The STRONgrid library: A modular and extensible software library for IEEE C37.118.2 compliant synchrophasor data mediation

Maxime Baudette, Seyed Reza Firouzi, Luigi Vanfretti

Article history:
Received 12 January 2018
Received in revised form 4 August 2018
Accepted 13 August 2018

Keywords:
IEEE C37.118.2
PMU
PDC
Synchrophasors
WAMPAC

A B S T R A C T
The electric power grids expose highly dynamic behaviors that can be mitigated by exploiting Wide-Area Monitoring, Protection And Control (WAMPAC) technologies. These technologies rely on utilizing the synchrophasor measurements provided by the Phasor Measurement Units (PMUs). The IEEE C37.118.2 standard defines a method for real-time communication of synchrophasor data between PMUs, Phasor Data Concentrators (PDCs) and other applications. This paper describes the Smart Transmission Grid Operation and Control (STRONgrid) library, which serves as a real-time data mediator (i.e. interface) between the IEEE C37.118.2 protocol and applications consuming PMU measurements. The STRONgrid library packages all the necessary components to allow the researchers to focus on the development of synchrophasor applications in higher level programming environments (e.g. LabVIEW).

© 2018 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Motivation and significance

Electric power systems throughout the world have begun to experience unpredictable dynamic behaviors with the connection of intermittent generation units that use renewable energy sources. The rapid development of new tools that exploit Information and Communication Technologies (ICTs) give opportunities to address such new challenges [1].

Time-synchronized phasor (i.e. synchrophasor) measurements are subsecond level voltage and current measurements of the grid. These measurements are acquired by Phasor Measurement Units (PMUs) that are equipped to use a GPS clock. PMU measurements are the key building blocks used in Wide-Area Monitoring, Protection And Control (WAMPAC) systems as one of the building blocks for smart grids [2].

The potential of WAMPAC systems can be exploited by designing, implementing, and testing new synchrophasor-based software and hardware applications. In order to acquire the real-time synchrophasor data, such applications communicate with PMUs or Phasor Data Concentrators (PDCs). The communication protocol for synchrophasor measurement data exchange is standardized through the IEEE C37.118.2 standard [3].

In the IEEE C37.118.2 protocol, synchrophasor data transfer is handled by exchanging four message types: data, configuration, header, and command frames. The full details of the protocol can be found in the standard [3], or in a more concise interpretation in [4,5].

A sample IEEE C37.118.2 message exchange mechanism, which has also been implemented in the STRONgrid library, is presented
PMU/PDC
Server

IEEE C37.118.2
Client

Command Message: "Turn-off Transmission of Data Frames"

Command Message: "Send Header Frame"

Header Message

Command Message: "Send CFG-2 Frame"

CFG-2 Configuration Message

Command Message: "Turn-on Transmission of Data Frames"

Data Message

Data Message

Fig. 1. Sample IEEE C37.118.2 message exchange for initiating a connection to a PMU/PDC server and start data streaming.

in Fig. 1. Here the C37.118.2 client initially sends a Command message to stop any transmission of Data frames by the PMU/PDC server. Then by sending another Command message, the client requests the server for the descriptive information available in the Header message. Before starting to receive the Data messages containing real-time synchrophasor data, the client needs to receive the PMU/PDC configurations available in the Configuration Type-2 (CFG-2) message. Once the CFG-2 message received and parsed, the PMU/PDC server initiates the transmission of Data messages.

Most PMU applications are deployed in a monolithic software environment, often proprietary, which makes it difficult to quickly prototype new PMU applications. A different approach would be to develop applications that can execute modularly by dissociating the PMU data mediation functions from the PMU data based applications. This is achieved by building a real-time data mediator as an interface to a higher level programming language more accessible to researchers in the field of power systems so to help with rapid prototyping. Hence, researchers can focus on the development of synchrophasor monitoring and control applications on a personal computer (e.g. LabVIEW) or even embedded systems (e.g. National Instrument Compact RIO controller). While some have attempted several implementations of the real-time mediation component, such as Bablefish [6] and Khorjim [5], this paper focuses on the Smart Grid Synchrophasor Software Development Kit (S3DK) [7] that was developed as part of the Smart Transmission Grid Operation and Control (STRONgrid) project [8]. The S3DK is comprised of two parts: the Real-Time Data Mediator (RTDM) core library, and a LabVIEW user interface (VI) and functions library.

