

Architecture for automatically generating an efficient IEC 61850-based communications platform for the rapid prototyping of protection schemes

Steven Blair, Campbell Booth, Graeme Burt

Institute for Energy and Environment, University of Strathclyde, Glasgow, UK

steven.blair@eee.strath.ac.uk

Abstract

Although offering many benefits, using IEC 61850 in the course of research and development of new protection schemes can be costly and time-consuming. This paper describes an approach for automatically generating the low-level communications code required for an Intelligent Electronic Device (IED) to provide protection functionality, specifically Generic Substation Event (GSE) and Sampled Value (SV) messaging.

The platform is designed to be lightweight; only GSE and SV messages – crucial for protection – are implemented. The communications infrastructure will ensure that the required GSE and SV data are available to the subscribing Logical Nodes. Therefore, the system designer can focus on the design and implementation of the protection scheme, rather than the underlying communications infrastructure. The architecture leverages the Eclipse Modelling Framework to manage the complexity of the IEC 61850 standard; this is achieved by automating several aspects of the code generation process.

1 Introduction

For protection researchers, taking advantage of the benefits of IEC 61850 in the development of new protection schemes is desirable, and becoming increasingly necessary. Nevertheless, using IEC 61850 in the course of research and development activities can be costly and time-consuming. The process may require a time-consuming “ground-up” approach, the purchase of a relatively expensive – although fully-featured – software library and compatible hardware, or modification of an existing Intelligent Electronic Device (IED). This paper describes an approach for automatically generating the low-level communications code required for an IED to provide protection functionality, specifically Generic Substation Event (GSE) and Sampled Value (SV) messaging.

This paper provides relevant background on IEC 61850, and a high-level overview of the proposed architecture and its advantages; this is covered in Section 2 of the paper. The implementation is described in detail in Section 3. Section 4 describes applications of the existing architecture, further work which could extend its capabilities, and other types of applications – relating to IEC 61850 – which could benefit from the approach described in this paper.

2 Overview

The overall objective of the proposed approach is to automatically generate C code that implements a device – or several devices – specified in a Substation Configuration Description (SCD) file [1], as illustrated in Figure 1. An SCD is an XML document which defines all IEDs and their communications within a substation – or, in general, within any power system area defined by the SCD. The XML syntax used to create SCD documents is known as the Substation Configuration description Language (SCL) [1].

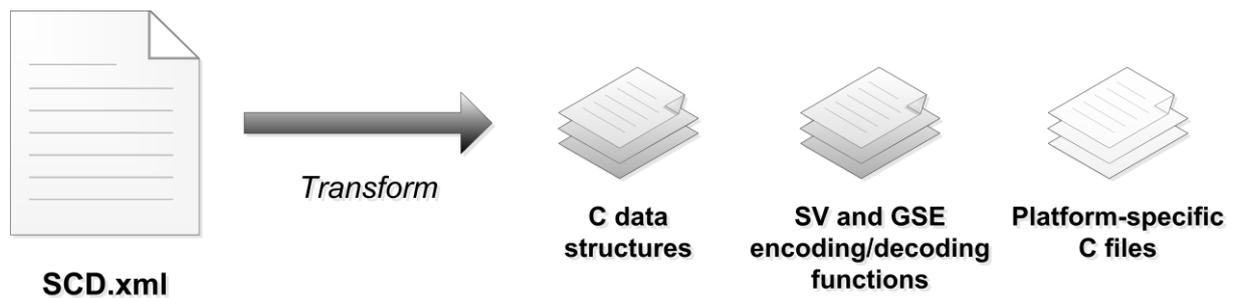


Figure 1: High-level objective

Parts 8-1 [2] and 9-2 [3] of IEC 61850 define the mapping of IED data to established communication protocols, for GSE and SV messages respectively. The motivation for this work is to create a “lightweight” implementation of IEC 61850-8-1 and -9-2 which offers a practical approach for the rapid prototyping of novel protection schemes and eliminates a significant engineering burden. Furthermore, by supporting the IEC 61850 standard, the software generated by this process can communicate with other IEC 61850 IEDs.

