THE ALICE DATA-ACQUISITION SOFTWARE FRAMEWORK DATE V5

F. Carena, W. Carena, S. Chapeland, R. Divià, I. Makhlyueva, J-C. Marin,
K. Schossmaier, C. Soós, P. Vande Vyvre, A. Vascotto for the ALICE collaboration
CERN, Geneva, CH-1211, Switzerland

Abstract

The data-acquisition software framework DATE for the ALICE experiment at the LHC has evolved over a period of several years. The latest version DATE V5 is geared for deployment during the test and commissioning phase. The DATE software is designed to run on several hundred machines being installed with Scientific Linux CERN (SLC) to handle the data streams of approximately 400 optical Detector Data Links (DDLs) from the ALICE sub-detectors and to write full events onto transient/permanent data storage at a rate of up to 1.25 GB/s.

DATE V5 consists of a collection of software packages that are responsible for the data flow and its formatting to carry out the readout, the event-building, and the recording. Additional software packages are in charge of the control, the system configuration, the status and error message reporting, the electronic logbook, the data quality and performance monitoring, and the memory management. The interfaces to the Experiment Control System (ECS) and to the High-Level Trigger (HLT) are implemented, whereas the interfaces to the Detector Control System (DCS) and to the Trigger System (TRG) are in design status.

This paper will present the software architecture of DATE V5, the practical experience acquired at various detector integration setup, and future extensions.

INTRODUCTION

ALICE (A Large Ion Collider Experiment) [1] is one of the four Large Hadron Collider (LHC) experiments carried out at CERN. Its heart is a general purpose, heavy ion particle detector designed to study the physics of strongly interacting matter and the quark-gluon plasma in nucleus-nucleus collisions. The ALICE detector being constructed in the cavern at LHC Point 2 is composed of 17 sub-detector systems, each one with their specific detector technology and Front-End Electronics (FEE).

The ALICE Data Acquisition (DAQ) system [2] is located in a counting room in the access shaft at the pit. All the FEEs are connected to the DAQ machinery with the ALICE standard Detector Data Link (DDL) [3]. The link cards provide a well-defined interface and protocol to move data streams in both directions over optical fiber cables with a rate up to 2125 Mbps. About 400 DDLs are allocated in order to cover the bandwidth requirements of the ALICE sub-detectors. The connection between the DAQ system and the Permanent Data Storage (PDS) situated at the CERN computing center is based on 10 Gigabit Ethernet technology over optical fiber cables. The required sustained bandwidth to the PDS is 1.25 GB/s. The estimated size of an event is 86.5 MB in Pb-Pb interactions and 2.5 MB in p-p interactions.

The ALICE Data Acquisition and Test Environment (DATE) is the software framework of the ALICE DAQ system. DATE has evolved over a period of several years and the latest version V5 is ready to be deployed for the test and commissioning phase. Earlier versions of DATE have been used successfully in various ALICE test-beam setups and other fixed target experiments. The DATE V5.*x* kits (a minor release *x* rolls out around every month) can be downloaded as RPM packages (~20 MB) from our Web site [4] along with the user's guide [5]. The kits are precompiled for IA32 compatible computing platforms running Scientific Linux CERN version 3 (SLC3) as the recommended Linux operating system.

The main part of this paper gives an overview how DATE V5 is organized in terms of packages and how it deals with the issues of data flow, control, configuration, and monitoring. It presents at the end some performance measurements which have been acquired at test setups for detector integration.

SOFTWARE PACKAGES

DATE V5 has been developed in the C programming language, using CVS as source code management tool. It depends on a few standard system libraries and some other open source software (e.g. MySQL, Tcl/Tk, DIM, SMI, libshift). The DATE V5 kit is structured in the following software packages:

- The *runControl* package contains all the control logic to operate the DATE system including the human interfaces. It also deals with the liaison to the Experiment Control System (ECS).
- The *readout*, the *readList*, and the *cole* packages provide the software to perform the readout of the FEE either via DDLs or VMEbus boards. In addition there are software-based data generators available for test purposes.
- The *eventBuilder* package is in charge of merging together several streams of sub-events into a single stream of full events. The *edm* package takes care of the load balancing.
- The *hltAgent* package deals with the liaison to the High-Level Trigger (HLT) system.
- The *recordingLib* package provides to all DATE elements a general recording facility, whereas the

mStreamRecorder package is optimized for the recording to PDS.

