Bookkeeping, a new logbook system for ALICE

Martin Boulais[1,*], *George C.* Raduta[1,**] and *Jik* Huijberts[1,***]

[1]CERN, Geneva, Switzerland

Abstract. The ALICE experiment at CERN's Large Hadron Collider is designed to study the physics of strongly interacting matter at extreme energy densities, where a quark-gluon plasma is expected to be formed. After a major upgrade, the new ALICE Online-Offline computational system expects to read an estimated throughput of 3.5TB/s of raw data and store and index 900Gb/s reconstructed data. The complexity of this endeavour implies the need for a well-developed and integrated logbook platform, able to keep track of the experiment's activities and readily provide a history state of the system. The Bookkeeping application has been developed as part of the new ALICE Online graphical interfaces suite and allows users to insert, filter, track, and search system updates. Bookkeeping plays a central role in its integration with other components, which need to either read or update the system state. Furthermore, it builds global and individual system performance statistics which in turn help improve the overall efficiency of the experiment. This paper introduces the new Bookkeeping platform, it showcases its functionalities and purpose, details the means that have been put in place to fulfil all the requirements and presents an overview of its use during the first year of ALICE Run 3 data taking.

1 Introduction

ALICE [1] (A Large Ion Collider Experiment) is one of the four main detectors at the LHC (Large Hadron Collider). It focuses on heavy-ion collisions (Pb-Pb) to reproduce the quark-gluon plasma which is expected to be the state of matter formed after the big-bang [2].

The lifetime of the LHC is split in successive periods named "Runs" and "Long Shutdowns", representing periods dedicated to data taking and periods dedicated to system upgrades. After the successful Runs 1 and 2, it has been decided that a new set of GUIs (Graphical User Interfaces) [3] would be set-up for the incoming Runs 3 and 4 which are expected to last about ten years. These new GUIs, based on web technologies, take into consideration the feedback [4] gathered during the previous Runs. They allow for easy remote access, implement a role access control based on CERN authentication and a centralised authorisation system, while at the same time providing a similar look and feel that facilitates the switch from one application to another.

Scientific experiments, such as ALICE, need to have a constant, real-time overview of their configuration at all time. This is critical to analyse the data, reproduce the experiments in the same conditions, compute the efficiency or quickly catch, understand and solve issues.

*email: mboulais@ceren.ch

**email: graduta@cern.ch

***email: jik.huijberts@cern.ch

Moreover, scientists working on the experiment need to log events that occur at any time. A logbook system fulfills these needs, and especially a digital logbook that allows to automatize tasks and facilitate data mining.

2 Logbook System requirements for Runs 3 and 4

The logbook is one of the central applications in the experiment, as it is being used not only by a wide range of users with highly different backgrounds but also by other software components that are sending and retrieving information. It should gather, process and redistribute information from and to a large panel of human and machine clients.

2.1 System state and configuration

One of the basic needs fulfilled by a logbook is to store the system state and configuration at any time. The ALICE experiment is composed of several detectors, from which data are readout and processed by hundreds of computers, each of them being highly configurable. Data analysis requires to have knowledge of the exact conditions in which data have been acquired, and this is the reason why a logbook has to store the system configuration at any time. Moreover, to ensure continuous operations to cover as many collisions as possible, on-call experts are expected to intervene to help fix issues. When called, the experts need to have a detailed overview of the current and past system state and this information has to be easily accessible in logbook.

2.2 Logging system

The ALICE experiment is composed of several geographically spread teams. A logbook must provide a way for collaborators to work together in an asynchronous way. It needs to allow users to log noteworthy events and discuss about it. Parts of the logs require the attention of specific users or group of users which should be notified accordingly.

Most of the information stored in the logbook is critical, such as the data quality flag, a flag that determines the quality of data taking for a given period of time, based on which the system will store or discard the data acquired. Therefore, the system needs to be capable of tracking changes of this flag and notify users and systems accordingly.

2.3 End of Shift reports

In order to be successfully operated on a continuous basis, the day in the system has been split in 3 equal parts, called an 8h shift. Because of the complexity of the system, the operation team, shift crew, is further divided. Each crew is composed of multiple shifters that fulfil specific roles and focus on specific tasks of the experiment.