The architecture is generic and flexible: code can be generated for IEDs from any valid SCD file. The generated code defines the internal “database” of an IED, with wrapper functions to copy data to/from GSE and SV packets. Therefore, code can be more efficient than a runtime interpretation of the SCD because the IED does not need to maintain an internal model of the SCD. This “native-code” approach maximises runtime performance and facilitates the use of relatively low-cost embedded devices [4].

3 Implementation

As implied by Figure 1, the first step required by the architecture is to import an SCD file into a programming language environment. There are many tools, implemented in many programming languages, for parsing XML documents. However, most of these tools deal with XML in a generic manner without understanding the structure, rules and semantics that are specific to the SCL. The SCL model is defined by XML Schema in IEC 61850-6 [1]. Therefore, it is appropriate to use a tool that can import the XML Schema which describes the model, as well as importing an *instance* of the model (i.e., an SCD file). The Eclipse Modeling Framework (EMF), built on the Eclipse platform, is designed to assist with the development of software that is based on a structured model [5]. Its application to the SCL and, in particular, GSE/SV code generation, is described in Section 3.1.

3.1 Leveraging the Eclipse Modeling Framework

The EMF automatically generates a Java code representation of a model from, in this case, the XML Schema defined in IEC 61850-6. It also generates an XML parser for SCD files, which is tailored to this model. The output of the XML parser is a model instance, which can be queried in software and thereby transformed to a C implementation. This process is summarised in Figure 2.

The strength of this approach is that the class hierarchy is generated automatically, and is automatically populated with data from an SCD file. Although possible, it would be time-consuming to manually create the class hierarchy defined in IEC 61850-6 and write functions to instantiate the model. Furthermore, the EMF approach can more easily adapt to future changes to IEC 61850-6.

