Development of DAQ-Middleware

Y Yasu[1,5] , K Nakayoshi[1], H Sendai[1], E Inoue[1], M Tanaka[1], S Suzuki[1], S Satoh[1], S Muto[1], T Otomo[1], T Nakatani[2], T Uchida[3], N Ando[4], T Kotoku[4] and S Hirano[4]

[1] High Energy Accelerator Research Organization, 1-1 Oho, Tsukuba, Ibaraki, 305-0801, Japan
[2] J-PARC Center, Tokai-mura, Naka-gun, Ibaraki, 319-1195, Japan
[3] The University of Tokyo, Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan
[4] National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba Central 2, Tsukuba, Ibaraki 305-8568, Japan

E-mail: yoshiji.yasu@kek.jp

Abstract. DAQ-Middleware is a software framework of network-distributed DAQ system based on Robot Technology Middleware, which is an international standard of Object Management Group (OMG) in Robotics and its implementation was developed by AIST. DAQ-Component is a software unit of DAQ-Middleware. Basic components have been already developed. For examples, Gatherer is a readout component, Logger is a data logging component, Monitor is an analysis component and Dispatcher, which is connected to Gatherer as input of data path and to Logger/Monitor as output of data path. DAQ operator is a special component, which controls those components by using the control/status path. The control/status path and data path as well as XML-based system configuration and XML/HTTP-based system interface are well defined in DAQ-Middleware framework. DAQ-Middleware was adopted by experiments at J-PARC while the commissioning at the first beam had been successfully carried out. The functionality of DAQ-Middleware and the status of DAQ-Middleware at J-PARC are presented.

1. Feature of DAQ-Middleware

1.1. What is DAQ-Middleware based on RT Middleware

DAQ-Middleware [1,2,3] is based on Robot Technology (RT) Middleware [4]. RT-Middleware is an international standard of Object Management Group (OMG) not only for Robotics but also embedded systems. The software package of RT-Middleware was developed by AIST [5]. We have collaborated on study of DAQ-Middleware with AIST since 2006.

Aim of RT Middleware is not to repeat any more cycle of scrap and building in robotics and to make software infrastructure of robotics and embedded systems. OMG international standard, Robotic Technology Component Specification Version 1.0 includes a Platform-Independent Model (PIM) expressed in UML and Platform-Specific Models (PSMs) expressed in OMG IDL [4]. RTComponent (RTC) is a logical representation of a hardware and/or software entity that provides well-known functionality and services. PSM is still independent of programming languages and operating systems.

[5] To whom any correspondence should be addressed.

An implementation of the specification is OpenRTM-aist software package [5]. We use the package, which implements in C++ and Python as programming languages and CORBA as communication protocol.

