Abstract—The ATLAS High Level Trigger (HLT) system provides software-based event selection after the initial LVL1 hardware trigger. It is composed of two stages, the LVL2 trigger and the Event Filter (EF). The LVL2 trigger performs event selection with optimized algorithms using selected data guided by Region of Interest pointers provided by the LVL1 trigger. Those events selected by LVL2, are built into complete events, which are passed to the EF for a further stage of event selection and classification using off-line algorithms. Events surviving the EF selection are passed for off-line storage. The two stages of HLT are implemented on processor farms. The concept of distributing the selection process between LVL2 and EF is a key element in the architecture, which allows it to be flexible to changes (luminosity, detector knowledge, background conditions etc.) Although there are some differences in the requirements between these sub-systems there are many commonalities. An overview of the dataflow (event selection) and supervision (control, configuration, monitoring) activities in the HLT is given, highlighting where commonalities between the two sub-systems can be exploited and indicating where requirements dictate that implementations differ. An HLT prototype system has been built at CERN. Functional testing is being carried out in order to validate the HLT architecture.

I. INTRODUCTION

ATLAS is a general-purpose high-energy physics experiment for recording proton-proton collisions, currently under construction at the Large Hadron Collider (LHC) at CERN. ATLAS has been designed to study the largest possible range of physics at the LHC including searches for as yet unobserved phenomena such as the Higgs boson and super-symmetry.

The ATLAS Trigger and Data Acquisition (TDAQ) system will have to deal with extremely high data rates, due both to the high bunch crossing frequency at the LHC (40 MHz) and the large amount of data produced by the ATLAS detector itself (1-2 Mbytes per event). The task of the TDAQ system is to select from this unprecedented amount of data, the most interesting events and save them for later analysis at a rate of about 200 per second. ATLAS relies on a three-level trigger system to perform the selection: a very fast, coarse granularity, hardware-based LVL1 trigger which reduces the event rate to 75 kHz followed by two software-based triggers, the LVL2 trigger and the Event Filter (EF) which perform increasingly fine-grained selection of events at lower rates. The LVL2 trigger, working on a sub-set of full granularity detector data reduces the rate to about 3 kHz and finally the EF using full event data reduces the rate to about 200 Hz after which the selected events are written to mass storage. The LVL2 and EF comprise the ATLAS High level trigger (HLT) system.

II. OVERVIEW OF THE HLT

The role of the HLT is to reduce the LVL1 trigger rate to a rate compatible with writing the events to mass storage. Note that this is no longer a technical constraint. It is constrained by the cost of storage devices and on the amount of computing and manpower available to analyze the data off-line.

The HLT is a software trigger implemented as software applications running on large processor farms consisting of commodity components connected via high-speed Ethernet networks. The starting point for the HLT is the LVL1 trigger result. The LVL1 trigger is a coarse-grained hardware trigger using information from the calorimeters and muon detectors only. In addition to providing the trigger result the LVL1 trigger identifies the geographical locations in the detector of candidate muons, electrons, photons, jets and hadrons. These are known as Regions of Interest (RoIs). The RoIs are classified into two types. Primary RoIs that...
caused the LVL1 accept, and secondary RoIs, which were not used in the LVL1 accept. Both types of RoI are used to seed the LVL2 selection. The concept of seeded reconstruction is fundamental to the LVL2 trigger.

A diagram summarizing the flow of data and messages within the HLT system is shown in Fig. 1.

The flow of data within the HLT is as follows. Once an event has been accepted by the LVL1 trigger, data for that event are sent from the detector front-end electronics to the Read-Out Systems (ROS) which contain ~1600 Read-Out Buffers. In parallel, information from the LVL1 concerning the accepted event is sent to the LVL2 trigger via the RoI builder, which assembles the information on different RoIs into a single combined record. The RoI information is received by the LVL2 Event Supervisor. The role of the LVL2 Event Supervisor is to allocate the event to one of many LVL2 Processing Units (L2PUs).

Fig. 1. Diagram summarizing the exchange of messages and event data between the various components of the ATLAS High Level Trigger system, the names in brackets are the names of the applications where these functions are implemented.

