ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOMPL010

DATA STREAMING WITH APACHE KAFKA FOR CERN SUPERVISION, CONTROL AND DATA ACQUISITION SYSTEM FOR RADIATION AND ENVIRONMENTAL PROTECTION

A. Ledeul, G. Segura Millan, A. Savulescu, B. Styczen, CERN, Geneva, Switzerland

Abstract

The CERN HSE - occupational Health & Safety and Environmental protection - Unit develops and operates REMUS - Radiation and Environmental Unified Supervision - , a Radiation and Environmental Supervision, Control and Data Acquisition system, covering CERN accelerators, experiments and their surrounding environment.

REMUS is now making use of modern data streaming technologies in order to provide a secure, reliable, scalable and loosely coupled solution for streaming near real-time data in and out of the system.

Integrating the open-source streaming platform Apache Kafka allows the system to stream near real-time data to Data Visualization Tools and Web Interfaces. It also permits full-duplex communication with external Control Systems and IIoT - Industrial Internet Of Things - devices, without compromising the security of the system and using a widely adopted technology.

This paper describes the architecture of the system put in place, and the numerous applications it opens up for REMUS and Control Systems in general.

INTRODUCTION

CERN Radiological and Environmental Monitoring System

CERN HSE Unit is in charge of the implementation of CERN Safety Policy, which includes Radiation Protection and Environmental Impact Monitoring.

This CERN-wide monitoring program measures the radiological and environmental impact of CERN, and ensures workplace safety. It also provides the necessary data for reporting to the public and host states authorities.

In order to achieve these objectives, the HSE Unit sets up and operates heterogeneous instrumentation, able to measure the nature and quantity of ionizing radiations produced by the accelerators, possible contaminations and conventional environmental parameters.

In addition, CERN HSE Unit provides a SCADA - Supervision, Control And Data Acquisition - system, called REMUS [1], based on WinCC Open Architecture - WinCC OA [2], interfacing this instrumentation and allowing its operation from various control rooms scattered across the Organization.

At the time of writing, REMUS is interfacing 80 device types. It contains 650,000 tags, manages 84,000 alarms and handles a throughput of 3,700 changes per second. REMUS archives roughly 38 billion measurements per year.

Need for a Stream Processing Platform

One of the main challenges of SCADA systems is to exchange data to and from other systems, in particular data visualization tools, web applications, data storage solutions and other SCADA systems.

Furthermore, the expansion of Industrial IoT devices and their respective monitoring solutions, created a need for a fast and standard intercommunication solution that would allow the SCADA to interface with these devices.

Recent developments in feature-rich data streaming platforms, such as Amazon Kinesis [3] or Apache Kafka [4] have made this achievable with minimal investment.

In the context of Safety SCADA Systems, certain requirements must be met for such data streaming solutions:

- Security: data shall be streamed and accessed in a secure manner; Access Control Lists - ACLs - must be put in place.
- Reliability: given the important nature of data, the streaming solution has to be fault tolerant and highly available.
- Scalability: the platform shall be scalable horizontally in order to face the steady growth of data volumes and exchanges.
- Performance: the platform shall be able to handle an important throughput, and provide the data with a low latency.
- Loose coupling: the SCADA system must be able to stream data in and out with low to no impact on the system itself. In the context of near-real time data streaming, an asynchronous transmission of data is preferable, in order to reduce latency, diminish producer / consumer dependency and accommodate simultaneously multiple types of data consumers.

Open-source Data Streaming

With regards to previous requirements, REMUS opted for Apache Kafka, a scalable, open-source and widely used distributed streaming platform, originally developed by LinkedIn for log processing [5]. It is today used by more than 13,000 companies, including Netflix, Uber, Spotify, Cisco, Yahoo, Twitter, Square, and overall a third of the Fortune 500 companies.

Apache Kafka provides the following key functionalities:

- Publisher / Subscriber mechanism for streams of data (Asynchronous message queuing system)
- Built in Kerberos / TLS - Transport Layer Security - based authentication, authorization and encryption mechanisms
- Distributed and fault-tolerant data retention

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOMPL010

- Partitioning
- Topic-based communication model
- On-arrival data streams processing
- Exactly Once delivery semantics
- Idempotent data publication

Apache Kafka at CERN

CERN IT and Beams Departments operate Kafka clusters, used by NXCALS [6] - Next CERN Accelerator Logging Service - (Beams Department), IT Security Event Operations and CERN Data Center Infrastructure Monitoring (IT Department).

