NOTE-xml-schema-req-19990215

XML Schema Requirements

W3C Note 15 February 1999

This version: http://www.w3.org/TR/1999/NOTE-xml-schema-req-19990215
Latest version: http://www.w3.org/TR/NOTE-xml-schema-req
Editors: Ashok Malhotra (*petsa@us.ibm.com*) for IBM
Murray Maloney (*murray@muzmo.com*) for Veo Systems Inc.

Status of this document

This is a W3C Note published on 15 February 1999 as a deliverable of the XML Schema Working Group, which is part of the W3C XML Activity. It lists a base set of agreed requirements for an XML schema language.

This document represents a compromise that leaves many design questions open, creating opportunity for decision-making in the design phase. As the XML Schema work continues, the concrete implications of these requirements for the design will be worked out and documented. This document is a living document; it will be reviewed regularly by the Working Group, and may be revised to reflect changes in the Working Group's understanding. The Working Group does not anticipate substantial changes, but may decide to refine existing requirements or add new ones.

Comments about this document should be addressed to the XML Schema Requirements Comments list at www-xml-schema-comments@w3.org. Comments accumulated by 1 March 1999 will be reviewed in March 1999.

A list of current W3C technical reports and publications, including working drafts and notes, can be found at http://www.w3.org/TR.

Abstract

This document specifies the purpose, basic usage scenarios, design principles, and base requirements for an XML schema language.

Table of Contents

1. Overview

The XML 1.0 specification defines the concepts of well-formedness and validity; it is very simple to check a document for well-formedness, while validation requires more work but allows the user to define more powerful constraints on document structure. XML validity requires that a document follow the constraints expressed in its document type definition, which provides the rough equivalent of a context-free grammar for a document type.

For some uses, applications may need definitions of markup constructs more informative, or constraints on document structure tighter than, looser than, or simply different from those which can be expressed using document type definitions as defined in XML 1.0. There is also a widespread desire to allow markup constructs and constraints to be specified in an XML-based syntax, in order to allow tools for XML documents to be used on the specifications.

By charter, the XML Schema Working Group is assigned to address the following issues:

structural schemas
: a mechanism somewhat analogous to DTDs for constraining document structure (order, occurrence of elements, attributes). Specific goals beyond DTD functionality are
  - integration with namespaces
  - definition of incomplete constraints on the content of an element type
  - integration of structural schemas with primitive data types
  - inheritance: Existing mechanisms use content models to specify part-of relations. But they only specify kind-of relations implicitly or informally. Making kind-of relations explicit would make both understanding and maintenance easier

primitive data typing
: integers, dates, and the like, based on experience with SQL, Java primitives, etc.; byte sequences ("binary data") also need to be considered

conformance
: The relation of schemata to XML document instances, and obligations on schema-aware processors, must be defined. The Working Group will define a process for checking to see that the constraints expressed in a schema are obeyed in a document (schema-validation); the relationship between schema-validity and validity as defined in XML 1.0 will be defined.

The XML Schema work is interdependent with several other areas of W3C activity. These are listed below under Design Principles.

2. Purpose

The purpose of the XML schema language is to provide an inventory of XML markup constructs with which to write schemas.

The purpose of a schema is to define and describe a class of XML documents by using these constructs to constrain and document the meaning, usage and relationships of their constituent parts: datatypes, elements and their content, attributes and their values, entities and their contents and notations. Schema constructs may also provide for the specification of implicit information such as default values. Schemas document their own meaning, usage, and function.

Thus, the XML schema language can be used to define, describe and catalogue XML vocabularies for classes of XML documents.

Any application of XML can use the Schema formalism to express syntactic, structural and value constraints applicable to its document instances. The Schema formalism will allow a useful level of constraint checking to be described and validated for a wide spectrum of XML applications. For applications which require other, arbitrary or complicated constraints, the application must perform its own additional validations.

3. Usage Scenarios

The following usage scenarios describe XML applications that should benefit from XML schemas. They represent a wide range of activities and needs that are representative of the problem space to be addressed. They are intended to be used during the development of XML schemas as design cases that should be reviewed when critical decisions are made. These usage scenarios should also prove useful in helping non-members of the XML Schema Working Group understand the intent and goals of the project.

