STANDARDS-BASED OPEN-SOURCE PLC DIAGNOSTICS MONITORING

B. Copy, M. Zimny, H. Milcent
CERN, Geneva, Switzerland

Abstract

CERN employs a large number of Programmable Logic Controllers (PLCs) to implement industrial processes. These PLCs provide critical functions and must be placed under permanent monitoring. However, owing to their proprietary architecture, it is difficult to both monitor the status of these automates using vendor-provided software packages and integrate the resulting data with the CERN accelerator infrastructure, which itself relies on CERN-specific protocols and configuration facilities.

This paper exposes the architecture of a stand-alone "PLC diagnostics monitoring" Linux daemon, which provides live diagnostics information through standard means and protocols, namely file logging, CERN protocols, and Java Management Extensions. Such information is currently consumed by the EN-ICE MOON supervision software [1] used by the EN-ICE Standby Service to monitor the status of critical industrial applications used in the LHC and the CERN DIAMoN monitoring console [2] used by the LHC operators. Both applications are used daily to monitor and diagnose critical PLC hardware running all over CERN.

PLC DEVICES AND THE RELEVANCE OF INDUSTRIAL STANDARDS

PLCs are off-the-shelf industrial components designed to operate in a sheltered network environment. Moderate, predictable network traffic to and from the equipment and a small number of concurrent network-based accesses (in order to perform control with near real-time precision) are both essential parameters for ensuring reliable behavior on the part of the controller.

Many such controllers are in operation inside the LHC complex. While PLC device implementations are typically subject to IEC standards, ensuring that code is somewhat portable between vendors, there are no such standards for PLC diagnostics. In order to be able to query the internal status of a PLC (such as its last known cycle time, its operational mode or the current usage of network resources), it is necessary to resort to mechanisms specific to each PLC vendor. However, owing to the large number of PLCs deployed at CERN, we cannot simply monitor them and configure their location one-by-one; we must have a common, vendor-independent way of identifying the PLCs under monitoring.

Finally, the data resulting from diagnostic calls made to PLCs is also non-standard. It must be interpreted, transformed to a standard data format, and then stored if it is to be integrated with an established infrastructure monitoring system, such as the EN-ICE MOON platform or the CERN DIAMoN console.

To summarize, obtaining diagnostics information from a PLC is subject to three requirements:

- **Requirement 1:** To minimize as much as possible the impact of status monitoring on the PLC itself and impose absolutely no changes to the program the PLC is running (for instance, we cannot require that the PLC asset owner implements a diagnostics routine for us; it must remain completely transparent).
- **Requirement 2:** To be able to configure the inventory of PLCs to be monitored in a vendor-independent manner.
- **Requirement 3:** To bridge between the vendor-specific diagnostic data formats and a centralized device status history visualization facility.