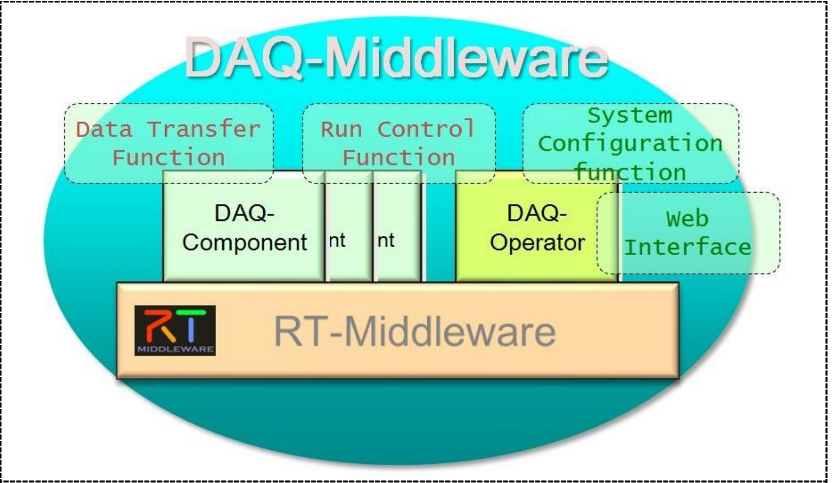


Figure 1. DAQ-Middleware based on RT-Middleware

Figure 1 shows DAQ-Middleware, based on RT-Middleware. DAQ-Component is a software unit used to build an integrated DAQ system. DAQ-Operator is a special DAQ-Component which controls other DAQ-Components. DAQ-Middleware uses databases for system configuration called configuration database and equipment/analysis parameters called condition database. DAQ-Middleware has WEB interface as system interface to communicate with external world. Its protocol is XML/HTTP [6,7]. DAQ-Middleware also has booting mechanism of DAQ-Components over network.

1.2. Autonomous component model of DAQ-Middleware

DAQ-Component inherits InPort/OutPort of RTC for data port and Consumer/Provider of RTC for service port. The data port and the service port are used for data path and command/status path, respectively. The main thread is used for core logic with handling command/status flow and data flow. The thread does not work just in client/server model, but is autonomous with state machine. Those features are shown in figure 2.

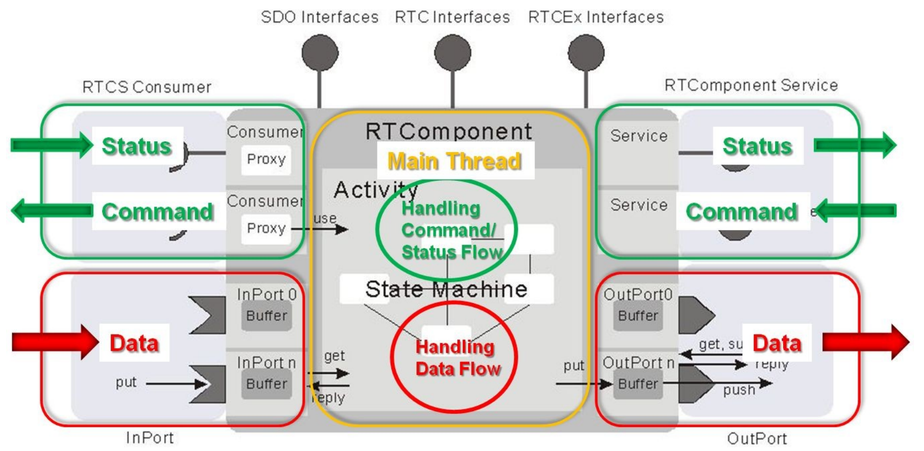


Figure 2. Feature of DAQ-Middleware component

1.3. Configuration and condition databases

XML documents are used for the databases. The configuration database is parsed by a DAQ operator component directly and the parameters are used for dynamic connection of DAQ-Components including data path and command/status path, for an example. On the other hand, the condition database written in XML document is automatically converted into JSON [8] document, which is parsed by a DAQ-Component. The parameters are used for setup of equipment parameters and analysis parameters, for examples. Figure 3 shows the databases in DAQ-Middleware.

