ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA163

NXCALS - ARCHITECTURE AND CHALLENGES OF THE NEXT CERN ACCELERATOR LOGGING SERVICE

J. Wozniak†, C. Roderick, CERN, Geneva, Switzerland

Abstract

CERN's Accelerator Logging Service (CALS) is in production since 2003 and stores data from accelerator infrastructure and beam observation devices. Initially expecting 1TB / year, the Oracle based system has scaled to cope with 2.5TB / day coming from >2.3 million signals. It serves >1000 users making an average of 5 million extraction requests per day. Nevertheless, with a large data increase during LHC Run 2 the CALS system began to show its limits, particularly for supporting data analytics. In 2016 the Next CERN's Accelerator Logging Service (NXCALS) project was launched with the aim of replacing CALS from Run 3 onwards, with a scalable system using "Big Data" technologies. The NXCALS core is production-ready, based on open-source technologies such as Hadoop, HBase, Spark and Kafka. This paper will describe the NXCALS architecture and design choices, together with challenges faced while adopting these technologies. This includes: write / read performance when dealing with vast amounts of data from heterogenous data sources with strict latency requirements; how to extract, transform and load >1PB of data from CALS to NXCALS. NXCALS is not CERN-specific and can be relevant to other institutes facing similar challenges.

INTRODUCTION

The CERN Accelerator Logging Service (CALS) [1] was designed in 2001, has been in production since 2003 and stores data from all of CERN's accelerator infrastructure and beam observation devices. Initially expecting 1TB of data per year, the Oracle–based CALS system has scaled to cope with a throughput of 2.5TB per day coming from more than 2.3 million signals. It stores 1TB per day for the long-term and serves more than 1000 users from across CERN, who collectively submit 5 million extraction requests per day on average.

CALS is considered as being mission-critical and the go-to service when investigating problems with accelerator equipment or unexpected beam behaviour. Whenever a new system is commissioned, or a new mode of operation is established – there are inevitably subsequent requests to setup the corresponding data logging. This is magnified when new machines or facilities are commissioned.

Since the start of LHC, the scope of CALS and the demands placed upon it have evolved significantly and continue to do so. Figure 1 shows the restart of LHC at the end of 2009, there was an order of magnitude increase in data being logged – mainly coming from the new Quench Protection System. More recently, for the restart of LHC post LS1 (Long Shutdown 1) there was another almost order of magnitude increase in data being logged – mainly due to the need for more beam related data on a bunch-by-bunch and turn-by-turn basis.

Figure 1: CALS long-term daily storage evolution.

The trend of evolving logging needs has continued in recent years with the arrival of new facilities such as AWAKE and MEDICIS, as well as for the commissioning of new machines such as LINAC4.

The CALS system has scaled well in terms persisting acquired data and providing linear response times for data extractions. However, with basic accelerator operation reaching a high level of maturity, attention has turned to more complex analyses such as studying beam effects over longer periods of time. In other words, CALS has increasingly been subjected to extraction of much larger datasets over longer periods of time to support advanced data analytics. It is in this domain, during LHC Run 2, that the CALS system quickly started to show its limits. The CALS Oracle-based architecture is difficult to scale horizontally and does not perform particularly well for large data processing for signals with complex data structures. A key issue is that in order to perform moderately advanced analyses, the data first needs to be extracted and in certain scenarios this takes too long (e.g. for some use cases, it takes 12 hours to extract 24 hours of data). With increasing data volumes, more challenging analyses to be performed and a desire to quickly get answers to questions that can support operations – it was clear that actions needed to be taken.

In 2016, with all of the above knowledge and also aware that the Hi-Luminosity LHC (HL-LHC) machine is scheduled for commissioning in the not so distant future, the NXCALS project was launched with the aim of fully replacing CALS from LHC Run 3 (2021) onwards. The aim being to gain solid operational experience with NXCALS during several years and then have time to further adapt the system as needed during LS3 (Long Shutdown 3), while still ahead of HL-LHC commissioning.

