# CONTROL AND MONITORING OF ON-LINE TRIGGER ALGORITHMS USING GAUCHO

E. van Herwijnen[1]
[1]*CERN, Geneva, Switzerland*

## ABSTRACT

In the LHCb experiment, the trigger decisions are computed by Gaudi (the LHCb software framework) algorithms running on an event filter farm of around 2000 PCs. The control and monitoring of these algorithms has to be integrated in the overall experiment control system (ECS). To enable and facilitate this integration Gaucho, the GAUdi Component Helping Online, was developed.

Gaucho consists of three parts: a C++ package integrated with Gaudi, the communications package DIM, and a set of PVSS panels and libraries. PVSS is a commercial SCADA system chosen as toolkit and framework for the LHCb controls system.

The C++ package implements monitor service interface (IMonitorSvc) following the Gaudi specifications, with methods to declare variables and histograms for monitoring. Algorithm writers use them to indicate which quantities should be monitored. Since the interface resides in the GaudiKernel the code does not need changing if the monitoring services are not present.

The Gaudi main job implements a state machine with states (stopped, configured, running and paused). The Gaudi job starts a DIM server. The Gaucho monitor service implements DIM services and commands to publish the monitored objects and receive commands, such as requests for state changes. PVSS panels provide the user interface. Using a DIM client written in the PVSS scripting language commands can be sent to the running job (configure, start, pause, stop) and to the algorithm to send data. 1D histograms are displayed using the PVSS trending tool and 2D histograms are displayed using ROOT.

Any histogram on the Gaudi Transient Store can also be dynamically published. Some configuration of algorithms is possible on the fly, and a future objective is to initialize algorithms using the data stored in the LHCb online configuration database.

We give some user experience of monitoring a large number of software processes.

## GAUCHO ARCHITECTURE

The key component of Gaucho [1,2] is DIM [3], the Distributed Information Manager. DIM allows for lightweight exchange of messages between running programs, in our case Gaudi [4], PVSS [5] and ROOT [6].

Figure 1 shows the interaction between the Gaucho components.

Using Gaucho, one can control and monitor jobs on an arbitrary number of computers.

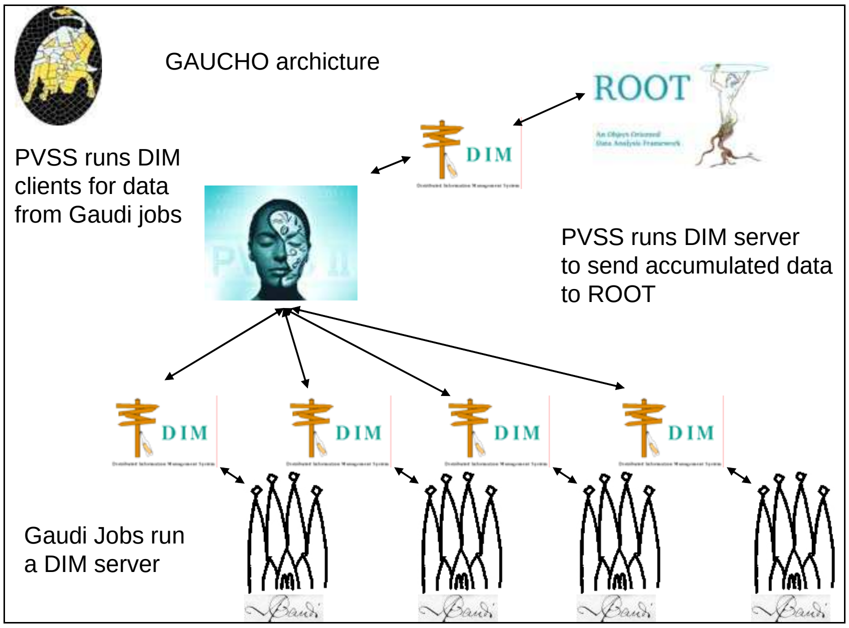

GAUCHO archicture
PVSS runs DIM clients for data from Gaudi jobs
ROOT
DIM
PVSS runs DIM server to send accumulated data to ROOT
Gaudi Jobs run a DIM server

Fig. 1. The Gaucho Architecture.

## THE GAUDI GAUCHO PACKAGE

### *Publishing data from algorithms using the MonitorSvc*

To publish counters, other variables and histograms, the Gaudi algorithm writer can use the Gaucho MonitorSvc class.

