**3GPP TSG-SA5 Meeting #145-e *S5-225475***

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

**Source: Nokia, Nokia Shanghai Bell**

**Title: pCR 28.831 Add special XPath considerations for JSON and YANG**

**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

[w] XForms 2.0 (https://www.w3.org/community/xformsusers/wiki/XForms\_2.0)

[a] 3GPP TS 32.158: "Management and orchestration; Design rules for REpresentational State Transfer (REST) Solution Sets (SS)"

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

##### 4.2.5.2.2 Special considerations for JSON

As stated in the previous clause W3C did not provide a real mapping from JSON to the XPath data model. It is possible though to use a mapping from JSON to XML defined in XForms2.0 [w], clause 5.2.2. The resulting XML document can then be mapped to the XPath data model.

Note, XForms2.0 is a W3C draft from 2010 and not a W3C standard. The produced XML document contains in XML attributes with information about the original JSON data taypes. JSON arrays are mapped in a very specific way to XML (see example below). These mapping rules preserve all information for mapping the XML document again back to the original JSON document.

For example, the JSON snippet

"load": [0.31, 0.33, 0.32]

is mapped according to XForms2.0 to

<load type="array">

<\_type="number">0.31</\_>

<\_type="number">0.33</\_>

<\_type="number">0.32</\_>

</load>

Other best practices propose to map the JSON snippet as follows:

<load>0.31</load>

<load>0.33</load>

<load>0.32</load>

There is hence not one and only one standard specifying the mapping from a JSON document to an XML document. One could argue that the XML document is anyway a conceptual document only that is not visible on the wire and used only internally in the MnS producer helping to apply the XPath expression to a JSON document. However, the exact way how JSON is mapped to XML has implications on the XPath expression to selectm nodes.

For example, when the first mapping method is used, the XPath expression snippet to select the first array item is

/load/\_[1]

When the second mapping method is used, the XPath expression snippet is

/load/[1]

For a really interoprable solution, the MnS consumer needs to know the exact way the MnS producer maps JSON to XML. This means in turn, that the mapping method needs to be standardized by SA5.

A second option is to provide a mapping from the information elements in a JSON document to the XPath data model directly without an XML mapping in-between. Such a mapping is not provided at all by W3C.

Here after, a proposal to directly map a JSON document to the XPath data model:

* The JSON document is mapped to the root node.
* A JSON value, which is a string, a number, or one of the tree literals "false", "true" or "null" is mapped to a text node.
* A JSON object is mapped to one element node. The name of the element node is equal to the name of the name/value pair whose value is the JSON object. The parent of this element node is the containing element node. The child element nodes are the members (name/value pairs) of the JSON object.
* Each JSON array item is mapped to one element node. The name of the element node is equal to the name of the name/value pair whose value is a JSON array. The parent of these element nodes is the containing element node.

In case XPath is chosen as node selection language, then it is necessary to specify in normative fashion either the JSON to XML mapping, or the JSON to XPath data model mapping.

Now we will look at if due to some inhereant properties of JSON all XPath concepts are applicable, when the original document from which nodes are selected, is a JSON document.

A JSON object is an unordered collection of zero or more name/value pairs. This is why the concept of document order (clause 5 of XPath 1.0 [2]) is not applicable when an XPath expression is applied to an XPath data model that was generated from a JSON document. This has the following implications:

* The following axes cannot be used: following, following-sibling, preceding, preceding-sibling.
* The following functions cannot be used: position, local-name, namespace-uri, name.

JSON does not have a namespace concept. This is why XPath concepts related to name spaces are not applicable. This has the following implications:

* The following axes cannot be used: namespace
* The following functions cannot be used: local-name, namespace-uri, name

JSON does not have a concept similar to XML attributes. This is why XPath concepts related to attributes are not applicable. This has the following implications:

* The following axes cannot be used: attribute

Out of the seven node types specified in XPath 1.0 [2] only the root node, element node and text node are used. The attribute nodes, namespace nodes, processing instruction nodes and comment nodes are not used.

The name of the root element node is the class name of the base object. The base object is the node that contains the "NtfSubscriptionControl" instance (that in turn has an attribute whose value is the XPath expression).

Note that the root element node (document element) is not the same as the root node. The root element node is the mapping of the top-level name/value pair in the JSON document, whereas the root node is the (conceptual) parent of that object. The root node is the mapping of the JSON document.

When the value of the top-level name/value pair is an array, which is always the case for JSON defined NRMs in SA5, this array can contain only one item, which is the base object, in the special context of notification subscription. Considerations on howe to handle the case where a top-level array can contain multiple items are hence not required.

For example, assume the information model described by annex A.1 in TS 32.158 [a]. Further assume that the "ManagedElement" with the id "ME1" contains a "NtfSubscriptionControl" instance for which a node selection XPath expression shall be constructed. The base object is the "ManagedElement" with the id "ME1". The JSON document, to which the XPath expression is applied to, is as follows:

|  |
| --- |
| {  "ManagedElement": [  {  "id": "ME1",  "objectClass": "ManagedElement",  "objectInstance": "SubNetwork=SN1,ManagedElement=ME1",  "attributes": {  "userLabel": "Berlin NW 1",  "vendorName": "Company XY",  "location": "TV Tower"  },  "XyzFunction": [  {  "id": "XYZF1",  "objectClass": "XyzFunction",  "objectInstance": "SubNetwork=SN1,ManagedElement=ME1,XyzFunction=XYZF1",  "attributes": {  "attrA": "xyz",  "attrB": 551  }  },  {  "id": "XYZF2",  "objectClass": "XyzFunction",  "objectInstance": "SubNetwork=SN1,ManagedElement=ME1,XyzFunction=XYZF2",  "attributes": {  "attrA": "abc",  "attrB": 552  }  }  ]  }  ]  } |

or

|  |
| --- |
| {  "id": "ME1",  "objectClass": "ManagedElement",  "objectInstance": "SubNetwork=SN1,ManagedElement=ME1",  "attributes": {  "userLabel": "Berlin NW 1",  "vendorName": "Company XY",  "location": "TV Tower"  },  "XyzFunction": [  {  "id": "XYZF1",  "objectClass": "XyzFunction",  "objectInstance": "SubNetwork=SN1,ManagedElement=ME1,XyzFunction=XYZF1",  "attributes": {  "attrA": "xyz",  "attrB": 551  }  },  {  "id": "XYZF2",  "objectClass": "XyzFunction",  "objectInstance": "SubNetwork=SN1,ManagedElement=ME1,XyzFunction=XYZF2",  "attributes": {  "attrA": "abc",  "attrB": 552  }  }  ]  } |

Since a well formed XML document has one and only one root elemet the first alternative is preferred. It contains the key "ManagedElement" that maps to the name of the XML root element. The XPath expression to identify the base object only is then

/MangedElement/attributes

or

/\*/attributes

##### 4.2.5.2.3 Special considerations for YANG

The mapping from YANG to XML is defined in RFC 7950 [z]. No special considerations are required.

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