

Commissioning of the ALICE Data Acquisition system

**T. Anticic⁽¹⁾, V. Barroso⁽²⁾, F. Carena⁽²⁾, W. Carena⁽²⁾, S. Chapeland⁽²⁾,
O. Cobanoglu⁽³⁾, E. Dénes⁽⁴⁾, R. Divià⁽²⁾, U. Fuchs⁽²⁾, T. Kiss⁽⁴⁾, I. Makhlyueva⁽²⁾,
F. Ozok⁽⁵⁾, F. Roukoutakis⁽²⁾, K. Schossmaier⁽²⁾, C. Soós⁽²⁾, P. Vande Vuyvre⁽²⁾,
S. Vergara⁽⁶⁾**

(for the ALICE collaboration)

⁽¹⁾ Ruder Bošković Institute, Zagreb, Croatia

⁽²⁾ CERN, Geneva, Switzerland

⁽³⁾ INFN Turin, on leave from Istanbul University, Istanbul, Turkey

⁽⁴⁾ KFKI-RMKI, Budapest, Hungary

⁽⁵⁾ Istanbul University, Istanbul, Turkey

⁽⁶⁾ Benemerita Universidad Autonoma de Puebla, Mexico

Abstract. ALICE (A Large Ion Collider Experiment) is the heavy-ion detector designed to study the physics of strongly interacting matter and the quark-gluon plasma at the CERN Large Hadron Collider (LHC). A flexible, large bandwidth Data Acquisition System (DAQ) has been designed and deployed to collect sufficient statistics in the short running time foreseen per year for heavy ions and to accommodate very different requirements originated from the 18 sub-detectors. The Data Acquisition and Test Environment (DATE) is the software framework handling the data from the detector electronics up to the mass storage. This paper reviews the DAQ software and hardware architecture, including the latest features of the final design, such as the handling of the numerous calibration procedures in a common framework. We also discuss the large scale tests conducted on the real hardware to assess the standalone DAQ performances, its interfaces with the other online systems and the extensive commissioning performed in order to be ready for cosmics data taking scheduled to start in November 2007. The test protocols followed to integrate and validate each sub-detector with DAQ and Trigger hardware synchronized by the Experiment Control System are described. Finally, we give an overview of the experiment logbook, and some operational aspects of the deployment of our computing facilities. The implementation of a Transient Data Storage able to cope with the 1.25 GB/s recorded by the event-building machines and the data quality monitoring framework are covered in separate papers.

1. Introduction

The ALICE experiment [1],[2] consists of 18 detectors aiming at studying the interaction of particles colliding at the LHC. It primarily targets heavy-ion reactions, but it will also be able to cope with proton-proton and proton-ion collisions.

The Pb-Pb collisions will occur during a few weeks per year, characterized by big events (86.5 MB), large bandwidth to mass storage (1.25 GB/s), low interaction rate (10 kHz), and complex triggers. On the other hand, the pp and pA collisions will occur during several months per year and produce relatively small events (2.5 MB) at a high-interaction rate (200 kHz), needing less bandwidth and simpler triggers with increased selectivity.

ALICE will collect physics data in a large set of detector configurations: detectors will be able to work altogether or separately, in stand-alone operation or synchronized data taking.

Based on these heterogeneous requirements, the ALICE DAQ was developed, deployed, and thoroughly tested to transform the 25 GB/s aggregated throughput from the detectors into a set of recorded physics data files. We will review the architecture and components used to implement the data flow, how the components are operated, and how the DAQ is interfaced with the other systems.

2. Online data flow

The data flow is controlled by the following online systems:

- The Trigger (TRG) is in charge of selecting the events, initiating the detectors read-out, and synchronizing the experiment with the LHC machine clock.
- The Data Acquisition (DAQ) is responsible for the data-flow from the detector electronics to the permanent storage, and for the control of this data-flow.
- The High-Level Trigger (HLT) provides filtering information to optimize the amount of interesting data in the available bandwidth.

Each system, described in details in a Technical Design Report [3], is made of several pieces of hardware equipments and software components interacting as described in Figure 1.