Some of the public member functions defined by IMonitorSvc are shown in Table 1, together with the DIM command used in their implementation. The service name (infoname) is constructed by concatenating the variable name with a unique job identifier called the UTGID (see below for a definition). By using the declareInfo function with an appropriate signature, an object can be published to a DIM client such as PVSS.

| Function | Data type | DIM Command |
|---|---|---|
| declareInfo(string &name, int &var, string &desc, IInterface *owner) | int | DimService(infoname.c_str(), (int&) var) |
| declareInfo(string &name, double &var, string &desc, IInterface *owner) | double | DimService(infoname.c_str(), (double&) var) |
| declareInfo(string &name, string &var, string &desc, IInterface *owner) | string | DimService(infoname.c_str(), (char*)var_cstr()) |

| | | |
|---|---|---|
| declareInfo(string &name, AIDA::IHistogram *var, string &desc, IInterface *owner) | AIDA::IHistogram | DimRpc(histoname.c_str(), "I:1","F") |

Table 1: IMonitorSvc public member functions

The IMonitorSvc interface is part of the GaudiKernel. Any code making calls to MonitorSvc functions will compile and run without modifications in the offline or online context. The information thus published can be made available to PVSS in a way that is completely transparent to the trigger applications.

### *The Dim server and Gaudi FSM*

To make the published data available to PVSS, the Gaudi Main program needs to be changed so that a DIM server is started and the Application Manager is placed under its control (the GaudiDimController). The name of the DIM server is the UTGID (see the next section). The Finite State Machine (FSM) implemented by the GaudiDimController is shown in Table 2.

| State | Commands accepted |
|---|---|
| not_connected (this state is only known to PVSS) | config |
| configured | start/stop |
| running | pause/stop |
| paused | stop |
| stopped | |

Table 2: GaudiDimController FSM

### *The UTGID*

Jobs on the event filter farm are identified by a unique string called the UTGID. We choose the convention: "Node"+"2 digit subfarm number with leading 0"+"2 digit node number with leading 0"+"algorithm type, L1 or HLT"+"the string job"+"1 digit job index, 1 or 2". For example Node0101HLTjob1. Our subfarm consists of 44 (or less) nodes with 2 CPUs, each being able to receive two jobs.

## THE GAUCHO PVSS COMPONENT

The Gaucho PVSS part is packaged as a JCOP Framework [7] compatible component. It consists of: a PVSSCTRL manager to be started from the PVSS console, a CTRL library, a set of datapoints and a set of panels.

### *The datapoints*

Each Gaudi job running under Gaucho's control has a datapoint attached to it to receive the state of the job; it also contains the hostname of the node where the job should run. Datapoints for each counter and histogram are published.

For each set of jobs running in a node, there is a "controller" datapoint that contains information such as the names of the published histograms and counters. For each histogram and counter there is a datapoint containing the summed/averaged information of the data for that node.

For all the jobs running in the subfarm, a similar datapoint structure exists. Overall there are about 30 datapoints per job, with 10 datapoint elements per datapoint. By using the datapoints for the subfarm, a panel can be created that gives the combined information of all the jobs running on the farm.

## *The GAUCHO PVSSCTRL manager*

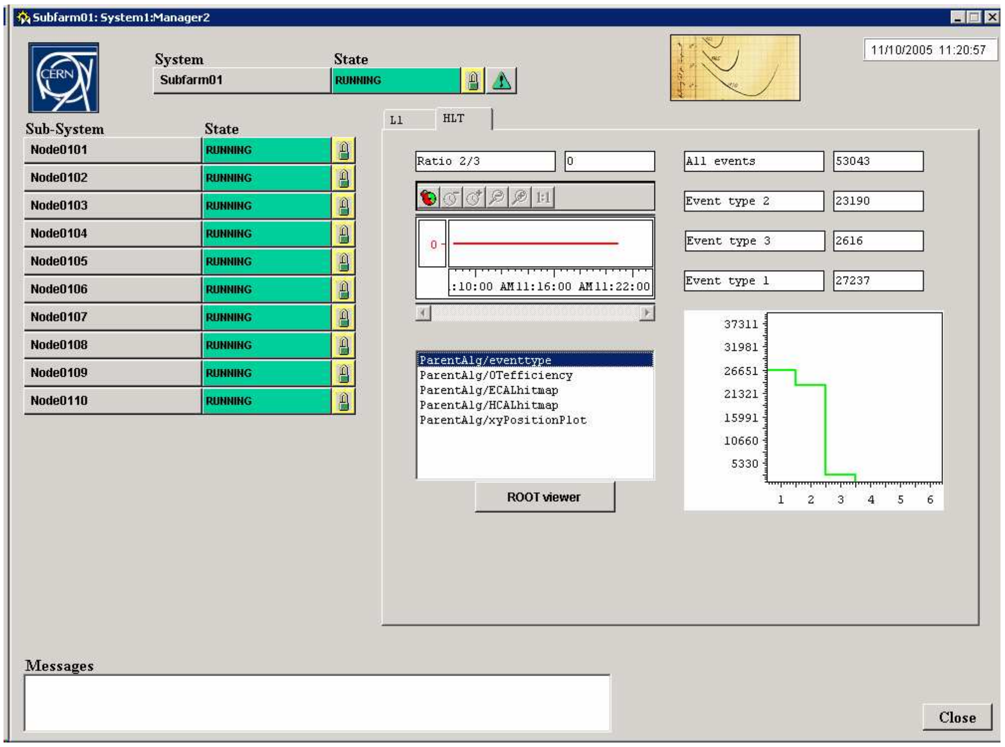
All DIM services are subscribed to by this manager. It has configuration parameters to indicate the number of nodes, published histograms and counters and their refresh rate. By default counters and histograms are updated once every 20 seconds. The subscription to the DIM services can be done independently of whether they exist or not, so they are setup once and for all. When all services have been subscribed to, there are about 100 DIM services per job.

