A Study for the Message Exchange of the EVSE Communication Controller using XML Schema Transformation

Koaunghi Un*†, Hyuksoo Jang*, Myongsoo Kim**, Hwimin Kim*

*Myongji University, 116 Myongji-ro, Cheoin-gu, Yongin-si Gyeonggi-do, Korea  
**KEPCO Research Institute, Korea Electric Power Corporation, 105 Munji-ro Yusung-gu, Daejeon 34056, Korea  
†koaunghi@gmail.com

Abstract
An EVSE and the EV exchange XML formatted communication messages with each other, whose structure is described by the V2G CI schema in the ISO/IEC 15118 international standard, and the EVSE and the power grid exchange messages based on SCL file with each other, whose structure is described by SCL schema in the IEC 61850 international standard. Because XML files can be restructured by XSLT, V2G messages can be transformed to/from SCL files using XSLT. In this study, the two schemas are analyzed and compared in order for the restructuring of the two differently structured XML instances. As a result, two XSL scripts can be produced for the transformation between XML files conforming ISO/IEC 15118 V2G CI schema and IEC 61850 SCL schema respectively to avoid manual data mapping which can happen in every application development cases.

Keyword: EV, EVSE, V2G communication, XSL transformation, EV2G-XT

I. INTRODUCTION

When charging EV according to the ISO/IEC 15118 international standard [8], the communication messages are written in form of XML files based on the V2G CI schema [9] in the ISO/IEC 15118 international standard and their encoded EXI files are exchanged between EV and EVSE. The communication data between EVSE and power grid are also stored in form of XML file based on the SCL schema [5] in the IEC 61850 international standard [4]. XML files can be validated by corresponding schema and the schema definition is also described in XML format. A XML document can be transformed to other XML document by XSL.

The EVSE is connected to EV on one hand and to power grid on the other hand. The 15118 server in the EVSE exchanges communication messages with EV and the 61850 server with power grid [1]-[3] as shown in Fig. 1. Because these messages and data are differently structured due to the different XSDs, the XML structure need to be transformed and the data need to be mapped with the proper XML elements and attributes. The restructuring and mapping can be automated by applying XSL script and XSL processor. The XSL processor denoted as EV2G-XT (EVSE V2G XSL Transformer) in Fig. 1 transforms 15118 XML file to 61850 XML file and vice versa as described in the 15118-to-61850.xsl script or 61850-to-15118.xsl script respectively. XSLT is a specification of W3C. Manual mapping between 15118 and 61850 XML data is unnecessary by applying the schema-level transformation modules. Even industrial application developers with short understanding of both international standards can implement an EVSE application with a proper W3C’s XSLT.

In this paper, the ISO/IEC 15118 and IEC 61850 XSDs are studied to enable the building of a schema-based XML transformation system between their instances. For the understanding of the XSDs, explanations of XSD language itself are made minimally when needed. The semantics of the elements in an example instance of 15118 server is mapped to the corresponding instance of 61850 server and a 15118-to-61850.xsl script is constructed to demonstrate the transformation system, which can be used by the EV2G-XT.

Throughout the paper, the following abbreviations apply:
II. ISO/IEC 15118 V2G CI SCHEMA AND IEC 61850 SCL SCHEMA

A. The schema files

1) V2G CI schema files

V2G CI schema is defined in the files of ISO/IEC 15118-2 international standard [9] as shown in Fig. 2 [11]. They are also distributed by ESTI. There are differences in the content between XSD files of ESTI and ISO/IEC 15118-2 due to the different version. The latest version of the ISO/IEC 15118-2 schema is used in this paper.

The XML schema element import adds multiple schemas with different target namespace to a document, and V2G CI imports other schemas. The import hierarchy is shown in Fig. 2. The AppProtocol.xsd is for protocol handshake messages. It doesn’t import any other schema. MsgDef.xsd is for the message structure and imports MsgHeader.xsd for the message header and MsgBody.xsd for the message body. Both schema files import MsgDataTypes.xsd for the data types, which imports in turn xmlsig-core-schema.xsd. MsgBody.xsd defines the XML elements to be used for the individual charging sequence. These elements consist of request/response pairs (-Req/-Res) with following name stems: SessionSetup, ServiceDiscovery, ServiceDetail, PaymentServiceSelection, PaymentDetails, Authorization, ChargeParameterDiscovery, PowerDelivery, MeteringReceipt, SessionStop, CertificateUpdate, CertificateInstallation, ChargingStatus, CableCheck, PreCharge, CurrentDemand, WeldingDetection.

