Journal of Physics: Conference Series



# Development, Validation and Integration of the ATLAS Trigger System Software in Run 2

To cite this article: Robert Keyes and on behalf of the ATLAS Collaboration 2017 *J. Phys.: Conf. Ser.* **898** 032008

IOP Conf. Series: Journal of Physics: Conf. Series **898** (2017) 032008 doi:10.1088/1742-6596/898/3/032008

# Development, Validation and Integration of the ATLAS Trigger System Software in Run 2

**Robert Keyes**[1] **on behalf of the ATLAS Collaboration**

[1]Department of Physics, McGill University, QC, Canada

E-mail: `robert.aidan.keyes@cern.ch`

**Abstract.** The trigger system of the ATLAS detector at the LHC is a combination of hardware, firmware, and software, associated to various sub-detectors that must seamlessly cooperate in order to select one collision of interest out of every 40,000 delivered by the LHC every millisecond. These proceedings discuss the challenges, organization and work flow of the ongoing trigger software development, validation, and deployment. The goal of this development is to ensure that the most up-to-date algorithms are used to optimize the performance of the experiment. The goal of the validation is to ensure the reliability and predictability of the software performance. Integration tests are carried out to ensure that the software deployed to the online trigger farm during data-taking run as desired. Trigger software is validated by emulating online conditions using a benchmark run and mimicking the reconstruction that occurs during normal data-taking. This exercise is computationally demanding and thus runs on the ATLAS high performance computing grid with high priority. Performance metrics ranging from low-level memory and CPU requirements, to distributions and efficiencies of high-level physics quantities are visualized and validated by a range of experts. This is a multifaceted critical task that ties together many aspects of the experimental effort and thus directly influences the overall performance of the ATLAS experiment.

## 1. Introduction

The ATLAS experiment is a multipurpose particle detector located at the Large Hadron Collider (LHC) at the European Center for Nuclear Research (CERN) in Geneva, Switzerland [1]. The detector itself consists of five main sub-detectors [2]. The inner detector (ID) is a precision silicon tracking detector located closest to the beam-pipe and housed inside a superconducting solenoid, whose purpose is to reconstruct tracks, primary vertices, and secondary decay vertices from heavy flavor particles. Surrounding the ID is a lead-liquid argon sampling electromagnetic (EM) calorimeter whose purpose is to measure electrons and photons. Surrounding the EM calorimeter is a steel/tungsten-scintillator sampling calorimeter whose main purpose is to measure hadronic signals (jets). The outermost layer of detectors is the muon spectrometer, consisting of a superconducting toroid magnet, fast response resistive plate chambers for triggering, and thin gap chambers for accurately measuring trajectories of muons. Finally, ATLAS has a variety of forward detectors whose purpose is to provide measurements of the delivered luminosity and inelastic/diffractive scattering events. From 2010 to 2013 (Run 1) the LHC provided proton-proton (pp) collisions at a center of mass energy of 7 and 8 TeV, heavy ion Pb+Pb collisions at 2.76 TeV, and heavy ion p+Pb collisions at 5.02 TeV. Following a long shutdown and upgrade period for both ATLAS and the LHC, in 2015 the LHC began providing pp collisions at a center of mass energy of 13 TeV and higher than ever instantaneous luminosities (Run 2). Throughout these periods, ATLAS was tasked with optimally recording data in these various collision environments using evolving hardware. ATLAS performed well yielding many interesting measurements, most notably the discovery of the Higgs Boson in 2012 and the observation of jet quenching in 2010 [3]. The ATLAS trigger system is the crux of this optimization problem as its job is to select 1 event of interest out of every 40,000 delivered by the LHC every millisecond. Considering that the rate of interesting events is $10^{14}$ smaller than the background rate ($10^{-5}$ Hz vs $10^{9}$ Hz), and that the scope of the ATLAS physics program is very broad, this is a monumental challenge. Every physics analysis hinges on the performance of the trigger as its online performance is irreversible. The trigger hardware settings and the trigger software undergo continuous scrutiny, development, and validation. This ongoing development and validation is a multi-pronged effort that requires coordination with online trigger operations, draws on various experts, developers and users, makes heavy use of computational resources, and makes use of various collaborative tools to facilitate its execution.

