Event Streaming in the Online System: Real-Time Organization of ATLAS Data

S. Klous, on behalf of the ATLAS collaboration

Abstract—In this article we describe the event streaming functionality as implemented in the ATLAS online system. Event streaming allows real-time classification and distribution of raw events into different subsets based on the decisions taken by the trigger system. Use cases, requirements and implementation of the main features are discussed. Furthermore, we present the ATLAS commissioning and monitoring activities related to event streaming.

I. INTRODUCTION

The Large Hadron Collider (LHC), currently being commissioned at CERN in Geneva, is a circular 27-kilometer-circumference machine, accelerating bunches of protons in opposite directions [1]. The first bunches circulated through LHC during a brief engineering run in September 2008. First collisions were provided at 450 GeV per beam at the end of November 2009 and high-energy collisions (3.5 TeV per beam) followed early 2010.

The bunches cross at four different interaction points, resulting in collisions with a center-of-mass energy of 7 TeV. The bunch-crossing frequency will be 40 MHz and on average 23 proton-proton interactions per bunch crossing are expected at a nominal instantaneous design luminosity of $10^{34}$ cm$^{-2}$s$^{-1}$.

Around the four interaction regions, large experiments are constructed to detect the many particles produced in the collisions. Due to technical and economical limitations, only a small part of the enormous amount of information collected by these experiments can be stored permanently. Advanced real-time selection systems, known as trigger systems, are installed by all experiments to select the most interesting information for storage and further study.

ATLAS [2], the largest LHC experiment, is a general purpose hermetic detector, with a design that takes physics goals, as well as technical and financial constraints into account. On each bunch crossing, the ATLAS detector records the signals induced by particles traversing the detector components. When this happens a total of about 1.5 MB of data are collected to determine the properties of the event, e.g., the type and energy of the detected particles. This results in an online data rate of about 60 TB/s flowing out of the detector. Note that the available event storage space is limited to about 6 PB per year [3] with an ATLAS operational period of about $2 \times 10^7$ s per year [4]. As a result, the recording rate is limited to roughly 300 MB/s, requiring a data reduction factor of 200,000. This reduction is achieved with the trigger and data acquisition (TDAQ) system.

Events included in the recording rate budget are already subdivided and organized by ATLAS during data acquisition. This procedure is known as “event streaming”, which is defined as “the splitting of raw data into logical entities by the online system based on event content”. So, the TDAQ system not only takes care of data reduction, but also organizes the collected events. Due to the central role it plays in event streaming, the TDAQ system is introduced in detail below.

A. The Trigger and Data Acquisition System

The ATLAS collaboration has designed a data driven multi-level trigger system that selects the most interesting events in real-time [4]. This system should be highly reliable and efficient, since failures or inefficiencies result in the loss of valuable data. Furthermore, the system should be modular, so it can be split into several independent and fully-functional entities. This facilitates parallel debugging and commissioning operations for detector components and the online system. A schematic diagram of the trigger and data acquisition system is presented in Fig. 1.

The first level trigger (LVL1) [5] is implemented in custom hardware, enabling selection of interesting events and reduction of the event rate from the bunch crossing rate of 40 MHz to a design value of 75 kHz, upgradable to 100 kHz. Information from 160 different input channels is combined to decide if an event is accepted or not. The LVL1 input signals can be generated by the subdetector systems or by external systems (e.g., beam pick-up or calibration instruments). Note that some of the subdetectors are able to trigger a LVL1 accept and others are not. In Fig.1, the inputs from the detectors are therefore divided into a triggering and a non-triggering group. The LVL1 is able to identify geometrical Regions of Interest (RoIs) in the detector. This feature allows the LVL1 to accept events based on the number of RoIs and hence the particle candidate multiplicity.

On a LVL1 accept, data are transferred from all detector Front-end Electronics (FE) via Read Out Drivers (ROD) to about 1600 Read Out Buffers (ROB). This allows further access to the detector data. The RoIs are organized, formatted and distributed to the LVL2 nodes by the RoI builder. The LVL2 selection algorithms request only partial event data associated with the LVL1 RoIs via about 150 Read Out Subsystems (ROS). This allows ATLAS to handle the LVL1 event accept rate with a standard gigabit Ethernet network.