## *The CTRL library*

The FarmCtrl library contains functions for submitting jobs, and for combining counters and histograms. Jobs are submitted from PVSS using the TaskManager [8] in the event filter farm nodes. The submitted command points to a fixed path containing the executable and its runtime options.
Some cleanup is done when jobs are stopped – however, combining information continues until all jobs are stopped.

## *The panels*

The Gaucho panels are used for starting/stopping jobs and for monitoring the information that they publish.
A three level panel hierarchy is available, starting from the subfarm (see Figure 1), down to nodes, terminating with panels displaying information from individual jobs.


Subfarm01: System1:Manager2

11/10/2005 11:20:57

| System | State |
|---|---|
| Subfarm01 | RUNNING |

| Sub-System | State |
|---|---|
| Node0101 | RUNNING |
| Node0102 | RUNNING |
| Node0103 | RUNNING |
| Node0104 | RUNNING |
| Node0105 | RUNNING |
| Node0106 | RUNNING |
| Node0107 | RUNNING |
| Node0108 | RUNNING |
| Node0109 | RUNNING |
| Node0110 | RUNNING |

| | |
|---|---|
| Ratio 2/3 | 0 |
| All events | 53043 |
| Event type 2 | 23190 |
| Event type 3 | 2616 |
| Event type 1 | 27237 |

```
ParentAlg/eventtype
ParentAlg/OTefficiency
ParentAlg/ECALhitmap
ParentAlg/HCALhitmap
ParentAlg/xyPositionPlot
```

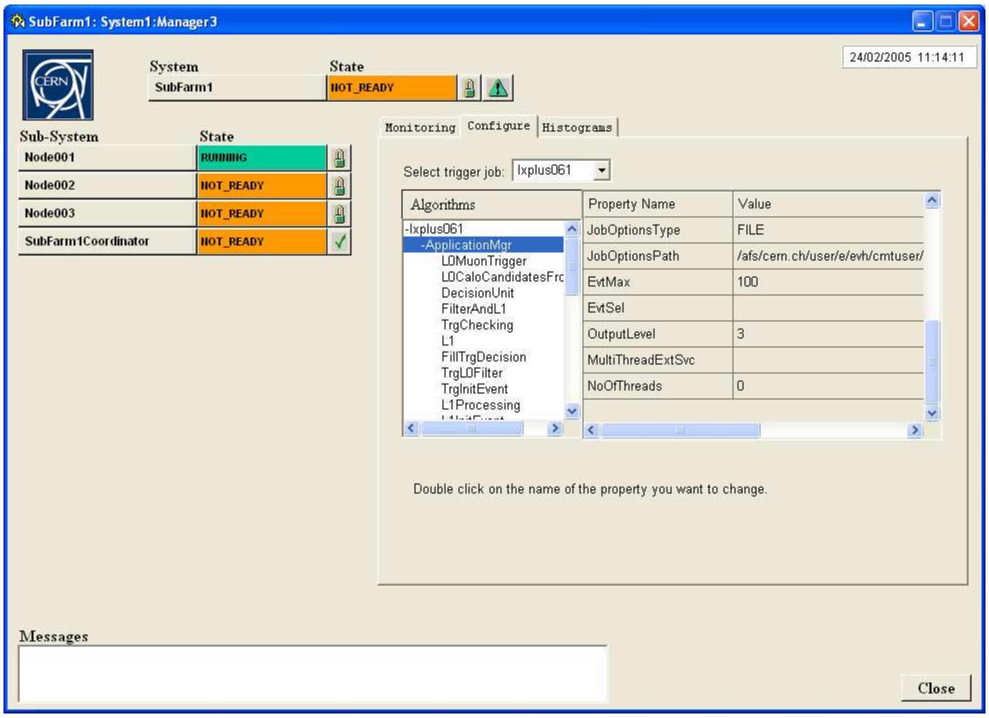
Figure 1: Histograms with results added from several jobs running simultaneously

The fields on the panel are connected to datapoints that contain the summed/average information per counter/histogram for the nodes under the control of the panel.
One-dimensional histograms are displayed using the PVSS trend tool while more-dimensional histograms are displayed in a separate ROOT viewer..