NEED OF "BIG DATA" TECHNOLOGIES

In recent years, the so-called "Big Data" technology landscape has evolved significantly to support large scale

† jakub.wozniak@cern.ch, chris.roderick@cern.ch

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA163

data logging and analysis, opening up new possibilities to perform efficient analysis of large data sets.

To gain experience with these technologies and help choose a direction for NXCALS, a Proof of Concept (PoC) Logging System was developed in collaboration with the CERN IT-DB group in early 2016. The PoC was based on the open-source Apache Hadoop technology - as a replacement for the current Oracle-based CALS system. Both Apache Impala and Apache Spark were evaluated as query / analysis engines. Figure 2 shows the existing CERN use cases, both Impala and Spark outperformed Oracle when it comes to querying and extracting data above a certain number of records (i.e. longer time windows), with Spark in the lead.

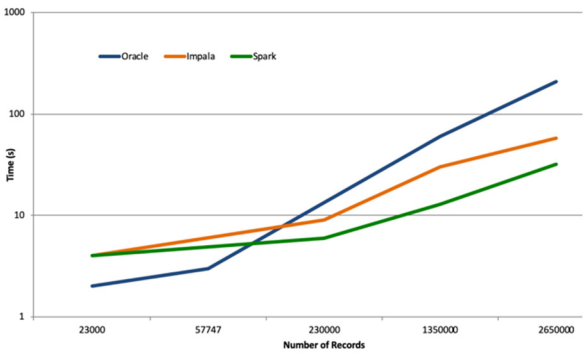

Figure 2: NXCALS PoC data extraction times (lower is better) in function of number of records extracted.

The PoC work clearly demonstrated the potential to replace the current system and improve performance and scalability for an overall lower hardware cost than the current CALS system (not considering the Oracle Licensing costs). The subsequent approval by CERN management led to the start of full-scale development of the new NXCALS system.

NXCALS ARCHITECTURE & TECHNOLOGIES

Development of the NXCALS system built on the experience gained during the aforementioned PoC project and led to a microservices architecture as shown in Fig. 3.

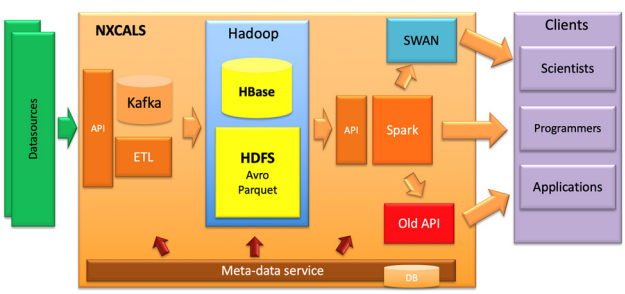

Figure 3: Overview of the NXCALS architecture.

The aim of this architecture is to be able to easily upgrade or replace different aspects of the system in the future as necessary, without being forced to put in place a completely new system. From a technology perspective, NXCALS is based on in-house developments combined with open-source software such as Hadoop (HDFS and HBase), Kafka, Spark, and Jupyter notebooks. The use of these technologies is described below.

Data Storage

Data is stored in the Hadoop layer. There is a need to have relatively fast access to the most recent data and also to be able to keep data over a long period of time in an efficient manner with good analysis and extraction performance. As such, NXCALS has a so-called Lambda architecture comprised of two main parts:

1. An HBase database which serves as a low-latency repository from which data of the last 36 hours can be extracted by users.
2. An HDFS file system which serves as the long-term storage of data.

Incoming data is frequently written into many small files which need to be compacted into larger files in order to optimise the data access performance. Therefore, using HBase for the most recent data gives time for an in-house developed software process to execute a daily de-duplication and compaction process of the data from the previous day.

Apache Parquet was chosen for HDFS storage as the columnar format is highly compressible, is compatible with all processing frameworks and supports efficient pruning of the columns to optimise data access.