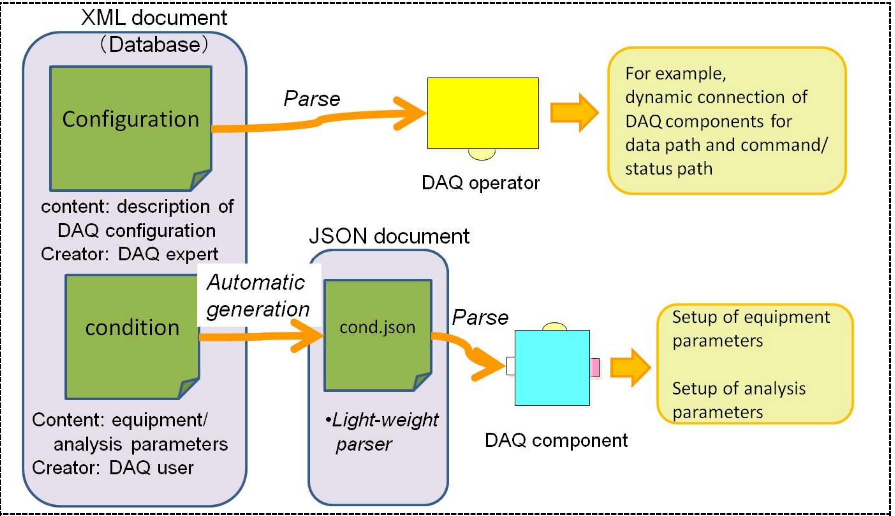

XML document
(Database)
Configuration
content: description of DAQ configuration
Creator: DAQ expert
condition
Content: equipment/ analysis parameters
Creator: DAQ user
Parse
DAQ operator
For example, dynamic connection of DAQ components for data path and command/ status path
Automatic generation
JSON document
cond.json
•Light-weight parser
Parse
DAQ component
Setup of equipment parameters
Setup of analysis parameters

Figure 3. Databases in DAQ-Middleware

1.4. System interface

The system interface is necessary for accessing DAQ operator via User Interface or some management server. We chose XML/HTTP as communication protocol between DAQ-Middleware and the external world. Figure 4 shows the detail. The system interface and DAQ operator are connected by the message passing client program written in python & C++ in DAQ operator, and the message passing server program written in C++ with Apache HTTP server and mod_python in the system interface [9,10].

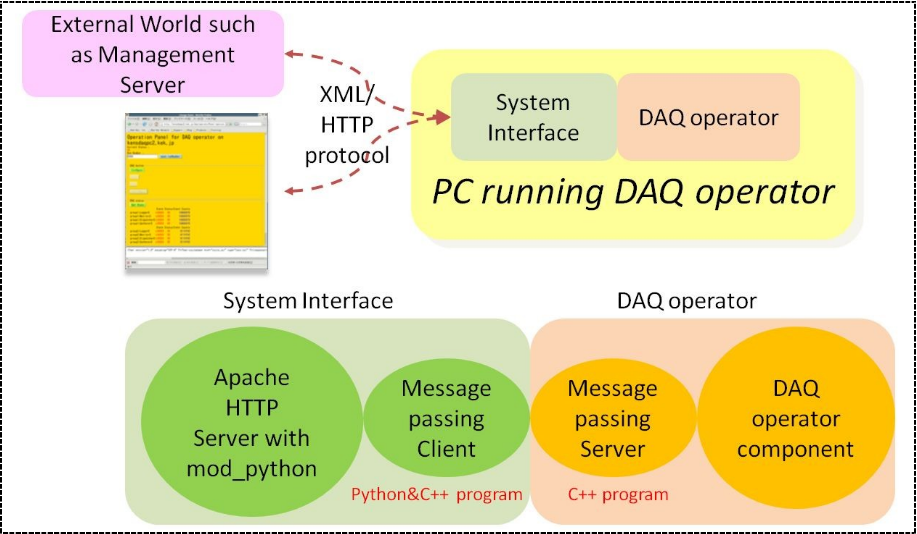


Figure 4. System interface of DAQ-Middleware

1.5. An architecture of DAQ-Middleware based DAQ system

Figure 5 shows the architecture. A basic DAQ unit usually resides on a PC and consists of a set of components required to read and store data that are typically a Gatherer component, a Logger component, a Monitor component and a Dispatcher component. It is assumed that there are a DAQ operator component with system interface on a PC, and basic DAQ units on multiple PCs. A control panel of user interface on WEB browser communicates with the DAQ operator. The DAQ operator reads the configuration file written in XML. After that, the DAQ operator can accept "configure" command from the control panel for system configuration, for an example. In same way, the control panel can issue "begin" and "end" commands to control the readout, the data logging and the online data analysis. The online histogram using ROOT[11] and online displaying on WEB browser are shown in figure 5.

