2.4 Document Subsets

Some applications require the ability to create a physical representation for an XML document subset (other than the one generated by default, which can be a proper subset of the document if the comments are omitted). Implementations of XML canonicalization that are based on XPath can provide this functionality with little additional overhead by accepting a node-set as input rather than an octet stream.

The processing of an element node \( E \) MUST be modified slightly when an XPath node-set is given as input and the element's parent is omitted from the node-set. This is necessary because omitted nodes SHALL not break the inheritance rules of inheritable attributes \[C14N-Issues]\ defined in the xml namespace.

[Definition:] **Simple inheritable attributes** are attributes that have a value that requires at most a simple redeclaration. This redeclaration is done by supplying a new value in the child axis. The redeclaration of a simple inheritable attribute \( A \) contained in one of \( E \)'s ancestors is done by supplying a value to an attribute \( Ae \) inside \( E \) with the same name. Simple inheritable attributes are xml:lang and xml:space.

The method for processing the attribute axis of an element \( E \) in the node-set is hence enhanced. All element nodes along \( E \)'s ancestor axis are examined for the nearest occurrences of simple inheritable attributes in the xml namespace, such as xml:lang and xml:space (whether or not they are in the node-set). From this list of attributes, any simple inheritable attributes that are already in \( E \)'s attribute axis (whether or not they are in the node-set) are removed. Then,
lexicographically merge this attribute list with the nodes of
E's attribute axis that are in the node-set. The result of
visiting the attribute axis is computed by processing the
attribute nodes in this merged attribute list.

The xml:id attribute is not a simple inheritable attribute and
no processing of these attributes is performed.

The xml:base attribute is not a simple inheritable attribute
and requires special processing beyond a simple
redeclaration. Hence the processing of E's attribute axis
needs to be enhanced further. A "join-URI References"
function is used for xml:base fix up. It incorporates xml:base
attribute values from omitted xml:base attributes and
updates the xml:base attribute value of the element being
fixed up, as follows.

An xml:base fixup is performed on an element E as follows.
Let E be an element in the node set whose ancestor axis
contains successive elements En...E1 (in reverse document
order) that are omitted and E=En+1 is included. (It is
important to note that En..E1 is for contiguously omitted
elements, for example only e2 in the example in section 3.8.)
The fix-up is only performed if at least one of E1 ... En had
an xml:base attribute. In that case let X1 ... Xm be the values
of the xml:base attributes on E1 ... En+1 (in document order,
from outermost to innermost, m <= n+1). The sequence of
values is reduced in reverse document order to a single
value by first combining Xm with Xm-1, then the result with
Xm-2, and so on by calling the "join-URI References"
function until the new value for E's xml:base attribute
remains. The result may also be null or empty (xml:base=""),
in which case xml:base MUST NOT be rendered.
Note that this xml:base fixup is only performed if an element with an xml:base attribute is removed. Specifically, it is not performed if the element is present but the attribute is removed.

The join-URI-References function takes an xml:base attribute value from an omitted element and combines it with other contiguously omitted values to create a value for an updated xml:base attribute. A simple method for doing this is similar to that found in sections 5.2.1, 5.2.2 and 5.2.4 of RFC 3986 with the following modifications:

- Perform RFC 3986 section 5.2.1. "Pre-parse the Base URI" modified as follows.
  - The scheme component is not required in the base URI (Base). (i.e. Base.scheme may be null)

- 5.2.4. "Remove Dot Segments" is modified as follows:
  - Keep leading "./" segments
  - Replace multiple consecutive "/" characters with a single "/" character.
  - Append a "/" character to a trailing "..

- Perform RFC 3986 section 5.2.2. "Transform References" modified as follows to ignore the fragment part of R
  - After parsing R set R.fragment = null

Then, lexicographically merge this fixed up attribute with the nodes of E’s attribute axis that are in the node-set. The result of visiting the attribute axis is computed by processing the attribute nodes in this merged attribute list.
Attributes in the XML namespace other than xml:base, xml:id, xml:lang, and xml:space MUST be processed as ordinary attributes.

3.8 Document Subsets and XML Attributes

```
<!DOCTYPE doc [
<!ATTLIST e2 xml:space (default|preserve) 'preserve'>
<!ATTLIST e3 id ID #IMPLIED>]

<doc xmlns="http://www.ietf.org" xmlns:w3c="http://www.w3.org" xml:base="something/else">
  <e1>
    <e2 xmlns="" xml:id="abc" xml:base="bar/">
      <e3 id="E3" xml:base="foo"/>
    </e2>
  </e1>
</doc>
```

Document Subset Expression
```
//. | //@* | //namespace::*
[ self::ietf:e1 or (parent::ietf:e1 or count(id("E3")|ancestor-or-self::node()) ]
```

Canonical Form
```
<e1 xmlns="http://www.ietf.org" xmlns:w3c="http://www.w3.org" xml:base="something/else">
  <e3 xmlns="" xml:id="E3" xml:base="something/bar/foo" xml:space="preserve"></e3>
</e1>
```

Demonstrates:
- xml:id not inherited.
- simple inheritable XML attribute inherited (xml:space)
- xml:base fixup performed

Appendix A

Remove text in Example A up to and including "Some Examples", retain table. Add following text before tables.
The following informative table outlines example results of the modified Remove Dot Segments algorithm described in Section 2.4.