Software Framework Developed for the Slice Test of the ATLAS Endcap Muon Trigger System

S.Komatsu*, C.Fukunaga, Y.Ishida, K.Tanaka,
Tokyo Metropolitan University, 1-1 Minami-Osawa, Hachioji, 192-0397, Japan,
*Corresponding author: skomatsu@comp.metro-u.ac.jp,

K.Hasuko, H.Kano, Y.Matsumoto, Y.Nakamura, H.Sakamoto,
ICEPP, University of Tokyo, 7-3-1 Hongo, Tokyo, 113-0033, Japan,

M.Ikeno, K.Nakayoshi, O.Sasaki, Y.Yasu,
KEK, 1-1 Oho, Tsukuba, 305-0081, Japan

Y.Hasegawa, M.Totsuka
Shinshu University, 3-1-1 Asahi, Matsumoto, 390-8621, Japan

K.Mizouchi, S.Tsuji,
Kyoto University, Kitashirakawa-Oiwake, Kyoto, 606-8502, Japan

T.Maeno, R.Ichimiya, H.Kurashige,
Kobe University, 1-1 Rokkodai, Kobe, 657-0022, Japan

Abstract

A sliced system test of the ATLAS end cap muon level 1 trigger system has been done in 2001 and 2002 separately. We have developed an own software framework for property and run controls for the slice test in 2001. The system is described in C++ throughout. The multi-PC control system is accomplished using the CORBA system. We have then restructured the software system on top of the ATLAS online software framework, and used this one for the slice test in 2002. In this report we discuss two systems in detail with emphasizing the module property configuration and run control.

I. INTRODUCTION

Since the autumn of 2001, we have started the full slice test (SLT) of the ATLAS level 1 (LVL1) end-cap muon trigger system. Thin Gap Chamber (TGC) is employed for muon detection in the end-cap region to supply hit information to the level 1 trigger system. TGC can measure muon hits in two dimension (r with wire and ϕ with strip information). Detailed discussions of the TGC electronics and LVL1 muon end-cap system will be found in [1], [2].

We charge the TGC electronics system with two tasks, the one is to generate the level 1 end-cap muon trigger signals after identifying muon tracks coming from proton-proton interaction in the ATLAS detector, and the other one is to readout muon hit coordinate to utilize them for the second coordinate information in the ATLAS overall muon detection system. Present SLT system has been developed with full-chain minimum units but sufficient number of input channels to produce trigger signals of the original TGC electronics system. Prototype ASICs or VME modules has been used in the system. SLT is intended to verify the design of the electronics system, namely its validity of operation and performance (trigger latency and readout throughput) [3].

We have so far made two SLTs in 2001 and 2002 with two different software frameworks. The framework used in SLT2001 has been developed as a stand-alone system, and is relatively compact and simple because it was dedicated only for the primary commissioning of all the hardware elements used in the SLT system. In the SLT2002 we have restructured the software to comply with the ATLAS online software framework. Although we have developed the software frameworks in two stages, by designing the software as well as database structures with a strict object-oriented method from beginning, we have not made a hard work for the rewriting of the framework consequently.

In section III, we discuss detailed characteristics of two software frameworks after introducing the hardware setup of SLT in section II. We give some discussions and summary about the software development in section IV together with an outlook for future extension of the software framework.

II. OVERVIEW OF THE SLICE TEST SETUP

Detailed discussions of the hardware setup and performance results will be found in ref. 3, which is presented in this workshop. In Table 1, the components (modules) used for the slice test are listed along with ASICs mounted on modules. In Fig.1, we show the connection diagram of the components.

