Document Title	TPS_ARXMLSerializationRules: Complete Change Documentation 4.3.0 - 4.3.1
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	695

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.3.1

Table of Contents

1 TPS_ARXMLSerializationRules 3
1.1 Specification Item TPS_ASR_00013 3

1 TPS_ARXMLSerializationRules

1.1 Specification Item TPS_ASR_00013

Trace References:

RS_IOAT_00001

Content:

The AUTOSAR XML Schema location hint URI in the AUTOSAR XML description shall be the file name of a XML Schema document provided by AUTOSAR. This file name follows the pattern:

```
AUTOSAR_{number}.xsd
```

{major} , {minor} and {patch} correspond to the major, minor and patch number of the AUTOSAR revision the number} corresponds to the specific AUTOSAR release the given AUTOSAR XML Schema belongs to.

In particular no path shall be part of the AUTOSAR XML Schema location hint URI.

RfCs affecting this spec item between releases 4.3.0 and 4.3.1:

- RfC #76497: Schema versioning revisited

  Problem description:

  Until now, we've had a consistent naming scheme of the XML schema file:

  AUTOSAR_<major>-<minor>-<patch>.xsd

  This name would then be used in the xsi:schemaLocation of ARXML files to indicate the schema taken to create the file. Using the numbering schema it was always unambiguously possible to determine which schema was back-wards compatible to which other schema.

  This approach is now discontinued as the adaptive platform introduces a new versioning pattern that is only used for releases of the adaptive platform. This would lead to the following naming pattern for the schema file:

  AUTOSAR_<YYYY>-<MM>.xsd

  This approach breaks the easy way to determine whether a schema is back-wards compatible to other schemas and makes tool implementations more difficult

to maintain.

Therefore, WP-M came up with the idea to decouple the versioning of the XML schema from the versioning pattern of the AUTOSAR releases:

AUTOSAR_<number> where number has five digits and starts (to avoid confusion with existing releases) with the value 00042.

This RfC shall be used to finalize the discussion about the switch in versioning and document the change properly.

Although the RfC could only be raised against the schema file of the classic platform, the general approach shall also extend to the adaptive platform.
–Last change on issue 76497 comment 1–

Agreed solution:

General approach:

1. we have a master for the Meta model
2. we have a replica for every Standard
3. the meta model is named by an integer number starting with 42
4. this number is incremented on every release of a standard
5. Changes essential for more than one standard are performed
directly in Master and synchronized to the standards
this ensures that these changes are tested in all standards.
it also ensures that no intermediate change of one standards appears in the other standards.
6. the meta models in the standards have the same number as the last merged master model plus an addendum
7. the system identifier of the schema reflects the standard in which the schema was built.
8. if a system identifier has an addendum indicating the standard: then it is a development build.
9. when a release is made: and the release is built (without an addendum)
10. after a release was made: the master number is incremented, synchronized with other standards and addendum attached

* the product branches shall be suffixed by the product-ID
* if a product is released only the changes of this product are merged to master
* those released changes are merged to the other products

For the following example, we assume the following release sequece

1. CP 4.3.0 –> 41
2. AP 2017-03 –> 42
3. CP 4.3.1 –> 43
4. AP 2017-10 –> 44

master AP CP XX

41 -> is as of 4.3.0

41 —> 42AP
————————> 42CP
————————————-> 42XX

now AP makes a release: AP-2017-03

42 <———— 42AP (merge)
43 ————-> 43AP (this could even be a copy)
————————> 43CP (merge)
————————————-> 43XX (merge)

now CP makes a release CP-R4.3.1

43 <————————— 43CP (merge)
44 —————————-> 44CP (could be a copy)
————> 44AP (merge)
————————————> 44XX (merge)

now Adaptive releases: AP-2017-10

44 <———— 44AP (merge)
45 ————-> 45AP (this could even be a copy)
————————> 45CP (merge)
————————————-> 45XX (merge)

Steps before a release
* the meta model of this standard release is synchronized with master (both directions!)
* the suffix is removed

* artifacts are generated

Steps after a release

* master number is incremented for upcoming development
* master is synchronized into all other products

