genHdr and associated scripts functionality

[1. FEATURES 2](#_Toc421293547)

[1.1 VALIDATION 2](#_Toc421293548)

[1.2 CONTROL XML GENERATION 2](#_Toc421293549)

[1.3 DATA XML GENERATION 3](#_Toc421293550)

[1.4 MODELLED XML GENERATION 3](#_Toc421293551)

[1.5 REGENERATING ORIGINAL XML BACK FROM ID ADDED XML 3](#_Toc421293552)

[1.6 MERGING 3](#_Toc421293553)

[2. STEPS TO GENERATE TARGET XMLS 4](#_Toc421293554)

[3. FLOW CHART 6](#_Toc421293555)

# FEATURES

## 1.1 VALIDATION

Validates xml against xsd. You can validate independently or while generating id added control xmls

EXAMPLE:

python genHdrnMergeXML.py -v validate.xsd IP\_control.xml

## 1.2 CONTROL XML GENERATION

Takes control xmls as input and

1. Generates IDs
2. Encodes all the attributes into ‘Flag’ attribute
3. Generates ‘Enum’ attribute by taking tr181 datatype xml as reference
4. Merges all the xmls if multiple xmls are provided
5. Generates header file

EXAMPLE:

1. python genHdrnMergeXML.py -c tr-181-2-8-0\_dataType.xml 1\_control.xml 2\_control.xml -o 1\_2\_control.xml 🡪 generates 1\_2\_control\_id.xml, 1\_2\_control\_id.h
2. python genHdrnMergeXML.py -c default PPP\_control.xml 🡪 **generates PPP\_control\_id.xml, PPP\_control\_id.h, PPP\_data\_id.xml**

**NOTE: The second example with ‘–c default’ option works on following assumptions**

1. **Script is being executed from the place where it is located**
2. **Only one file is given as input**
3. tr-181-2-8-0\_dataType.xml **is available in its location**
4. **if appropriate data xml (PPP\_data.xml in this case) is also available in the path where control xml is present, id added data xml will also be generated**

**These assumptions enable user of the script to generate, id added control xmls, header files, id added data xmls in one go**

## 1.3 DATA XML GENERATION

Takes data xmls as input (with reference to its id added control xml counterpart) and

1. Syncs **‘Id’ and ‘Flag’** attribute of each node with that of the control xml
2. Merges xmls if multiple xmls are provided

EXAMPLE:

1. python genHdrnMergeXML.py -d WiFi\_control\_id.xml WiFi\_data.xml –o Wifi\_data\_id .xml 🡪 generates Wifi\_data\_id.xml
2. python genHdrnMergeXML.py –d default WiFi\_data.xml 🡪 **generates WiFi\_data\_id.xml**

## 1.4 MODELLED XML GENERATION

Takes contro\_id/data\_id xml as input with reference to control/data model xml and modifies them as model xml specifies

EXAMPLE:

python genModeledXMLs.py IP\_control\_id.xml -m IP\_ctrl\_model\_EL.xml -o IP\_control\_id.xml 🡪 generates IP\_control\_id\_modeled.xml

## 1.5 REGENERATING ORIGINAL XML BACK FROM ID ADDED XML

Takes id added control xml as input and generates original control xml

EXAMPLE:

python genModeledXMLs.py –p default PPP\_control\_id.xml 🡪 generates PPP\_control.xml

## 1.6 MERGING

1. Takes input files and merges all the nodes as per their hierarchy

EXAMPLE:

python merge.py lan\_control\_id.xml network\_control\_id.xml -o target\_control.xml

# STEPS TO GENERATE TARGET XMLS

1. Take few service xmls. E.g IP\_control.xml, PPP\_control.xml
2. Use db tool gui to edit attributes, add vendor extension objects etc

E.g Change min entries to IPV4, add vedor extension object in PPP\_control.xml, add vendor extension parameter to IP\_control.xml

1. Use dbtool to generate data xmls
2. Manually add one more IPv4 instance to IP\_data.xml

NOW WE HAVE **IP\_control.xml, IP\_data.xml, PPP\_control.xml and PPP\_data.xml** for genHdr related scripts to operate on

1. Generate id added control xmls and header files

python genHdrnMergeXML.py PPP\_control.xml -c tr-181-2-8-0\_dataType.xml -o PPP\_control\_id.xml 🡪 generates PPP\_control\_id.xml

python genHdrnMergeXML.py IP\_control.xml -c tr-181-2-8-0\_dataType.xml -o IP\_control\_id.xml 🡪 generates IP\_control\_id.xml

1. Generate ‘id added data xmls’ by taking its’ id added control xml’ counterpart into reference

python genHdrnMergeXML.py PPP\_data.xml -d PPP\_control\_id.xml -o IP\_data\_id.xml 🡪 generates PPP\_data\_id.xml

python genHdrnMergeXML.py IP\_data.xml -d IP\_control\_id.xml -o IP\_data\_id.xml 🡪 generates IP\_data\_id.xml

1. Generate modelled control xmls

python genModelledXML.py IP\_control\_id.xml -m IP\_ctrl\_model\_EL.xml -o IP\_control\_id\_modeled.xml 🡪 generates IP\_control\_id\_modeled.xml

1. Merge modelled xmls

python merge.py IP\_control\_id\_modeled.xml PPP\_control\_modeled.xml –o target\_control.xml

python merge.py IP\_data\_id\_modeled.xml PPP\_data\_modeled.xml –o target\_data.xml

# FLOW CHARTd:\Users\TejaR\Desktop\genHdr and related scripts workflow.png