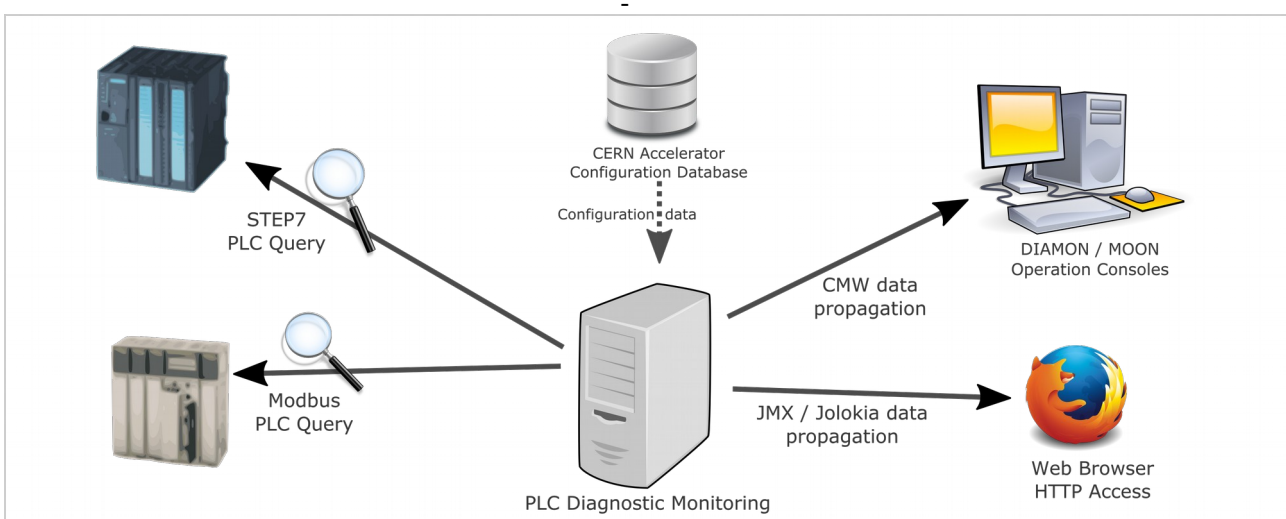

Figure 1: PLC Diagnostic Monitoring.

ISBN 978-3-95450-148-9

QUERYING PLC DIAGNOSTICS INFORMATION

As explained in **Requirement 1**, the action of monitoring equipment must be as innocuous as possible: we must not impact control logic; we must minimize as much as possible the usage of PLC network resources; and we must perform a type of black-box monitoring, so that the PLC asset owner does not have to adapt anything to the monitoring tool.

One simple way to achieve this is to resort to the built-in, yet proprietary, diagnostics capabilities that the PLCs deployed at CERN expose. Their proprietary nature, however, implies implementing a specific approach for each brand of PLC architecture we wish to monitor.

SIEMENS, for instance, offers the programmable API known as SOFTNET [3], which is comprised of a C library and a privileged Linux daemon process that together allow one to send STEP7 network frames to a given SIEMENS PLC. Through SOFTNET, it is theoretically possible to query diagnostics memory areas that are not accessible to other STEP7 implementations. LIBNODAVE [4] is another well-known, yet non-official, open-source library allowing unprivileged access to SIEMENS PLCs and the STEP7 protocol. We will consider in the case study section of this paper the pros and cons of using each SIEMENS-compatible communication library stack.

Schneider, on the other hand, allows access to diagnostics memory areas through regular Modbus [5] calls (provided that the memory addresses and their internal structure are known in advance).

The results of these calls made to diagnose a PLC differ vastly between two vendors – the data structures resulting from a diagnostics query must therefore be unpacked and interpreted specifically, that is, in relation to the PLC's hardware configuration and device manufacturing model.

INTEGRATING WITH THE CERN ACCELERATOR INFRASTRUCTURE

The CERN accelerator infrastructure under monitoring comprises over one hundred and fifty PLC devices. Any device taking part in the accelerator operation (PLCs included) must be featured in the inventory known as the CERN Accelerator Configuration Database [6], a centralized, independent, Oracle database that acts as a device inventory.

In order to fulfil **Requirement 2**, the PLC Diagnostics Monitoring tool must therefore use this central Oracle database as a basis for configuration and ensure that the database carries all required parameters to configure network access to each device.

Furthermore, the CERN Accelerator infrastructure employs the Controls Middleware (CMW) [6] protocol as its lingua-franca. The PLC Diagnostics Monitoring therefore exposes metrics through CMW in order to integrate with the CERN DIAMoN console. Figure 1 above gives a description of how the tool integrates in the CERN accelerator infrastructure ecosystem.

The two mechanisms described above are entirely CERN-specific: neither the CERN Accelerator Configuration Database nor the CMW protocol mean anything outside the CERN environment. Thankfully, the PLC Diagnostics Monitoring tool supports two generic and standard ways of configuring the inventory of PLCs under monitoring:

- flat file configuration, through a local folder containing JSON files describing each PLC;
- and through a secure HTTP interface (based on Jolokia / JMX [7]) that allows one to add and remove PLCs from the inventory, pause the monitoring, and even restart the tool.

COMMUNICATING PLC STATUS TO A CENTRAL MONITORING FACILITY

Even if the PLC Diagnostics Monitoring tool is able to query instantaneous PLC status metrics, it would not be of any use for monitoring purposes if no historical data could be maintained and visualized. Given the tool itself has no intention to act as data storage, the information it queries must be forwarded to a central monitoring facility.

Furthermore, the monitoring tool itself broadcasts a regular “heartbeat” signal (a monotonically increasing counter) used to assert the continuous availability and responsiveness of the monitoring process.