The second trigger level (LVL2) [6] is software based. The LVL2 processing algorithms run on a computer farm with a

1Note that losing data is bad, but not knowing how much is lost is even worse.
baseline configuration of 500 quad core dual CPU nodes at a clock frequency of approximately 2 GHz. After a LVL2 accept about 100 applications running on dedicated computers, called Subfarm Input (SFI) nodes, pull the required data from the ROSs over the event builder network. The collected data is assembled and made available to the third level trigger, the Event Filter (EF), in a procedure known as “event building”. The EF baseline configuration consists of a cluster of 1800 quad core dual CPU nodes with a clock frequency of about 2 GHz. It is the only trigger level that deals with assembled events. The LVL2 and the EF are collectively known as High Level Trigger (HLT).

The HLT is able to make decisions based on complex algorithms, which are error-prone and resource intensive. The data acquisition (DAQ) components on the other hand are supposed to be light weight and robust. A thin software layer, known as “steering” [7], separates the algorithms running on the hardware depicted on the left hand side of Fig. 1 from the data acquisition path on the right hand side. The strict separation allows the data acquisition system to recover from, e.g., algorithm failures or timeouts.

Events with various kinds of characteristics are selected based on the results of algorithms running on both HLT levels, LVL2 and EF. Many of these algorithms run in series and/or in parallel, resulting in a list of trigger decisions for each event. This list is recorded as part of the event data and can be read back during offline processing. A set of LVL2 or EF algorithms running in series is called a LVL2 or EF chain respectively. A complete set of trigger items and chains at L1, L2, and EF is called a “trigger menu”.

The Sub Farm Output (SFO) computers are the final elements of the online system. They write data onto RAID arrays and serve as proxy and/or gateway to the offline mass storage facility. The RAID arrays provide a buffer of about 24 hours in case access to the mass storage facility is disrupted.

The trigger and data acquisition system has a significant impact on the event streaming strategy of ATLAS and vice versa. On the one hand technical constraints of the trigger and data acquisition system have to be taken into account. On the other hand, new features are introduced in the online system to allow for and exploit the classification and organization of events. The different kinds of events and their use cases are presented in the next section, in order to identify the set of required features for the online system.

II. CLASSIFICATION AND ORGANIZATION OF EVENTS

In ATLAS four different types of events are included in the recording rate budget of 300 MB/s. They are shown at the bottom of Fig. 1, distinguished by the purpose they serve:

Calibration events are triggered by detectors or by algorithms that focus on specific detector or HLT features. Depending on their purpose, the information needed from these events can be as small as 1 - 100 kB/event. They are intended for detector or HLT studies, e.g., signal shapes from Front-end Electronics, detector alignment, or resource utilization.

Physics events contain interaction data considered of interest for physics studies, like the production of a high-energy photon, muon or jet. Physics events contain full detector information and are expected to dominate in terms of processing, bandwidth and storage requirements.

Express events are a subset of the physics events. They are reconstructed offline in quasi-real time and looked at promptly. These events are used to check the data quality, monitor the status of the detector and calculate calibration constants. These results are used for the reconstruction of the physics event sample.

Debug events are events with errors that prevent further processing. These events can not be classified by the online system and need to be kept and handled carefully offline. They enable identification and fixing of problems in the trigger and data acquisition system that caused the errors.

Events of different types should not be mixed on mass storage. For example, calibration events should not be retrieved from mass storage for physics analysis. Debug events should be kept separate from the rest of the events to minimize the risk of crashes during reconstruction and to allow for efficient debugging by the appropriate experts.

In practice it is useful to further differentiate events within one event type. For example, splitting physics events into multiple subsets can facilitate a fast reconstruction turn-around for particular subsets of the data despite huge total data volumes. It helps prioritize (re-)processing by partitioning the large volume of raw data into logical entities. Calibration
A. Event Streaming Requirements in ATLAS

In ATLAS, streaming is based on trigger decisions, which has a couple of advantages. First of all, since trigger decisions associated with an event are immutable, so are the stream assignments. As explained above, this avoids recollection and redistribution of raw data at a later stage. Furthermore, streaming is performed at the end of the data acquisition chain, where trigger decisions provide the best available summary information about the content of the event.

On top of the distinction between different event types, ATLAS provides the flexibility to allow for multiple streams, per event type. The number of streams per type and their exact definition in terms of trigger decisions is configurable. Typically a maximum of 10 streams per type is expected. The overlap between streams (i.e. events assigned to more than one stream) can be dealt with in two different ways. Figure 2 shows an example with just two "signatures", where a signature is a characteristic of the event that makes it pass a specific trigger decision, like a high momentum muon candidate or a large energy deposition in the calorimeters. The two different ways of solving the overlap are:

- **Exclusive streaming**: Events that are assigned to more than one stream go into a single overlap stream. The philosophy behind this choice is that working on different streams is like working on different data samples. Each stream should be self containing. In case of a single stream analysis, this is the easiest method (there is no need to worry about events missing from the stream). Storage space consumption increases proportionally to the overlap fraction.

