ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA037

FUTURE ARCHIVER FOR CERN SCADA SYSTEMS

P. Golonka[†], M. Gonzalez-Berges, J. Guzik*, R. Kulaga, CERN, Geneva, Switzerland

Abstract

The paper presents the concept of a modular and scalable archiver (historian) for SCADA systems at CERN. By separating concerns of archiving from specifics of data-storage systems at a high abstraction level, using a clean and open interface, it will be possible to integrate various data handling technologies without a big effort. The frontend part, responsible for business logic, will communicate with one or multiple backends, which in turn would implement data store and query functionality employing traditional relational databases, as well as modern NOSQL and big data solutions, opening doors to advanced data analytics and matching the growing performance requirements for data storage.

INTRODUCTION

Robust and scalable archiver (historian) for large control systems such as those of CERN's LHC accelerator and is associated experiments, has been a recurrent topic discussed over the past decade at the ICALEPCS conference. In particular, the evolution of the RDB Oracle Archiver component of the WinCC Open Architecture (OA) [1] commercial SCADA system has been presented in [2][3].

With the already planned ramp in luminosity of the CERN accelerator complex, one may expect the corresponding rise to be required in the throughput of control systems and in the amount of produced and processed data. Moreover, a more and more widespread adoption of data analytics is a notable trend for industrial control systems, empowering the operators with new tools such as predictive maintenance, event classification and plant optimization. In this context, the archiver becomes one of the key component of modern control systems bridging the gap between domains of classical control systems and data analytics. On the other end, the Internet-of-Things revolution yields appliances capable of embedding advanced control systems into small form factor devices with ultra-low power consumption, requiring scaled-down versions of archiver to be running locally, or in cloud/swarm deployments.

Based on the experience gathered over a decade of operation of the WinCC OA-based control systems at CERN, notably using the archiving systems based on large relational Oracle databases and anticipating future challenges, we propose an architecture for the new scalable and modular archiver for industrial-control systems. Even though primarily aimed to be used with WinCC OA (de-facto standard for CERN deployments), the open architecture allows its integration with "any" other control system that needs to store and query historical process data (such as C2MON [5]).

The NextGen Archiver project presented in this paper is developed in collaboration between CERN and Siemens/ETM (the vendor of WinCC OA), in the framework of the CERN openlab project [4].

RDB Archiver

The RDB archiver was originally developed for the LHC experiments and later on, extended to other CERN systems (accelerators and technical infrastructure). The main driving force for the new development were the requirements in terms of writing/reading performance as well as the long-term storage of the data. At the time of the development (mid 2000s) the ubiquitous technology for databases was based on the relational model. A first attempt was made to have a generic relational archiver; however, it didn't meet the requirements and a specialized version had to be developed for Oracle, the CERN IT chosen solution at the time [3]. The project was a very successful common effort between several CERN groups and ETM. The main outcome was a scalable archiver solution that could be tailored, with the appropriate hardware, to the different CERN installations.

The RDB archiver was put in production for the LHC start-up and used since, with systematic modifications and a major evolution during the first LHC Long Shutdown to meet the ever-growing performance requirements.

Since the initial development, a number of technologies have come up in the domain of data logging and analysis (e.g. Hadoop ecosystem, timeseries databases, etc). This made necessary to reconsider the initial choice of technologies for the major upcoming upgrades for the CERN control systems linked to the High Luminosity LHC.

MOTIVATION

Based on the fruitful experience of the collaboration on the RDB Archiver, CERN and ETM/Siemens decided to start a R&D project on the next generation archiver for WinCC OA systems, in the scope of the Openlab project.

Following the trends observed in the IT sector, the main motivations in this project for CERN were:

- Provide an archiving solution capable of scaling-up to data throughputs required for next upgrades of the LHC and its experiments beyond 2020.
- Enable effective use of data analytics tools.
- Enable the usage of NOSQL (Not Only SQL) data store and processing technologies.

