AUTOMATED FRAMEWORK FOR GENERATING CYBER-PHYSICAL RANGE FOR SMART GRID

SG-ML: SMART GRID CYBER RANGE MODELLING FRAMEWORK

Muhammad M. Roomi, Suhail S.M. Hussain, Daisuke Mashima,
Ee-Chien Chang, David M. Nicol
Illinois Advanced Research Center, National University of Singapore
Singapore

July 1, 2025

Contents

1 Overview

In this document, we describe the specification of SG-ML, the smart grid cyber range modelling framework, for automated cyber range generation. SG-ML is defined as a set of XML schemas for describing configurations of a smart grid cyber range in not only a machine-readable but also a human-friendly manner. SG-ML is to be written or customized by users according to their needs on cyber range (e.g., what cyber and/or physical topology needs to be emulated, what devices are involved, etc.). The configured SG-ML model will be parsed according to the schema and utilized to instantiate the cyber range by a toolchain (called SG-ML Processor) we will develop.

SG-ML includes XML schema for defining the following information that are required to instantiate a smart grid cyber range:

(1) Power system topology and configuration (e.g., single-line diagram, configuration and status of power system devices, such as circuit breakers, load profile)

(2) Cyber network topology and configuration (e.g., connectivity among devices, network bandwidth, etc.)

(3) Device configurations (e.g., network addresses, communication models, and/or functionalities of SCADA HMI, PLCs, and IEDs)

SG-ML takes advantage of established, standardized smart grid modelling framework, namely IEC 61850 Substation Configuration Language [1], as well as IEC 61131 PLCopen XML, which defines the schema for describing PLC logic in XML format. For instance, IEC 61850 SCL includes XML schema that defines a single-line diagram of a substation system, which can be used for aforementioned point (1). SCL schema also defines cyber network topology, communication model, and device configurations, which can be used for (2) and (3). Because these are the widely-used models and are often readily available to power grid system operators, employing these as part of SG-ML will reduce the amount of efforts for creating SG-ML model, by recycling such existing assets. (For example, if a power grid operator wants to model a cyber range corresponding to their substation model, relevant sections of the SCL files they already have can be used as part of SG-ML model.) Having that said, through our study we identified the gap between what are available in these standardized models and what we need for defining cyber range. To fill the gap we also defined our proprietary schemas for providing supplementary information.

In sum, the SG-ML consists of components and processes shown on the bottom part of Figure 1. In the rest of this document, we elaborate on how these components are utilized for automated generation of cyber range, alongwith description of some utilized standard models. The representation of the smart grid modelling framework is depicted in Fig. 2.

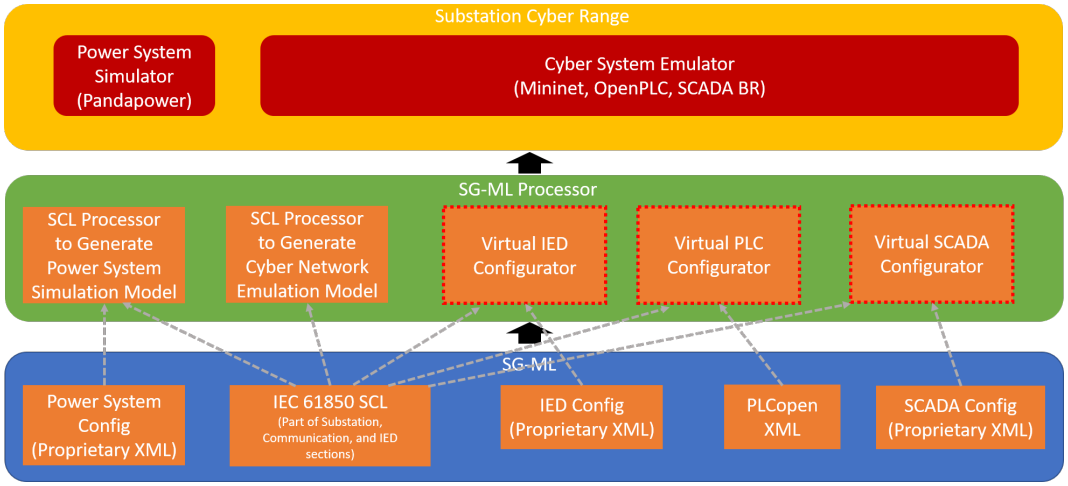


Figure 1: SG-ML Overview

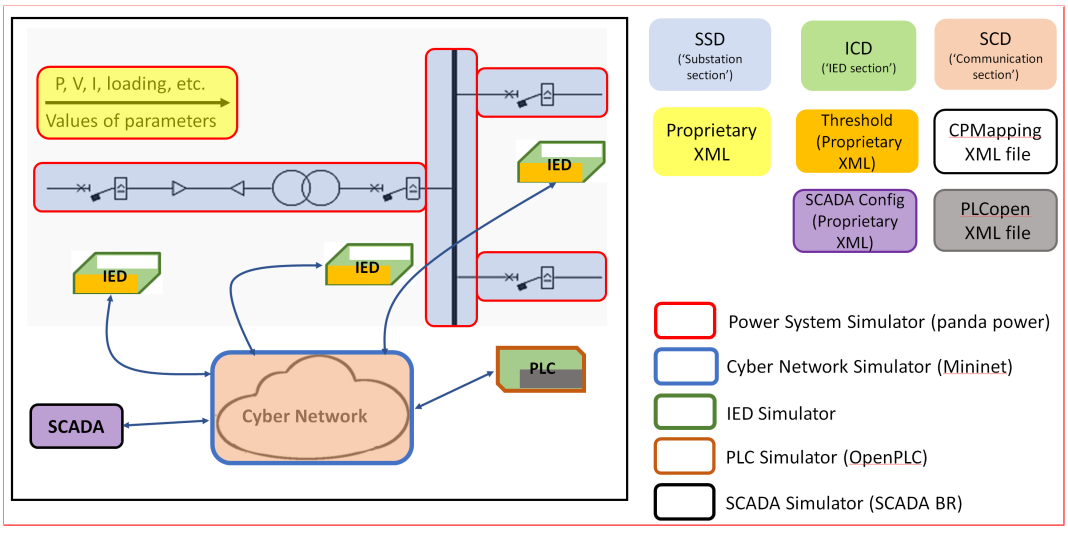
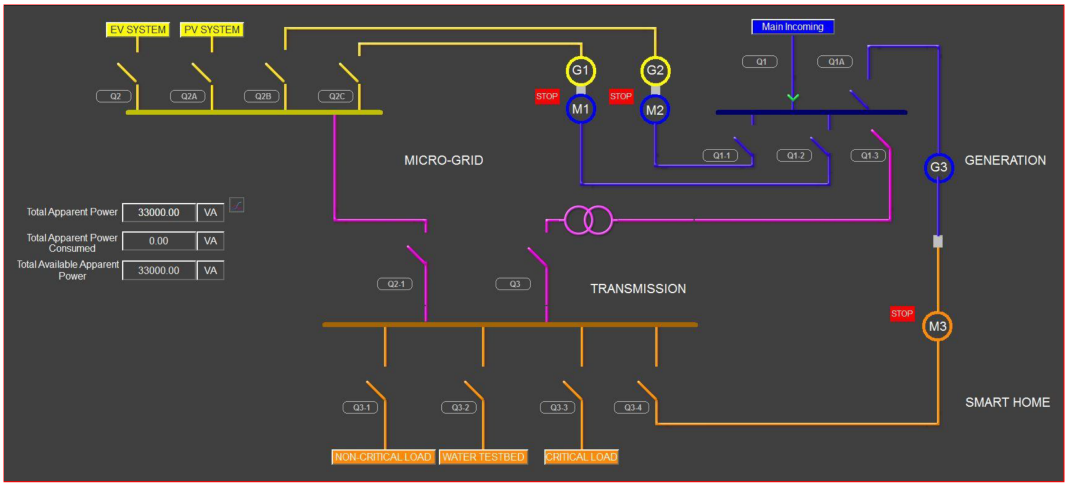


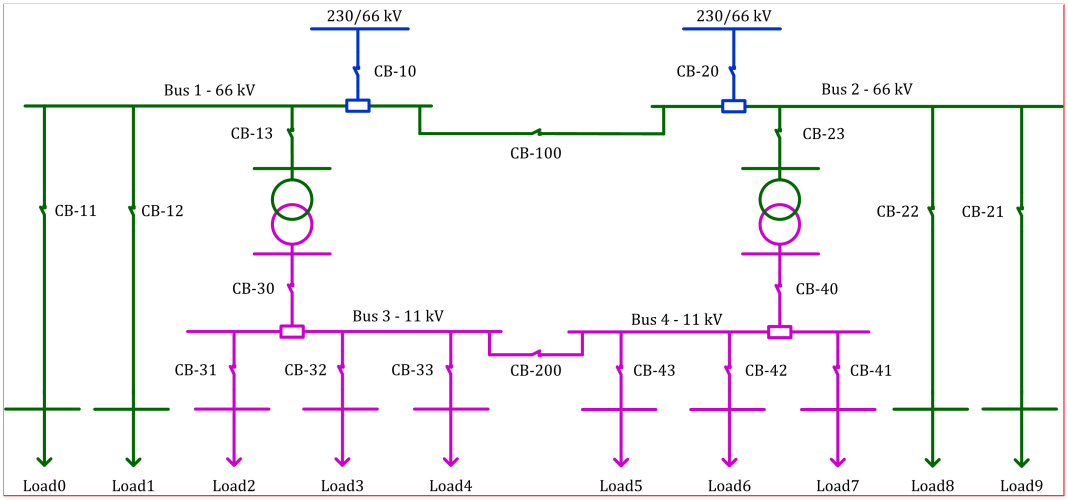
Figure 2: SG-ML Framework

2 Systems under study

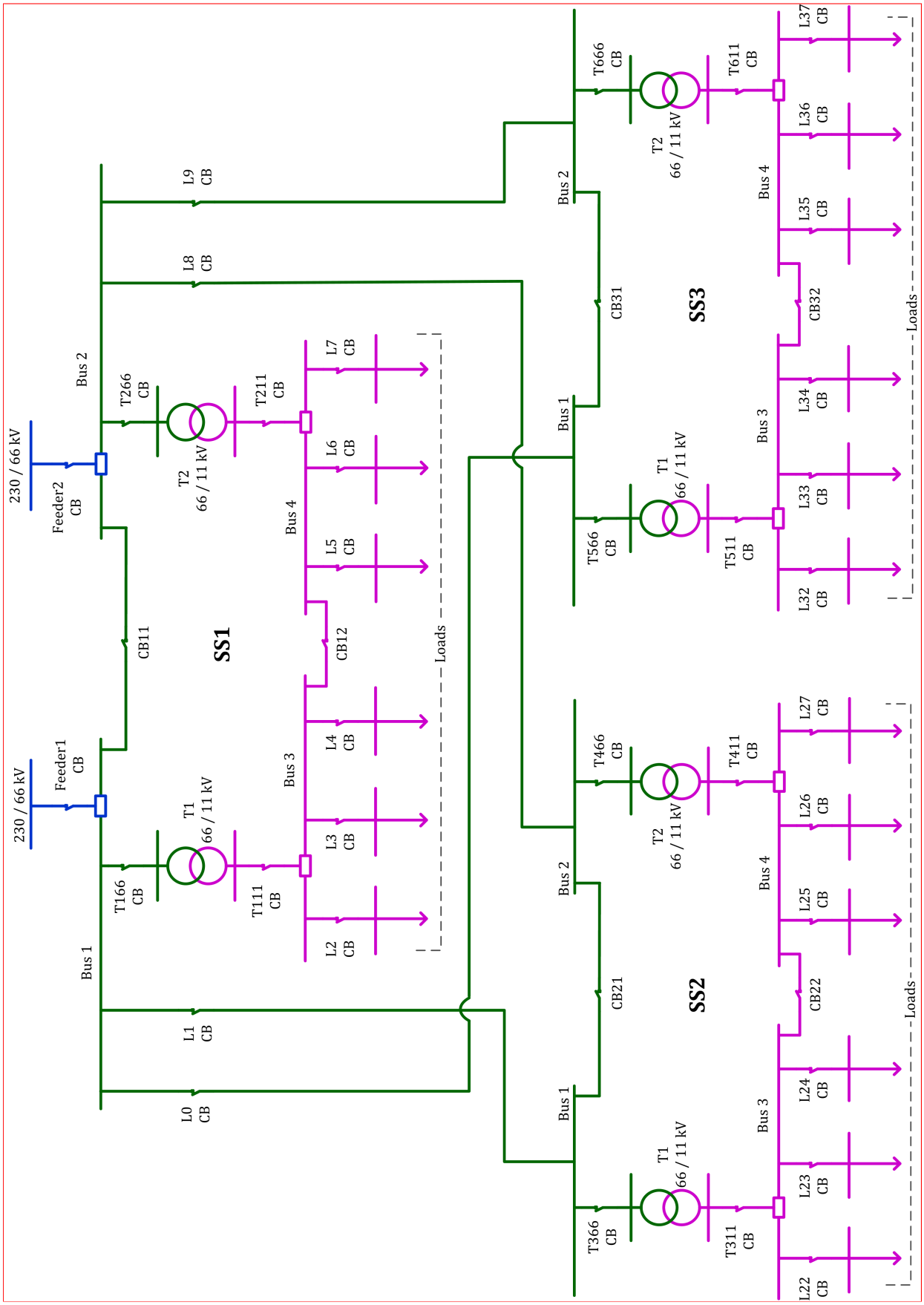
2.1 Electric Power and Intelligent Control Testbed:



2.2 A typical sub-transmission level substation model:



2.3 A typical Three-substation system model:


230 / 66 kV
Feeder1
CB
Feeder2
CB
Bus 1
Bus 2
Bus 3
Bus 4
T166
CB
T266
CB
T111
CB
T211
CB
T1
66 / 11 kV
T2
66 / 11 kV
L0
CB
L1
CB
L2
CB
L3
CB
L4
CB
L5
CB
L6
CB
L7
CB
L8
CB
L9
CB
CB11
CB12
SS1
Loads
T366
CB
T466
CB
T311
CB
T411
CB
CB21
CB22
SS2
T566
CB
T666
CB
T511
CB
T611
CB
CB31
CB32
SS3
L22
CB
L23
CB
L24
CB
L25
CB
L26
CB
L27
CB
L32
CB
L33
CB
L34
CB
L35
CB
L36
CB
L37
CB

3 Substation Configuration Files

Aim	Substation Configuration Language Files
Software	SCL Matrix [2]
Outcome	System Specification Description (SSD), IED Capability Description (ICD), System (Substation) Configuration Description (SCD)

SCL is the descriptive language defined by IEC 61850 for configuring electrical substations devices. The main parts of an SCL include Header, Substation, Communication, IED, Datatype templates.

3.1 System Specification Description (SSD) file:

This file includes the complete specification of the substation automation system (SAS). The single line diagram (SLD) of the SAS and its functionalities (allocation of logical nodes (LN) to the physical system components) are defined. The main component of the file is the 'Substation' section. A sample of the Substation section in an SSD file is shown in Fig. 3.

```
<Substation name="EPIC">
 <VoltageLevel name="400 V" numPhases="3" nomFreq="50">
  <PowerTransformer name="ATR" type="PTR" sxy:x="45" sxy:y="18" sxy:dir="horizontal">
   <TransformerWinding name="W1" desc="W1" type="PTW">
    <Terminal connectivityNode="EPIC/400 V/Mains/CN" substationName="EPIC" voltageLevelName="400 V" bayName="Mains" cNodeName="CN" />
    <Terminal connectivityNode="EPIC/400 V/T1/ATRCBRCN" substationName="EPIC" voltageLevelName="400 V" bayName="T1" cNodeName="ATRCBRCN" />
   </TransformerWinding>
  </PowerTransformer>
  <Voltage unit="V" multiplier="k">0.4</Voltage>
  <Bay name="G3" sxy:x="16" sxy:y="4">
   <ConductingEquipment name="CON" type="CON" sxy:x="1" sxy:y="4" sxy:dir="vertical">
    <Terminal connectivityNode="EPIC/400 V/G3/CONDISCN" substationName="EPIC" voltageLevelName="400 V" bayName="G3" cNodeName="CONDISCN" />
   </ConductingEquipment>
   <ConductingEquipment name="Q0" type="CBR" sxy:x="0" sxy:y="7">
    <Terminal connectivityNode="EPIC/400 V/G3/CONDISCN" substationName="EPIC" voltageLevelName="400 V" bayName="G3" cNodeName="CONDISCN" />
    <Terminal connectivityNode="EPIC/400 V/Gen/CN" substationName="EPIC" voltageLevelName="400 V" bayName="Gen" cNodeName="CN" />
   </ConductingEquipment>
   <ConnectivityNode name="CONDISCN" pathName="EPIC/400 V/G3/CONDISCN" sxy:x="1" sxy:y="5" />
  </Bay>
 </VoltageLevel>
</Substation>
```

Figure 3: Description of SLD in SCL format.

The 'Substation' section in the SSD file includes the sub-section 'Voltage Level', which in turn consists of 'Bay'. A single substation may contain one or more voltage levels and a single voltage level may contain one or more bays. Voltage level includes details such as operating voltage, number of phases, operating frequency, alternating transformer, etc. A Bay represents a single feeder line in the substation. Therefore, it includes the details of the physical components (represented as 'ConductingEquipment' in SCL) such as circuit

breaker, relay, disconnectors, etc., that are present in the feeder line. A node is the point of connection between two physical components. The 'Terminal connectivityNode' in the above figure represents the location of a particular in the substation, with which the location of any physical components can be determined. The above SSD figure demonstrates the simple SCL structure of the EPIC testbed model without the allocation of Logical Nodes (LN) to the power system components. The LN can be assigned to the physical equipment after the integration of Intelligent Electronic Device (IED) description file to the substation.

3.2 IED Capability Description (ICD) file:

IEDs are widely deployed in power automation system due to its interoperability feature and its flexibility to integrate. The primary function of an IED is to provide protection function. Some common types of IED include protective relay devices, circuit breaker controllers, voltage regulators, re-closer controllers, capacitor bank switches, tap change controllers, etc. IEDs receive measurement from sensor and other power equipment. According to the measurements, IEDs can either provide control commands to trip circuit breakers if any anomalies in voltage, current or frequency is detected or adjust the tap controllers to maintain the voltage at the desired level. The file which describes the functionalities of an IED is called the IED Capability Description (ICD) file, usually provided by the manufacturer based on the requirements.

The main section of the ICD file is the 'IED' section. The template for an IED is shown in Fig. 4. This section contains the services included for a particular IED. As depicted in the figure, IED section provides the details of the Logical Devices (LDs) and the Logical Nodes (LNs). Every LD may contain group of LNs.

LNs in IEC 61850 is used to define the basic functions in an IED and it contains a group of Data Objects (DOs). IEC 61850 part 7-4 defines logical node type or class. All the logical node classes are grouped based on their functions. Among the various LN classes, the protection class and the measurement class are defined in the virtual IEDs that are utilised in this project. The protection functions include Over-current protection (PTOC), Over-voltage/Under-voltage protection (PTOV/PTUV), Under-frequency protection (PTUF), Reverse power flow protection (PDOP), Thermal over-load (PTTR). Similarly, the measurements of three-phase voltages, three-phase currents, active power and frequency contribute towards the measurement class, which is represented as 'MMXU'. Additionally, a single IED may contain multiple instances any LN type/class and this is represented by an instance number added as a suffix (e.g. MMXU1, MMXU2).

3.2.1 Control Blocks

The communication between IEDs and Supervisory Control And Data Acquisition (SCADA) mostly follows Client-Server communication service. Manufacturing Message Specification (MMS) protocol is used in IEC 61850 to implement the communication between IEDs and

```
<IED name="TEMPLATE" manufacturer="Grid Software" type="Control Device" configVersion="1.0" desc="IED Specification">
 <Services>
  <DynAssociation />
  <GetDirectory />
  <GetDataObjectDefinition />
  <DataObjectDirectory />
  <GetDataSetValue />
  <DataSetDirectory />
  <ConfDataSet max="10" maxAttributes="100" modify="true" />
  <ReadWrite />
  <ConfReportControl max="10" />
  <GetCBValues />
  <ReportSettings cbName="Conf" datSet="Conf" rptID="Conf" optFields="Conf" bufTime="Conf" trgOps="Conf" intgPd="Conf" />
  <GSESettings cbName="Conf" datSet="Conf" appID="Conf" dataLabel="Conf" />
  <GOOSE max="10" />
 </Services>
 <AccessPoint name="AP1">
  <Server timeout="30">
   <Authentication />
   <LDevice inst="LD0">
    <LN0 lnClass="LLN0" inst="" lnType="LLN0_1c43b518-0447-4dd1-aea2-db2a15942ede">
    <LN lnClass="LPHD" inst="1" lnType="LPHD_75f376bf-0cb6-4b0f-a86b-136a26b269b4" />
    <LN lnClass="PTRC" inst="1" lnType="PTRC_497bc644-f90b-4af6-96dd-506e769d5fef" />
    <LN lnClass="MMXU" inst="1" lnType="MMXU_dadf927c-e647-4e8d-9d03-23b17f9de009" />
   </LDevice>
   <LDevice inst="Control">
    <LN0 lnClass="LLN0" inst="" lnType="LLN0_a284d13b-a0e9-4e37-b1eb-7244e3d3d217" />
    <LN prefix="CON" lnClass="XCBR" inst="1" lnType="XCBR_73ff6c9a-a07b-4a50-8475-42e7e90dec9d" />
    <LN prefix="CON" lnClass="CSWI" inst="1" lnType="CSWI_e8211917-1dff-49e0-94a1-a2800f7b0c11" />
   </LDevice>
  </Server>
 </AccessPoint>
</IED>
```

Figure 4: Description of ICD (IED section) in SCL format.

SCADA. IED acts as the server, waiting for request from SCADA. The Client SCADA initiates the communication and requests to read data or control command, upon which IED responds to the corresponding request.

In order to log the report event operation, IEDs need to be configured to prepare datasets containing the necessary data points and the report control blocks to specify how the data has to be communicated. Fig. 5 demonstrates the details of the report control block that includes the data related to the protection functionalities (thermal over-load and protection trip conditioning). Additionally, data objects related to measurements can be included in the control block.

Generally, the communication to send event reports between IEDs and SCADA are established when there is a change in the data or quality attribute. However, periodical communication even when there is no change in the attributes can also be configured.

3.2.2 GOOSE Control Block

Generic Object Oriented Substation Events (GOOSE) communication is used in IEC 61850 to establish communications between IEDs. This communication follows the Publish - Subscribe service. The GOOSE-sending IED publishes the data to all IEDs in the network.

```
<LDevice inst="PROT">
 <LN0 lnClass="LLN0" inst="" lnType="PROT_LLN0">
  <DataSet name="rdsA" desc="(created on Tuesday, 29 September 2020 15:16:17)">
   <FCDA ldInst="PROT" lnInst="1" lnClass="PTTR" doName="Mod" daName="stVal" fc="ST" />
   <FCDA ldInst="PROT" lnInst="1" lnClass="PTTR" doName="Mod" daName="q" fc="ST" />
   <FCDA ldInst="PROT" lnInst="1" lnClass="PTTR" doName="Mod" daName="t" fc="ST" />
   <FCDA ldInst="PROT" lnInst="1" lnClass="PTRC" doName="Mod" daName="stVal" fc="ST" />
   <FCDA ldInst="PROT" lnInst="1" lnClass="PTRC" doName="Mod" daName="q" fc="ST" />
   <FCDA ldInst="PROT" lnInst="1" lnClass="PTRC" doName="Mod" daName="t" fc="ST" />
  </DataSet>
  <DataSet name="rdsB" desc="(created on Tuesday, 29 September 2020 15:18:25)">
   <FCDA ldInst="PROT" lnInst="1" lnClass="PTTR" doName="Mod" daName="ctlModel" fc="CF" />
   <FCDA ldInst="PROT" lnInst="1" lnClass="PTRC" doName="Mod" daName="ctlModel" fc="CF" />
  </DataSet>
  <ReportControl name="rcbA" bufTime="500" buffered="true" confRev="1" intgPd="1000" indexed="true" datSet="rdsA" desc="(created on Tuesday, 29 September 2020 15:19:13)">
   <TrgOps gi="true" dchg="true" qchg="true" dupd="true" period="true" />
   <OptFields bufOvfl="true" seqNum="true" timeStamp="true" dataSet="true" reasonCode="true" dataRef="true" entryID="true" configRef="true" />
   <RptEnabled max="1" />
  </ReportControl>
 </LN0>
 <LN lnClass="PTTR" inst="1" lnType="PROT_PTTR" />
 <LN lnClass="PTRC" inst="1" lnType="PROT_PTRC" />
</LDevice>
```

Figure 5: Description of ICD (Report Control Blocks) in SCL format.