- **Inclusive streaming**: A copy of the event appears in each stream it is assigned to. The philosophy behind this choice is that working on different streams is like working on different data samples. Each stream should be self containing. In case of a single stream analysis, this is the easiest method (there is no need to worry about events missing from the stream). Storage space consumption increases proportionally to the overlap fraction.

Since most of the analyses will be single stream, inclusive streaming is preferred. Extensive studies showed that the expected overlap at nominal luminosity is of the order of 10%, which is considered acceptable. In case of excessive overlap the streams will be redefined. Exclusive streaming will only be considered if no definitions can be made with acceptable overlap fractions.

An event can be assigned to multiple streams of different types, where the type of a stream is defined by the event type it holds. Assignments of events to streams of one type are independent from assignments to streams of another type. So, an event assigned to calibration streams and physics streams, will be put into the calibration streams as well as in the physics streams. Note that assignment to different types is redundant, both in the inclusive and exclusive streaming scenarios. It is envisaged to configure a hybrid system, where an event is for example assigned inclusively to multiple calibration streams and exclusively to the overlap physics stream.

The debug stream requires special attention. Events with a processing error end up in the debug stream only. Nevertheless, since the system is well understood one might wish to write these events to the other assigned streams as well. Hence, each trigger level should be configured to either respect or ignore assignments to other stream types besides the assignment to a
Physics data streams are subdivided in files covering short (few minutes) time intervals of approximately constant instantaneous luminosity and data taking conditions, like detector status and trigger menu, called luminosity blocks. This facilitates subsequent efficiency and luminosity calculation under varying running conditions and acts as a safeguard against large portions of recorded data becoming unusable for analysis. Apart from luminosity block boundaries, files are closed when the maximum configured file size is reached. Since other streams are not used for physics analysis, they do not need division by luminosity block, so files respect only the file-size limits. The result is a mixed configuration as shown in Fig. 3.

### IV. Impact of Streaming on the Online System

As explained in section I-A, the steering decouples the HLT algorithms from the data acquisition. This modularity has to be respected by the event streaming implementation. A single communication vehicle is introduced between the steering and the data acquisition system, known as "the StreamTag". All assigned StreamTags are recorded in the event header by the data acquisition system. So, each event contains a list of one or more StreamTags, which determines the destination stream(s) of the event data. This decouples the streaming functionality in the data acquisition from the algorithm results and makes it fully data driven.

The StreamTag (see Fig. 4) contains a number of fields with different purposes. The trigger decisions for an event are mapped to a label, which is stored in the name field of the StreamTag. The mapping between trigger chains and StreamTags is part of the trigger configuration. This configuration is stored in a dedicated trigger configuration database (known as "TriggerDB"), which records and manages the evolution of menus and stream definitions [8].

One or more HLT chains can be associated with a single stream (described by a StreamTag name and type). For example, EF chains EFe20 (electron with $p_T > 20\text{GeV}$) and EFe15i (isolated electron with $p_T > 15\text{GeV}$) will both be associated with the "egamma" physics stream. At each trigger level, the steering creates the StreamTags of the event based on the trigger decision results. In the previous example, if either EF$_{e20}$ or EF$_{e15i}$ accepts the event, a "egamma" physics StreamTag is added to the event header.

The actual event streaming is performed at the end of the data acquisition chain, by the SFOs. They identify the StreamTag(s) in the event header and store the events in matching files. Note that each SFO opens its own files, the content written by different SFOs is merged during first pass reconstruction.

In the TriggerDB, each stream can be configured to respect luminosity block boundaries or not. This information is passed to the data acquisition system via the lumi field in the StreamTag. In case a stream respects luminosity block boundaries, a change in luminosity block number triggers the SFOs to create new files. Otherwise, the SFOs only create new files when a limit is reached in file size or number of events.

Since the data acquisition system is asynchronous, the SFOs have a configurable grace period to allow for late arrival of events from previous luminosity blocks. Files of a luminosity block are only closed after this period expires. Events that arrive at the SFO after the files of the corresponding luminosity block have been closed are sent to a debug stream for late arriving events.