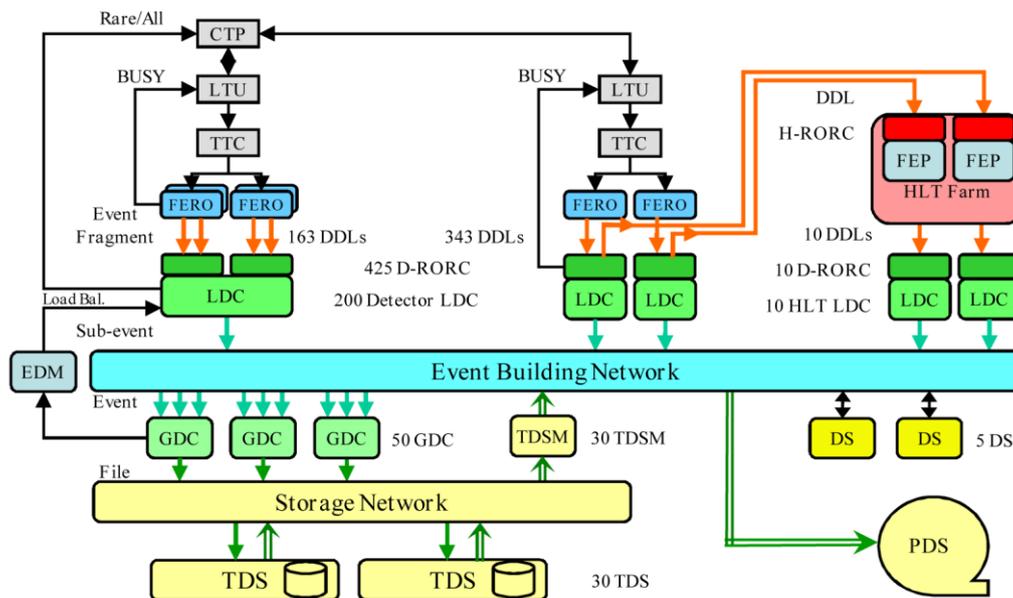


Figure 1: ALICE Data-Acquisition architecture

2.1. The Trigger system

The Central Trigger Processor (CTP) receives the input from the trigger detectors and the LHC clock. For every bunch crossing, and according to the busy status of all the detectors, the CTP produces trigger decisions which are transmitted to each detector via its Local Trigger Unit (LTU). The LTU converts these decisions into messages which are distributed to the detector electronics via the Timing, Trigger and Control (TTC) broadcast system. In ALICE, different types of triggers are generated, involving different sets of Front-End Readout Electronics (FERO) to be read-out.

2.2. The Data Acquisition

The readout electronics of all the detectors is interfaced to the ALICE standard Detector Data Links (DDL). The DAQ uses more than 500 DDLs, each link being able to transport about 200 MB/s over the ~100 meters long fibers, from the detector in the experimental area to a counting room close to the surface.

Each DDL is connected on one side to the FERO by a Source Interface Unit (SIU). At the receiving side of the DDLs there are PCI-X boards, called DAQ Read-Out Receiver Cards (D-RORC), hosted by PCs, the Local Data Concentrators (LDCs). Each D-RORC hosts two DDL interfaces and each LDC can handle one or more D-RORCs. The SIU is a FPGA board housed inside the detectors' electronics, and has been designed and tested [4] to cope with the radiation level in this area. D-RORCs use DMA through the 100MHz PCI-X bus to efficiently transfer the data in the memory of the host LDCs. The selected computers can host up to 6 D-RORCs.

The event fragments are read out by the LDCs and shipped by TCP/IP through a standard Gigabit Ethernet network to another pool of computers, the Global Data Collectors (GDC), which build the full events.

The load of GDCs is balanced by an Event Distribution Manager (EDM) process. The events are then temporarily stored by the GDCs on a local Transient Data Storage (TDS) system [5], residing on an independent storage network to avoid traffic congestions. It is implemented by mounting on the GDCs a shared file system installed on disk arrays accessed through Fiber Channel.

The files are later on migrated by Transient Data Storage Movers (TDSM) to Permanent Data Storage (PDS) in the remote CERN computing centre, using a safe 20Gb/s uplink (redundant pair of 10 Gb Ethernet).