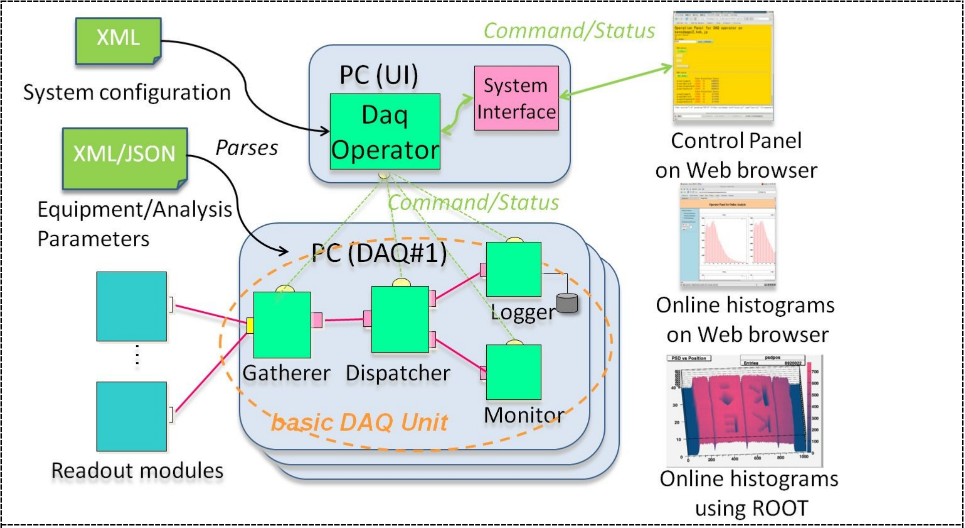


Figure 5. An architecture of DAQ-Middleware based DAQ system

1.6. Features of DAQ-Middleware from user's point of view

In Figure 6, the features from user's point of view are summarized.

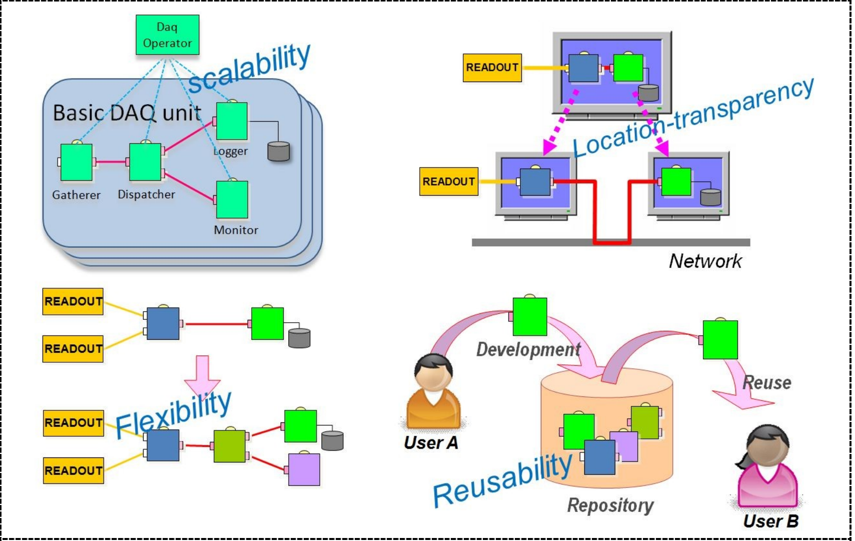


Figure 6. Features of DAQ-Middleware from user's point of view

1.6.1. Scalability. The scalability means that basic DAQ units can be added into the system only with modification of configuration database. For example, you can add a new basic DAQ unit in the system when the readout modules are increased for additional detectors. A DAQ operator can handle not only the previous basic DAQ unit but also new one only with modifying the configuration database.

1.6.2. Location transparency. It is assumed that there are a Gatherer and a Logger in a PC. When the beam intensity increases, the workload of those DAQ-Components may increase. In this case, you can separate the DAQ-Components into different PCs according to the balance of the workload. It is reconfigurable by using configuration database.