The messages sent to STDOUT by the Gaudi job are also published via DIM and are displayed in the "Messages" window.
The text labels before the counters are read from the "desc" argument in the declareInfo function and the names of the histograms are read from the "coordinator" datapoint.

## *Configuring algorithms on the fly*

By clicking the "Configure" tab on the panel for a single job one can see the algorithm tree of that job (see Figure 2).



| System | State |
|---|---|
| SubFarm1 | NOT_READY |

| Sub-System | State |
|---|---|
| Node001 | RUNNING |
| Node002 | NOT_READY |
| Node003 | NOT_READY |
| SubFarm1Coordinator | NOT_READY |

Select trigger job: lxplus061

Algorithms

```
-lxplus061
  -ApplicationMgr
      L0MuonTrigger
      L0CaloCandidatesFrc
      DecisionUnit
      FilterAndL1
      TrgChecking
      L1
      FillTrgDecision
      TrgL0Filter
      TrgInitEvent
      L1Processing
```

| Property Name | Value |
|---|---|
| JobOptionsType | FILE |
| JobOptionsPath | /afs/cern.ch/user/e/evh/cmtuser/ |
| EvtMax | 100 |
| EvtSel | |
| OutputLevel | 3 |
| MultiThreadExtSvc | |
| NoOfThreads | 0 |

Double click on the name of the property you want to change.

Figure 2: The algorithm tree of a running Gaudi job

By double clicking on a parameter, it can be changed.

## *Inspecting the Transient Event Store*

Similarly, by clicking on the "Histograms" tab one can see all histograms published on the transient event store. In case a histogram was not published by the author of the algorithm, it can be published dynamically by clicking on it. However, currently its contents will not be combined for all nodes in the farm.

## *The ROOT viewer*

The Gaucho PVSS component acts as a DIM server for the combined histogram information. This information can be accessed by any DIM client, such as for example the ROOT viewer. This is necessary for viewing 2D histograms as PVSS does not yet have a tool for viewing 2D histograms (scatter plots). The ROOT viewer is launched by selecting a histogram from the list, then clicking on the "ROOT viewer" button.
It is intended to save all combined information regularly (on request or at the end of a run) in a ROOT file, to allow further analysis to be done.
It is hoped that in a future version of PVSS which will use the QT windowing system, the ROOT viewer can be seamlessly integrated with PVSS.

## USER EXPERIENCE AND CONCLUSIONS

A first experience during the LHCb Real Time Trigger Challenge (RTTC) [9] was done with L1 and HLT applications running on a single node. In the case of the L1 application, the online trigger decision could be compared to the offline computed result.

Gaucho has been tried in a configuration with 20 jobs running over 10 lxplus nodes. In this test a simple Gaudi example job was used, not a real trigger algorithm reading data. The initial results look encouraging although some performance issues still need to be resolved.

We will now integrate Gaucho on the event filter farm to perform tests with realistic trigger algorithms, and to integrate the Gaucho FSM with the overall control system.

## ACKNOWLEDGMENTS

I would like to acknowledge the help of Helder Lopes and Pere Mato (for the implementation of the MonitorSvc), Clara Gaspar (for help with DIM, the FSM and PVSS), Beat Jost (for the implementation of the OnlineMessageSvc), and Sascha Schmeling (for help with PVSS and the JCOP framework).

## REFERENCES

[1] P. Vannerem, “Distributed Control and Monitoring of High-Level Trigger Processes on the LHCb on-line Farm ”, ICALEPCS’2003, Gyeongiu, Korea, October 2003.
[2] E. van Herwijnen, “Gaucho Installation and User’s Guide”, LHCb Note 2004-096.
[3] C. Gaspar, “DIM, a Portable, Light Weight Package for Information Publishing, Data Transfer and Inter-process Communication”, Proc. of CHEP, Padova, Italy, February 2000.
[4] G. Barrand et al., “GAUDI - A software architecture and framework for building LHCb data processing applications”, Proc. of CHEP 2000, Padova, Italy, February 2000.
[5] ETM, “PVSS II, Version 3.0”, Eisenstadt, Austria, 2004.
[6] F. Rademakers and R. Brun, “ROOT”, version 4.03, CERN, Geneva, 2004.
[7] IT-COBE group, “JCOP framework, version 2.3.3”, CERN, Geneva, 2005.
[8] F. Bonifazi et. al., “The Task Manager for the LHCb On-Line Farm”, LHCb Note 2004-099.
[9] U. Marconi et. al., “The LHCb Real Time Trigger Challenge”, LHCb Note to be published, 2005.