2) SCL schema files

SCL is the substation configuration description language for IED defined in the IEC 61850-6 [5]. SCL.xsd is the main schema definition file.

The XML schema element include adds multiple schemas with the same target namespace to a document, and SCL.xsd includes Substation.xsd for the substation grammar, IED.xsd for the IED grammar, Communication.xsd for the communication grammar, and DataTypeTemplates.xsd for the grammar of the data type template. These schemas include BaseTypes.xsd for the basic complex types used in the other schema files, which include Enums.xsd for the enumeration types. The include hierarchy is shown in Fig. 3. The root element SCL is defined in the SCL.xsd.

B. V2G CI message

V2G CI schema is divided into two parts: AppProtocol.xsd for the protocol handshake and others covering all the individual charging sequences.

The XML messages for the protocol handshaking and the other individual charging sequences are exchanged independently on every events during charging process as defined in ISO/IEC 15118 international standard. The V2G CI schema consists of the separate grammars for all the individual sequences.

1) V2G CI protocol handshake message in AppProtocol.xsd

The content is shown in Fig. 4. The elements are used to exchange various protocol information between EV and EVSE. Maximal 20 AppProtocol with Priority, ProtocolNamespace, SchemaID for each one must be defined.

2) The root element of the individual charging sequences: V2G_Message

The element V2G_message is defined in MsgDef.xsd (Fig. 5). It is a complexType element which consists of a Header element and a Body element. The MessageHeaderType is used to construct the message Header as defined in MsgHeader.xsd and the BodyType for the message Body defined in MsgBody.xsd.

3) The Header element of V2G CI

The MessageHeaderType is a complex type defined in the MsgHeader.xsd as shown in Fig. 6. The type sessionIDType of element SessionID and the type NotificationType of element Notification are defined in the MsgDataTypes.xsd, the reference element Signature is defined in the xmlsig-core-schema.xsd. The element SessionID appears only once, whereas other two elements may not appear at all. The sessionIDType is of type simpleType and defined as follows.

---

1 Variables and terms are expressed as slanted words throughout this paper.

2 The filename prefix “V2G_CI_” is omitted for all the V2G CI schema files, xmlsig-core-schema.xsd is an exception.

3 The filename prefix “SCL_” is omitted for all the SCL schema files, except SCL.xsd.

4 The figures are captured from Enterprise Architect.

5 The closing XML tags can be taken from the indentation.
The element restriction restricts the definition of simpleType, simpleContent, complexContent. It shows that the value of the sessionID is a hexadecimal number with maximal 8 characters.

4) The element Body of V2G CI

The element Body defined in MsgDef.xsd is of a complex type BodyType as shown in Fig. 7. The BodyType is a reference element BodyElement, which is abstract type. The attribute abstract is an optional attribute of XML description language, which determines whether the corresponding element can appear in the instance document. The default value is false. If it is true, the corresponding element itself (BodyElement) cannot appear in the instance document. Some other element with substitutionGroup attribute set to the value of the element (BodyElement) can appear instead.

All the substitution group elements in MsgBody.xsd and MsgDataTypes.xsd are listed in Table 1. The attribute type of the substitution group name is given in parenthesis.

The BodyElement has 34 substitution group elements, which are used for the individual charging sequences. Each element of the conceptual group is extended from the BodyElement.

The type of ChargeParameterDiscoveryReq is a complex type ChargeParameterDiscoveryReqType, which extends the element BodyBaseType to take the elements MaxEntriesSAScheduleTuple, RequestedEnergyTransferMode and the reference element EVChargeParameter as child nodes.

