ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-TUSDSC07

WEB DASHBOARDS FOR CERN RADIATION AND ENVIRONMENTAL PROTECTION MONITORING

Adrien Ledeul*, Alexandru Savulescu, Gustavo Segura
CERN, Geneva, Switzerland

Abstract

CERN has developed and operates a SCADA system for radiation and environmental monitoring, which is used by many users with different needs and profiles. To provide tailored access to this control system's data, the CERN's Occupational Health & Safety and Environmental Protection (HSE) Unit has developed a web-based dashboard editor that allows users to create custom dashboards for data analysis.

In this paper, we present a technology stack comprising Spring Boot, React, Apache Kafka, WebSockets, and WebGL that provides a powerful tool for a web-based presentation layer for the SCADA system. This stack leverages WebSocket for near-real-time communication between the web browser and the server. Additionally, it provides high-performant, reliable, and scalable data delivery using low-latency data streaming with Apache Kafka. Furthermore, it takes advantage of the GPU's power with WebGL for data visualization.

This web-based dashboard editor and the technology stack provide a faster, more integrated, and accessible solution for building custom dashboards and analyzing data.

INTRODUCTION

CERN Radiation and Environmental Monitoring

The highest priority of CERN is to ensure that its research infrastructure, whose core assets are particle accelerators and experimental areas, operates safely for the workers and for the public, while minimizing the organization's activities environmental impact. To that regard, CERN has established and implemented a Safety Policy where radiation safety and environment protection are central.

The implementation of the safety policy includes the setup and operation of a radiation and environment monitoring infrastructure that provides a real-time view of the effects of the operation of CERN. This monitoring infrastructure helps the organization to early detect deviations and gives the possibility to steer the processes (accelerators or operational parameters) to avoid approaching safety boundaries. The infrastructure also allows recording radiation and environment measurements, which are reported to the authorities under the form of comprehensive regulatory reports and are also published in public environment reports [1].

The monitoring infrastructure is composed of the extensive network of instruments that continuously measure radiation and conventional environment parameters in accelerators, experimental areas and in public CERN places across the campus, but also within closed perimeters outside CERN fences.

Radiation and Environment Monitoring Unified Supervision

The Radiation and Environment Monitoring Unified Supervision (REMUS) [2] is the Supervisory Control and Data Acquisition (SCADA) system that orchestrates the instrumentation. REMUS is built on WinCC Open Architecture (WinCC OA) [3]. It collects measurements from the instrumentation and immediately makes them available to the CERN control rooms operators and to experts in charge of radiation protection and environment monitoring. REMUS also supervises the correct operation of the monitoring infrastructure by detecting faulty instruments, network communication issues, power failure, and faults in its internal software components. This is supported by the fully redundant architecture of the REMUS system.

REMUS supervises an infrastructure of 2 800 instruments. These instruments are very heterogeneous with up to 87 different types of devices. REMUS contains 900 000 SCADA tags, manages 125 000 distinct alarm tags and handles a throughput of 25 000 input/output operations per second. REMUS archives 80 billion of measured values per year.

The big amount of data generated and recorded by REMUS are made available to its more than 250 active users in real-time SCADA client software and in long-term data stores namely Oracle and the Next CERN Accelerator Logging Service (NXCALS) [4].

REMUS accommodates a wide range of professional profiles (accelerator or experiment operators, radiation protection specialist, environmental protection specialist, physicist, instrumentation engineers and technicians, SCADA software engineers, etc.), with different needs and functional expectations from REMUS. For these users, REMUS provides software tools adapted or adaptable to their specific needs, which furthermore can be tailored and adaptable down to each individual user. However, the need for adaptive user interfaces with wider customization capabilities and openness grows constantly. In order to respond to these increasing user needs, REMUS has implemented and deployed a modern web dashboard editor.

RATIONALE

This section introduces the rationale behind the creation of web dashboards, and solutions that were considered to

* adrien.ledeul@cern.ch

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-TUSDSC07

implement it.