However, the GOOSE data being published is retrieved by the IED that has subscribed and utilises the data.

This communication involves at least one GOOSE-sending IED and one or more GOOSE-receiving IEDs. Similar to the report control blocks, data points that are to be published are created as a dataset. Subsequently, a GOOSE control block is configured that specifies how the GOOSE message has to be communicated. Fig. 6 illustrates the GOOSE control block with the GOOSE dataset. This dataset includes the data points related to the circuit breaker position.

```
<LDevice inst="LD0">
 <LN0 lnClass="LLN0" inst="" lnType="LLN0_12189f99-9dbf-4aff-ab43-5ab5f1bfaea8">
  <DataSet name="gooseDataset" desc="(created on Wednesday, 19 August 2020 14:33:47)">
   <FCDA ldInst="Control" lnInst="1" lnClass="XCBR" doName="Pos" daName="stVal" fc="ST" />
   <FCDA ldInst="Control" lnInst="1" lnClass="XCBR" doName="Pos" daName="q" fc="ST" />
  </DataSet>
  <GSEControl name="GooseCB" type="GOOSE" fixedOffs="false" confRev="60001" securityEnable="None" appID="NULL" datSet="gooseDataset" desc="(created on Wednesday, 19 August 2020
   14:39:26)" />
 </LN0>
 <LN lnClass="LPHD" inst="1" lnType="LPHD_1974448a-341e-427e-b31b-906900de30ff" />
</LDevice>
```

Figure 6: Description of ICD (GOOSE Control Blocks) in SCL format.

The configuration of GOOSE communication can be described in three steps:

- Setting up GOOSE-sending IED (data points grouped in to datasets and configuring GOOSE control block)
- Setting up the GOOSE-receiving IEDs
- Configuring the GOOSE-receiving IED by mapping the GOOSE data to the user logic of the IED

3.2.3 Communication Section

One another important section in the ICD file is the 'Communication Section'. This section contains the details of the communication services supported by the IED. Fig. 7 represents the communication section in the ICD file. It incldues details such as the SubNetwork name, communication protocol, IP address details, Physical Connection features.

```
<Communication>
  <SubNetwork name="SubNetwork1" type="8-MMS">
    <ConnectedAP iedName="TEMPLATE" apName="AP1">
      <Address>
        <P type="IP" xsi:type="tP_IP">0.0.0.0</P>
        <P type="IP-SUBNET" xsi:type="tP_IP-SUBNET">255.255.255.0</P>
        <P type="IP-GATEWAY" xsi:type="tP_IP-GATEWAY">0.0.0.0</P>
      </Address>
      <PhysConn type="Connection">
        <P type="Type">FOC</P>
        <P type="Plug">LC</P>
      </PhysConn>
    </ConnectedAP>
  </SubNetwork>
</Communication>
```

Figure 7: ICD (Communication Section) in SCL format.

3.3 System (Substation) Configuration Description (SCD) file:

The SSD and ICD files are processed by the System Configurator Tool (SCT), the resultant of which is the System (Substation) Configuration Description (SCD) file. This file includes the physical and cyber aspects of the entire substation. Once the processing is done, the resultant SCD files will have additional details, when compared with the individual SSD and ICD files. The changes in the configuration are listed in the following subsections.

3.3.1 Configured SSD in SCD file

Fig. 8 exemplifies the details included in the 'Substation' section of the SSD file. In the SCD file, the Logical Nodes in the ICD files are allocated to the physical components in the SSD file. Therefore, attributes associated with measurements, protection, switch gear, etc. are defined in the SCD files. For e.g. consider the power transformer (PowerTransformer

name=“TR1”) in the figure. The choice of which specific LN is allocated to the power transformer from a particular IED is one among the details realised. Subsequently, the physical location of the physical component in the substation is also described.

```
<Substation name="EE1">
 <PowerTransformer name="TR1" type="PTR" sxy:x="21" sxy:y="39">
  <LNode iedName="T_110_11" ldInst="Prot" lnClass="PDIS" lnInst="1" lnType="PDIS_0c128dd8-9416-46c5-a150-3e5583640ed8" />
  <LNode iedName="T_110_11" ldInst="Prot" lnClass="PDIF" lnInst="1" lnType="PDIF_67e4c3ef-ee9e-4cc9-839b-8019066956e9" />
  <LNode iedName="T_110_11" ldInst="Prot" lnClass="PTOC" lnInst="1" lnType="PTOC_b024441a-d7f5-4518-aa5f-0ae5428b851e" />
  <LNode iedName="T_110_11" ldInst="Prot" lnClass="PTRC" lnInst="1" lnType="PTRC_5c3b8cc5-8dd1-4a98-879d-3f3b77cad02a" />
  <LNode iedName="T_110_11" ldInst="Equipment" lnClass="YPTR" lnInst="1" lnType="YPTR_f24ff6dd-9e60-4a39-98d4-d7b590c5a7e4" />
  <TransformerWinding name="W1" desc="W1" type="PTW">
   <Terminal connectivityNode="EE1/110/Q01/PTRQB9CN" substationName="EE1" voltageLevelName="110" bayName="Q01" cNodeName="PTRQB9CN" />
  </TransformerWinding>
  <TransformerWinding name="W2" desc="W2" type="PTW">
   <Terminal connectivityNode="EE1/11/Q02/QB9PTRCN" substationName="EE1" voltageLevelName="11" bayName="Q02" cNodeName="QB9PTRCN" />
  </TransformerWinding>
 </PowerTransformer>
 <VoltageLevel name="110" numPhases="3" nomFreq="50" sxy:x="7" sxy:y="0">
  <Voltage unit="V" multiplier="k">110</Voltage>
  <Bay name="Q01" sxy:x="8" sxy:y="11">
   <LNode iedName="Q01_110" ldInst="Prot" lnClass="PTRC" lnInst="1" lnType="PTRC_5c3b8cc5-8dd1-4a98-879d-3f3b77cad02a" />
   <LNode iedName="Q01_110" ldInst="Prot" lnClass="PTOC" lnInst="1" lnType="PTOC_b024441a-d7f5-4518-aa5f-0ae5428b851e" />
   <LNode iedName="Q01_110" ldInst="Prot" lnClass="PDIF" lnInst="1" lnType="PDIF_67e4c3ef-ee9e-4cc9-839b-8019066956e9" />
   <ConductingEquipment name="QB1" type="DIS" sxy:x="5" sxy:y="2">
    <LNode iedName="Q01_110" ldInst="Control" lnClass="CSWI" lnInst="2" lnType="CSWI_9abea116-d9ce-4a0d-acbe-cdd0dcd12882" />
    <Terminal connectivityNode="EE1/110/Q01/QA1QB1CN" substationName="EE1" voltageLevelName="110" bayName="Q01" cNodeName="QA1QB1CN" />
    <Terminal connectivityNode="EE1/110/110/CN" substationName="EE1" voltageLevelName="110" bayName="110" cNodeName="CN" />
   </ConductingEquipment>
   <ConductingEquipment name="QA1" type="CBR" sxy:x="5" sxy:y="6">
    <LNode iedName="Q01_110" ldInst="Control" lnClass="XCBR" lnInst="1" lnType="XCBR_e202effa-d30c-4fe1-a7f1-8eb6befeec47" />
    <LNode iedName="Q01_110" ldInst="Control" lnClass="CSWI" lnInst="1" lnType="CSWI_477af066-36cd-44f2-acfe-7408afbf1a85" />
    <Terminal connectivityNode="EE1/110/Q01/QB9QA1CN" substationName="EE1" voltageLevelName="110" bayName="Q01" cNodeName="QB9QA1CN" />
    <Terminal connectivityNode="EE1/110/Q01/QA1QB1CN" substationName="EE1" voltageLevelName="110" bayName="Q01" cNodeName="QA1QB1CN" />
   </ConductingEquipment>
   <ConductingEquipment name="QB9" type="DIS" sxy:x="5" sxy:y="10">
    <LNode iedName="Q01_110" ldInst="Control" lnClass="CSWI" lnInst="3" lnType="CSWI_44ac8522-baf0-4eb8-bd9a-eccf8435b5c4" />
    <Terminal connectivityNode="EE1/110/Q01/PTRQB9CN" substationName="EE1" voltageLevelName="110" bayName="Q01" cNodeName="PTRQB9CN" />
    <Terminal connectivityNode="EE1/110/Q01/QB9QA1CN" substationName="EE1" voltageLevelName="110" bayName="Q01" cNodeName="QB9QA1CN" />
   </ConductingEquipment>
   <ConnectivityNode name="PTRQB9CN" pathName="EE1/110/Q01/PTRQB9CN" sxy:x="5" sxy:y="17" />
   <ConnectivityNode name="QB9QA1CN" pathName="EE1/110/Q01/QB9QA1CN" sxy:x="5" sxy:y="8" />
   <ConnectivityNode name="QA1QB1CN" pathName="EE1/110/Q01/QA1QB1CN" sxy:x="5" sxy:y="4" />
  </Bay>
 </VoltageLevel>
</Substation>
```

Figure 8: Configured SSD in SCL format.