1.6.3. Flexibility. It is assumed that there are a Gatherer and a Logger in a PC. When you want to add online data analysis as a Monitor component, you can add Dispatcher and Monitor components into the system. It is easy to add or remove the DAQ-Component in DAQ-Middleware only with modification of configuration database.

1.6.4. Re-usability. When a user makes a DAQ-Component, he will add the component into the repository. On the other hand, another user may use the component with modification of configuration data and/or condition database. The re-usability is important for a community because the re-usage is potentially high.

2. Performance of DAQ-Middleware

The purpose of the performance measurement is to measure the total throughput of a basic DAQ unit of DAQ-Middleware for requirements of J-PARC/MLF [12] which were first target experiments.

2.1. Requirements

2.1.1. Basic DAQ unit. The DAQ unit has a Gatherer, a Dispatcher, a Logger and a Monitor on a PC. A readout PC governs a VME crate (20 modules). Maximum throughput on the module is 1.28MB/s because a module has 8 channels of Position Sensitive Detector (PSD) and the maximum event rate of a PSD is 20kcounts/s (160kB/s).

2.1.2. Total throughput. The average is about 30MB/s on multiple DAQ units.

2.1.3. Way of data logging. Acquired data are stored into files in unit of the module.

2.2. Equipment for readout used by measurement

Typical detector for MLF neutron experiments is PSD. NEUNET [13] is the readout module with SiTCP which is a hardware-based TCP processor [14]. For the measurement, we used NEUNET emulator because the emulator has a traffic shaper circuit to control the data flow of the readout.

2.3. Test-bed and setup

Figure 7 shows the test-bed for the measurement. There are 30 NEUNET emulators for data generation, network switches for gathering data into a PC running DAQ-Middleware.

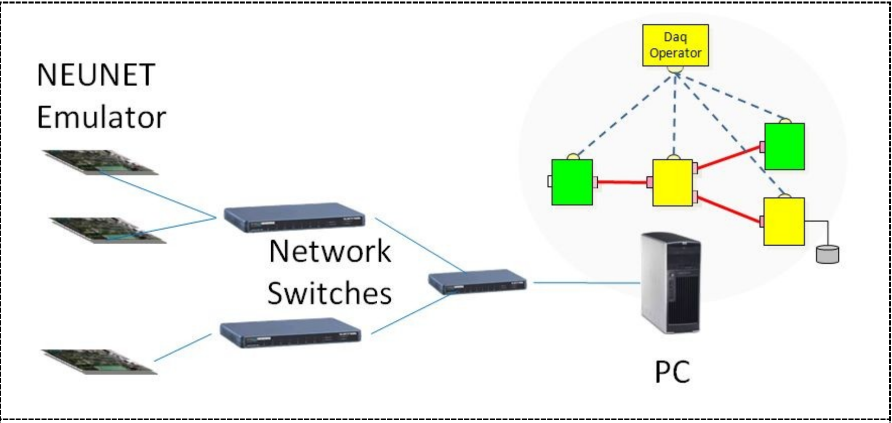


Figure 7. test-bed for performance measurement

The brief description of the equipments and setup parameters is shown in the table 1.

Table 1. brief description of equipments and setup parameters

Parameter	Description
NEUNET emulator	Xilinx Starter Kit x30
Network switch	Cisco Catalyst 2960G-24TC-L x2
	Cisco Catalyst 2960G-8TC-L x 1
PC(HP xw8600)	CPU: Quad-Core Xeon x 2
OS	Red Hat Enterprise Linux Client release 5.3
	kernel: 2.6.18-128.1.1.el5PAE
	gcc: gcc version 4.1.2 20080704 (Red Hat 4.1.2-44)
File System	ext3
Kernel Parameters	/proc/sys/net/ipv4/tcp_rmem 4096 4194304 4194304
	/proc/sys/net/ipv4/tcp_wmem 4096 65536 4194304
	/proc/sys/vm/dirty_background_ratio 1 or 10(default)
	/proc/sys/vm/dirty_expire_centisecs 2999
	/proc/sys/vm/dirty_ratio 40

2.4. Conditions and results

Under several conditions, the measurement was done. First, the total throughput without logging data was measured. There was no data loss.