REMUS takes advantage of this robust infrastructure, accessible from the different CERN networks, to set up its data streaming solution (see Fig. 1).

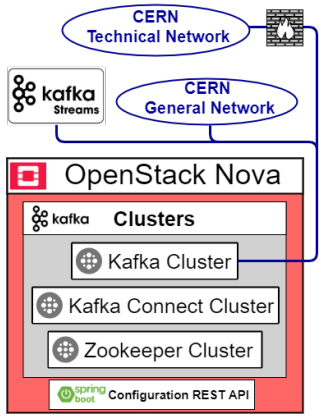

Figure 1: REMUS Kafka cluster architecture.

REMUS DATA STREAMING

CERN HSE Unit developed an open source WinCC OA / Kafka full-duplex data transmission solution [7], built on top of the WinCC OA C++ Driver API, libdrkafka and cppkafka libraries, which allows to publish and/or consume messages onto and/or from a Kafka cluster. The operation of this connector has opened up new possibilities and improvements for this SCADA system, as it became compatible with a widely adopted, secured and powerful application databus (see Fig. 2).

Measurements Data Streaming from WinCC OA

All ongoing measurements retrieved by the REMUS supervisory system are published in near-real time through Kafka, in a dedicated Kafka Topic. These messages are generic JSON strings, structured as follows: the identification of the data stream source (typically the identification of the instrument acquiring the data), the recorded value, the measurement unit, and the associated acquisition timestamp, plus additional flags such as the equipment health and alarming statuses. The identifier is also used as the Kafka key. Kafka guarantees that for a given key, all corresponding messages are stored in the same partition, in the order they are sent, as long as the number of partition remains constant. This mechanism helps with the optimization of consumers' processing, based on "At Least Once" delivery semantics.

New publication tags are added automatically, at run-time, to the SCADA when a new equipment is declared,

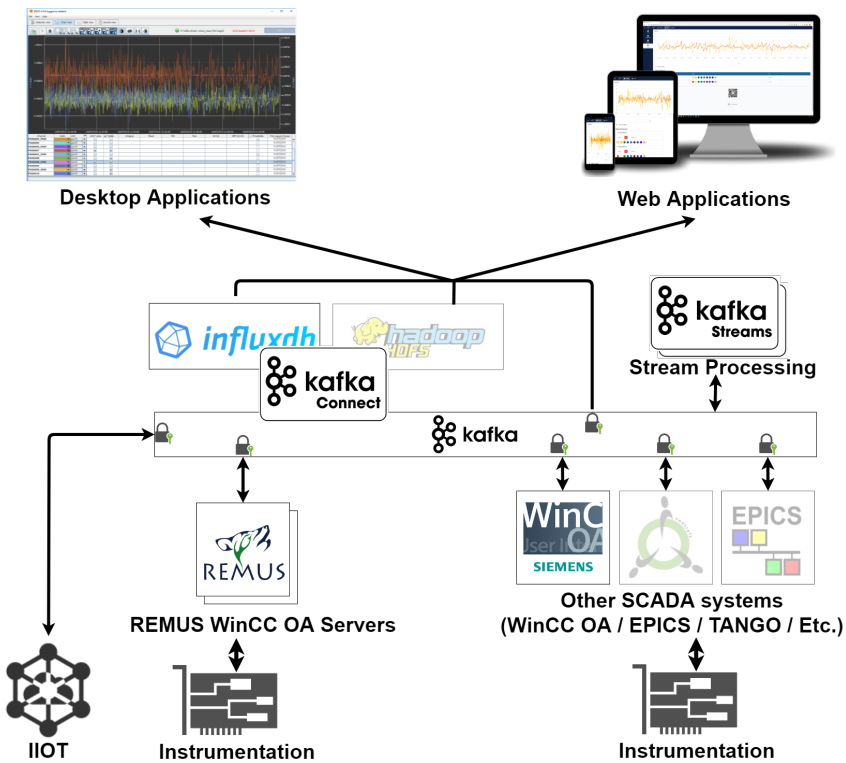

Figure 2: REMUS data streaming architecture.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOMPL010

allowing immediate streaming. These tags are declared and constructed through transformations and compositions of existing tags (timestamp, value, etc.), and are created at runtime by a dedicated process. Then, every publication tags update is streamed as a message to the Kafka cluster by the WinCC OA / Kafka connector. In case source tags (identifier, value, timestamp, unit and flags) are updated asynchronously by the instrumentation, the connector features a debouncing configuration for each publication. This mechanism ensures consistent streams (see Fig. 3).