This means in particular

42 contains AP 2017-03 CP 4.3.0 -> AP - Release

43AP contains AP 2017-0x CP 4.3.0
43CP contains AP 2017-03, CP 4.3.0+x

43 contains AP 2017-03, CP 4.3.1 -> CP - Release

44AP contains AP 2017-0x, CP 4.3.1
44CP contains AP 2017-03 CP 4.3.1+x

44 contains AP 2017-10, CP 4.3.1 -> AP - Release

45CP contains AP 2017-10, CP 4.3.1+x
45AP contains AP 201x-xx, CP 4.3.1

and so on. xxpp contains last common point "xx" and product changes "pp"

Implementation tasks
==========

Latex-Styles:

line 1983 has a hardcoded "." to compute the version number.
remove ".\{}docrevision in line 1983

TOOL_Scripts:

let generator for ar_autosar_metadata.tex generate \{}DocRelease4.3.1 as full version identifier

update tool config approach to support different versioning schemes between the standards
in particular the parameters:

* product.metamodeliteration
* product.previousmetamodeliteration
based on these parameters, we create the xsd system - identifier
* product.meta.release either takes the major version, or the "agile - version -id"

* product.meta.standardscoveredbyschema: The information about covered standards is to be maintained in the configuration of the standard in question

* Document the process in Wiki

Standards:

1. create the folder 26_Standards/80_Common
2. configure it such that schema-generation works
3. update config of every other standard

in _readme_templ.erb https://svn.autosar.org/repos/work/26_Standards/10_CP_R4/01_Sources/0
https://svn.autosar.org/repos/work/26_Standards/30_AP_R1/01_Sources/MMOD_XMLSchema_2
==========

The schema releases match to the AUTOSAR standards as follows:

AP 17-03 AUTOSAR_00042
AP 17-10 AUTOSAR_00043
CP 4.3.1 AUTOSAR_00044

TPS_ARXMLSerializationRules
==========

Put some explanatory text to the document somewhere in chapter 2.2.4

–8<—-8<—-8<—-8<—-8<—-8<—-8<—-8<—-8<–
Traditionally, AUTOSAR has implemented a three-element version scheme consisting of major, minor, and patch version. Versions specified this way have been used in ARXML files as part of the definition of the xsi:schemaLocation, for example:

xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_4-2-2.xsd"

With the advent of the AUTOSAR adaptive platform, AUTOSAR decided to

implement a different versioning scheme for the releases of the adaptive platform (the classic platform would just keep the existing approach to versioning). This new version scheme for the adaptive platform consists of just two elements, the year and month of release.

The original approach was to simply use the two-element scheme of the adaptive releases also for the definition of the xsi:schemaLocation for ARXML files containing models for the adaptive platform.

xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_2017-03.xsd"

Over time, this approach would have created a hard-to-disentangle history of three-element and two-element values for xsi-schema location and it would have been hard to guess which releases of the AUTOSAR XML Schema were actually backwards-compatible to a given ARXML file.

In order to mitigate the problem, AUTOSAR also decided to invent a completely new versioning scheme for the schema releases, independent of whether the individual schema release would be triggered by the AUTOSAR classic platform or the AUTOSAR adaptive platform.

The new versioning scheme for being used in the xsi:SchemaLocation foresees the existence of just one element, a positive number that is increased with every AUTOSAR release, whether the release focuses on the classic or adaptive platform does not matter.

xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00042.xsd"

Each value of the one element in the xsi:schemaLocation can be unambiguously identified with a specific AUTOSAR release. Plus, it is still easily possible to understand the backwards compatibility status of a given ARXML file.

The XML schema contains the latest releases of the AUTOSAR standards. This means that there is no dedicated AUTOSAR XML schema that contains only model elements of AUTOSAR classic or adaptive platform.

Insert attached figure.
–8<—-8<—-8<—-8<—-8<—-8<—-8<—-8<—-8<–

Fix example 2.6 to contain the new value of xsi:schemaLocation

Fix [TPS_ASR_00013]

[TPS_ASR_00013] Pattern for AUTOSAR XML Schema location hint URI d
The AUTOSAR XML Schema location hint URI in the AUTOSAR XML description shall be the file name of a XML Schema document provided by AUTOSAR. This file name follows the pattern:
AUTOSAR_number.xsd
number corresponds to the specific AUTOSAR release the given AUTOSAR XML Schema belongs to.
In particular no path shall be part of the AUTOSAR XML Schema location hint URI. c(RS_IOAT_00001)

Fix listings 2.3 and 2.4 accordingly
–Last change on issue 76497 comment 20–

BW-C-Level:

Application	Specification	Bus
1	1	1