TRILATERAL: Software Product Line based Multidomain IoT Artifact Generation for Industrial CPS

Aitziber Iglesias, Markel Iglesias-Urkia, Beatriz López-Davalillo, Santiago Charramendieta and Aitor Urbieta

IK4-Ikerlan Technology Research Centre, Information and Communication Technologies Area, P. J. M. Arizmendiarrieta, 2, 20500, Arrasate-Mondragón, Spain

Keywords: Internet of Things, Cyber-Physical System, Domain Specific Language, Software Product Line, IEC 61850.

Abstract: Internet of Things (IoT) devices are usually advanced embedded systems that require functionalities monitoring and control. The design, development and validation of these devices is complex, even more when communication capabilities need to be included. In industrial environments, where safety is of critical importance, reducing this complexity can help to achieve the vision of Industry 4.0 by reducing development time and costs as well as increasing quality. To this end, the use of Model-Driven Engineering (MDE) methodology and the Software Product Line (SPL) paradigm is becoming increasingly important as they help to accelerate and ease the development of software, while reducing bugs and errors. Thus, in this work we present TRILATERAL, a SPL Model Based tool that uses a Domain Specific Language (DSL) to allow users to graphically define the IEC 61850 information model of the Industrial Cyber-Physical System (ICPS). TRILATERAL automatically generates the source code for communicating devices with the monitoring framework, also supporting a variety of communication protocols, namely HTTP-REST, WS-SOAP and CoAP in order to control/monitor any ICPS. In addition, the solution was evaluated deploying it in different industrial domains (Wind Farm, Smart Elevator, Catenary-free Tram) from which we gained important lessons.

1 INTRODUCTION

Industry 4.0 is playing an important role in many industrial domains, such as Smart Grids, Smart Manufacturing and Smart Logistics (Leitão et al., 2016; Suri et al., 2017). This paradigm involves the technical integration of Cyber-Physical Systems (CPS) along with the use of the Internet of Things (IoT) in industrial processes (Kagermann et al., 2013). IoT devices in industrial scenarios are embedded devices that usually have advanced requirements in terms of monitoring and control (Tao et al., 2014). CPS are *"integrations of computation with physical processes. Embedded computers and networks monitor and control the physical processes"* (Kagermann et al., 2013). In Industry 4.0, the monitoring and control of Industrial CPS (ICPS) is becoming essential, as it allows to control it from outside the ICPS and enable automated analysis, decision making, and anomaly detection. Thus, monitoring and controlling enables the transition of a traditional industrial system towards an ICPS. The number of devices an ICPS can have is high, and capturing data and being aware of the state of the ICPS during operation is important (Iglesias et al., 2018). In these scenarios, IoT communication protocols make the communication between the stakeholder and the ICPS possible. Each stakeholder has different needs, hence, monitoring/controlling systems and IoT communication protocols must be adapted.

For this reason, IoT communication protocols will vary according to the stakeholder's needs or due to the industrial environment. Likewise, every ICPS is composed by different devices even if they are from the same domain. Therefore, in industrial environments, where safety is of critical importance, reducing complexity of the design, development and validation can help to achieve the vision of Industry 4.0. In this manner, development time and costs are reduced while increasing quality. Thus, Software Product Line (SPL) paradigm can be beneficial to improve productivity and reduce costs (Capilla et al., 2014). In addition, a Domain Specific Language (DSL) promotes effective communication with stakeholders thanks to the simplification of complex codes (Fowler, 2011). Thanks to DSL, the flexibility of a system is improved in ad-

Iglesias, A., Iglesias-Urkia, M., López-Davalillo, B., Charramendieta, S. and Urbieta, A.
TRILATERAL: Software Product Line based Multidomain IoT Artifact Generation for Industrial CPS.
DOI: 10.5220/0007343500620071
In *Proceedings of the 7th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2019)*, pages 62-71
ISBN: 978-989-758-358-2

dition to providing a more immediate response to the stakeholder (Kelly and Tolvanen, 2008), and with a SPL users can produce specific systems by reusing common elements and configuring the variability.

In order to transfer ICPS data via IoT, we analyzed an international standard, IEC 61850. The International Electrotechnical Commission (IEC) defined the IEC 61850 standard (TC-57, 2003) to model, control and monitor electrical substations. The standard defines a Basic Information Model, services to interact with it, and some recommendations for the use of different communication protocols. It is divided in different parts for general specifications, configuration, defining the model and communications, and testing.

In the Industry 4.0 context, the needs of stakeholders and industry to control and monitor an ICPS have to be taken into account to provide a common solution, both in terms of devices and communications. That is why in this paper TRILATERAL (sofTware pRoduct lIne based muLtidomain iot ArTifact gEneration for industRiAL cps) is presented, a tool to join IEC 61850, Industry 4.0 (including IoT and CPS), and SPL. TRILATERAL is a Model Based Engineering (MBE) tool using the IEC 61850 standard, SPL paradigm and DSL. Thanks to our solution, the user can graphically configure IoT communication protocols (HTTP-REST, WS-SOAP and CoAP) and the tool automatically generates the source code according to the configuration. The code allows to transmit data between the *cyber* and the *physics* part in order to monitor/control a ICPS.