Highly optimized algorithms run on the L2PUs and perform the selection, which is broken down into a series of steps known as sequential processing. Typically the first step involves confirming the LVL1 trigger RoIs. Full-granularity event data from the calorimeters and muon detectors are requested from the relevant ROSs and analyzed. In subsequent steps data are requested from other detectors, e.g. tracking detectors, corresponding to the RoI and analyzed. At each step, if the result is not consistent with a possible physics candidate, the event is rejected and no further processing occurs. Only the events surviving all steps are accepted by the LVL2 trigger. For these, the LVL2 result is sent to the LVL2 Event Supervisor.

The RoI mechanism and sequential processing allow full-granularity event data, including information from the inner tracking chambers not available to the LVL1 trigger, to be used in the LVL2 selection process. Only a few percent of the entire event data needs to be transferred to the LVL2 trigger, thereby considerably reducing the network bandwidth required (to ~2.5 Gbyte/s).

The LVL2 Event Supervisor passes the IDs of events selected by the LVL2 trigger to the Data Flow Manager (DFM) in the switch-based Event Builder (EB). In parallel the L2PUs send a more detailed LVL2 result to a so-called “pseudo-ROS” (pROS) to be included in the event to be built. Event fragments in the ROSs and pROS are collected under guidance from the DFM application and assembled into complete event records by Sub-Farm Input (SFI) applications, one per output port of the event building switch. The bandwidth required for the event building is ~5 Gbyte/s.

The complete events are passed to the EF, the second stage of the HLT where the final stage of the on-line event selection is performed. The events are sent from the SFIs to Event Filter dataflow tasks (EFDs) running in the Event Handler. There would normally be one EFD application
running per processing node. The role of the EFD is to distribute the events to processing tasks (PTs) located on the same processing node which run the filtering algorithms, collect the results and send them to permanent storage, via a sub-farm output (SFO) application in the Event Filter IO. The processing tasks use off-line algorithms, with limited adaptation, to perform the selection process using the entire event data. It is foreseen that the LVL2 result passed via the pROS should be used in the EF to seed the selection in order to reduce the time taken to obtain a result.

There are many similarities between the two stages of the HLT. The boundary between LVL2 and EF has been left configured to take into account different running environments. The commonalities and differences between the two systems comprising the HLT (i.e. LVL2 and the EF) are described in more detail in the following section.

III. FUNCTIONAL COMPONENTS OF THE HLT

The HLT consists of three main functional parts:
1. the event selection software (algorithms) and the interfaces required for obtaining event data,
2. the dataflow code, i.e. the code responsible for transferring the event data and trigger decisions to and from the selection algorithms,
3. the supervision system, the software responsible for preparing the HLT for data-taking activities.

The aim of this section is to discuss each of these aspects in turn, describing where similarities exist between LVL2 and the EF and how these may be exploited and why in other cases, due to differing requirements, implementations differ.

A. Event Selection Software

The tasks of the Event Selection Software (ESS) are “event selection” and “event classification”. HLT algorithms construct objects representing physics candidates such as electrons, jets and muons. An event is selected if the reconstructed object is consistent with at least one physics signature. In both the LVL2 and the EF, events are rejected if they do not pass any of the selection criteria.

The ESS in the LVL2 trigger is implemented by algorithms specially written by the Physics and Event Selection Architecture (PESA) group [2] running in the LVL2 Processing Units and in the EF by off-line algorithms possibly with some modification running in the Processing tasks. This implementation is imposed by the different environments in which the algorithms run. In the LVL2 trigger the average trigger processing time must be short (~10 ms) because of the high rate to be handled. Therefore, algorithms must be highly optimized. Secondly, algorithms running in the LVL2 environment must satisfy stringent thread-safety rules. In the L2PU the selection algorithms run inside one of several “Workerthreads”, where each Workerthread is responsible for processing one event. This multi-threaded approach has been chosen to avoid stalling the CPU when waiting for requested RoI data to arrive from the ROSs. It also allows efficient use of multi-CPU processors. In the EF the requirement on the trigger decision time is less severe. Secondly the algorithms run in single-threaded processing tasks. Therefore, algorithms need not be thread-safe. However, studies are underway [2] in the EF to assess the suitability of all the off-line algorithms for use in the on-line environment and changes implemented where necessary. For example, although the trigger decision time is less critical in the EF, it may still not be entirely practical to run the algorithm with all the precision that it would have in an off-line environment because it takes too long. It may be “de-tuned” in order to achieve a decision more quickly.