NXCALS has been designed as a multitenant system allowing to store data from distinct source systems. Data is partitioned in files according to the policy for the source system and then by time. With such an approach it is possible to use NXCALS for archiving data from CERN systems which previously required their own dedicated data archive such as the Post-mortem and Alarm systems.

Data Acquisition

In-house developed data acquisition processes called "Data sources" acquire data via CMW (Controls Middleware) [2] and send it to NXCALS using the NXCALS data ingestion API. The Data sources software is written in Java using the Akka framework and is fully horizontally scalable and fault tolerant. This means to satisfy requirements for sending more data, more data sources processes can be configured to run, possibly on additional hardware. The data sources are configured in the central Controls Configuration Service, using the standard CCDE (Controls Configuration Data Editor) Web application [3] as shown in Fig. 4.

The CCDE allows users to easily configure what should be logged, in a tool that they undoubtedly already use for configuring other aspects of the control system. Furthermore, having an integrated configuration facilitates keeping coherency across the control system in case controls device coordinates are modified.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA163

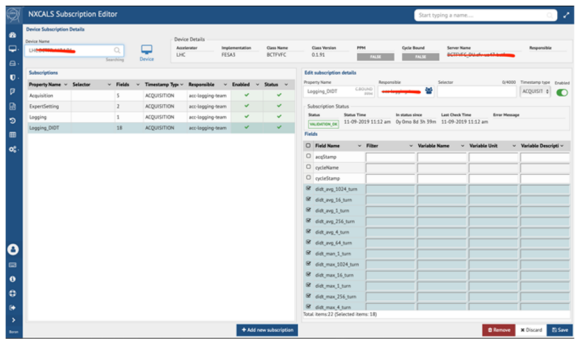

Figure 4: CCDE to configure NXCALS data acquisition.

Data Ingestion

The NXCALS data ingestion API used by the Data Sources, sends acquired data to Apache Kafka in Apache Avro format. Kafka was selected for being a highly reliable, high-throughput, low-latency platform for handling real-time data feeds. In NXCALS, data is stored on Kafka until it has been transferred into Hadoop by in-house ETL (Extract-Transform-Load) processes which periodically pull data from Kafka, convert it to Parquet format and write it in parallel to HBase (for immediate access) and HDFS temporary storage where it later gets compacted and deduplicated into the final storage. The need for deduplication comes from the fact that the Kafka ingestion chain only guarantees the "at least once" message delivery semantics and the messages can be repeated due to transmission errors. These technology and architectural choices facilitate high performance and reliability, where the aim is to guarantee zero data loss despite huge workloads.

Data Analysis and Extraction

Based on the experience gained during the PoC study and given the rich set of functionalities available – Apache Spark was selected to serve as the large-scale distributed data processing / analysis engine. In order to allow NXCALS to users extract data and/or perform data analyses, an NXCALS client API has been developed based on the Apache Spark APIs for Java and Python. The API includes NXCALS-specific extensions related to entity metadata. The API can be used as a standalone Python or Java bundle, or via SWAN.

SWAN (Service for Web based ANalysis) [4] is a CERN platform to perform interactive data analysis from the Web using Apache Jupyter notebooks. It has been configured to integrate with Spark and NXCALS (Fig. 5), allowing users to interact with NXCALS directly from the Web, executing, storing and sharing their analysis notebooks with others.

Meta-data Service

In order to properly manage the overall coherency of the system from data ingestion, to data storage and compaction, to data analysis and extraction – an in-house developed Meta-data service is employed to manage the meta-data describing:

- the entities for which timeseries data can be stored and their evolution over time in terms of internal structure,
- the location of corresponding data (files etc.),
- the mapping of higher-level concepts such as "variables" and the corresponding entity fields which hold the data.

The Meta-data service is written in Java and is backed by a relational database with no vendor-specifics constraints.

