EPJ Web of Conferences **337**, 01143 (2025) https://doi.org/10.1051/epjconf/202533701143
*CHEP 2024*

# Adapting to Change: A look at the evolution of ALICE's Quality Control framework

*Piotr* Konopka[1,*], *Barthélémy* von Haller[1,**], and *Michal* Tichák[1,***] for the ALICE Collaboration

[1]The European Organization for Nuclear Research CERN, Experimental Physics Department, Geneva, Switzerland

**Abstract.** Since the mid-2010s, the ALICE experiment at CERN has seen significant changes in its software, especially with the introduction of the Online-Offline (O²) computing system during Long Shutdown 2. This evolution required continuous adaptation of the Quality Control (QC) framework responsible for online Data Quality Monitoring (DQM) and offline Quality Assurance (QA).

After a general overview of the system, this talk delves into the evolving user requirements that shaped the QC framework from its initial prototyping phase to its current state. We will explore the changing landscape of performance needs and feature demands, highlighting which initial requirements persisted, which emerged later, and which features ultimately proved unnecessary.

Additionally, we will trace the framework's development in relation to other software components within the ALICE ecosystem, offering valuable insights and lessons learned throughout the process. Finally, we will also discuss the challenges encountered in balancing development team resources with the evolving project scope.

## 1 Introduction

The ALICE (A Large Ion Collider Experiment) experiment at CERN investigates primarily the quark-gluon plasma, a state of matter formed under extreme energy densities, through heavy-ion collisions.

The ALICE Online-Offline (O²) computing system is the result of merging data acquisition and processing into a unified online and offline system [1]. It was designed and implemented as a part of the major ALICE upgrade that took place during the LHC Long Shutdown 2 (2019-2022). The O² system uses a message-passing approach to handle high-throughput data efficiently, with the Data Processing Layer (DPL) [2] as framework and FairMQ for its transport mechanism. Effectively, computations are split among multiple actors which process data in parallel and exchange messages via shared memory. This architecture ensures low-latency, scalable processing for the vast data produced by the ALICE experiment.

*e-mail: piotr.konopka@cern.ch
**e-mail: barthelemy.von.haller@cern.ch
***e-mail: michal.tichak@cern.ch

The data Quality Control (QC) system integrates the traditional online Data Quality Monitoring (DQM) and offline Quality Assurance (QA) into a unified framework [3–5]. It ensures a quick detection of issues related to data and processing during data taking (in the order of minutes) and a more complete assessment of the quality and suitability for analyses of physics data during the later asynchronous reprocessing.

## 2 Gathering requirements

The QC system was first mentioned in 2013 during the O² project kickoff. A dedicated Computing Working Group (CWG) was created bringing together people (experts and users) from the traditional online DQM and offline QA. Its first purpose was to collect requirements for the future Quality Control system.

Regular meetings took place throughout the years till 2015 when a first prototype was released. The collection of requirements mainly took place during these meetings by interviewing the various stakeholders and incorporating the experience from Run 1. The gathered feedback indicated preferences towards familiar, existing solutions. At that stage, the DQM users were more active in the working group, which consequently resulted in the requirements set being more precise in the online domain, rather than the offline QA.

Finally, similar systems in the other LHC experiments were reviewed early on, to assess whether critical requirements had been overlooked and to nurture and challenge our own views of what such a system should look like.

The collection of needs continued long after the first prototype and actually never stopped. This is expected as new requirements emerge when using the software. This is particularly true in a completely new computing system such as O². Regular feedback and discussions with the users and a fast release cycle have been key to provide the community with a tool that fits their needs. We noticed that most of the specific feature requests only came after the users could use the new software.

Looking back at the process, it appears that the most valuable and on-point feedback came from experienced users of similar systems. Although they are biased towards familiar solutions, they are very clear on what they need. Guiding discussions toward expected features helped prevent them from getting too deep into implementation details, which would would sometimes happen with users able to program.