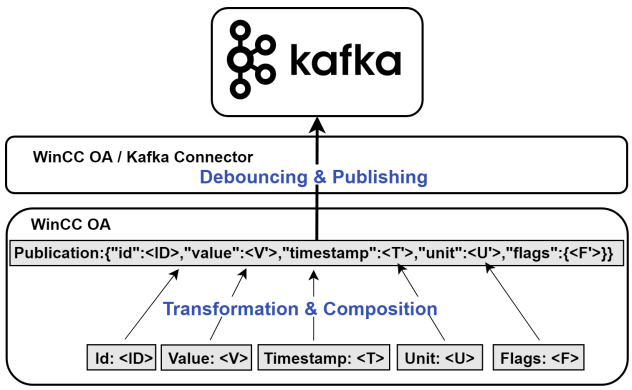

Figure 3: REMUS measurements publication.

At the time of writing, REMUS contains 3,300 publication tags, sending about 600 messages per second to CERN Kafka brokers.

REMUS measurement Kafka topic is composed of 24 partitions, leveraged by 3 brokers with a replication factor of 3, therefore assuring high availability.

Alarms/Faults Data Streaming from WinCC OA

Similarly to the measurements streaming mechanism already in production, the short term plan is to implement the streaming of all alarms and fault state changes. This feature will provide a straightforward way of sharing alarm state changes to external systems.

DATA STREAMS APPLICATIONS FOR CONTROL SYSTEMS

Stream Processing and Data Enrichment

The integration of Stream processing frameworks/ platforms such as Apache Spark [8], Apache Storm [9], Apache Flink [10] and Apache Kafka Streams facilitates the usage of stream processing for big data purposes. Stream processing, as opposed to batch processing, allows data processing and transformation in near real-time for large sets of data, providing a useful functionality for Control Systems.

REMUS is relying on Kafka Streams, released in 2016, for its stream processing functionalities. Kafka Streams provides "Exactly Once" semantics and benefits from Kafka scalable and distributed architecture advantages.

In the context of REMUS measurement data streaming, the messages are enriched by a stateless stream process. The enrichment consists of decoding the unit identifier and the flags to a user-friendly representation, suitable for display in front-end applications. Since the messages are immutable once sent to the topic, another Kafka topic is used for the storage of enriched messages (see Fig. 4).

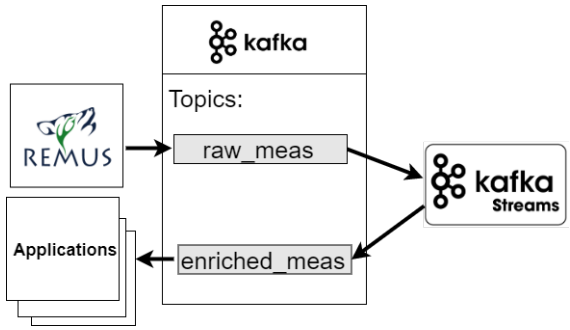

Figure 4: REMUS measurement messages enrichment.

The processing of alarms state changes will be done by a stateful stream process, also based on Kafka Streams.

Kafka Connect and Data Archiving

Kafka Connect is an open-source framework providing an API for building Kafka connectors. Connectors can either take data from a Data Source and push it into a Kafka cluster or pull the data from a Kafka cluster into a Data Sink.

A large collection of connectors is already available, interfacing Kafka with PostgreSQL, MySQL, ActiveMQ, Elasticsearch, S3, HDFS, Influx DB, etc. [11] REMUS utilizes a connector between its enriched data stream and the open-source time series database Influx DB [12], for fast querying of data series from client applications.

Other connectors could be used in the future, in order to push the data to other data storage solutions optimized for specific usages.

Desktop Data Visualization Tool

REMUS Data Visualization Tool, ERGO - Environment and Radiation Graphic Observer -, is used in CERN control rooms and by Environmental and Radiation Protection experts for analytics of NRT - Near Real-Time - and historical data.

ERGO now takes advantage of this new Data Streaming Platform, consuming newly published measurement messages for near real-time display.

This set of measurements is completed with historical data coming from REMUS long term measurement database, populated through an independent batch processing pipeline.

In addition to its standard message subscription mechanism, Kafka integrates data retention capability, allowing ERGO to fetch messages sent to the Kafka cluster in the last minutes by offsetting the Kafka partition logs. This feature ensures continuity of measurements coming from the long term database, which are available after a couple of minutes, and the near real-time measurements (see Fig. 5).