## 2. The ATLAS Detector Trigger System

The ATLAS trigger system is a combination of hardware- and software-based algorithmic decision making that takes the 40 MHz collision rate provided by the LHC and outputs recorded events at a rate of 1 kHz. The first stage of the trigger system, referred to as Level 1 (L1), is hardware/firmware based, making use of custom fast electronics and field programmable gate arrays [5]. The L1 system uses coarse granularity input signals from the muon and calorimeter systems to evaluate threshold and topological selection criteria, reducing the rate of accepted events to 100 kHz. The second stage of the trigger system, referred to as the High Level Trigger (HLT), applies software-based decision-making using reconstructed quantities that are either the same as or closely mimic those used in offline event reconstruction for data analysis [6]. The HLT software is processed on a trigger farm made up of roughly 40,000 CPUs in close proximity to the detector itself. The HLT software is developed by hundreds of developers that are often also involved in related physics analyses.

### *2.1. Trigger Signatures*

The development of HLT software is organized into working groups based on synergies between different detector subsystems and desired physics signals, which are known as trigger signature groups. Each signature group develops trigger software and online calibrations, validates online performance and data quality, develops offline reconstruction algorithms and calibrations for data analysis, and validates offline quality of reconstructed data. The ATLAS trigger signature groups include jets, missing transverse energy, calorimeter signals, electrons and photons, taus, B-physics (using muons), muons, b-jets (using tracks and vertices in the inner detector), the inner detector tracking, and minimum bias & forward detectors.

### *2.2. The Trigger Menu*

The trigger menu is the catalog of L1 and HLT selection algorithms and parameters that can be used in a given configuration [7]. There are various menus, the most heavily used being the physics menu for pp data-taking and the associated Monte Carlo (MC) menu for simulations. The menu must be synchronized with the algorithms contained in a given release, requiring coordination between signature groups (developers) and physics groups (end users). Each signature group must validate changes to their software and menu before they are integrated into the software release. This is accomplished by running compilation tests and processing tests with simulated and real data. New or modified menu items cannot be deployed without calculating the expected rate with which the items will accept events.

## 3. Trigger Software

### *3.1. Release Structure*

As depicted in Figure 1, the ATLAS HLT software releases are organised into a hierarchical structure with three levels and are developed within the current major release of the ATLAS analysis framework, Athena [8]. The "base" HLT release is built roughly twice per year to incorporate stable patches. The "patch" release incorporates updates that must be built on top of the base release, and is in fact the release that is used online. The "development" release incorporates all pending updates, is built on top of the current patch release, and is validated on a weekly basis. Once validated, the development release gets rolled into a new patch release that can be deployed for online use. The first number in the release name indicates the major release of the ATLAS software (for Run 2 this number is 20). The second number indicates the specific purpose of the software, in the present scheme 11 stands for the HLT online, 3 stands for reconstruction, and 7 stands for simulation. The third number corresponds to the base release, the fourth to the patch release, and then the fifth to the development release.

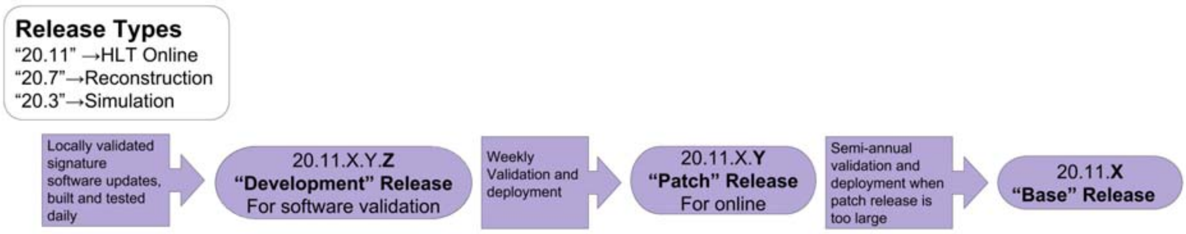


**Figure 1.** ATLAS software release structure

### *3.2. Trigger Hardware Emulation*

To validate a release before it is used online, the HLT software and menu are processed using special datasets designed to emulate the types of events processed online by the HLT. These datasets, referred to as Enhanced Bias (EB) datasets, are described in section 4.1. This data reprocessing is run on the Grid using the ATLAS production system [9] with a typical turn around time of less than 24 hours. As depicted in Figure 2, this emulation of a run produces RAW files in the same format as the data coming from the trigger system online, and from this point its treatment is the same. The raw data is processed using a release of the ATLAS production reconstruction software yielding output data that is in the same format as that of an online run and usable for validation. This entire process takes between 24 and 48 hours and typically requires on the order of 7000 hours of CPU, with each job requiring roughly 4 GB of virtual memory. At each stage of this process output metrics are produced to evaluate the performance of the software. This process is typically carried out for both the physics and MC menus.

