STRATEGIES AND TOOLS FOR ATLAS ON-LINE MONITORING

W. Vandelli, University and INFN Pavia, Pavia, Italy on behalf of the ATLAS TDAQ MWG[*]
D. Burckhart, M.J. Costa, M. Hauschild, C. Padilla, T. Pauly, CERN, Geneva, Switzerland
M. L. Ferrer, K. Kordas, INFN Laboratori Nazionali di Frascati, Frascati, Italy
R. Ferrari, G. Gaudio, INFN Sezione di Pavia, Pavia, Italy
M. Della Pietra, INFN Sezione di Napoli, Napoli, Italy
M. Bosman, S. Sushkov, IFAE, Institut de Fisica de Altes Energies, UAB/Barcelona, Spain
M. Mineev, JINR, Dubna, Russia
J. Von Der Schmitt, MPI for Physics, Munich, Germany
M. Caprini, National Institute for Physics and Nuclear Engineering, Bucharest, Romania
P. Adragna[†], Queen Mary, University of London, London, UK
H. Hadavand, B. Kehoe, Southern Methodist University, Dallas, USA
S. Hillier, University of Birmingham, Birmingham, UK
I. Eschrich, S. Kolos[‡], University of California Irvine, Irvine, California, USA
D. Salvatore, Università della Calabria and INFN Cosenza, Cosenza, Italy
P.F. Zema, Università della Calabria and INFN Cosenza, Cosenza, Italy and CERN, Geneva, Switzerland
I. Riu, Université de Geneve, Geneva, Switzerland
A. Dotti, C. Roda, University and INFN Pisa, Pisa, Italy
I. Scholtes, University of Trier, Germany
R. Mcpherson, University of Victoria, Vancouver, Canada

Abstract

ATLAS [1] is one of the four experiments under construction along the Large Hadron Collider (LHC) ring at CERN. The LHC will produce interactions at a center of mass energy equal to $\sqrt{s}$=14 TeV at 40 MHz rate. The detector consists of more than 140 million electronic channels. The challenging experimental environment and the extreme detector complexity impose the necessity of a common scalable distributed monitoring framework, which can be tuned for the optimal use by different ATLAS subdetectors at the various levels of the ATLAS data flow. This note presents the architecture of this monitoring software framework and describes its current implementation, which has already been used at the ATLAS beam test activity in 2004. Preliminary performance results, obtained on a computer cluster consisting of 700 nodes, will also be presented, showing that the performance of the current implementation is in the range of the final ATLAS requirements.

ATLAS

The layout of the ATLAS experiment is driven by the magnet configuration: a solenoidal field in the inner cavity and a toroidal field for the muon spectrometer. Track measurement in the Inner Detector (ID) is obtained combining different techniques: high resolution silicon pixels (Pixel), silicon strips (SemiConductor Tracker, SCT) and straw tubes (Transition Radiation Tracker, TRT). A radiation-resistant highly-granular liquid-argon electromagnetic sampling calorimeter (LArg) followed by a sampling scintillator/lead hadronic section (TileCal) covers the central region. The LArg technology is also used for the hadronic calorimetry in the end-cap and forward regions. The calorimeter system is surrounded by a muon spectrometer which is built exploiting different technologies in order to obtain in each rapidity region high precision tracking (Monitored Drift Tubes, MDT and Cathode Strip Chambers, CSC) and fast response (Resistive Plate Chambers, RPC and Thin Gap Chambers, TGC), while maintaining a high enough radiation resistance. The whole ATLAS detector consists of about 140 million electronic channels. LHC will provide collisions with a frequency of 40 MHz and the output of the first level trigger will be less than 75 kHz. This frequency will be further reduced by the higher trigger levels and finally some hundreds of events will be selected and stored every second.

Considering the huge number of channels and the high event rate, a monitoring system is an essential tool to assess the status of the hardware and the quality of the data while they are being acquired.

ATLAS TDAQ

In ATLAS there are several levels of data flow [2]. Data are acquired by the front-end electronics (FE), located next to the detectors, and are collected, step by step, until the full event is assembled (see Fig. 1).