3.3.2 Configured IED description (CID) in SCD file

Once the ICD files are utilised to build the SCD file using a SCT, details of the communication between the IEDs, and the communication between IEDs and SCADA systems are updated in the SCL files. The details of this can be extracted from the SCT as Configured IED description (CID) files. Fig. 9 demonstrates the communication section extracted from the SCD files. As seen in the figure, the SCL code initially contains the SubNetwork name and details. Subsequently, the connected access points (AP) of each IEDs in the substation system are displayed. Each ‘ConnectedAP’ section lists the details of the IP address, Generic Substation Event (GSE) details and Physical Connection details. GSE details include the datapoints that are to be published by a particular IED in the form of GCB, and also the MAC-address details of the IED. The Physical Connection contains the details of the ports through which the IEDs are connected between each other and to the SCADA/HMI (Human Machine Interface).

```
<Communication>
  <SubNetwork name="SubNet" type="8-MMS" desc="">
    <BitRate unit="b/s" multiplier="M">100</BitRate>
    <ConnectedAP iedName="E00FSLOI132" apName="P1">
      <Address>
        <P type="IP" xsi:type="tP_IP">10.10.2.234</P>
        <P type="IP-SUBNET" xsi:type="tP_IP-SUBNET">255.255.255.0</P>
        <P type="IP-GATEWAY" xsi:type="tP_IP-GATEWAY">0.0.0.0</P>
      </Address>
      <PhysConn type="Connection">
        <P type="Port">1</P>
        <P type="Type">Connection</P>
        <P type="Cable">C12</P>
      </PhysConn>
      <PhysConn type="Connection">
        <P type="Port">2</P>
        <P type="Type">Connection</P>
        <P type="Cable">C09</P>
      </PhysConn>
    </ConnectedAP>
    <ConnectedAP iedName="E01F11LP1" apName="AP1">
      <Address>
        <P type="IP" xsi:type="tP_IP">10.10.2.11</P>
        <P type="IP-SUBNET" xsi:type="tP_IP-SUBNET">255.255.255.0</P>
        <P type="IP-GATEWAY" xsi:type="tP_IP-GATEWAY">0.0.0.0</P>
      </Address>
      <GSE ldInst="System" cbName="gcb01">
        <Address>
          <P type="VLAN-ID" xsi:type="tP_VLAN-ID">000</P>
          <P type="VLAN-PRIORITY" xsi:type="tP_VLAN-PRIORITY">4</P>
          <P type="MAC-Address" xsi:type="tP_MAC-Address">01-0C-CD-01-00-00</P>
          <P type="APPID" xsi:type="tP_APPID">0000</P>
        </Address>
        <MinTime unit="s" multiplier="m">4</MinTime>
        <MaxTime unit="s" multiplier="m">10000</MaxTime>
      </GSE>
      <PhysConn type="Connection">
        <P type="Port">1</P>
        <P type="Type">Connection</P>
        <P type="Cable">C10</P>
      </PhysConn>
      <PhysConn type="Connection">
        <P type="Port">2</P>
        <P type="Type">Connection</P>
        <P type="Cable">C07</P>
      </PhysConn>
    </ConnectedAP>
    <ConnectedAP iedName="E01F11LP2" apName="S1">
      <Address>
        <P type="IP" xsi:type="tP_IP">10.10.2.12</P>
        <P type="IP-SUBNET" xsi:type="tP_IP-SUBNET">255.255.255.0</P>
        <P type="IP-GATEWAY" xsi:type="tP_IP-GATEWAY">0.0.0.0</P>
      </Address>
      <GSE ldInst="LD0" cbName="gcbStrBF">
        <Address>
          <P type="VLAN-ID" xsi:type="tP_VLAN-ID">000</P>
          <P type="VLAN-PRIORITY" xsi:type="tP_VLAN-PRIORITY">4</P>
          <P type="MAC-Address" xsi:type="tP_MAC-Address">01-0C-CD-01-00-00</P>
          <P type="APPID" xsi:type="tP_APPID">0000</P>
        </Address>
        <MinTime unit="s" multiplier="m">4</MinTime>
        <MaxTime unit="s" multiplier="m">10000</MaxTime>
      </GSE>
      <PhysConn type="Connection">
        <P type="Port">1</P>
        <P type="Type">Connection</P>
        <P type="Cable">C11</P>
      </PhysConn>
      <PhysConn type="Connection">
        <P type="Port">2</P>
        <P type="Type">Connection</P>
        <P type="Cable">C08</P>
      </PhysConn>
    </ConnectedAP>
</SubNetwork>
```

Figure 9: CID (Communication Section) in SCL format.

3.4 System/Substation Exchange Description (SED) file:

The SG-ML utilizes the IEC 61850 SED (System Exchange Description) file for configuring the IEDs in both the substations for enabling the inter-substation communication. Prior to configuring inter-substation communication, the individual substations are already configured and their SCD files are available.

Utilizing the SCD files of two substations SED file is created. The SED file contains the information related to IEDs involved in inter-substation communication from both the substations. The SED file contains information about the electrical connection between the two substations and the communication network information, control blocks and semantics (LN model) of IEDs involved in inter-substation communication.

Using the required portions from the SED file and a proprietary XML file a virtual cyber range for inter-substation communication is developed. The proprietary XML file contains the necessary additional parameters required for power system and cybernetwork simulation.

Example:

Fig. 10 shows the topology of three substation model and the IEDs associated to it. An SED file for inter-substation communication between substation 1 (S-1) and substation 2 (S-2) over Line L2 is considered and discussed here. Differential protection scheme is considered on Line L2. The MIED12 sends the sample measured values to PIED12 and PIED51. Similarly, the MIED51 sends the sample measured values to PIED51 and PIED12.

The main parts of the SED file for the inter-substation communication is shown in Figs. 11, 12 and 13. Fig. 11 shows the part of SED file related to the electrical connection and terminal nodes between two substations.

Figs. 12 and 13 shows the communication portion of SED file. Fig. 12 shows the Subnet of substation 1 which contains additional information of the IEDs from substation 2 such as the network information, control blocks and semantics (LN model) of IEDs from substation 2. Similarly, the Fig. 13 shows the Subnet information of substation 2 containing information of IEDs from substation 1.

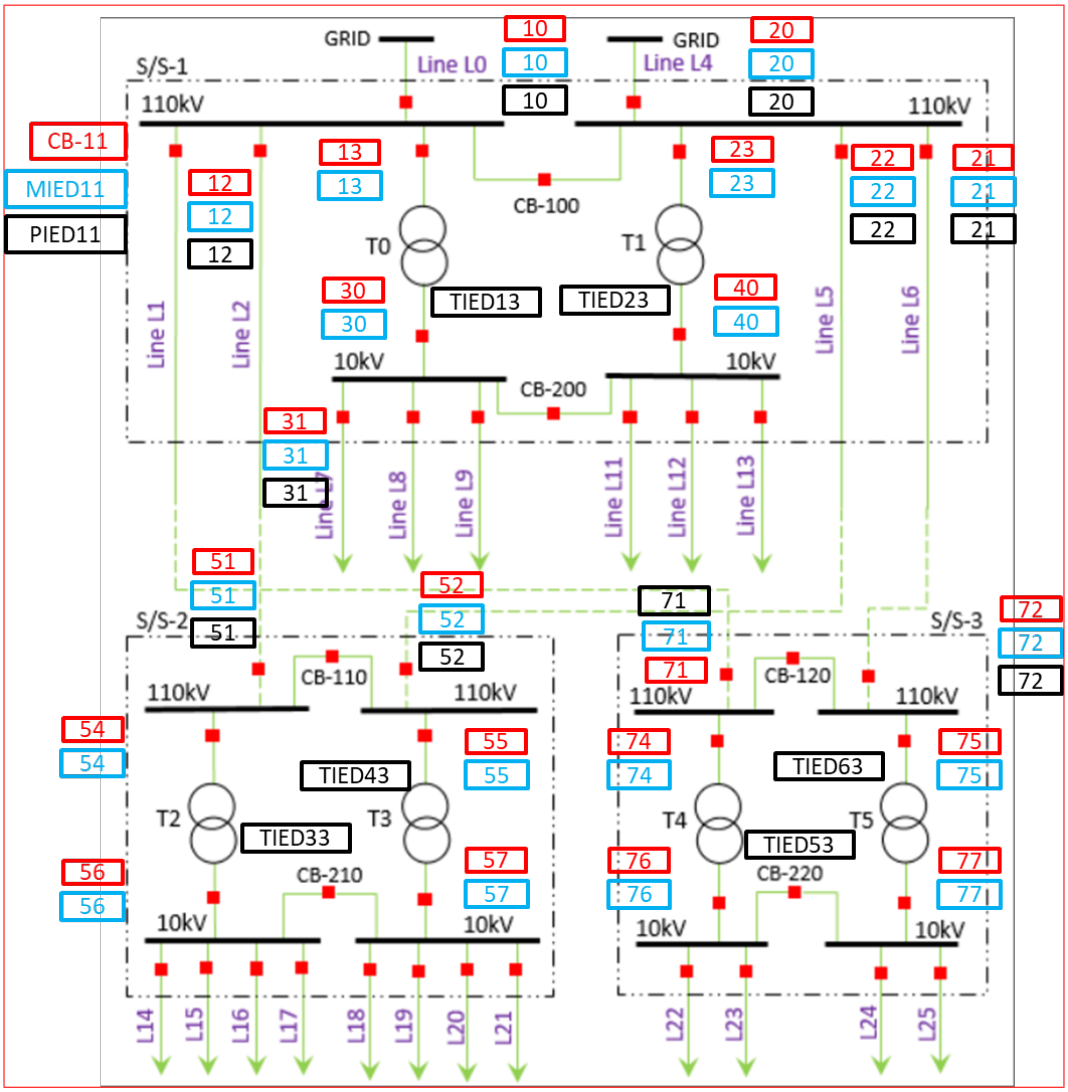


Figure 10: Topology of three substaiton model and the IEDs associated.

```
</Substation name = “S1” desc=”substation”>
</substation>
</Substation name = “S2” desc=”substation”>
</substation>

</Substation name = “LineL2” desc=”Line between S1 and S2”>
<VoltageLevel name="H1" desc="Line Voltage Level">
<Bay name="L2" desc="Bay" sxy:x="55" sxy:y="62" sxy:dir="vertical">
<ConductingEquipment name="LL2" desc="Overhead line" type="LIN" sxy:x="2" sxy:y="12">
<Terminal name="S1H1B4N1" connectivityNode="S1/D1/B4/N1" substationName="S1" voltageLevelName="H1" bayName="B4" cNodeName="N1" />
<Terminal name="S2H1B1N1" connectivityNode="S2/H1/B1/N1" substationName="S2" voltageLevelName="H1" bayName="B1" cNodeName="N1" />
</ConductingEquipment>
</Bay>
</VoltageLevel>
</substation>

<Communication>
<SubNetwork name="S1net" desc="IEC61850 through both stations" type="8-MMS">
<ConnectedAP iedName="PIED12" apName="S1">
<Address>
<P type="SA">0</P>
<P type="IP">192.168.1.12</P>
<P type="IP-SUBNET">255.255.0.0</P>
```

Figure 11: Portion of SED file related to the electrical connection and terminal nodes between two substations.

```
</Bay>
</VoltageLevel>
</substation>


<Communication>
<SubNetwork name="S1net" desc="IEC61850 through both stations" type="8-MMS">
<ConnectedAP iedName="PIED12" apName="S1">
<Address>
<P type="SA">0</P>
<P type="IP">192.168.1.12</P>
<P type="IP-SUBNET">255.255.0.0</P>
<P type="OSI-AP-Title">1,3,9999,23</P>
<P type="OSI-AE-Qualifier">23</P>
<P type="OSI-TSEL">0001</P>
<P type="OSI-PSEL">00000001</P>
<P type="OSI-SSEL">0001</P>
</Address>
</ConnectedAP>
<ConnectedAP iedName="MUIED51" apName="S1">
<Address>
<P type="SA">0</P>
<P type="IP">192.169.1.2</P>
<P type="IP-SUBNET">255.255.0.0</P>
<P type="OSI-AP-Title">1,3,9999,23</P>
<P type="OSI-AE-Qualifier">23</P>
<P type="OSI-TSEL">0001</P>
<P type="OSI-PSEL">00000001</P>
<P type="OSI-SSEL">0001</P>
</Address>
<SMV cbName="L2Diff51-R-SV" desc="SV from MU51" ldInst=”LD1">
           <Address>
                          <P type="IP">238.0.0.2</P>
                          <P type="APPID">0001</P>
                          <P type="VLAN-ID">0</P>
                          <P type="VLAN-PRIORITY">4</P>
            </Address>
</SMV>
</ConnectedAP>
</SubNetwork>

<SubNetwork name="S2net" desc="IEC61850 through both stations" type="8-MMS">
<ConnectedAP iedName="PIED51" apName="S2">
<Address>
<P type="SA">0</P>
<P type="IP">192.169.1.3</P>
```

Figure 12: Portion of SED file related to Subnet communication information of substation 1 with info of IEDs from substation 2.

```
<SubNetwork name="S2net" desc="IEC61850 through both stations" type="8-MMS">
<ConnectedAP iedName="PIED51" apName="S2">
<Address>
<P type="SA">0</P>
<P type="IP">192.169.1.3</P>
<P type="IP-SUBNET">255.255.0.0</P>
<P type="OSI-AP-Title">1,3,9999,23</P>
<P type="OSI-AE-Qualifier">23</P>
<P type="OSI-TSEL">0001</P>
<P type="OSI-PSEL">00000001</P>
<P type="OSI-SSEL">0001</P>
</Address>
</ConnectedAP>
<ConnectedAP iedName="MUIED12" apName="S2">
<Address>
<P type="SA">0</P>
<P type="IP">192.168.1.11</P>
<P type="IP-SUBNET">255.255.0.0</P>
<P type="OSI-AP-Title">1,3,9999,23</P>
<P type="OSI-AE-Qualifier">23</P>
<P type="OSI-TSEL">0001</P>
<P type="OSI-PSEL">00000001</P>
<P type="OSI-SSEL">0001</P>
</Address>
<SMV cbName="L2Diff12-R-SV" desc="SV from MU12" ldInst=”LD1">
          <Address>
                          <P type="IP">238.0.0.1</P>
                          <P type="APPID">0002</P>
                          <P type="VLAN-ID">0</P>
                          <P type="VLAN-PRIORITY">4</P>
           </Address>
</SMV>
</ConnectedAP>
<ConnectedAP iedName="CB-10" apName="S2">
<Address>
<P type="SA">0</P>
<P type="IP">192.168.1.1</P>
<P type="IP-SUBNET">255.255.0.0</P>
<P type="OSI-AP-Title">1,3,9999,23</P>
<P type="OSI-AE-Qualifier">23</P>
<P type="OSI-TSEL">0001</P>
<P type="OSI-PSEL">00000001</P>
<P type="OSI-SSEL">0001</P>
</Address>
<GSE ldInst="LD1" cbName="CB10Trip-R-GOOSE">
              <Address>
                       <P type="IP">238.0.1.1</P>
                       <P type="APPID">0005</P>
                       <P type="VLAN-ID">0</P>
                       <P type="VLAN-PRIORITY">4</P>
              </Address>
                   <MinTime multiplier="m" unit="s">1000</MinTime>
                   <MaxTime multiplier="m" unit="s">60000</MaxTime>
</GSE>
</ConnectedAP>
<ConnectedAP iedName="CB-110" apName="S2">
<Address>
```

Figure 13: Portion of SED file related to Subnet communication information of substation 1 with info of IEDs from substation 2.