Second, the total throughput with logging data using single disk storage was measured. Figure 8 shows the results. The horizontal axis is number of the emulator and the vertical axis is the total throughput in MB/s. The data rate at an emulator is 1.28MB/s. The plot data with closed square symbol shows that readout data were lost while the kernel parameter called dirty_background_ratio was default. The parameter means a percentage of total system memory, the number of pages at which the pdflush background writeback daemon will start writing out dirty data. In the next measurement, the parameter was changed to 1 because the daemon will shorten the duration of writing data into disk storage or the size of written data at once. The plot data with closed triangle symbol shows that the total throughput was improved and nearly reach ideal situation, but readout data were still lost around in case of usage of 30 emulators. Finally, the striping disks were used. With usage of striping disks and better value of dirty_background_ratio parameter, the total throughput shown by plot data with closed circle symbol reached ideal situation without data loss of readout.

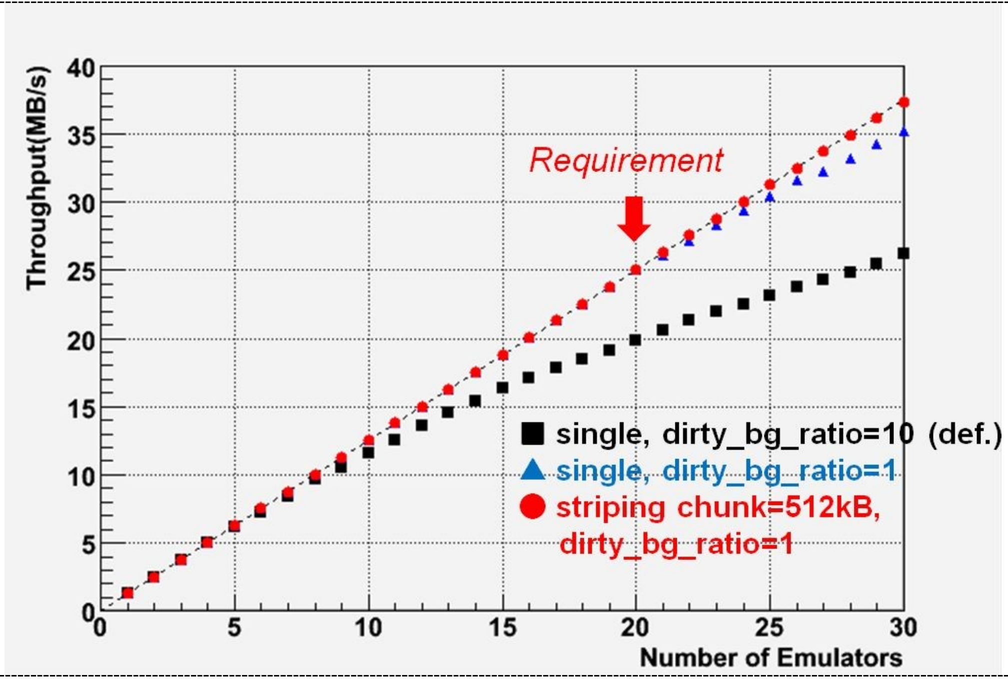


Figure 8. Total throughput w/ data logging

3. DAQ-Middleware at J-PARC

3.1. MLF in J-PARC
The JAEA-KEK Joint Facility for High Intensity Proton Accelerators, called J-PARC, stands for Japan Proton Accelerator Research Complex. It is a high intensity proton accelerator facility using MW-class high power proton beams at both 3 GeV and 50 GeV. J-PARC aims for the frontier in materials and life sciences, and nuclear and particle physics.

MLF stands for Materials and Life Science Facility, which is aimed at promoting materials science and life science using the world highest intensity pulsed neutron and muon beams produced using 3-GeV protons.