Because MaxEntriesSAScheduleTuple’s attribute minOccurs is 0, this element may not appear at all. The default value of minOccurs and maxOccurs is 1, and MaxEntriesSAScheduleTuple’s maxOccurs is the default value of 1. This means that this element appears 0 or 1 times in the instance document. It is shown as [0..1]
This document. This element is defined in MsgDataTypes.xsd and has the attribute abstract set to true.

Table 2 shows the semantics and type definition for the element ChargeParameterDiscoveryReq and Table 3 shows the 6 possible choices for the element RequestedEnergyTransferMode.

The element EVChargeParameterType of EVChargeParameter has a subelement DepartureTime and is abstract type and substituted by AC_EVChargeParameter or DC_EVChargeParameter. The element AC_EVChargeParameter has 4 child elements: EAmount, EVMaxVoltage, EVMaxCurrent, EVMinCurrent, which are of PhysicalValueType. Table 4 shows the semantics and type definition for the element AC_EVChargeParameter.

PhysicalValueType consists of elements Multiplier, Unit, and Value. Table 5 shows their semantics and type definition.

C. SCL schema and MMS message

There are special rules for the element names in the SCL schema. The element names with initial t are types of schema and the element names with initial ag are attribute groups, and the name of attribute starts with a lowercase letter and the name of element with an uppercase letter.

The content of SCL.xsd schema looks simple in the first glance, contains rather numerous definitions by including other schemas. The extension element of XML schema language extends the function of previously defined simpleType or complexType element. The complexType tBaseElement element is extended. The uniqueSubstation element is defined as a unique element, which doesn’t allow duplication. The IEDKey and LNodeTypeKey is defined to be used as keys. The ref2LNodeTypeDomain1, ref2LNodeTypeDomain2, ref2LNodeTypeLLN0, and refConnectedAP2IED are defined as keyref to be used as reference keys. The element SCL is shown in Fig. 10.

1) Basic element tBaseElement

Almost all the SCL elements are derived from the basic element tBaseElement, which is defined in BaseTypes.xsd. It allows the addition of Private part and Text part for the description of the element, subelement and attribute from other name spaces.

The next level of element type is complexType, which extends the tBaseElement. The tUnnaming, the tNaming and the tIDNaming belong to this category.

All three elements have the attributeGroup element, which
groups a series of attribute declarations together to cooperate as a group to the complex type definition. The attribute group *agDesc* has the name *desc*, is a type of string, in which *LF, CR, TAB* characters are filtered out. The default value of *agDesc* is an empty string and it’s an optional attribute. The type *tIDNaming* needs additional *name* attribute, the type *tIDNaming* needs additional *id* attribute.

```xml
<xs:attributeGroup name="agDesc">
  <xs:attribute name="desc" type="xs:normalizedString" use="optional" default=""/>
</xs:attributeGroup>
```

2) SCL instance document structure **Header**

Within the root element SCL, 5 sub-elements based on the
element `<tBaseElement>` can appear. The element `Header` appears only once and the other elements are optional. The element `Substation` and `IED` can appear unlimited times and the element `Communication` and `DataTypeTemplates` are allowed to appear only once.

The element `Header` is a type of `<tHeader>`, the element `Text` and `History` can be used once optionally. The accessible attributes are `id`, `version`, `revision`, `toolID`, `nameStructure` and the attribute `id` is mandatory. The type `<tText>` used for `Text` element is an extended type of `<tAnyContentFromOtherNamespace>`, which has the `abstract` attribute and can use any characters, elements or attributes defined in other namespaces with no limit. In this case, the schema used for the other namespaces should be provided but no error is produced if the schema cannot be found. The element `Hitem` of type `<tHitem>` within the element `History` can appear one or more times, which is extended from `<tAnyContentFromOtherNamespace>` and has the attributes of `version`, `revision`, `when`, `who`, `what`, `why`. The attributes `version`, `revision`, `when` are mandatory.

3) SCL instance document structure `Substation`

The `Substation` part describes the function structure of substations and used to identify the primary devices and the electrical connections. The logical node `<LNode>` can be connected to all the levels in the structure such as substation, voltage level, bay, device, functions related to the subsidiary devices, subsidiary functions. The power transformer `<PowerTransformer>` can be connected to substation, voltage level, and bay. The conducting equipment `<ConductingEquipment>` can only be connect to the bay level.