Some DAQ Services (DS) machines provide various central facilities to the other computers, including configuration database, publish and subscribe information servers, monitoring and logging repositories.

2.3. The High-Level Trigger

On most of the D-RORCs, one of the two DDL links is used to transfer the received detector data to the HLT system. This information is processed in a farm of computing elements, the Front-End Processors (FEP). They read data from the DDL via an HLT Readout Receiver Card (H-RORC). The analysis results and filtering decisions are injected back in the DAQ by another set of DDLs attached to dedicated LDCs.

3. Deployment and operation

3.1. Processes control and configuration

The Experiment Control System (ECS) [6] provides a unified view of the experiment and a central point from where all operations are initiated and controlled. It permits independent, concurrent activities on part of the experiment by different operators and coordinates the operations of the online systems (TRG, DAQ, HLT, and Detector Control System) within the groups of detectors called "partitions". The ECS consists of a software layer interfacing with the online systems. The processes are controlled and synchronized by a set of Finite State Machines (FSM) describing the behavior of each logical entity or experiment component.

The DATE software [7],[8] is the distributed system managing the DAQ data flow, consisting of several processes executed on every node, depending on its role in the data acquisition. Their overall control and synchronization is achieved by the DATE runControl, which also uses FSM to describe the states and transitions of the software processes.

FSM are implemented using SMI [9]. The main FSM elements are running on the DS servers and communicate with remote items using the DIM protocol [10]. DIM is a publish-and-subscribe communication system based on TCP/IP. These logic daemons run only on a few machines.

The hierarchy of detectors and all the configuration parameters are stored in a database. Each DAQ component loads at startup appropriate data from the central repository, according to its role and partition membership. In this way, it is not necessary to propagate configuration updates by pushing them on the hosts. The database can be edited manually, but for the large production setup it is populated automatically based on minimal information (list and assignment of machines, identification of data links, reference configuration files).

A logging mechanism collects the messages from all the DAQ processes, which are recorded centrally and can be browsed from a single interface (see Figure 2). It provides a full framework to generate, transport, collect, store and consult log and error messages.

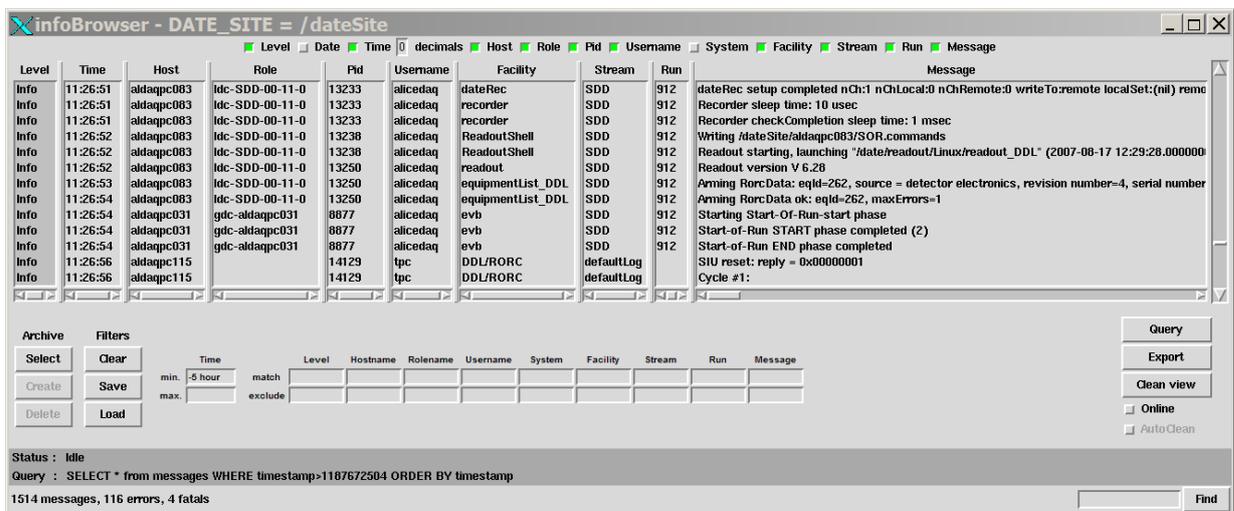


Figure 2: infoBrowser console

3.2. Computing infrastructure

All the DAQ computing equipment is installed in the same area, up in the pit near the surface. The DAQ machines reside on a private network, both for performance and security reasons. Incoming and outgoing traffic is closely restricted, and done through a limited number of gateways with adequate firewall rules. The network backbone (Force10 E-1200) demonstrated a performance largely meeting the requirements. In particular, it showed excellent linearity in throughput as function of the number of data producers.