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOMPL010

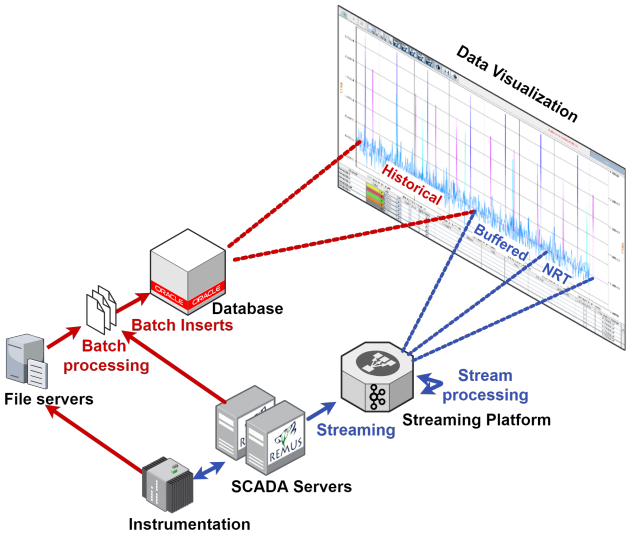

Figure 5: ERGO data source mix.

Thanks to the Kafka partitioning and Kafka consumer groups concepts, ERGO scales its data consumption horizontally by launching several consumers on different threads.

A map that links keys to partitions is cached locally, in order to prioritize consumers that will consume data from the exact partitions containing the measurement channels the user chooses to load at start-up (see Fig. 6).

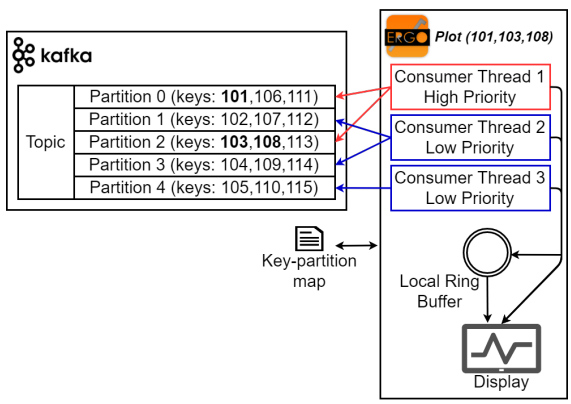

Figure 6: ERGO Kafka Consumers start-up strategy.

Relying on this streaming infrastructure allowed for a secured, reliable and scalable near-real time data source, with a latency of less than 1 second between the data acquisition from the distributed instrumentation, the stream processing and the near real-time display of measurements in CERN control rooms. It is in operation since March 2019 and has not suffered any downtime at the time of writing.

One considered evolution of this model is the usage of Influx DB as data source of the buffered data, instead of the Kafka partition logs.

Web and Mobile Data Visualization Tool

Web-based applications such as Grafana or Chronograf provide data visualization and analytics tools out-of-the-box, through Influx DB data source plugins.

The streams of data generated from REMUS are also exploited by a responsive web application, REMUS Web, able to display measurements in near real-time, among other information about the SCADA processes. Write access to REMUS Kafka Topics from the SCADA is secured through TLS Encryption and Kerberos Authentication.

Access to the web application is protected by the CERN SSO - Single Sign On - service [13].

Automatic generation of QR Codes from the web interface provides a fast, user-friendly yet secured way for accessing near real-time data from any terminal device having internet connectivity (see Fig. 7).

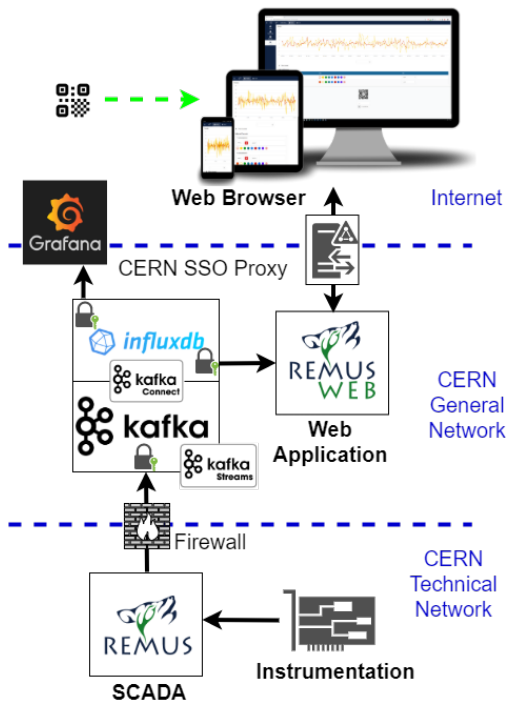

Figure 7: REMUS web measurement data pipeline.