- The *db* and the *editDb* packages manage the static configuration parameters of all DATE elements. The associated data can be stored either in text files or in a MySQL database.
- The *monitoring* package offers a library for the development of user-specific monitoring programs both online and offline in C, C++ or ROOT.
- The *infoLogger* package supplies facilities to send, collect and browse log messages created by all DATE elements. It also contains a bookkeeping service which gathers statistics of each run.
- The *physmem* package as well as the *rorc* package provide a Linux kernel module together with the corresponding API. The *physmem* driver gives access to a block of reserved physical memory (up to 2 GB for the IA32 family). The *rorc* driver gives access to the DDL hardware, the *rorc* library is the low-level interface to it, and the *rorc* utilities allow stand-alone operations for test purposes.
- The *banksManager*, the *bufferManager*, and the *simpleFifo* packages are concerned with the management of memory banks, buffers structures, and single-producer/single-consumer FIFOs. Their implementation is tailored towards low latency.

DATA FLOW

A schematic view of the overall data flow is illustrated in Figure 1. It follows a data driven push-down approach. For each bunch crossing in the LHC machine (25 ns in p-p mode, 125 ns in Pb-Pb mode) the Central Trigger Processor (CTP) decides whether to collect the raw data. These trigger decisions are distributed coherently and timely to the FEEs of each sub-detector through their corresponding Local Trigger Unit (LTU) and the Trigger, Timing, and Control (TTC) system. Upon reception of the trigger messages, the digitized data are then transferred over the DDLs from the FEEs to the DAQ system.

The architecture of the DAQ system lays down two sets of computing platforms: the Local Data Concentrators (LDCs) and the Global Data Collectors (GDCs). The role of the LDCs is to perform the readout from the various data sources and to transmit the sub-events to the GDCs. The role of the GDCs is to build the full events and to store them onto the Transient Data Storage (TDS). All these computing platforms are interconnected by the event-building network which is able to convey TCP/IP streams. In addition there are servers which are executing global tasks such as controlling and monitoring. More information about the DAQ hardware can be found in another CHEP06 paper [6].

The DATE V5 software is running on all the LDCs, GDCs and servers. It can be configured to the minimum of one LDC/GDC which is usually the situation for FEE integration setups. In the final installation stage of the ALICE DAQ, the DATE software has to cope with ~200 LDCs, ~50 GDCs and ~10 servers.