In addition to the name and lumi fields, each StreamTag contains a type field with a value that can be one of the following: "physics", "calibration", "express", "debug" or "unknown". So, many different StreamTag names can be of the same type. The type field has been introduced in order to optimize online resource and storage space consumption. Specifically, a number of features are controlled through the content of the type field:

- The SFOs can be configured to stream events inclusively or exclusively per type. This makes it possible to configure for example exclusive streaming for physics streams
and inclusive streaming for calibration streams.

- Early identification of event type makes it possible to assemble only the relevant information for events of calibration type. This feature is known in ATLAS as "partial event building" (PEB) and is explained in detail in section IV-A.

- The event path through the online system is optimized for different event types. This feature is known in ATLAS as "event routing" and is explained in detail in section IV-B.

### A. Partial Event Building

A large amount of events will be needed for fast and precise calibration of the ATLAS subdetectors. The sole purpose for most of these events is to calibrate the detector, they are not going to be used for physics studies. They are only sent to calibration streams, so they just contain the relevant information for calibration. The rest of the event information is discarded to optimize network load and storage space consumption.

Building an event with a sub-set of detector data is called "partial event building", which is a unique feature of the ATLAS event building system. The size of partially built events is usually significantly smaller (orders of magnitude) than the full event size would be. Partial event building is possible by virtue of the pull protocol chosen for event building. An SFI can build partial events by requesting data only from those ROSs that contain relevant information. This results in an economic use of available resources. Large partial event building rates can be achieved without compromising the building of full events for physics analyses [9].

The partial event building functionality is split into a decision making part and a data acquisition part, reflecting the HLT structure. A LVL2 algorithm selects events for calibration of a given subdetector and creates a list of data to be stored. The list is passed to an SFI, where the appropriate fragments are collected and assembled. Each sub-detector can have different event selection requirements, so a number of customized LVL2 algorithms can exist and run in parallel, matching the needs of the different subdetectors.

### B. Event Routing

The stream type determines the storage, processing, and replication of the data offline. In addition events are routed through the online system based on the content of the type field, added at the end of LVL2 processing and added or modified at the end of EF processing. This allows us to minimize resource consumption in the online system by applying the following kind of optimizations (a flow scheme is presented in Fig. 5):

- Any event causing a processing error in the online system (related to algorithms or data acquisition) will get a StreamTag with its type set to debug. Most likely these events would cause more problems and timeouts in subsequent processing steps. Hence, events with a debug tag at the LVL2 are, after event building, routed directly to mass storage without processing at the EF. The data acquisition components are robust enough to handle these events and route them to mass storage independent of any data corruption inside the event.

- When only calibration triggers fire in LVL1, all LVL2 physics and calibration algorithms are bypassed and dedicated LVL2 chains are activated. These chains do not process any other event information. They just setup the information needed for streaming and partial event building. The data acquisition system routes these events directly to the SFO, bypassing the EF processing.

- The events that fired LVL1 physics triggers are processed by the LVL2 physics and calibration algorithms. After that, some of them are rejected for physics studies but accepted for calibration purposes. The assigned StreamTag type for these events is calibration and they will only contain partial event information. These events do not need further online processing after LVL2. The data acquisition system routes them directly to the SFO, bypassing the EF processing.

- Events tagged with physics type at LVL2 require full HLT processing. The data acquisition components route them to the EF steering and they are processed by the EF algorithms. The EF steering assigns new StreamTags based on the EF algorithm results and the LVL2 StreamTags are deleted, i.e., the StreamTag history is not kept. Note though that this information remains traceable through the recorded LVL2 trigger decisions.

- The LVL2 is able to assign multiple StreamTags of different types to an event. Hence an event that enters the EF can have two (or more) StreamTags, e.g., one of calibration type and one of physics type. These events must be processed by the EF physics algorithms, where the physics type StreamTags are reconsidered. The calibration type StreamTags are forwarded by the EF steering without further processing. As a result, the event in the example will at least have a calibration StreamTag and possibly a physics StreamTag (depending on the EF algorithm decisions) when it arrives at the SFO. Events with both physics and calibration StreamTags assigned at LVL2 are fully built. Their size is reduced at the
earliest possible convenience in a procedure known as "stripping". There are two options:

- When the event is rejected by the EF physics algorithms, the physics StreamTag is deleted and the event is stripped before it is sent to the SFO (only leaving the calibration parts) to reduce the network load and the CPU load on the SFOs.
- When the event is accepted by the EF physics algorithms, the physics StreamTag is kept and the SFO receives a full event. Full events are written to the physics stream(s) and stripped down copies to the calibration streams.

