Positioning Additional Constraints

Issue

XML Schema 1.1 allows additional constraints to be imposed on elements and attributes, above and beyond the constraints specified by their data type. Where should those additional constraints be positioned? Does an additional constraint on a descendant belong with the descendant or with an ancestor?

Example

Consider this book store document:

```
<?xml version="1.0"?>
<BookStore storename="Barnes and Noble">
        <Book>
                <Title>Don't Make Me Think</Title>
                <Author>Steve Krug</Author>
                <Date>2006</Date>
                <ISBN>0-321-34475-8</ISBN>
                <Publisher>New Riders</Publisher>
        </Book>
</BookStore>
```

The document is used by a community that has this business rule:

Business Rule

The value of Publisher depends on the store:

If the store is Barnes and Noble then the value of Publisher can be either Wrox Press or New Riders

If the store is Borders then the value of Publisher can be either Norton Press or friendsofed

Suppose that the XML Schema declares the Publisher element like this:

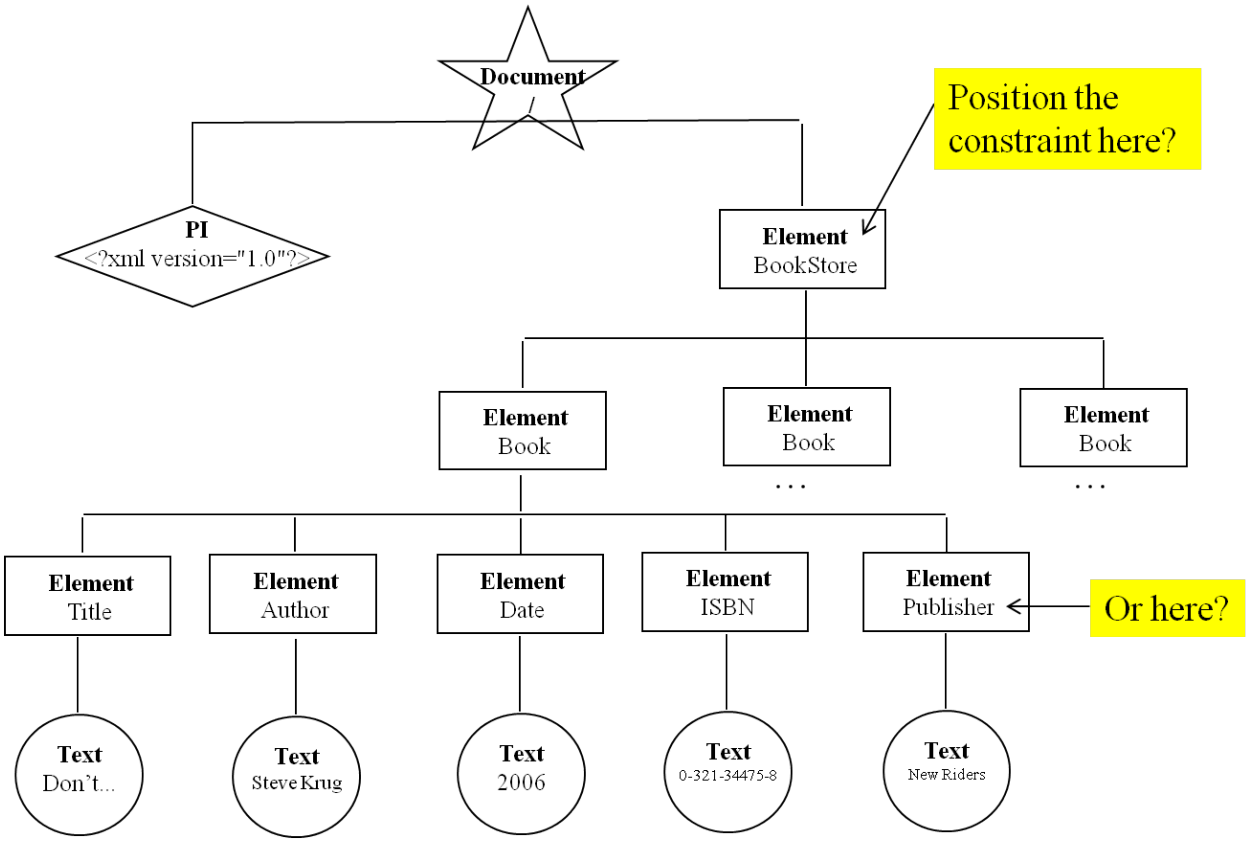
```
<element name="Publisher" type="string" />
```

Clearly the business rule imposes an *additional constraint* on the value of Publisher, above and beyond the constraint specified by its data type.

Given the business rule constraint, the following document is invalid (because 'friendsofed' is an invalid Publisher value for the store 'Barnes and Noble'):

```
<?xml version="1.0"?>
<BookStore storename="Barnes and Noble">
        <Book>
                <Title>DOM Scripting</Title>
                <Author>Jeremy Keith</Author>
                <Date>2005</Date>
                <ISBN>1-59059-533-6</ISBN>
                <Publisher>friendsofed</Publisher>
        </Book>
</BookStore>
```

Does the business rule constraint belong with the Publisher element or the BookStore element?



Sometimes the answer may be arrived at by asking these questions:

When the business rule constraint is violated, what is invalid? Is BookStore invalid? Or is Publisher invalid? Is the business rule a statement about what are valid BookStores? Or, is the business rule a statement about what are valid values of Publisher given its context?

In this example both viewpoints are equally plausible—the BookStore has invalid Publisher data and the Publisher has invalid data given that it is in a Barnes and Noble document. If both viewpoints are equally plausible, how does one determine where to position the business rule constraint? Here is something to consider: does the position of the business rule constraint influence the design of the XML vocabulary? Yes it does. Let's see.