4 SCL (SSD) to XML Schema

Aim	System Specification Description to eXtensible Markup Language
Programming Language	Python [3]
Outcome	SCL Processor to generate Power System Simulation Model

In the section, the focus is on creating an XML schema from the SSD file using parser in Python. The framework for the process is depicted in Fig. 14.

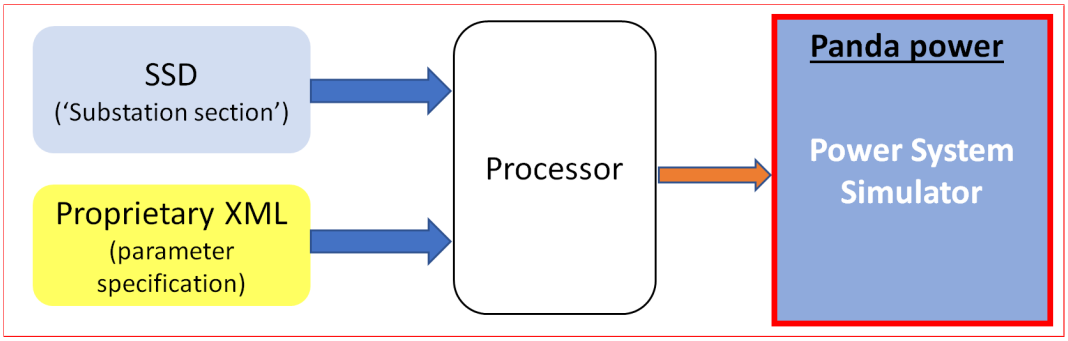


Figure 14: Framework for SSD to XML process.

In order to achieve this two files are required. As depicted in Fig. 1, one file is an SCL file and another is an Proprietary XML file. The 'substation' section in the SSD file describes the physical components and their connections. However, this file does not include the parameter specification of the physical components in the substation. Therefore, an additional Proprietary XML file is created that describes the specification of each component in the substation. Merging these two files via a parser in Python, the outcome would be a processor that process the SCL and the proprietary file to generate the power system simulation model. The following figures (Fig. 15 and Fig. 16) show the LN related to a physical component in SSD and their parameter specification in the corresponding Proprietary XML files, respectively.

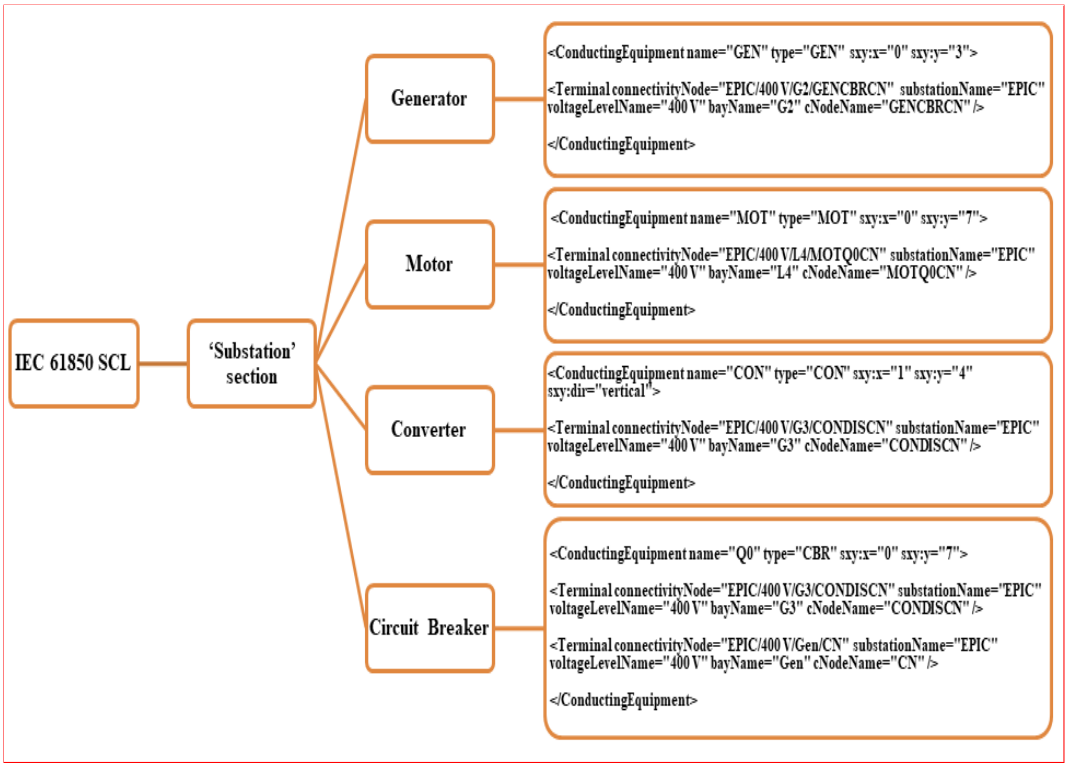


Figure 15: IEC 61850 SCL ('substation' section).

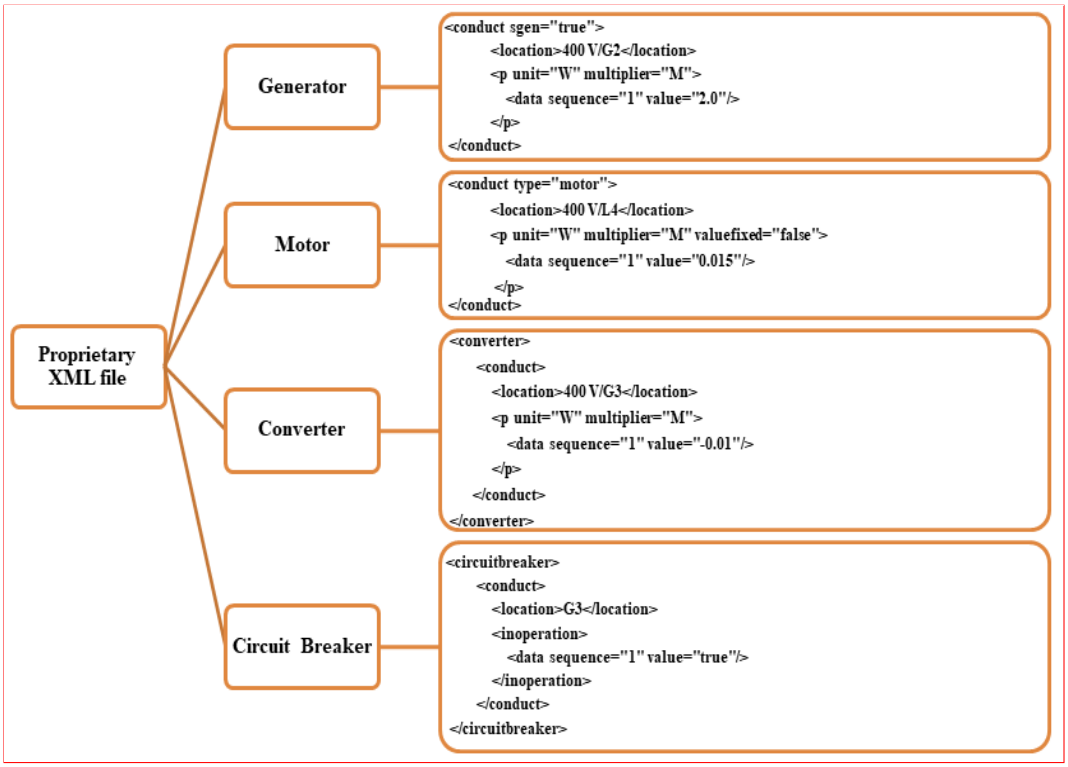


Figure 16: Proprietary XML (parameter specification).

5 Virtual IED

Aim	XML Schema for ICD and Threshold Logic
Software	Eclipse [4]
Outcome	Virtual IED

The following requirements should be addressed to create a virtual IED.

- Creating an XML schema for the input ICD files
- Creating an XML schema for the Proprietary settings file

The framework for the aforementioned process is depicted in Fig. 17.

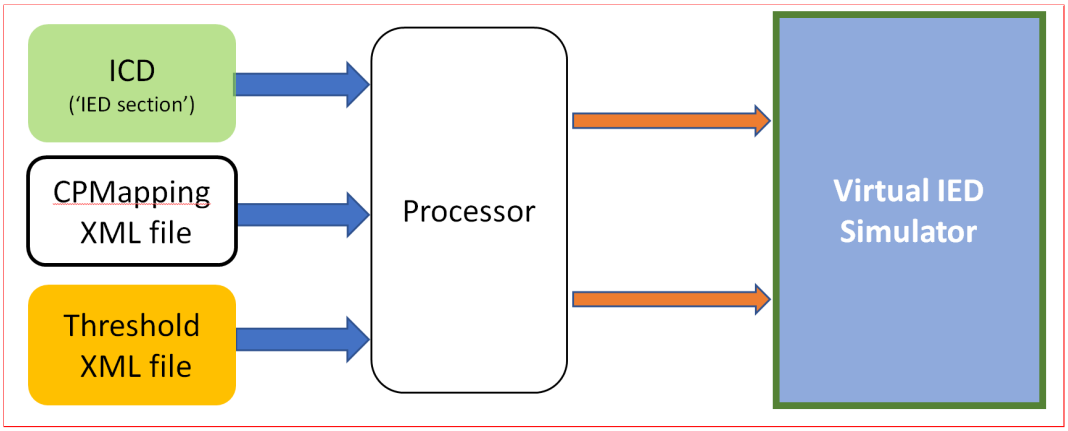


Figure 17: Framework for Virtual IED process.

5.1 XML schema for input ICD files

In order to create the XML schema for input ICD files, the attributes that are needed for the project are initially identified. As such, an ICD file that contains the attributes related to physical measurements such as voltages, currents and power is created. Fig. 18 depicts the LD ('MEAS'). This section contains the details of the data attributes associated with every LN instances, the data points that are used to create the datasets, and the allocation of datasets to the report control blocks.