[†] email:Piotr.Golonka@CERN.CH
* supported by Siemens through CERN openlab collaboration

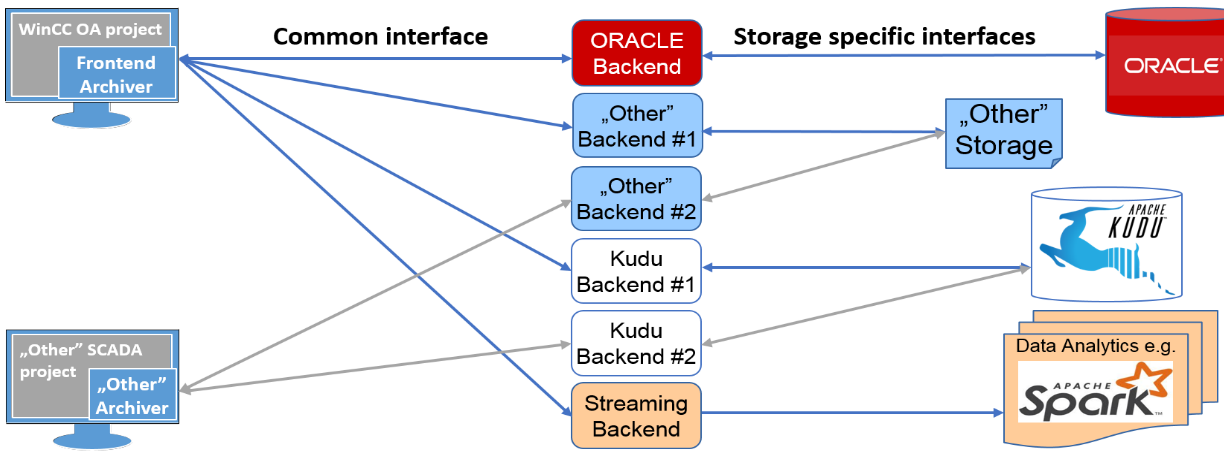

Figure 1: Architecture of the NextGen archiver.

Direct input from users was to query and efficiently process historical data in much larger time windows, with higher throughput and quick "interactive" response to operators. Two use cases for data queries which scale poorly with RDB Archiver need to be addressed in particular: the *event screen*, and *event replay*. The former allows to list a sequence of events that happened at a specified time window for a specified set of devices; hence it requires that the archiver provides historical values for many devices, with possibly complex filters and yields moderate amount of data. The latter should allow to replay the complete sequence of events through the system starting at a specified point in time and requires massive extraction of data for all devices of a control system or its partition. Moreover, the exact start and end time for the query need to be established from data. To reconstruct the state of the process image at a specified starting time, one needs to find the most recent value change prior to it, for all signals in the system; realizing that some of these values might have not changed since a long time, one needs to execute an unbound search through the history of recorded timestamps of signals. This is a particularly costly operation with the current RDB Archiver implementation of WinCC OA.

For ETM, the project would permit to propose lower-cost, scalable archiver solution for customers, and extend the set of supported data-store technologies beyond Oracle DBMS. For both parties, the project of the new archiver was the opportunity to consider and prototype enhancements and new features that were not possible to achieve with the existing RDB Archiver implementation. Improved reliability, data integrity and monitoring were declared to be as important as throughput. Ultimately, an architecture permitting to easily extend the archiver with new data storage and processing backends was demanded to be able to promptly react to the dynamically changing landscape of technologies.

ARCHITECTURE

The key design decision for the architecture of the next generation archiver was to separate the concerns of WinCC OA and database storages into a frontend and backend modules, and to connect them with a platform-neutral communication channel, which would define the common interface between the parts. This modular architecture, presented in Fig. 1, has a number of advantages.

Primarily, it allows to integrate new data-storage systems, as new backends, with no need to implement a complete new archiver, and with no need for backend developers to be familiar with complexity of WinCC OA programming interfaces. And conversely, it allows to use the existing backends with any other (non-WinCC OA) system by implementing a custom frontend module. Assuming that the communication interface is platform-independent, it becomes possible for frontend and backends to run on different physical machines, with different operating systems, and implemented with different programming languages. For example, a WinCC OA system running on a Linux server and implemented in C++ to be integrated with database services only running on Windows, or requiring implementation in Java or Python.

Another architecture decision was to permit the use of multiple backends simultaneously. As a result, it is possible to store data with different decimation/aggregation and data retention strategies. For example, one could store unfiltered data to a short-time, shallow backend storage, and store hourly-averages into another backend that holds them for the lifetime of the application. Ultimately, one could consider a backend capable of streaming value changes into a data analytics platform such as Apache Spark, or building mirrors of the control process images.

The common interface is divided into three modules, that form a universal Application Programming Interface:

- *writing interface:* responsible for forwarding value changes and alarms to the backends; implemented as a bidirectional channel which ensures that the backends acknowledge reception of chunks of data, and provides buffering of data in the event of temporary loss of connectivity or unavailability of the backend. Implementations of this interface would typically write data to their storage, using technology-specific interfaces.