At the end of each shift, several reports titled "End of Shift report" must be created. They summarize the activities that took place in the past 8 hours. A report is made of 2 parts, a common one, shared by all reports with general information such as information from the previous shifter, and a unique one, shift type specific, with information dedicated to the kind of operations the shifter had to complete.

2.4 System Performance Analysis

Monitoring the system performance helps improving it. The operation of the LHC is not only dictated by ALICE, and beam collisions will continue to run even if one of the detectors is not available to take data. Therefore, statistics about the system performances, detectors availability and runs are needed to determine how the system can be improved further.

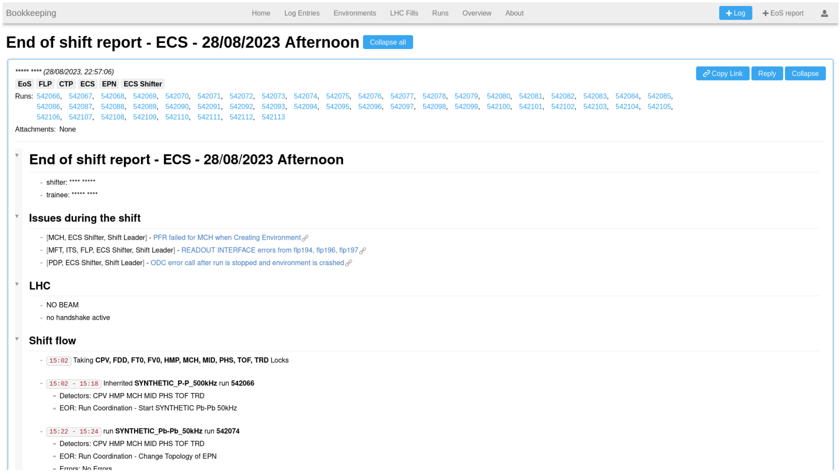

Figure 1. Example of an ECS End of Shift report.

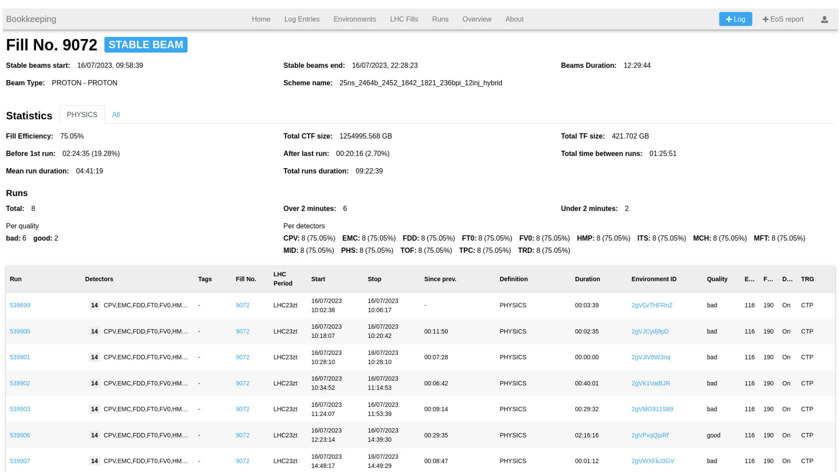

Figure 2. Information and statistics about a given LHC Fill.

3 Bookkeeping Design Overview

An in-house online logbook, named Bookkeeping [5], provides the solution to these requirements. Already in production at LHC Point 2, it is expected to be developed and maintained throughout Runs 3 and 4 until 2032. Considering this, the design of the system and the choice of technologies being used, need to be well thought out. The requirements might change during the system lifetime and developers should to be able to quickly implement the new requests, while still being able to ensure system robustness and quality.