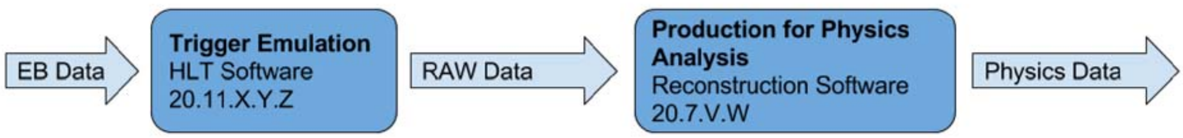


**Figure 2.** ATLAS HLT software flow using Enhanced Bias data and trigger hardware emulation.

### 3.3. Integration
An important aspect of trigger software development is ensuring seamless integration into the overall ATLAS detector data acquisition system. From the HLT computing farm nodes themselves, to the final reconstruction of the data, the HLT software must cooperate seamlessly and be harmonized with the hardware and software of these systems. Though they do not participate in the validation of the trigger software directly, behind the scenes there is an HLT integration coordinator who oversees the online software performance and scrutinizes online errors (debug events). This coordinator also works on planning and implementing upgrades to the various components of the data acquisition system.

## 4. Trigger Software Validation Strategies
### 4.1. Enhanced Bias Datasets
In order to measure the anticipated rate of a given HLT algorithm, in principle an unbiased sample of events is needed (zero bias). By definition, the purpose of most trigger algorithms is to select rare events, which means that an enormous sample of zero bias events would be needed to have sufficient statistics in the phase space where such rare events occur. Enhanced-bias (EB) datasets are created using L1 triggers and by collecting fewer events in the regions of trigger phase space with the more abundant statistics resulting in a more uniform statistical power across the spectrum of the trigger phase space [10]. The regions where fewer events are collected can then be corrected back to the zero-bias equivalent yield by reweighting the events according to the prescale (i.e. the number of rejected events for every accepted event) that was used to collect the dataset. Examples of L1 triggers that might be used to test jet triggers include a random trigger for the low jet transverse energy region, a medium threshold jet trigger seeded by a random trigger with a prescale for the medium jet transverse energy region, and a high threshold jet trigger without a prescale for the high jet transverse energy region. These compact EB datasets typically comprise one million events, requiring on the order of 24 hours to be fully reprocessed, and are used to test the performance of the trigger offline. EB datasets are taken for all types of run conditions provided by the LHC, and then validated and reprocessed to provide benchmark comparisons between trigger software releases.

### 4.2. Trigger Signature Performance Metrics
The following sections describe the performance metrics used to validate HLT software using EB datasets.

*4.2.1. Run Time and Memory Consumption* The first validation of a new trigger software release occurs by examining the top-level performance of the HLT software grid jobs. The distribution of the run time and memory consumption of the full batch of jobs provides a first check that the software performs as expected and is compatible with the HLT farm.

*4.2.2. Event Counts* Comparing the number of accepted events (event counts) of a given trigger to that of a previous release provides a top-level check on physics performance and expected impact on data-taking of the new trigger software. Before going into detail, the signature experts have an expectation of how a new release should impact the performance of their triggers. For instance, the electron/photon expert might expect an improvement in the electron trigger efficiency because of an update to the tracking algorithms. In this case they would anticipate an increase in the electron counts, and no effect on the photon counts. If this is not what they see, then they immediately know that further investigation is needed.

IOP Conf. Series: Journal of Physics: Conf. Series **898** (2017) 032008 doi:10.1088/1742-6596/898/3/032008

*4.2.3. Signature Performance* The physics performance of a new HLT software release is validated in detail by having experts from each signature group examine monitoring histograms, which include the distributions of input and discriminating variables used by their triggers. These distributions include efficiencies, hit maps, vertex resolutions, energy spectra, and more. A broad selection of comparison histograms (hundreds) is assembled, organized by signature, and uploaded to a web-interface so that the validation can be carried out by each expert using only a web browser. Automated comparison checks are performed based on distribution shape comparisons to the reference, providing flags that help the expert identify performance changes (distributions exhibiting changes are flagged as either red or yellow, unchanged distributions are green). Continuing with the example from the previous section, if the electron/photon expert expects an improvement in the electron trigger efficiency because of an update to the tracking algorithms, they would look in detail at the electron track resolution to evaluate the impact of the new software and decide if it meets the expectations.