Needs for Web Dashboards

Light Clients Hundreds of users access REMUS SCADA client software every week, averaging 60 concurrent client connections during working days. SCADA clients are relatively complex and resource-heavy in displaying real-time data, which gets constantly updated via server connections. These connections are fed by a central SCADA message broker and incur a significant load on REMUS system. REMUS is based on WinCC OA, and therefore it inherits from the limitations of WinCC OA whose message broker ("event manager" in WinCC OA terminology), is single threaded. The same message broker handles both message queues for the instrumentation layer and message queues for the user interfaces. Since most users of the system only require read-only access to limited sets of data, it is desired to provide a solution that will decrease the strain on the message broker.

Accessibility Even though REMUS Graphical User Interfaces are accessible through both CERN Common Console Manager (CCM) system and a cluster of Windows terminal servers, it remains inaccessible outside CERN without using a gateway. This limitation was emphasized specially since the recent COVID crisis and consequent increase of teleworking practices that drive REMUS users to connect the SCADA from outside CERN. We observe that, in most cases, remote users access REMUS for gathering specific information or dealing with very specific part of the full system and therefore connecting a terminal server or another equivalent solution to run a full remote SCADA client software is not the most convenient solution. A lighter solution better suits these users.

Easiness of Use Some of REMUS users are not necessarily familiar with the usage of a complex SCADA system such as REMUS, but still need to be trained, even if they only require limited read-only access. This is an issue specially for occasional users that do not interact with SCADAs on a regular basis.

Temporary Installations During temporary monitoring campaign or commissioning of a beam line / accelerator / experiment, it is very useful to have a view that is restricted to the corresponding monitors in place. REMUS can facilitate creating such views by allowing the design of custom user interfaces [5], however it entails some high privilege access and requires training.

Mobile Access Considering CERN's wide geographical distribution, having a system accessible from mobile phones and tablets is a major advantage.

Wrap-Up All the reason mentioned above drives us to develop user-configurable dashboards as a new paradigm in user interfaces for SCADA. These interfaces are designed to be lighter, easier to access via a web browser, more user-friendly and last but not least they provide a facade over the SCADA software and its specific requirements.

Charts and Their Requirements

When it comes to data interpretation, charts play an essential role in SCADA systems, leveraging real-time and historical data for purposes ranging from real-time control and decision-making to post-mortem analysis or even forecasting. Extrapolating from the capabilities of our legacy Java Swing data visualization tool, the Environment and Radiation Graphic Observer (ERGO), we have gathered the following technical and functional chart requirements:

- Refresh time up to 1hz.
- Near-real-time data handling.
- A single chart shall accommodate up to 20 data series supporting roughly 500k data points.
- Embeddable in responsive web-apps.
- Feature-rich with minimal integration effort.
- Out-of-the-box support for (at least):
  - Multiple line charts with logarithmic scale and multiple Y axis.
  - Heatmap charts.
  - Boxplot charts.
  - Pie charts.

Candidate Technologies and Chosen Solution

Considering the need of providing web dashboards and looking at the identified chart requirements, we have explored three possible solutions.

Generic Visualization Web Application Web applications such as Grafana [6] or Chronograf [7] or similar solutions: This solution has the advantage of being quickly exploitable for production, but yields several caveats:

- It is not easily editable for end-users, who need to know the structure of the data and at least one query language.
- Custom plugins are necessary in order to merge different data sources into one.
- Not well adapted to mobile display, although latest versions have brought major progress in this regard.
- No near-real-time data at the time the decision was made; full data-sets had to be reloaded on refresh (this feature has been introduced later by 8.0 Grafana Live).

Open Source Chart Libraries We have studied a few well-known and widely used open-source data visualization libraries, such as D3.js or Chart.js.

D3.js turns out to be a very powerful library that allows the creation of custom charts and visualizations, but it comes with great complexity and a steep learning curve. Consequently, it would have considerably increased the total cost of ownership for implementing and maintaining all our needs.

Chart.js is an easier-to-use library that provides a set of predefined charts and related features. However there is

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-TUSDSC07

less customization available and it doesn't match our chart requirements, proving to be non-suitable for complex data visualizations or large data sets.