### 2.1 Complete set of requirements

The requirements gathered during the process described in the previous section are the following.

The Quality Control (QC) system should allow for the comprehensive monitoring and assessment of data quality in the ALICE experiment. It must integrate with the existing and future data processing frameworks of the experiment. In particular it should run within the ALICE Online-Offline computing system. As a consequence, it should support the processing happening in real-time (synchronous), in offline (asynchronous) and in simulation (Monte Carlo), on any kind of computing nodes (event processing, DAQ, Grid, Laptop). Also, it should be able to scale together with the main data and processing flow, which can be distributed across hundreds or thousands of nodes.

The framework should provide a way for detector teams to define algorithms (called Tasks) which may process physics data and create plots, and algorithms which perform automatic evaluation of data quality (Checks). They should be interchangeable, pluggable and reusable.

Tasks should be able to consume messages from any physics data processing step (raw, partially processed, processed) and access information from other subsystems such as Detector Control Systems (DCS), Bookkeeping (ALICE logbook), Condition and Calibration DataBase (CCDB). Tasks should be provided with either 100% of requested physics data or a sampled subset, but they should not cause backpressure to the main data flow unless explicitly allowed to. Tasks may access more than one data source. Sampled data should be shipped to a given Task even if it is running on a different node (within reasonable performance limits). The framework should provide the following methods to select (sample) data messages:

- pseudo-randomly, with well-proven statistical soundness and a defined average percentage of messages. If a given event is distributed over multiple nodes, the provided sample should be complete. The sampling sequences should be reproducible.
- *n* consecutive messages in evenly spaced intervals
- by message payload size
- by a user-defined algorithm which should be loaded from a user library

Tasks should be able to perform user-defined actions during process configuration, start of run and end of run. If they are considered critical to healthy data-taking or processing, they should stop the whole ongoing activity in case of fatal errors. Tasks may create objects (plots) which are typical data structures used for data quality monitoring: histograms, graphs, tables provided by the ROOT framework [6]. Tasks can run either on data processing nodes (local mode) or on dedicated QC nodes (remote mode). The framework should merge objects created by multiple instances of Tasks distributed among processing nodes before any following steps, such as running Checks and storage. The code which handles object merging (called Mergers) should be made available also for other uses in the experiment's software. During data taking, the users should see objects regularly updated with a defined cycle duration. The cycle duration can be made shorter at the beginning of a data-taking run. If needed, the users should be able to define the data time range that objects should correspond to and the information about contained time range should be known to the user. Tasks may have user-defined parameters in configuration files which should be provided by the framework.

Checks should be able to evaluate any set of objects produced by Tasks. If required objects are created by more than one Task, the user may select an update policy of a Check. Checks should return information about data quality with any comments containing details of an issue and a set of flags indicating the expected effect on usability of data for physics analyses. These results should have an associated time range corresponding to the processed physics data.

The Check results can be aggregated in order to obtain a higher level quality of a detector or an aspect of data processing. The user should be able to define how the qualities are combined or use one of the pre-defined algorithms and select an update policy. The quality aggregation should be possible in any number of stages.

The objects by Tasks (plots) and Checks (qualities) should be stored for later access and preservation. The storage should be centralized, provide an API and be accessible from the experimental site and the Grid. It should accept writing at O(1000) objects per minute and be able to store multiple versions of a given object (typically a few versions for each run permanently). Additionally, the Check results should be optionally propagated to Bookkeeping for the use of data preparation and selection for physics analyses.

The QC results should be easily accessible via a GUI which should not require installation and should be reachable also from outside the CERN network. The user may browse all available objects and any versions of them, as well as create and share layouts of selected

| Detector | 2014 (estimates) | 2020 (estimates) | 2022 (measured) | 2024 (measured) |
|---|---|---|---|---|
| A | 4 | 23,055 | 42 | 149 |
| B | 9 | 3 | 3,315 | 906 |
| C | 70 | 3,690 | 289 | 643 |
| ... | ... | ... | ... | ... |
| Total | 5,000 | 52,000 | 12,644 | 28,836 |

**Table 1.** Evolution of the number of objects per detector over time. The detector acronyms were anonymised.

plots. It should be possible to find an object version for a given run and asynchronous data processing pass.