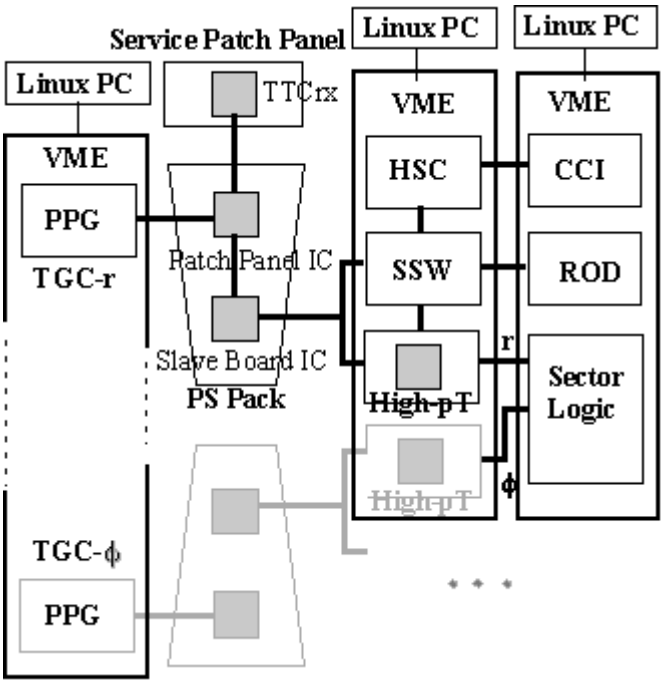

Figure 1: Connection Diagram of modules used for the slice test

III. SOFTWARE FRAMEWORK

For SLT, the software must have at least two facilities. The one is interactive control of the modules used in the system for various run configurations, and the other one is run-control for the test setup. In the SLT2001, we have developed the system from scratch while we have built the system on top of the ATLAS online software framework in the SLT2002.

A. Software used for the slice test 2001

The two most typical features of the framework are a CORBA based distributed control system for the VME modules, and module configuration control. We have focused to develop most critical parts for the test with already established software technology and with minimum man-power and developing time.

1) CORBA based VME access

As described in the previous section and shown in Fig. 1, the slice test system consists of multiple crates, which are controlled by different PCs running the Linux system. A driver used for PCI-VME adapter is vmehb-2.2.7, which is developed by NIKHEF[4]. Each PC in the system has the PCI-VME adapter. We have introduced CORBA [5] to control modules installed in a VME bus crate connected with another PC. In order to make API consistent with one for controlling modules with VME bus connected with own PC, we have achieved a seamless module control system. With this CORBA used system, anyone of PCs in the system can be a main controller for the slice test.

2) Module Control, Database system and GUI for the configuration

In order to configure the modules interactively within a short time, an aid of GUI is indispensable. In order to introduce GUI, we divided the software framework for the module control into three parts as the database (DB), GUI and control application, which actually accesses the hardware module via VME bus. An identical class structure (Board Object and Chip Object structure chain) has been introduced for the database, GUI, and control application as shown in Fig.2. An actual module used is inherited from the BoardObject, and a chip on the board is inherited from the ChipObject. The original database is described in XML and prepared for every module. In the configuration process, an XML parser (Xerces c++ XML parser) read DB and a board object, chip objects and property objects for all the chips on the board are made instantiated, and GUI window is opened. In this framework GUI is made constructed using unix/X11 version of Qt [6]. An example of a GUI window (for High pT board) is shown in Fig.3. Once an operator sets the property parameters, and clicks a "Configure" button in the window, the corresponding module control application receives the parameter values and access to the module registers through the VME bus.