Commercial Chart Libraries Probing popular commercial chart libraries, HighCharts [8] stood out as the most sensible solution for our use case. It fulfilled all the requirements mentioned above, and was already used at CERN by several projects. Additionally, it has an extensive user base and support, and fits well in our pre-existing React front-end using a React wrapper [9]. The choice of this technology enabled our team to integrate, in matter of weeks, powerful charts widgets for the dashboards.

ARCHITECTURE SUMMARY

A summary of the architecture is shown in Fig. 1.

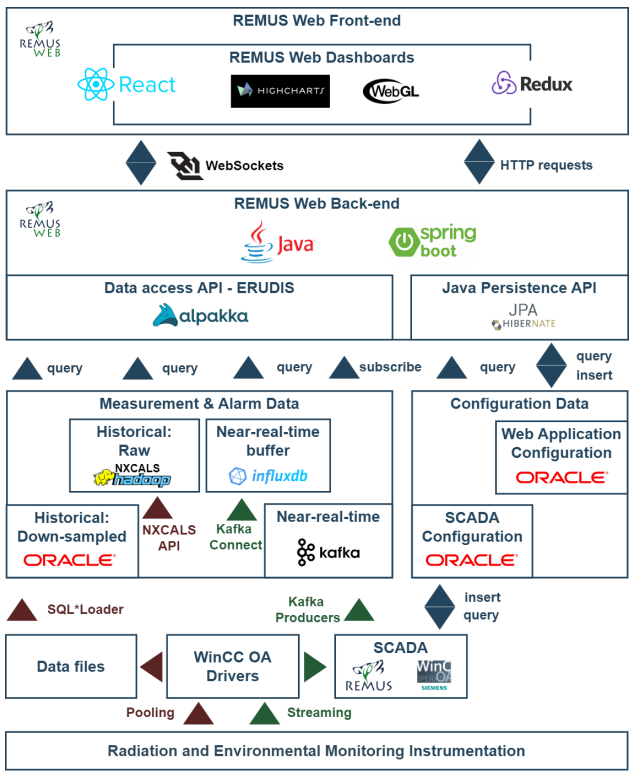

Figure 1: Architecture of REMUS Web Dashboards. From the bottom to top: Field and Control layers, SCADA layer, Data stores, REMUS Web Back-end and finally REMUS Web Front-end displaying the Dashboards.

DATA SOURCES

The information displayable on the dashboards are of two types:

- Configuration database exploration, such as entities meta-data, and parameters applied to the different instruments over time.
- Measurement and Alarm data.

While the configuration database is solely fetched from Oracle storage, Measurement and Alarm data leverage four data sources:

- Oracle: Stores long term, down-sampled measurement data . Ensures low latency access to data, thanks to the partitioning and down-sampling.
- NXCALS (CERN Hadoop-based solution): Stores long term, raw measurement data. Ensures high resolution access to the data.
- Apache Kafka [10]: Provides subscribable near-real-time data coming directly from the SCADA layer [11].
- InfluxDB: Fed by Kafka via Kafka Connect [12], it provides one week buffer of the most recent data.

ERUDIS Library

The Environment and Radiation Unified Data Integration Service (ERUDIS) library, developed by our team, serves as an API for data access. It has been designed to simplify the interaction with the various data sources listed above, providing a high-level abstraction facade over the different data storage technologies [13]. It is used in all the REMUS data visualization applications, including the dashboards. The API's architecture is rooted in Akka [14], an actor concurrency model framework, and utilizes the Alpakka [15] library, purpose-built for data stream handling. These tools not only facilitate the creation of interfaces that unify disparate data sources but also offer capabilities for interactive data stream management. This includes features like buffering, advanced error handling, and built-in redundancy mechanisms.

BACK-END

Language and Frameworks

The dashboard back-end capitalizes on the previously developed web application back-end of REMUS Web. It is developed in Java 17 and Spring Boot [16], which is an open-source Java web framework. Spring Boot extends the capabilities of the well-established Spring Framework [17], making it easier to configure standalone Spring applications. It also offers a structured approach to leveraging Spring and third-party libraries, simplifying the development process. In the context of the dashboard, the back-end mainly consists of REST API end-points, Data Transfer Objects (DTO) mapped to the configuration database entities and dashboard configurations, and an interface between the previously mentioned ERUDIS Library and WebSockets that provides data to the front-end.