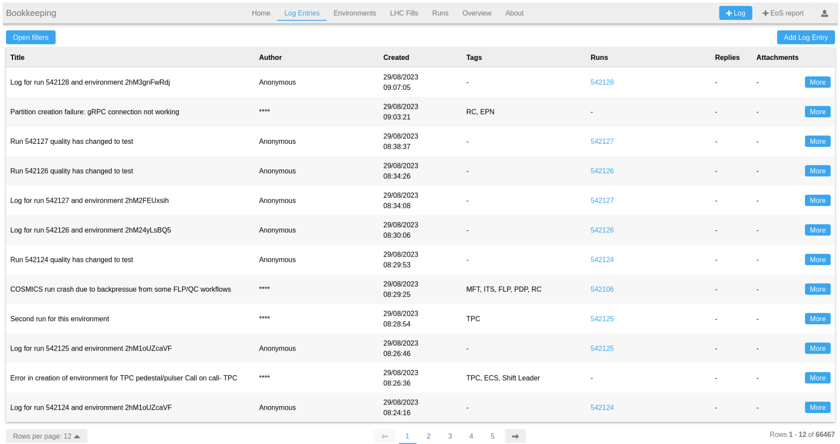

Figure 3. List of logs, created either automatically by the system or manually by users.

3.1 Bookkeeping Concepts

3.1.1 System state and configuration

The system configuration is represented by two concepts, which are the environments and runs (which do not relate to LHC Runs).

An environment describes the system at a given point (including hardware and software configuration) as a state machine; e.g. which detectors are enabled, what processes were executed, on which machines and under which conditions? The list of environments is available in Bookkeeping and a detail page shows, for each one of them the list of related runs, the history of its states (see Fig. 4), and a list of user/software log entries which describe notable events that took place.

A run represents a period of data acquisition with a given environment and the particular conditions of the data taking. As for environments, a page listing all runs is available in Bookkeeping, and the details of every run can be seen in a dedicated page (see Fig. 5).

3.1.2 Automated and manual log entries with notifications

In Bookkeeping, users can create free text logs to share relevant operational information. If logs target a specific group of people, a system of tags has been put in place to automatically send emails or Mattermost (an open source messaging system) notifications [6] to pre-defined targets when the tag is applied to a log.

To ensure critical information is properly propagated, Bookkeeping automatically creates logs on specific user actions or events that need to be registered, adding relevant tags to trigger notifications. One such action is defined by the change of the run quality flag which defines if the data will be stored or not. Thus, not only the action will automatically create a log entry and notify run coordinators but it will also prompt the user to indicate the reason for the change.

For easy data access, a page is available in Bookkeeping to provide an overview of all the logs. A filtering panel allows the user to search through the items on specific criteria (see Fig. 3).

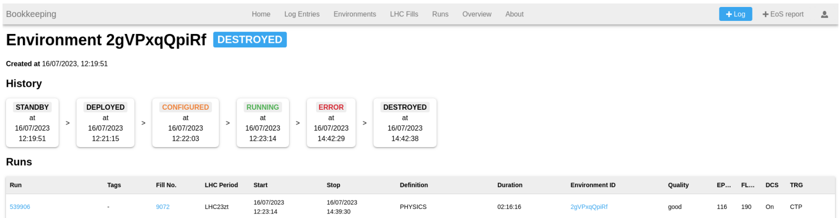

Figure 4. Environment display of a given run.

3.1.3 Automated End of Shift report

To facilitate a smooth end of shift, Bookkeeping provides a tool to automatically generate the End of Shift report by gathering all the information with regards to events that occured in the past 8 hours and by providing a form with minimal input needed from the user to fill. Information collected depends on the type of runs that have been active as well as on log entries which describe activities that happened throughout the shift.

Once the shifter confirms that the information is correct, the report is created and made available to the users of Bookkeeping. Thus, not only shifters can focus on operations rather than collecting data during their 8 hours period, but also, the next crew of shifters can quickly read the report to be aware of the activities that happened before their shift starts (see Fig. 1).

By narrowing the manual user input to its strictest form we ensure End of Shift report will have a common structure. Doing so, information retrieving and processing is facilitated.

3.1.4 On the spot statistics

Bookkeeping stores all the metadata about data taking, such as configurations, system states and events that occur. Using this information, it is able to combine them with data provided by the LHC services to automatically compute the statistics needed by the run coordination. At the time of writing, statistics are available per LHC fill (see Fig. 2), with new ones being added soon, allowing users to view them based on a range of runs and fills defined using a filtering panel.

3.2 Multi-client constraints

Bookkeeping centralizes the configuration, state and events of the whole distributed system. Thus, it needs to provide an API to a wide range of clients. These clients are implemented by various teams using a large set of technologies: Go and C++ for the control system [7], C++ for readout [8] and data quality monitoring [9], Java for the interface to the LHC DIP system [10] and other web applications.