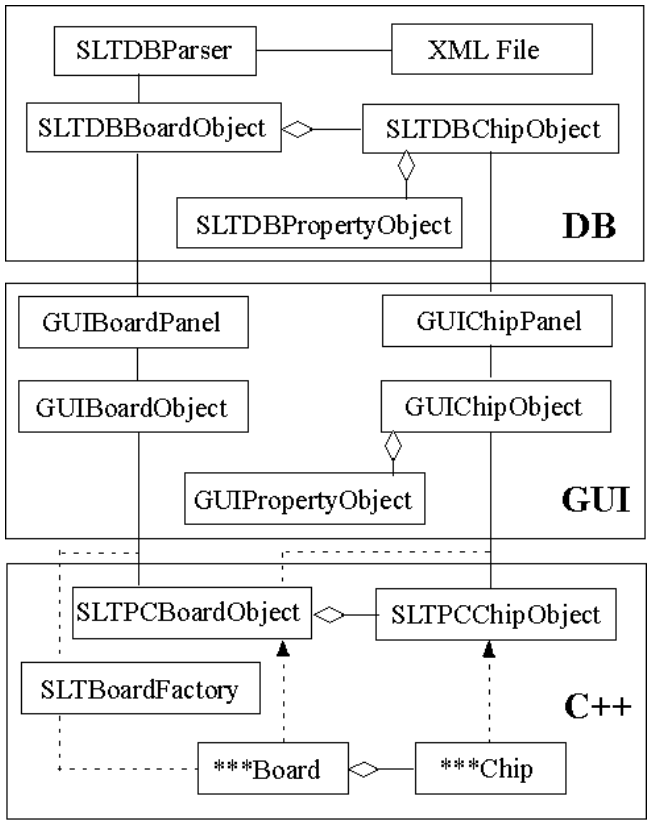

Figure 2: Class Diagram of SLT2001 Module control system

As far as the level1 trigger system test is concerned, no module used in the SLT is actively data produced except pattern generators (PPGs) that simulate the trigger input pattern. Once the configuration has been done for all the modules, no module will modify dynamically its configuration during the system test.

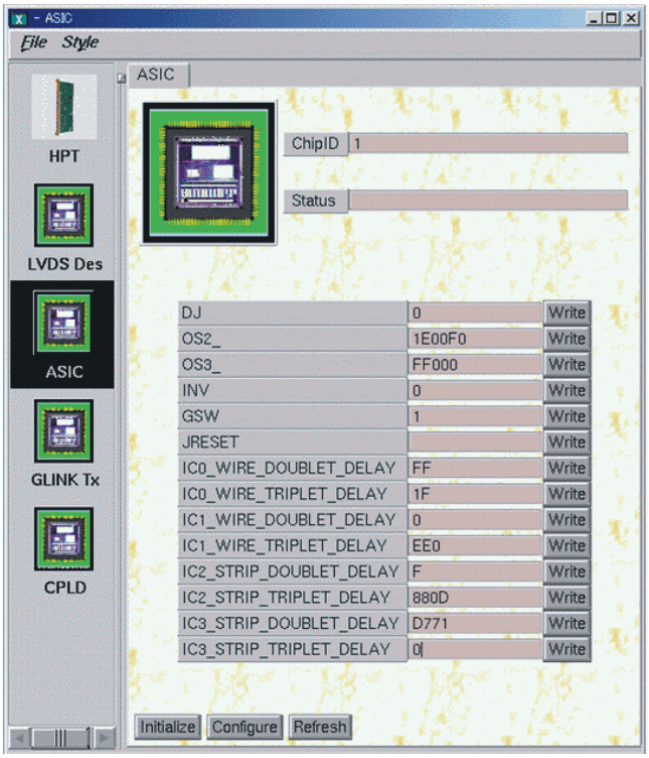

Figure 3: An example of module control GUI used in SLT2001

3) Test Management (Run Control)

As validation tests (LVL1 generation test) with input of a few tens of thousand of trigger patterns must be smoothly and quickly performed, to have an intelligent test management system (namely this is a conjunction of a run control, input data provider and output data collector/verifier systems) is very important.