This paper presents the RTDM core that is a Dynamic Link Library (called STRONgrid DLL) created from a set of C++ source codes that implement the IEEE C37.118.2 standard, among other functionalities. Thus, the STRONgrid Library will handle the communication with PMUs/PDCs through the Transmission Control Protocol and the Internet Protocol (TCP/IP) on the Microsoft Windows OS platform. The library also features an Application Programming Interface (API) that can be accessed by the user and that has specialized methods that make it accessible to LabVIEW.

The LabVIEW user interface, is a set of LabVIEW blocks that provide a user interface to configure and initiate connections to PMUs/PDCs, as well as a set of functions to programmatically access the same functionalities and access the PMU data for further processing. The S3DK LabVIEW VI was developed to use the STRONgrid library through its API. More details on that LabVIEW user interface will be documented in a future publication. The open source software package is already available online at https://github.com/ALSETLab/S3DK.

2. Software description

The STRONgrid library is a real-time data mediator with additional functions that allows further processing of PMU measurements in custom WAMPAC applications. Thus, it was designed as an intermediary program that translates PMU measurements sent using the IEEE C37.118.2 protocol, and delivers the message through an API. The connection to the PMU/PDC server is established by providing the library with the following information, as for any regular PMU application: IP Address, Port Number, and IDCODE (or PMU ID).

Interactions with the library are carried out through the library’s API, by sending successive commands. In the STRONgrid project, the goal was to develop a solution, the S3DK, capable of delivering PMU measurements in the LabVIEW development environment. In this context, the API’s implementation was designed to include methods tailored for LabVIEW’s existing functionalities to call external libraries. These public methods, provide the necessary mapping and interface between the STRONgrid DLL’s and LabVIEW’s data-types, and thus are considered to be LabVIEW specific.

The existing methods of the API can also be used with other programming languages, as shown with the StronggridDLL-StressTest example. Additional methods can be implemented to extend it even further and provide other language specific methods. The library is, however, limited to the Microsoft Windows platform, as it relies on dependencies in the Microsoft Windows SDK and Microsoft Visual C++ libraries. These dependencies provide functionalities for string manipulation and network communication over TCP/IP.

The STRONgrid DLL and its interactions with other software components can be summarized as illustrated in Fig. 2.

2.1. Software architecture

The STRONgrid library source code is provided as a Microsoft Visual Studio project developed in C++. The code is structured in a modular architecture with four main blocks separating the main components of the library:

- StronggridBase
- StronggridClientBase
- StronggridDLL
- StronggridDLLStressTest

The StronggridBase block contains the code and methods dedicated to interacting with and parsing the IEEE C37.118.2 protocol. It also implements methods to encode messages according to the protocol’s specifications.

The StronggridClientBase block implements the IEEE C37.118.2 client functionality as an object of the PdcClient class. It is in this object that the private methods emulating a PDC client are implemented with the necessary logic to establish a connection to a remote PMU/PDC, and stream the real-time PMU data. These methods also extract the information to make it available through the API.

The StronggridDLL block implements the API that will be used by the client application to access the PMU/PDC data. Note that the API can easily be extended by implementing new public methods in this file.

Finally, the StronggridDLLStressTest block implements a simple client application to test the DLL compiled from the other
In the current version, all the functionalities to build a client application have been implemented, but the communication with the PMU/PDC server is limited to the TCP/IP protocol, and the library can handle the IEEE C37.118.2 protocol in its 2005 and 2011 versions. The implementation can, however, easily be extended to support additional protocols, such as the UDP/IP protocol for the remote connection, or the extension of the IEEE C37.118.2 protocol to a new revision. The library is also set up to add a PDC server functionality.

2.2. Software functionalities

The library includes all the necessary functionalities to enable a client application to stream PMU measurements in real-time. To achieve this, the three functionalities of the STRONgrid library can be summarized as follows:

- Configuration handling of the PMU/PDC connection,
- Parsing of the data from IEEE C37.118.2 protocol over TCP/IP,
- Real-Time PMU data publishing through the API.

