W3C RDF Schema Validation Workshop

Experiences with the W3C XML Schema Definition Language

Noah Mendelsohn
Tufts University
Email: nrm@arcanedomain.com
Web: http://www.arcanedomain.com

I was for many years a member of the W3C XML Schema working group, and am a co-editor of the original W3C XML Schema Definition Language Structures 1.0 Recommendation.

This talk is based on my experiences with the design of that language, and with the response it has received from users.

The opinions expressed here are my own. They do not necessarily represent the position of current or past schema working group members, or of any of my present or past employers.

Topics

- **Use cases**
- **What gets validated & why?**
- **The form of a schema**
- **Combining schemas**
- **Versioning**

For each topic I will describe XML Schema experiences and suggest insights pertinent to RDF schema design.

Use cases

- **XML Experience**
  - Different communities have different expectations
  - Tons of use cases
  - Very important to understand them, but…
  - Addressing all of them led to complex design
- **Suggestions:**
  - Clarify use cases carefully and early
  - Important to exclude as well as include
  - Don’t over-simplify but…
  - …simpler goals allow for simpler solutions

What gets validated and why?

- **XML Experience**
  - Validating a *text XML document* was obvious choice
  - We had “need” to support in memory and abstract documents for databases & APIs
  - Whole document vs. elements?
    - Element in context or context-free (do your parents/sibblings matter?)
    - Constructs like IDREF inherenetly context-sensitive
  - We chose context-free elements in infoset
- **Suggestions:**
  - Decide carefully the form of what gets validated?
    - Abstract triples seems an obvious answer for RDF
  - How much gets validated?
    - Always in context of global graph?
    - Bounded subgraph? Who specifies it? How?

How are schemas modeled & exchanged?

- **XML Experience**
  - Assumption: schemas should be XML documents -- never seriously challenged
  - Resulting syntax was very clumsy for users, some of whom chose the non-XML syntax of RelaxNG
- **Suggestions:**
  - Consider the underlying model for RDF schemas. Is it (presumably) triples?
  - Will RDF schemas be exchanged using a generic RDF syntax (N3, RDR/XML) or something more convenient?

How are schemas combined

- **XML Experience**
  - Use cases: schemas for types and fragments of XML must be composable:
    - Shipping address
    - Version control
    - Employee element
    - Different namespaces
  - I think we did OK on this at the element level; the underlying composition mathematics was hard to specify and never done right
- **Suggestions:**
  - Decide whether RDF users need to combine schemas
  - Obviously: hatch a design that meets the need

How are schemas combined

- **XML Experience**
  - Use cases: schemas for types and fragments of XML must be composable:
    - Shipping address
    - Version control
    - Employee element
    - Different namespaces
  - I think we did OK on this at the element level; the underlying composition mathematics was hard to specify and never done right
- **Suggestions:**
  - Decide whether RDF users need to combine schemas
  - Obviously: hatch a design that meets the need

Anticipate versioning

- **XML Experience**
  - User's data models aren't static
  - User's expect schemas to help them deal with data from the past (backwards compatibility) and the future (forwards compatibility
  - In XSD 1.1 we built mechanisms to distinguish content that's truly expected vs. tolerated
- **Suggestions:**
  - Consider the need for versioning and partial validation