1. Publishing and syndication

   Distribution of information through publishing and syndication services. Involves collections of XML documents with complex relations among them. Structural schemas describe the properties of headlines, news stories, thumbnail images, cross-references, etc. Document views under control of different versions of a schema.

2. Electronic commerce transaction processing.

   Libraries of schemas define business transactions within markets and between parties. A schema-aware processor is used to validate a business document, and to provide access to its information set.

3. Supervisory control and data acquisition.

   The management and use of network devices involves the exchange of data and control messages. Schemas can be used by a server to ensure outgoing message validity, or by the client to allow it to determine what part of a message it understands. In multi-vendor environment, discriminates data governed by different schemas (industry-standard, vendor-specific) and know when it is safe to ignore information not understood and when an error should be raised instead; provide transparency control. Applications include media devices, security systems, plant automation, process control.

4. Traditional document authoring/editing governed by schema constraints.

   One important class of application uses a schema definition to guide an author in the development of documents. A simple example might be a memo, whereas a more sophisticated example is the technical service manuals for a wide-body intercontinental aircraft. The application can ensure that the author always knows whether to enter a date or a part-number, and might even ensure that the data entered is valid.

5. Use schema to help query formulation and optimization.

   A query interface inspect XML schemas to guide a user in the formulation of queries. Any given database can emit a schema of itself to inform other systems what counts as legitimate and useful queries.

6. Open and uniform transfer of data between applications, including databases

   XML has become a widely used format for encoding data (including metadata and control data) for exchange between loosely coupled applications. Such exchange is currently hampered by the difficulty of fully describing the exchange data model in terms of XML DTDs; exchange data model versioning issues further complicate such interactions. When the exchange data model is represented by the more expressive XML Schema definitions, the task of mapping the exchange data model to and from application internal data models will be simplified.

7. Metadata Interchange

   There is growing interest in the interchange of metadata (especially for databases) and in the use of metadata registries to facilitate interoperability of database design, DBMS, query, user interface, data warehousing, and report generation tools. Examples include ISO 11179 and ANSI X3.285 data registry standards, and OMG's proposed XMI standard.

4. Design Principles

In the design of any language, trade-offs in the solution space are necessary. The following design principles should guide the working group in making these trade-offs. Design principles are desirable, but not fully measurable, characteristics.

The XML schema language shall be:

1. more expressive than XML DTDs;
2. expressed in XML;
3. self-describing;
4. usable by a wide variety of applications that employ XML;
5. straightforwardly usable on the Internet;
6. optimized for interoperability;
7. simple enough to implement with modest design and runtime resources;
8. coordinated with relevant W3C specs (XML Information Set, Links, Namespaces, Pointers, Style and Syntax, as well as DOM, HTML, and RDF Schema).

The XML schema language specification shall:

1. be prepared quickly;
2. be precise, concise, human-readable, and illustrated with examples.

5. Requirements

Structural requirements

The XML schema language must define:

1. mechanisms for constraining document structure (namespaces, elements, attributes) and content (datatypes, entities, notations);
2. mechanisms to enable inheritance for element, attribute, and datatype definitions;
3. mechanism for URI reference to standard semantic understanding of a construct;
4. mechanism for embedded documentation;
5. mechanism for application-specific constraints and descriptions;
6. mechanisms for addressing the evolution of schemata;
7. mechanisms to enable integration of structural schemas with primitive data types.

Datatype requirements

The XML schema language must:

1. provide for primitive data typing, including byte, date, integer, sequence, SQL & Java primitive data types, etc.;
2. define a type system that is adequate for import/export from database systems (e.g., relational, object, OLAP);
3. distinguish requirements relating to lexical data representation vs. those governing an underlying information set;
4. allow creation of user-defined datatypes, such as datatypes that are derived from existing datatypes and which may constrain certain of its properties (e.g., range, precision, length, mask).

Conformance

The XML schema language must:

1. describe the responsibilities of conforming processors;
2. define the relationship between schemas and XML documents;
3. define the relationship between schema validity and XML validity;
4. define the relationship between schemas and XML DTDs, and their information sets;
5. define the relationship among schemas, namespaces, and validity;
6. define a useful XML schema for XML schemas;