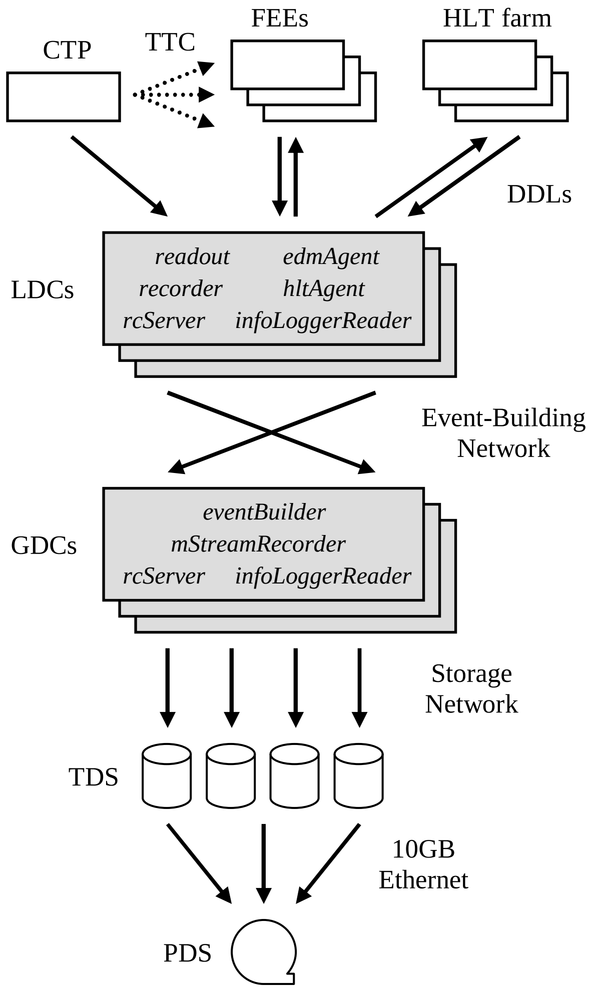

Figure 1: Schematic view of the data flow

LDC software

The *readout* process and the *recorder* process are running on all the LDCs participating in the data taking. The readout software is arranged in terms of so-called “equipments” which is a set of 5 routines including their parameters that handle the readout of a single data source. DATE V5 provides a library of equipments for the DDL, for some VMEbus boards, and for software-based data generators. Further equipments can be added to the library as needed. This concept of equipments allows a uniform and flexible way to perform the readout of one or more DDLs per LDC from the FEEs, the HLT farm, the CTP, or other online systems. The received data fragments from the DDLs are assembled into sub-events by means of descriptors avoiding memory-to-memory copy operations. The *recorder* process is capable to streamline the sub-events and writes them onto local disks or sends them to the GDCs. All sub-events that belong to same trigger need to be sent to one GDC. In addition, the Event Distribution Manager (EDM) executes a load balancing algorithm for the GDCs, whereas the *edmAgent* process instructs the LDC how to dispatch the sub-events to the GDCs.

The interface between the DAQ system and the HLT farm has two directions. A readout equipment configures the DDL hardware sitting in an LDC to transfer a copy of the data fragment coming from an FEE to an HLT node by using again a standard DDL. The results of the HLT processing, the trigger decisions, and the compressed data are transferred to an LDC as if it comes from a detector. The *hltAgent* process is handling the HLT specific data and instructs the LDC to drop or to keep the sub-event.

GDC software

The *eventBuilder* process is running on all the GDCs participating in the data taking. Each such process handles the incoming TCP/IP streams from the LDCs and stores temporarily the sub-events in data buffers. The assembly of the full event is guided by the event-building rules and the headers of the sub-events, see Figure 2. In addition the *eventBuilder* process informs the EDM about its load.

Each full event is either recorded directly onto files or moved to a post-processing stage by means of a copy-less memory mapped access scheme. The *mStreamRecorder* process is attached in the latter way. It provides enhanced recording features for multiple stream recording by using different data formats (raw binary, ROOT tree) and transmission protocols (standard, RFIO, ROOTd).

DATA FORMAT

The data format at the various processing stages is depicted in Figure 2. The transmitted fragments over the DDLs from the FEEs and the HLT farm to the LDCs are composed of the Common Data Header (CDH) and the payload. The 32 byte CDH is defined in [7] and contains trigger information (e.g. orbit number, bunch crossing number, trigger class) and status/error information about the FEE. The LDCs assemble those fragments into sub-events and append to each arriving fragment the 24 byte equipment header (e.g. size, equipment type/id) and one 68 byte LDC header (e.g. size, event type/id, trigger information, timestamp). Finally, the GDCs merge the appropriate sub-events into full events and append the 68 byte GDC header (same structure as the LDC header).