Event routing is a data acquisition feature. Hence, failure will directly affect the reliability and efficiency of event collection in the ATLAS experiment. Reliability is achieved with the data driven and modular StreamTag approach, as explained in section IV. Data acquisition components read the content of the StreamTag type fields in the event header (and write a StreamTag with debug type if necessary) to drive an event through the data acquisition system.

The StreamTag configuration for data acquisition components is completely decoupled from the algorithm part in the trigger database. The mapping of StreamTag types to routes is stored in a dedicated database (the DAQ database) that describes all relations of the data acquisition infrastructure. The configuration is much more restrictive than the one in the trigger database.

Consistent definition of the StreamTag type field contents amongst the two databases is crucial to avoid data loss due to misconfiguration. A limited set of keywords is allowed in the type field to enforce data acquisition consistency. Inconsistent event types are silently converted into unknown, which guarantees backward compatibility and allows to deal with incompatible data. Data acquisition components handle events of unknown type as an error condition. Hence, these events end up in the debug stream and they are not lost.

The StreamTag cannot handle sudden introduction of new types without code changes as a result of the limited set of keywords allowed in the type field. Limited flexibility is introduced with the definition of an additional type, called reserved. The reserved type can be assigned temporarily until the new type is introduced in a subsequent software version.

V. TESTING AND COMMISSIONING

Many components in the online and offline software provide features related to event streaming, making it a challenging project from integration perspective. Mistakes found in integration tests included:

- Misconfiguration in the trigger menus, calibration triggers are accidentally assigned to physics streams and vice versa.
- Inconsistent definition of the luminosity boundary StreamTag field, i.e. some streams within one type do and others don’t respect luminosity block boundaries, which causes errors in case of exclusive streaming (due to the overlap stream).

- Mismatches between StreamTag definitions in online and offline software. All components should inherit a common StreamTag definition in order to avoid such mismatches.

These issues demonstrate that thorough commissioning and monitoring are crucial for a reliable streaming implementation, as is the definition of proper operating procedures. These are the topics discussed in the following sections.

A. Commissioning

ATLAS followed a generic commissioning procedure for new trigger and data acquisition features. The new features were first introduced in technical runs. These were runs conducted with the HLT system every few months lasting one to two weeks. During these runs the output from LVL1 and the interface between LVL1 and LVL2 were simulated. The activities resembled the ATLAS running mode with a fully operational LHC. The technical runs included all available trigger and data acquisition hardware and aimed to verify new functionality. Routing and streaming, including partial event building, was successfully introduced during the April 2008 technical run and has been an integral part of the data acquisition system ever since.

The technical runs were interleaved with detector commissioning runs to exercise the full data recording chain, from detector to disk, including the TDAQ system. In July 2008 ATLAS collected cosmic ray data in a set of combined runs. The following subdetectors were fully or partially included in the read-out: silicon strip tracker, transition radiation tracker, electromagnetic calorimeter, hadronic calorimeter, and muon chambers. The LVL1 menu contained triggers from the muon system and the calorimeters [10]. The additional advantage of the commissioning run with respect to the technical run, from a routing and streaming perspective, was the opportunity to test handling of realistic error conditions.

Routing and streaming strategies as outlined in this paper were tested and debugged in most of their aspects during the commissioning runs. Physics, calibration and error stream treatment was thoroughly exercised. In total about 6 million events were collected during cosmic ray commissioning. Almost 500.000 of these went to the debug streams, about 8% of the total. In most cases (80%), this was the result of timeouts in the trigger algorithms or the read-out system [10]. Another significant fraction (20%) was caused by errors in the HLT electron and photon reconstruction algorithms.

B. Event Recovery

A new offline failure analysis and recovery framework has been put in place in July 2008 to analyze the contents of the debug streams in quasi real-time [11]. The computing power for performing these tasks is provided by a general purpose batch facility, from which about 50 cores can be assigned simultaneously to the analysis of the debug stream. Feedback could be provided before the end of a run, e.g., results of the over night runs are reported during the morning meetings.
Besides initial error analysis, the framework allows the recovery of events that suffered from transient errors (e.g., communication and processing timeouts). The recovery procedure combines two methods:

- Reconstruction of derived event information that got lost in the online system due to packet transmission timeouts. Some of this information can be reconstructed offline from the event data (e.g., the results of the trigger selection algorithms running at LVL2).

