External access to ALICE controls conditions data

2014 J. Phys.: Conf. Ser. 513 012015

(http://iopscience.iop.org/1742-6596/513/1/012015)

External access to ALICE controls conditions data

J Jadlovský[1], A Jadlovská[1], J Sarnovský[1], Š Jajčišin[1], M Čopík[1], S Jadlovská[13], P Papcun[1], R Bielek[1], J Čerkala[1], M Kopčík[1], P Chochula[2], A Augustinus[2]
[1] Technical University of Košice, Faculty of Electrical Engineering and Informatics, Department of Cybernetics and Artificial Intelligence, Košice, Slovak Republic
[2] CERN, the European Organization for Nuclear Research, Geneva, Switzerland

Abstract. ALICE Controls data produced by commercial SCADA system WINCCOA is stored in ORACLE database on the private experiment network. The SCADA system allows for basic access and processing of the historical data. More advanced analysis requires tools like ROOT and needs therefore a separate access method to the archives. The present scenario expects that detector experts create simple WINCCOA scripts, which retrieves and stores data in a form usable for further studies. This relatively simple procedure generates a lot of administrative overhead - users have to request the data, experts needed to run the script, the results have to be exported outside of the experiment network. The new mechanism profits from database replica, which is running on the CERN campus network. Access to this database is not restricted and there is no risk of generating a heavy load affecting the operation of the experiment. The developed tools presented in this paper allow for access to this data. The users can use web-based tools to generate the requests, consisting of the data identifiers and period of time of interest. The administrators maintain full control over the data - an authorization and authentication mechanism helps to assign privileges to selected users and restrict access to certain groups of data. Advanced caching mechanism allows the user to profit from the presence of already processed data sets. This feature significantly reduces the time required for debugging as the retrieval of raw data can last tens of minutes. A highly configurable client allows for information retrieval bypassing the interactive interface. This method is for example used by ALICE Offline to extract operational conditions after a run is completed. Last but not least, the software can be easily adopted to any underlying database structure and is therefore not limited to WINCCOA.

1. Introduction

The ALICE experiment at CERN [1] is dedicated to the study of ultrarelativistic heavy ion collisions provided by the Large Hadron Collider (LHC). To cope with the extreme track densities, 18 subdetectors based on different technologies have been deployed in ALICE. As a result, the Detector Control System (DCS) has to access large numbers of different components and associated data [2]. The whole DCS has been designed as a strictly hierarchical distributed system consisting of

[3] Author to whom any correspondence should be addressed (e-mail: slavka.jadlovska@tuke.sk)

autonomous detector systems responsible for detector control and monitoring. Most of the values monitored by the DCS are stored in a central database (the DCS archive) for later processing. Each detector uses an isolated database schema to avoid possible conflicts. The access to archived values requires a detailed knowledge of the detector structure, controls implementation and overall design of the distributed system. This knowledge is maintained in the central DCS, but is not easily accessible to external consumers. A dedicated *ALICE MANager for Dcs Archives*, known as AMANDA, has been implemented to serve as an interface between the proprietary DCS database and remote clients [3].

During the LHC RUN1, the first generation of the AMANDA software suite successfully served its purpose – an exchange of conditions data between the DCS and ALICE offline. It was though designed as a simple server/client tool without any ambition for wider use by interactive clients. The main design goal of the presented AMANDA 3 package is to provide a mechanism that allows for concurrent access to DCS archives using multiple clients, and consequently to improve data availability for more users.

2. ALICE Detector Control System

The ALICE Detector Control System (DCS) is an information and control system which ensures the safe and correct operation of the ALICE experiment. It is also responsible for the acquisition, processing and archivation of data for control, monitoring, and configuration purposes.