*4.2.4. Rate and Cost predictions* It is also important to predict and check the trigger rates and compute resource usage of a software release. This is done using a technique called "cost monitoring" which provides a fine-grained breakdown of these metrics by individual triggers and algorithms. This data is collected online by sampling 10% of events regardless of whether or not they are accepted by the HLT. The same monitoring is performed offline when a new release is tested with the emulation described in section 3.2.

Determining of the expected trigger rates for a new software release requires knowledge of both the expected performance changes and the anticipated changes to the LHC collision parameters. For instance, at the beginning of a new run period such as the start of pp running in Run 2, as the machine luminosity increases so does the number of interactions per bunch crossing. A new software release might improve the robustness of jet calibrations in this busier environment, for example, and have a significant impact on the rate of accepted jet events. In order to anticipate the rate and to choose an optimal menu and prescales, EB runs with more interactions per bunch crossing will be processed and validated in advance. Typically at the start of a new run scheme, for which there are no prior EB datasets, the HLT is disabled while a L1 menu with conservative prescales is used. The first available EB run taken in the new scheme is then processed as quickly as possible (less than 24 hours) specifically such that the rates analysis can be carried out providing a definitive measure and prediction of the HLT performance. This EB run is then reused to evaluate and optimize the HLT performance for upcoming runs with similar collision parameters.

## 5. Trigger Software Validation Cycle

The trigger software validation cycle is typically carried every week for the most recent development release build. Additional validation cycles arise for special cases, such as to prepare for the change from pp to heavy ion runs, for new pile-up conditions, or specifically to test updated algorithms or new signature calibrations. This cycle requires a coordinated effort between many groups and associated experts. Figure 3 displays the involved experts and the cycle in its entirety. The process begins with the collection and assignment of bugs that arose in the previous validation cycle or through feedback from online usage. Next, the signature groups will create new software tags that address these issues. Under the instruction of the trigger operation coordinator, who considers the upcoming run plan and the time required to complete a validation cycle, the HLT software release expert collects all the new tags and builds the release. Next the menu expert will use the release to build an updated menu, and the release expert will deploy the new release to the grid using the CERN Virtual Machine File System (CVMFS) [11] in preparation for large scale reprocessing of EB data. Once the menus are prepared and the release is available, the software validation expert will proceed to launch the reprocessing of EB data to test the HLT software. This processing consists of three steps, first the processing of raw EB data emulating the HLT, followed by the reconstruction of the accepted data (mimicking the reconstruction chain during online data-taking), followed by production of output metrics for validation. Once the data has been processed, the software validation expert then makes available the output metrics to the signature experts and solicits their feedback. If sign-off is given by every signature group then the software release can be deployed for running in ATLAS, if not then new bug tickets are created and the process begins anew. Overall this cycle typically takes between three and seven days.