The rest of this paper is organized as follows. The next section introduces the problem while Section 3 summarizes the technical background explaining the IEC 61850 standard. Section 4 presents the used IoT protocols and TRILATERAL for generating code based on the model, followed by three industrial use cases where TRILATERAL is used. Section 5 describes the lessons learned during the development of the tool and deployment of the ICPS. Next, we present the related work to finally conclude the paper with some conclusions and future lines for continuing the work.

2 USE CASE ANALYSIS

Thanks to the proximity of our institution to the industrial reality, it is possible to know the necessities of different industrial domains. Thus, considering the domain analysis fulfilled in the previous work (Iglesias et al., 2017), we realize that different domains have common needs (i.e., monitor a ICPS to capture information with the final goal of being aware of the state of the industrial domain) due to the important role that Industry 4.0 is playing. Although a common monitoring solution was a real need, when analyzing other industrial domains, i.e., Smart Elevators (Section 2.2), Wind Farms (Section 2.1) and Catenary-free Trams (Section 2.3), we realized that variability in IoT communications exists, i.e., how to communicate the devices (physical systems) with the software platform that monitors/controls the devices (cyber system), which was not considered in our previous work.

We noticed that depending on the industrial needs, the IoT communications are different, e.g., CoAP is more lightweight but for large payloads the overhead can be bigger than HTTP-REST (Iglesias-Urkia et al., 2018). Additionally, we realize that for some domains using more than one communication protocol can be beneficial (e.g. Catenary-free Tram, Section 2.3). That is why a new challenge was found after analyzing these three domains, being entirely applicable to other domains such as automated warehouses or press machines.

Considering that the SPL paradigm shows potential to improve productivity and reduce costs when variabilities and commonalities exist (Capilla et al., 2014; Iglesias et al., 2017), we consider important and necessary to use this paradigm to provide a solution to our challenge. Therefore, a SPL was designed and developed to give the user the opportunity to configure different IoT communication protocols. Likewise, with the purpose of improving this configuration, a DSL was created, because it provides simplicity as well as flexibility when making the necessary configurations (Fowler, 2011; Hussain et al., 2018).

Taking into account the need to use different IoT communication protocols to communicate the physical world with the cyber world, we analyzed an international standard, IEC 61850. This standard has also showed being valuable for other industrial domains outside electrical substations, such as Press Machines or Automated Warehouses (Iglesias et al., 2017).

2.1 Wind Farm

In a Wind Farm, wind turbines generate energy using wind. These need to be monitored, to know in real time how much energy they are generating. As they are critical infrastructures that can create huge problems if they fail, different parameters have to be monitored to ensure safety. To this end, it is necessary to represent the particular features in every wind turbine that exist in a specific wind farm. Thus, variability exist and generating code for every use case is time consuming.

In addition, Wind Farms are usually located in remote locations, where wind is stronger or more constant. They are located far from cities, on top of mountains or offshore, places where connectivity can be limited. The environment is not mobile, but to be able to connect to the Internet, heavyweight protocols can be a handicap.

2.2 Smart Elevator

In a big building there are several elevators, and a system can be created to control and monitor the elevators, and make a more efficient use of them. Currently, some elevators are equipped with batteries. The energy generated when the elevator is moving down is stored for its later use to improve the energetic efficiency of the building (stored energy is released when energy cost is more expensive). In this case, if we are able to monitor and control the batteries in each elevator, we will be able to make decision, i.e., it allows us to remotely monitor the state of the different energy storage resources and to activate them when necessary.

Note that elevators are located inside buildings, therefore, a controlled environment. This controlled environments do not usually have connectivity or energy supply issues, thus, the constraints on the communication protocols are not as severe as in other scenarios.

2.3 Catenary-free Tram

Any transportation system has different parameters to monitor its state, from critical parameters such as speed, direction or maintenance related information (e.g., state of the power source, break wear, etc.) to non critical systems such as information or multimedia features, or climate systems. Hence, an efficient energy system needs to be developed in order to charge trams via induction on stops and to control the amount of energy needed to reach the next stop (controlling the climate system to optimize the amount to charge the tram in each stop). Thus, we can operate autonomously, without human intervention.

In this scenario, two different use cases need to be taken into account. On the one hand, when the tram is ongoing, the environment is mobile and can have power or connectivity limitations. On the other hand, the tram can bulk much more information when it is on a station or a stop, while charging or not, where the aforementioned constraints are not applicable. However, since the Tram has to follow a schedule, the time for bulking the data is limited, so the data transfer time must be taken into account.

3 TECHNOLOGICAL BACKGROUND

To model the intelligent electronic devices at electrical substations, IEC 61850 makes use of two building classes, i.e., Basic Information Model and Control Blocks (CB) for additional functions. The Basic Information Model defines the elements of the real world and defines their information with a simple structure:

- **Server:** exposes systems to the outside and includes one or more Logical Devices (*LDs*).
- **Logical Device:** virtual representation of a real device, composed of one or more Logical Nodes (*LNs*).
- **Logical Node:** virtual abstraction application functionalities. All *LDs* have a **Logical Node Zero (LLN0)** to represent common data for the *LD*.
- **Data:** physical world information, associated to a *LN*.
- **DataAttribute:** information piece of a *Data*, e.g., value, timestamp. The values of a *DataAttribute* are defined by a type (e.g., Float, boolean).
- **Dataset:** group of existing *Data* and *DataAttributes* of a *LD*.