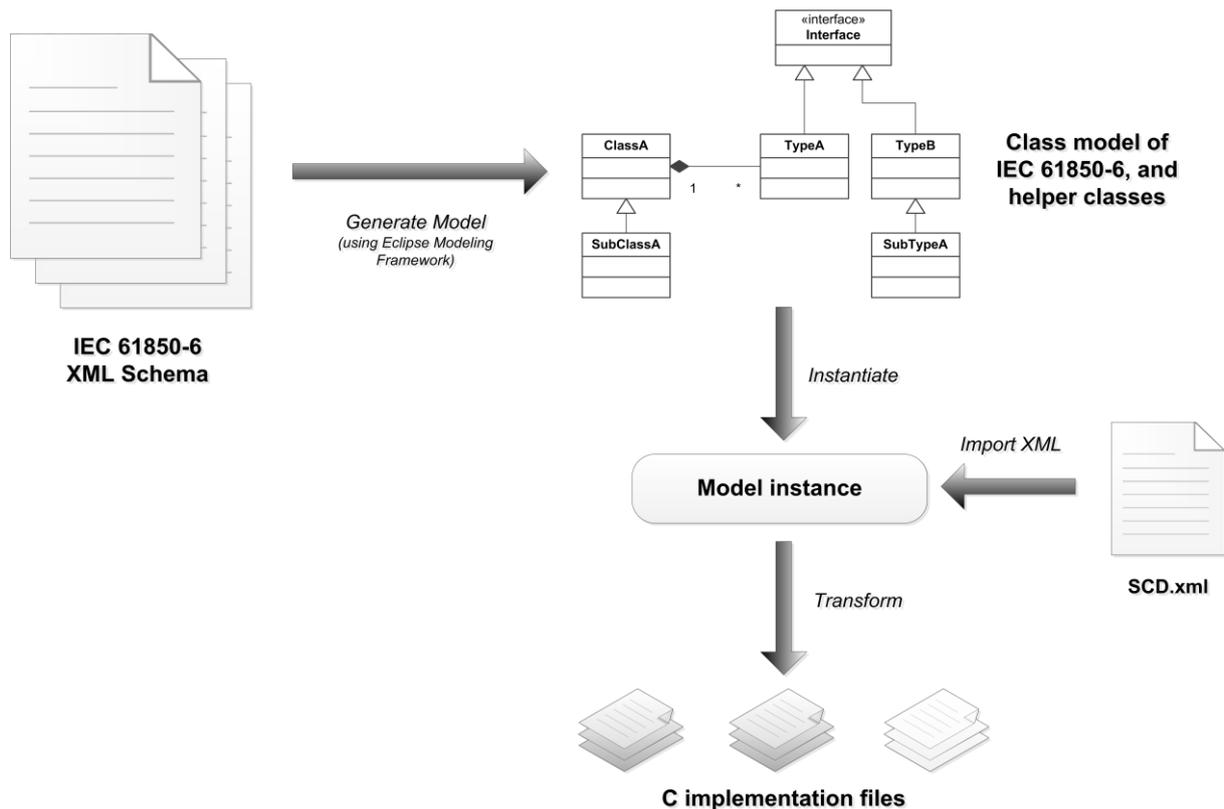


Figure 2: Use of the Eclipse Modeling Framework

3.2 Transforming the EMF model to a C implementation

An SCD file includes definitions of all types used within the file, such as Logical Node types, Data Object types and Data Attribute types. Each type can be mapped directly to a C data structure, resulting in a hierarchy of C data structures. An example of this mapping is given in Figure 3. A consistent naming convention is used to ensure that the C code will compile correctly.

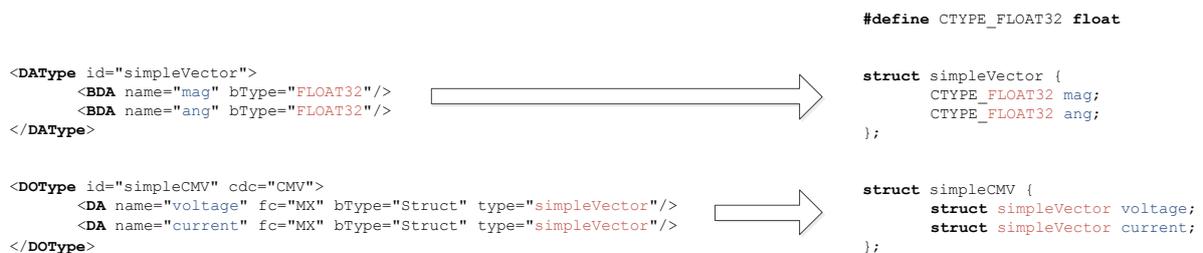


Figure 3: Example of SCD data type to C data type mapping

Each GSE and SV message has a dataset, which may consist of a number of Data Objects and Data Attributes. Therefore, each Data Object type and Data Attribute type must have corresponding functions for serialisation into a dataset, and de-serialisation from a dataset. GSE datasets are encoded using Basic Encoding Rules (BER) [6], but SV datasets use fixed-length fields, as defined in IEC 61850-9-2. Therefore, GSE and SV require separate (de-)serialisation functions; these are generated during the transformation. Additional wrapper functions are provided to, for example, encode an entire SV Ethernet frame to an array of bytes; the array of bytes may then be passed to the appropriate platform-dependent function to send an Ethernet frame. Furthermore, if specified in the SCD, multiple SV Application Service Data Units (ASDUs) are automatically grouped into a single Application Protocol Data Unit (ADPU).

The encoding/decoding of basic types, such as a 32-bit signed integer or a 32-bit floating point number, is platform-specific. Therefore, the generated C files must be accompanied by platform-

specific code to ensure consistency with IEC 61850 – regardless of the platform’s native data type definitions and endianness.

3.3 Case study

The main intended application of the existing architecture is the implementation of IEDs, for the rapid prototyping of protection schemes which require communications. The target hardware could be a standard PC or an embedded platform (e.g., for Process Bus applications). The generated C implementation has been tested on x86 PCs and an ARM-based microcontroller. It is compatible with both C and C++ compilers.