The CERN control room employs two monitoring visualization platforms: the DIAMoN console and the ENICE MOON monitoring platform. Both platforms have been developed at CERN to provide LHC operators with a global, yet detailed, overview of the status of the entire accelerator infrastructure. Both tools can provide historical data on the status of equipment and can integrate PLC Diagnostics data through CMW.

Additionally, with the intention of being reusable outside of the CERN environment, the PLC Diagnostics Monitoring tool offers such for the Java Monitoring Extensions (JMX). JMX is an open-source community specification and reference implementation that describes both a monitoring meta-model and a remote procedure call protocol. Thanks to JMX, any Java developer can easily expose objects and their properties, to be read or modified, and functions to be called remotely. A large number of monitoring tools on the market support JMX out of the box and can incorporate JMX data in their visualizations.

One drawback of JMX is that it is initially Java-centric and thus accessible through Java-specific protocols only: the open-source Jolokia project [8] provides a trivial way to make any JMX-enabled application access through HTTP. Jolokia does not require any code changes, but only the runtime inclusion of its library (and, optionally, a

ISBN 978-3-95450-148-9

security configuration) to make all JMX metrics accessible through a simple REST interface, exposing data in standard JSON format.

As a result of using Jolokia, it is trivial to include PLC Diagnostics Monitoring status data into a standard monitoring solution, such as the ubiquituous ELK [8] stack (ElasticSearch, Logstash, Kibana).

CASE STUDY: USING SIEMENS SOFTNET IN THE FIELD

As mentioned above, SIEMENS SOFTNET is a Linux-compatible library that allows the interoperation of Linux with SIEMENS PLCs, such as the S7-400 series.

SOFTNET supports integration through two elements:

- a non-privileged application library written in C, on top of which any software developer can write code to interact with a SIEMENS PLC;
- and a privileged daemon process, to be installed by a root Linux user (the daemon must also be configured by a root user and run continuously; it acts as a gateway between PLCs and non-privileged applications).

SOFTNET also requires the use of text files that follow a very specific format; it is thus very sensitive to simple formatting errors (like the usage of tabs instead of spaces for parameter separation). This separation of privileges and the non-standard configuration mechanisms employed make it awkward to deploy and integrate. While the C API invites the development of third-party software, both the daemon and configuration workflow prove to be a serious hindrance.

Finally, through usage of the library, we were able to expose severe shortcomings and crashes when operating in multi-threaded mode, making it only suitable to monitor one PLC at a time in sequence.

During the course of development, SOFTNET was eventually dropped in favour of LIBNODAVE, which provides identical functionality without any of the complications imposed by SOFTNET.

CASE STUDY: IMPACT OF THE MONITORING ON A RUNNING PLC

Measuring the impact caused by the monitoring is essential to prove that it will not adversely affect the performances and control functions of the PLC.

To this end, we have carried out extensive testing against two typical SIEMENS PLCs commonly deployed at CERN. Figures 2 and 3 below summarize our measurements. We can observe that monitoring in itself affects a low-end device more significantly than a high-end one, but overall remain negligible when the PLC is under load (*i.e.* executing complex control logic and communicating with a SCADA software and its administration console).

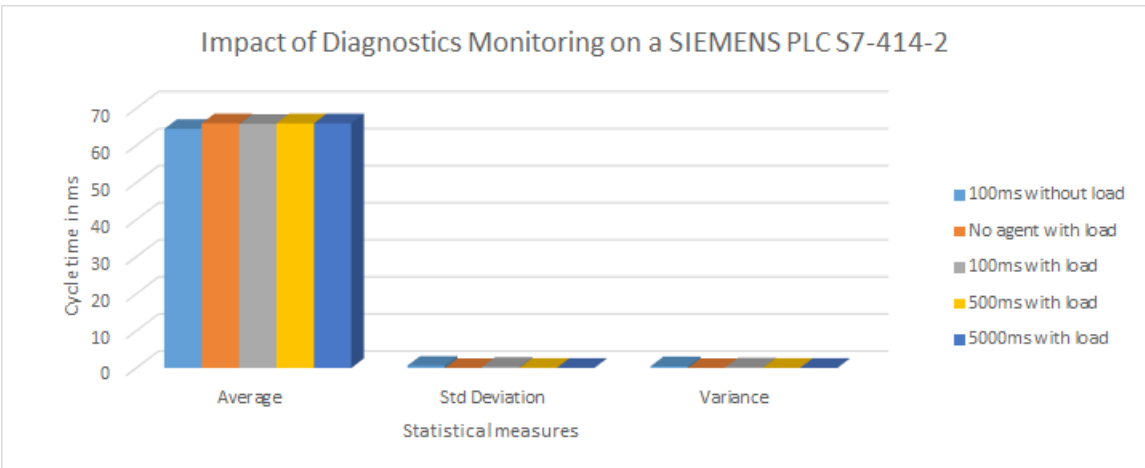

Figure 2: Impact of diagnostics monitoring on a SIEMENS PLC S7-315-2PN/DP.

