Extensible Schema Documentation with XSLT 2.0

Felix Michel
ETH Zürich

Erik Wilde
UC Berkeley

ABSTRACT

XML Schema documents are defined using an XML syntax, which means that the idea of generating schema documentation through standard XML technologies is intriguing. We present *X2Doc*, a framework for generating schema-documentation solely through XSLT. The framework uses *SCX*, an XML syntax for XML Schema components, as intermediate format and produces XML-based output formats. Using a modular set of XSLT stylesheets, X2Doc is highly configurable and carefully crafted towards extensibility. This proves especially useful for *composite schemas*, where additional schema information like Schematron rules are embedded into XML Schemas.

Categories and Subject Descriptors: D.2.7 [**Software**]: Software Engineering — Distribution, Maintenance, and Enhancement — Documentation

General Terms: Documentation, Management, Languages

Keywords: XML Schema, SCX, X2Doc

1. INTRODUCTION

The idea of generating documentation for XML Schemas [1] seems compelling: XML Schema documents use an XML syntax, and with XSLT a powerful transformation language for XML is readily available. However, the task is not as simple as it might seem. This is mostly due to the misalignments between the transfer syntax and the abstract data model of XML Schema. Because of the difficulties of retrieving the relevant schema structures from the transfer syntax, authors of comparable applications usually choose to employ programming languages like Java, where dedicated schema APIs are available.

In contrast, *X2Doc*, the documentation framework presented here, is based on the *Schema Component XML Syntax (SCX)* [2]. This format represents a fully assembled XML Schema in terms of its schema components, and does so in a self-contained way. It thus includes all components from schema documents that have been imported or included, as well as the built-in type definitions, and it reflects all rules defined by the specification.

2. APPLICATIONS

The main advantages of a documentation-generating application based on open technologies like XSLT are high portability, versatile configuration, and simple and powerful extensibility.

WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.
ACM 978-1-59593-654-7/07/0005.

Extensibility is especially important in the context of *composite schemas*, where other, complementary XML-based schema fragments, such as Schematron rules, are embedded into XML Schemas. X2Doc can easily be extended to cover such additional parts by adding corresponding template rules.

The following example uses the schema from the *Schema Primer* of XML Schema 1.0. We assume a company, which uses this schema, to have defined certain rules for schema management and documentation: Schema documents are annotated with a company-internal XML vocabulary identified by a namespace mapped to the prefix `doc`.

```
<complexType name="RegionsType">
 <annotation><appinfo>
  <doc:uri>documentation.html</doc:uri>
  <doc:part>RegionsType</doc:part>
  <doc:author mail="...">Peter Sample</doc:author>
 </appinfo></annotation>
 <sequence><element name="zip" maxOccurs="unbounded">
 ...
```

The annotations relate schema components to further documentation held externally in HTML format.

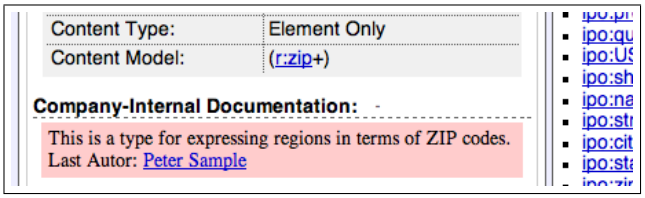
```
<div id="RegionsType"><p>This is a type for expressing...
```

While most documentation tools cannot adapt to this convention without having some code rewritten, documentation generated by X2Doc is readily extensible to incorporate such embedded parts. It requires only a template rule that matches application information components. This can be easily done, and the resulting stylesheet looks as follows:

```
<xsl:import href="X2Doc-xhtml.xsl"/>
<xsl:template match="scx:application-information">
 <h4>Company-Internal Documentation:</h4>
 <div class="exampleComDoc">
  <xsl:sequence select="id(doc:part, doc(doc:uri))"/>
  <p>Last Autor: <a href="mailto:{doc:author/@mail}">
   <xsl:value-of select="doc:author"/></a></p></div>
</xsl:template>
```

The first line imports the XHTML module of X2Doc. XSLT's import precedence makes sure that the above template rule overrides the rule defined in the imported module. The figure shown on top of the next page shows the resulting documentation, extended to cover the company-specific annotations. In this example, X2Doc uses a custom CSS for formatting the company-specific section.