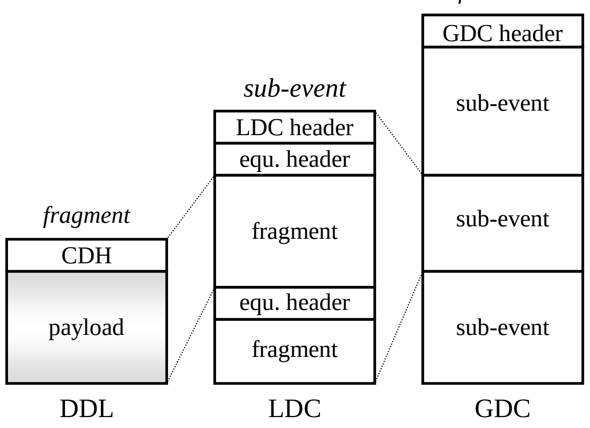

Figure 2: Data format

A data taking run of physics events produces a clearly defined sequence of event types: two “Start_Of_Run” events enclosing zero or more “Start_Of_Run_Files” events, the “Start_Of_Data” event for the synchronization that all data sources are ready [8], eventually the series of “Physics_Event” events of this run, the “End_Of_Data” event for the synchronization that all data sources are stopped [8], and two “End_Of_Run” events enclosing zero or more “End_Of_Run_Files” events. Further types are defined to signify events for calibration, software trigger, and start/end of burst.

CONTROL

Several data acquisitions can run at the same time on one DATE V5 installation, given there are no conflicting resources. Each such data acquisition is controlled by a *runControl* process, which is usually running on a server. This process has an interface to receive commands from an operator via a human interface or from the ALICE top-level control called Experiment Control System (ECS). When the run parameters and options are settled, the *Logic Engine* process is started which hosts the SMI-based finite state-machines for starting and stopping all the DATE processes.

The *rcServer* process runs on all the LDCs and GDCs, see Figure 1. It is launched remotely via the *xinetd* Linux daemon. The *rcServer* process is endowed to start and stop the DATE processes according to the commands received from the *Logic Engine* processes. These objects communicate with each other via the DIM protocol. At startup time the *rcServer* process creates a shared memory control region which is used for synchronization and inter-process communication. This control region is also the place where the status display picks up its values, e.g. number of sub-events, sub-event rate, events recorded.

CONFIGURATION

For any DATE V5 setup the following items need to be configured: the roles of the machines, the relationship between triggers and detectors, the event-building rules, the memory schemes, the choice of readout equipments, the server whereabouts, and the pertinent parameters of the various application processes. In DATE V5 almost all static configuration data, i.e. this information is valid across runs, are stored in a MySQL database which can be edited with the help of graphical tools. For backward compatibility, all the configuration data can be placed in text files as well.

Besides the basic configuration of the DATE V5 system itself, there are a number of facilities to configure/control the surroundings, for example the FEEs via the backward channel of the DDLs, see Figure 1. Data blocks and commands can be downloaded with the *rorc* library by using Front-end Control and Configuration (FeC2) scripts or the provided C API. These scripts and programs can be stored in the MySQL database and launched automatically during the “Start Of Run” (SOR) or “End Of Run” (EOR) phases on LDCs.

MONITORING

The DAQ system is distributed over a large number of machines which can change over time due to evolution, upgrades and failures. In order to observe the status of the overall system, the quality of the acquired data at different stages, as well as the utilization of computing resources, the following monitoring services are integral part of DATE V5 or available as supplementary packages.

An *infoLoggerReader* daemon is started on the LDCs, GDCs and servers, see Figure 1. These daemons collect the log messages including their attributes from the locally running DATE processes and send them to a server using TCP/IP, where they are archived in a MySQL database or in text files. There are tools to display, to search, and to export messages from the log repository. Based on this mechanism, the bookkeeping service gives out statistics of each run via Web access.

The DATE add-on called "Monitor of Online Data and Detector Debugger" (MOOD) is a software framework aiming for general purpose data quality monitoring. It is based on the DATE *monitoring* library, hence events can be monitored online (LDC/GDC data flow) as well as offline (recorded files). MOOD is written in C/C++ with plug-in ROOT modules to decode and display the detector specific payload.