A simple example is illustrated in Figure 4. The microcontroller – an “mbed” [7], with a 96MHz ARM Cortex-M3 processor – transmits three-phase current sampled values, at 16 samples per cycle (at 50Hz fundamental frequency). For the purpose of demonstrating the communications, the instantaneous current values are generated using a sine function within the mbed. The PC receives the sample values, computes the Discrete Fourier Transform (DFT) for each phase, and periodically transmits a GOOSE message containing the vectors representing the fundamental magnitude and relative phase components of the current waveforms. The process can be monitored using a network protocol analyser, such as Wireshark [8].

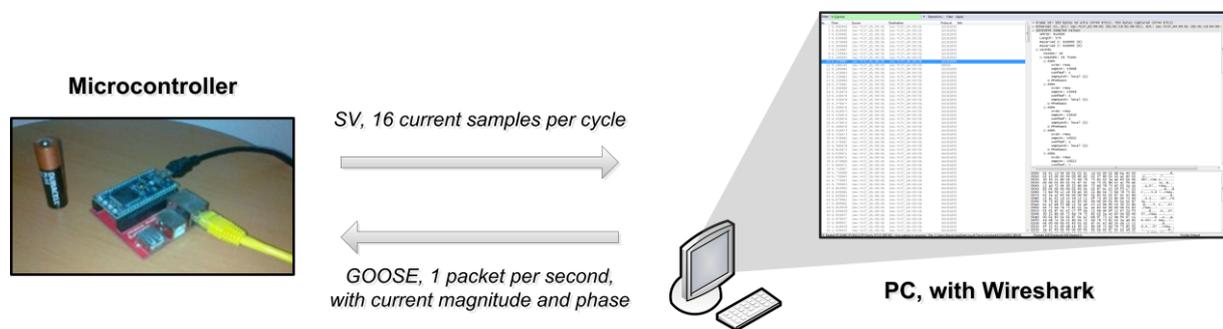


Figure 4: Simple example of functionality

Although basic, this example demonstrates the flexibility and convenience of the software described in this paper. Once the SCD is specified, the only manual code required is the DFT implementation (which is readily available), and some relatively simple platform-specific system calls to send and receive Ethernet packets. Note that SV transmission rates are possible at more typical rates of 4000 or 4800 samples per second and these rates can be achieved using a relatively inexpensive embedded platform.

4 Future applications

4.1 Applications in power system protection research

4.1.1 Multi-IED simulation

The architecture proposed in the paper can be used to generate code that implements a particular IED per hardware device, as demonstrated in Section 3.3, or multiple IEDs on one hardware device. The latter can be used to verify the operation of GSE and SV communications for an entire substation, providing early confidence in the validity of a system design.

4.1.2 Communications simulation

The generated C code could be used as a software entity – known as a “node” – which represents an IED within a communications simulator such as OMNeT++ [9]. Although OMNeT++ does not model a power system (unless OMNeT++ is embedded into a power system simulation), it could be used to monitor the communications between IEDs to, for example, assess various network topologies.

4.1.3 Power system simulation

The architecture could be linked with power system simulation tools in a number of ways. For example, the Simulink Real-Time Workshop (RTW) [10] can generate C/C++ code that implements a SimPowerSystems model. The RTW code could therefore provide valid measurement data for GSE and SV packets – allowing the demonstration of a fully-functional protection system prototype.

The generated code could also be used in real-time power system studies. As described in Section 3.3, a simple IEC 61850-compatible IED can be implemented on an embedded platform with analogue and digital inputs. These inputs could be power system parameters (e.g., voltages, currents, or circuit breaker status) supplied by a real-time simulator, such as the RTDS [11].

4.2 Extending the existing architecture

In future, the architecture could be extended to support other features of IEC 61850, such as: Reports for automation and control schemes, Settings Groups for adaptive protection applications, and Logging.