`tEquipmentContainer` is an extended form of `<tPowerSystemResource>`, which includes `<tLNodeContainer>`, and the `PowerTransformer` and `GeneralEquipment` are added to it.

<xs:complexType name="tEquipmentContainer" abstract="true">
  <xs:attribute name="pathName" type="tRef" use="required"/>
  <xs:attribute name="nameStructure" type="tName"/>
  <xs:attribute name="toolID" type="tName"/>
  <xs:attribute name="id" type="tName"/>
  <xs:attribute name="version" type="tName"/>
  <xs:attribute name="revision" type="tName"/>
  <xs:attribute name="History" type="tHistory"/>
  <xs:attribute name="GeneralEquipment" type="tGeneralEquipment" minOccurs="0" maxOccurs="unbounded"/>
</xs:complexType>

The power system equipment is divided into `PowerTransformer` and `ConductingEquipment`.

The schema `Enum.xsd` lists items to be used to configure the substation. The schema element `union` unifies the values of simple types. Enumerated lists are formed mostly by using `union` to unify a number of smaller lists. For example, `<tNLClassEnum>` is an enumerated list, which consists of `<tPredefinedLNClassEnum>` and `<tExtensionLNClassEnum>`. `<tPredefinedLNClassEnum>` is a union of `<tLPHDEnum>`, `<tLN0Enum>`, `<tDomainLNEnum>`, `<tExtensionLNClassEnum>` contains a list of 4 byte strings with initial uppercase letter. Besides the detailed list, `<PType>`, `<PTypePhysConn>`, `<AttributeName>`, `<ConductingEquipment>`, `<PowerTransformer>`, `<TransformerWinding>`, `<Equipment>`, `<ServiceSettings>`, `<Phase>`, `<Authentication>`, `<AssociationKind>`, `<tAnyContentFromOtherNamespace>` can appear unlimited times and the element `` is optional.

The `Bay` element is of type `<tBay>` and is an extension of `<tEquipmentContainer>` and includes the elements `<ConductingEquipment>`, `<ConnectivityNode>`, `<Function>`, `<VoltageLevel>`, `<PowerTransformer>`, `<TransformerWinding>`, `<Equipment>`, `<ServiceSettings>`, `<Phase>`, `<Authentication>`, `<AssociationKind>`, `<tAnyContentFromOtherNamespace>` can appear unlimited times and the element `` is optional.

The `ConnectivityNode` adds `pathName` to `<tNodeContainer>`, and the attribute `pathName` delimits the absolute path of connection node within SCL file using the delimiter "/".

The diagram shows the relationship between the classes and attributes in the SCL instance document structure.
LNClass, CDC, FC, BasicType, ValKind, GSEControlType, SIUnit, UnitMultiplier, Right, DSO, D4C, SmpMod, PhysConnType, and ServiceType are defined.

4) SCL instance document structure IED

As shown in the Fig. 13, the IED section is divided into Services for the communication services offered and AccessPoint for the process bus access points. The access point is described by one of the elements: Server, ServerAt or LN list. It may optionally contain the security-related elements GOOSESecurity and SMVSecurity. An access point may belong to a server with logical devices, which contain logical nodes. An access point always needs a server, if the IED is to be supervised remotely, because the LN0 and LPHD of the server’s logical device are used to supervise and control the IED. Only if all LNs on an IED use an access point as a client only, and the IED is not supervised, may an IED without a server be used.

The optional DOI elements in an LN definition can be used to define special instance-related values for data objects and their attributes by using SDI elements for data object or attribute structure parts, if needed, and DAI elements per final attribute. The DOI attribute within the DOI defines the attributes and the related values to be set. The value of the data attribute is stored as simpleContent of the 0 or more child element Val of DAI. The name of the data object should be stored as the value of attribute name of DAI. The type of the element Val is xs:normalizedString.

