WS-I Submission for the W3C Workshop on XML Schema 1.0 Specification User Experiences

Date: April 24, 2005

Version: 1.0

Author: Erik Johnson, Epicor (ejohnson@epicor.com)

Table of Contents

1. Introduction

The Web Services Interoperability Organization (WS-I) herein offers a submission to the W3C Workshop on XML Schema 1.0 User Experiences. The WS-I appreciates this opportunity to contribute to the Workshop and looks forward to working with the W3C in fostering broad adoption of the XML Schema 1.0 Specification.

Unlike other specifications relevant to web services, the WS-I had initially felt that there were no clear ambiguities or feature pathways of the *W3C XML Schema 1.0 Specification 2nd Edition* itself that merited development of a WS-I XML Schema profile. In fact, the WS-I Basic Profile 1.0 expressly allows the use of all W3C XML Schema 1.0 Specification constructs and types.

In 2003 however, the WS-I commissioned a Working Group to study interoperability issues with XML Schema raised by WS-I Community members, specifically end-user organizations. The XML Schema Work Plan Working Group (WS-I SWPWG) was then chartered to produce a recommendation for possible further action to the WS-I Board. The WS-I SWPWG began work in November of 2004 to study the issue claims and define how the WS-I might in fact take action.

This submission summarizes portions of the conversation and consensus from the work of the WS-I XML Schema Work Plan Working Group.

2. Schema Authoring

The WS-I Organization includes member organizations that produce XML Schema documents in order to standardize web services for specific industries. One category of problems has been about how to best employ XML Schema constructs in common compositional situations.

XML Schema producers often try to define abstractions and/or reuse mechanisms meant to improve flexibility and overall longevity of schema sets. Ideally, XML Schema documents should be extensible by implementers, who may also incorporate these schemas into derivative schemas and WSDL documents. In attempting this, problems often occur in determining how to manage namespaces. This is an issue not only in the base schemas, but also in prescriptive guidance to those adopting and extending the base schemas. Some schema producers also try to factor out data from behavioral aspects of service definitions when constructing WSDL documents.

No matter what abstraction mechanisms are employed, human implementers and toolkits alike usually fail to recognize them. Modularity that is carefully built into the design of an XML Schema set often becomes implemented as a monolithic system.

Some XML Schema producers attempt to achieve forward compatibility by declaring explicit extensibility points in their schemas. This typically means insertion of wildcard element declarations throughout the schema document, which lessens readability. Schema Producers sometimes unintentionally violate the Unique Particle Attribution (UPA) rule.

In fact, much of the user community is completely unaware of the UPA rule. One example is an **`all`** compositor containing an element declaration with the attribute **`maxOccurs="unbounded"`**. Reasonably proficient developers do not understand why this is illegal. There are also reports that some toolkits intentionally disregard the UPA rule.

The WS-I Community understands that the W3C has been looking at the problem of XML Schema evolution for sometime. The WS-I Community is also aware of some proposals in circulation, like the idea of "weak references" as described in *An Approach for Evolving XML Vocabularies Using XML Schema* by Noah Mendelson[1]. That solution or perhaps a similar idea would be useful to the WS-I Member Community.

3. Toolkit Support for XML Schema

Platforms and toolkits have supported the runtime requirements for web services (SOAP, WSDL, UDDI, and XML) quite well for some time and the overall state of these tools continues to improve. With the rest of the web services platform maturing, developers are focusing more on the design-time experience, ostensibly to improve productivity. One acute pain point for developers is in mapping XML Schema constructs to programming languages.

Most toolkits are able to generate valid XML Schema documents from most or all constructs present in the toolkit programming language(s). The inverse situation – the ability to generate language types from most or all XML Schema 1.0 Specification constructs – is more problematic. Toolkit vendors typically have to introduce non-standard language extensions or have type serializers and metadata control the mapping of XML data to and from language types.

One interesting observation is that few web service implementers employ W3C XML Schema validation processors. Many users "validate" SOAP messages using only inherent SOAP processing mechanisms, possibly with some uncoordinated help from type serializers. This situation often means that XML Schema constructs like type facets and the post-schema-validated Infoset (PSVI) are ignored when web service

[1] [http://lists.w3.org/Archives/Public/www-tag/2004Aug/att-0009/NRMVersioningProposal.txt]

messages are processed, which in turn discourages the use of such constructs by Schema Authors.

In many cases, XML Schema and WSDL documents are crafted independently of any specific implementation stack. Some XML Schema producers prefer this methodology for a number of reasons including the strong general desire to avoid bias toward a specific toolkit or platform stack. There are also strong feelings that XML has inherent capabilities that cannot be expressed adequately from many existing programming languages.