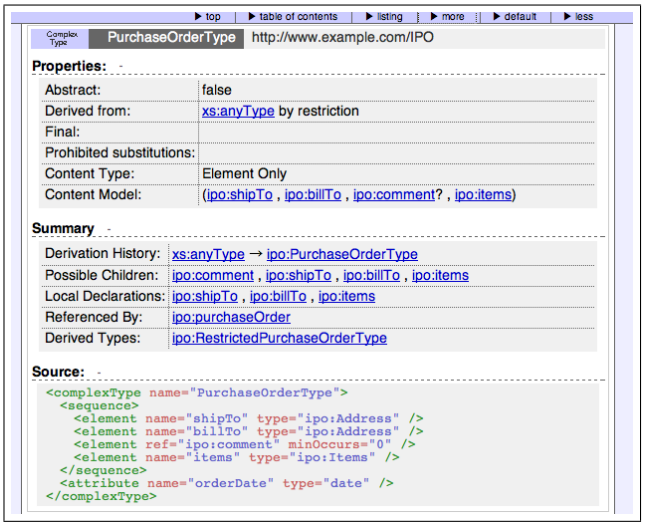
Configurability of X2Doc is possible in different ways: Substantial structural changes can be done by adapting the XSLT template rules. For instance, the appearance and order of basic structural *blocks* (e.g., the table of contents) can be influenced in a central switching template rule. A



wide variety of configurations can be made in a configuration XML document. The choice of CSS documents is one example, definition of XML namespace prefixes, configuration of the TOC, and control of output format and of serialization options are others. Finally, the most important parameters can be overridden using stylesheet parameters.

3. DOCUMENTATION FEATURES

Based on SCX's access to schema components, the properties of these components can be displayed straightforward. Had the transformation to be carried out on the XML transfer syntax, many of those component properties would need to be collected cumbersomely. Furthermore, the XSLT function library, which is part of SCX, allows for convenient navigation of the relationships between schema component, for example traversal of the type hierarchy. As a result, the documentation generated from SCX is highly hyperlinked. This feature proves to be especially useful for capturing the complex structure of XML Schema.



The figure above shows the description of a complex type definition. In the section *Summary*, many of the relationships mentioned above are made navigable through hyperlinks. This includes the complete path of derivation steps, a list of element declarations referencing the type definition given, a list of types that are derived from this type, and the set of elements that may appear in this type's model group.

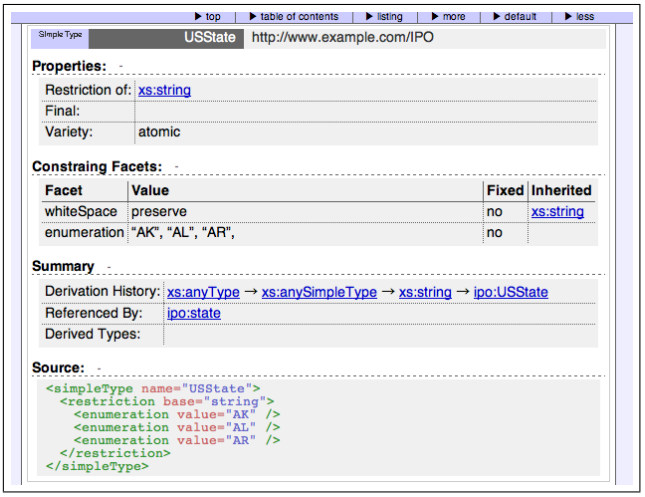
The latter list might seem hard to be determined, but using SCX, only two lines of codes are required:

```
<xsl:apply-templates
 select="scx:content-type//scf:element-declaration(.)"
 mode="listing"/>
<xsl:apply-templates
select="scx:content-type//scx:wildcard" mode="listing"/>
```

This might also give an idea how easy custom template rules for alternative documentation formats can be written.

One of the drawbacks of the verbosity of the transfer syntax is that many structures are difficult to understand at first sight. For example, in the syntax of DTDs, model groups are much more concisely defined for the human reader. X2Doc therefore provides a DTD-like notation of content models.

The last figure is the documentation for a simple type definition. Here, the list of constraining facets also contain a hyperlink to the simple type definition that originally defined the respective facet. Inherited facets are notoriously hard to track in the transfer syntax.



4. CONCLUSIONS

The main advantage of XSLT-based generation of documentation from XML Schemas is high extensibility, versatile configuration, and the interoperability with other XML-based formats and technologies. The X2Doc framework presented here exemplifies all these properties.

X2Doc is currently a work in progress; at present, we work on the completion of the core stylesheets to cover all schema components, and the extension of the hyperlinked connectivity. The next steps might be the addition of further output formats (using XSL-FO) and the generation of graphics (using SVG).

As XML Schema becomes increasingly used in large and heterogeneous projects, e.g. in the context of XML pipelines, the need for configurable and extensible documentation grows as well. We expect X2Doc to be a useful tool in this context; in particular in connection with SCX, which is ideally suited to support analysis and transformation of XML Schemas, which can then be documented using X2Doc.

5. REFERENCES

[1] Henry S. Thompson, David Beech, Murray Maloney, and Noah Mendelsohn. XML Schema Part 1: Structures Second Edition. World Wide Web Consortium, Recommendation REC-xmlschema-1-20041028, October 2004.

[2] Erik Wilde and Felix Michel. XML-Based XML Schema Access. In *Poster Proceedings of the 16th International World Wide Web Conference*, Banff, Alberta, May 2007. ACM Press.