The CBs are specialized classes to interact with the information model through some additional functionalities:

- **Reporting:** *Buffered Report Control Blocks (BRCP)* and *Unbuffered Report Control Blocks (URCB)* define the generation of reports, the former ensures that the reports arrive to the destination while the latter works on a best effort basis.
- **Logging:** *Log Control Block (LCB)* configures the creation of logs from Datasets, what to log and under which circumstances.
- **Configuration:** *Setting Group Control Block (SGCB)* groups settings and allows to change between the defined groups.
- **Eventing:** *Generic Object Oriented Substation Event (GOOSE)* and *Generic Substation State Event (GSSE)* are respectively managed by *GOOSE Control Block (GoCB)* and *GSSE Control Block (GsCB)* to deliver Datasets containing DataAttributes and basic state change information. The events are based on publish-subscribe communications.
- **Sampled Values:** manage the transfer of sampled information in Datasets of DataAttributes in

a time controlled way. It can be implemented in two ways: using multicast communication with a *Multicast Sample Value Control block (MSVCB)* or unicast communication with an *Unicast Sample Value Control Block (USCVB)*.

Figure 1 defines the different elements, both on the Basic Information Model and the CBs. All the elements have a name and an absolute reference to uniquely identify them throughout the entire model.

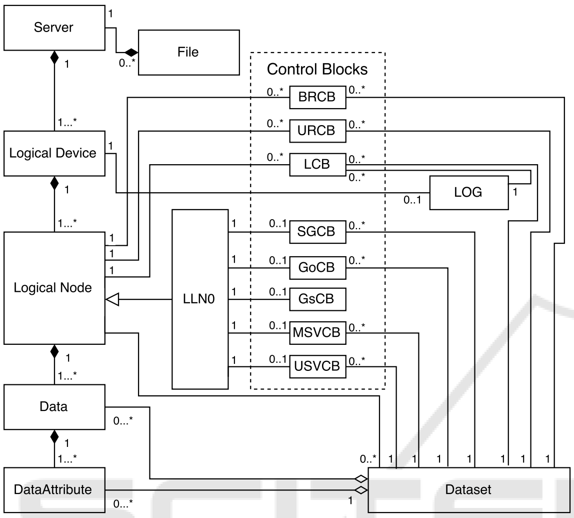

Figure 1: The information model of the IEC 61850 standard.

4 CONFIGURING IoT COMMUNICATIONS

The need to monitor/control each ICPS makes IoT communications essential. IoT communications can communicate the *Cyber* part with the *Physical* part. Each protocol has different characteristics, so their use depends on the use case. Thus, we have developed a MBE tool (TRILATERAL) using the SPL paradigm and DSL to make it easier for the user to graphically configure the IoT communication protocols in order to monitor/control the ICPS. TRILATERAL is based on the IEC 61850 standard.

4.1 IoT Communication Protocols

There are several communication protocols. For this work, we used WS-SOAP, HTTP-REST and CoAP. However, other protocols might be included in the tool in the future, if the need for them arises, e.g., MQTT, DDS or AMQP (Iglesias-Urkia et al., 2017a).

WS-SOAP uses HTTP to communicate, sending XML files using POST requests. In HTTP-REST, JSON resource representation and different request methods are used, such as GET, PUT, POST and DELETE. Finally, CoAP (Iglesias-Urkia et al., 2017b),(Iglesias-Urkia et al., 2018) is a lightweight protocol with a smaller header than the rest and also uses GET, PUT, POST and DELETE methods and pub-sub communication with the observe extension (Iglesias-Urkia et al., 2018). For more extended description of the protocols and a comparison in their performance using concrete examples with the IEC 61850 standard see (Iglesias-Urkia et al., 2018) and (Iglesias-Urkia et al., 2018).

WS-SOAP and HTTP-REST were not designed for resource constrained devices. However, they have historically been used on machine-to-machine communications, the precursor of IoT, so although they are not technically IoT communication protocols, in this work, we also refer to them as IoT communication protocols.

4.2 TRILATERAL: Overview

Figure 2 presents the different parts to configure an ICPS to monitor/control. TRILATERAL is a MBE practice based on the SPL paradigm to generate a specific software for each ICPS and uses a DSL for the selection and configuration of the generated software. TRILATERAL is divided into two parts: 1) the *server model* definition and 2) the *data model* definition. Both of them are configured by the user using the DSL (e.g., Figure 4). The former one, i.e., *server model* definition, is used for configuring the IoT communications for data transition. The latter one, is for configuring the *data model* based on IEC 61850 with the main objective to represent a ICPS.

Thanks to the created DSL, configuring the server in addition to the *data model* is faster and less error-prone. TRILATERAL is used to configure the model graphically. In addition, it is not only valid for a specific use case, but thanks to the SPL and the designed DSL, several industrial domains such as Wind Turbine Farms, Smart Elevators, and Catenary-free Trams can be configured.