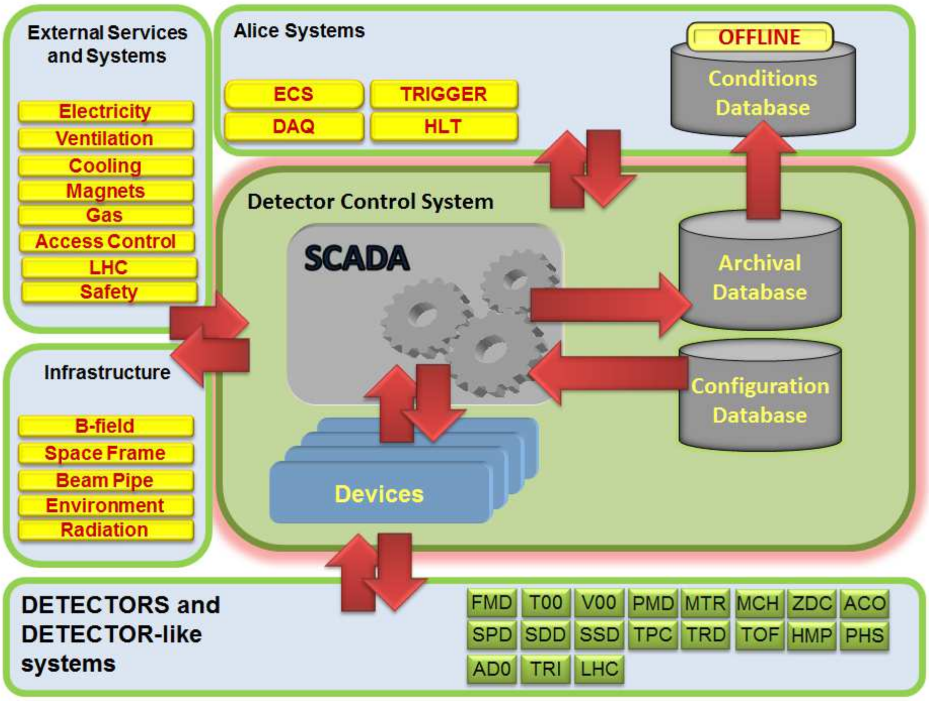

Figure 1. ALICE DCS architecture.

The main DCS data flow and interactions with external systems are shown in figure 1. The whole ALICE experiment consists of 18 subdetectors, each with a number of subsystems (high/low voltage, front-end electronics, cooling) that need to be configured and controlled. Additional information acquired from external systems is needed for the experiment operation. The DCS collects all necessary external parameters, distributes them to consumers and provides required feedback [2][4].

To allow for coherent and parallel operation of the ALICE equipment, the DCS is internally grouped into logical blocks. A set of devices such as PLCs and PC servers enable direct connection of subdetectors and actuators for the acquisition and preprocessing of monitored physical characteristics as well as direct experiment control. Hardware-wise, the DCS is composed of 1,200 network-connected devices and several hundreds of electronics modules. A variety of network interfaces (CANbus, JTAG, ProfiBus, RS-232, Ethernet) is used to communicate with the individual devices. The DCS supervises about one million parameters, with the typical readout frequency of ~1Hz.

The commercial product WINCC OA (*WINCC Open Architecture*) has been selected as the SCADA (*Supervisory Control and Data Acquisition*) system. Communication of WINCC OA with the commercial devices is based on the standard OPC (*OLE for Process Control*) protocol, ensuring easy integration with various control systems wherever possible. Custom modules are controlled via standardized drivers, based on CERN DIM (*Distributed Information Management*) toolkit. To set up and configure the experiment to requested run conditions, about 6GB of data is read from the *configuration database* and sent to front-end devices.

During a physics run, the DCS acquires about *300,000 values/s* from subdetectors, measuring equipment and external subsystems. The device drivers and software readout modules filter this data and forward only 10% (*30,000 values/s*) into WINCC OA. Values tagged for archival are then stored in the *archivation database* (*DCS Archive*) for further processing. The *archivation database* only accepts up to 0.33% of detected data (*1,000 values/s*) from the WINCC OA. After each run, a subset of this data, relevant for physics analysis, is copied from the archivation database into the *offline database* (*conditions database*) [2][5].

3. AMANDA 3 software suite overview

The interface between the archivation database and the ALICE offline (AMANDA) was originally designed as a C++ console application (AMANDA 2) with the independent server/client parts implemented over TCP/IP [3]. AMANDA 3 was developed by the authors of this paper as an upgrade to the original version to provide a novel approach to the task of retrieval of large amounts of DCS data and to enable faster access to selected data. The main design goal of AMANDA 3 was to optimize the transfer of data blocks with respect to the amount of data and the number of recipients: the upgraded version had to be able to select as much data as possible and transfer it to the highest possible number of users, while also avoiding system overload caused by repeated requests for the same data. Logging, monitoring and administrative possibilities needed to be expanded as well.