WebSockets

REMUS Web dashboards leverage the WebSocket protocol [18], a powerful technology that enables the creation of full-duplex, near-real-time communication channels between clients and servers. Unlike traditional HTTP requests, which are essentially one-way, request-response interactions, WebSocket connections remain open, facilitating continuous, bidirectional communication. This

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-TUSDSC07

innovation is particularly advantageous for REMUS Web dashboards as it ensures that data generated by the SCADA system can be seamlessly streamed to the client with minimal latency.

Combined with a suitable charting library, this allow dashboards' widgets to display near-real-time data with particularly low latency, since only new data points are added to the charts.

FRONT-END

React, React-Grid-Layout and Redux

The REMUS Web dashboards are built on top of the same Javascript library as the rest of the application, React [19]. This technical choice over other concurrent library and frameworks, namely Vue.js [20] and Angular [21], was taken in 2018, before this web dashboard project started, at the level of our team. It was mainly motivated by relatively easier learning curve, growing popularity and wide ranges of additional third party libraries available.

Dashboards pages do make use of such an additional library, react-grid-layout [22]. This library enables the creation of flexible and responsive grid layouts in React applications. It permits end-users to arrange and manage components in a grid format, allowing for dynamic resizing, repositioning, and customization of grid items. This library also empowers the well known Grafana web application, ever since they switched their front-end from Angular to React back in 2018 [23].

In React, basic state management features are sufficient for straightforward projects. However, as applications become more complex and the requirement to share states among various components arises, a more advanced state management solution becomes essential. Redux [24] is the prevalent choice and is what we have employed in REMUS Web.

REMUS Web also relies on the react-query [25] library, a prominent data-fetching solution for React applications. The library enables individual components to fetch their own data in parallel. React-query offers built-in caching mechanisms and supports background updates, enhancing overall application performance.

HighCharts

HighCharts library is used to display all the dashboards' widgets containing measurement data. Such data can be displayed either as multiple line charts, or as box plots, permitting a quick visualization of averages, quartiles and outliers over various data series. HighCharts is particularly performant when its Boost module is activated instead of its default SVG renderer, as it enables GPU acceleration using WebGL [26]. This feature enables the rendering of millions of data points in a matter of seconds.

REMUS WEB DASHBOARDS FEATURES

As of today, REMUS Web dashboards provide the following widgets:

- Trends: Multiple measurement data series line charts.
- Measurements Statistics: Box plots or table of metrics of the measurement data series.
- Snapshots: List or aggregations of snapshot data (spectrum or logically-linked measurements).
- Alarms: Near-real-time and read-only replication of the SCADA alarm screen.
- Parameters: List of current parameters of an instrument.
- Parameters Statistics: Box plots or list of current parameters of multiple instruments.
- Information: Meta-data of an instrument.
- Text: Markdown-enabled text area.

Access rights can be managed per dashboard by group of users. Dashboards can be shared by link or via QR codes.

In addition, there are three display modes: standard, full-screen and edition providing that the permission is granted.

Dashboards are built in a Progressive Web Application (PWA) [27], therefore usable from any platform using a standard-compliant browser, including desktop, tablets and mobile devices.

The end result, in visualisation mode, is shown in Fig. 2.

CONCLUSION

Modern technologies and remote working trends have paved the way for novel methods of accessing, interacting and analyzing SCADA data. Similar data pipelines and data visualization tools are getting adopted more and more by institutions [28] [29], and as well by many industries that rely on big data. REMUS users now have the power to design and share web dashboards tailored to their needs, all while abstracting away from the underlying SCADA software and its specificity. The benefits are many-fold, ranging from cutting down on user training requirements to decreasing the load on the SCADA system. By using modern popular technologies, we built a future-proof web-based presentation layer that is complementary to the SCADA system, one that is high-performant, reliable and scalable.

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, Control and Information Technology groups for their fundamental contribution to the success of the REMUS project.

