ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-WE3BC004

IMPROVING OBSERVABILITY OF THE SCADA SYSTEMS USING ELASTIC APM, REACTIVE STREAMS AND ASYNCHRONOUS COMMUNICATION*

I. Khokhriakov†, V. Mazalova, O. Merkulova, DESY, Hamburg, Germany

Abstract

As modern control systems grow in complexity, ensuring observability and traceability becomes more challenging. To meet this challenge, we present a novel solution that seamlessly integrates with multiple SCADA frameworks to provide end-to-end visibility into complex system interactions. Our solution utilizes Elastic APM to monitor and trace the performance of system components, allowing for real-time analysis and diagnosis of issues. In addition, our solution is built using reactive design principles and asynchronous communication, enabling it to scale to meet the demands of large, distributed systems. This paper will describe our approach and discuss how it can be applied to various use cases, including particle accelerators and other scientific facilities. We will also discuss the benefits of our solution, such as improved system observability and traceability, reduced downtime, and better resource allocation. We believe that our approach represents a significant step forward in the development of modern control systems, and we look forward to sharing our work with the community at ICALEPCS 2023.

INTRODUCTION

With the advancement of technology, modern scientific instruments have evolved to exhibit remarkable levels of complexity. This evolution is characterized by complicated architectures, multifaceted functionalities, and the integration of diverse subsystems and technologies. Alongside this development, there is a notable increase in data acquisition rates and volumes which are boosted by the pursuit of higher precision and the exploration of broader scientific phenomena [1, 2, 3, 4].

This increase in data acquisition creates a challenge, as instruments generate vast amounts of data at an accelerated pace, making it necessary to create advanced solutions for efficient data handling, processing, and storage. The complex nature of these instruments, coupled with big and high-velocity data they produce, underscores the imperative for enhanced observability and traceability within Supervisory Control and Data Acquisition (SCADA) systems. Addressing these challenges is crucial for ensuring the reliability and efficacy of scientific experiments and for unlocking new approaches in science.

Navigating the complexity of modern scientific instruments and managing data they produce demands a focus on the observability of SCADA systems. Observability [5, 6], in this context, is not just a luxury but a critical requirement. It enables operators to gain deep insights into system performance, identify anomalies in real-time, and troubleshoot issues promptly, thereby ensuring reliability of the entire control system.

Given the vital role of SCADA systems in managing and monitoring complex processes, a lack of observability can lead to inefficiencies, operational disruptions, and compromised data quality. Furthermore, in an era when scientific experiments are rapidly progressing, the ability to observe and understand the inner workings of SCADA systems is crucial for optimizing system performance and adapting to evolving requirements.

Thus, enhancing the observability of SCADA systems emerges as an essential undertaking, driving the development of innovative solutions and methodologies to meet this imperative need. This paper introduces a novel approach, leveraging Elastic APM, reactive design principles, and asynchronous communication, to significantly improve the observability and traceability of SCADA systems across various scientific facilities [7, 8].

In the subsequent sections of this paper, we are going to look into two distinctive approaches to incorporating observability into SCADA systems, addressing the unique challenges posed by the increasing complexity and high data acquisition rates of modern scientific instruments.

1. **Dedicated Microservice Approach**: The first approach we explore involves the development and implementation of a dedicated microservice designed to listen to messages as they pass through the system. By forming transactions from these messages, this method aims to provide a granular view of system interactions, thereby enhancing the ability to monitor, trace, and diagnose potential issues within the SCADA environment.
2. **Integrating Agents into Software Components**: The second approach takes a more integrated route, embedding agents directly into corresponding software components of the SCADA system. This method seeks to ensure a seamless and comprehensive monitoring capability, offering real-time insights and quick responsiveness to any anomalies or system performance deviations.

* work supported by the consortium DAPHNE4NFDI
† igor.khokhriakov@desy.de

These approaches are well examined and presented in their application, benefits, challenges, and the potential they hold in enhancing the observability of SCADA systems in various scientific settings. The exploration of these methods provides a foundation for discussions and considerations on the evolution of SCADA system observability.

MODERN OBSERVABILITY TOOLS: A GENERAL OVERVIEW

As the complexity of systems and the scale of data continue to expand, the landscape of modern observability tools has evolved to meet the emerging needs and challenges. These tools are essential for monitoring, understanding, and optimizing the performance of complex and distributed systems, including SCADA systems in scientific environments. Below, we provide a general overview of the characteristics, types, and applications of contemporary observability tools.