Bookkeeping provides two different APIs for clients to communicate: a gRPC API and an HTTP API.

3.2.1 gRPC

gRPC [11] is a robust open source framework based on protocol buffers, a well-described serialization protocol. The gRPC framework provides the following advantages:

- It automatically compresses the data transferred, which reduces the network load.
- Being based on protocol buffer, the API endpoints and structures are strongly typed, which works particularly well with typed languages such as Go and C++.

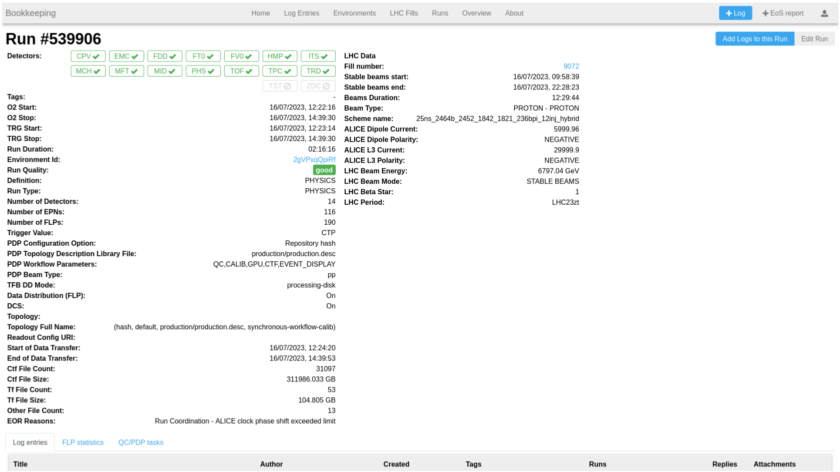

Figure 5. Run display.

3.2.2 HTTP

On the other hand, the HTTP framework is easier to use by clients that do not rely on strictly typed API structure. The Bookkeeping frontend communicates with the backend through this API but not only, other clients uses it such as Calibration and Conditions Database that sends storage information or counters through HTTP requests.

Such a separation of APIs across two clients requires to carefully design the application to avoid code duplication and ease the maintenance: a software layer needs to extract data from API requests and format API responses, and this layer needs to be strictly limited to this purpose. A representation of the system architecture is available in (see Fig. 6)

3.3 Technical Stack

One of the main limitations of the previous generation GUIs was the difficult access from outside of the control room. To overcome this limitation, choice has been made to create a new set of web-based GUIs. These are written in JavaScript, which is the reference in web development. It provides a smooth learning curve while still being powerful, and by using the Node.js [12] runtime it allows to have the same language both on backend and on fronend, which eases the development process. Node.js has the particularity of being non-blocking, which allows to handle a high amount of requests in parallel. On top of Node.js, the backend HTTP API is served using Express [13], a middleware-based library that abstracts the low-level HTTP system implementation and relies on a series of functions taking a representation of the HTTP request and building the HTTP response. The frontend is a single page application built using a customized version of Mithril.js [14]. Mithril.js is a lightweight, component-based framework that facilitate the development of highly maintainable browser-based applications.

To simplify the communication of C++ clients with Bookkeeping through gRPC, an in-house developed library is provided to abstract completely the communication from Bookkeeping: the clients have C++ functions to call which will transparently make the gRPC requests and wrap the response in C++ objects. This also brings for free the freedom to change the communication medium if needed, because the clients have no explicit dependency on gRPC.