3.2. First beam at MLF
First proton beam and produced pulsed neutrons beams were supplied on May 2008, and the extraction of muon beam was also succeeded in on September 2008.

Commissioning using first beam of neutron at iMATERIA (IBARAKI material design diffractometer), Beam Line 20 (BL20), was successfully done on following items;

- Electronics
- DAQ-Middleware
- Software Framework (called Working Desktop)
- Offline analysis

Commissioning of DAQ-Middleware functions are as follows;

- readout and storage of data
- Web-enabled run control
- Web-enabled online monitoring (online histograms for position and time of flight)

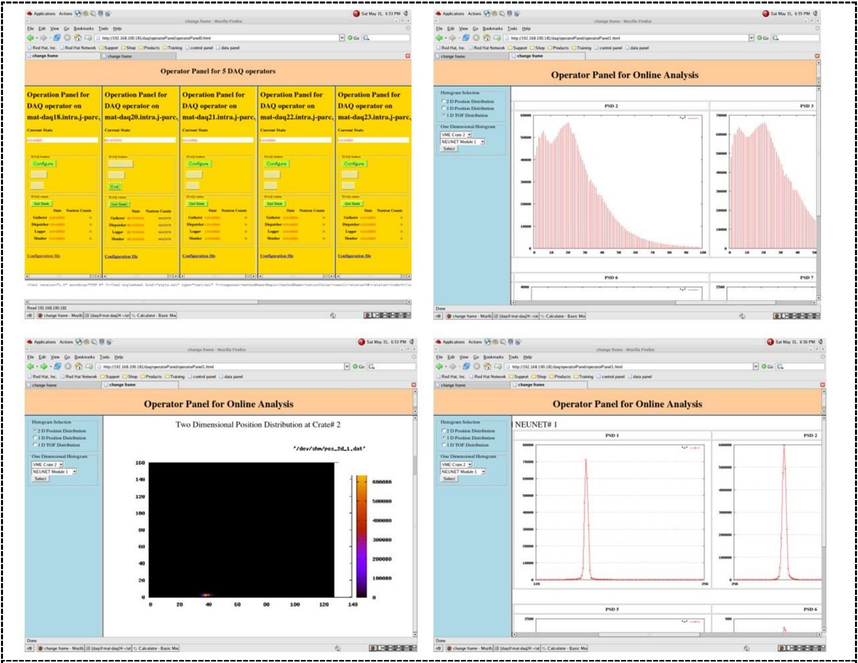
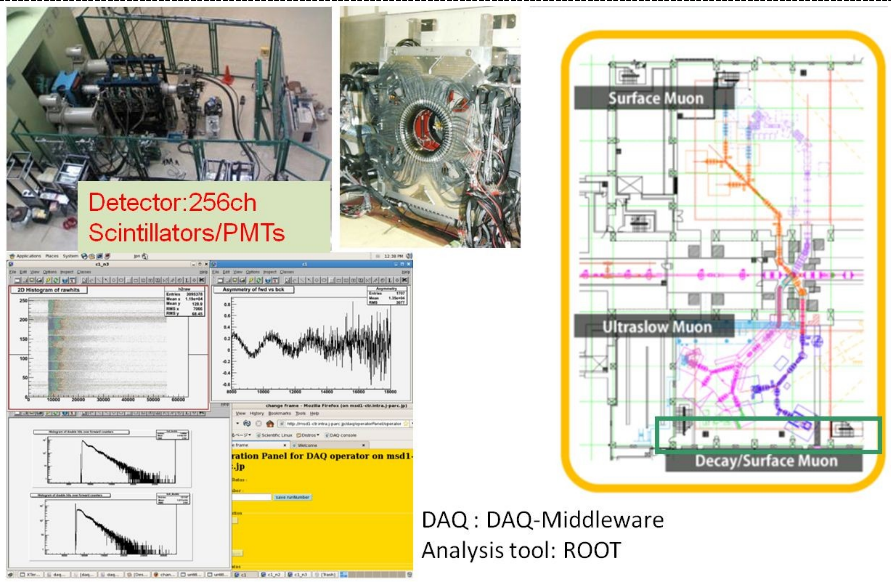