Modern observability tools exhibit a range of features designed to provide deep insights into system performance and behavior. Some key characteristics include [9]:

- **Real-Time Monitoring**: These tools offer the ability to monitor system performance and metrics in real-time, enabling immediate detection and response to issues.
- **High Cardinality**: They can handle a high volume of unique metadata and metrics, which is crucial for understanding the behavior of distributed and microservices-based architectures.
- **Anomaly Detection**: Advanced algorithms and machine learning models are employed for detecting anomalies and irregularities in system behavior.
- **Log and Trace Analytics**: Integrating log analytics and distributed tracing, these tools provide detailed information on system interactions and transactions.
- **Visualization and Dashboarding**: Comprehensive visualization options and customizable dashboards allow for easy interpretation and analysis of the collected data.
- **Scalability and Adaptability**: Modern tools are designed to scale with the growth of the system and adapt to varying requirements and configurations.

The ecosystem of modern observability tools can be categorized into several types, each serving specific purposes and use cases:

- **Application Performance Monitoring (APM)**: Tools such as Elastic APM and New Relic provide insights into application performance, error tracking, and user experience.
- **Infrastructure Monitoring**: Tools like Prometheus and Grafana specialize in monitoring the health and performance of the underlying infrastructure, including servers, containers, and networks.
- **Log Analytics**: Solutions like ELK Stack (Elasticsearch, Logstash, Kibana) and Splunk are focused on collecting, analyzing, and visualizing log data for operational intelligence.
- **Distributed Tracing**: Tools like Jaeger and Zipkin are designed to trace requests as they travel across multiple services, providing end-to-end visibility into system interactions.
- **Security Information and Event Management (SIEM)**: Solutions like Splunk and IBM QRadar help in identifying and responding to security events and incidents.

Applications in SCADA Systems

In the context of SCADA systems, modern observability tools play a vital role in ensuring system reliability, efficiency, and security. Their application extends to:

- **Performance Monitoring**: Observability tools monitor the performance of various components and subsystems within SCADA, identifying bottlenecks and optimizing resource allocation.
- **Anomaly Detection and Diagnostics**: They detect anomalies in system behavior and facilitate diagnostics, thereby reducing downtime and maintaining system integrity.
- **Security and Compliance**: By monitoring system interactions and analyzing logs, these tools aid in identifying security threats and ensuring compliance with industry standards.
- **Data Management and Analysis**: Handling high volumes of data generated by SCADA systems, observability tools assist in data management, analysis, and informed decision-making.

The approach of modern observability tools has significantly shaped the way we understand and interact with complex systems. Their diverse features and capabilities are crucial for navigating SCADA systems, particularly in scientific environments dealing with high data acquisition rates and volumes. In the light of these requirements, Elasticsearch APM emerges as a fitting choice for our demonstration due to its robust performance monitoring capabilities, adaptability to varying system architectures, and comprehensive visualization features. The exploration of its applications and integrations forms a basis for the subsequent sections, where we delve into specific approaches for incorporating observability into SCADA systems using Elasticsearch APM.

SLOW CONTROL SYSTEM FOR AXSIS SPECTROMETER

To test our suggestion and create a demonstration of the two approaches of incorporating observability into SCADA systems, we turn our focus towards a specific project — a slow control system designed for the Axsis Spectrometer. This project is an ideal showcase because of its innovative design principles and unique requirements due to the spectrometer's portability and integration needs [7].

The main idea behind the project was to develop a reactive, event-driven system which is capable of quickly responds and is efficient to various events and changes within the spectrometer and integrated SCADA systems. This design paradigm ensures that the system remains highly responsive, scalable and resilient and allows seamless interactions and adaptability to changing workloads and operational demands.

One of the key requirements of the project was the ability of the slow control system to integrate seamlessly with a variety of other SCADA systems. This necessity came from the inherent portability of the Axsis Spectrometer which is designed to be utilized across multiple scientific facilities. Each of these facilities may operate its own host SCADA system. This arise a requirement of a flexible and interoperable design which allows the spectrometer's control system to function effectively within diverse technological environments.

The portability of the Axsis Spectrometer introduced several challenges and considerations in the design and implementation of the slow control system. It was crucial to ensure consistent performance, reliability and data integrity across different facilities and SCADA systems. Additionally, the system had to be capable of adapting to the specific configurations, protocols and requirements of each host SCADA system while maintaining the core functionalities and performance attributes of the spectrometer's control system.