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-TUSDSC07

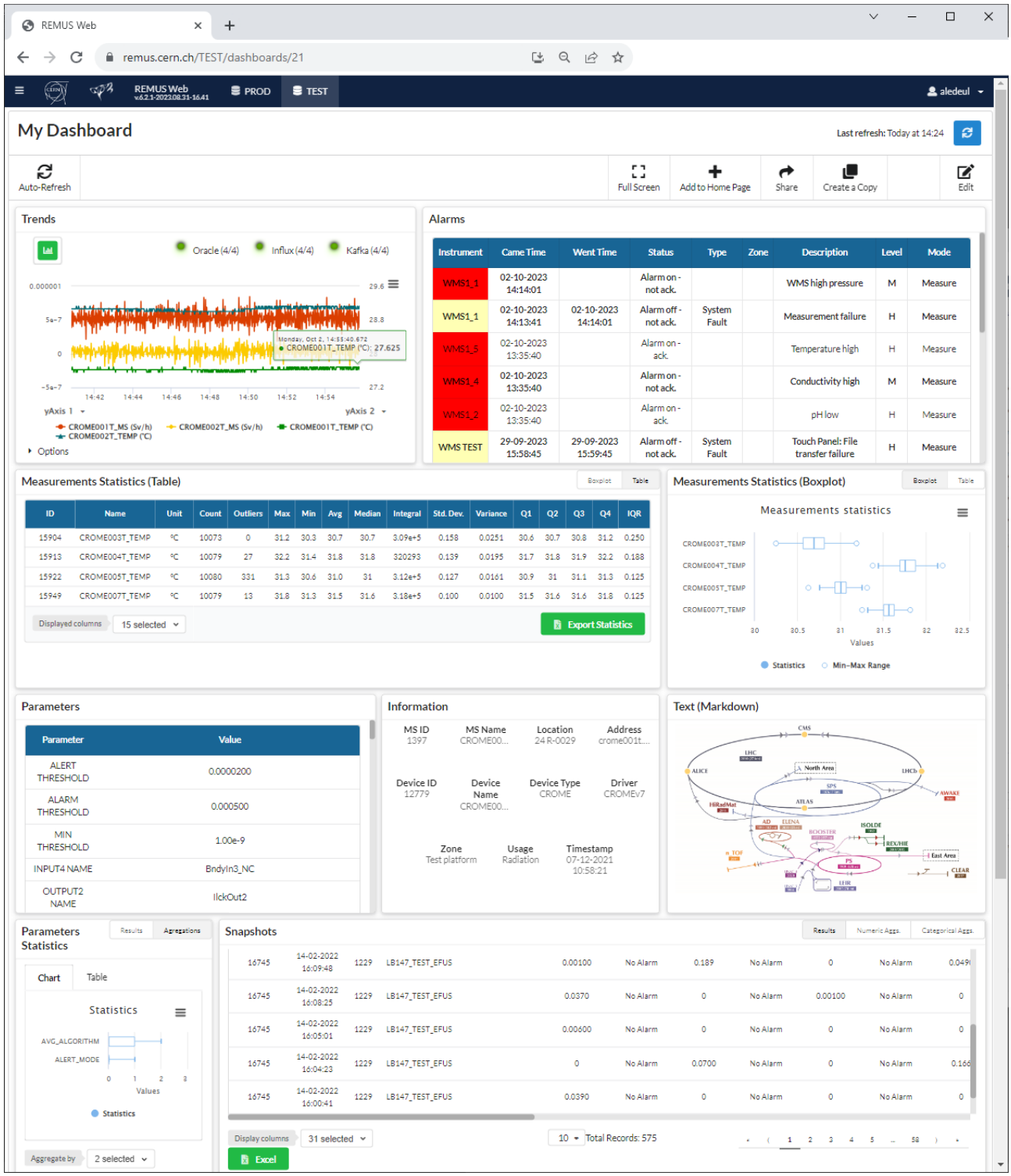

Figure 2: Screenshot of a dashboard show-casing all available widget types. Data shown in this screenshot are coming from a test bench and are not actual operational data.