[*] http://atlas-tdaq-monitoring.web.cern.ch/atlas-tdaq-monitoring/
[†] Universit`a degli Studi di Siena and INFN Gruppo di Siena, Siena, Italy
[‡] on leave from PNPI, Gatchina, Russia

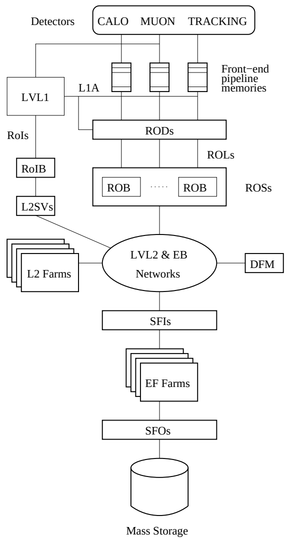

Figure 1: ATLAS TDAQ layout.

The level one trigger electronics (LVL1) is devoted to the first selection level; if an event is accepted, all the FE boards send the data to the Read Out Drivers (RODs), which are detector-specific custom modules. Each ROD is point-to-point connected to one Read Out Buffer (ROB) to which it sends detector-specific data.

The event is then assigned to one processing node of the second level trigger (LVL2) farms, which collects from the Read Out Systems (ROSs) the data fragments belonging to the detector regions selected by the LVL1 and starts its filtering algorithms. Accepted events are assigned to a Sub-Farm Input (SFI), which collects all the data fragments from the ROSs and assembles the complete event.

The last filtering stage is the Event Filter (EF), the second component, together with the LVL2, of the High-Level Trigger (HLT) sub-system. Built events are sent to EF farm processing nodes, in which a Processing Task (PT) completely reconstructs and analyzes the data with high-precision algorithms taken from the ATLAS off-line analysis framework (Athena). Events accepted by the EF are passed to the Sub-Farm Output (SFO) for the transmission to the mass storage.

The ATLAS TDAQ infrastructure is a large distributed environment, including thousands of computing nodes and custom modules.

MONITORING FRAMEWORK

In order to verify the good quality of the data sent to the permanent storage, the whole triggering system, the DAQ system and the ATLAS sub-detectors should be constantly monitored. To fulfill this mandate, the ATLAS monitoring system is organized as a distributed framework and includes several applications, ranging from low-level information-sharing components up to high-level graphical interfaces. This separation permits to isolate the problems and to optimize the applications for specific needs. The main programming languages used in the development of the monitoring components are C++ and Java.

On-line monitoring services

A fundamental feature provided by the monitoring framework is the routing of many sorts of data produced by the TDAQ components. These data may include simple parameters as well as more complex information, like histograms or event fragments. The on-line monitoring services [3], Information Service (IS), On-line Histogramming Service (OHS) and Event Monitoring (Emon), are devoted to this task. They provide different information-sharing channels abstracting the underlying complexity of the distributed environment and adopting network and CPU load minimization algorithms. Moreover, they give the possibility to perform the operational monitoring, namely the collection of many functional parameters published by hardware and software components, like busy statuses or data rates.

Information Service (IS) IS allows to share simple variables as well as user-defined data; it supports three main types of interactions: information providers can create, update or delete information, while information readers can get the value of the information. Moreover information subscribers can subscribe to the repository to be notified about changes.

In addition, through IS any application is able to send commands to any of the running Providers. This is useful to control the IS information flow: for example an application may ask a particular Provider to increase the frequency of information updates or to republish a particular information.

On-line Histogramming Service (OHS) OHS is based on IS and extends its functionalities to handle histogram objects, in particular raw and ROOT histograms. The OHS is not responsible for the booking, filling, storage and presentation of histograms.

Event Monitoring (Emon) Emon provides a framework to enable event sampling and distribution. User programs may request event fragments with selected properties, like trigger or sub-detector type, from a specific sampling point.

In order to minimize the load, requesting programs with the same selection criteria are arranged in a tree. Hence the sampling application forwards the events only to first the requester in any tree. The distribution of the data along the tree is done transparently to the users.

Monitoring Tasks

The possibility to analyze sampled events and produce histograms or other results is essential to assess the status of the detector and the functionality of HLT and DAQ sub-systems.

In the ATLAS monitoring framework, Monitoring Tasks supply this possibility, exploiting the on-line services to collect data and to make the results available.

Gnam Gnam [4] is a light-weight configurable framework for detector functionality monitoring that can be used to perform many sorts of jobs, thanks to a plug-in design that separates common actions and analysis algorithms, which are stored in dynamic libraries loaded at run-time.