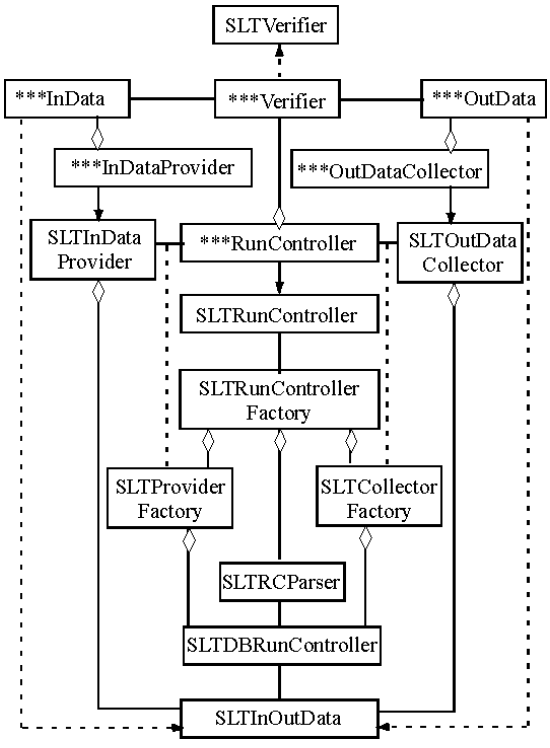

Figure 4: Class Diagram of SLT2001 Test Management System

The class diagram of the test management system is shown in Fig. 4. The diagram mainly consists of InDataProvider, which specifies input chamber hit pattern information, and OutDataCollector, which specifies the output data storage, and RunController, which manages input/output data access, control command handling (Run, Pause, Stop). Verifier class is used to compare output data with expected values calculated by the trigger simulation program with the same input data. A database dedicated for the run control, which is written in XML is prepared for the test management. It contains the hardware information of input, output as well as verifier modules, their location (server). A GUI window is also used for the database file and input/output data file specification as well as run control command submission.

B. Software used for the slice test 2002

For the second slice test performed in 2002, we have moved the software framework from our own one to the ATLAS online software. Although we had to develop many of new software ingredients for adapting for the ATLAS online software, the developments of the configuration database (confDB) and new run control scheme have been the two that we have concentrated to work. For our primary purpose of the slice test, histograming and event display software packages are actually not foreseen to be used so extensively in the data acquisition whereas these packages will be required in very near future for more elaborate system tests.

1) Configuration Database

In the SLT2001, we have used two kinds of database; one is for the module configurations, and the other one is for input/output specifications for the test management. In the new scheme of the ATLAS online software, these two databases can be unified in the one database, although we can split it into plural files [7].

In Table 2 we have summarized the file list used for the confDB.

i) Module Configuration

As shown in Fig.5, the Module class has simply single layer structure in the confDB description, while the corresponding class “board” has two structures of board and chip in the framework used in SLT2001. We have then implemented a two layer structure in the attribute “Name” of the Parameter “class”. In the SLT2001 framework, for example, the property of GLINK_MONITOR is attributed to the object “csr” of the class “chip”, which is a subclass of the board object “ROD”, while it is a simply one (id = GLINK_MONITOR) of “parameter” class objects (total 23). “Module” class object with id=ROD001 contains those “parameter” objects. Then in order to introduce Board-Chip strucutre in the ConfDB, we bring the layered naming convention in the corresponding NAME attribute of the class “parameter” like the following fragment of XML description;

```
<obj class=”parameter” id=”GLINK_MONITOR”>
<attr “Name”  type=”string”>
```

```
"Csr/GlinkMonitor"</attr>
<attr "Value" type="string">"0"</attr>
</obj>
```

Otherwise simple accommodation to the ATLAS confDB has been done.

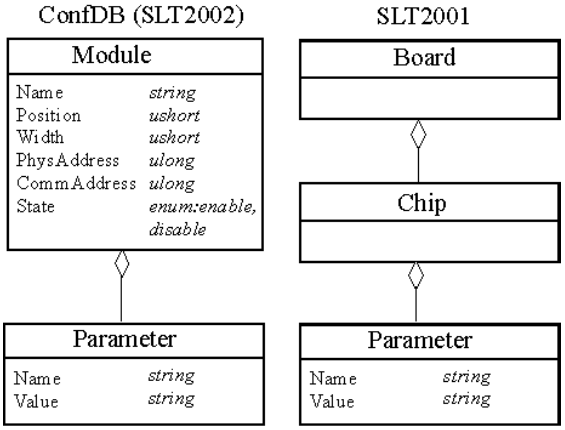

Figure 5: Comparison of the Database structure of one used in 2001 (right) and one of ATLAS online software (left)