From Fig. 18, the attributes associated with each physical measurement can be derived. Subsequently, an XML schema has to be created, to map the physical measurements with the IED attributes. Fig. 19 demonstrates the schema for mapping between the physical (real-time measurements) and cyber (IED attributes) aspects for a substation. An example of the XML file that describes the physical and cyber parameters for the system is shown in Fig. 20. The example depicted in Fig. 20 represents the attributes related to 'Load 0' in Fig. 2.2. The three-phase load voltages for 'Load0' are measured, and are assigned to physical

```
<LDevice inst="MEAS">
 <LN0 lnClass="LLN0" inst="" lnType="MEAS.LLN0">
  <DataSet name="CurrDS" desc="(created on Monday, 09 November 2020 15:07:29)">
   <FCDA ldInst="MEAS" lnInst="0" lnClass="MMXU" doName="A.phsA" daName="cVal" fc="MX" />
   <FCDA ldInst="MEAS" lnInst="0" lnClass="MMXU" doName="A.phsB" daName="cVal" fc="MX" />
   <FCDA ldInst="MEAS" lnInst="0" lnClass="MMXU" doName="A.phsC" daName="cVal" fc="MX" />
  </DataSet>
  <DataSet name="VoltDS" desc="(created on Monday, 09 November 2020 15:07:55)">
   <FCDA ldInst="MEAS" lnInst="0" lnClass="MMXU" doName="PhV.phsA" daName="cVal" fc="MX" />
   <FCDA ldInst="MEAS" lnInst="0" lnClass="MMXU" doName="PhV.phsB" daName="cVal" fc="MX" />
   <FCDA ldInst="MEAS" lnInst="0" lnClass="MMXU" doName="PhV.phsC" daName="cVal" fc="MX" />
  </DataSet>
  <DataSet name="PowDS" desc="(created on Monday, 09 November 2020 15:08:17)">
   <FCDA ldInst="MEAS" lnInst="0" lnClass="MMXU" doName="TotVA" daName="mag" fc="MX" />
   <FCDA ldInst="MEAS" lnInst="0" lnClass="MMXU" doName="TotVAr" daName="mag" fc="MX" />
   <FCDA ldInst="MEAS" lnInst="0" lnClass="MMXU" doName="TotW" daName="mag" fc="MX" />
  </DataSet>
  <ReportControl name="CurrCB" bufTime="500" buffered="true" confRev="40010" intgPd="1000" indexed="true" datSet="CurrDS" desc="(created on Thursday, October 1, 2020 12:20:23 PM)">
   <TrgOps gi="true" dchg="true" qchg="true" dupd="true" period="true" />
   <OptFields bufOvfl="true" seqNum="true" timeStamp="true" dataSet="true" reasonCode="true" dataRef="true" entryID="true" configRef="true" />
   <RptEnabled max="1" />
  </ReportControl>
  <ReportControl name="VoltCB" bufTime="500" buffered="true" confRev="10010" intgPd="1000" indexed="true" datSet="VoltDS" desc="(created on Thursday, October 1, 2020 12:25:43 PM)">
   <TrgOps gi="true" dchg="true" qchg="true" dupd="true" period="true" />
   <OptFields bufOvfl="true" seqNum="true" timeStamp="true" dataSet="true" reasonCode="true" dataRef="true" entryID="true" configRef="true" />
   <RptEnabled max="1" />
  </ReportControl>
  <ReportControl name="PowCB" bufTime="500" buffered="true" confRev="10010" intgPd="1000" indexed="true" datSet="PowDS" desc="(created on Thursday, October 1, 2020 12:28:01 PM)">
   <TrgOps gi="true" dchg="true" qchg="true" dupd="true" period="true" />
   <OptFields bufOvfl="true" seqNum="true" timeStamp="true" dataSet="true" reasonCode="true" dataRef="true" entryID="true" configRef="true" />
   <RptEnabled max="1" />
  </ReportControl>
  <GSEControl name="CurrGCB" type="GOOSE" fixedOffs="false" confRev="0" securityEnable="None" appID="NULL" datSet="CurrDS" desc="(created on Monday, 09 November 2020 15:09:05)" />
  <GSEControl name="VoltGCB" type="GOOSE" fixedOffs="false" confRev="0" securityEnable="None" appID="NULL" datSet="VoltDS" desc="(created on Monday, 09 November 2020 15:09:37)" />
  <GSEControl name="PowGCB" type="GOOSE" fixedOffs="false" confRev="0" securityEnable="None" appID="NULL" datSet="PowDS" desc="(created on Monday, 09 November 2020 15:09:50)" />
 </LN0>
 <LN lnClass="MMXU" inst="0" lnType="MEAS.MMXU0" />
</LDevice>
```

Figure 18: Input ICD file for creating XML schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="CPMappings">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="CPMapping" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Physical" type ="xs:string"/>
                            <xs:element name="Cyber" type="xs:string"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Figure 19: XML schema for cyber-physical mapping.

attributes ‘Load0.Voltage.phsA’, ‘Load0.Voltage.phsB’ and ‘Load0.Voltage.phsC’. Now the three physical attributes are mapped to their cyber attributes ‘IED1.MMXU.PhV.phsA.cVal’,

'IED1.MMXU.PhV.phsB.cVal' and 'IED1.MMXU.PhV.phsC.cVal', respectively. These cyber attributes are present in the IED that has been connected to 'Load0' line in Fig. 2.2.

```
<?xml version="1.0" encoding="UTF-8"?>
<CPMapping>
    <Physical>
        Load0.Voltage.phsA
    </Physical>
    <Cyber>
        IED1.MMXU.PhV.phsA.cVal
    </Cyber>
</CPMapping>

<CPMapping>
    <Physical>
        Load0.Voltage.phsB
    </Physical>
    <Cyber>
        IED1.MMXU.PhV.phsB.cVal
    </Cyber>
</CPMapping>

<CPMapping>
    <Physical>
        Load0.Voltage.phsC
    </Physical>
    <Cyber>
        IED1.MMXU.PhV.phsC.cVal
    </Cyber>
</CPMapping>
```

Figure 20: XML file for cyber-physical mapping.

5.2 XML schema for Proprietary Settings file

The implementation of virtual IED has to include the logic that are implemented for providing protection functionalities. Therefore, an XML schema that contains the threshold values has to be created. The logic considered in this schema are over over-/under-current, over-/under-voltage, reverse power protections. Fig. 21 exemplifies the threshold conditions for each protection. As can be seen, two protection conditions has been implemented. The first

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="ThresholdS">
        <xs:complexType>
            <xs:all>
                <xs:element name="Thresholds">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="IED" maxOccurs="unbounded">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="PTOC50" minOccurs="0" maxOccurs="unbounded">
                                            <xs:complexType>
                                                <xs:all>
                                                    <xs:element name="Threshold" type="PositiveDecimalType"/>
                                                </xs:all>
                                                <xs:attribute name="instance" type="xs:positiveInteger" use="required"/>
                                            </xs:complexType>
                                        </xs:element>
                                        <xs:element name="PTOC51" minOccurs="0" maxOccurs="unbounded">
                                            <xs:complexType>
                                                <xs:all>
                                                    <xs:element name="CurrentNominal" type="PositiveDecimalType"/>
                                                </xs:all>
                                                <xs:attribute name="instance" type="xs:positiveInteger" use="required"/>
                                            </xs:complexType>
                                        </xs:element>
                                        <xs:element name="PTOV59" minOccurs="0" maxOccurs="unbounded">
                                            <xs:complexType>
                                                <xs:all>
                                                    <xs:element name="AlarmThreshold">
                                                        <xs:complexType>
                                                            <xs:all>
                                                                <xs:element name="Limit_p.u." type ="PositiveDecimalType"/>
                                                                <xs:element name="Period_s" type ="PositiveDecimalType"/>
                                                            </xs:all>
                                                        </xs:complexType>
                                                    </xs:element>
                                                    <xs:element name="TripThreshold">
                                                        <xs:complexType>
                                                            <xs:all>
                                                                <xs:element name="Limit_p.u." type ="PositiveDecimalType"/>
                                                                <xs:element name="Period_s" type ="PositiveDecimalType"/>
                                                            </xs:all>
                                                        </xs:complexType>
                                                    </xs:element>
                                                </xs:all>
                                                <xs:attribute name="instance" type="xs:positiveInteger" use="required"/>
                                            </xs:complexType>
                                        </xs:element>
                                        <xs:element name="PTUV27" minOccurs="0" maxOccurs="unbounded">
                                            <xs:complexType>
                                                <xs:all>
                                                    <xs:element name="AlarmThreshold">
                                                        <xs:complexType>
                                                            <xs:all>
                                                                <xs:element name="Limit_p.u." type ="PositiveDecimalType"/>
                                                                <xs:element name="Period_s" type ="PositiveDecimalType"/>
                                                            </xs:all>
                                                        </xs:complexType>
                                                    </xs:element>
                                                </xs:all>
                                                <xs:attribute name="instance" type="xs:positiveInteger" use="required"/>
                                            </xs:complexType>
                                        </xs:element>
                                        <xs:element name="PTUV27" minOccurs="0" maxOccurs="unbounded">
                                            <xs:complexType>
                                                <xs:all>
                                                    <xs:element name="AlarmThreshold">
                                                        <xs:complexType>
                                                            <xs:all>
                                                                <xs:element name="Limit_p.u." type ="PositiveDecimalType"/>
                                                                <xs:element name="Period_s" type ="PositiveDecimalType"/>
                                                            </xs:all>
                                                        </xs:complexType>
                                                    </xs:element>
                                                    <xs:element name="TripThreshold">
                                                        <xs:complexType>
                                                            <xs:all>
                                                                <xs:element name="Limit_p.u." type ="PositiveDecimalType"/>
                                                                <xs:element name="Period_s" type ="PositiveDecimalType"/>
                                                            </xs:all>
                                                        </xs:complexType>
                                                    </xs:element>
                                                </xs:all>
                                                <xs:attribute name="instance" type="xs:positiveInteger" use="required"/>
                                            </xs:complexType>
                                        </xs:element>
                                        <xs:element name="PDOP32" minOccurs="0" maxOccurs="unbounded">
                                            <xs:complexType>
                                                <xs:all>
                                                    <xs:element name="Threshold_p.u." type="PositiveDecimalType"/>
                                                </xs:all>
                                                <xs:attribute name="instance" type="xs:positiveInteger" use="required"/>
                                            </xs:complexType>
                                        </xs:element>
                                    </xs:sequence>
                                    <xs:attribute name="name" type="xs:string" use="required"/>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                        <xs:attribute name="name" type="xs:string" use="required" fixed="Protection"/>
                    </xs:complexType>
                </xs:element>
            </xs:all>
        </xs:complexType>
    </xs:element>
    <xs:simpleType name="PositiveDecimalType">
        <xs:restriction base="xs:decimal">
            <xs:minInclusive value="0"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>
```

Figure 21: XML schema for Proprietary File (Thresholds Schema).

condition is an ‘Alarm threshold’, the outcome would be an error message. The second condition is ‘Trip threshold’ at which point a trip command will be issued by the IED. Table. 1 tabulates the threshold limits included in the proprietary settings file. An example of the XML file that describes the threshold settings for the proprietary settings file for the protections implemented in the IED is shown in Fig. 22. Table. 2 tabulates the explanation of the attributes utilised in Fig. 22.

Table 1: Threshold limits for Proprietary Settings File

Logical Nodes	Description	Threshold
PTOC (50)	Instantaneous over-current	3 to 4 times * nominal current
PTOC (51)	Time over-current	1.05p.u. (starts picking up)
PTOV (59)	Over-voltage	1.1p.u. (10s) (alarm) & 1.2p.u. (2s) (trip)
PTUV (27)	Under-voltage	0.8 p.u. (10s) (alarm) & 0.7 p.u. (2s) (trip)
PDOP (32)	Reverse Power	Any small amount (trip)

Table 2: Attributes in XML Proprietary Settings file

Attributes	Description
‘IED name’	Represents the name of the IED
‘PTOC’, ‘PTOV’, ‘PDOP’	IEC 61850 based protection
‘instance’	Varies based on the number of devices in a load/line that requires a single type of protection
‘AlarmThreshold’	Raises an alarm (message) in HMI
‘TripThreshold’	Send ‘trip’ command to CB
‘p.u.’	per-unit - the expression of system quantities as fractions of a defined base unit quantity
‘period’	the time after which the IED sends a message or trip command

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Thresholds name="Protection">
    <IED name="ied1">

        <PTOC50 instance="1">
            <Threshold>50</Threshold>
        </PTOC50>
        <PTOC50 instance="2">
            <Threshold>50</Threshold>
        </PTOC50>
        <PTOV59 instance="1">
            <AlarmThreshold>
                <Limit_p.u.>1.1</Limit_p.u.>
                <Period_s>10</Period_s>
            </AlarmThreshold>
            <TripThreshold>
                <Limit_p.u.>1.2</Limit_p.u.>
                <Period_s>2</Period_s>
            </TripThreshold>
        </PTOV59>
        <PTOV59 instance="2">
            <AlarmThreshold>
                <Limit_p.u.>1.1</Limit_p.u.>
                <Period_s>10</Period_s>
            </AlarmThreshold>
            <TripThreshold>
                <Limit_p.u.>1.2</Limit_p.u.>
                <Period_s>2</Period_s>
            </TripThreshold>
        </PTOV59>
        <PDOP32 instance="1">
            <Threshold_p.u.>0.1</Threshold_p.u.>
        </PDOP32>
        <PDOP32 instance="2">
            <Threshold_p.u.>0.1</Threshold_p.u.>
        </PDOP32>
    </IED>
</Thresholds>
```

Figure 22: XML schema for Proprietary File (Thresholds Schema).

