

# Grouping of L1 Prescale Sets in the ATLAS TriggerTool

Author: Vince Croft

Supervisor: Tiago Perez Cavalcanti

Desy Summer School 2010

## Abstract

The ATLAS experiment has provided scientists and engineers some of the most challenging problems of the last decade in many areas from construction, electronics and computing. The ATLAS trigger system is implemented as a controlling part of the data acquisition system to filter the large quantity of events into a much smaller subset of data containing an enhanced population of low crosssection processes. The trigger tool serves as a interface between the highly complicated interdependent settings that control this data flow in the system and the user. This report will explain the use of the trigger tool in conjunction with the ATLAS trigger system and give an example of how the java programming language is utilised through the addition of an option to group several triggertool data sets.

## ATLAS Trigger system

The ATLAS experiment, shown in figure 1, is designed to receive data at a rate of around 40 MHz. The ATLAS data acquisition system however can only commit data to permanent storage at average optimal rate of a few hundred Hz. The trigger system is therefore employed to select 'interesting' data from the event. It endeavors to avoid discarding any potentially useful data without overloading the data aquisition system. A trigger chain is constructed as a set of logical criteria that discard uninteresting events. The ATLAS trigger system operates in three levels. The first level being hardware based and the other 2 are software based. Level one (L1) receives data at a rate of 40 MHz. It's Job is to reduce down the data flow to around 75 kHz. It does this by judging the multiplicities and energy thresholds of signals released from the calorimeter sections or possibly with signals from the muon spectrometer.

The most common criteria for this level 1 system is the amount of transverse energy required to set off the trigger. If for example a trigger has a threshold of 2GeV in a certain section, any events that have signals corresponding to energy deposits in the detector with more than 2GeV that data sets off the trigger. If an

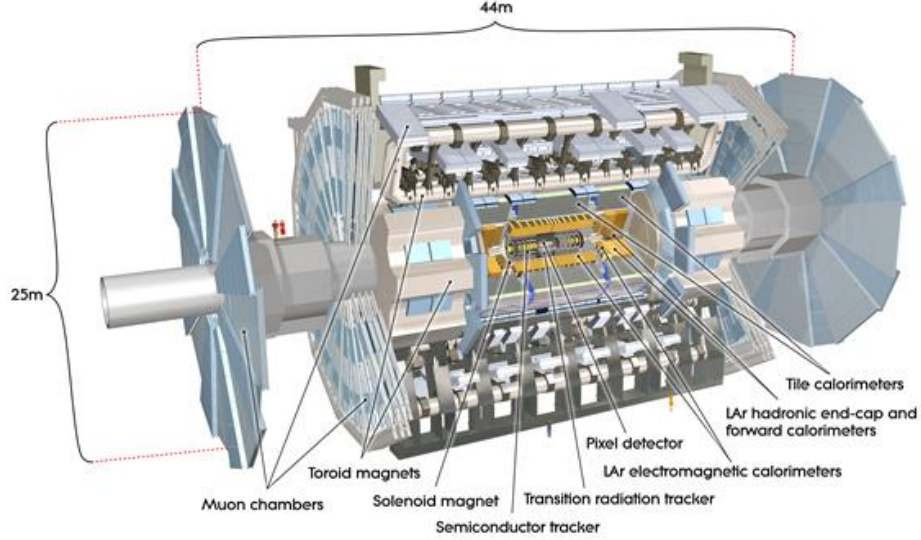


Figure 1: The ATLAS detector layout

event satisfies the criteria of level 1 the data produced in the area surrounding the triggering signal is passed on to the next level. Level two (L2) receives the data from level one in the form of Regions of Interest (ROIs). These regions of interest can be processed faster than the analyzing over the whole event and allows the level 2 trigger combines data from the calorimeters, tracking chamber and muon spectrometers to provide a high level of data rejection. The processing achieved by the second level algorithms allow data to be reduced down to around 1 kHz. The Level 2 and Event filter are both software based and are referred to together as the High Level Trigger (HLT). The Event filter (EF) works with the seeds in the ROIs given by L2 combined with the event buffers and use similar algorithms to those used in the offline reconstruction to measure as many parameters of the event as possible. In this way data from the whole event may be processed if necessary, quickly and efficiently rejecting all unwanted events. A pictorial overview of the ATLAS trigger system is shown in figure 2.