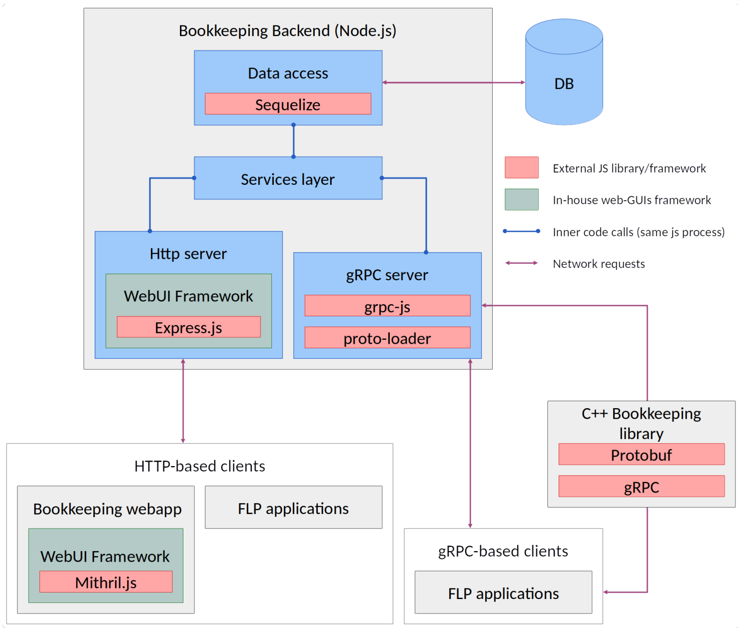

Figure 6. Global architecture of the Bookkeeping project

3.4 Common WebUI framework

To optimise code reusability between the several ALICE web GUIs, an in-house framework named WebUI has been built, providing tooling for both the frontend and the backend. Using a common framework not only increases development speed by sharing as much code as possible between the different projects, but also improves the security of applications by focusing the security concerns in a single point. This allows developers to focus only on developing features and fixing bugs, without having to deal with purely technical issues and boilerplate code. Using a common framework also makes the look and feel of all the GUIs alike, and allows users and developers to switch from one application to another without trouble.

4 Operational Experience

Bookkeeping has been deployed to production since day one of Run 3 operation period (mid-2021). During this time[1], more than 850 users have used it. Together with the dependent services, they have created almost 70.000 entries, out of which 30.000 were created in 2023 with more than 3.800 attachments, and described by a set of more than 100 tags.

With regards to runs, Bookkeeping is storing information for about 40.000 (12.000 in 2023) runs with a total overlapped duration of approximately 800 days, distributed over 1500 LHC fills (more than 600 in 2023)[1].

[1] At the time of writing

5 Conclusion

The ALICE logbook, named Bookkeeping, is currently being used in production while being actively maintained and developed to include new features as per users' feedback. Being one of the most used among the new GUIs of the ALICE experiment, it provides a web application to its users but also two different APIs to integrate with other software components. As it is based on an in-house framework, developers can focus on implementing new features and fixing bugs while providing to the users a similar look and feel as the other ALICE GUIs to ease context switching.

The next Bookkeeping developments will focus on adding new automated statistics and improve the integration with other systems by adding new endpoints.

References

[1] ALICE Collaboration, JINST **3**, S08002 (2008)

[2] The ALICE Experiment - A journey through QCD, *arXiv:2211.04384*

[3] From design to production: State-of-the-art web user interfaces to operate ALICE offline and online system, G. C. Raduta (ALICE), J. Phys.: Conf. Ser. **2438**, 012041 (2021)

[4] The ALICE Collaboration, Tech. Rep. CERN-LHCC-2015-006. ALICE-TDR-**019** 232 (2015), https://cds.cern.ch/record/2011297

[5] Bookkeeping project, *https://github.com/AliceO2Group/Bookkeeping*, accessed: 2023-09-13

[6] A. Wegrzynek, and V. Gioacchino. "The evolution of the ALICE O2 monitoring system." EPJ Web of Conferences. Vol. **245**, EDP Sciences, 2020.

[7] T. Mrnjavac, K. Alexopoulos, V. Chibante Barroso, and G.C. Raduta (ALICE), EPJ Web of Conferences **245**, 01033 (2020)

[8] S. Chapeland and F. Costa (ALICE), EPJ Web of Conferences **214**, 01041 (2019)

[9] P. Konopka and B. von Haller (ALICE), EPJ Web of Conferences **245**, 01027 (2020)

[10] E. Mandilara, I.P. Barreiro and F.V. Rodriguez, 2017. Monitoring of CERN's data interchange protocol (DIP) system. *https://accelconf.web.cern.ch/icalepcs2017/papers/THPHA162.pdf*

[11] gRPC framewowk, *https://grpc.io*, accessed: 2023-09-13

[12] Node.js JavaScript Runtime Environment, *https://nodejs.org*, accessed: 2023-09-13

[13] Express framewowk, *https://expressjs.com/*, accessed: 2023-09-13

[14] Mithril.js framewowk, *https://mithril.js.org/*, accessed: 2023-09-13