III. XSL TRANSFORMATION

A. V2G CI message: ChargeParameterDiscoveryReq

The element ChargeParameterDiscoveryReq is a member of substitutionGroup of BodyElement, which the Body element of the root element V2G_Message should have exactly one time as subelement. It is explained in 4) on page 73 already. It consists of the 3 subelements MaxEntriesSAScheduleTuple, RequestedEnergyTransferMode, and EVChargeParameter. The element EVChargeParameter with one subelement DepartureTime is abstract type, is replaced by either AC_EVChargeParameter or DC_EVChargeParameter. For AC charging system, AC_EVChargeParameter replaces EVChargeParameterType. AC_EVChargeParameter consists of 4 subelements: EAmount, EMaxVoltage, EMaxCurrent, EMinCurrent. They are of type PhysicalValueType. PhysicalValueType has 3 subelements: Multiplier, Unit, Value.

The structure of this V2G message is simplified as follows. The grey colored elements have attribute minOccurs set to 0.

B. Logical nodes of SCL for EVSE

The logical nodes for EVSE defined in IEC 61850 are listed in [7].

The configuration of the EVSE including modeled data and
communication services is represented by these logical nodes. The data received by 15118 server from EV are to be stored in these logical nodes as appropriate data objects by 61850 server, in order to eventually transfer them to power grid. The data stored in these logical nodes may also be sent to the EV as the communication messages for the charging sequences. Because of the discrepancy between the structures of the V2G messages and SCL files, both XML files should be transformed in the structures as described by the corresponding schema.

This transformation should obey the rules of the schema. Some elements can have for example certain number of attributes, certain type of subelements, and certain type of data values, etc. Some elements or attributes are mandatory and others optional.

As shown in the Table 7, five data objects of the logical node DEEV are related to the message data for the AC_EVChargeParameter. The relevant data attributes for the common data classes TSG and ASG are listed in Table 8.

From the section “C. SCL schema and MMS message”, the entire structure for the logical nodes can be constructed as follows, with required attribute given after the symbol @. The elements, among which one element can be chosen (xs:choice), are separated with the symbol |.

### Table 6. The logical nodes for EVSE in IEC 61850

| LN   | Description                  | LN   | Description                          |
|------|------------------------------|------|--------------------------------------|
| CSWI | Charging spot switch         | DEEV | Information of EV connected to EVSE  |
| DEOL | Information to monitor/control the feature of EVSE outlet | DESE | Information to monitor/control the feature of EVSE |
| DSCH | Schedule control for energy and/or auxiliary service | DSCH | Schedule for energy and/or auxiliary service |
| LCCH | Communication channel between EV and EVSE | LLN0 | Logical node 0                        |
| LPD  | Physical device information  | MMTR | Energy measurement                   |
| MMXU | Power measurement            | ZCEV | EV in plug-in state                   |
| ZCHS | charging spot                |      |                                      |

### Table 7. Data objects of LNGroupD::DEEV [6]

| Data object name | Common data class | T   | Explanation                          | M-O-C nds/ds |
|------------------|-------------------|-----|--------------------------------------|--------------|
| DptTm            | TSG               |     | Departure time is used to indicate when the vehicle intends to finish the charging process. A value of zero (0) indicates that the charging process shall be finished as fast as possible. | O / F        |
| ExAmnt           | ASG               |     | Amount of energy required by the EV until the departure time has been reached or the EV battery's SOC is at 100%. This might include the amount of energy the EV consumes for other vehicle features than solely charging the EV battery. | O / F        |
| VMax             | ASG               |     | Maximum voltage supported by the EV. This is the voltage measured between one phase and neutral. | O / F        |
| AMax             | ASG               |     | Maximum current supported by the EV per phase. | O / F        |
| AMin             | ASG               |     | Minimum current supported by the EV per phase. | O / F        |

### Table 8. Some data attributes of common data class TSG/ASG

| Common data class name | Data attribute name | Type     | Semantics                      |
|------------------------|---------------------|----------|--------------------------------|
| TSG                    | setTm               | TimeStamp| The value of a time setting.   |
|                        | setMag              | AnalogueValue | The value of an analogue setting or set point. |
|                        | units               | Unit     | Units of the attribute(s) representing the value of the data. |
| ASG                    | sVC                 | ScaledValueConfig | Scaled value configuration. Shall be used to configure the scaled value representation. |
|                        | minVal              | AnalogueValue | Defines together with maxVal the setting range for ctlVal (CDC INC, BSC, ISC), setVal (CDC ING) or setMag (CDC APC, ASG). |
|                        | maxVal              | AnalogueValue | Defines together with minVal the setting range for ctlVal (CDC INC, BSC, ISC), setVal (CDC ING) or setMag (CDC APC, ASG). |

C. XSL Transformation from V2G CI message to SCL file

For example, the 15118 XML instance for the ChargingParameterDiscoveryReq for AC charging in [9] is illustrated in Fig. 14.