Gnam can also handle asynchronous commands coming through the OHS to modify at run-time histogram properties or to execute custom functions defined in the analysis libraries.

Athena Monitoring The EF is a natural place to perform some monitoring activities: indeed events are completely reconstructed leading to the possibility to re-use the information, to monitor high-level physics quantities and to perform cross-detector checks, without requiring additional CPU power.

Since high-level physics monitoring at different level of the data flow is fundamental to check the filtering systems, the PT input-output system has been modified to transparently work with different data sources, also out of the EF framework. This led to Athena Monitoring [5], namely the possibility to exploit the off-line algorithms on data coming from any data flow step.

Gatherer In the LVL2 and EF farms many processing nodes run in parallel the same algorithms, therefore, to obtain a meaningful information, the histograms they produce have to be summed together. A specific configurable application, the Gatherer [6], will run in background, requesting from OHS the histograms published by the processing tasks and in turn publishing the summary histograms.

Graphical Interfaces

A complete monitoring system should also provide flexible and configurable GUI to allow a fast and user-friendly control of the status of the monitored items.

The ATLAS monitoring framework provides viewers for the on-line services: IS Monitor can show the content of IS servers, while OH Display permits to browse the histograms in the OH server, acting on them with all the ROOT graphic features. To assess the format of the data, Event Dump can sample events in the data flow and show them in structured tables.

Online Histogram Presenter (OHP)

OHP [4] is a highly configurable histogram presenter based on ROOT and Qt. OHP can operate in two different modes: it can browse the OHS and/or show a configurable set of on-line or reference histogram in a series of tabs. The two modes allow both the detector experts and the standard shifters to have all the needed functionalities within the same application. Furthermore ROOT context menus are available, enabling operations like fitting or zooming. Users can also send commands to the monitoring tasks using preconfigured buttons and panels.

FURTHER DEVELOPMENTS

Analysis framework

Due to the ATLAS complexity, a framework for reference histogram comparison, statistical checks and alarm generation is definitively needed. The development of such a framework is just starting now, collecting requirements, investigating software and experiences from other HEP experiments and implementing prototype functionalities in the existing tools.

Monitoring Data Archiving

In ATLAS it is foreseen to have tens of GB of monitoring results per run. Most of this data has to be stored and kept available for a limited time, in order to be able to cross-check the off-line analysis results with the on-line monitoring ones. Besides an archiving system will be needed to store reference histograms too. The monitoring system has then to provide common APIs and command-line tools to store, manage and retrieve the monitoring data.

The proposed architecture includes a local data cache, which holds monitoring results after the end of run, and an off-line archiver application, which stores data in the mass storage and register their location in a database. This design avoids a delay at the end of run transition.

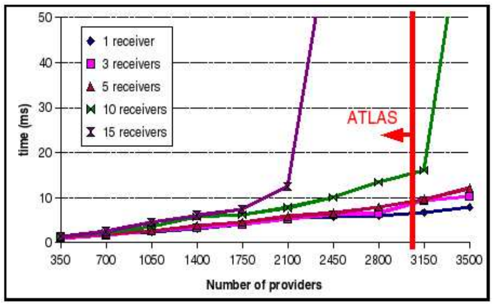

Figure 2: IS publishing time as a function of the numbers of providers and receivers.

LARGE SCALE TEST 2005 (LST)

In the summer 2005, tests of functionality of the HLT/DAQ system and of single selected components have been carried out using up to 700 nodes of the LXBATCH cluster at CERN [7]. IS, OHS and Emon components have been deeply tested with positive results: no failures were observed and the scalability of the architecture has been proven. Test conditions, in terms of service load, have been chosen to be as close as possible to the final ones.

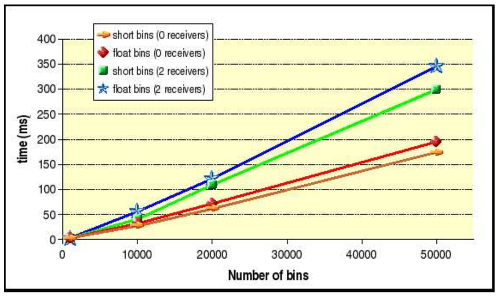

Figure 3: OHS publishing time as a function of the numbers of histogram bins and receivers.

