up Virtual Hierarchies

- Membership of PCDATA in a particular hierarchy
- Record that information using XPath syntax
- Gather information from multiple document instances into a base file
- Query membership in and across hierarchies with BUVH
A Simple Example (1)

• Four separate (overlapping) hierarchies

This is text$^a$

---

a texs A

in a$^b$ base file

---

b an C
A Simple Example (2)

1. Page view

```xml
<pages>
  <page id="p1">
    <line id="l1">This is</line>
    <line id="l2">text</line>
  </page>
  <page id="p2">
    <line id="l1">in a base</line>
    <line id="l2">file</line>
  </page>
</pages>
```
A Simple Example (3)

2. Text view

<text>
  <para id="p1">
    This is text in a base file
  </para>
</text>
A Simple Example (4)

3. Linguistic view

<clauses>
  <clause id="c1">
    <subject>This</subject>
    <predicate>is</predicate>
    <complement>text</complement>
    <adjunct>in a base file</adjunct>
  </clause>
</clauses>
A Simple Example (5)

4. Textual variant view (using out-of-line markup)

<variants xmlns:xlink="http://www.w3.org/1999/XLink">
  <app id="tv1">
    <rdg xlink:href="base.xml#id(w3)"
         wit="A" val="texs"/>
  </app>
  <app id="tv2">
    <rdg xlink:href="base.xml#id(w5)"
         wit="C" val="an"/>
  </app>
</variants>
A Simple Example (6)

To encode these hierarchies in a single file, one view must be selected as base hierarchy.

This is text in a base file.

Inconsistent nesting

Loss of parent-child relationship

This is text in a base file.
A Simple Example (7)

• BUVH Approach

1. Create common base file with divisions for
   Atomic PCDATA (here word divisions)

```xml
<baseFile>
  <w id="w1">This</w>
  <w id="w2">is</w>
  <w id="w3">text</w>
  <w id="w4">in</w>
  <w id="w5">a</w>
  <w id="w6">base</w>
  <w id="w7">file</w>
</baseFile>
```
A Simple Example (7)

• BUVH Approach

1. Create common base file with divisions for Atomic PCDATA (here word divisions)

2. For each Atomic PCDATA element:
   a. Locate in each hierarchy
   b. Construct XML Membership XPath Expression describing its position within the hierarchy
   c. Add Tree Structure Position Attribute for element’s position in hierarchy to element in base file
A Simple Example (7)

• BUVH Approach

c. Add TSP Attribute

<baseFile>
  <w id="w1">
    pg:pages="/pages/page[1][@id='p1']/line[1][@id='l1'/*[1]"
    tx:text="/text/para[1]/[@id='p1']/*[1]"
    sn:clauses="/clauses/clause[1][@id='c1']/subject[1]/*[1]">
      This
    </sn:clauses>
  </w>
</baseFile>

3. Linguistic hierarchy:

/clauses/clause[1][@id="c1"]/subject[1]/*[1]
This is text
A Simple Example (8)

```xml
<w id="w4"
    sn:clauses="/clauses/clause[1][@id='c1']/a[1]/*[1]"
    tx:text="/text/para[1][@id='p1']/*[4]"
    pg:pages="/pages/page[2][@id='p2']/line[1][@id='l1']/*[1]"
>in</w>

<w id="w5"
    sn:clauses="/clauses/clause[1][@id='c1']/a[1]/*[2]"
    tx:text="/text/para[1][@id='p1']/*[5]"
    pg:pages="/pages/page[2][@id='p2']/line[1][@id='l1']/*[2]"
    vr:variants="/variants/app[2][@id='tv2']/rdg[1][@wit='C'][@val='an']"
>a</w>

<w id="w6"
    sn:clauses="/clauses/clause[1][@id='c1']/a[1]/*[3]"
    tx:text="/text/para[1][@id='p1']/*[6]"
    pg:pages="/pages/page[2][@id='p2']/line[1][@id='l1']/*[3]"
>base</w>

<w id="w7"
    sn:clauses="/clauses/clause[1][@id='c1']/a[1]/*[4]"
    tx:text="/text/para[1][@id='p1']/*[7]"
    pg:pages="/pages/page[2][@id='p2']/line[2][@id='l2']/*[1]"
>file</w>

</baseFile>
```
Queries across different hierarchies can be carried out using XPath expressions, e.g. using XSLT

**Example 1:**

Locate words that have textual variants and are found on page 2

XPath query:

```
//w[@vr:variants][contains(@pg:pages,'p2')]
```

Result:

```xml
<w id="w5"
    sn:clauses="/clauses/clause[1][@id='c1']/a[1]/*[2]"
    tx:text="/text/para[1][@id='p1']/*[5]"
    pg:pages="/pages/page[1][@id='p2']/line[@id='l1']/*[2]"
    vr:variants="/variants/app[2][@id='tv2']/rdg[1][@wit='C'][@val='an']"
>
<a/>
```

A Simple Example (9)

• Queries across different hierarchies can be carried out using XPath expressions, e.g. using XSLT

Example 2:

• Locate words in the first clause that do not occur on the first line of their page

XPath query:

```
//w[contains(@sn:clauses,'clause[1]')]
not(contains(@pg:pages,'line[1]'))
```

Result:

```
<w id="w3"
    sn:clauses="/clauses/clause[1] [@id='c1']/c[1]/*[1]"
    tx:text="/text/para[1] [@id='p1']/*[3]"
><text>text</text></w>

<w id="w7"
    sn:clauses="/clauses/clause[1] [@id='c1']/a[1]/*[4]"
    tx:text="/text/para[1] [@id='p1']/*[7]"
><file>file</file></w>
```
Summary: BUVH Approach

- Authoring of XML occurs within a single hierarchy (any XML editor)
- Automatic construction of base file with any XSLT processor
- Query with any XSLT processor
- Unlimited hierarchies
Future Plans

• Development of XSLT Extensions to process BUVH Base File

• Base file format (possible use of Xalan’s DTM format?)

• Testing of BUVH against more complex examples

• Use of XLink with BUVH for read-only or large corpora
<partingThought>
Markup is metadata about #PCDATA
</partingThought>