The framework should provide a way to post-process objects stored in the repository, both during data-taking and after and create any new objects as a result. Just as Tasks, the users should be able to define their own algorithms or use pre-defined ones, which should cover typical use-cases - creating simple trends and correlations. The post-processing should be triggered based on selected criteria, such as: an appearance of a new object version in the QC repository or CCDB, at start or end of run or periodically.

### 2.2 Performance requirement evolution

The Quality Control framework is designed to operate with data acquisition (synchronously) and during offline data reconstruction (asynchronously). Due to their performance-critical nature, synchronous operations are described with more detail.

In 2018, initial estimates assumed an input to the data acquisition farm of approximately 3.5 TB/s. Thanks to FPGA-based zero-suppression and software-based compression on the First Level Processors (FLPs), the input data rate would be decreased to 500 GB/s before reaching the Event Processing Nodes (EPNs). After synchronous processing, the projected storage rate was 90 GB/s. The system was initially designed to accommodate 250 First Level Processors and between 1500-4000 Event Processing Nodes.

For the Quality Control processing framework, key performance parameters included the maximum data rate on a First Level Processor, which impacts Data Sampling and Tasks, and the number of Event Processing Nodes, which correlates with the input data throughput of Mergers. The object repository's requirements were primarily influenced by the number of unique paths (objects), the quantity of object versions stored permanently, and the frequency of read and write requests.

The estimated total number of object paths increased from 5000 in 2014 to 52000 in 2020. Objects were expected to be published and stored at intervals of several minutes, with average sizes ranging from 250 kB to 1 MB. Determining the requirements for the number of Tasks and objects presented challenges, particularly due to uncertainties in plot formats from future detector module developers. Balancing user needs with system capabilities was crucial in defining the expected usage of the framework. Often, consolidating multiple 1-dimensional histograms into a single 2-dimensional histogram could significantly reduce performance demands.

Table 1 illustrates the evolution of requirements for the number of objects over time.

By late 2019, revised estimates reduced the expected number of Event Processing Nodes to 750. Due to ongoing uncertainties in various parameters, the calculated input data throughput to Merger nodes ranged widely from 400 MB/s to 166 GB/s, indicating significant uncertainty in the estimates.

As of 2024, the O² system comprises approximately 200 First Level Processors and 350 Event Processing Nodes (operating as 700 virtual nodes). During a typical lead-lead (PbPb) data-taking run at maximum rates, the total input data throughput to Merger nodes is measured at 7.8 GB/s.

Currently, the Quality Control repository houses 26 million objects arranged in more than 22000 paths, occupying nearly 1 TiB of storage. During data acquisition, the database processes approximately 40 write and 15 read requests per second.

Detailed benchmarks of the framework components have been published in [3, 4], providing empirical data concerning the system performance.

### 2.3 Discussion

In complex projects, the evolution of requirements is a common phenomenon. Some feature needs are identified only after software deployment, while others, initially planned, may be underutilized or prove infeasible to implement due to various constraints.

An illustrative example of late feature requests includes quality aggregation and reference plot comparison, both built upon the foundation of Checks. These features, absent in the ALICE Data Quality Monitoring (DQM) system for Run 2, were introduced in the Quality Control (QC) framework for Run 3. This suggests that the lack of prior experience with Checks may have limited the proposals for potential applications built upon them.

The original QC design proposed two methods for obtaining plots with complete statistics from parallel computing nodes: merging plots generated on parallel nodes (local mode) and utilizing a single Task instance to consume samples from all nodes (remote mode). In practice, the remote mode saw limited application beyond the early commissioning stages. Users quickly recognized that a single-threaded Task was typically insufficient to process the required sample rate, necessitating the parallelization of sample consumption.

