Upgrade of the ATLAS Control and Configuration Software for Run 2

Igor ALEKSANDROV[a], Gabriel ANDERS[b], Giuseppe AVOLIO[b], Mihai CAPRINI[c], Alina CORSO RADU[d], Matteo D'ASCANIO[e], Julio DE CASTRO VARGAS FERNANDES[f], Maria Teresa DOVA[g], Andrei KAZAROV[h], Nikolai KLOPOV[h], Bernard KOLOBARA[i], Vladislav KOTOV[a], Andrew LANKFORD[d], Florian LAURENT[j], Giovanna LEHMANN MIOTTO[b], Luca MAGNONI[b], Serguei OLECHKO[h], Lykourgos PAPAEVGENIOU[k], Yury RYABOV[h], Alejandro SANTOS[g], Diana SCANNICCHIO[d], Jose SEIXAS[f], Igor SOLOVIEV[d], Gokhan UNEL[d], Yoshiji YASU[l]

a: Joint Inst. for Nuclear Research – JINR (RU), b: CERN, c: IFIN-HH Bucharest (RO), d: University of California Irvine (US), e: Universita' degli Studi di Roma Tre (IT), f: Univ. Federal do Rio de Janeiro (BR), g: Universidad Nacional de La Plata (AR), h: B.P. Konstantinov Petersburg Nuclear Physics Institute – PNPI (RU), i: Ministere des Affaires Etrangeres et Europeennes (FR), j: École Polytechnique Fédérale de Lausanne (CH), k: University of Athens (GR), l: High Energy Accelerator Research Organization (JP)

Introduction

The trigger and data acquisition[1] (TDAQ) system of the ATLAS[2] experiment at the LHC is controlled and configured via a software infrastructure that takes care of coherently orchestrating the data taking, the Control and Configuration software. Its overall architecture, established at the end of the 90's, has proven to be solid and flexible, and has contributed to the experiment's excellent data taking efficiency and, therefore, its Physics research achievements. Nevertheless, many additional modifications were introduced in the software in the course of Run 1 (2008-2012) in order to fix bugs, enhance error detection and handling capabilities, automate procedures and accommodate other needs and user requests.

In the course of Run 1, software changes driven by operations left some parts of the code unclean and hard to maintain, such that a complete re-implementation of individual software components was deemed necessary by the start of the first **long shutdown** (LS1) of the accelerator.

In addition, in the course of Run 1 new requirements were defined. Some of those could not be implemented in a running system and/or with the available manpower. Those requirements were therefore addressed in LS1.

Last but not least, several IT technologies matured in the past few years: after careful evaluation we chose to introduce some of these in the Control and Configuration software. The choice of this new technology led to advantages in terms of user experience, development and maintenance costs.

This poster gives an overview of the main changes carried out on the Control and Configuration software during LS1. We will describe the new technologies that were adopted and the context in which they are used, present some of the main results obtained, and indicate the areas in which other upgrades may be envisaged in the future.

Core Modifications

- **Configuration Database[5]**

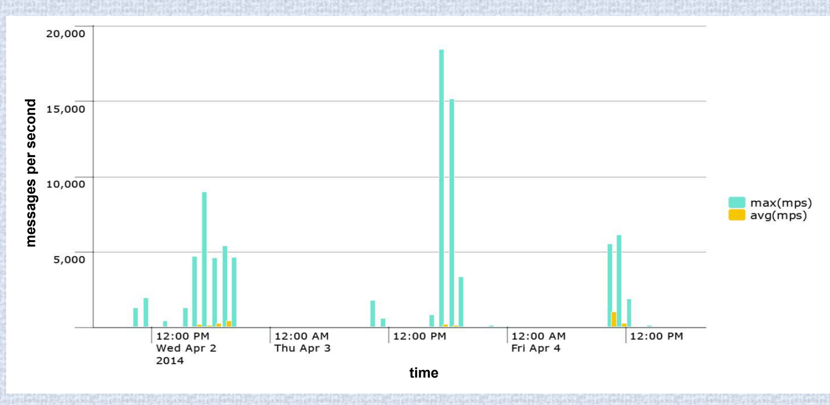
Databases schemas and algorithms have been modified to describe similar components by using few template objects. These changes have **increased the maintainability of the system** (editing one object instead of tens) and **reduced the overall configuration size** (a reduction factor of 6 can be noticed).