- *configuration/monitoring interface:* allows to integrate backend configuration and monitoring tasks with the supervisory layer, transmits the metadata-change information (such as creation/deletion/renaming of datapoints), provides standardized error notification, heartbeat mechanisms between a frontend and backends, and allows for administrative tasks such as restart or reconnection to be commanded from the supervisory layer to the backend. For this reason, it is designed as a multi-purpose, bidirectional channel.
- *reading interface:* responsible for forwarding data-queries and returning their values. The data format for requests and response is designed to be independent from query systems of either SQL or NOSQL data storages, and focuses on typical use cases required for SCADA systems.

As it is not required to implement all the interfaces, one may easily develop backends with read-only (e.g. to retrieve data from legacy storage systems) or write-only functionality (forwarding data).

The frontend and backend parts may run as separate processes but they may also be executed inside the same process. This allows to simplify deployment scenarios in particular for smaller architectures where data is stored on the same server as the control system itself, and could embed all the necessary libraries. On the other hand, for large, distributed deployments, one could consider connecting many frontend modules (from individual parts of control system) to a dedicated high-availability server hosting the backend part of the archiver, possibly also collocated with the data storage/processing.

TECHNOLOGY SELECTION

The prototyping phase for the project was preceded by the selection and evaluation of technologies that could serve as the interfacing medium; solicited were those offering implementations for many programming languages and operating systems that provide clean interfaces and simple yet powerful architectures. Insights provided in [6] were the source of our inspiration too.

We decided on the use of ZeroMQ [7] as a transport layer and Google Protocol Buffers [8] for data serialization and interfacing. The choice was driven by the popularity and widespread adoption of these technologies, their availability on multiple platforms, vivid community of users, as well as licensing terms that would be permissive for both: open source and commercial use. Already with first exercises we were convinced about suitability of this choice and the flexibility of the architecture it enables. Functional and performance tests performed later confirmed our choice.

As the primary aim of the project is to provide a new archiver for WinCC OA, we focused on implementing the NextGen archiver's first frontend, using WinCC OA's C++ API. The use of C++11 standard library with the Qt framework allowed to embark upon the development for Linux and Windows platforms. Clang-tidy used in addition for test driven development allowed for delivery of successive prototypes with high quality of the code and increased reliability even at early stages of the project.

An effort to select the initial set of data storage and processing technologies for the backends addressing various use cases has also been initiated. For smaller deployments a technology that could functionally replace the existing file archiver and bringing the benefits of modern NOSQL solution is being investigated with projects such as MongoDB [9] and InfluxDB [10] in evaluation[11]. For compatibility with existing large deployments, notably those at CERN, a backend for Oracle databases, compatible with the currently used RDB Archiver has been declared as necessary for smooth transition, as well as to access existing historical data. Ultimately, a BigData backend enabling data analytics as well as high-performance data queries on large data sets is being worked on. At the current prototyping stage, we focus on the Apache Kudu [12] as underlying technology. We are also considering the use of NXCALS [13], CERN's new logging system based on Hadoop ecosystem, for long-term storage and data analytics.

CURRENT STATE

The NextGen Archiver project delivered its first functional prototype, with the complete initial set of interfaces and the implementation covering the most important functionality. The communication and messaging layer has been thoroughly tested from the functional point of view. The functionality of the writing interface, with the option for buffering of data to disk, was found to work very well, as well as simple configuration/monitoring and data queries.

Initial implementation of the backends compatible with RDB Archiver, Kudu and a handful of other backends has been achieved proving that the architecture allows for rapid integration of data-storage technologies, as well as to aid in the process of technologies selection as backends.

The openness of the architecture was demonstrated with implementation of the *simplified frontend* for data generation and testing purposes. Implemented as a student project in summer 2017, it allows to generate and inject test data into backends without using WinCC OA. Similarly, the *reference backend*, ignoring all value changes and responding to queries with ad-hoc generated data was delivered and becomes indispensable for the envisaged performance tests.

In addition, a library of high-level functions has been implemented to spare C++ programmers from using low-level interfaces of the transport/serialization libraries; for other programming languages similar API packages may be provided, if necessary.

An intensive campaign for scalability and performance test is in preparation at CERN for the autumn of 2017. It should allow to measure and test the performance of the prototype at the scale of large systems already used at CERN and beyond to estimate the limits of the architecture and technologies.

