Translating a DTD to W3C XML Schema with special considerations of IPTC standards

2003-09-09

Ulf Wingstedt

ulf.wingstedt@cnet.se

CNet Svenska AB

1 Abstract

Although challenged by competing schema proposals, the *de facto* standard XML Schema from W3C is now replacing the DTD technology on a broad scale providing increased expressive power and stronger validation. Existing legacy XML applications defined using the DTD technology can be converted into XML Schemas that replace the DTDs.

A semi-automatic process is proposed for converting a DTD into an XML Schema. It is possible to use conversion functionality found in a small set of commercial XML editing tools to create a starting point for manual schema design and editing. However, the automatically generated schemas are often erratic and cannot be used as is, without manual corrections. Moreover, the full expressive power of XML Schema cannot be used through automatic conversion from the less expressive DTD model. Manual editing and design is a prerequisite for a successful schema translation project.

IPTC's DTDs for NewsML, SportsML and NITF can successfully be translated into XML Schemas with equivalent content models as in existing DTDs, adding stronger type checking from the XML Schema datatype model

Schema Technologies - Overview

Soon after the release of the XML standard from W3C in February 1998, voices were raised for the need of a more expressive XML vocabulary definition language than the DTD technique included in the XML specification. While the “fathers of XML” envisaged document editing and publishing applications on the Internet as the primary application area for XML, the new born standard was immediately captured by the IT industry for additional use in database driven publishing and systems integration applications.

As a consequence, the need for a schema language with support for “database like” strong data types instead of simple strings was evident. In addition, a new schema language should use XML syntax in order to enable simple integration and use in standard XML tools.

Over the years, many schema languages addressing these requirements have been proposed by vendors, individuals and industry standard bodies. Finally the W3C released a recommendation for XML Schema that is now the *de facto* standard for definitions of XML vocabularies. During the last year, W3C’s XML Schema has received increasing support in commercially available XML editing tools and is finally replacing the DTD on a broad scale. Existing XML DTDs are being translated and replaced by XML Schema definitions and new initiatives creating new XML vocabularies head directly for XML Schema definitions without creating a DTD.

Compared to the DTD, the main features of XML Schema are:

- **Strong typing on attributes *and* elements –** While the DTD only allows simple strings and enumerated value sets for attributes, XML Schema has a rich data type model that moreover can be attached to both attribute and element values. Examples of datatypes are number types such as integers, non-negative integers and floats as well as date and datetime.
- **Type derivation –** The built in datatypes can be used as a base for creation of derived types. For instance, the built in type non-negative integers can be used as base for a type allowing only values in the range 30-60.
- **Alternate content models based on subtypes –** Based on type values found in XML instance documents, one of several alternate content models can be selected. For instance, a sports result element can have different child elements depending on the type of sport that is reported in the XML instance document.
- **Support for XML Namespaces –** Namespace support is well integrated in XML Schema.
- **XML based syntax –** Although syntax issues usually are not so important, the fact that it is XML based is important as it allows us to use standard XML parsers, XSLT processors and stylesheets etc to process, publish and maintain the schema. Custom made applications can parse and access definitions in a schema, XSLT stylesheets can be used to publish schemas as HTML, or PDF documents.

Although W3C's XML Schema is the *de facto* standard, it is not unchallenged. The main competitor today is the RELAX NG initiative, a technical committee within OASIS (http://www.oasis-open.org) proposed and chaired by James Clark. The main objectives for RELAX NG are very similar to the objectives behind W3C's XML Schema and both schema languages are at the same level of expressive power and detail control. The largest difference is that RELAX NG has the objectives of "simple" and "easy to learn" as top priorities.

W3C's XML Schema is, correctly, considered as complex and difficult to learn and also verbose due to its syntax. It is however not obvious (for the author of this report) that RELAX NG really achieves its objectives to be simpler in this area. Both approaches share the same problems of complexity and verboseness due to high expressive power and use of XML syntax. Also, syntax is much a matter of personal taste and the benefits of a schema language should not be judged by its syntax, but from its expressive power where both W3C's XML Schema and RELAX NG are at a similar level. The main advantage for W3C's XML Schema over RELAX NG is its position as a well accepted de facto standard, implemented in many tools and parsers, and there is really no need for a competing approach that does not provide any substantial additional benefits compared to XML Schema (again, an opinion of the author).

Converting a DTD to XML Schema

Work on revisions and new versions of existing XML vocabularies today often include replacing a DTD based definition with XML Schema. Since the expressive power of the DTD technology is less than XML Schema's, the main issue is to decide how much more rules and definitions that should be included in the formal schema, if any.