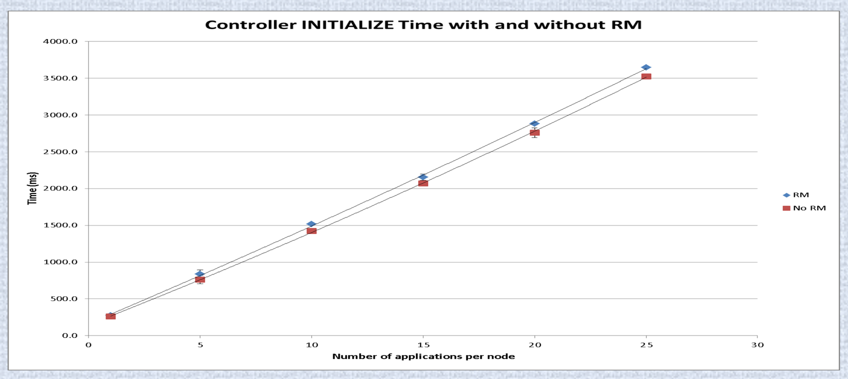
- **Test Manager[6]**

The Test Manager is a service devoted to the verification of the TDAQ system functioning by executing tests on request. This component has been completely re-designed to be configured from the configuration database. In order to obtain that, new features have been introduced: test policies, new types of tests (CORBA[17] interfaces), handling possible failures (**extended diagnostic and recovery scenarios related to them**).

- **Message Transport System[7]**

MTS underwent a review of the requirements that led to a complete redesign and new implementation to match its actual role (fast and reliable transport layer for TDAQ Error Reporting System[8] messages). The redesigned system is **reliable, scalable and its performance has been improved.** The plot shows the rates of messages reported in MTS in technical runs in April 2014. In the condition of the plot (60000 applications running) MTS reached a maximum rate of 18kHz of delivered messages



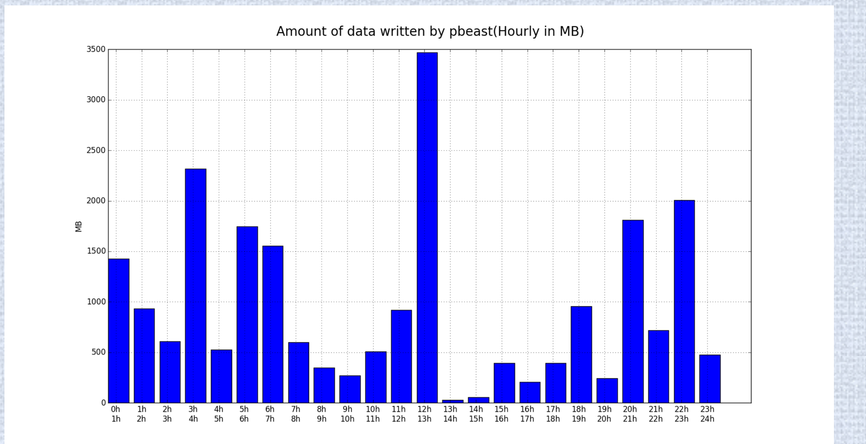


- **Resource Manager**

After an initial review and simplifications of the requirements, the system underwent partial changes with the introduction of Boost multi-index container. As a result **the code base has been reduced by 40%** against the previous implementation thus leading to **more maintainable system**. The plot shows that there is no overhead by the inclusion of the resource manager.

- **Information System Archiver, P-BEAST[9] (a Persistent Back-End for the ATLAS TDAQ)**

P-BEAST is a **new component** designed and implemented to archive operational monitoring information for analysis by experts. It provides CORBA and REST interfaces for data access. Its implementation is based on Google protobuf (data persistence), CORBA (internal protocol and user programming interface) and libmicrohttpd (Web server).

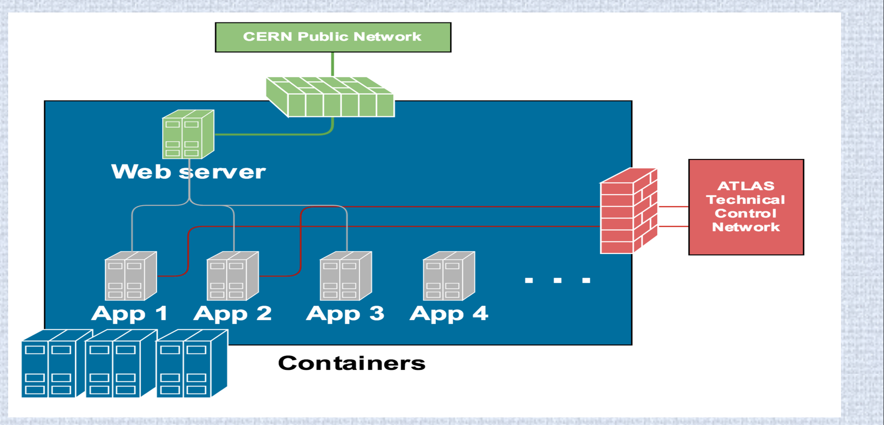


Web application deployment improvements

The main idea behind the new design is to isolate web applications inside a DMZ, limiting the access to resources located inside the ATLAS Technical Control Network . This arrangement ensures separation between Web and core applications.