The message data received by 15118 server are DepartureTime = 100, EAmount = 18·10³ Wh, EVMaxCurrent = 230·10⁰ A, and EVMinCurrent = 0·10⁰ A.

To transform 15118 XML instance to 61850 XML instance, the following XSL script is constructed by associating V2G CI schema with SCL schema. The V2G CI message is to be transformed by using the script in Fig. 15.

The XSL script is written in XML format, which conforms XSL schema. The root element is xsl:stylesheet. The element xsl:template has an attribute match, whose value is XPath string. If a node in XML instance is matched with the value of the attribute match, then the rules of the xsl:template are applied. The element xsl:value-of extracts the value of a selected XPath node given by the attribute select. XSL consists of XSLT, XPath, XQuery and would not be described in details in this paper.

The restructured 61850 SCL file using the XSL processor of Stylus Studio® is in Fig. 16.
The result is not a valid SCL file because of lacking the subelements of SCL, e.g., Header, Substation, and Communication, etc. They are to be taken from the EVSE’s own SCL file. The XSL transformation is focused on the V2G message exchanges, for example AC_EVChargeParameterReq in this case.

IV. CONCLUSION

The EV charging sequence is defined in the ISO/IEC 15118 and the communication messages for the charging sequence are exchanged in the form of XML files with EVSE. The communication messages between EVSE and power grid are described in SCL of ISO 61850. The SCL file in form of XML is based on XML schema description. The ISO/IEC 15118 XML file and ISO 61850 SCL file can be transformed to each other by using XSL Transformation engine, EV2G-XT, as depicted in Fig. 1.

XSL is used to build transformation modules between V2G CI schema and SCL schema. 15118-to-61850.xsl could test the existence of XPath information e.g. “//V2G_Message/PaymentDetails” by using XSL function <xsl:if test="expression" />, which could identify the current charging sequence. EV2G-XT could store e.g. the negotiated communication protocol ID at the stage of supportedAppProtocolReq in some place, which is used in other charging sequences.

ACKNOWLEDGEMENT

This work was supported financially by the KEPCO in “Fundamental Study 2014 for Electric Power Industry” by grants from KESRI as a project “XML Schema Translation Engine Development for Combo-type EVSE Communication Controller Between EV and Power System” (project management No.: R14XA02-51).

REFERENCES

[1] Han, A., et al, “A Study on Communication Controller of Electric Vehicle Supply Equipment for Information Exchange between Electric Vehicle and Power Grid”, The Transactions of KIEE Vol. 63, No. 11, pp. 1564-1570, 2014.
[2] Lee, D.Y., et al, “A Study about Data Mapping of Charging Device Server for Electric Vehicle”, KIEE Summer Conference 2013, 7. 10-12.
[3] Un, K., et al, “A Study for a Transformation of Communication Message within the EVSE Using XML Schema Transformation”, KIEE Summer Conference 2015, 7. 15-17.
[4] “Draft IEC/TR 61850-1 Ed.2.0: Communication networks and systems for power utility automation - Part 1: Introduction and...
overview", Ed.2.0, IEC Draft TR 61850-1, 2010. 10. 1.
[5] “IEC 61850-6 Ed.2: Communication networks and systems for power utility automation - Part 6: Configuration description language for communication in electrical substations related to IEDs”, FDIS, IEC 61850-6 Ed.2, 2009. 9. 4.
[6] 850-7-420 Ed. 2.0: Communication networks and systems for power utility automation - Part 7-420: Basic communication structure - Distributed energy resources logical nodes", Ed. 2.0, IEC Draft.
[7] “IEC TR 61850-90-8: Communications Systems for Distributed Energy Resources – Part 90-8: Object model for electric mobility”, Edition 1.4.24, IEC TR 61850-90-8, 2014. 10. 4.
[8] “ISO 15118-1: Road vehicles - Vehicle to grid communication interface - Part 1: General information and use-case definition”, Ed.1.0, ISO 15118-1:2013(E), 2013. 4. 15.
[9] “ISO 15118-2: Road vehicles - Vehicle-to-Grid Communication Interface - Part 2: Network and application protocol requirement”, Ed.1.0, ISO 15118-2:2014(E), 2014. 4. 01.
[10] “ISO/FDIS 15118-3: Road Vehicles - Vehicle to grid communication interface - Part 3: Physical layer and Data Link layer requirements”, Ed1, ISO/FDIS 15118:2014(E), 2014. 12. 5.
[11] http://forge.etsi.org/websvn/listing.php?repname=LIBS.Libs&pat h=%2Ftrunk%2Fxsd%2FV2G%2FV2Gxsd&rev=828&peg=513#a79befbd 578f3285be41100703a6d67
[12] http://www.w3schools.com/schema/default.asp.