The detailed functionalities of the library are documented in the API, where each method corresponds to an individual functionality. There are two kinds of methods, the methods that can be polled to retrieve all available information (e.g., list the available PMUs), and the methods that can be provided with a set of parameters to retrieve a specific information (e.g., measurements from a single PMU). A non-exhaustive list of the methods available in the API is presented in Listing 1.

In practice, the client application using the STRONgrid library should implement a workflow involving several of the available methods in the API. It is through this workflow that the three functionalities summarized above were implemented. As an example, a typical flowchart for a single thread–single PMU/PDC implementation is presented in Fig. 4.

The workflow is initiated by providing the connection information when calling the connectPdc method. This will create a PdcClient object and establish a socket connection using the credentials passed as arguments. On success, the API returns 1 and a pseudoPdcId, uniquely identifying the PDC.

The PdcClient will be identified through the pseudoPdcId, when calling the other methods in the process. The data streaming requires to first read the header frame, that is requested through the readHeaderData method. The DLL will then transmit a "Send Header Frame" Command message to request that information.
Fig. 5. Output file after executing the StrongridDLLStressTest executable.

Listing 1: Source code snippets showing some of the Stronggrid DLL API methods

```c
int connectPdc( char *ipAddress, int port, int32_t pdcId, int32_t* pseudoPdcId);
int disconnectPdc( int32_t pseudoPdcId);
int readHeaderData( int32_t timeoutMs, int32_t pseudoPdcId);
int readConfiguration( int32_t timeoutMs, int32_t pseudoPdcId);
int startDataStream( int32_t pseudoPdcId);
int stopDataStream( int32_t pseudoPdcId);
int readNextFrame( int32_t timeoutMs, int32_t pseudoPdcId);
int getPdcConfig( pdcConfiguration* pdcCfg, int32_t pseudoPdcId);
int getPmuConfiguration( pmuConfig* pmuconf, int32_t pseudoPdcId, int32_t psmlIndex);
int getPdcRealData( pdcDataFrame* rd, int32_t pseudoPdcId, int32_t psmlIndex);
int getPmuRealData( pmuDataFrame* rd, PmuStatus* status, int32_t pseudoPdcId, int32_t psmlIndex);
int getHeaderMsg( char* msg, int maxMsgLength, int32_t pseudoPdcId);
```

The application can then access the information by calling the `getHeaderMsg` method. Similarly, the PMU/PDC configurations are requested by calling the `readConfiguration` method and accessed by using `getPdcConfig` and `getPmuConfiguration` methods.

At this point the PMU/PDC is fully identified and the connection is established, data streaming can be initiated by calling the `startDataStream` method. A loop can then proceed to consume the real-time data, calling the `readNextFrame` method, to decode the next IEEE C37.118.2 data message, and the synchrophasor data are accessed using the `getPdcRealData` and `getPmuRealData` methods.

At any moment, the application can interrupt the data streaming with the `stopDataStream` method, which will send a “Turn-off Transmission of Data Frames” Command message to the remote PMU/PDC. At the end, the remote connection will be closed by calling the `disconnectPdc` method.

Note that the presented workflow includes only one remote PMU/PDC. More complex workflows can be built, and in the current implementation, the polling of the different PMU/PDC for new data is synchronous, through a synchronization layer.

3. Illustrative examples

As previously described, the STRONgrid library seeks to facilitate the utilization of real-time PMU data in client applications. Thus, the examples presented here are client applications that use the library.

3.1. StronggridDLLStressTest executable

The StronggridDLLStressTest is a simple program developed as part of the STRONgrid library as a testing suite. Upon compilation of the STRONgrid library Visual Studio Project, both the DLL and the StronggridDLLStressTest executable will be created. This program will test the API methods of the DLL and attempt to establish a connection to a remote PMU/PDC to stream synchrophasor data. The program actually implements a workflow very similar to that of Fig. 4. All the test results are written in a log file to help diagnose any problem.

