SAXONICA.COM
XSLT AND XQUERY PROCESSING

XML Schema Moves Forward

Michael Kay, Saxonica

So what's the problem?

- XML Schema 1.0 is a success
- But
  - it doesn't have all the functionality people need
  - it has more functionality than many people want
  - it's hideously complex
  - it's not as interoperable as it could be
  - it's very hard to read and write

How did this happen?

- Working group tried to unify
  - DTD, XML-Data, XDR, DCD, SOX, DDML...
- Full spectrum from documents to data
- Rag-bag of popular data types
- Validation and data binding

Net result:

Everyone uses it

No-one loves it

XML Schema 1.1 fixes some of these problems

- It doesn't remove any functionality
- It does reduce the complexity a little
  - by means of improved drafting and simplifying some rules
- It might improve interoperability a little
  - by making some of the rules clearer
- It does provide many of the features that users are asking for

Key new features

- Relaxations on Content Models
- Changed rules for derived types
- Co-occurrence Constraints
  - "Conditional Type Assignment"
- Changes to data types
- Schema modularity: xs:override
- Assertions

Assertions

- Borrowed from Schematron
  - but without the user-defined error messages
- Allow arbitrary boolean conditions to be defined for any simple or complex type
- Written using XPath 2.0 syntax
- Defined for any subtree of the document, evaluated within that subtree

Example: assertion on complex type

```
<xs:complexType name="article">
  <xs:assert test=
          "if (exists(affiliation))
           then exists(author)
           else true()"/>
```

Example: assertion on simple type

```
<xs:simpleType name="futureDate">
  <xs:restriction base="xs:date">
    <xs:assert test=". ge current-date()"/>
  </xs:restriction>
</xs:simpleType>
```

Assertions: limitations

- The latest draft
  - requires processors to support the whole of XPath 2.0,
  - but with restrictions on the context
    - no external documents
    - no access outside subtree
    - NOT schema-aware
- No user-defined messages
  - Scope for vendor extensions

Validation by grammar
vs
Validation by predicates

- Both approaches have value
- Grammar is more suited to narrative XML
- Predicates mirror integrity constraints in SQL
- Overlapping functionality
  - some constraints are more easily expressed in grammatical terms, some in terms of predicates

Assertions: impact

- Likely to be very widely used
- May sometimes be used in preference to complex grammars
- Competes with Schematron
- Cost of validation becomes very open-ended
- Saxon implementation allows escaping to Java code

Restriction using assertions

- Restriction by grammar
  - requires repeating the content model
  - maintenance nightmare
- Restriction by assertion
  - just say what's different
  - can do "deep restriction"
    - test="empty(.//@currency[. ne 'USD'])"

Context-dependent assertions

- Spec disallows calls to doc()
  - considered “a bridge too far”
- Spec allows current-dateTime()
- Saxon allows calls out to Java
  - enables cross-document validation
  - validation against DB content
  - validation relative to processing context, e.g document lifecycle

Assertions and Performance

- Cost is open-ended (rope to hang yourself)
- This worries some people
- However, people are currently checking the same constraints using non-XSD technology.

Conditional Type Assignment

- Defined on an element declaration
- Allows alternative types, depending on the values of the attributes of the element
- Designed so the content model can be decided by examining the start tag alone
- Generalizes xsi:type
  - any attribute name
  - any XPath condition (with caveats as before)
- Allows assignment of the error type
  - so it overlaps assertions

CTA Example

```
<xs:element name="CreditCard" type="CreditCard">
<xs:alternative test="@issuer='Visa'"
                type="VisaCreditCard"/>
<xs:alternative test="@issuer='MasterCard'"
                type="MasterCardCreditCard"/>
<xs:alternative test="@issuer='Amex'"
                type="AmexCreditCard"/>
<xs:alternative type="OtherCreditCard"/>
</xs:element>
```

CTA vs Assertions

- Many constraints can be expressed either way
- CTA is guaranteed streamable so may be more efficient
- CTA may give better diagnostics
- But Assertions are more powerful and perhaps easier to understand

Other facilities in XML Schema 1.1

- Generalized <xs:all> groups
  - allows cardinality constraints without ordering constraints
- Elements may belong to several substitution groups
- Define attributes that are allowed on any element
- Open content models: allows every element to contain wildcard children anywhere

Other facilities (cont'd)

- "Not-in-schema" wildcard
- Multiple xs:ID attributes
- Revised rules for valid type restriction
- Target Namespace can be defined on local element declarations
- xs:precisionDecimal data type

Conclusions

- XSD 1.0 is very widely used despite the known drawbacks
- XSD 1.1 adds most of the high-priority new features that users need
- It only makes slight improvements to the other problems:
  - over-complexity
  - poor interoperability

Finally

- The XML Schema WG needs help
  - new members
  - reviews and comments
- Try out some of the new features with Saxon
  - 75% implemented in 9.1
  - 90% implemented in 9.2