First, the user configures the *server model* by choosing the IoT communication protocol (step 1). They can configure more than one IoT communication protocol if required. Hence, the user must choose the protocol that best suits their ICPS. Currently, TRILATERAL provides three IoT communication protocols, i.e., WS-SOAP, HTTP-REST and CoAP, with the latter's implementation explained in (Iglesias-Urkia et al., 2018). Thus, the user can configure the server according to the stakeholder's needs. Figure 3 shows the feature model where the combina-

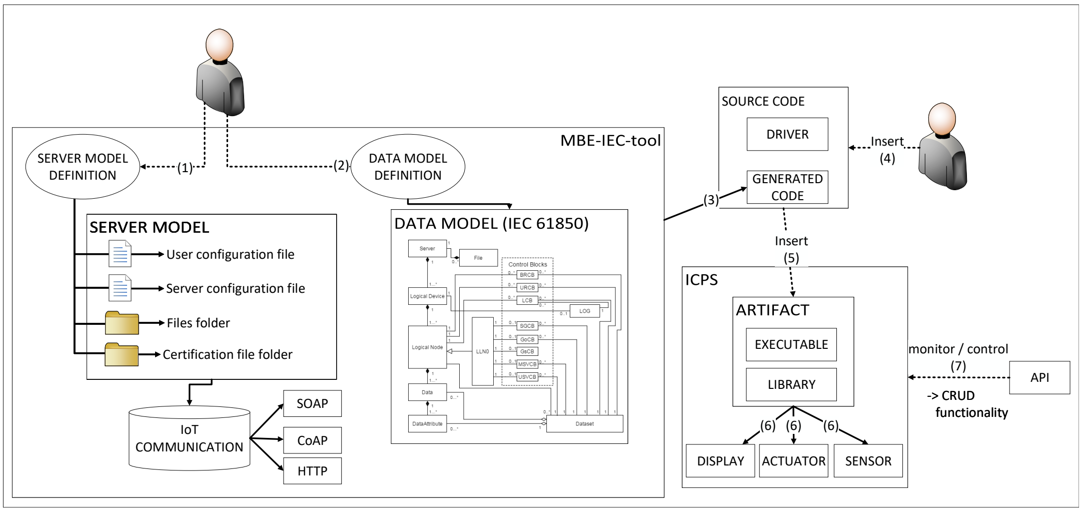

Figure 2: TRILATERAL overview.

tions that the user can choose to configure the *server model*.

Once the IoT communication protocol is chosen and the user has selected all the configurations it needs, TRILATERAL automatically creates user and server configuration files where the user will enter the necessary information for the system to be adapted. Then, TRILATERAL automatically generates two directories: one for certificates, where all security certificates are stored; and another one where the files related to the file management functionalities offered by the IEC 61850 standard are uploaded or downloaded. This is possible thanks to the DSL and the interaction with the user. Thus, the *server model* is automatically created considering stakeholder needs and avoiding configuration errors.

Afterward, the user configures the *data model* based on IEC 61850 (step 2). Figure 1 shows the detailed information model of the IEC 61850. To do so, the user configures the *Servers*, *LDs*, *LNs*, *Datas*, *DataAttributes* and the *Datasets* with TRILATERAL. Therefore, once the user configures the *server model* and the *data model*, TRILATERAL automatically generates the code (*generated code*) using model to text (M2T) transformation (step 3). The *generated code* is specific for an ICPS. Then (step 4), the user compiles the *generated code* with a specific driver (the connector that links the devices with the *data model*). The driver is different in each case, it depends on the hardware of the ICPS. Hence, it needs to be developed separately in order to create the specific communication between the *generated code* and the hardware. Once the driver is compiled with the *generated code*, some libraries and executables are created, i.e., *artifact*.

The *artifact* is introduced in the ICPS (step 5) and this is the responsible for monitoring/controlling all the devices inside the ICPS, i.e., *displays*, *actuators*, and *sensors* (Iglesias et al., 2017) (step 6). But to do so, an external *API* is necessary to interact with the ICPS, i.e., all the data needs to be controlled by the stakeholder somehow. That is how the stakeholder is able to interact with the ICPS remotely, using the *API* by the selected IoT communication protocol, i.e., WS-SOAP, HTTP-REST, CoAP.

An IEC 61850 compliant external *API* is able to do Create, Read, Update and Delete (CRUD) operations on the ICPS. To do so, the *API* uses predefined functions such as *DeleteDataSet* or *GetAllDataValues*. These functionalities are specific to the IEC 61850 standard and they are explained in more detail in (Iglesias-Urkia et al., 2017c).

Thus, thanks to TRILATERAL, an IoT communication middleware is created in order to monitor/control the entire ICPS. Although there is an interaction with the user to achieve the objective, it is important to note that if the stakeholder has its requirements and knows the real structure of its industrial domain, the generation of the *generated code* through TRILATERAL is faster and less error-prone. In addition, despite the standard being created for electrical substations, it is also suitable for other industrial domains (Iglesias et al., 2017; Iglesias-Urkia et al., 2017c).