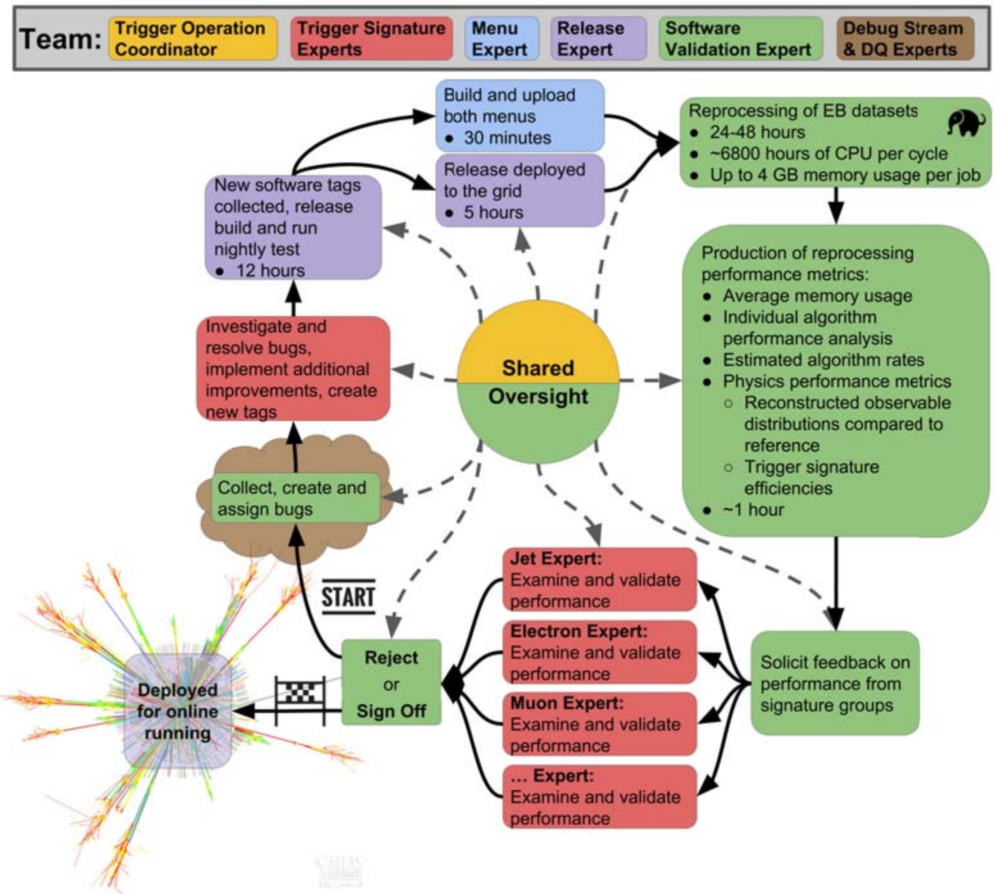


**Figure 3.** The HLT software validation cycle and participants.

## 6. Tools and Strategies

The trigger software validation is an ongoing effort that requires coordination with the online trigger operation and the various experts involved at different stages of the validation process, and a seamless transition from week to week between experts. This section highlights the main tools and strategies that are used to ensure the ongoing integrity of the validation process.

### *6.1. Daily Coordination Meetings*

Every day there is a general trigger operations coordination meeting that ties together all ongoing trigger-related tasks (online and offline). In this meeting an update of the software validation status is given, providing a forum for the coordinator to highlight important revelations, to solicit input or feedback on ongoing issues, and to solicit action by specific experts to expedite the validation cycle.

### *6.2. Jira*

Outside the daily coordination meeting, almost all interactions between experts involved in the trigger software validation effort occur through the issue- and project-tracking web-platform Jira [12]. Each reprocessing and validation of a given dataset with a given release and menu has its own Jira ticket. All the involved experts are "watchers" on the ticket, meaning that they receive email notifications for all actions on the ticket. The ticket is assigned to the relevant expert who is required to take the next action in the validation process, and once that action is finished, they assign it to the next one. The platform provides a discussion thread in which it is possible to tag other experts to solicit their input. The ticket contains a detailed description that describes its status and lists relevant links and comparisons. When specific issues arise, sub-tickets can be opened that can be followed up separately by the relevant working groups. Once all the feedback has been collected and the software validation has occurred, the ticket is closed, indicating the final status and conclusion of the reprocessing.

### *6.3. AMI and Prodsys2*

AMI, the ATLAS Meta-data Interface [13], is used by the software validation expert through a web interface to manage and store the configurations and parameters that steer the HLT and reconstruction software. Using this tool the expert can look up and compare existing configuration tags, and also clone them to avoid making new tags from scratch. Reprocessing tasks are built, launched, monitored and managed by the software validation expert through the Prodsys2 web interface [9]. Launching a reprocessing task involves setting many parameters and comes with some responsibility since the jobs use a large amount of grid resources with a high priority. The price paid for launching a bad reprocessing task is two-fold: wasting time in the validation cycle while ATLAS needs to optimize its data-taking, and wasting grid sources. Typical reprocessing jobs are configured in Prodsys2 to only fully launch once an initial small batch of test jobs has completed without error. This configuration ensures that grid resources are not wasted by paying the typically acceptable expense of some additional run time. Most reprocessing tasks launch successfully, however in the usual cycle it is acceptable for a new task's scout jobs to fail once or twice as the expert converges on the optimal the configuration. On rare occasion the test job step is not used when imminent feedback is required, necessitating that the utmost care is taken when configuring the tasks to ensure the jobs are correctly configured.