3.1. AMANDA 3 Server

To meet the presented requirements, the original single-server application was replaced by a decentralized system composed of several independent cooperating modules, written in C#. The process of data transfer and request handling has therefore been divided into smaller parts designed to perform specialized operations, and can be handled by independent processors on one or several networked computers. *Windows Communication Foundation* (*WCF*) service system was chosen to interconnect all decentralized parts in this design and to ensure fast and reliable mutual communication between modules. It is based on the multithread approach where separate threads are assigned to individual requests. All requests are therefore processed at the equivalent speed, which decreases as the data transfer rate approaches the maximum rate enabled by hardware.

An external signal from the client begins the life cycle of a request which then passes as a token between the individual modules. As shown in figure 2, web clients connecting to the server first access the *WebServer* module, which separates clients from the *AMANDA 3 Server internal network* connected to the database. The request coming from the *WebServer* is handled by the central coordination *Core* module which filters, evaluates and monitors client requests while acknowledging every significant step of request processing. The relevance of the module increases whenever separate computers or multiple services of the same type are involved.

The task of user authentication is delegated by *Core* to the *Auth* module, which can be integrated into the existing security and account system (e.g. a database with an existing list of users). Each user is assigned a unique combination of username/password and a role that grants them specific rights to access system functions or particular data.

Logging has been implemented in AMANDA 3 on several levels. Global logging of information with higher-level (administrator) significance is concentrated in the *LogS* module which accesses the database of logs. Each module also supports more detailed, lower-level logging into files stored on a local computer, which also serve as backup in case of database failure. Finally, logging of individual steps of request processing is supported so that the cause of a fault or alarm could be located and any required steps could consequently be applied.

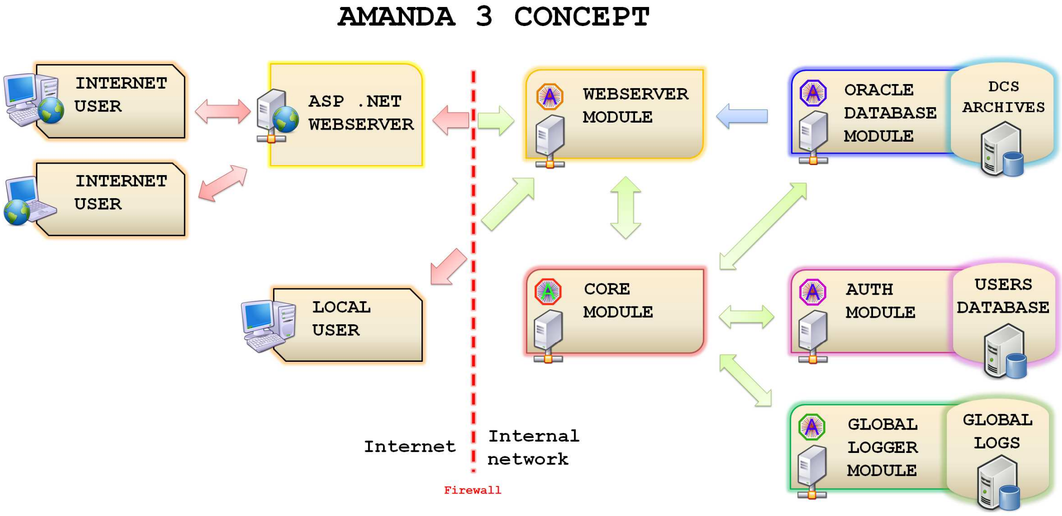

Figure 2. AMANDA 3 distributed system.

The *OrDB* module is the only module with direct access to the DCS database, and has the most demanding hardware requirements. The module accepts the request to select data according to user specifications from the *Core* module, and delivers the downloaded data directly to the *WebServer* service module. Predefined ORACLE procedures are used to ensure efficient transfers of large data blocks. Management of concurrent downloads is handled by employing separate independent threads to process individual requests, hence the client starts receiving packets immediately as the download session is launched. The unified packet format carries additional information apart from the data, such as the number of transferred entries. Consequently, the module estimates the sizes of transmitted data and computes transfer statistics.