6 PLCOpen

Aim	Virtual Programming Logic Controller
Standard & Software	PLCOpen XML [5] & OpenPLC [6]
Outcome	Virtual PLC Auto-configurator

SG-ML utilises PLCOpen XML schema for modelling of PLC logic.

An example of PLCOpen is demonstrated below, in which PLCOpen XML in structured text format is extracted from OpenPLC editor.

```
<pou name="Main_ST" pouType="program">
    <interface>
      <localVars>
        <variable name="PB1" address="%MX0.0">
          <type>
            <BOOL/>
          </type>
        </variable>
        <variable name="PB2" address="%MX0.1">
          <type>
            <BOOL/>
          </type>
        </variable>
        <variable name="LAMP" address="%MX1.0">
          <type>
            <BOOL/>
          </type>
        </variable>
      </localVars>
    </interface>
    <body>
      <ST>
        <xhtml:p><![CDATA[LAMP := NOT(PB2) AND (LAMP OR PB1);]]></xhtml:p>
      </ST>
    </body>
</pou>
```

Figure 23: PLCopen XML: Structured Text.

```
1  LAMP := NOT(PB2) AND (LAMP OR PB1);
2
```

Figure 24: Structured Text representation in OpenPLC editor.

The future work includes the representation of PLCOpen XML from ladder logic, function blocks, etc.

The framework for Virtual PLC Auto-configurator is depicted in Fig. 25.

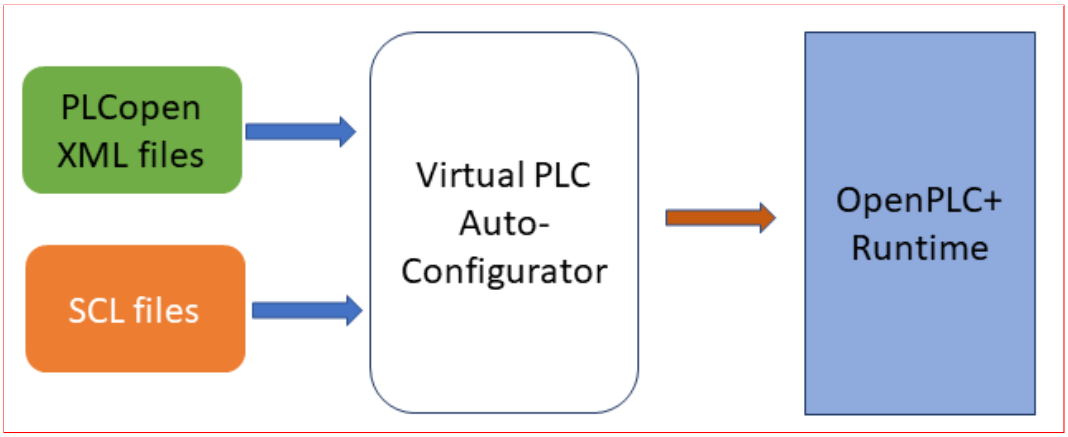


Figure 25: Virtual PLC Auto-configurator.

7 SCADA Configuration

Aim	SCADA Configuration
Software	ScadaBR [7]
Outcome	SCADA Configurator Tool

SG-ML utilises Schema and XML file to model the SCADA configurator tool. The schema contains information regarding the address and the attributes of the information received from PLC. As an intial step, The validator would validate the XML file against the Schema. If the validation is successful, the XML file will proceed to the Scada Tool Configurator. The XML file will be converted to a JSON format and generate a JSON file as the output. This output would be uploaded to ScadaBR for Scada to process the given information. The process is depicted in Fig. 26.

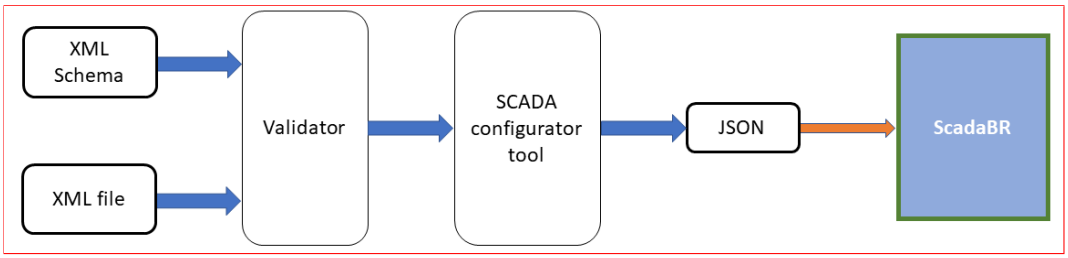


Figure 26: Framework for SCADA configurator.

The important attributes are stored under "dataSources" and "dataPoints" in the schema file. The format of all files in Fig. 26 is depicted below. The description of the attributes in the schema/XML file is tabulated in Table 3.

Table 3: Attributes in XML Proprietary Settings file

Attributes	Description
dataSources	Connection set up to a database from a server, e.g. Customer data base
dataPoint	A piece of data that is collected during monitoring, e.g. Cusomter ID
xid	ID of a data source or data point
type	Type of data source
updatePeriodType	Update period unit of data source in milliseconds, seconds etc.
transportType	Type of protocol which the data source is using
host	IP address of the data source
name	Name of the data source or data point
port	Port number of data source
updatePeriods	Updating frequency of data source
range	Memory range of data points
offset	To offset the Modbus initial register value in data point
modbusDataType	Type of memory stored for data points
engineeringUnits	Measuring units of a data point
dataSourceXid	Data source ID which the data points are connected to
deviceName	Name of the data source which the data points are connect to

```
<xs:element name="dataSources" maxOccurs="unbounded">
    <xs:complexType>
        <xs:all>
            <xs:element name="xid" type ="xs:string"/>
            <xs:element name="type" type ="xs:string"/>
            <xs:element name="updatePeriodType" type ="DSupdatePeriodType"/>
            <xs:element name="transportType" type ="DStransportType"/>
            <xs:element name="host" type ="xs:string"/>
            <xs:element name="name" type ="xs:string"/>
            <xs:element name="port" type ="DSport"/>
            <xs:element name="updatePeriods" type ="DSupdatePeriods"/>
        </xs:all>

<xs:element name="dataPoints" maxOccurs="unbounded">
   <xs:complexType>
       <xs:all>
           <xs:element name="xid" type ="xs:string"/>
           <xs:element name="pointLocator">
               <xs:complexType>
                   <xs:all>
                       <xs:element name="range" type="DPRange"/>
                       <xs:element name="modbusDataType" type="DPmodbusDataType"/>
                       <xs:element name="offset" type="xs:integer"/>
                   </xs:all>
               </xs:complexType>
           </xs:element>
           <xs:element name="engineeringUnits" type ="DPengineeringUnits"/>
           <xs:element name="dataSourceXid" type ="xs:string"/>
           <xs:element name="deviceName" type ="xs:string"/>
           <xs:element name="name" type ="xs:string"/>
       </xs:all>
```

Figure 27: Schema for Datasources and Datapoints.

```xml
<dataSources>
    <xid>DS_051944</xid>
    <type>MODBUS_IP</type>
    <updatePeriodType>MILLISECONDS</updatePeriodType>
    <transportType>TCP</transportType>
    <host>192.168.10.11</host>
    <name>plc1</name>
    <port>502</port>
    <updatePeriods>500</updatePeriods>
</dataSources>
<dataPoints>
    <xid>DP_862782</xid>
    <pointLocator>
        <range>HOLDING_REGISTER</range>
        <modbusDataType>FOUR_BYTE_FLOAT</modbusDataType>
        <offset>2848</offset>
    </pointLocator>
    <engineeringUnits/>
    <dataSourceXid>DS_051944</dataSourceXid>
    <deviceName>plc1</deviceName>
    <name>line_load_percent00</name>
</dataPoints>
```

Figure 28: XML file containing datasource and datapoint.

```python
with open("test1.xml", 'r') as xml_file:
    xml_string = xml_file.read()
    obj = xmltodict.parse(xml_string)
    xml_file.close()

json_string = json.dumps(xmltodict.parse(xml_string), indent=4)
json_data = json.loads(json_string, object_hook=_decode)["root"]

with open("data.json", "w") as json_file:
    save_data = json.dump(json_data, json_file, indent=4)
    json_file.close()
```

Figure 29: Scada Tool Configuator.

```
{
  "dataSources": [
    {
      "xid": "DS_051944",
      "type": "MODBUS_IP",
      "transportType": "TCP",
      "updatePeriodType": "MILLISECONDS",
      "host": "192.168.10.11",
      "name": "plc1",
      "port": 502,
      "updatePeriods": 500
    }
  ],
  "dataPoints": [
    {
      "xid": "DP_862782",
      "pointLocator": {
        "range": "HOLDING_REGISTER",
        "modbusDataType": "FOUR_BYTE_FLOAT",
        "offset": 2848
      },
      "engineeringUnits": null,
      "dataSourceXid": "DS_051944",
      "deviceName": "plc1",
      "name": "line_load_percent00"
    }
  ]
}
```


Figure 30: JSON output.

8 Multi-Substation configuration

8.1 Multi-substation SCD file generation

The parser combines the different SCD files of each substation to create a merged SCD file for multi-substation system. The merged SCD file for multi-substation is concatenation of different 'SubNetwork' sections. Each 'SubNetwork' section contains communication" section which is extracted from the SCD file of each substation. The information of all switches (i.e., 'IED' elements) corresponding to different substations are listed in the merged SCD file for multi-substation. Figure 31 below depicts the framework for combining different SCD files to one merged SCD file for multi-substation system. This merged SCD file is used to emulate the cyber-network topology of the multi-substation system.