The Data Sampling component allows for defining any algorithm of selecting interesting data for further data quality processing, for example to filter only potentially bad events. Despite an interesting shift compared to traditional approach of processing randomly sampled subset of data, we did not see any interest in using this feature. We assume it was due to a concern of having less predictable load on Tasks, missing important messages or simply a habit of more experienced users.

The postprocessing framework was designed to enable trending of selected observables and production of correlation plots. This functionality was provided with a reusable algorithm, where trending was conceptualized as a correlation against timestamps. Despite initial interest, the use of correlation plots was limited. We hypothesise that it was due to a need for substantial statistics before such analyses become meaningful. Consequently, an increase in usage is expected towards the end of Run 3.

The original design of the O² system proposed a paradigm shift in the concept of data-taking runs, treating them as collections of timeframes gathered under similar conditions. This design assumed rather frequent system reconfiguration, for example to adjust sampling rates or enable more sophisticated Tasks for issue investigation. However, due to multiple technical challenges in other O² components and their integration, this approach was largely abandoned. A re-evaluation of the feasibility of flexible Task addition, removal, and reconfiguration is planned during the Long Shutdown 3 of the LHC.

During the early stages of QC framework development, there was significant interest in employing Machine Learning (ML) methods for anomaly detection in detector health monitoring. Initial investigations were conducted. However, comprehensive testing of these approaches was hindered by the lack of real or simulated data from the upgraded detector,

which is crucial for training and testing ML algorithms. With a substantial data sample now accumulated, there is potential to revitalize interest in these methods.

## 3 Development and framework adoption

Figure 1 illustrates the project's code activity, annotated with allocated annual developer time and the number of unique user contributors. The period between 2015 and 2017 involved the development of initial demonstrators and prototypes. During this phase, several dependencies crucial to Quality Control were either absent or in early stages of development. Consequently, the prototype's primary value lay mostly in demonstrating the design assumptions and gaining initial experience with them.

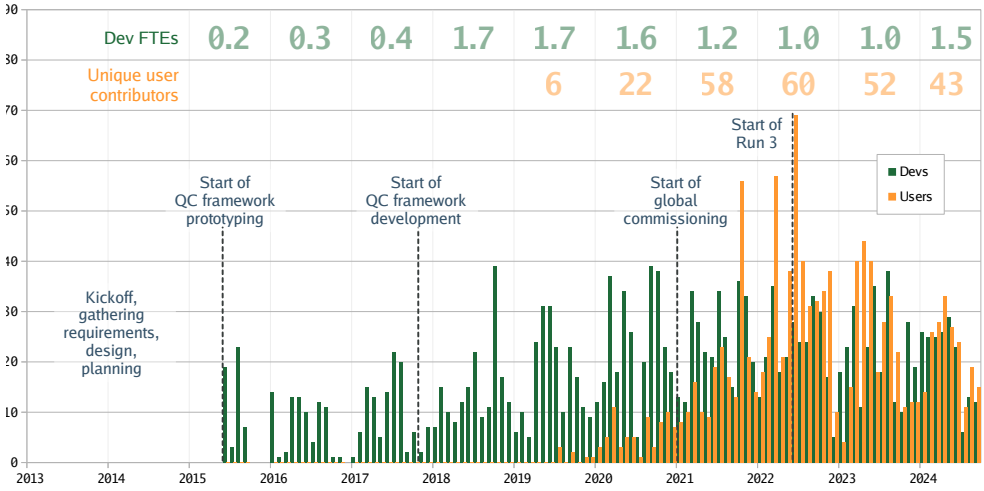


**Figure 1.** Commit frequency per month of the Quality Control framework and modules compared with developer manpower and user activity.