The base computing platform consists of Intel 32 bit PC running the Scientific Linux CERN version 4 (SLC4). The number and the aggregate bandwidth of PCI-X slots were critical to select the LDCs, whereas CPU and memory speed was crucial for the event building process performed by the GDCs. For optimal performance of the database servers, AMD Opteron processors were selected, with the 64-bit version of the same operating system. The storage devices are 4TB disk arrays with FC-4G optical connection, each of them providing a RAID 6 configuration for maximum speed and reliability. All the PCs can be remotely operated by Keyboard Video Mouse (KVM) console through a Web interface when necessary.

Machines are mounted in refrigerated racks to fit the limited space (70m²) of the counting room and cope with the heat dissipation. The total electricity power reaches 200kW, out of which 37kw are on Uninterruptible Power Supply (UPS) with a run time of 10 minutes. Each rack contains a Power Distribution Units (PDU) accessible by the network to control and monitor power usage, and some temperature and smoke detection probes. A database stores a detailed list, description and characteristics of all the DAQ hardware running in the counting room. It keeps track of the large number of identifiers, labels, serial numbers of this computing centre, together with the hierarchical

arrangement of components and history of hardware interventions. A Web interface is used to browse the content of the database in a functional and logical way.

All the nodes are installed from a local software repository. PXE is used to boot from network and initiate the installation process. A Linux Kickstart script defines the groups of packages to be installed on each type of machine, and the post-install configuration steps to be performed. At the end of the procedure (a matter of few minutes), computers are fully operational and ready to fulfill their role in the DAQ system. When needed, they are then upgraded without the need of a full re-install.

All software components are installed from RPM packages, stored in three distinct YUM [11] repositories: one for the base CERN SLC4 bundle, one for the updates, and one for the DAQ kits (and dependencies) not part of the standard distribution. When new versions are committed to the repositories, a global upgrade is easily triggered (machines reboot) and updated RPMs are propagated automatically with YUM to ensure consistent software on all the nodes.

The databases are implemented in MySQL. It provides very good performance, minimal administration overhead, and open-source access. The data are backed-up daily, and a second node always ready to take over in case of a major hardware problem. When applicable, the database structure is upgraded transparently with DATE updates.

3.3. System operation

The system is highly flexible to allow a wide variety of runtime configurations, from a single detector standalone run to parallel and independent data taking with different groups of detectors.

The DAQ and the detectors processes are operated from the comfortable ALICE control room (see Figure 3) located at ground level. Eleven working positions are equipped with a silent computer connected by digital D-DVI links to six 19" LCD flat panels, assembled in a single wide 114x90 cm screen. This huge (3840 x 3072 pixels) graphics area is refreshed by 3 dual-head video cards, and seen as a single desktop by windowed applications with the help of the Xinerama feature of the X server. It provides a huge space to open the control Human Interfaces (HI) and be able to see them all at a glance.



Figure 3: ALICE control room

The HI programs interact only with few top-levels DAQ and ECS components at runtime, and offer a seamless connection to the control processes executed remotely in the counting room. Only

lightweight transient commands and status variables need to circulate between the operator post and the central server hosting the runControl and ECS state machines. The main DAQ and ECS HI programs allow to view the state and to launch actions on the selected subsystem. Access control enforcing is done when logging in. Users can then control only the subsystems for which they have privilege, for example the DAQ of a single detector.