ii) Partition Configuration

Partition class is used to define for both software and hardware as a unit of the data acquisition. The concept of Partition has been introduced originally to separate a set of the detector elements (elements for the detector, data acquisition, back-end system and software etc.) into disjoint subsets so that every element belongs to one set. Thus the Partition class defines the hierarchically top classes to be used in particular set of the data acquisition elements. One partition class is defined in the present DB. As hardware, the Partition class describes the set of detectors, set of crates to participate the data acquisition, and as software, it defines set of software applications, the database files to instantiate the partition class itself. Fig.6 shows the Partition class diagram for the SLT2002 configuration (id =slt-trg), which is produced by the embedded GUI system.

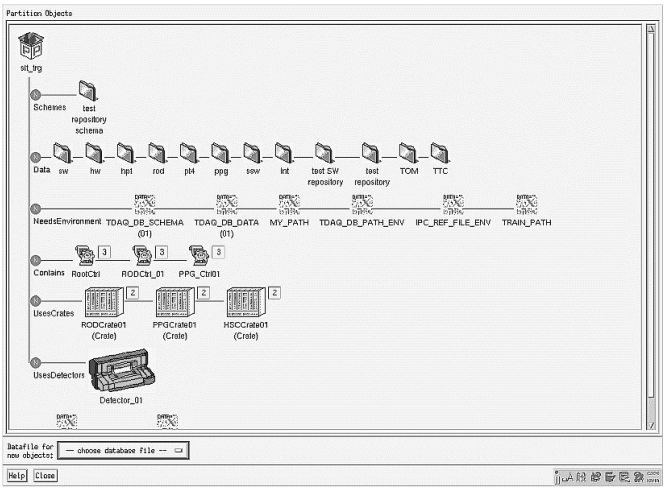

Figure 6: GUI for editing confDB (partition)

2) Run Control

An example of the run control GUI window is shown in Fig.7. Run control is built from the skeleton that is provided by the online software group.

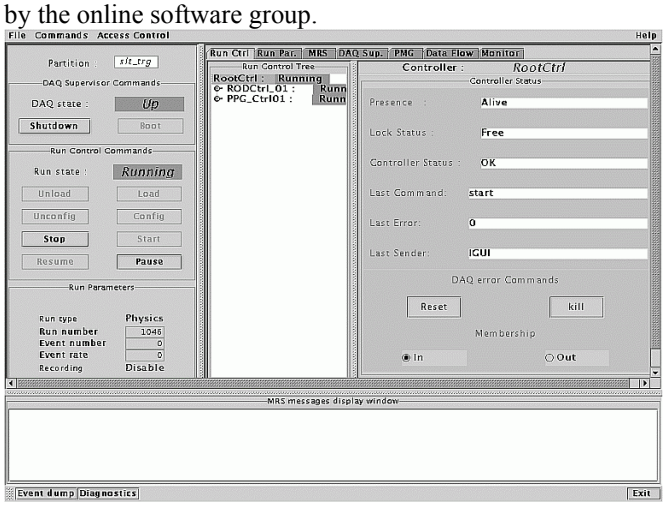

Figure 7: Run control GUI window used in SLT2002 based on ATLAS online software framework

IV. Discussion and Summary

One of the purposes of SLT is to confirm whether a particular input signal combination will give a correct LVL1 signal with our intended algorithm. We have also started to build a simulation code for the LVL1 trigger generation in the endcap muon region independently. Taking an input signal set from the simulation, providing it to the SLT system through the PPGs and comparing the actual output result with one predicted by the simulation, we can conclude the validity of timing, logics implemented in ASICs and stability of data communication links between modules.

