XPath Expressions and Namespaces

DFDL includes an expression language based on XPath.

Turns out that there are some "issues" with XPath 1.0 and XML namespaces.

No XPath way to bind prefixes

If you have XML data that has a namespace, such as:

```
<data xmlns="urn:someNamespaceOrOther"><a>75</a></data>
```

Well, it turns out that there is ***no standard way*** to get this XPath to work:

```
/data/a
```

There is no XPath 1.0-standard mechanism for associating a namespace with a prefix (or with the default namespace). By that I mean there is nothing you can put in the path expression itself to specify the namespaces. Such mechanisms are available on APIs specific to the XPath-1.0 processor.

XPath-1.0 processors typically provide a way to bind namespace external to the XPath 1.0 expression. For example, in JAXB, namespaces are bound to prefixes (and to the default namespace) using the NamespaceContext method.

See XPath.html and the discussion on QNames in the class overview. Also see NamespaceContext.html for details on how to bind the default namespace.

Daffodil constructs a namespace context object to provide resolution of prefixes for XPath.

The way this works, is that whenever we have a DFDL expression, we also have the encapsulating XML object that contained it. The namespace scope of that XML object defines what the prefixes in the expression mean. So we grab the namespace scope from the XML object, and massage it slightly to what Daffodil wants to provide those definitions.

The 'xmllint' command (and libxml2 library on which it is based), binds names in XML Schema paths (such as in xs:key and xs:unique selectors) and applies a default namespace if one is defined, and the name is unqualified.

XSLT however, does NOT do this. Any default namespace binding is ignored in XSLT match and selector paths.

Some Surprises

Managing namespace prefixes can be a little bit tricky.

Suppose you write a schema but use xmlns="http://www.w3.org/2001/XMLSchema".

This allows you to avoid the "xs:" prefix on all the XML Schema elements.

However, it will require all expressions to use an explicit prefix on every path step, since an unprefixed path step will imply the XML Schema namespace. This shouldn't be surprising. All QNames in the schema referring to other elements, types, groups, defineFormat or defineEscapeScheme also must be prefixed in this case. But path expressions it is very easy to remember that one must prefix all the steps as those are QNames as well.

If you really want to write expressions like /a/b/c, i.e., without any prefixes on the steps, then you have to use the default namespace the same as the schema's target namespace.