ISBN 978-3-95450-148-9

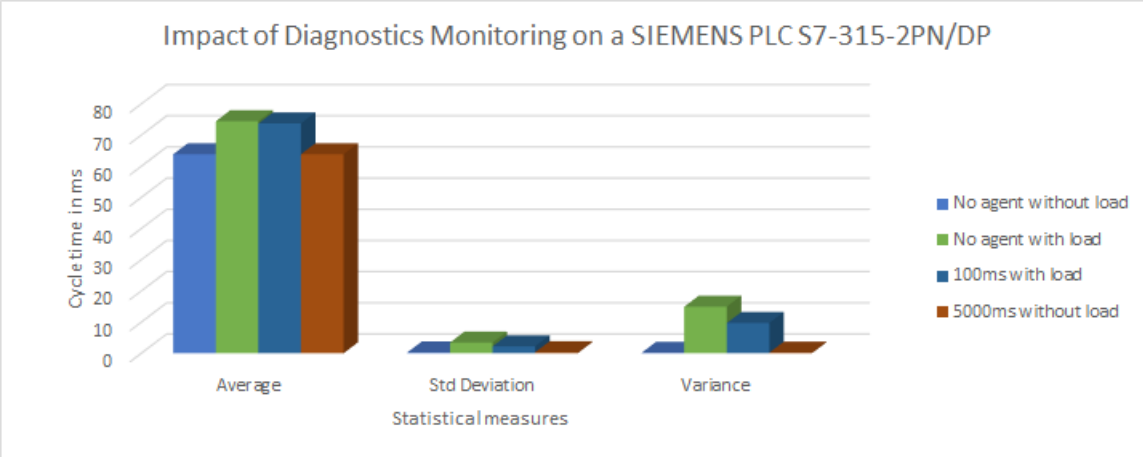

Figure 3: Impact of diagnostics monitoring on a SIEMENS PLC S7-414-2.

CONCLUSION

In conclusion, the PLC Diagnostics Monitoring tool offers a simple way to monitor the health of PLCs deployed in the field. While it integrates with the CERN infrastructure, it can easily be reused in any Linux environment and integrated with standard monitoring tools such as the Hawt.IO console [9] or the ELK monitoring stack.

REFERENCES

[1] F. Bernard et al., "Monitoring Controls Applications at CERN", ICALEPCS'2011, Grenoble, France (2011)

[2] W. Buczak et al., "DIAMON2-Improved monitoring of CERN's Accelerator control infrastructure", Oct 2013, ICALEPCS'13, San Francisco, USA

[3] SIEMENS AG, "SOFTNET S7 Linux", 25 Sept 2014, http://w3.siemens.com/mcms/human-machine-interface/en/customized-products/customized-software/portfolio/pages/softnet-linux.aspx

[4] T. Hergenhahn, "Exchange data with Siemens PLCs", 22 May 2014, http://libnodave.sourceforge.net/

[5] A. Dworak et al., "The new CERN Controls Middleware", 19th International Conference on Computing in High Energy and Nuclear Physics, (CHEP 2012), May 2012, New York, USA

[6] H. Wong et al., "Java Management Extensions", Java Specification Request (JSR-3), rev. 04 Mar 2014, https://www.jcp.org/en/jsr/detail?id=3

[7] R. Huss, et al. "Jolokia : JMX on Capsaicin", 11 July 2015, http://jolokia.org/about.html

[8] S. Banon, "ElasticSearch : you know, for search", 12 June 2012, http://thedudeabides.com/articles/you-know-for-search-inc

[9] J. Strachan et al., "Hawt.IO", retrieved 09 Sept 2015, http://hawt.io/

ISBN 978-3-95450-148-9