Event and performance monitoring is achieved in several ways. At the higher level, the DAQ and ECS control HI display the status of subsystems and react upon changes. Errors are escalated from the lower layers to the top and appropriate measures taken automatically: for example, stopping the run in case of a link failure detected by a LDC. The infoBrowser, the HI to the DAQ logging mechanism, displays in real time messages sent from all the processes, and is convenient to look for detailed information on the cause of errors. This facility provides complete identification of the source of the message (host, process and user id, role in the DAQ, partition, timestamp), as well as means to selectively filter the information depending on the context or other message characteristics.

Specific tools have also been developed to evaluate the DAQ performance. AFFAIR (A Flexible Fabric and Application Information Recorder) is an integrated monitoring tool graphically reporting the status of system and DAQ metrics, and gives appropriate feedback on the entire data flow.

Data quality monitoring is insured by MOOD (Monitor of Online Data and Detector Debugger), with specific displays for each detector, and the new AMORE framework [12] that automatically collects physics statistics derived from the actual payload of the detectors data. These home-grown solutions complete the standard fabric monitoring system, called Lemon [13], and used to monitor the health of the computing farm at the node level. These flexible monitoring tools allow real time monitoring and offline analysis to tune the whole system.

Bookkeeping is also a key component of a system built with long time life in mind (>10 years) and operated continuously by different crews. This task is fulfilled by the run logbook developed for the DAQ. It consists of a database storing all run related events and statistics, and a Web-based interface (written in PHP) used to consult and populate it (see Figure 4). The base entries are written automatically by the DAQ and ECS processes: run number, date/time and duration, detectors and DAQ nodes participating, amount and rate of data taking, etc. Operators can also add entries, either related to a run number or not. This may include text messages, but also images or other type of files. The interface allows browsing the logbook content by different entries: by run, by time, by files (with thumbnails), with some filtering and search capabilities. It offers a convenient way to flag runs with special characteristics, report events and their cause, leave messages for later analysis, and more generally keep a memory of runtime operation. Secure access and authentication mechanisms are built in.

Run	Start Time	Duration	LDCs	GDCs	Detectors	Total Events	Total Data (MB)	Data Rate (MB/s)	Events/s	Run Type
(1) 912	21/08/2007 11:26:57	na	1	1	1	na	na	na	na	DAQ
(2) 911	21/08/2007 10:52:41	24 s	1	1	1	2 955	123.14	334.29	334.29	DAQ
(2) 910	21/08/2007 10:47:27	36 m	1	1	1	443 407	240 801	112.16	206.52	DAQ
(2) 909	21/08/2007 10:45:01	22 s	1	1	1	4 673	2 538	115.35	212.41	DAQ
(2) 908	20/08/2007 23:37:47	15 m	1	1	1	93 707	102 566	112.22	102.52	DAQ
(2) 907	20/08/2007 23:09:15	27 m	1	1	1	167 120	182 918	112.08	102.40	DAQ
(2) 906	20/08/2007 22:56:29	12 m	1	1	1	74 152	80 909	112.22	102.85	DAQ

Figure 4: Screenshot of the DAQ electronic logbook

4. Integration and testing

Standalone tests of the DAQ software have been conducted continuously during its development, sometimes by emulating the outside components not yet available. This procedure allowed validating the behavior and performances of the whole chain of DATE processes and data flow. In particular, this task was achieved during the various data challenges performed at CERN to measure the sustained event building rate and writing speed to permanent data storage (see Figure 5).