One considered evolution of this pipeline is the addition of other data sources than Influx DB, such as REMUS long term measurement database.

SCADA Systems Inter-messaging

The wide adoption of Apache Kafka as a solution for asynchronous data streaming opened up possibilities for REMUS to communicate with external SCADA systems based on different technologies such as Tango, EPICS [14], WinCC OA etc.

Industrial IoT

Utilizing a bi-directional connector between WinCC OA and Kafka permits communication with any IIoT layer having Kafka connection capability.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-MOMPL010

Furthermore, advanced Kafka configurations such as Log compaction and mirroring, and native connectivity with machine-to-machine oriented protocols such as MQTT makes Kafka a suitable technology for the application layer databus of a Layered Databus Architecture pattern [15].

CONCLUSION

The CERN HSE Unit enriched its WinCC OA based SCADA system functionalities using Apache Kafka as a data streaming Platform.

Through a minimal time investment, providing publishing and consuming Kafka topics capability to a SCADA system increases considerably its possibilities of developing loosely coupled functionalities in the domains of data processing, data archiving, data monitoring and inter-systems communication.

ACKNOWLEDGMENT

We would like to thank all the members, past and present, of the REMUS Project team as well as colleagues from Radiation Protection, Environment and Control groups for their fundamental contribution to the success of the REMUS project.

We particularly would like to thanks the CERN Information Technology Department for their expertise and help provided during the interfacing of REMUS with CERN Kafka infrastructure.

REFERENCES

[1] A. Ledeul, G. Segura Millan, A. Savulescu, B. Styczen, and D. Vasques Ribeira, "CERN Supervision, Control and Data Acquisition System for Radiation and Environmental Protection", in *Proc. 12th International Workshop on Personal Computers and Particle Accelerator Controls (PCaPAC'18)*, Hsinchu City, Taiwan, Rep. of China, 2018, pp. 248-252. doi:0.18429/JACoW-PCaPAC2018-FRCC3

[2] WinCC OA, https://www.winccoa.com

[3] Kinesis, https://aws.amazon.com/fr/kinesis

[4] Kafka, https://kafka.apache.org

[5] J. Kreps, N. Narkhede and J. Rao, "Kafka: A distributed messaging system for log processing", in *Proc. 6th Workshop on Networking Meets Databases (NetDB'11)*, Athens, Greece, Jun. 2011.

[6] J. P. Wozniak and C. Roderick, "NXCALS - Architecture and Challenges of the Next CERN Accelerator Logging Service", presented at the 17th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'19), New York, NY, USA, Oct. 2019, paper WEPHA163.

[7] Github repository for WINCCOAkafkaDrv, https://github.com/cern-hse-computing/WCCOAkafkaDrv

[8] M. Zaharia *et al.*, "Apache Spark: A Unified Engine for Big Data Processing", in *Communications of the ACM*, vol. 59, pp. 56–65, Nov. 2016. doi:10.1145/2934664

[9] A. Toshniwal *et al.*, "Storm@Twitter", in *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD'14)*, Snowbird, Utah, USA, Jun. 2014, pp. 147–156. doi:10.1145/2588555.2595641

[10] P. Carbone *et al.*, "Apache Flink: Stream and Batch Processing in a Single Engine", in *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, vol. 36, no 4, pp. 28-38, Jan. 2015, http://sites.computer.org/debull/A15dec/issue1.htm

[11] Kafka Connectors, https://www.confluent.io/hub

[12] Influx data, https://www.influxdata.com

[13] E. Ormancey, "CERN single sign on solution", *Journal of Physics: Conference Series*, vol. 119, no. 8, p. 082008, Jul. 2008. doi:10.1088/1742-6596/119/8/082008

[14] D. Werder *et al.*, "EPICS Data Streaming and HDF File Writing for ESS Benchmarked Using the Virtual AMOR Instrument", in *Proc. 16th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'17)*, Barcelona, Spain, Oct. 2017, pp. 1815-1819. doi:10.18429/JACoW-ICALEPCS2017-THPHA167

[15] S. W. Lin *et al.*, "The Industrial Internet of Things Volume G1: Reference Architecture", *Industrial Internet Consortium*, 2017, https://www.iiconsortium.org/IIC_PUB_G1_V1.80_2017-01-31.pdf