Due to the different environments in which the ESS runs, the interface used by the ESS to collect event data is different in LVL2 and EF as well as in the off-line situation. In the case of LVL2, a sub-set of detector data, in on-line format, corresponding to LVL1 RoIs is delivered to algorithms over the network from the ROSs. In the EF, full-event data, in on-line format, is delivered to the algorithms running in the processing tasks from the EFD via a memory-mapped file. In the off-line environment, the event data will be delivered to the analysis algorithms from databases, in off-line format. However, a key design decision has been that the data should be delivered to the algorithms in exactly the same way in all environments. The algorithms themselves are not aware of where they are running. This has been achieved by using the off-line Gaudi and ATHENA [3] frameworks to implement the LVL2 and EF interfaces. The interfaces also provide data conversion services to transform the incoming event data from the on-line format into the off-line format expected by the algorithms.

Creating a common framework in which to run the ESS in the HLT greatly simplifies migration of algorithms between the various environments. For instance, although LVL2 algorithms will not be used in the off-line context to analyze data they can be developed and tested in the more user-friendly off-line environment. Furthermore, it becomes trivial to move LVL2 algorithms to the EF. For instance, LVL2 algorithms might be moved to the EF to cross-check the efficiency of the algorithms, or to take advantage of more accurate calibration information. Due to the short decision time of the LVL2 trigger, L2PUs will only be able to access calibration databases once per data-taking run. In the EF it will be possible to access calibration databases on an event-by-event basis. It should also be remembered that flexibility in the configuration of the HLT will be vital for tuning the trigger to select events corresponding to novel, unpredicted physics.
B. Dataflow

The dataflow system is responsible for moving event data and trigger decisions to and from the algorithms described in the previous section. Implementations differ between LVL2 and the EF. In the LVL2 trigger, dataflow activities are performed by the LVL2 Event Supervisor and the LVL2 Processing Unit. These are two of the applications implemented using the framework developed by the DataCollection group [4]. DataCollection is a system of the TDAQ responsible for the movement of event data from the ROS to mass storage via the LVL2 trigger and the EF. The LVL2 Event Supervisor sends LVL1 Results to the L2PUs. In order to achieve load balancing on the L2PUs, it is foreseen that LVL1 results are assigned to L2PUs on a round-robin or a least-queued basis. The LVL1 result contains only the LVL1 event ID, trigger type and a list of primary and secondary RoIs. It does not contain any event data. It is the responsibility of the L2PU to access the event data corresponding to the RoIs. Due to the short decision time required in LVL2, the LVL2 system is situated before event building. Therefore, the L2PUs need to access the ROSs directly over the network to obtain the event fragments corresponding to the RoIs. It is one of the tasks of the interface to the LVL2 algorithms to collect these data and make them available to the LVL2 algorithms.

The EF is located after event building, where complete events are already assembled and available, allowing dataflow and selection tasks to be clearly separated. The dataflow is implemented by the Event Filter Dataflow (EFD) process [5], which receives complete events from the SFIIs at the output of the Event Builder and delivers them to the PTs, where event selection takes place. It subsequently collects the analyzed events and delivers them to mass storage via the SFO. The EF is the first part of the TDAQ system in which complete events are available. It is therefore a good place to perform physics monitoring, calibration and alignment procedures. Therefore, in addition to making events available to the PTs running the selection algorithms, the EFD must sort events, using information in the event headers to different types of processing tasks according to the event type. For example, it should send calibration events to the appropriate detector calibration task.

The EFD functionality is divided into specific tasks that can be dynamically interconnected to form an EF dataflow network internal to the EFD process. The tasks can be purely internal (sorting to different data streams, internal monitoring of dataflow) or provide the interface to external components (SFI, SFO, PTs). The internal dataflow is based on reference passing, only the pointer to the event (stored in a memory-mapped file) flows between the different tasks. The events are never copied.

C. Supervision

The supervision system [6] is responsible for all aspects of software task management and control in the HLT. Mandates of the supervision system include: configuration of the HLT software processes, synchronizing the HLT processes with data-taking activities in the rest of the experiment, monitoring the status of HLT processes, e.g., checking that they are running and restarting crashed processes. Both the LVL2 trigger and the EF are implemented as hundreds of software processes running on large processor farms, split for reasons of practicality into a number of sub-farms. In view of this the supervision requirements for the two systems are very similar and an integrated HLT supervision system has been developed. It has been implemented using services provided by the Online Software (OnlineSW) [7]. The Online Software is a system of the TDAQ project. It encompasses the software to configure, control and monitor the TDAQ. It does not contain any elements that are detector specific and has been successfully adapted for use in the HLT trigger farms.