To increase security applications that are running on the same host should be isolated from each other. Moreover, by not sharing the environment, different applications are allowed to use conflicting technologies. This can be achieved using virtual machines or more lightweight solution like Linux containers.

Dockerfiles[15] can be used to specify all requirements and build instructions for a Linux container. A tool like Puppet[16] can be used to set up environments inside a VM.



Conclusions

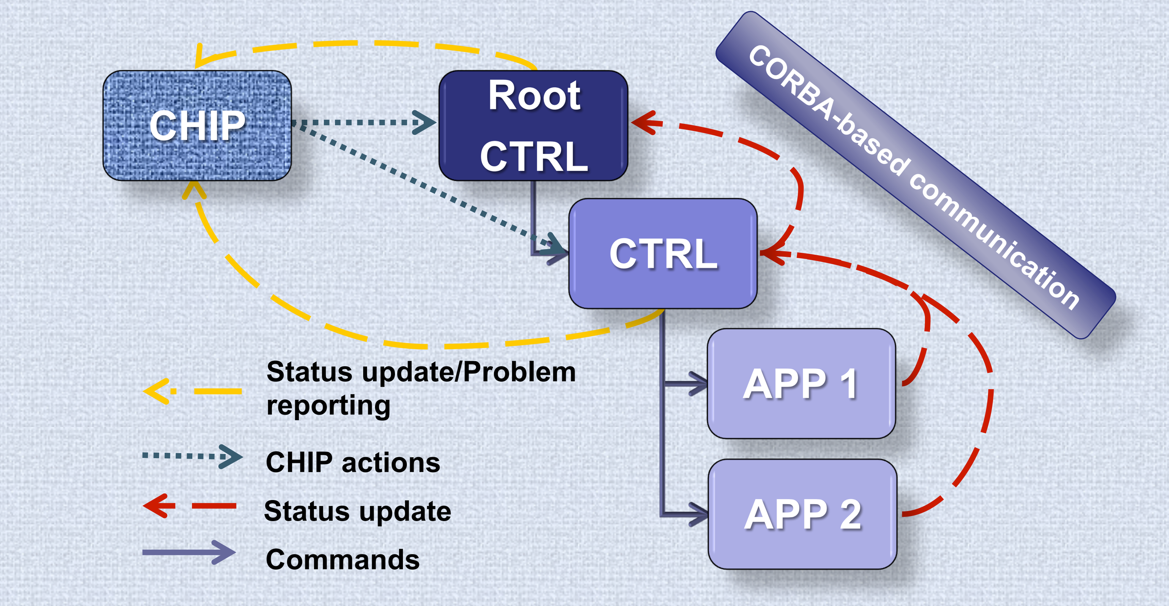
During the LS1 many components from the Control and Configuration Software have been improved or re-implemented in order to meet the requirements that arose during Run 1 and to keep the system aligned with newer technologies than those available during the first development of the system.

We believe the system is prepared to meet its requirements and needs of ATLAS for Run 2. An assessment of the system's performance will be done at the end of 2015, after the first months of physics data taking. This will be the occasion for planning any immediately required corrective actions of the Control ad Configuration software suite and also to start preparing the long term strategy for upgrades for Run 3.

Run Control & CHIP

The **Run Control[3]** (RC) and the **Central Hint and Information Processor[3]** (CHIP) are key components of the **Control and Configuration Software**.

The RC system steers the data acquisition by starting and stopping processes and by carrying all data-taking elements through well-defined states in a coherent way. During the LS1 the RC has been completely re-designed with state of the art C++ technologies such as boost[12] and Threading Building Blocks[21] (Intel's).



Given the size and complexity of the TDAQ system (2000+ PCs, 30000+ applications, 9000+ network ports,…), errors and failures are bound to happen and must be dealt with. The data acquisition system has to **recover from these errors promptly and effectively**, possibly without the need to stop data taking operations. To achieve this, **CHIP** has been introduced.