The framework's development started in late 2017, supported by two developers working almost exclusively on the project. Taking into account the lessons learned from prototyping phase, majority of the framework components were rewritten to adapt to a new data processing framework in ALICE (Data Processing Layer [2]) and several services. Progress was systematically reported to and evaluated by prospective users, with new features being actively promoted. Approximately 18 months prior to the global commissioning of the upgraded detector and computing system, initial QC framework users began developing their modules. In 2020, the project had attracted 22 unique user contributors. The early adoption of the software proved to be exceptionally beneficial in aligning requirements and functionalities with user expectations, subsequently mitigating developer workload during the global commissioning phase.

Global commissioning of the upgraded ALICE detector and software started in 2021. It brought a substantial increase in user activity, as illustrated in Figure 1, and user feedback. Many smaller features in the framework were proposed only at that time. Unfortunately, the surge in activity of users was not matched by a corresponding increase in developer resources, requiring a stricter task prioritisation and postponing improvements in code maintainability.

Software integration efforts were structured into monthly Milestone Weeks, each with specific objectives. While this approach facilitated the progress in global commissioning, it also reduced flexibility in maintaining a sustainable software development process. Additionally, we encountered situations where the QC software dependencies failed to provide expected features within the anticipated timeframe, necessitating the implementation of temporary solutions within the QC framework. To avoid these issues in the future, we aim to improve flexibility of the developer team, so that members are trained across multiple project areas. This approach will allow us to quickly shift resources to critical tasks as needed.

Physics data acquisition commenced in April 2022. From that point forward, Quality Control contributed to asserting the quality of each data-taking run. Subsequent work focused on minor extensions based on user requests and addressing discovered bugs. Users continued to enhance their modules with additional Tasks and improved plots.

The software development cycle incorporated version control (git), a pull request review process, and an automated build and test system. Throughout the development cycle, the framework was consistently covered with unit tests and basic functional tests. New releases were subsequently tested in conjunction with other project components using automated integration tests. Successful releases were then manually validated on the ALICE staging environment.

## 4 Summary

The ALICE Quality Control (QC) framework has been ensuring the quality of data in the experiment since the beginning of Run 3, providing both almost real-time Data Quality Monitoring (DQM) and offline Quality Assurance (QA). Designed to operate within the Online-Offline ($O^2$) computing system, it integrates seamlessly with ALICE's data acquisition and processing workflows.

The development of the framework highlighted the importance of continuous requirement gathering and user feedback. In a complex and evolving computing environment, many needs only became apparent after initial prototypes were tested in real-world conditions. Users often required direct interaction with the system to refine their expectations, leading to an iterative development process. Balancing the expansion of features with available developer manpower proved to be a key challenge, as did integrating the framework with the broader ALICE software ecosystem.

As of 2024, the QC framework is fully operational, supporting ALICE's data-taking and analysis efforts. Future improvements will focus on improved automation and exploring advanced techniques such as machine learning for anomaly detection. The experience gained throughout this process will inform further refinements, particularly in preparation for Run 4 and beyond.

## References

[1] P. Buncic, M. Krzewicki, P. Vande Vyvre, Tech. Rep. CERN-LHCC-2015-006. ALICE-TDR-019 (2015), https://cds.cern.ch/record/2011297

[2] Eulisse, Giulio, Rohr, David, The O2 software framework and GPU usage in ALICE online and offline reconstruction in Run 3, EPJ Web of Conf. **295**, 05022 (2024). 10.1051/epjconf/202429505022

[3] P. Konopka, Ph.D. thesis, AGH University of Science and Technology, Cracow, Poland (2022), https://cds.cern.ch/record/2814071

[4] Konopka, Piotr, von Haller, Barthélémy, The ALICE O2 data quality control system, EPJ Web Conf. **245**, 01027 (2020). 10.1051/epjconf/202024501027

[5] von Haller, Barthélémy, Konopka, Piotr, The ALICE Data Quality Control, EPJ Web of Conf. **295**, 02026 (2023). 10.1051/epjconf/202429502026

[6] R. Brun, F. Rademakers, ROOT — An object oriented data analysis framework, Nuclear Instruments and Methods in Physics Research A **389**, 81 (1997). 10.1016/S0168-9002(97)00048-X