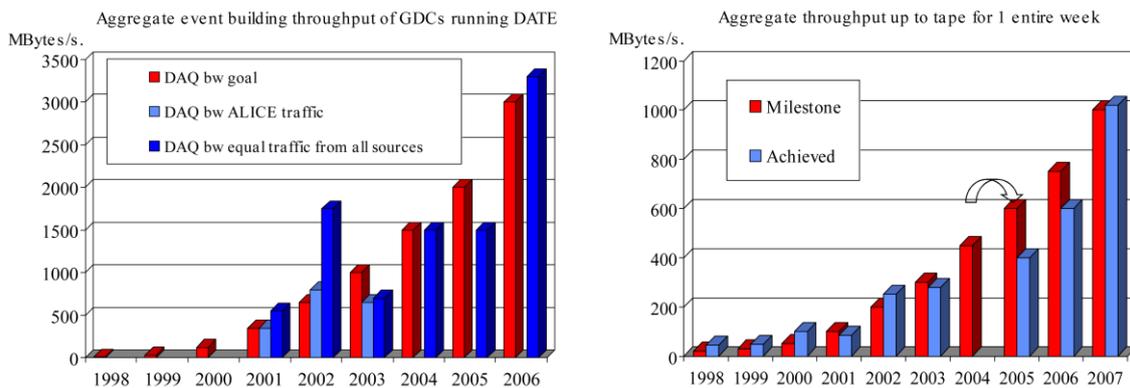


Figure 5 – Data Challenge achievements

At the time of integration, it was critical to verify the interfaces with the final components in place. This includes the main assignment of the DAQ, detector readout to data recording, as well as the other activities which are essential for the experiment control or offline analysis of the measurements.

For all the interfaces, the first (and non-trivial) step was to check the physical connections, like attenuation-free peer-to-peer optical links between specified ports, or correct wiring of Ethernet cables and ad-hoc routing of TCP/IP packets. The second step was to verify accordance of the high-level protocols with specifications.

4.1. Detector electronics interface

Data is received from all the detectors in the same way by the DDLs. Some detectors also use the full-duplex capability of the DDL to configure their electronics.

A hardware data source, the DDL Data Generator (DDG), was designed to check the DDL interface prior to detector integration. It is connected to the trigger via the TTC, and provides a realistic emulation of the detector electronics injecting data in the DDL. It has been used to validate the whole DAQ chain with Trigger.

The DATE readout process, in charge of handling the data from the D-RORC cards, implements extensive checks of the data format and event protocol. The transmitted fragments over the DDLs from the FEROs and the HLT farm to the LDCs are composed of a Common Data Header (CDH) and a payload. The 32 byte CDH contains trigger information (e.g. orbit number, bunch crossing number, trigger class) and status/error information about the FERo. The CDH of each event is verified at run time, as well as the presence of the special Start Of Data and End of Data events that start and end the stream of all runs.

Before the final integration in the experimental area, the detector electronics is connected to a standalone test setup consisting of DAQ and Trigger, controlled by ECS. Data taking is launched for long periods with all the checks enabled, and successful runs validate the DAQ/detector interface. The correct handling of erroneous trigger sequences (e.g. bad framing time) is also tested. Through such demanding tests, errors are quickly detected and implementation problems in the FERo spotted. These

software verifications will also provide valuable information in production to detect events corrupted by the radiations.

It should be noted that having a single and common transport layer and protocol to FERO was a key point to successfully integrate the many 18 detectors with the DAQ.

4.2. Trigger interface

The DAQ does not interface directly to the Trigger for the control of the data flow. The ECS synchronizes both systems in the following sequence at each run: DAQ processes are started, trigger is enabled; then to stop the run trigger is disabled and DAQ stopped (it exits once End of Data event is received from FERO, showing that all FERO buffers are empty). However, the recording of all the trigger messages sent to the detectors need to be stored for reference and physics measurements. A special CTP readout program was developed to inject such information in the main data flow. A DDL, with a specific protocol, delivers the interaction records and trigger messages from the CTP to the DAQ. A process dispatches this stream by TCP/IP to the various DAQ systems running concurrently, where it is mixed with the detector readout data.

4.3. HLT interface

The interface to High-Level Trigger is bidirectional: part of the detector data is copied transparently by the D-RORC card to a set of DDLs readout by HLT, which in return injects HLT decisions and analysis results by another set of DDLs coming back to DAQ. These DDLs are readout by a standard readout process (in the same way as for detectors). A special process, the `hltAgent`, catches, handles and propagates the decisions before the event is shipped to the event builder. This work flow was tested in a first step by injecting simulated HLT decisions from a DDG, and then with the real HLT facility in the experimental area. The ECS synchronizes DAQ and HLT at start of run, and in particular provides configuration information on the current setup (e.g. from which detectors the HLT receives data, and where the decisions should be sent) based on the latest database information.

4.4. Offline interface

DAQ and Offline are fully decoupled at run time: the DAQ writes to permanent data storage, and the Offline reads from it for analysis. There is consequently very little interaction as far as the data is concerned, and the interface is limited to the data registration to the Grid and to the Offline file catalogs. This procedure was exercised during Data Challenges, test beams, and detector commissioning runs.