## Prescales and Prescale Sets

The data flow of each trigger is tuned by prescaling. The L1 triggers are processed by custom hardware modules with fixed timing. They are always active, whereas the High Level Trigger (HLT) is only called upon when a certain trigger chain is desired. To adjust the data content, soft physics triggers or triggers on events with very high cross-sections are scaled down. This avoids

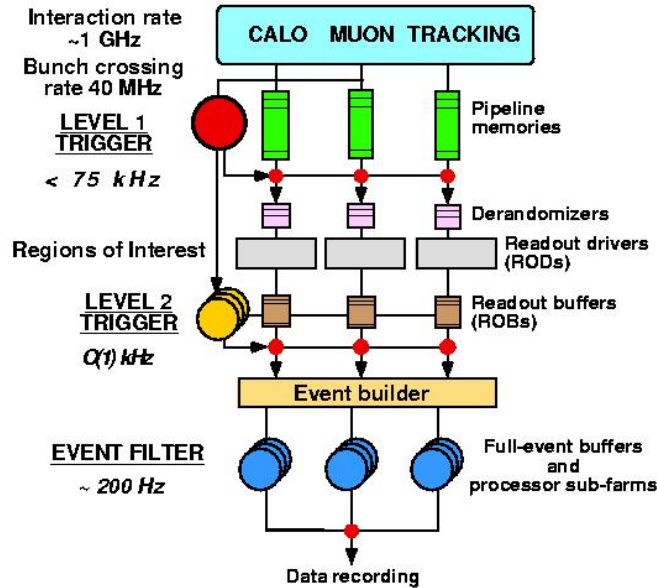


Figure 2: An overview of the ATLAS trigger system.

any high cross-section events from dominating the final sample. L1 prescales are prescaled after taking data due to being hardwired into the system. The trigger rates are calculated statistically and the data flow controlled dynamically. Prescaling guarantees that the ATLAS detector remains multipurpose with triggers searching for different physics processes given different priority to account for their different cross-sections. Prescales operate over the whole system with each individual trigger chain scaled to match the desired content of the final data sample. These sets are created, stored and controlled using the TriggerTool. A pictorial representation of the data flow is shown in figure 3.

## Trigger tool

The TriggerTool (TT) provides users the functionality to control, monitor and access the ATLAS data acquisition system running parameters. It implements benefits of the JAVA programming language in both object orientation and graphical functionality. It provides a dynamic working environment in which users with different permissions may interface graphically to the database that contains all of the data needed to run the atlas trigger system. The TriggerDB Stores and protects all data needed for the configuration of all three levels of the central part of the configuration system including the central trigger processor. TriggerDB is implemented making use of relational databases meaning

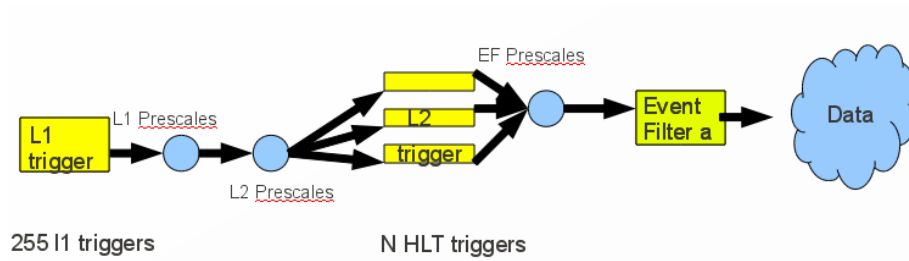


Figure 3: data flow in the ATLAS trigger system. Yellow boxes represent criteria to be fulfilled by the data, blue circles represent prescale points. Data is rejected at each node in the flow diagram.

the Trigger tool must have a very sophisticated back end to make proper use of the TriggerDB design.

## Usage of Java in HEP computing

Java is a platform independent programming language mostly licensed by Sun Micro-systems as a subsidiary group of Oracle under the GNU General Public License. Java is a general-purpose, concurrent, class-based, object-oriented language, designed to have as few implementation dependencies as possible. This allows JAVA to be highly effective on cross platform projects. Primarily JAVA's object model works with references and only passes entire objects when explicitly required. This makes the performance of JAVA's object orientation very effective whilst remaining robust. Swing Java is used for the TriggerTool, Swing is a graphical user interface library for the Java SE platform.