OUTLOOK

The prototype of the NextGen archiver presented in this paper is at the moment a part of the research and development project held in collaboration between ETM/Siemens and CERN. Considering the promising results achieved, we believe that it might become the basis of the much improved open archiving system for WinCC OA.

The architecture of the NextGen Archiver, as suggested in the introduction, enables its use for other purposes as it can act as a general-purpose data interface. We await with excitement the collaboration with the ALICE O2 project presented at this conference [14], to use the NextGen Archiver to effectively forward the complete process image of the distributed Detector Control System of ALICE to their Data Acquisition system. The prototyping should start in Autumn 2017.

We also envisage to prototype the integration with NXCALS [13], as it will soon become the main data storage and analytics platform for accelerator controls at CERN. Once it is sufficiently tested, the integration with NXCALS might replace the currently existing process of data transfer between the online controls database and the Accelerator Logging database. A storage backend for this purpose could run in parallel to backends that archive data for online use.

In the long-term perspective, the NextGen archiver opens new possibilities, in particular when applied in non-standard setups. Similarly to the O2 intgration project, it may be used to forward data to enable event stream processing or complex event processing.

Ultimately, it could also be used to provision offline copies of the SCADA control process image to external sources, such as web applications, allowing for secure, read-only access to controls data for large user communities.

CONCLUSION

We presented the concept of the modular archiving system for controls data and its first prototype that would allow for WinCC OA-based control systems to use a variety of data storage backends. The open architecture allows to extend the project to other SCADA systems or integration of new storage solutions through implementation of high-level interfaces. New domains of usability as scalable data forwarder become also possible. Upcoming scalability and performance tests will validate those concepts in practice and show what are their scalability limits.

ACKNOWLEDGEMENT

The project progresses at the fast pace thanks to constant support, collaboration, as well as funding from ETM and Siemens, in particular in the framework of the CERN openlab collaboration. The concept and prototype presented by the authors of this paper were achieved by common effort with our ETM/Siemens colleagues: Rene Heissinger, Ewald Sperrer, Andreas Vogler and Tobias Woegerbauer. We are grateful to the CERN Database group, in particular Zbigniew Baranowski, Luca Canali, Prasanth Kothuri for sharing their experience, providing development and testing infrastructure and their valuable input to discussions on technology selection. We also thank Urishita Puri, CERN openlab Summer Student 2017 for her efforts as a developer in the project.

REFERENCES

[1] Simatic WinCC Open Architecture (previously PVSS) SCADA Software from ETM (Siemens subsidiary), http://www.etm.at

[2] P. Golonka *et al*, “Database Archiving System for Supervision Systems at CERN: A Succesful Upgrade Story”, Conference Proceedings ICALEPCS2015, Melbourne, Australia, Oct. 2015, contribution MOPGF021, ISBN 978-3-95450-148-9, DOI: 10.18429/JACoW-ICALEPCS2015-MOPGF021

[3] M. Gonzalez-Berges “The High Performance Database Archiver for the LHC Experiments”, Conference Proceedings ICALEPCS2007, Knoxville, USA, 2007, contribution ROPA02

[4] CERN openlab project: http://openlab.cern

[5] B. Copy et al, “C2MON SCADA Deployment on CERN Cloud Infrastructure”, ICALEPCS2017, contribution THBPL01

[6] A. Dworak et al, "Middleware Trends and Market Leaders 2011", Conference Proceedings ICALEPCS 2011, p.1334 (2011)

[7] ZeroMQ Distributed Messaging, http://zeromq.org and whitepapers available on the site

[8] Google Protocol Buffers, https://developers.google.com/protocol-buffers/

[9] MongoDB document database http://www.mongodb.com

[10] InfluxDB purpose built time series database, part of InfluxData time series platform https://www.influxdata.com/time-series-platform/

[11] "WinCC OA NextGen Archiver: OSS Database Selection Process", presented at CERN Openlab Open Day, https://indico.cern.ch/event/633007/

[12] Apache Kudu: a storage layer for Hadoop ecosystem to enable fast analytics on fast data http://kudu.apache.org

[13] NXCALS: next generation implementation of CERN Accelerator Logging Service: Spark Summit 2017 in preparation

[14] P. Chochula *et al*, “Challenges of the ALICE Detector Control System for the LHC RUN3”, ICALEPCS 2017, contribution TUMPL10