CHIP is an **intelligent system** having a global view on the TDAQ . It is based on a third party open source java based **Complex Event Processing (CEP)** engine, ESPER[4]. It aims to supervise ATLAS data acquisition by taking operational decisions and handling abnormal conditions.

Applications in the ATLAS TDAQ system are organized in a tree-like hierarchical structure (the **run control tree**), where each application is managed by a parent **Controller**. The topmost node of the tree is the **Root Controller**. Controller applications are responsible to keep the system in a coherent state by starting and stopping their child applications and by sending them the proper commands needed to reach a state suitable for data-taking. Controller applications are also the ones that interact with **CHIP** by informing it about any changes (their own or their children), allowing CHIP to take the needed actions.

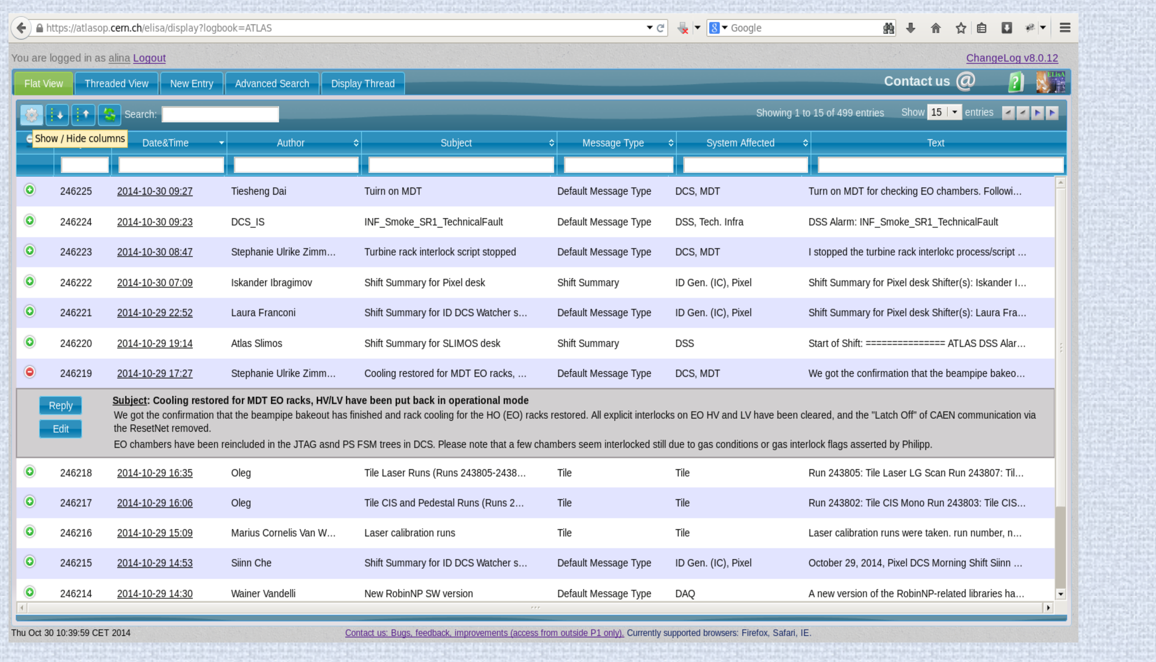
User Tools

- **P-BEAST Dashboard**

This web application provides an interface to visualize the amount of resources in use by TDAQ system through dashboards and plots, configurable by the user according to own needs. The data are provided by P-BEAST and the application is based on the Grafana[19] project and has been adapted to support a custom data source by AngularJS[17] framework.