## Example: Implementation of a option to group L1 prescale Sets

### Problem

The TriggerTool contains a GUI panel for displaying the prescale sets for a particular trigger menu. There 255 L1 prescales used in the ATLAS experiment. These are used in various combinations according to data stored in the trigger DB, TriggerTool then displays these combinations of prescales as a prescale set. When a set is created the nomenclature can be confusing and similar sets maybe hard to associate. The idea is that the prescale sets should be grouped according to their type. e.g. Physics versions, PP/HI, calibration, testing etc. This would allow a person to more easily identify which prescales are being used for which groups.

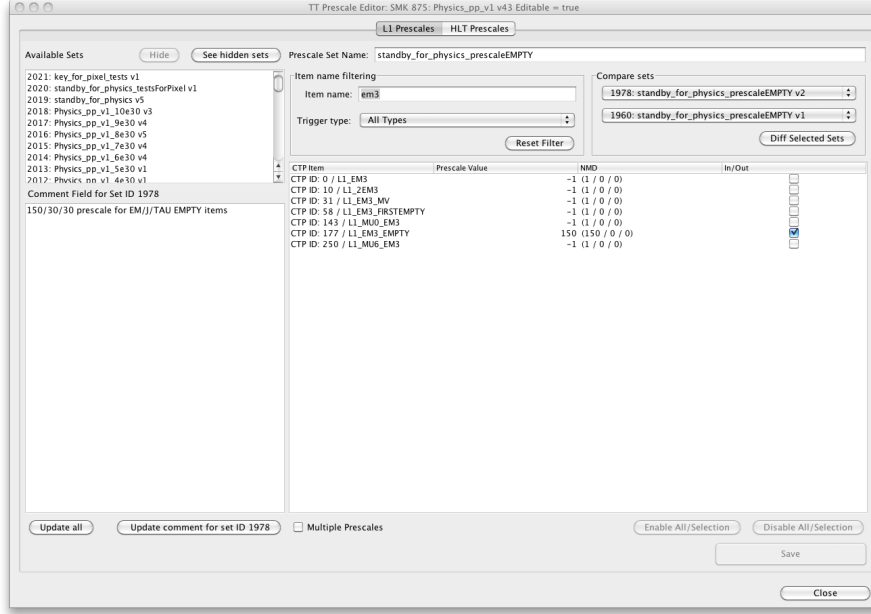


Figure 4: the prescale edit pannel of the TriggerTool. The available prescale sets are shown in the top left hand field.

## Restrictions

- One of the primary concerns with the operation of the TT currently is the operational speed. Any changes must not significantly increase the loading speed of the TT.
- Any changes to the currently operational code must be minimal to reduce the amount of code changed with the addition of additional function.
- The addition of a new function to the TT must not interfere with any of the currently operational functions.

## Solution

Any change to the TT must be done in 3 sections. Addition of a new table to the Database, a new method and an extension to the current TT back-end. A drop-down menu may then be placed within GUI that updates the list of displayed Prescale Sets (PSS) to show only those belonging to the desired group. The first section is to the DB (may also be performed last however is the first step necessary to begin testing the code) the DB needs the addition of a table to contain the group information (name, description, author etc) and a column to the table of prescale sets containing an id number designating which group it belongs to. A new class must be written into the JAVA code to access the

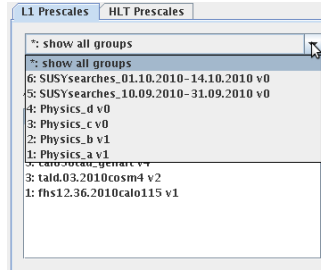


Figure 5: A JComboBox displaying different example group options to select.

new table in the database and implementation to set and retrieve the data it contains. The existing class concerning the prescale set data must also be edited so that it also accesses the information concerning the group. The GUI must finally make use of these changes with the addition of a Swing component called a JComboBox. This will be filled with the names of all of the groups found in the database. On selection of one of these groups the list of prescale sets will be filtered so that only the prescale sets belonging to a certain group are displayed. figure 5 shows how such a menu would look like.

## Database

For this example a MySQL test database was set up by creating a copy of the TrigDB to a local MySQL server. Considering the different options it decided that the new group primary key should be the destination of a foreign key from the table containing L1 prescale sets. The changes were made and tested using an open source SQL editor and frontend called Squirrel-SQL. The new table must follow the criteria laid down and required by the abstract table definition in the TT so that the DB Table must contain the Name, Id number, author, date modified, used Boolean and description. The group id number links the l1 Prescale set table to the group table. Figure 6 describes the link between the prescale set table and the prescale set group table.