4.3 TRILATERAL: Technical Detail

TRILATERAL is a graphical eclipse plugin developed with the Eclipse Modeling Framework[1]. The *data model*, i.e., the tree structure of the ICPS to be

[1]https://www.eclipse.org/modeling/emf/

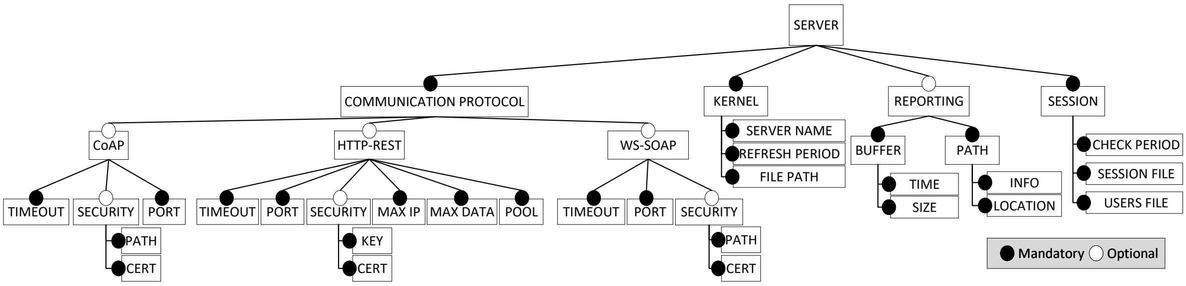

Figure 3: TRILATERAL server feature model.

monitored/controlled, can be represented with TRILATERAL. Figure 4 shows a screenshot of TRILATERAL, where a *Server* named *SERVER_NODE* can be seen. The *Server* has the *SmartElevator_LD LD*, which includes several *LNs*. Those *LNs* then have different *Datas*, *Datasets* and *CBs*.

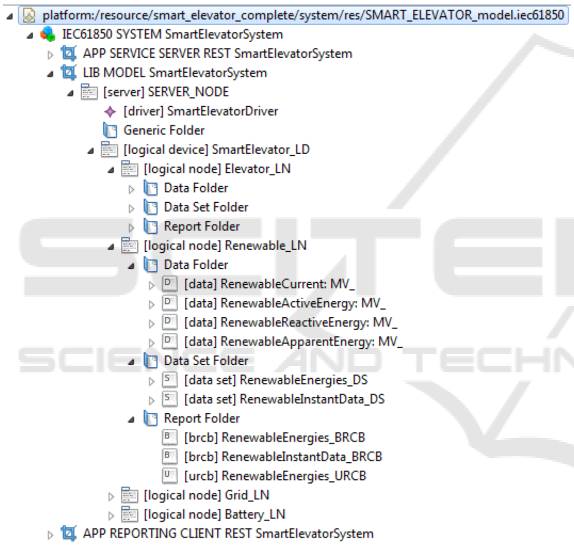

Figure 4: Screenshot of TRILATERAL.

As previously mentioned (Section 4.2), the code is generated automatically from the designed *data model* with TRILATERAL. Additionally, the user can choose one or more protocol between WS-SOAP, HTTP-REST or CoAP. The code which is generated composes several Eclipse projects, in three different layers, i.e., kernel, libraries and applications. The layers of projects and libraries are shown in Figure 5. The kernel of the ICPS is in the lower level. It includes the lib-model-kernel for the generic parts of the model along with the lib-model-*specific-model*, which is the model generated with TRILATERAL. On top of that (i.e., libraries layer), lib-service-server-rest, lib-service-server-soap and lib-service-server-coap are the libraries to create the servers for HTTP-REST, WS-SOAP and CoAP respectively. Other auxiliary libraries are also in this layer, i.e., libcoap for CoAP communication, libcbor for CBOR information representation, jsoncpp for JSON representation, microhttpd for HTTP-REST communication, and gSOAP for using WS-SOAP. The server for each protocol is on the top layer, i.e., application layer.

Even though the reporting servers are generated apart from the general server, they also have the same three layers. WS-SOAP and HTTP-REST need separate servers and clients because the communication paradigm changes. This does not happen with CoAP due to its *Observe extension* for Pub/Sub communication. Each of HTTP-REST and WS-SOAP reporting server and clients are generated from its own library, namely lib-reporting-server-rest, lib-reporting-client-rest, lib-reporting-server-soap and lib-reporting-client-soap. For reporting functions both WS-SOAP and HTTP-REST work in a similar way, where the service server stores the reports on a folder defined in the system's configuration file. The reporting client has to periodically check if there is any report on that folder, and if so, it sends it to the reporting server. CoAP implementation includes the *Observe* extension to allow Pub/Sub communications, thus, the client just subscribes to the reports and the server pushes them to the client when generated.

5 DEPLOYMENT AND LESSONS LEARNED

TRILATERAL was successfully used on three different industrial domains and thanks to the development and deployment we learned several lessons.