<?xml version='1.0'?><xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"><xsl:output indent="yes"/>
<xsl:template match="/">
  <SCL xmlns="http://www.iec.ch/61850/2003/SCL" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.iec.ch/61850/2003/SCL SCL.xsd" version="2007" revision="A">
    <IED name="IEDName">
      <AccessPoint name="AccessPointName">
        <LN lnType="tLNName" lnClass="tLNClassEnum" inst="tLNInst">
          <LDevice>
            <xsl:for-each select="//*[local-name()='AC_EVChargeParameter']/*">
              <xsl:choose>
                <xsl:when test="local-name()='DepartureTime'">
                  <DOI name="DptTm">
                    <DAI name="setTm"><Val><xsl:value-of select="./text()"/></Val></DAI>
                </DOI>
              </xsl:when>
              <xsl:when test="local-name()='EAmount'">
                <DOI name="EnAmnt">
                  <DAI name="setMag"><Val><xsl:value-of select="./text()"/></Val></DAI>
                  <DAI name="units"><Val><xsl:value-of select="./text()"/></Val></DAI>
                  <DAI name="sVC"><Val><xsl:value-of select="./text()"/></Val></DAI>
                </DOI>
              </xsl:when>
              <xsl:when test="local-name()='EVMaxVoltage'">
                <DOI name="VMax">
                  <DAI name="setMag"><Val><xsl:value-of select="./text()"/></Val></DAI>
                  <DAI name="units"><Val><xsl:value-of select="./text()"/></Val></DAI>
                  <DAI name="sVC"><Val><xsl:value-of select="./text()"/></Val></DAI>
                </DOI>
              </xsl:when>
              <xsl:when test="local-name()='EVMaxCurrent'">
                <DOI name="AMax">
                  <DAI name="setMag"><Val><xsl:value-of select="./text()"/></Val></DAI>
                  <DAI name="units"><Val><xsl:value-of select="./text()"/></Val></DAI>
                  <DAI name="sVC"><Val><xsl:value-of select="./text()"/></Val></DAI>
                </DOI>
              </xsl:when>
              <xsl:when test="local-name()='EVMinCurrent'">
                <DOI name="AMin">
                  <DAI name="setMag"><Val><xsl:value-of select="./text()"/></Val></DAI>
                  <DAI name="units"><Val><xsl:value-of select="./text()"/></Val></DAI>
                  <DAI name="sVC"><Val><xsl:value-of select="./text()"/></Val></DAI>
                </DOI>
              </xsl:when>
              <xsl:otherwise>
                <xsl:comment>Unknown element <xsl:value-of select="name()"/></xsl:comment>
              </xsl:otherwise>
            </xsl:choose>
          </LDevice>
        </LN>
      </AccessPoint>
    </IED>
  </SCL>
</xsl:template></xsl:stylesheet>

Fig. 15. The script transforming V2G CI message
Fig. 16. The restructured 61850 SCL file using the XSL processor of Stylus Studio