Information Service

IS tests have been carried out using 350 nodes, with 1 to 10 providers per node, publishing 250 bytes of data, and with 1 to 15 receivers that read all the available information. In ATLAS about 100 receivers are foreseen which however collect different data subsets.

IS fulfilled the required performances (Fig. 2). The sudden increase of the time is due to the saturation of the Fast Ethernet bandwidth, whereas the proper ATLAS infrastructure will exploit a Gigabit network.

On-line Histogramming Service

OHS has been tested using 650 nodes with one provider per node publishing one histogram every 10 seconds, in the case of no receivers, or every 30 seconds in the case of two receivers. Receivers were receiving all the published histograms. Tests have been performed using different histogram sizes, ranging from few kB up to about one MB.

The service performed well (Fig 3), considering that, in the worst case, OHS was managing the publishing and the collection of roughly 650 MB every 30 seconds. However OHS allows readers to subscribe only to be notified about changes, avoiding unnecessary histogram transportations and therefore reducing the system load.

As for IS, in ATLAS there are expected to be more than two histogram receivers, but subscribing for distinct histogram subsets.

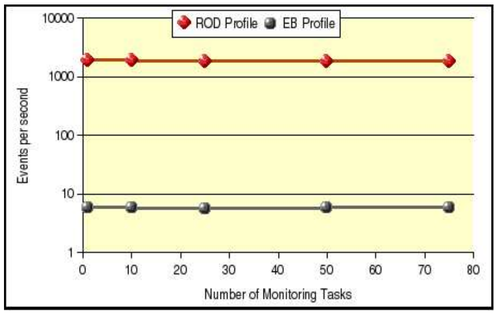

Figure 4: Emon event distribution rate as a function of the number of sampling channels.

Event Monitoring Service

The Emon performances have been checked in two different scenarios: using the smallest ATLAS sub-fragment (ROB profile, 2kB/event) or the largest one (Event Builder profile, 2MB/event). As can be seen in Fig. 4, Emon provides a constant event rate, regardless of the number of sampling channels, at least in the foreseen ATLAS working range.

Moreover during the test a negligible CPU utilization by the sampling thread has been observed.

CONCLUSIONS

A first implementation of the monitoring framework for ATLAS is already present, from fundamental communication services to high-level analysis and graphical applications.

Existing monitoring components were used during the Combined Test Beam 2004, where they proved to be extensible and configurable enough to satisfy the different sub-detectors needs and being a fundamental tool for the shift crew and the detector experts. Now, the same tools are used during the sub-detectors commissioning to assess the hardware status. Moreover, preliminary tests on a large distributed environment suggest that, at least for the fundamental services, the actual implementation is suitable for ATLAS

The development is going ahead, improving the existing applications but also adding new functionalities and features. This should lead to a complete monitoring system, as required in order to bring ATLAS to its maximum performance and exploit its discovery potential.

REFERENCES

[1] ATLAS Collaboration, "ATLAS Technical Proposal", CERN/LHHCC/94-43, LHCC/P2, CERN, Geneva, Switzerland, 1994

[2] ATLAS Collaboration, "ATLAS, High-Level Trigger, Data Acquisition and Controls", CERN/LHCC/2003-022, CERN, Geneva, Switzerland, 2003

[3] S. Kolos et al., "Online Monitoring software framework in the ATLAS experiment", CHEP 2003, La Jolla, California, USA, 2003

[4] P.F. Zema et al., "The GNAM monitoring system and the OHP histogram presenter for ATLAS", 14th IEEE-NPSS RTC 2005, Stockholm, Sweden, 2005

[5] M. Bosman et al.,"Development and Tests of the Event Filter for the ATLAS Experiment", 14th IEEE-NPSS RTC 2005, Stockholm, Sweden, 2005

[6] P. Conde-Muino et al., "Portable Gathering System for Monitoring and Online Calibration at ATLAS", CHEP 2004, Interlaken, Switzerland, 2004

[7] ATLAS HLT/DAQ developers and testers, "ATLAS DAQ/HLT Software Large Scale Functionality and Performance Tests July 2005 ", EDMS Note, ATL-D-TR-0003, CERN, Geneva, Switzerland, 2005