5.1 Deployment

For each of the three presented use cases, TRILATERAL was used to generate first the model and then the source code that allows to construct the set of artifacts that are deployed on the ICPS. These artifacts compose and provide the core functionality that al-

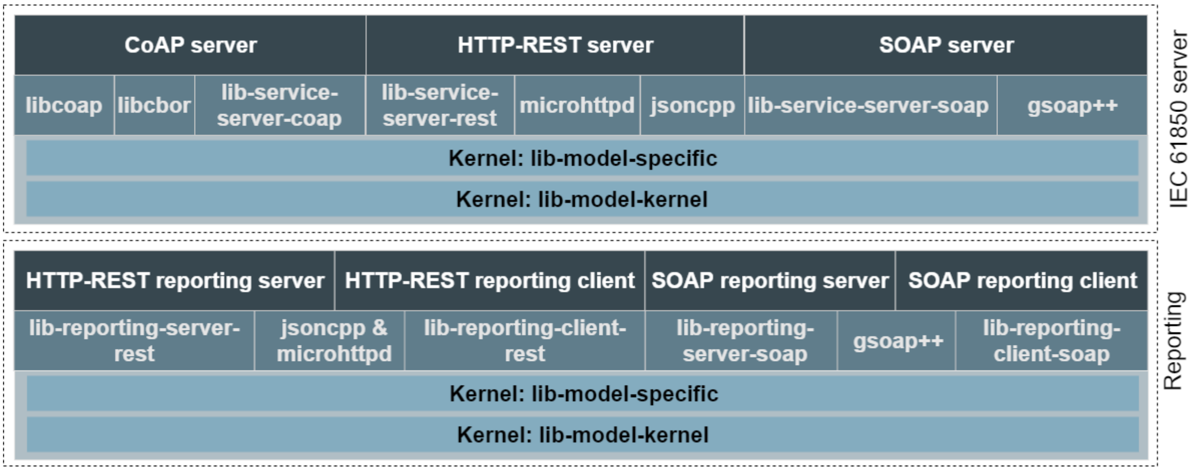

Figure 5: Layers of the IEC tool.

lows to remotely monitor/control IEC 61850 compliant devices using various IoT communication protocols.

The first deployment was the Smart Elevator, where a controlled environment had to be modelled and there are no connectivity or power issues, hence, WS-SOAP communication performs satisfactorily. In this scenario, we used one *LD* for each elevator. Then, we deployed the second use case, i.e., Catenary-free Tram. For this, two different use cases were defined: 1) when the tram is on route, or 2) when it is on a station. In addition, there were three *LDs*, one to represent the batteries and electrical components, a second one to represent the active demand management system and finally, a last one to represent the climate system of the tram. An issue arouse on the deployment process, i.e., the system was too heavyweight. This led to a change in TRILATERAL, making the kernel more modularized and providing the selection of what modules of the kernel should be included in each part of the system, thus, decreasing the weight of each part of the system. Finally, it became clear that this was a good change in the implementation of the Wind Farm, where we used one *LD* for each wind turbine. With this change, the first use case, i.e., Smart Elevator, was easily rebuild in order to update the system and the new build was more lightweight. In the Table 1, the Wind Farm, Smart Elevator and Catenary-free Tram scenarios are connected to the used protocols, describing the characteristics of the scenarios and why they fit with each communication protocol.

In summary, TRILATERAL has allowed IK4-Ikerlan to improve the engineering process of ICPS development, not only reducing time and costs but also improving the validation and maintenance tasks. It has also allowed to open new opportunities to extend TRILATERAL to other IoT domains such as Automated Warehouses or Press Machines.

5.2 Lessons Learned

TRILATERAL was deployed and tested in different industrial use cases (Section 2). These three use cases used TRILATERAL with different configurations, resulting on diverse generated code. Based on the development and deployment of these artifacts the following lessons have been learned:

5.2.1 Decrease Development and Deployment Time

The time to develop and deploy the *Artifact* has radically decreased from previous similar projects. This is because of the improvement of the overall ICPS development process, removing tasks that where being done manually and automating source code generation. This has led to a reduction of time and costs of development as well as a improvement of the quality of the delivered artifacts. This has also reduced engineering and maintenance costs of the overall process.

5.2.2 Useful in Other Domains

The use cases are mainly related to energy IoT systems. The variability of these systems is so high that they represent characteristics of very diverse IoT systems (device-to-cloud systems and device-to-device systems). Therefore, we believe that the proposed solution can be used in any IoT environment (manufacturing, transport, etc.) that needs to be monitored/controlled remotely.

Table 1: IoT communication protocols for each use case characteristics.

Protocol	Use case		
	Wind Farm	Smart Elevator	Catenary-free Tram
WS-SOAP		- Controlled environment - No connectivity/power issues	
HTTP-REST	- Remote location - Connectivity issues		Station - Controlled environment - Smaller header with big payloads
CoAP			On route - Mobile - Connectivity/Power issues

5.2.3 IEC 61850 Outside the Electrical Substations

The validation of the system has also been improved due to the fact that the same IEC 61850 kernel code is used on all scenarios. In this way, a new scenario need not be extensively evaluated because its core has already been validated and verified before with each model's development.

5.2.4 SPL and DSL Benefits Industry 4.0

Thanks to this work, we realized how beneficial the SPL paradigm can be in industrial domains. Because even though many domains exist they share similar requirements, as explained in Section 2, and many commonalities exist between them. In addition we also noticed the benefits a DSL can achieve. Although its development was complex, once it was well designed and developed, the configuration of an IoT communication protocol becomes much simpler, mainly due to the use of a visual environment. That is why thanks to the union of SPL and DSL, as mentioned in the first lesson learned, the development time decreases considerably. We have therefore learned the significance of using SPL and DSL in industry, considering both beneficial.