The OnlineSW configuration database is used to describe the HLT in terms of the software processes and hardware (processing nodes) of which it is comprised. The HLT supervision and control system uses information stored in the OnlineSW configuration database to determine which processes need to be started on which hardware and subsequently monitored and controlled. The smallest set of HLT elements, which can be configured and controlled independently from the rest of the TDAQ system is the sub-farm. This allows sub-farms to be dynamically included/excluded from partitions during data-taking. Synchronization with the rest of the TDAQ system is achieved using the OnlineSW run control system. Each sub-farm has a local run controller, which will interface to the Online Software run control via a farm controller. The controller collaborates with a sub-farm supervisor, which provides process management and monitoring facilities within the sub-farm. The controller and supervisor cooperate to maintain the sub-farm in the best achievable state by keeping each other informed about changes in supervision or run-control state and by taking appropriate actions, e.g., restarting crashed processes. Where possible, errors should be handled internally within the HLT processes. Only when they cannot be handled internally should errors be sent to the supervision and control system for further consideration.

Scalability tests carried out at CERN [8] using a prototype supervision system implemented with the OnlineSW showed that execution times taken for starting and stopping HLT trigger processes and for performing run control transitions to prepare for data-taking do not depend strongly on the number of controlled nodes. The times are shorter than 3 seconds for configurations of a size varying between 50 and 230 nodes and running up to approximately 1000 HLT software.
processes.

IV. HLT INTEGRATED TESTBED

An integrated HLT prototype system has been built at CERN. The system is a functional integration of previously existing LVL2 and EF prototypes. The integrated prototype consists of the following components:

1. a ROS preloaded with event data from a file. The data in the file correspond to ~1000 LVL1 trigger accepted events containing electron and jet RoIs,
2. a LVL2 system consisting of a LVL2 Event Supervisor and LVL2 Processing Unit. The L2PU runs the TCALO algorithm for e/gamma identification,
3. an EB consisting of a DFM and SFI,
4. an EF system consisting of one EFD and 2 PTs running the TCAL algorithm in the ATHENA environment,
5. A sub-farm output (SFO) to which the EFD sends the event once the processing tasks have completed the selection,
6. OnlineSW supervision infrastructure to provide synchronization and process control

The computing infrastructure from previous LVL2 and EF standalone tests has been reused. The ROS, LVL2 and EB (including the SFI and SFO applications) run on a system in building 32 at CERN, using PCs and a local switch. The EF components, consisting of the EFD and the ATHENA PTs run on a system located in building 513, using PCs and a local switch.

The main emphasis of the prototype is on functionality. Detailed performance measurements have been made in separate LVL2 and EF slice tests [10], [11] and the execution time of algorithms can be measured in off-line mode. However, some of the measurements made for the standalone systems will be repeated in the integrated testbed to verify its correct functioning.

V. STATUS OF TESTBED

The LVL2 and EF control and supervision infrastructures have been successfully integrated to create a common control structure. This supervision system has been used to start all the HLT processes and bring them to the “running” state. Events have been observed to flow through the full system. Tests are currently underway to repeat key measurements made on the standalone testbeds, in particular, the throughput in the EF will be measured as this depends critically on the correct functioning of the link between the LVL2 and EF parts of the system.

VI. CONCLUSIONS

A final design now exists for the HLT system in preparation for the ATLAS TDAQ Technical Design Report (TDR) [12]. LVL2 and EF prototypes based on the design have already been implemented and the required performance demonstrated in the critical area of data access and the principal aspects of system scalability. Integration of the LVL2 and EF prototypes into a coherent HLT prototype is currently underway and tests will be performed to test functionality and validate the design. Results will be included in the ATLAS TDAQ TDR.

VII. ACKNOWLEDGMENT

The authors would like to thank the many people in the ATLAS collaboration who made this work possible, particularly in the other parts of ATLAS Trigger/DAQ.

VIII. REFERENCES