Taking into concidiration project's context and requirements, Elasticsearch APM standed out as a suitable tool to demonstrate the incorporation of observability into the SCADA system. Elasticsearch APM has robust features, adaptability and visualization capabilities which align with the project's needs.

The new slow control system for the Axsis Spectrometer provides a valuable context for exploring and demonstrating the two approaches to enhancing observability in SCADA systems. The following sections will cover the specifics of these approaches, use of Elasticsearch APM and illustrate how they address the challenges and fulfil the requirements of this project.

MICROSERVICE FOR SCADA OBSERVABILITY

The `AxsisObserver` [10] class is a Java-based microservice designed to enhance the observability of SCADA system. This class listens to various events and messages on a designated channel, processes them asynchronously and incorporates Elasticsearch APM to monitor and trace the performance of system components in real-time.

The service starts by attaching the Elasticsearch APM to the application through `ElasticApmAttacher.attach()`. It initializes repositories to store ongoing transactions using concurrent hash maps and ensures thread safety for asynchronous operations. The service communicates via a specific channel named `axsis-xes`.

This microservice employs reactive programming principles using RxJava. It creates `PublishSubject` instances for `moveAction` and `positionAction` events. It also subscribes to these subjects to handle various Axsis move actions and position actions which are crucial events in the Axsis Spectrometer's slow control system.

The microservice creates an instance of `SseMagixClient` to connect to the SSE server. It utilizes the SSE client to observe and process different types of messages such as `txnStart`, `txnBody` and `txnEnd` and to sent over the `axsis-xes` channel. The microservice employs multiple filters and transformations to process these messages correctly based on their action types such as `move`, `done` and `error`.

The `AxsisObserver` integrates Elastic APM to start, monitor and end transactions based on the received Axsis messages. It sets the transaction name and timestamp, captures exceptions when an error occurs and ends the transaction upon completion. The microservice uses the buffering mechanism to group related events and process them as a single transaction in Elastic APM.

It is robust and includes exception handling mechanisms. If an unsupported message is received or any other exception occurs, it is handled, ensuring the stability and reliability of the microservice.

The `AxsisObserver` demonstrates a meticulous approach to incorporating observability into SCADA systems for the Axsis Spectrometer. By employing reactive streams, asynchronous communication and Elastic APM, this microservice offers a scalable and efficient solution to monitor and trace various events and transactions in real-time. Thereby, it adresses the challenges posed by the increasing complexity of modern scientific instruments.

INJECTING APM AGENTS INTO COMPONENT SOURCE CODE: AN ALTERNATIVE APPROACH

An alternative to developing a dedicated observability microservice is the direct injection of Application Performance Management (APM) agents into the source code of the SCADA system components. This approach aims to integrate monitoring and tracing capabilities more closely with each component's operation. Thus, it offers immediate insights into their performance and interactions.

Injecting APM agents involves incorporating monitoring code directly into the source files of each component. This allows real-time tracking of system's interactions, transactions and events and ensures immediate detection of anomalies or performance bottlenecks. By residing within the components themselves, the APM agents can provide more contextual data about the system's behavior.

The integrated agents are configurable. Developers may tailor the level of observability based on the specific needs and requirements of each component. This allows a balance between comprehensive monitoring and resource efficiency as not every component may need the same depth of analysis.

Since the agents are embedded within the components, they can offer insights specific to each component's functionality and performance. This gives a better understanding of the system and ability to point issues at the component level and address them promptly.

Advantages

- **Detailed Monitoring**: Direct integration allows more detailed monitoring of each component's behaviour and interactions.
- **Immediate Insights**: Real-time tracking makes it possible to immediately identify issues, thus, reducing system downtime.
- **Customization**: The ability to configure the level of observability for each component allows resource-efficient approach.

Disadvantages

- **Intrusiveness**: Injecting code into components can be more intrusive and may affect the original functionality or performance.
- **Maintenance Overhead**: Any updates or changes to the APM agents would lead to modifications in the source code of each component, leading to higher maintenance overhead.

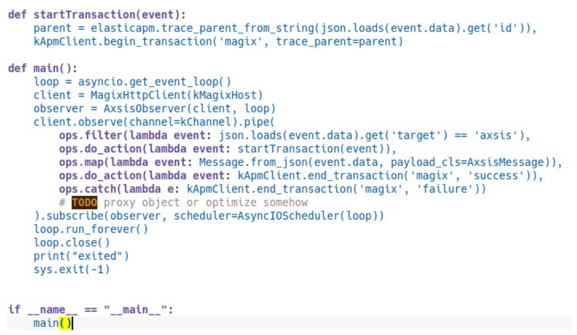
Comparison with Microservice Approach

While the direct injection of APM agents offers several advantages, it stands in contrast to the microservice approach which offers modularity and is easy in maintenance. The microservice approach, shown in `AxsisObserver`, ensures that the observability function remains separate from the components. It reduces the risk of interference and makes it easier for updates and scalability.