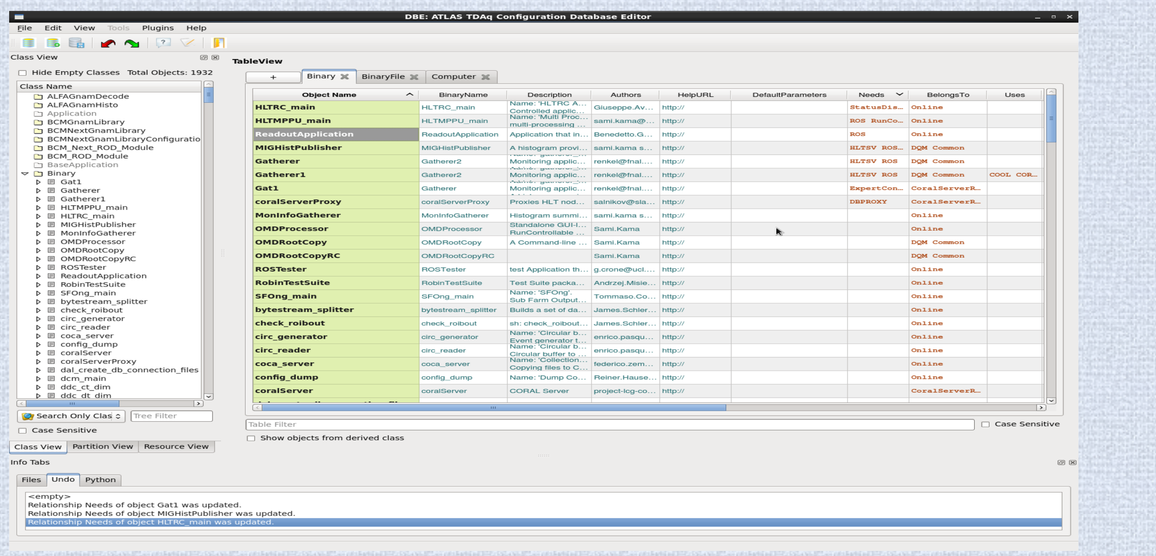
- **ELisA[10]**

The ATLAS electronic logbook (ELisA) is a web application used to record and share messages about ATLAS data taking activities by system operators, experts and automated services. The information is stored in an ORACLE database. The adoption of a MVC-driven architecture has allowed to focus code development on specific features of the project, while profiting from the reliability of established third-party technologies such as the Spring[13] framework. The tool provides as well an HTTP-based REST API for programmatic access to its features.



- **DBE**

This database editor application allows the user to create, modify and delete Configuration Databases. The service provides to the user the possibility to have multiple views on the data stored in the configuration database and configure its views to one's needs. It replaces the current MOTIF based implementation with a more modern alternative written in C++11 with the Qt[14] library .



- **SAReplay** *(Parallel session Track4, 4/14 17:00)*

It is a service to test new or modified Shifter Assistant[11] (an expert system which informs the status of ATLAS subsystems) directives by executing the system inside a sandbox, and using a configurable replica of past and archived data, events and configuration. This approach allows the validation of directives in a controlled and independent environment, keeping a history of past tests for further iterations and collaboration between developers. It is based on a software stack using a Oracle database, P-BEAST, and Java at the back-end, and Python and Django[20] at the Web user interface front-end.

References

1. ATLAS high-level trigger, data-acquisition and controls: Technical Design Report , 2002, The ATLAS Collaboration
2. The ATLAS Experiment at the CERN Large Hadron Collider, J. Instrum. 3, 2008, The ATLAS Collaboration
3. Intelligent Operations of the Data Acquisition System of the ATLAS Experiment at the LHC, 2014, G. Anders, G. Avolio, G. Lehmann Miotto, L. Magnoni
4. EsperTECH, http://esper.codehaus.org/ (May 2014)
5. Configuration and Control of the ATLAS Trigger and Data Acquisition, 2008 , Controls and Configuration(CC) working group
6. A Dynamic Test Management Framework for the ATLAS Experiment, 2014, G. Avolio, A. Corso-Radu, A. Kazarov, G. Lehmann Miotto, L. Papaevgeniou, I. Soloviev, G. Unel
7. A Scalable and Reliable Message Transport Service for the ATLAS Trigger and Data Acquisition System, 2014, A. Kazarov, M. Caprini, S. Kolos, G. Lehmann Miotto, I. Soloviev
8. The Error Reporting in the ATLAS TDAQ System, 2014, S. Kolos, A. Kazarov, L. Papaevgeniou
9. New Persistent Back-End for the ATLAS Online Information Service, 2014, I. Soloviev, A.Sicoe
10. The Electronic Logbook for the Information Storage of ATLAS Experiment at LHC (ELisA), J. Phys.: Conf. Ser. 396 (2012) 012014, A. Corso Radu, G. Lehmann Miotto, L. Magnoni
11. The AAL project: automated monitoring and intelligent analysis for the ATLAS data taking infrastructure, J. Phys.: Conf. Ser. 368 (2008) 012004, A. Kazarov, G. Lehmann Miotto, L. Magnoni
12. http://.boost.org
13. http://spring.io/
14. http://qt-project.org/
15. http://docker.com/
16. http://puppetlabs.com/
17. http://angularjs.org
18. http://corba.org
19. http://grafana.org
20. https://djangoproject.com
21. https://threadingbuildingblocks.org
22. https://www.gnu.org/software/libmicrohttpd/