

# XML Schema Languages and the Flora of North America

Kevin Reiss

6/10/03

# Order of Operations

- XML Schema Overview
- Case Study: Flora of North America (XML Schema to support an Information Retrieval Application)
  - Discuss schema design
  - Discuss potential applications
- Software tools for XML Schema

# What is a Schema?

- A formal expression of the structure of a particular class of XML documents
- In layman's terms
  - Defines the names of the elements and attributes that can appear in a document
  - Defines the content models (structure) of that those elements and attributes can take
  - Interestingly, a schematic is NOT required with an XML document by XML applications

# Schema Scraps

## XML 1.0 DTD

```
<!ELEMENT greeting (#PCDATA)>
```

```
<!ATTLIST greeting type CDATA #REQUIRED>
```

## RELAX NG

```
name = element greeting { greeting.attlist, text }  
greeting.attlist = attribute type {text}
```

# Schema Scraps Cont.

## W3C XML Schema

```
<xs:element name="greeting">  
  <xs:complexType>  
    <xs:simpleContent>  
      <xs:extension base="xs:string">  
        <xs:attribute name="type" type="xs:string" />  
      </xs:extension>  
    </xs:simpleContent>  
  </xs:complexType>  
</xs:element>
```

# All this for?

```
<?xml version="1.0"?>  
<greeting type="salutation">  
  Hello World!  
</greeting>
```

# Why Schemas?

- Validation
- Documentation
- Query Support
- Data Binding
- Guided Editing

Source: XML Schema by O'Reilly  
Author: Eric Van Der Vlist

# Schema Types

- 3 Major Players
  - XML 1.0 DTDs
  - W3C XML Schema
  - RELAX NG Schema
  - Other Players
    - Schematron
    - RDF Schema

# Schema Classification

Grammar Based	XML DTDs	RELAX NG
Object-Oriented	W3C XML Schema	RDF Schema
Rule-Based	Schematron	

# XML DTDs

- The first schema for XML
- Descended from the DTDs used in SGML
- Very well-supported by most XML-related software
- Limited support for defining the datatype of the document content (only attribute values)
- Uses parameter entity declarations to implement modularity in a DTD

# RELAX NG

- More flexible cousin of DTDs
- Excellent Mixed Content Support
- Datatype Aware
- Allows a “Datatype Library” to provide datatypes beyond character data
- Most often this library is the W3C simple types defined for W3C XML Schema
- Has both an XML syntax and an alternative non-XML syntax similar to that of DTDs
- Gaining support, especially in open source projects

# W3C XML Schema

- Explicitly Object-Oriented
- Oriented towards data applications
- Revolves around notions of complex and simple types
- Simple types akin to primitives in Java
- Complex types akin to objects in Java
- Single-inheritance is used to derive from both complex and simple types
- Strong commercial support, reasonably good support in open source tools

# Schema Pitfalls

- Document vs. Data
- Structure vs. Content
- Reusability vs. Detail (beware the underscore)
- Container Elements
- Mixed Content
- Element vs. Attribute
- Restrictive content models
- Document modeling isn't as mature or methodical as data modeling with the relational or even OO model

# Schema Modularity

- Why?
  - Reusable Content Models
  - Extensible Content Models
- How?
  - Inheritance and other OO constructs built-in to W3C XML Schema
  - The [Pizza](#) model for DTDs or RELAX NG (core element set with user-chosen “toppings” = modules)

# Flora of North America

- Multi-Volume set of taxonomic descriptions (taxons) of the plants of North America
- Family: [family file](#)
- Genus: [genus file](#)
- Species: [species file](#)
- The same basic structure & hierarchy appears in each type of taxon
- Semi-structured data

# Original Schema for FNA

- FNA DTDs
  - [Family DTD](#)
  - [Species DTD](#)
  - Both are very general; define only top-level elements
  - Each top-level element is treated as a field
  - Flat structure

# FNA Issues/Needs

- Issues:
  - Potential need to expand the granularity of top-level sections
  - Presence of global elements
  - Loose content models required
  - Mixed content prevalent, especially in the <description> element
- Needs:
  - One schema for all types of taxons in the taxonomic hierarchy
  - Datatype checking for element/attribute content necessary
  - Potential need to deliver this schema in other formats (other schemas and configuration files)

# Choice of Schema Language

- RELAX NG
- Strong Support for mixed content; the interleave operator
- Relatively easy to author w/o IDE using compact syntax
- Support for W3C built-in datatypes
- Compatible for conversion to either DTD or W3C Schema with few restrictions
- Reliable conversion tools exist

# Modularization

- Driver file containing root content model and top-level elements
- Module containing global elements
- Modules for elements that have content models beyond the global elements
- These include the taxonomic hierarchy, nomenclature, description, distribution, and the visual/interactive key
- Create potentially reusable element classes within the modules

# Description Module

- Need to adequately represent the characters and the states of those characters within the description
- Need to enforce strong typing on some character's state values to support indexing of numeric and range data present
- Toughest decision to make
  - Should each character have a unique element declared for it?
  - Ultimately the indexing tools dictated the decision

# Compromise

- Create element class “top.level.characters” that contain a unique element name for each character
- Use the interleave operator to ensure that these appear only once
- State values for lower-level characters that have numeric data are marked with the general element “char”

# Application: Indexing

- Schema provides validation/query support for this activity
- When you increase the granularity of FNA markup indexing it becomes harder
- Current Swish-ex approach is inadequate

# Swishe-x Shortcomings

- Swishe-x treats XML documents like records with fields
- Doesn't use a standard XML processing model, DOM, SAX or XPath
- Can't handle mixed content
- Can't handle global elements
- Can't handle attribute values

# Application: Transformation

- Deliver the FNA XML documents as HTML to support the query interface
- Use an XSLT [stylesheet](#)
- Output: [species](#), [family](#), [genus](#)
- Pass parameters to the stylesheet to get ensure specific features for the taxon type and the record

# Application: Interface Configuration

- Schema serves as the configuration file
- Example [configuration file](#)
- XSLT could be used to convert a schema in XML to any type of configuration file format

# Alternative: Try the W3C OO Approach

- Create a super class complex type for Taxon
- Types for species, genus etc. could be derived from this
- Create a super class character complex type
- Derive all top-level types from this type
- Use simple-type derivation to create reusable content type for state values
- Use substitution groups to improve extensibility

# Schema Editing Tools

- Commercial:
  - XML Spy W3C Schema Tools
- Open Source:
  - Emacs/PSGML

# Schema Conversion Tools

- Sun [Relax NG Converter](#)
- [Trang](#) Conversion Tool
- Commercial Solutions (XML Spy, etc..)

# Schema Validation

- DTDs - Almost Any XML Parser
- W3C Schema - [Xerces 2.0](#) < Higher, [Sun MSV](#)
- Relax NG - [Jing](#), [Sun MSV](#)

# Questions?

- Project:
  - <http://www.isrl.uiuc.edu/~kmreiss/projects/fnaschema/public/>
  - Thanks for your time!