Injecting APM agents directly into the source code is an alternative for enhancing system observability. While it offers customization, there are some challenges such as higher maintenance. A careful comparison with the dedicated microservice approach helps in selecting the most suitable strategy based on the specific needs of the project.

Reactive event-driven systems, such as the one designed for the Axsis spectrometer, offer benefits including scalability, responsiveness and resilience. However, they also have some challenges when it comes to integrating APM agents, particularly when it comes to ensuring that transaction IDs are accurately extracted and mapped to enable proper visualization of dependent spans in Elasticsearch APM.

In a reactive event-driven architecture each component acts in response to events or messages which might *not* follow a linear or synchronous flow. This makes the extraction of transaction IDs less straightforward. Injected APM agents require additional code to accurately map and associate transaction IDs with the correct asynchronous events and messages. This ensures that Elasticsearch APM can effectively visualize the relationships and dependencies between different spans. An example of such code within the scope of a single Python-based component [11] is given in Fig. 1.



```python
def startTransaction(event):
    parent = elasticapm.trace_parent_from_string(json.loads(event.data).get('id')),
    kApmClient.begin_transaction('magix', trace_parent=parent)

def main():
    loop = asyncio.get_event_loop()
    client = MagixHttpClient(kMagixHost)
    observer = AxsisObserver(client, loop)
    client.observe(channel=kChannel).pipe(
        ops.filter(lambda event: json.loads(event.data).get('target') == 'axsis'),
        ops.do_action(lambda event: startTransaction(event)),
        ops.map(lambda event: Message.from_json(event.data, payload_cls=AxsisMessage)),
        ops.do_action(lambda event: kApmClient.end_transaction('magix', 'success')),
        ops.catch(lambda e: kApmClient.end_transaction('magix', 'failure'))
        # TODO proxy object or optimize somehow
    ).subscribe(observer, scheduler=AsyncIOScheduler(loop))
    loop.run_forever()
    loop.close()
    print("exited")
    sys.exit(-1)


if __name__ == "__main__":
    main()
```

Figure 1: This Python code snippet showcases an asynchronous approach to initiating and managing transactions in an observability system with a particular emphasis on extracting and using the trace ID. The script sets up an event loop and establishes an observer for the Axsis system, It focuses on events targeting 'axsis' and uses an asyncio library. It extracts the trace ID, initiates a transaction with Elastic APM using the extracted trace ID and subsequently concludes the transaction. This is marked as either a success or a failure. This extraction and utilization of the trace ID is crucial for ensuring accurate tracking and monitoring of transactions, enabling reliability and efficiency in SCADA environments.

This additional code is essential for managing the asynchronous nature of interactions. It helps in maintaining context across different components and asynchronous boundaries and makes the APM agents to create an accurate representation of the system's behavior and interactions. Without this the visualization in Elasticsearch APM might be fragmented or inaccurate and lead to challenges in diagnosing issues and understanding system performance.

In the context of the Axsis spectrometer project the system is designed to be portable and to seamlessly integrate with various SCADA systems. In this case accurate transaction ID mapping becomes even more critical.

To address this challenge, specific strategies and practices are employed:

1. **Context Propagation**: Implementing mechanisms for context propagation across asynchronous boundaries to maintain transaction continuity.
2. **Custom Middleware**: Developing custom middleware to extract, map and associate transaction IDs accurately with the corresponding events and messages.
3. **APM Configuration**: Fine-tuning configuration of APM agents to ensure they are optimized for the reactive event-driven nature of the Axsis system.

Integrating APM agents in a reactive event-driven systems (like the Axsis spectrometer) involves additional considerations and adaptations. The requirement for additional code so that to accurately handle transaction IDs is essential. It ensures coherent and insightful observability through Elasticsearch APM. By addressing these specifics and challenges, the Axsis project can benefit from both reactive architecture and enhanced system observability.

CONCLUSIONS AND SUMMARY

The increasing complexity of modern scientific instruments and the exponential growth in data acquisition rates and volumes underscore the need for comprehensive system observability. Integration of observability tools is not just a luxury but a need to ensure the traceability, reliability and optimal performance of SCADA systems in complex and diverse environments.