The validity check can be performed only after we can setup the hardware (module configuration) and appropriate input/output data set correctly. As regards this point, therefore, the control (for modules and runs) software system is in the leading role for efficient SLT running and system debugging. Since we could achieve practical SLT and system verification, the validity of our software system has been proven.

Based on the experience acquired in the software development of the first year (2001), we have constructed further efficient and reliable software system à la ATLAS online software framework (2002). Although we would foresee some difficulty and hard work especially in the adaptation of the databases, we could end up the modification without troubles. This is because of similarity of structure of the databases (both have been described in object-oriented basis), and flexibility of the database access library (DAL) prepared in the ATLAS online software.

As the next step, we would complete the system by adding histogram data processing and event display using the infrastructure embedded in the online software. This evolution is to make a system capable for both more elaborate SLT and for future beam test.

We have used PCI-VME I/F throughout SLT, but we must construct a flexible system, which can be run with single computer boards.

V. ACKNOWLEDGEMENTS

This work was done in Japanese contribution framework of the ATLAS experimental project. We are grateful to all the members of Japanese ATLAS TGC construction group and KEK electronics group. We would like to express our gratitude to Prof. Taka Kondo of KEK for his support and encouragement.

VI. References

[1] ATLAS First Level Trigger Technical Design Report CERN/LHCC/98-14 (1998) and TDR home page: http://atlasinfo.cern.ch/Atlas/GROUPS/DAQTRG/TDR/tdr.html

[2] K.Hasuko et al., "First-Level End cap Muon Trigger System for ATLAS", Proceedings of LEB2000, Krakow, Poland, Sept., 2000, pp328.

[3] H.Kano et al., "Results of a Sliced System Test for the ATLAS End-cap Muon Level-1 Trigger", presentation in this workshop

[4] N.Kruszynska, VME bridge driver vmehb for Linux, http://www.nijhef.nl/~natalia/projects/vmehb.html

[5] OMG, CORBA, http://www.corba.org

[6] Trolltech inc: Qt, http://www.trolltech.com/products/qt/

[7] ATLAS Configuration Database User Guide, http://atddoc.cern.ch/Atlas/Notes/135/Confdb-ug.html

[8] ATLAS Run Control User Guide, http://atddoc.cern.ch/Atlas/Notes/107/

Table 1: Summary of modules used for the slice test in 2001 and 2002

Name	Function	Module	ASICs/Main Chips	Main Task
Service Patch Panel	Fan-out of TTCrx signals	N/A	TTCrx	
PS pack	Synchronization Low pT Coincidence	N/A	Patch Panel (PP), Slave Board (SLB), JRC	Trigger Readout
High pT	High pT Coincidence	VME	HpT* G-link Tx	Trigger
HSC	Remote VME crate controller	VME	CPLD, JTAG bus controller	Control
CCI	Local VME interface for Remote VME crate	VME	G-link Tx/Rx	Control
Star Switch (SSW)	Readout Transmitter	VME	FPGA,G-link Tx	Readout
PT4	Prototype Star Switch	VME	FPGA,G-link Tx	Readout
Sector Logic	R-ϕ coincidence	VME	FPGA,G-link Rx	Trigger
ROD	Readout Driver	VME	G-link Rx, TTCrx, S-link Tx	Readout
PPG	Pulse Pattern Generator	VME		

Table 2: Summary of files used for the configuration database for the slice test 2002

DB file	Main classes	Data to be described
slt_trg.data.xml	RunControlApplication Partition	Applications to be run-controlled Definitions of Partitions
slt_trg.hw.data.xml	Detector Module, Crate, Workstation	Computers, hardware components with their connections and relations
slt_trg.sw.data.xml	SW object Program	Controller software objects, and program implementations
PPG.data.xml	Module Parameter	Properties of PPG module
HPT.data.xml ROD.data.xml TTC.data.xml SSW.data.xml	Module Parameter	Properties of TGC electronics Modules, HpT, ROD, TTC….