**3GPP TSG-SA5 Meeting #145-e *S5-225473***

**e-meeting, 15th Aug 2022 - 24th Aug 2022**

**Source: Nokia, Nokia Shanghai Bell**

**Title: pCR 28.831 Add XPath as potential solution**

**Document for: Approval**

**Agenda Item: 6.8.2.4 - FS\_eSBMAe\_WoP#4**

# 1 Decision/action requested

***The group is requested to discuss and approve the pCR below***

# 2 References

[1] 3GPP TS 28.831: " Management and orchestration; Study on basic Service-Based Management Architecture (SBMA) enabler enhancements"

# 3 Rationale

None.

# 4 Detailed proposal

The following changes are proposed for TR 28.831[1].

|  |
| --- |
| **Begin of modifications** |

# 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] XML Path Language (XPath), Version 1.0, W3C Recommendation 16 November 1999 (Status updated October 2016), (https://www.w3.org/TR/1999/REC-xpath-19991116/)

[3] XML Path Language (XPath) 2.0, W3C Recommendation 14 December 2010 (Link errors corrected 3 January 2011; Status updated October 2016), (https://www.w3.org/TR/xpath20/)

[4] XML Path Language (XPath) 3.1, W3C Recommendation 21 March 2017 (<https://www.w3.org/TR/xpath-31/>)

[x] XML Information Set (Second Edition), W3C Recommendation 4 February 2004 (<https://www.w3.org/TR/xml-infoset/>)

[y] XQuery and XPath Data Model 3.1, W3C Recommendation 21 March 2017 (<https://www.w3.org/TR/xpath-datamodel-31/>)

[z] RFC 7950: The YANG 1.1 Data Modeling Language

|  |
| --- |
| **Next modification** |

#### 4.2.5.2 XPath 1.0

##### 4.2.5.2.1 General considerations

XPath 1.0 [2] has been designed primarily to select one or more nodes of an XML document. However, XPath is a notatition that is not tied to XML and "operates on the abstract, logical structure of an XML document or JSON object, rather than its surface syntax.", as clarified in XML Path Language (XPath) 3.1 [4], clause 1. This is because an XPath expression operates on documents with an own XPath specific data model, and not on the XML or JSON surface syntax. This data model is specified in XPath 1.0 [2], clause 5. Note the data model for XPath 2.0 [3] and XPath 3.1 [4] is described in XQuery and XPath Data Model 3.1 [y].

Since an XPath expression works on an own data model, it is necessary to translate XML of JSON input documents into that data model.

For XML this translation is clearly defined by W3C. It is a two-step process. The mapping from the information in an XML document to the XML Information Set is described in XML Information Set [x]. The mapping from the XML Information Set to the XPath data model is described in XPath 1.0 [2], annex B. The mapping from YANG to XML is defined in RFC 7950 [z].

However, W3C did not define a clear mapping of a JSON document into the XPath data model, see the following clause for more details.

XPath models an XML or JSON document as a tree of nodes. NRM objects and NRM attributes are both mapped to a specific node type of the XPath data model, the element node. XPath 1.0 does hence not distinguish between element nodes representing NRM objects and element nodes representing NRM attributes.

Furthermore, XPath 1.0 does not distinguish between child element nodes that represent NRM attributes and those child nodes (at the same level in the the tree hierarchy) that represent name contained objects.

These two observations have some important implications when using XPath 1.0 for selecting NRM objects:

The first implication is that it is not possible to select all NRM attributes of an NRM object unless some measures are taken. Assume the XPath expression "SubNetwork/\*", it selects all attributes of SubNetwork, but also the name contained objects. Fortunately, the attributes are embedded in an "attributes" container in the YANG and JSON defined NRMs. Therefore, it is easy to construct an XPath expression selecting all attributes of an object. The XPath expression is "SubNetwork/attributes" (when the "attributes" element node and the attribute element nodes are selected) or "SubNetwork/attributes/\*" (when only the attribute element nodes are selected).

The second implication is that when selecting the child objects of an object, for example with the XPath expression "SubNetwork/ManagedElement", which selects all "ManagedElement" childs of "SubNetwork", then also the objects name contained by the "ManagedElement" childs are selected, and so forth. In other words, the child axis works based on element nodes and not on NRM objects as desired. XPath 1.0 does not offer any solution for this problem. Only the attributes container may be used again to select only the attributes of the "ManagedElement" childs, the XPath expression to select the "ManagedElement" childs is hence "SubNetwork/ManagedElement/attributes".

Note XPath 1.0 is verbose and does not allow for compact expressions in many cases. For example, to select a subset of all attributes of an object, each attribute needs to be selected individually with the complete location path:

/SubNetwork/attributes/attrA | /SubNetwork/attributes/attrB

And when multiple objects shall be selected based on object instance names, then the complete location path and predicate needs to be repeated for each instance

//\*[objectInstance="DN1"]/attributes | //\*[objectInstance= "DN2"]/attributes

An XPath expression is evaluated within a context. The context consists of:

* a node (the context node)
* a pair of non-zero positive integers (the context position and the context size)
* a set of variable bindings
* a function library
* the set of namespace declarations in scope for the expression

The context needs to be clearly defined.

**XPath expression examples (assuming the presence of an "attributes" container in the info model):**

Select the specified base object

/\*/attributes

Select all objects in the tree starting at the specified base object:

//attributes

Select all objects on a specific level below a specified base object, in this case the second level below the base object:

/\*/\*/\*/attributes

Select the objects starting from a specified base object down to and including the objects on a specific level below the base object:

/\*/attributes | /\*/\*/attributes | /\*/\*/\*/attributes

Select objects based on their object instance identifiers in the tree starting at the base object:

//\*[objectInstance="DN1"]/attributes | //\*[objectInstance= "DN2"]/attributes

Select objects based on their object class

/SubNetwork/attributes

/SubNetwork[id="SN1"]/ManagedElement/attributes

//\*[objectClass="NtfSubscriptionControl"]/attributes

Select specific attributes and attribute fields:

<objectSelector>/attrA | <objectSelector>/attrB/subAttrB1

<objectSelector>/attributes/attrA | <objectSelector>attributes/attrB/subAttrB1

Select objects and attributes based on predicates (conditions):

<objectSelector>[<predicate>]

<objectSelector>/attributes<attributesSelector>[<predicate>]

#### 4.2.5.3 XPath 2.0

XPath 2.0 [3] introduces sequence expressions which help to render Xpath expressions more compact. For example, multiple attributes of an object can be selected by

/SubNetwork/(attrA, attrB)

and multiple objects based on instance names with

//\*[objectInstance=("DN1", "DN2")]/attributes

Other concepts added in XPath 2.0 like quantified expressiuons provide no obvious value for the use cases analysed.

#### 4.2.5.4 XPath 3.1

Concepts added in XPath 3.1 [4] provide no obvious value for the use cases analysed.

*Editor's note: Functions to deal with JSON arrays and maps may need further analysis.*

|  |
| --- |
| **End of modifications** |