## TT Back-end

The back end opens the database and provides access to view, edit and store the data it contains. Expert status is required to modify the data in the DB and users are given permission to view. This way filtering which users have permission to access and change the information it contains helps ensure the security of the system. A new class must be written into the JAVA Code to access the new table in the database and implementation to set and retrieve the data it contains with new methods to set and retrieve data. The existing class concerning the prescale set data must also be edited so that it also accesses the information concerning the group. A new method must be written to set pointers to the data in the Group table and the class that deals with the data

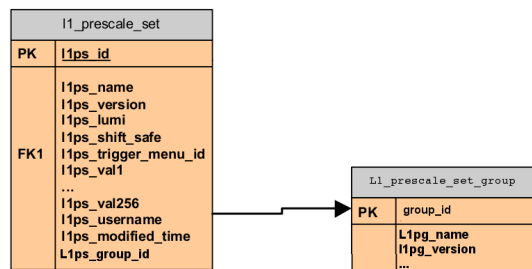


Figure 6: Relation of Prescale Set and Prescale Set Group tables

in the L1prescale set table must also be modified.

### Trigger Tool GUI

Since the TT GUI uses the JAVA swing libraries an interactive development environment called Netbeans was used to edit the swing components of the application. A JComboBox is filled on initialization with the data retrieved by the TT back-end. When a group is selected from this drop-down menu the list of all active (used equals true) PSS is no longer the basis for the list displayed in the GUI so the list of all active L1PSS is copied to a global array and it's contents filtered into a new list to be displayed only if their group id matches that of the selection pointer. Changes must also be made to implement this change of list so that all other functions now respond to the data displayed by the GUI. The filtered data must always be stored locally and retrieved from the displayed list only when necessary.

## Summary

The TriggerTool provides the ATLAS collaboration with the basis to implement the benefits of the JAVA programming language to effectively store, access, edit and govern the ATLAS trigger system. Charged with the task of improving this system I made changes to the trigger tool and the triggerDB that allows the TriggerTool to group the L1Prescale Sets in a way that is logical and allows easier access to similar data sets within a larger group of data. The changes to the database schema include the addition of a new table and new column, changes to the TT involve modifications that allow access to the database and changes to the user interface to give an effective display of the desired information. Over the course of the past few weeks an option has been included to group the prescale sets governing the data flow out of the L1 triggers into different groups for ease and clarity of use. figure 7 displays these changes

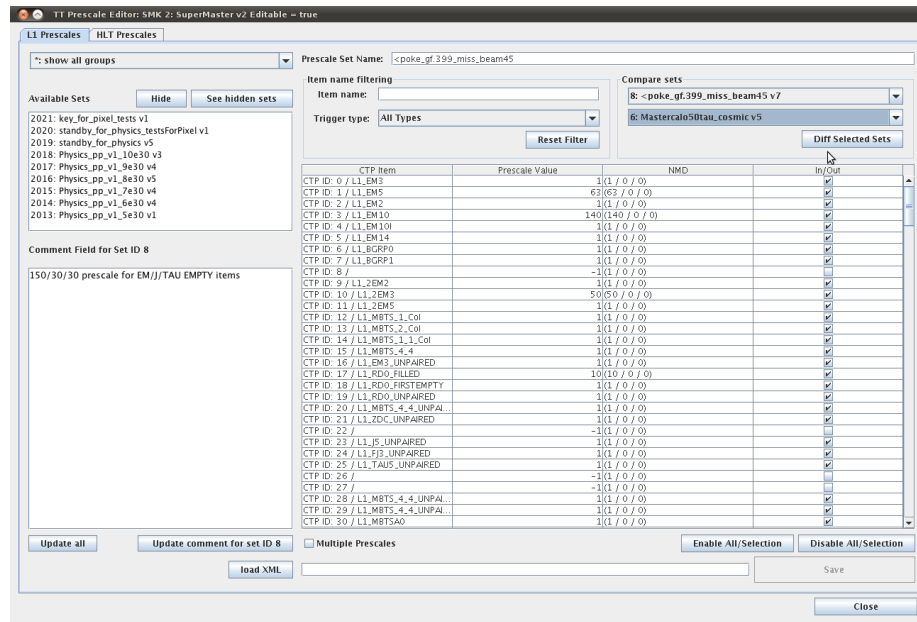


Figure 7: The Completed PrescaleEdit page of the TriggerTool with an option to group L1 prescale sets