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-TUSDSC07

REFERENCES

[1] CERN public environment report, https://hse.cern/environment-report

[2] A. Ledeul, A. Savulescu, G. Segura, B. Styczen, and D. V. Rivera, "CERN Supervision, Control and Data Acquisition System for Radiation and Environmental Protection", in *Proc. PCaPAC'18*, Hsinchu City, Taiwan, Oct. 2018, pp. 248–252. doi:10.18429/JACoW-PCaPAC2018-FRCC3

[3] Wincc oa, https://www.winccoa.com

[4] J. P. Wozniak and C. Roderick, "NXCALS - Architecture and Challenges of the Next CERN Accelerator Logging Service", in *Proc. ICALEPCS'19*, New York, NY, USA, 2020, pp. 1465–1469. doi:10.18429/JACoW-ICALEPCS2019-WEPHA163

[5] G. Segura, A. Ledeul, A. Savulescu, B. Styczen, and D. V. Rivera, "Innovative Graphical User Interfaces Development: Give the Power Back to Users", in *Proc. PCaPAC'18*, Hsinchu City, Taiwan, Oct. 2018, pp. 44–46. doi:10.18429/JACoW-PCaPAC2018-WEP07

[6] Grafana, https://grafana.com

[7] Chronograf, https://www.influxdata.com/time-series-platform/chronograf

[8] Highcharts, https://www.highcharts.com

[9] Github repository for highcharts-react, https://github.com/highcharts/highcharts-react

[10] J. Kreps, N. Narkhede, J. Rao, *et al.*, "Kafka: A distributed messaging system for log processing", in *Proceedings of the NetDB*, vol. 11, 2011, pp. 1–7.

[11] A. Ledeul, A. Savulescu, G. Segura, and B. Styczen, "Data Streaming With Apache Kafka for CERN Supervision, Control and Data Acquisition System for Radiation and Environmental Protection", in *Proc. ICALEPCS'19*, New York, NY, USA, 2020, pp. 147–151. doi:10.18429/JACoW-ICALEPCS2019-MOMPL010

[12] Kafka connect, https://www.confluent.io/product/connectors

[13] A. Ledeul, C. C. Chiriac, G. Segura, J. Sznajd, and G. de la Cruz, "Data-Centric Web Infrastructure for CERN Radiation and Environmental Protection Monitoring", in *Proc. ICALEPCS'21*, Shanghai, China, 2022, pp. 261–266. doi:10.18429/JACoW-ICALEPCS2021-MOPV045

[14] D. Wyatt, *Akka concurrency*. Artima Incorporation, 2013.

[15] Alpakka, https://doc.akka.io/docs/alpakka

[16] Spring boot, https://spring.io/projects/spring-boot

[17] Spring framework, https://spring.io/projects/spring-framework

[18] V. Pimentel and B. G. Nickerson, "Communicating and displaying real-time data with websocket", *IEEE Internet Comput.*, vol. 16, no. 4, pp. 45–53, 2012. doi:10.1109/MIC.2012.64

[19] React, https://reactjs.org

[20] Vue.js, https://vuejs.org

[21] Angular, https://angular.io

[22] Github repository for react-grid-layout, https://github.com/react-grid-layout/react-grid-layout

[23] An inside look at how react powers grafana's frontend, https://grafana.com/blog/2023/06/28/an-inside-look-at-how-react-powers-grafanas-frontend

[24] Redux, https://redux.js.org

[25] React-query library, https://tanstack.com/query/latest

[26] Highcharts boost module api, https://api.highcharts.com/highcharts/boost

[27] Progressive web apps, https://web.dev/progressive-web-apps

[28] S. Khan, E. Rydow, S. Etemaditajbakhsh, K. Adamek, and W. Armour, "Web performance evaluation of high volume streaming data visualization", *IEEE Access*, vol. 11, pp. 15 623–15 636, 2023. doi:10.1109/ACCESS.2023.3245043

[29] G. Vino, D. Elia, V. C. Barroso, and A. Wegrzynek, "A Monitoring System for the New ALICE O2 Farm", in *Proc. ICALEPCS'19*, New York, NY, USA, 2020, pp. 835–840. doi:10.18429/JACoW-ICALEPCS2019-TUDPP01