3.2. Web client and AMANDA 3 administration tools

The presented modules are part of the *AMANDA 3 Server* internal network, whose role is to protect the system against external violation. The *WebServer* module serves as the gateway which separates client requests from the *Core* module and subsequently from the ALICE databases. The connected ASP.NET server hosts a client website. As a result, the client running on an arbitrary Web browser removes the host operating system dependency.

After the user successfully logs in, a protected website is opened where a request for data transfer can be specified depending on the user role. In the original implementation, the selection of required data was based on a user-given time interval and a list of requested parameters. This list was read by a client from dedicated database or from a text file and sent to AMANDA server. This method expected that a static list of all available parameters has been compiled beforehand and therefore significantly reduced the interactive use of AMANDA for debugging [3]. AMANDA 3 maintains this feature and adds a possibility to generate the request interactively, by browsing the available parameters directly in the archival database. A mechanism is being developed that will narrow the specification of data scope to make data transfer more open and flexible. A desktop client and a dynamic link library for AMANDA data transfer will be available as well.

In AMANDA 3, the data file is no longer downloaded into the local computer like before, but rather to the hard drive of the webserver where the user can access it through a public URL. After logging in, the user is informed which data is already available for download from the webserver. This allows multiple users to download data without repeatedly having to extract from the database. Given the capabilities of a WCF-based distributed system, a user is limited to a single active download session. It is though possible to launch a session and log off; at the next login, the user can check the download process, or access the retrieved file. Through the administrator interface of the web client, extended possibilities for data selection and file removal which affect all users are available.

4. Conclusion

The design goal of the upgraded AMANDA was to facilitate the transfer of large data blocks between the DCS archives of the ALICE experiment and offline clients. The main feature of the new version is the server's decentralized architecture – unlike the original console application, a system of hosted services is employed. As a result, the solution runs in the background without any user interaction, and restarts automatically after power failure. The original data exchange functionality is extended by advanced download possibilities, user management, security and logging. Specific tasks are scattered across multiple computers, which allows for load balancing and system jam prevention. Individual modules can also serve as a backup for others in case of connection loss.

The present modules are designed so that any added features will not affect the original functionality of the application in any way. In addition to the web client which successfully removes the host operating system dependency, dedicated clients will be available as well. The system tray application, which monitors the functionality of basic AMANDA components, will be replaced by a more effective web client and later by a desktop client, both in cooperation with future users. After the test operation, additional functions will be implemented with respect to future user requirements.

The AMANDA 3 package was developed as a part of the ALICE DCS running the WINCC OA. However, it can be adopted to any underlying database structure as an independent application that provides multiple users with fast access to large amounts of data. This approach is currently being tested using the Distributed Control System, which is the technological framework used by the Center of Modern Control Techniques and Industrial Informatics at the DCAI at FEEI, TU of Košice.

References

[1] The ALICE Collaboration 2008 The ALICE experiment at the CERN LHC *Journal of Instrumentation* **3** 260

[2] Chochula P, Jirdén L, Augustinus A, De Cataldo G, Torcato C, Rosinsky P, Wallet L, Boccioli M, Cardoso L 2010 The ALICE Detector Control System *IEEE Trans.on Nuclear Science* **57** 2

[3] Chochula P, Augustinus A, Fekete V, Jirdén L, Kapusta S, Rosinský P 2007 Handling large data amounts in ALICE DCS *Proc. of Int. Conf. on Accelerator and Large Experimental Physics Control Systems* (Knoxville, USA)

[4] Augustinus A, Chochula P, Jirdén L, Lechman M, Rosinský P, Pinazza O, De Cataldo G, Kurepin A N, Moreno A 2011 Computing architecture of the ALICE Detector Control System *Proc. of Int. Conf. on Accelerator and Large Experimental Physics Control Systems* (Grenoble, France)

[5] Kapusta S, Augustinus A, Chochula P, Jirdén L, Rosinský P 2007 Data flow in ALICE Detector Control System *Proc. of 11th Vienna Conf. on Instrumentation* (Vienna, Austria)