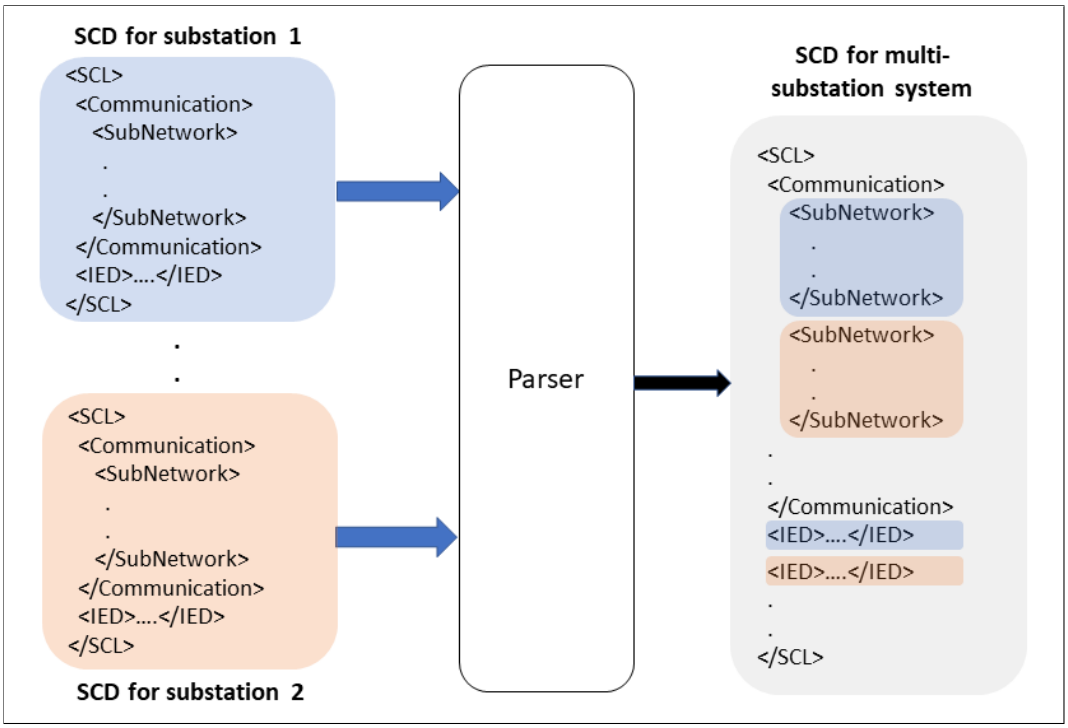


Figure 31: Framework for combining SCD files of different substations.

8.2 Multi-substation topology description (SSD file)

The SSD files contain the description of physical electrical topology of each substation. The description of interconnection between two substations is defined by SED file. A parser tool is developed to create one single merged SSD file for multi-substation system by combining all the SSD files and SED files related to the multi-substation model. Figure 32 depicts the framework for combining different SSD and SED files corresponding to different substations and interconnections respectively, to one final merged SSD file for multi-substation system.

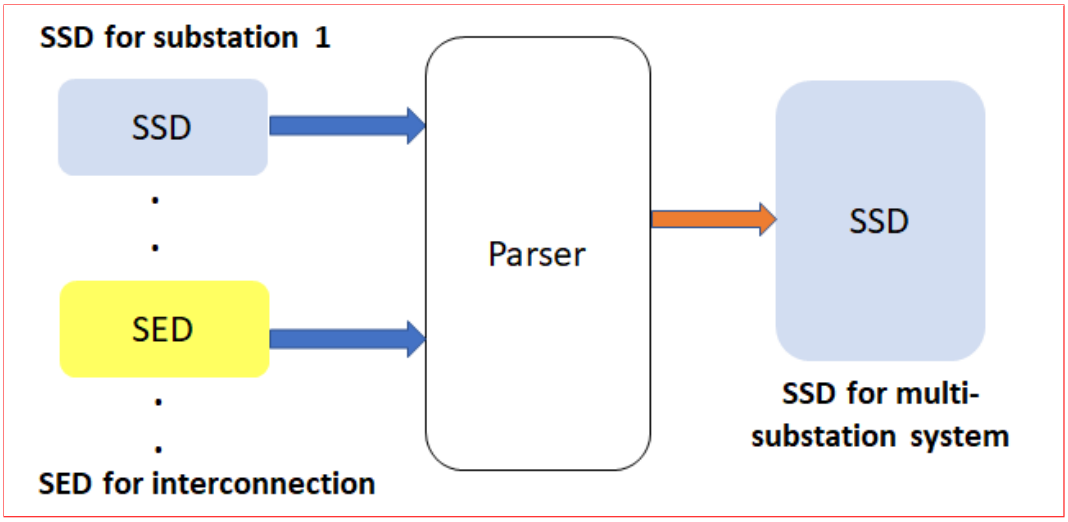


Figure 32: Framework for combining different SSD and SED files corresponding to different substations and interconnections.

The SSD file typically consists of the 'Substation' section, 'VoltageLevel' and 'Bay' sub-sections. The parser concatenates 'Substation' section data from different SSD files to one final SSD file. The 'VoltageLevel' tag in SED file is identified and its contents (i.e. 'Bay' sub-sections) are copied under the similar 'VoltageLevel' tag in final SSD file. This process is repeated for all the SED files. The procedure of creating the merged SSD file is illustrated in Figure 33. Thus, the final SSD file containing the complete electrical topology of multi-substation system is created.

Figure 34 depicts a sample SED file containing information regarding interconnection between two substations namely ADSC_SS1 and ADSC_SS3. The interconnecting link is listed under 'VoltageLevel' 66_1 section and its corresponding 'Bay' section. Figure 35 shows the final SSD file generated by parser which includes the interconnection information from SED file under the 'VoltageLevel' 66_1 section.

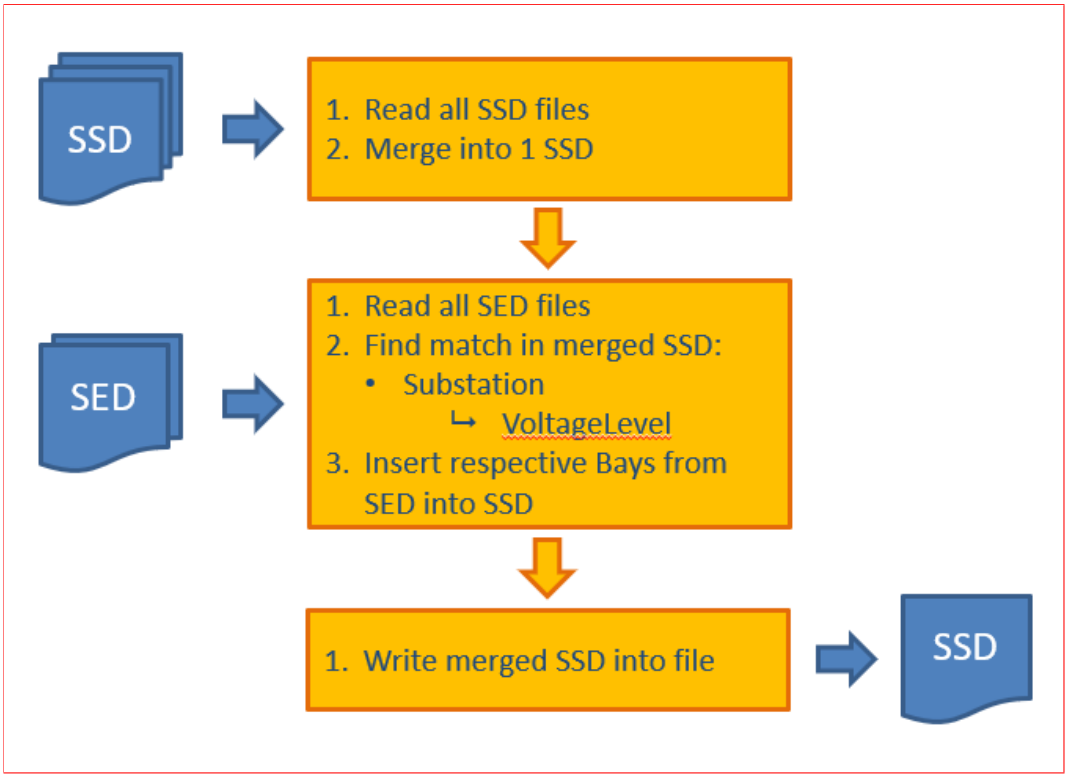


Figure 33: Process for merging SSD and SED files.

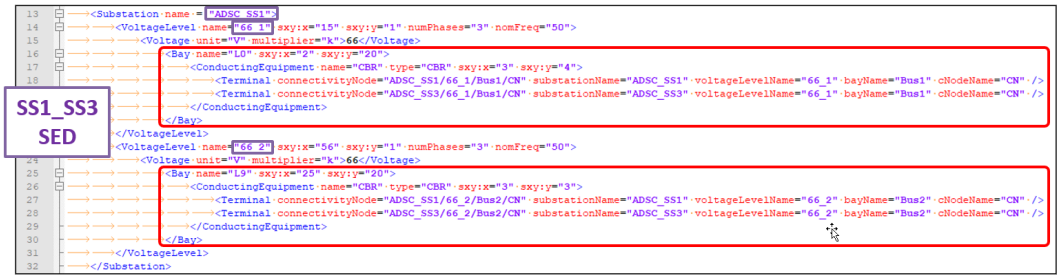


Figure 34: Sample SED file.

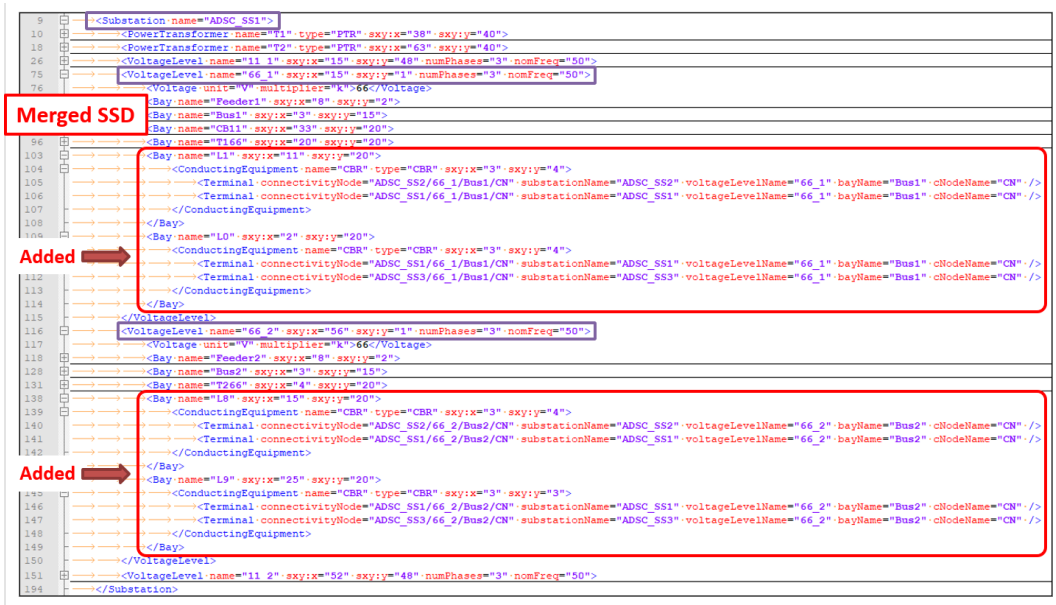


Figure 35: Sample final SSD file.

9 Useful Links

The work has been published in the industry track of Dependable Systems and Networks conference [8]. Our toolchain is open-sourced and are available online at `https://github.com/smartgridadsc/CyberRange`.

The demonstration to the demo video found at `https://osf.io/n6ycb/?view_only=f430f148f70c49b0860eb8ad9e5544ce`.

10 Acknowledgement

This research is supported in part by the National Research Foundation, Singapore, Singapore University of Technology and Design under its National Satellite of Excellence in Design Science and Technology for Secure Critical Infrastructure Grant (NSoE_DeST-SCI2019-0005), and in part by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme.

References

[1] "IEC 61850:2021 SER series," Available at https://webstore.iec.ch/publication/6028.

[2] "Grid software," Available at http://www.gridsoftware.com/products/sclmatrix.html.

[3] "Python," Available at https://www.python.org/.

[4] "Eclipse foundation," Available at https://www.eclipse.org/downloads/.

[5] "IEC 61131-2:2017," Available at https://webstore.iec.ch/publication/31007.

[6] "OpenPLC," Available at https://www.openplcproject.com/.

[7] "ScadaBR," Available at https://www.scadabr.com.br/.

[8] D. Mashima, M. M. Roomi, B. Ng, Z. Kalbarczyk, S. Hussain, and E.-C. Chang, "Towards Automated Generation of Smart Grid Cyber Range for Cybersecurity Experiments and Training." in *In Proc. of DSN 2023 (Industry Track)*.