6 RELATED WORK

Reducing development time in addition to maintenance cost is possible by using reuse techniques such as SPL and MBE. These two paradigms are becoming increasingly common in industry (Capilla et al., 2014; Young et al., 2017). Some research has previously been done in terms of ICPS, SPL and MBE (Iglesias et al., 2017; Tang et al., 2018; Sinnhofer et al., 2017; Ayala et al., 2015) where the variability between the different devices inside the ICPS is managed. Several authors also use DSL for managing variability in ICPS (Sinnhofer et al., 2017).

Regarding the IEC 61850, Iglesias et al. (Iglesias et al., 2017) use the combination of two standards (IEC 61850 + IEC 62264) to capture all monitored data from the ICPS by using SPL and MBE. Regarding the communication protocols used along the IEC 61850, (Iglesias-Urkia et al., 2017c) and (Iglesias-Urkia et al., 2018) review previous related work. Apart from the MMS communication protocol already included in the IEC 61850 standard documents, the related IEC 61400 also includes SOAP Web-Services (IEC TC-88, 2016). Some authors have proposed other protocols, such as CORBA (Sanz et al., 2001), DDS (Calvo et al., 2012; Bi et al., 2013), or a combination of both (Calvo et al., 2009). A RESTful approach has been considered (Pedersen et al., 2010; Parra, 2016) using HTTP and also a Pub/Sub communication through XMPP (Hussain et al., 2018). Following this evolution, the first proposals to use a specific IoT communication protocol use CoAP (Shin et al., 2017; Iglesias-Urkia et al., 2017c).

To the best of our knowledge, this is the first work where the IEC 61850 is modeled by a DSL for generating an *Artifact* based on SPL and MBE, capable of monitoring/controlling ICPS. Besides, we also give the chance to the user to configure the IoT communication protocol (e.g., HTTP-REST, WS-SOAP and CoAP).

7 CONCLUSION

This paper presents TRILATERAL, a MBE SPL solution using IEC 61850 for configuring IoT communication protocols graphically. Thanks to the definition of a DSL which is integrated in the solution we are able to represent the data model of any ICPS. Thus, IoT communication protocols are automatically developed considering stakeholders' needs. In addition, TRILATERAL was evaluated in different industrial domains i.e., Wind Farm, Smart Elevator and Catenary-free Tram, where some lesson have been learned. Even more, we think it may be applicable to other domains.

Some future lines are expected to be addressed in the short-medium term:

1. Use TRILATERAL in other domains, such as Automated Warehouses or Press Machine domains, that could allow to validate its suitability for any kind of ICPS.
2. Enhance TRILATERAL from a SPL to a dynamic SPL (DSPL), regarding physical element changes (update, add or remove physical nodes). Currently, the model is static and can only be changed by means of an update.
3. Add new functionality to the TRILATERAL that could allow the artifacts to be remotely updated. This way, the ICPS could easily adapt to the surrounding context and to changes on the *data model* of the ICPS and related hardware.
4. Automatically generate *drivers* that link the *data model* with the sensors, displays, and actuators located on the devices. Currently, this is a manual task and requires many resources. This task is also very dependant of the target hardware where the *artifact* is going to be deployed.

ACKNOWLEDGEMENTS

This work has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under the MegaM@Rt2 project (Grant agreement No. 737494) in the EU Horizon 2020 program and the Basque Government through the Elkartek program under the TEKINTZE project (Grant agreement No. KK-2018/00104). We thank Shaukat Ali of Simula Research Laboratory (Oslo, Norway) for comments that greatly improved the manuscript.

REFERENCES

Ayala, I., Amor, M., Fuentes, L., and Troya, J. M. (2015). A software product line process to develop agents for the iot. *Sensors*, 15(7):15640–15660.

Bi, Y., Jiang, L., Wang, X. J., and Cui, L. Z. (2013). Mapping of IEC 61850 to Data Distribute Service for digital substation communication. In *IEEE Power and Energy Society General Meeting*, pages 1–5.

Calvo, I., García De Albéniz, O., Noguero, A., and Pérez, F. (2009). Towards a modular and scalable design for the communications of electrical protection relays. *IECON Proceedings (Industrial Electronics Conference)*, pages 2511–2516.

Calvo, I., Garcia de Albéniz, O., and Pérez, F. (2012). A communication backbone for Substation Automation Systems based on the OMG DDS standard. In *Przeglad Elektrotechniczny*, volume 88, pages 146–150.

Capilla, R., Bosch, J., Trinidad, P., Ruiz-Cortés, A., and Hinchey, M. (2014). An overview of dynamic software product line architectures and techniques: Observations from research and industry. *Journal of Systems and Software*, 91:3 – 23.

Fowler, M. (2011). *Domain-Specific Languages*. The Addison-Wesley signature series. Addison-Wesley.

Hussain, S. M. S., Aftab, M. A., and Ali, I. (2018). Iec 61850 modeling of dstatcom and xmpp communication for reactive power management in microgrids. *IEEE Systems Journal*, pages 1–11.

IEC TC-88 (2016). Wind energy generation systems - part 25-4: Communications for monitoring and control of wind power plants - mapping to communication profile.

Iglesias, A., Arellano, C., Yue, T., Ali, S., and Sagardui, G. (2018). Model- based personalized visualization system for monitoring evolving industrial cyber-physical system. In *Accepted for Publishing in 25th Asia-Pacific Software Engineering Conference, APSEC 2018*.