### *6.4. TWiki and Whiteboard*

Reprocessing experts take on the role of coordinating all the reprocessing and associated validations on a weekly basis. There is a pool of roughly 10 experts and a significant fraction of yearly turnover in this pool. The procedure of the role is always evolving, and is documented on a TWiki page by the lead expert, and a TWiki-based whiteboard by the experts themselves. Regular training sessions occur to ensure that newcomers can join throughout the year and be up to date with the most recent practices. The TWiki page contains detailed instructions outlining all the tasks of the role, while the whiteboard provides specific details related to current reprocessing practices, such as the preferred priority settings and job/sub-job merging parameters for optimal running on the grid, or specific Athena commands that must be included for different types of data reprocessing (pp vs. heavy ion for instance).

## 7. Conclusion and Outlook

Leading up to and throughout Run 2 there have been numerous improvements made to the ATLAS HLT software. At each stage, the software is rigorously tested and validated, leading to a smooth and reliable operation of the ATLAS trigger. Software validation on such a large scale requires seamless coordination between many groups and experts that must perform detailed tasks, and has been achieved by adhering to a continuous systematic validation cycle. Cornerstones of the cycle are that experts meet daily to coordinate their activities and interact continuously using the Jira software project management web-platform. Trigger software is validated by reprocessing EB datasets that emulate online trigger performance. The software validation expert specifically relies heavily on AMI and Prodsys2 to execute the reprocessing of these datasets on the grid, and then provides performance metrics for experts to validate. Though there are currently no significant improvements planned for the validation procedure, the upcoming migration to multi-core grid processing for faster reprocessing in conjunction with a general streamlining of the ATLAS data reconstruction process will shorten the turn around of the reprocessing tasks and in turn the validation cycle.

## References

[1] Evans L and Bryant P(editors) 2008 LHC machine, *JINST* **3** S08001.
[2] ATLAS Collaboration 2008 The ATLAS experiment at the CERN Large Hadron Collider, *JINST* **3** S08003.
[3] ATLAS Collaboration 2016 ATLAS public results https://twiki.cern.ch/twiki/bin/view/AtlasPublic/Publications.
[4] ATLAS Collaboration 2012 Performance of the ATLAS trigger system in 2010, *EPJC* **72** 1849.
[5] ATLAS Collaboration 1998 ATLAS level-1 trigger: Technical design report, CERN-LHCC-98-014 https://cds.cern.ch/record/381429.
[6] ATLAS Collaboration 2003 ATLAS high-level trigger, data-acquisition and controls: Technical Design Report, CERN-LHCC-2003-022 https://cds.cern.ch/record/616089.
[7] ATLAS Collaboration 2015 2015 start-up trigger menu and initial performance assessment of the ATLAS trigger using Run-2 data, ATL-DAQ-PUB-2016-001 https://cds.cern.ch/record/2136007.
[8] Calafiura P, Lavrijsen W, Leggett C, Marino M and Quarrie D 2004 The athena control framework in production, new developments and lessons learned, *CHEP Proc* 456-458.
[9] Borodin M, De K, Garcia J, Navarro, Golubkov D, Klimentov A, Maeno T, A Vaniachine on behalf of the ATLAS Collaboration 2015 Scaling up ATLAS production system for the LHC run 2 and beyond: project ProdSys2, *J. Phys.: Conf. Ser* **664** 062005.
[10] ATLAS Collaboration 2016 Trigger monitoring and rate predictions using enhanced bias data from the ATLAS detector at the LHC, ATL-DAQ-PUB-2016-002 https://cds.cern.ch/record/2223498.
[11] Buncic P, Aguado-Sanchez C, Blomer J and Harutyunyan A 2011 CernVM: Minimal maintenance approach to virtualization, *J. Phys.: Conf. Ser.* **331** 052004.
[12] Atlassian 2016 https://www.atlassian.com/software/jira.
[13] Albrand S, Doherty T, Fulachier J, Lambert F on behalf of the ATLAS Collaboration 2008 The ATLAS metadata interface, *J. Phys.: Conf. Ser.* **119** 072003.