Fig. 5 shows the log file produced when testing the executable interacting with a PDC in cascade with a PMU. In this test, the PDC was streaming out data from a single PMU, named Top. The Top PMU contained a single phasor named “VAYPM”. Data streaming was initiated during the testing process and the received data was written in the log file.
Fig. 5 shows that the IEEE C37.118.2 client application requested the Header information, the Header message was provided by the PDC server in line #2 after a failed attempt (in line #1). The PMU and Phasor Configuration data can be found in lines #6–19, and the Data message with the first Phasor value is printed in line #30.

3.2. S3DK LabVIEW interface

The S3DK LabVIEW Interface is a set of LabVIEW VIs (i.e. library) that integrate the STRONgrid DLL with the LabVIEW development environment. The library seeks to minimize the development time and effort related to working with real-time PMU measurements by implementing the necessary data managements and operations. It enables the user to interact with the DLL and access the real-time PMU data by providing simple blocks.

The main components of the library are a graphical user interface (GUI) and a set of functions that can be integrated into an existing program. The S3DK GUI allows the user to directly interact with the DLL, as shown in Fig. 6. There the user can configure the connection information, initiate the connection and start the data streaming. The set of functions can easily be integrated into more complex programs consuming PMU measurements to automate the connection process and access the real-time PMU data provided as LabVIEW data-types (see Fig. 7).

The end goal of the library is to facilitate the development of more advanced program such as other Wide-Area Monitoring System (WAMS) application using real-time PMU data. Fig. 8 shows such an application with a frequency display, a frequency spectrum analyzer and a “oscillation energy meter”, essential components of a tool for real-time sub-synchronous oscillations detection; see [9] for more details.

4. Impact

As described in Section 1, the challenges being faced by electrical power grids will require to develop new and innovative WAMPAC systems. However, there are many obstacles to the development, and even more so prototyping/testing, of such solutions as the access to real-time PMU measurements is locked in monolithic software/hardware, and because researchers in the field are not proficient with the software development skills nor familiar with the technical know-how to deal with real-time data over the IEEE C37.118.2 protocol.

The STRONgrid library seeks to address these difficulties by providing an open source real-time data mediator for the IEEE C37.118.2 protocol. The implementation of this real-time data mediator was carried out in a low-level programming language (C++) to provide a computationally lightweight solution as compared to a LabVIEW-only implementation that would have consumed more hardware resources, as seen in previous implementation attempts by the authors [10]. The library is fully accessible through an API that provides additional specialized methods tailored for interacting with the LabVIEW development environment. Together with the S3DK LabVIEW Interface, it offers an attractive solution to power systems researchers by letting them design and prototype PMU-based applications in a high-level programming language.

In the literature, there are many ideas for developing new WAMPAC applications, but the examples of working prototypes or technology-demonstrators are still limited (to a few vendors) to have an impact in the actual adoption of these new technologies by the transmission system operators. The project seeks to facilitate the development of such prototypes.

5. Conclusions

The STRONgrid library was developed to provide all functionalities of a real-time data mediator capable of interfacing with PMU/PDC equipments using the IEEE C37.118.2 standard protocol, and applications that consume real-time PMU measurements. The library was developed together with a LabVIEW interface, which together build the Smart grid Synchrophasor SDK (S3DK). S3DK was successfully tested through the development of several prototype PMU applications (see [8]).

The STRONgrid library implements all the necessary functionalities to use it as a PDC client, receiving and decoding data from the IEEE C37.118.2 protocol. The included API is fully functional to work with LabVIEW together with the rest of the S3DK package.

The code can be extended with additional methods in the API, to add the PDC server functionality, or even add the support for UDP data streaming.

Acknowledgments

The work of Luigi Vanfretti was primarily supported by the ERC Program of the National Science Foundation, United States and DOE under NSF Award Number EEC-1041877. Other US government and industrial sponsors of CURENT research are also gratefully acknowledged.

The support of the following funding bodies is gratefully acknowledged: Statnett SF, the Norwegian Transmission System Operator; The STRONgrid project by Nordic Energy Research.
Fig. 8. PMU application display example.

The technical contribution and support provided by the following participants is sincerely acknowledged: Vemund Aarstrand, Enfo Energy AS, Oslo, Norway; Martin Sveningsson, Prevas AB, Stockholm, Sweden; CodeSmiths AS, Oslo, Norway.

References