Iglesias, A., Lu, H., Arellano, C., Yue, T., Ali, S., and Sagardui, G. (2017). Product line engineering of monitoring functionality in industrial cyber-physical systems: A domain analysis. In *Proceedings of the 21st International Systems and Software Product Line Conference, SPLC 2017, Volume A*, pages 195–204.

Iglesias-Urkia, M., Casado-Mansilla, D., Mayer, S., and Urbieta, A. (2018). Enhanced publish/subscribe in coap: describing advanced subscription mechanisms for the observe extension. In *Proceedings of the 8th International Conference on the Internet of Things, IOT 2018*, pages 14:1–14:8.

Iglesias-Urkia, M., Casado-Mansilla, D., Mayer, S., and Urbieta, A. (2018). Integrating electrical substations within the iot using iec 61850, coap and cbor. *Submitted to IEEE Internet of Things Journal*.

Iglesias-Urkia, M., Casado-Mansilla, D., Mayer, S., and Urbieta, A. (2018). Validation of a coap to IEC 61850 mapping and benchmarking vs HTTP-REST and WS-SOAP. In *23rd IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2018*, pages 1015–1022.

Iglesias-Urkia, M., Orive, A., Barcelo, M., Moran, A., Bilbao, J., and Urbieta, A. (2017a). Towards a lightweight protocol for industry 4.0: An implementation based benchmark. In *Proceedings of the 2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics, ECMSM 2017*.

Iglesias-Urkia, M., Orive, A., and Urbieta, A. (2017b). Analysis of coap implementations for industrial internet of things: A survey. In *Procedia Computer Science*, volume 109, pages 188–195.

Iglesias-Urkia, M., Orive, A., Urbieta, A., and Casado-Mansilla, D. (2018). Analysis of coap implementations for industrial internet of things: a survey. *Jour-*

nal of Ambient Intelligence and Humanized Computing.

Iglesias-Urkia, M., Urbieta, A., Parra, J., and Casado-Mansilla, D. (2017c). IEC 61850 meets coap: towards the integration of smart grids and iot standards. In *Proceedings of the Seventh International Conference on the Internet of Things, IOT 2017*, pages 3:1–3:9.

Kagermann, H., Helbig, J., Hellinger, A., and Wahlster, W. (2013). *Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0: Securing the Future of German Manufacturing Industry; Final Report of the Industrie 4.0 Working Group*. Forschungsunion.

Kelly, S. and Tolvanen, J. (2008). *Domain-Specific Modeling - Enabling Full Code Generation*. Wiley.

Leitão, P., Colombo, A. W., and Karnouskos, S. (2016). Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges. *Computers in Industry*, 81:11–25.

Parra, J. (2016). *Restful Framework for Collaborative Internet of Things Based on IEC 61850*. PhD thesis, Universidad del País Vasco - Euskal Herriko Unibertsitatea (UPV/EHU).

Pedersen, A. B., Hauksson, E. B., Andersen, P. B., Poulsen, B., Træholt, C., and Gantenbein, D. (2010). Facilitating a Generic Communication Interface to Distributed Energy Resources: Mapping IEC 61850 to RESTful Services. *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, pages 61–66.

Sanz, R., Clavijo, J. A., Segarra, M. J., de Antonio, A., and Alonso, M. (2001). CORBA-Based Substation Automation Systems. In *Proceedings of IEEE Conference on Control Applications*.

Shin, I.-J., Song, B.-K., and Eom, D.-S. (2017). International Electronical Committee (IEC) 61850 Mapping with Constrained Application Protocol (CoAP) in Smart Grids Based European Telecommunications Standard Institute Machine-to-Machine (M2M) Environment. *Energies*, 10(3):393.

Sinnhofer, A. D., Pühringer, P., Oppermann, F. J., Potzmader, K., Orthacker, C., Steger, C., and Kreiner, C. (2017). Combining business process variability and software variability using traceable links. In *Business Modeling and Software Design - 7th International Symposium, BMSD 2017, Revised Selected Papers*, pages 67–86.

Suri, K., Cuccuru, A., Cadavid, J., Gerard, S., Gaaloul, W., and Tata, S. (2017). Model-based development of modular complex systems for accomplishing system integration for industry 4.0. In *Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development - MODELSWARD,*, pages 487–495. INSTICC, SciTePress.

Tang, H., Li, D., Wang, S., and Dong, Z. (2018). CASOA: an architecture for agent-based manufacturing system in the context of industry 4.0. *IEEE Access*, 6:12746–12754.

Tao, F., Zuo, Y., Xu, L. D., and Zhang, L. (2014). Iot-based intelligent perception and access of manufacturing resource toward cloud manufacturing. *IEEE Trans. Industrial Informatics*, 10(2):1547–1557.

TC-57, I. (2003). Communication networks and systems in substations - part 7-1: Basic communication structure for substation and feeder equipment - principles and models.

Young, B., Cheatwood, J., Peterson, T., Flores, R., and Clements, P. C. (2017). Product line engineering meets model based engineering in the defense and automotive industries. In *Proceedings of the 21st International Systems and Software Product Line Conference, SPLC 2017, Volume A*, pages 175–179.