Coming Down From The Trees: Future of the Evolution of Markup?

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

Matthew Brook O'Donnell (OpenText.org)
matt@opentext.org

<altTitle> Declaring Trees: Future of the Evolution of Markup? </altTitle>

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

Matthew Brook O'Donnell (OpenText.org)
matt@opentext.org

ACH/ALLC 2002

- Michael Sperberg-McQueen (The Markup Bear) was heard to say:

- Only You Can Stop the Deforestation of Texts!

Markup, Syntax and Trees

- One tree per document
  - More precisely, one root per document
  - Tree syntax expressed from that single root
  - All markup recognized from that root
- Causes problems for text encoding
  - Overlapping hierarchies
  - Non-nesting phenomena
  - Complex relationships

Prior Solutions I

- Bottom-Up Virtual Hierarchies (BUVH)
- Concur (cf. Sema Group implementation)
- Fragmentation
- Layered Markup and Annotation Language (LMNL)
- Milestones
- Multiple versions

Prior Solutions II

- Non-SGML/XML markup
- Standoff Markup
- Prolog database
- Virtual Joins

Success of Solutions Varies

- All workarounds for:
  - single root plus tree syntax
  - all markup recognized
- Lack of broad community experience
- Sensitive to editing (multiple versions, BUVH, standoff)
- Utility depends on ability to process

Motivations

- Non-Trivial texts require:
  - Complex relationships between elements in a text
  - Differing views of the text (physical vs. logical structure)
  - Overlapping and differing views of structures within a text (Ex., commentators who see different formal and syntactic structures)
  - Versioning

“Treeness” and Markup

- A markup tree has how many roots?
  - Answer: 1
- Example: XML document:

```
<?xml version=“1.0” standalone=“yes”?>
<text>
  <p>A short document.</p>
</text>
```

“Treeness” and Markup II

- Reality Check
  - This tree has more than one!
- Agreed markup trees have only one
- Question is: When is that required?
- Answer: When it is processed!
- Solution: Declare the root of a markup tree for processing

Recognizing Markup

- Documents are divided into:
  - Markup
  - PCDATA
- When do we need to recognize markup?
- Answer: When it is processed!
- Solution: Declare the markup to be recognized for processing

Markup vs. Processing

- Current Model of Markup and Processing:
  - Single fixed root defined in syntax
  - Markup defined in syntax
- Isn't processing different from markup?
- What if we declare a root for processing?
- What if we declare the markup to process?
- Result: Just-In-Time-Trees (JITTs)!

Just-In-Time-Trees

- Moves root requirement from syntax to processing
- Moves markup (recognized) from syntax to processing
- No more overlap, simply processing declared roots and markup
- Markup limited only by your imagination

Current Practice vs. JITTs

Syntax (fixed)	vs.	Processing (declared)
Root		Root
Markup		Markup

Parser

Implementing JITTs

- Requirements
  - Recognizing markup
  - Discard markup/PCDATA prior to declared root
  - Discard markup/PCDATA after leaves
- Recognizing markup
  - SAX Filter (but using DTD or Schema)
- Discarding PCDATA
  - Similar to XPath and subtrees

JITTs

- Compatible with legacy texts
- Construction of light-weight DOM trees
- Markup can represent the text as it is found “in the wild” (rather than pruned)
- No tree requirement for markup syntax
- Markup based on attribute values (here be versioning, Zanadu?)

Evaluation of JITTs I

- Extreme 2001 – 10 Requirements
  - 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

Evaluation of JITTs II

- 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 extracted from existing documents encoded in XML markup

Evaluation of JITTs III

- JITTs also support:
  - Building documents with declared markup
  - SGML files
  - non-SGML/XML files (cf. MECS, TexMECS, data tag)
- All by changing markup recognition (ISO 8879 robustness question settled)

Future Work

- Use of Attributes, DTDs, Range Algebra, Regexes, Schemas, to declare root and markup
- Data structures for parse forests
- Layering SVG or VRML for display of multiple trees
- Using TAG (Tree Adjoining Grammar) parsers for parsing multiple trees
- Tree discovery techniques

Additional Resources

DyALog
http://atoll.inria.fr/~clerger/DyALog/dyalog_toc.html

Prague Stringology Club
http://cs.felk.cvut.cz/psc/

X-Diff -- Detecting Changes in XML Documents
http://www.cs.wisc.edu/~yuanwang/xdiff.html

Xerces2 Java Parser
http://xml.apache.org/xerces2-j/index.html

XTAG Project
http://www.cis.upenn.edu/~xtag/

Support for Research

- Support organizations that make this research possible!
  - SBL: http://www.sbl-site.org
  - OpenText.org: http://www.opentext.org
  - TEI: http://www.tei-c.org