However, another requirement arose to provide a faster analysis of data. To exploit as early as possible the physics data recorded, it is necessary to handle online most of the detector calibration procedures. Each detector has special needs in term of calibration, and those are taken into account whenever possible at the level of the DAQ software. The general task consists in taking data (during physics run or in dedicated calibration runs, with standard or particular triggers) and processing it to create calibration maps.

A special framework was implemented to allow the detectors to run some calibration code on the DAQ machines. Such processes are called Detector Algorithms (DA). Two classes of operation have been defined. DAs usually do not need full events, so the raw data can be retrieved directly from the LDCs.

The first type of calibration corresponds to a standalone run, where data is recorded on disk and analyzed directly on LDCs at end of run. The second corresponds to calibration procedures during a normal run, on dedicated machines (called monitoring servers) to avoid impacting the DAQ performance. The usual task of this second class of DAs is to accumulate histograms on a specific measurement. In both cases, the DA process connects directly to the data sources (LDCs processes or

files) using the same monitoring library. The monitoring library allows reading data from files, or from a specific set of machines, one or several detectors, and possibly a single type of event (identified by the trigger class). A central database is provided to store user's configuration (e.g. electronics map files, or other parameters subject to change), with tools to access this area.

Some rules apply on the running of DAs on DAQ machines. In particular, they must be packaged in a specific way with RPM, their runtime footprint should be under close control, and the code dependencies and input/output operations restricted to the predefined local storage space. The code for these DAs comes from a wide variety of sources, hence a strict approach was chosen to compile and validate them. The DA packages are created on an automatic build server, providing the required dependencies, and their performance assessed with some test raw data files. Only after this stage they are installed on the DAQ production machines.

DA results are exported to the offline computing reference database through a special interface, the File Exchange Server (FXS). It temporarily hosts the calibration results and associated metadata written at end of run by the DAs, before the files are retrieved by the Offline. The Offline process in charge of collecting the files also extracts important run characteristics from the DAQ logbook.

In addition, some calibration results may also be stored locally in the DAQ to configure at next run the electronics via DDL, or exported to the Detector Control System (DCS) via the FXS for the same purpose.

5. Conclusion

The ALICE DAQ architecture is a distributed system of hundreds of hardware and software components, coordinated by a state-machine based control system. This design has been successfully tested in a number of test-beams and Data Challenges, and was deployed in the experimental area where the detector commissioning is going on. The DAQ system has proved to be flexible enough to cope with the large range of configurations required for the experiment, and was extended accordingly with the latest requirements. The top-level configuration and control offers complete yet simple mechanisms to operate the system, hiding the low-level complexity. The appropriate use of general purpose elements and implementation of ad-hoc solutions has permitted to achieve the required performance and a smooth integration with the other subsystems.

References

- [1] ALICE Collaboration, Technical Proposal, CERN-LHCC-1995-71.
- [2] <http://aliceinfo.cern.ch/>
- [3] ALICE Collaboration, The Technical Design Report of the Trigger, Data-Acquisition, High Level Trigger, and Control System, CERN-LHCC-2003-062, January 2004.
- [4] E. Dénes et al., Radiation Tolerant Source Interface Unit for the ALICE Experiment, *Proc. of the Workshop on Electronics for LHC Experiments (LECC 2005)*, Heidelberg, Germany, 12-16 September 2005, 291-293.
- [5] U. Fuchs et al., The ALICE DAQ Online Transient Data Storage System, CHEP07.
- [6] <http://cern.ch/alice-ecs>
- [7] <http://cern.ch/alice-daq>
- [8] K. Schossmaier et al., The ALICE Data-Acquisition System DATE V5, CHEP06.
- [9] <http://smi.web.cern.ch>
- [10] <http://dim.web.cern.ch>
- [11] <http://linux.duke.edu/projects/yum/>
- [12] F. Roukoutakis et al., The ALICE-LHC Online Data Quality Monitoring Framework, CHEP07.
- [13] <http://cern.ch/lemon>