In practice, a DTD has never been expressive enough to include all rules that guide the use of an XML vocabulary. The DTD definition is normally accompanied by a written, verbal, specification that includes additional, normative but informal, rules that complements the formal rule set in the DTD. When converting to XML Schema, more of the informal rules from the specifications can be captured formally in the schema, although normally not all of them.

So, we have two main options:

1. Do a direct conversion that would create a 1-1 mapping from the DTD to the XML Schema, having identical sets of definitions and rules. This will give us the XML syntax, but not much more.
2. Do a translation and add enforced datatype controls according to what is said in the informal specification. We will achieve better type checking but some string values that were ok in a DTD based version before will now cause validation errors.

An existing DTD can be used as input for automatic conversion to XML Schema. Several tools exist that includes this functionality (see section below). However, as is always the case when trying to automatically convert from a less expressive model to a more expressive, it is impossible to fully use the functionality of the more expressive model. As in the two options above, an automatic conversion from a DTD to an XML Schema would effectively give the DTD expressed as an XML Schema and very little would be gained.

Also, any given DTD can be expressed in many different equivalent XML Schemas. In fact, an XML Schema describing an XML vocabulary can be constructed following one of several different design principles, while still solving the same problems as another schema using another approach. For instance, a DTD have a flat global element definition structure while XML Schema also allows nested element definitions inspired by the instance document element structure.

Selection of design guidelines to follow is up the application's requirements. It should be noted that XML Schema technology is still new, and experience and "best-practice" have only recently started to appear. Good starting points can be found at http://www.xfront.com and the xml dev mailing list, http://lists.xml.org/.

Conversion Tools

The number of players in the market for XML Schema and DTD editing tools is not very large. When we exclude tools that do not supply a conversion facility from a DTD to an XML Schema, there is only a handful left.

In general, the tools do a reasonable job in converting a given DTD to an XML Schema. The structure of the achieved schema is normally very close to the structure of the DTD, i.e. a flat element structure. XML Schemas ability to have a more document oriented element structure, with nested element and attribute definitions, can rarely be obtained in automatic conversions. In fact, only one of the tools studied here supports different kinds of conversion strategies where the user can select the one most suited for the task.

Another problem is related to DTDs that are divided into several separate files such as SportsML. When a tool loads a DTD, it creates an internal memory model of the DTD that spans all files. When the DTD is converted into a XML Schema, the information about its division into included sub files is lost and the resulting XML Schema is all in one single file.

DTDs have no built in support for XML namespaces. In case namespaces have been used, elements and/or attributes had to be defined in the DTD with its namespace prefix, e.g. xml:lang. In such cases, it is impossible for a conversion tool to create proper namespace definitions in the XML Schema and it has to be added manually.

We have selected and tested four commercial quality tools. Unfortunately, several tools were not 100% perfect and it is likely that all tools, under certain and varying circumstances, might generate non-valid XML Schemas! In the test, XMLWriter stands out because of frequent errors in generated schemas.

A particularly problem area seems to be the ANY content model from the DTD, where all tools fail to create valid XML Schema. (ANY is however easy to express in XML Schema and can be included with manual editing of the conversion result such as, e.g. has been done in the DataContent element definition in the NewsML XML Schema).

XML Spy

XML Spy (version 5) from Austrian company Altova (http://www.xmlspy.com) is probably the market leader today in the XML document and schema editing business. It is loaded with many different types of functions, including a DTD to XML Schema conversion function.

XML Spy is the only tool tested here that gives a user some control over which conversion strategy to use. An element from the DTD can either be defined as a global element in the XML Schema, or as Complex Types with local element definitions referring these types. The former strategy will give an XML Schema structure that is close to the DTD structure while the latter will create a structure closer to an XML document instance for the schema.

A schema generated by XML Spy is somewhat larger than the schema generated by, e.g. TurboXML. This is due to that XML Spy does not create element and attribute groups from entity definitions in the DTD but instead copies the entity definitions into each place where it was referenced in the DTD. The converter in TurboXML does a better job with regards to entities and creates the corresponding groups that then are used by reference.

XMLWriter

XMLWriter is an XML editor tool from the Australian company Wattle Software (http://www.xmlwriter.com). Its functionality scope is similar to XML Spy, including DTD/Schema based XML document editing, XSLT, DTD and XML Schema authoring support. The validation engine is built on top of Microsoft's MSXML4 parser.

XML Writer includes a function for converting a DTD to an XML Schema using a single default conversion strategy that produces a DTD like flat element structure. All attributes with enumerated value sets are converted into named simple types. However, the quality of the generated schema appears to be low. Even though the function has only been tested briefly, non-valid schemas were generated that had to be corrected manually.