This paper has explored two primary approaches of incorporating observability into SCADA systems: the development of a dedicated microservice and the direct injection of APM agents into the source code of software components. Both methods present their unique advantages and challenges. A dedicated microservice, `AxsisObserver`, was presented as an example. It offers modularity and provides monitoring and immediate insights by direct injection of APM agents.

The reactive event-driven nature of the Axsis project adds a layer of complexity to the integration of observability tools. Asynchronous flows and accurate transaction ID mapping through additional code are crucial to maintain a coherent observability. It is especially true in a system designed to integrate with various host SCADA systems across multiple facilities.

The advent of technologies like Kubernetes (K8s) has significantly simplified integration of observability tools by providing a scalable and manageable container orchestration platform. Furthermore, many Software as a Service (SaaS) providers have recognized the value of observability and have integrated observability tools directly into their platforms. This trend suggests a path forward for large scientific facilities to embed observability as a fundamental component of their system architectures, ensuring resilience, scalability and efficient resource allocation.

Large facilities which have complex and diverse SCADA systems can take inspiration from these developments in the software industry. By adopting and integrating observability tools as essential components of their platforms, they can enhance system reliability, traceability and performance. This integrated approach to observability will enable them to reduce downtime and optimize resources.

In an era of rapid technological progress and increasing system complexities, the integration of observability tools is crutial. SCADA systems may become more resilient and efficient by exploration and analysis of different approaches, container orchestration and the adoption of observability by SaaS providers. By applying these innovations and adapting them, we can ensure the continuous improvement and success of modern control systems in scientific facilities.

ACKNOWLEDGEMENTS

Authors wish to acknowledge the incredible tools and resources provided by Elasticsearch and Canonical, particularly MicroK8s, which have been fundamental in the realization of this research. Their technologies have not only facilitated our work but also enriched our understanding and application of observability in SCADA systems.

Furthermore, Igor extends his appreciation to his current supervisor – Anton Barty – for their unwavering support and encouragement within the scope of the DAPHNE consortium.

An acknowledgment, of course, also goes to the open-source communities at large. Their innovation, knowledge sharing and commitment to make technology accessible and inclusive have not only inspired us but have also made our endeavours possible. The spirit of collaboration and the wealth of resources available within these communities have been a constant source of motivation and learning.

This research and the resulting paper would not succeed without the collective efforts, support and resources from all those mentioned above.

REFERENCES

[1] R. Dimper *et al.*, "ESRF Data Policy, Storage, and Services", *Synchrotron Radiat. News*, vol. 32, no. 3, pp. 7-12, May 2019. doi:10.1080/08940886.2019.1608119

[2] D. Andrault *et al*., "ESRF Upgrade Programme Phase II (2015 – 2022) Technical Design Study", Grenoble, France, December 2014.

[3] D. A. Poştovei, C. Bulac, I. Triştiu, and B. Camachi, "The evolution and challenges of modern Distributed Control Systems," in *Proc. SACI'20*, Timisoara, Romania, May 2020, pp. 000089-000094. doi:10.1109/SACI49304.2020.9118829

[4] R. Pandit, D. Astolfi, J. Hong, D. Infield, and M. Santos, "SCADA data for wind turbine data-driven condition/performance monitoring: A review on state-of-art, challenges and future trends", *Wind Engineering,* vol. 47, no. 2, pp. 422-441, 2022. doi:10.1177/0309524X221124031

[5] What is Observability? IBM developers guide, https://www.ibm.com/topics/observability

[6] What is Observability? A Beginner's Guide, https://www.splunk.com/en_us/data-insider/what-is-observability.html

[7] I. Khokhriakov, O. Merkulova, A. Nozik, V. Mazalova, and P. Fromme, "A novel solution for controlling hardware components of accelerators and beamlines", *J. Synchrotron Radiat*. vol. 29, no. 3, pp. 644-653, May 2022. doi:Pages 644-653. doi:10.1107/S1600577522002685

[8] A. Nozik, "Controls-kt, a Next Generation Control System", Preprints 2023, 2023081746. doi:10.20944/preprints202308.1746.v2

[9] C. Majors, L. Fong-Jones, and G. Miranda, *Observability Engineering*. California, USA: O'Reilly Media, Inc., 2022.

[10] AxsisObserver, https://github.com/waltz-controls/axsis-observer

[11] AxsisMagixClient, https://github.com/waltz-controls/axsis-xes/blob/master/axsis.magix.py