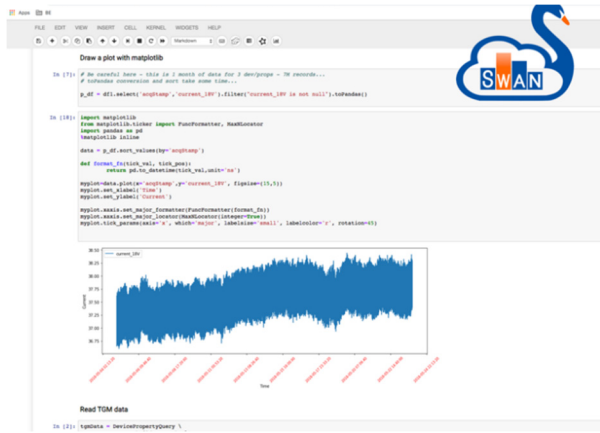

Figure 5: SWAN Web analysis using Jupyter notebooks.

PERFORMANCE AND SCALABILITY

The core technologies used in NXCALS are based on the concept of "horizontal scalability", which essentially means the ability to increase performance by adding more resources to the underlying infrastructure. From this perspective, the NXCALS system has the potential to adapt to the required performance needs of the future, provided sufficient resources can be financed and that sufficient physical hosting capacity is available.

In terms of potential data analysis performance, a key difference in NXCALS with respect to the existing CALS system is a change in paradigm. With the CALS system, users first extracted the data and then performed the analysis on their local machines. With the NXCALS system, users seeking high levels of data analysis performance need to submit their analysis algorithms to be executed on the NXCALS cluster, using Spark, and then retrieve the results. Following this change in paradigm has already shown cases where analysis times can be reduced from several days to under an hour.

CURRENT STATUS

In April 2018, the NXCALS service was moved to a new production computing cluster located in the CERN Computing Centre. From this point onwards, NXCALS has been considered as a production system in terms of core functionality, user support, monitoring and ensuring availability.

Figure 6 shows an overview of the current NXCALS production hardware with the core consisting 20 machines having 960 Cores and 8 TB RAM.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA163

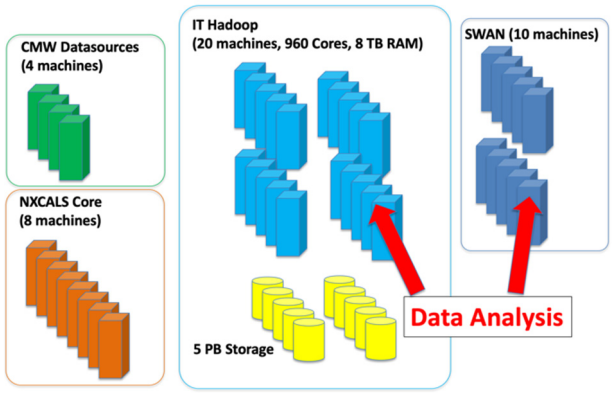

Figure 6: Overview of the current NXCALS hardware.

The CMW data ingestion processes are fully operational and logging data in parallel to both CALS and NXCALS systems. Some recent use cases such as LHC Diamond BLM analysis at IP7 & 16L2 (20-50 GB/day) could not be satisfied by the CALS system, therefore such data is only being sent to NXCALS from where it was successfully analysed throughout 2018.

A growing number of users are porting their extraction software to use the Apache Spark based data extraction API and / or make use of SWAN. To help such users (as well as new ones), a big effort has been made on documentation – using the MKDocs software to have relevant code examples that are built and tested together with the NXCALS system to ensure they are up-to-date and fully functioning.

In addition, work on adapting the current CALS data extraction client API to pull data from NXCALS is at an advanced stage. The aim is to enable existing CALS clients to move to NXCALS without forcing them to re-write their code and learn Spark.