- Reprocessing of events that caused timeouts in the online system. The events are reprocessed with the same HLT configuration, but with no time constraints.

This procedure enabled the recovery about 85% of the debug stream data in the commissioning phase. On recovery, the events were assigned to the appropriate physics and calibration streams. Since events stored in a debug stream do not have overlap with any other stream, they can be re-injected automatically in the reconstruction and analysis path. However, at the moment, they are stored in separate data sets and have to be included explicitly if needed for a physics analysis.

Both recoverable and the non-recoverable events are split into different error types. An initial error analysis is done with the information available in the event header. If the event information is available, further error analysis is done automatically on the available HLT processing results. Plots are made to summarize error statistics and other information e.g., correlations with event variables and time. Files with a few events are created for each error type to enable further debugging. Based on the initial and post-processing error analysis, problems are assigned to the appropriate experts.

C. Monitoring

The assignment of events to the various streams is monitored online at all stages of the TDAQ system. During every run, each trigger level counts how many events are accepted into each stream. The event counters are distributed over many nodes running either LVL2 or EF algorithms and are continuously collected centrally and summed up. This allows for cross-check that no events are lost from any stream except for the ones rejected by the LVL2 or EF. Due to the asynchronous nature of the system, an exact comparison can currently only be done at the end of the run. At the moment this is not done automatically, but during earlier technical runs a few inconsistencies were found and fixed. During running, shifters monitor both the individual trigger rates and the rate of each stream to ensure that the trigger is performing as expected. The debug streams are of particular interest, since they should always have a very low rate. Besides rates, cross-correlations between streams are monitored in the online system to identify unexpectedly large (or small) overlaps.

The express stream is processed offline in quasi-real-time for data quality assessment. Prompt reconstruction of the express stream is performed with information from the same calibration database as used by the online system. In parallel with the express stream, a number of calibration streams are processed to provide updated calibration data on alignment and energy scale for example. Part of the express stream might be reprocessed, e.g., with updated calibration constants for validation purposes or due to problems with conditions or reconstruction software that need to be fixed. If no unforeseen problems occur during the express stream validation procedure, bulk reconstruction of physics data starts between 24 and 36 hours after the end of a run.

VI. Conclusions

ATLAS successfully implemented and commissioned event streaming functionality in its trigger and data acquisition system to facilitate fine grained separation of calibration and physics data. Reliability and flexibility are two of the key factors that determine its success. Reliability is achieved with a data driven and modular approach, where online components write and/or read streaming information (StreamTags) in the event header to drive an event through the data acquisition system to mass storage.

Flexibility is provided by a mapping of trigger chains to streams in the trigger configuration database, the same database that contains the menu configurations. Furthermore, the system is able to run in exclusive, inclusive, or hybrid mode. In exclusive mode an event assigned to multiple streams is stored in an overlap stream. Thus, each recorded event is unique. In inclusive mode, a copy of such an event is written to each of the streams.

Partial event building, which enables the collection of small events (containing a subset of the full event information) at high rate for calibration purposes, is fully supported. A mechanism called routing is introduced to optimize the event path through the data acquisition system for calibration or other events that do not need further online processing, i.e., by sending them directly to mass storage. Similarly, events that cause problems in the online system end up in a separate debug stream. A failure analysis and recovery framework framework runs on a batch farm offline and tries to recover these events in quasi real-time. Error analysis is done for both the recoverable and non-recoverable events.

The streaming functionality is continuously monitored online as well as offline. Cross-correlation plots are produced by the online system to show the overlap between different trigger streams and to make sure that no events are lost. A dedicated stream, called the express stream, is available for offline data quality assessment. If no major problems are revealed, green light is given for reconstruction of physics data. This happens typically between 24 and 36 hours after a run.

Streaming already proved to be very beneficial during the commissioning phase. A good example during initial data taking in single beam mode was the split-off of cosmic ray events. As a result it is not necessary to run over all cosmic ray events to study, e.g., trigger efficiencies in the forward regions. On the other hand, the cosmic ray stream is exploited for tracking, timing, and detector alignment. The entire data taking infrastructure (including event streaming) has been tested thoroughly with the collection of many millions of events. The combined collection of physics and calibration data avoids separate calibration runs as much as possible and thus increase the operational efficiency of the ATLAS data acquisition system.
REFERENCES