This situation ties back to problems XML Schema authors have when trying to develop an XML Schema modularity and extensibility strategy. Abstraction mechanisms employed by schema producers – well thought-out or not – tend to *further broaden* the set of XML Schema constructs and features used in their solutions. This puts more pressure on the "downstream" platforms and toolkits to have complete and accurate support for the W3C XML Schema 1.0 Specification.

Issues with toolkits and especially language bindings in handling XML Schema are being perceived as interoperability problems – even though the W3C XML Schema 1.0 Specification is not specifically at fault. This situation is causing a growing fracture in the industry. Some believe that there ought to be an XML Schema Profile specifically to improve compatibility with widely-used toolkits and programming languages. Conversely, others believe that doing so would suppress the use of capabilities unique to XML and the Infoset while also limiting further innovation.

The WS-I Community has members falling on both sides of this debate. But there is general agreement that multiple standards for message descriptions would be detrimental to the growth of web services.

4. Testing Resources

Little of the negative experience in using XML Schema by the WS-I membership is directly related to the W3C XML Schema 1.0 Specification. Interoperability problems seem to be "witnessed" more in design-time situations, when XML Schemas and existing programming languages meet.

But determining toolkit, platform, or programming language support for a given XML Schema construct is not always a yes or no question. There is general agreement that toolkits that produce errors or otherwise prevent developers from processing XML Schema documents fall short of "supporting" the W3C XML Schema 1.0 Specification. However, a toolkit that does *not* have a language mapping to, say, the XML Schema `choice` compositor could "fallback" to expose the XML data as a Document Object Model (DOM) object instance or other generic container. Some consider this approach as workable while others do not.

There are developers who want their toolkits and programming languages to completely manage (or even hide) their XML data, XML Schema, and WSDL documents. Other toolkit vendors supply special-purpose languages where it is only practical to provide bindings to a few very simple XML Schema constructs (if any). And yet others will want to eschew language bindings and type serializers to build applications that manipulate XML data more directly.

It defeats the purpose of XML web services if developers creating or consuming services have to understand the toolkit and platform assumptions of their counterparts. So, toolkit support of XML Schema needs to be measured in the context of *suitability to purpose*. But there are many permutations of platform stacks, programming languages, and toolkits in use and the idea of suitability is clearly subjective.

WS-I members have discussed these issues from two viewpoints: The first is the need for guidance and clarification of the W3C XML Schema 1.0 Specification, especially around best practices for extensibility, versioning, and type composition (modularity). The second is the need for a testing capability that covers XML Schema constructs found in real-world schemas (good, bad, and ugly) rather than academic coverage of XML Schema features.

The second point is critical. WS-I members have widely varying platforms and toolkits. Even if accurate XML Schema compatibility reports existed for toolkits, their value is fleeting given the pace of change and that multiple platforms often have to be considered. WS-I members say they would be more effective in deploying applications that employ XML Schema documents if they could measure the suitability of their platforms tests built from XML Schema constructs from real-world, commercially produced XML Schema documents.

Also, users should be able to test their platforms and toolkits *locally* rather than relying on any 3rd-party organizations. These testing artifacts might include actual running endpoints (with schema validation of incoming messages). The test suite should be accompanied with documentation clarifying the XML Schema usage being tested and any recommendations for implementers and XML Schema producers alike.

5. Thanks

The WS-I Community includes toolkit providers, end users, solutions providers, and XML Schema producers. Our organization exists to foster the adoption of XML web services by continuing to improve interoperability. We are ready to help wherever possible and look forward to working with the W3C on this mission.

6. Acknowledgments

This document is the work of the WS-I XML Schema Work Plan Working Group, whose members have included:

James Barnette (DOD), Fadol Bassam (Accenture), David Burdett (SAP), Jon Calladine (BT), David Connelly (Open Applications Group), Ugo Corda (SeeBeyond), Paul Downey (BT), Tim Ewald (Mindreef), Chris Ferris (IBM), Tim Fowler (Ford Motor Company), Paul Gallivan (Fidelity), Marc Hadley (Sun Microsystems), Erik Johnson (Epicor), Ashok Malhotra (Oracle), Ron Marchi (UGS), Jonathan Marsh (Microsoft), David Maze (DataPower), John Miller (IBM), Duane Nickull (Adobe), Mark Nottingham (BEA), Santiago Pericas-Geertsen (Sun Microsystems), Doug Purdy (Microsoft), Rimas Rekasius (IBM), Mark Richards (Fidelity), Darren Self (MicroFocus), Dorival Simoes (Accenture), Jorgen Thelin (Microsoft), Sekhar Vajhala (Sun Microsystems), Bob Vaughn (Hewlett-Packard), Faisal Waris (Ford Motor Company), Alan Weissberger (NEC), Ian White (MicroFocus)