A key activity, somewhat aside from the new NXCALS system itself, is the migration of data from CALS to the new system. This has advanced very well over the summer, with the majority of data migrated (more than 32E12 records so far). The team continues covering special cases which require further custom developments, for example the treatment of LHC Fills and Beam modes.

MAJOR CHALLENGES

Replacing any large or mission-critical system will always present technical and strategic challenges to be overcome. Some of the major challenges encountered so far for NXCALS are described below.

Infrastructure

The very nature of Big Data technology requires a lot of machines to ensure maximum performance via parallel processing, as well as satisfactory data redundancy to ensure high service availability. These needs are multiplied once the need for distinct environments for development, testing, staging and production is factored in. Overall this represents a financial challenge in terms of purchasing and maintaining the infrastructure. There is also the logistics challenge when it comes to installing and running the hardware with suitable cooling and resilient powering strategies.

Evolving Technologies

Unlike CALS which was based on the well-established Oracle database platform, NXCALS is based on relatively new technologies which continue to evolve. On one hand this is very positive as it shows there is a large community pushing to improve the already impressive software and there is surely much untapped potential. However, on the other hand this presents a major challenge. For example, it is not always easy to find well established documentation or examples on how to best deal with demanding scenarios that go beyond the basic applications of the Big Data technologies. As another example, some problems encountered simply need to wait for community consensus, developments and subsequent releases.

DevOps, Quality and Monitoring

Managing a large-scale microservice architecture like NXCALS is extremely challenging. It is essential to have strong DevOps processes with well-thought, automated development pipelines enforcing strict quality assurance and ensuring reproducible deployments. An enormous effort was made on this starting with designing Continuous Integration / Continuous Delivery (CI/CD) pipelines which are implemented based on Jenkins and Ansible integrating with Gitlab. Multiple environments are configured to support development, testing, staging and production – all of which can be updated with a single click. Concerning code quality, the NXCALS team have a clear development process established based on GitLab Merge Requests, backed with a policy to have at least 70% code coverage. SonarSource's SonarQube software has been fully integrated into the NXCALS CI/CD, to automatically detect bugs and fail new builds that will result in a reduction in quality (Fig. 7).

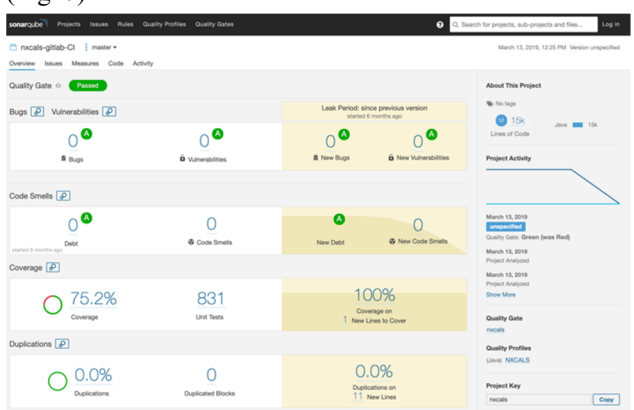

Figure 7: NXCALS code quality analysis via SonarQube.

Due to the large number of components involved, it is also a challenge to ensure that everything continues to work overtime. To tackle this, NXCALS is instrumented to expose different types of health metrics which are monitored using Prometheus/Alertmanager software that in turn

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA163

generates e-mail alerts in case of problems. Furthermore, all metrics are constantly visible on Grafana dashboards.

Software Compatibility

A challenge that is not so obvious from the outset, is ensuring compatibility between the versions of the many different software elements such as Hadoop, HBase, Spark, Python, Java etc. Sometimes a new version of one element is released that provides an important fix or new feature, however it may be incompatible with other parts of the system – forcing further upgrades. Here, the dedicated deployment environments and automated testing play a significant role. For other cases, like the upgrade to Java 11, it is necessary to wait for community developments and subsequent releases of new software versions.