4.3 Other uses for EMF with IEC 61850

An EMF model-based approach has several other features that can be used to automate other tasks relating to IEC 61850 [12]:

4.3.1 Validation of an SCD file

The XML Schema defined in IEC 61850-6 is the normative reference for the SCL. EMF uses this XML Schema to create an SCL XML parser; EMF therefore already checks SCD files for basic syntactical conformance with the XML Schema. EMF also supports a framework for further SCD validation, such as: ensuring each Control Block has a matching Dataset; ensuring each Logical Node Input has a corresponding Dataset source; and ensuring no circular Sub-Data Object (SDO) references occur.

4.3.2 Substation visualisation and monitoring

The EMF could be used as a platform for visualising the substation electrical topology which can be described in an SCD file, along with the location of IEDs and their communications services. The “coordinates” syntax extension to the SCL facilitates such applications. Furthermore, the visualisation could be linked to real-time data from IEDs for substation monitoring. EMF has code generation tools for other programming languages, other than the C/C++ example used in this paper, and especially favours Java.

4.3.3 SCD editor

Not only can EMF import existing SCD files, it provides a framework for editing SCDs. A simple tree-based SCD editor can be generated automatically from the XML Schema; more advanced, graphical editors are also supported within the framework.

EMF can also be used for implementing changes to the SCL model itself, i.e., to the XML Schema, and automatically generating the resulting XSD files.

5 Conclusions

This paper has presented an approach for automatically generating an IEC 61850-compliant implementation of IEDs. The architecture is very flexible, yet generates lightweight code – suitable for embedded applications.

It has been demonstrated how model-based development tools, such as the Eclipse Modeling Framework, can provide a powerful platform for the development of tools for power system protection research and operation. This paper represents an example of this paradigm for IEC 61850-related development – an approach that takes advantage of central definition of a power system through the SCD.

Acknowledgements

This work was supported by the EPSRC and by Rolls-Royce Plc, and was carried out at the Rolls-Royce University Technology Centre at the University of Strathclyde, Glasgow, UK.

References

- [1] IEC TC 57, "Communication networks and systems for power utility automation - Part 6: Configuration description language for communication in electrical substations related to IEDs (IEC 61850-6:2009)," 2009.
- [2] IEC TC 57, "Communication networks and systems in substations Part 8-1: Specific Communication Service Mapping (SCSM) - Mappings to MMS (ISO 9506-1 and ISO 9506-2) and to ISO/IEC 8802-3 (IEC 61850-8-1:2004)," 2004.
- [3] IEC TC 57, "Communication networks and systems in substations Part 9-2: Specific Communication Service Mapping (SCSM) - Sampled values over ISO/IEC 8802-3 (IEC 61850-9-2:2004)," 2004.
- [4] J. Starck and S. A. Kunsman, "Pushing the limits," *ABB review*, 2010.
- [5] The Eclipse Foundation, "Eclipse Modeling - EMF - Home," 2011. [Online]. Available: <http://www.eclipse.org/modeling/emf/>
- [6] ITU, "ITU-T X.690, ISO/IEC 8825-1: ASN.1 encoding rules," 2002.
- [7] mbed, "Rapid Prototyping for Microcontrollers - mbed," 2011. [Online]. Available: <http://mbed.org/>
- [8] Wireshark Foundation, "Wireshark," 2011. [Online]. Available: <http://www.wireshark.org/>
- [9] OMNeT++ Community, "OMNeT++ Network Simulation Framework," 2011. [Online]. Available: <http://www.omnetpp.org/>
- [10] The MathWorks, Inc., "Real-Time Workshop - Generate C code from Simulink models and MATLAB code - Simulink," 2011. [Online]. Available: <http://www.mathworks.com/products/rtw/>
- [11] RTDS Technologies, "Real Time Power System Simulation - RTDS Technologies," 2011. [Online]. Available: <http://www.rtds.com>
- [12] T. Kostic, O. Preiss, and C. Frei, "Understanding and using the IEC 61850: a case for meta-modelling," *Computer Standards & Interfaces*, vol. 27, no. 6, pp. 679–695, Jun. 2005.