Impact of the Position of a Constraint on XML Design

If you decide that the business rule is a statement about what are valid BookStores then, when you design your XML Schema, you should position an <assert> element on the BookStore element declaration:

```
<?xml version="1.0"?>
<BookStore storename="Barnes and Noble"> Assert: Book/Publisher = ('Wrox Press',
                                                                  'New Riders')
        <Book>
                <Title>Don't Make Me Think</Title>
                <Author>Steve Krug</Author>
                <Date>2006</Date>
                <ISBN>0-321-34475-8</ISBN>
                <Publisher>New Riders</Publisher>
        </Book>
</BookStore>
```

The XML Schemas are shown at the bottom of this document.

Rather than specifying the store name using an attribute value, the store name could be the name of the element. This is an equally fine XML design:

```
<?xml version="1.0"?>
<BarnesAndNoble> Assert: Book/Publisher = ('Wrox Press',  'New Riders')
        <Book>
                <Title>Don't Make Me Think</Title>
                <Author>Steve Krug</Author>
                <Date>2006</Date>
                <ISBN>0-321-34475-8</ISBN>
                <Publisher>New Riders</Publisher>
        </Book>
</BarnesAndNoble>
```

If you decide that the business rule is a statement about what are valid values of Publisher given its context then, when you design your XML Schema, you should position <alternative> elements in the Publisher element declaration:

```
<?xml version="1.0"?>
<BookStore storename="Barnes and Noble">
        <Book>
                <Title>Don't Make Me Think</Title>
                <Author>Steve Krug</Author>
                <Date>2006</Date>
                <ISBN>0-321-34475-8</ISBN>
                <Publisher>
                   Alternative: if @storename='BarnesAndNoble' then
                                text() = ('Wrox Press', 'New Riders')
                   Alternative: if @storename='Borders' then
                                text() = ('Norton Press', 'friendsofed')
                     New Riders
                </Publisher>
        </Book>
</BookStore>
```

Notice that the <alternative> elements make use of the storename attribute, which is an attribute of the ancestor BookStore element. The storename attribute must be declared "inheritable" for descendant elements to use it in their <alternative> elements.

XML Schema 1.1 does not permit <assert> elements to "look up" to ancestors. Further, only attributes (not elements) can be "inherited" and used in <alternative> elements. Consequently, if you decide that the business rule is a statement about what are valid values of Publisher given its context then the following XML design would make it impossible to implement the business rule:

```
<?xml version="1.0"?>
<BarnesAndNoble>
        <Book>
                <Title>Don't Make Me Think</Title>
                <Author>Steve Krug</Author>
                <Date>2006</Date>
                <ISBN>0-321-34475-8</ISBN>
                <Publisher>New Riders</Publisher>
        </Book>
</BarnesAndNoble>
```

Lesson Learned

Where you position your additional constraints may dictate your XML design.

Lesson Learned

Making liberal use of inheritable attributes allows descendants to make decisions based on ancestor attribute values.

Advantages and Disadvantages

If it is equally plausible for an additional constraint to belong with a descendent element or an ancestor element then here are the factors to consider:

Advantages

1. By positioning the additional constraint on an ancestor element, there are multiple XML design options available. The example above showed that the store name could be specified as an attribute value or as an element.
2. By positioning the additional constraint on the descendent, that descendant may be self-contained in terms of describing its constraints. There is no need to look to ancestor elements to see if any of them are imposing additional constraints (a.k.a., *action at a distance*). The descendant may be amenable to being copied and dropped into other XML Schemas, with all its constraints carried along.

Disadvantages

1. By positioning the additional constraint on an ancestor element, the ancestor element is exerting *action at a distance*. Examining the descendent in isolation will not reveal its true set of constraints.
2. By positioning the additional constraint on the descendent, there are restrictions on XML design. Namely, if ancestor values are needed then they must be attributes. The example above showed that the store name must be specified as an inheritable attribute.

Acknowledgements

The following people contributed to this document:

- Mike Brenner
- Edward Cheney
- Roger Costello
- Mukul Gandhi
- Stephen Green
- Michael Kay
- Noah Mehndelsohn
- Liam Quinn
- Michael Sperberg-McQueen
- Andrew Welch

XML Schemas

Here is the XML Schema that positions the business rule constraint on the root element:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified">

    <xs:element name="Book">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Title" type="xs:string"/>
                <xs:element name="Author" type="xs:string"/>
                <xs:element name="Date" type="xs:gYear"/>
                <xs:element name="ISBN" type="xs:string"/>
                <xs:element name="Publisher" type="xs:string" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="BarnesAndNoble">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Book" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:assert test=".//Publisher = ('Wrox Press',
                                             'New Riders')"/>
        </xs:complexType>
    </xs:element>

    <xs:element name="Borders">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Book" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:assert test=".//Publisher = ('Wrox Press',
                                             'New Riders')"/>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

Here is the XML Schema that positions the business rule constraint on the Publisher element:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified">


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

```
                <xs:element name="Author" type="xs:string"/>
                <xs:element name="Date" type="xs:gYear"/>
                <xs:element name="ISBN" type="xs:string"/>
                <xs:element name="Publisher">
                    <xs:alternative test="@storename eq
                                          'Barnes and Noble'">
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                               <xs:enumeration value="Wrox Press" />
                               <xs:enumeration value="New Riders" />
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:alternative>
                    <xs:alternative test="@storename eq 'Borders'">
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                               <xs:enumeration value="Norton Press" />
                               <xs:enumeration value="friendsofed" />
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:alternative>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="BookStore">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Book" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="storename" type="xs:string"
                          inheritable="true" />
        </xs:complexType>
    </xs:element>

</xs:schema>
```