Data Writing Performance

Initial tests for writing Industrial Controls (WinCCOA) data indicated the initial production load will be around 250K recs/sec. In our environment, HBase could not scale to this load out of the box and significant efforts were required to tune the cluster and the NXCALS system to achieve stable write and read performance. New data generators were created that can scale the load and simulate various types of data (CMW/WinCCOA). Thanks to the advanced monitoring in place, it was possible to observe the effects of changing configurations, draw conclusions and adapt. HBase performance tuning is not well documented and was extremely challenging, with the team resorting to reading HBase source code to fully understand what was happening and get ideas on how to improve. A number of issues with the ETL configurations and with the Hadoop infrastructure were identified and addressed leading to the required performance. It was also proven that the performance can indeed scale horizontally by adding machines.

Data Reading Performance

Users reported described regular analyses that took 1 hour to process 1 day of numeric data in CALS, that now return a week of data instantaneously in NXCALS. However, for other cases, performance is not yet satisfactory. A lot of efforts were made to tune extraction performance by adapting the daily data compaction process to arrange data in an optimal way with respect to data extraction patterns resulting in a ~10x improvement in extraction times.

Extraction times are linearly dependent on storage file sizes (partitions) plus data structures, compression ratios, selected columns, parallelism, etc. With a lot of scope to tune NXCALS, this will be major focus in the next years.

Data Migration

The migration of > 1PB historical data from CALS for some 2.6 million signals, is a project apart and an enormous challenge. In-house developed Java software reads historical data from CALS, transforms it to NXCALS representations and writes it to NXCALS using the same data ingestion mechanism used to receive live data. The act of migrating the data achieves the critical objective of preserving the historical data and at the same time provides an excellent test of the stability and performance of NXCALS.

Most historical data have been migrated at this time and along the way, many performance and stability issues were identified and solved – dealing with peak transfer rates exceeding 2.1°6 recs/sec. The migration processes were executed on ~100 CERN Openstack machines (virtual computers). A dedicated Migration Verifier application has been developed to identify any holes in the migrated data (e.g. due to infrastructure failures) and will be further extended to automatically re-migrate data in such cases.

NEXT STEPS AND FUTURE OUTLOOK

The aim is to switch off the CALS system by the end of 2020. In the meantime, the focus is on completion of historical data migration, data extraction API developments and providing a replacement for TIMBER. TIMBER is the general GUI used to visualise and extract logged data currently written in Java Swing. This will be re-developed using our standard Web technologies (Angular, Java, Spring) to interact with NXCALS instead of CALS.

Once all of the above is in place, key areas to focus will be further tuning the system performance and enhancing NXCALS as a platform for machine learning applications.

SUMMARY

NXCALS is the next generation of CERN's Accelerator Logging Service. The new system is at an advanced stage of development and already serving challenging use cases in a productional capacity. The microservice-based architecture and horizontally scalable technologies employed give a solid basis from which to evolve in the future to support the emerging data logging and analysis requirements for the entire CERN complex and the future High-Luminosity LHC machine. Meanwhile, the use of open source software and a non-CERN-specific modular design mean that NXCALS could be ported to fulfil similar data logging and analysis requirements in other institutes and domains.

REFERENCES

[1] C. Roderick *et al.*, "The CERN Accelerator Logging Service - 10 years in operation: A look at the past, present, and future", in *Proc. ICALEPCS'13, San Francisco, CA, USA*, Oct. 2013, paper TUPPC028

[2] W. Sliwinski, J. Lauener, "How to design & implement a modern communication middleware based on ZeroMQ", in *Proc. ICALEPCS'17, Barcelona, Spain*, Oct. 2017, doi:10.18429/JACoW-ICALEPCS2017-MOBPL05

[3] L. Burdzanowski *et al.*, "CERN Controls Configuration Service – A challenge in usability", in *Proc. ICALEPCS'17, Barcelona, Spain*, Oct. 2017, doi:10.18429/JACoW-ICALEPCS2017-TUBPL01

[4] SWAN, http://swan.web.cern.ch