TurboXML

TurboXML from Tibco (http://www.tibco.com) is a tool focused at editing of schema languages and supports a variety of different schema languages, including DTD and W3C XML Schema. Previously named XML Authority, the tool has been a pioneer in schema conversion.

When converting a DTD to W3C XML Schema, TurboXML applies a built-in conversion strategy that cannot be changed or controlled by the user. The result is a flat element structure (a copy of the DTD's structure) with attribute and value set definitions inline, as anonymous types.

Oxygen

Oxygen XML Editor from Oxygen (http://www.oxygenxml.com) is an XML editor with broad support for different XML technologies such as schema/DTD driven XML document editing, XSLT, XML Schema and DTD. Conversion between DTD and XML Schema is driven by the ***Trang*** converter written by James Clark (http://www.thaiopensource.com) that is integrated as a plug-in in Oxygen.

Converting current IPTC standards

XML Schema development and maintenance

A typical development plan for an XML Schema with a current DTD as a starting point would be:

1. Plan over all structure of the schema, including possibly division into several schema files. Study "best-practice" to decide design approach. For existing DTD based vocabularies, it can often enhance the user community's understanding of the schema if the structure is similar to the original DTD's structure.
2. Automatically convert the DTD to XML Schema using a suitable tool with a conversion approach that gets as close as possible to the expected structure.
3. Manually edit the converted schema to achieve the overall structure and control.
   a. Consider use of XML Namespaces
   b. Stronger type control for mission critical data. If possible, use built in XML Schema datatypes. (Note that it sometimes can be wise to not use stronger type checking in case of, e.g. not mission critical data, in order not to unnecessarily harm existing applications.)
4. Test and verify that the new schema validates the intended content models and values.

Development and maintenance of further versions is a work task very similar to how DTDs have been maintained earlier.

IPTC Standards

In this section, we look at how three existing DTD based IPTC standards may be converted into XML Schema. We have also tried the above tools on the DTDs.

NewsML

NewsML has already been translated into an equivalent schema that is distributed in parallel to the DTD:

" NewsML Schema Version 1.1. This Schema represents the same document structure as the NewsML DTD version 1.1. In addition is provides control over element and attribute content in accordance with the NewsML Specification." (from the XML Schema for NewsML 1.1)

SportsML

The division of the DTD into Core, Control and sports specific DTD files can be implemented also using XML Schema. However, conversion tools cannot generate such a file set automatically.

The requirements for alternate content models for different sports metadata can also be handled by XML Schema. It offers a mechanism, more elegant than the DTD, in typed instances. For instance, consider the following schema:

```
<xs:element name="sport" type="sportType"/>
```

A sport element defined as being of the type sportType.

```
<xs:complexType name="sportType">
    <xs:sequence>
        <xs:element name="name" type="xs:string"/>
    </xs:sequence>
</xs:complexType>
```

The generic sportType is defined as having a single element, name.

```
<xs:complexType name="hockeyType">
    <xs:complexContent>
        <xs:extension base="sportType">
            <xs:sequence>
                <xs:element name="hockey-penalty"
                    type="xs:string"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The hockeyType is defined as an extension to the generic sportType with an additional hockey-penalty element.

```
<xs:complexType name="soccerType">
    <xs:complexContent>
        <xs:extension base="sportType">
            <xs:sequence>
                <xs:element name="socker-penalty"
                    type="xs:string"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The soccerType is defined as another extension to the generic sportType with an additional soccer-penalty element.

In an XML instance document, the intended sport type is defined using the xsi:type attribute and all three of the below instance structures are thus valid according to the above schema:

```
Instance 1
<sport>
    <name>A generic sports event</name>
</sport>

Instance 2
<sport xsi:type="hockeyType">
    <name>An Hockey Event</name>
    <hockey-penalty>2 minutes</hockey-penalty>
</sport>


Instance 3
<sport xsi:type="soccerType">
    <name>A Soccer Event</name>
    <soccer-penalty>red card</soccer-penalty>
</sport>
```

Note: Namespace declarations in the instances has not been included

Examples of invalid instances are those that include types not defined, use soccer-penalty element without defined as a soccerType type.

NITF

The NITF DTD is quite straight forward to convert into an XML Schema with a similar, flat, structure as the existing DTD. No particular problems are envisaged.

Only TurboXML is able to generate a schema that validates the same test file (nitf-fishing.xml) as the existing DTD. All other tools created schemas with major or minor errors, the schema from XMLWriter was not even valid in itself, while Oxygen and XML Spy created schemas with content models that differed from the original content models in the DTD.