The DATE add-on called "A Flexible Fabric and Application Information Recorder" (AFFAIR) has been developed to monitor the performance of the overall DAQ system. The performance plots of DATE specific values (e.g. DDL throughput, LDC sub-events count, aggregate GDC throughput) are gained by sampling fields of the shared memory control region, whereas the performance plots of the system behaviour (e.g. CPU utilization) are derived from system calls. All these plots are accessible via a Web interface. The implementation of AFFAIR is based on a client/server architecture by using DIM, SMI, ROOT, and a so-called "Round Robin Database" (RRD).

TESTING

The DATE V5 kits are tested thoroughly. Before being released they are installed and operated on our so-called "reference system" which is a mid-size setup (currently 15 computers, 13 DDLs) to check all functionalities and to run long-term tests. Yearly executed "data challenges" at the CERN IT computing center focus on the scalability of DATE, the event-building performance, and the connection to the TDS and PDS. Test-beam applications assess the DAQ system under real conditions. Finally, there are several setups in place to accomplish the detector FEEs integration with DATE.

Figure 3 presents performance measurements obtained from a single LDC setup for the ALICE Silicon Pixel Detector (SPD) by using up to 4 DDLs and the trigger system (LTU in emulation mode). Their electronics permit a maximum event rate of 3.5 kHz, which can be easily observed at the beginning linear rise of the plots.

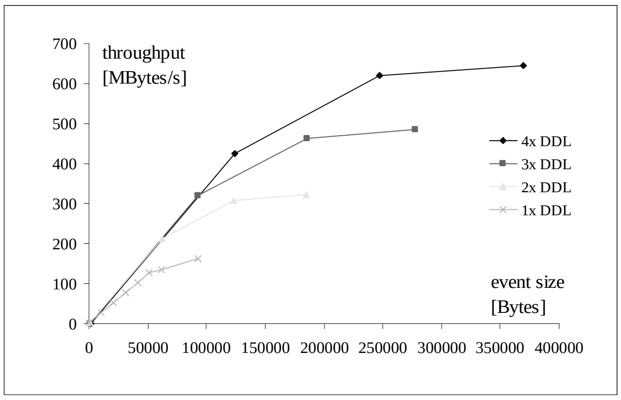

Figure 3: Performance of an LDC with 4 DDLs for the ALICE Silicon Pixel Detector

CONCLUSIONS

DATE V5 is designed to cover the whole range of data acquisition tasks for the ALICE experiment at the LHC. The development has reached a point where it can be deployed as DAQ system for the test and commissioning phase. The DATE kit is available as an RPM package together with a complete user's guide. The experience acquired at test-beams, data challenges, and detector integration setups has proven that DATE V5 satisfies the functional and performance requirements.

The development of the DATE software is continuing and the objectives for the next version V6 are being defined. This includes an adaptation to 64-bit computing platforms and to the emerging Scientific Linux CERN version 4 (SLC4). New features are foreseen such as an "electronic logbook" and an advanced Transient Data Storage Manager (TDSM). The interfaces to the other ALICE online systems will be finalized and all the data formats will be firmly fixed to facilitate the offline analysis.

REFERENCES

[1] ALICE Collaboration, Technical Proposal, CERN-LHCC-1995-71.

[2] ALICE Collaboration, TDR of the Trigger, Data Acquisition, High-Level Trigger, and Control System, CERN-LHCC-2003-062.

[3] http://cern.ch/ddl

[4] http://cern.ch/alice-daq

[5] ALICE DAQ Project, ALICE DAQ and ECS User's Guide, ALICE-INT-2005-015.

[6] T. Anticic et al., Architecture and Implementation of the ALICE Data-Acquisition System, CHEP06.

[7] R. Divià, P. Jovanovic, P. Vande Vyvre, Data Format over the ALICE DDL, ALICE-INT-2002-010.

[8] F. Carena, W. Carena, E. Evans, P. Jovanovic, A. Jusko, R. Lietava, J-C. Marin, K. Schossmaier, P. Vande Vyvre, O. Villalobos Baillie, Start of data and end of data events in ALICE, ALICE-INT-2005-019.