Operator Panel for 5 DAQ operators
Operation Panel for DAQ operator on mat-daq18.intra.j-parc.
Operation Panel for DAQ operator on mat-daq20.intra.j-parc.
Operation Panel for DAQ operator on mat-daq21.intra.j-parc.
Operation Panel for DAQ operator on mat-daq22.intra.j-parc.
Operation Panel for DAQ operator on mat-daq23.intra.j-parc.
Configure
Configuration file
Operator Panel for Online Analysis
PSD 2
PSD 3
PSD 6
PSD 7
Two Dimensional Position Distribution at Crate# 2
NEUNET# 1
PSD 1
PSD 5
PSD 6

Figure 9. snapshot at first beam of neutron: operation panel and online analysis display panels

A snapshot at first beam of neutron is shown in figure 9. There are an operational panel of DAQ-Middleware and display panels of online analysis. The operational panel on top left can send “DAQ command” such as “Begin” and “End” and retrieves “DAQ status” from the DAQ components. Other three panels are two dimensional histogram of neutron’s position, and one dimensional histograms of the position and the time of flight.

Figure 10 shows a snapshot at first beam of muon. At the commissioning, DAQ-Middleware could run successfully, and the time spectrum of μ-e decays and the precession were observed and displayed by the online analysis using ROOT.



DAQ : DAQ-Middleware
Analysis tool: ROOT

Figure 10. snapshot at first beam of muon

4. Conclusion

We have developed DAQ-Middleware, which have been applied to experiments of MLF. DAQ-Middleware has met the requirement.

5. Acknowledgments

We wish to thank Prof. J. Haba at KEK for his support of this project. This work was performed in next generation DAQ sub-group of the KEK Detector Technology Project at KEK. We also thank staff of experimental groups at MLF and KEK electronics system group for their help.

References

[1] Yasu Y, Inoue E, Nakayoshi K, Fujii H, Igarashi Y, Kodama H, Ando N, Kotoku T, Suehiro T, Hirano S, "Feasibility of data acquisition middleware based on robot technology, *CHEP2006 Macmillan Advanced Research Series* p458-461

[2] Yasu Y, Nakayoshi K, Inoue E, Sendai H, Fujii H, Ando N, Kotoku T, Hirano S, Kubota T, Ohkawa T, A Data Acquisition Middleware, *15th IEEE NPSS Real Time Conference*, 2007, doi:10.1109/RTC.2007.4382850

[3] Nakayoshi K, Yasu Y, Inoue E, Sendai H, Tanaka M, Satoh S, Muto S, Kaneko N, Otomo T, Nakatani T, Uchida T, Development of a data acquisition sub-system using DAQ-Middleware, *Nucl. Instr. and Meth.* A **600** (2009) 173-175.

[4] Object Management Group, Specification of Robotic Technology Component (RTC),Version 1.0, *http://www.omg.org/spec/RTC/1.0/*

[5] Ando N, Suehiro T, Kitagaki K, Kotoku T, Yoon W K, RT-Middleware: Distributed Component Middleware for RT (Robot Technology), *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2005),* pp.3555-3560, 2005.08, Edmonton, Canada

[6] World Wide Web Consortium, Extensible Markup Language (XML), *http://www.w3.org/XML/*

[7] World Wide Web Consortium, HTTP - Hypertext Transfer Protocol, *http://www.w3.org/Protocols/*

[8] Crockford D, The application/json Media Type for JavaScript Object Notation (JSON), *RFC* **4627** (Introducing JSON, http://www.json.org/)

[9] Apache Software Foundation, Apache HTTP Server Project, *http://httpd.apache.org/*

[10] Apache Software Foundation, Apache/Python Integration, *http://www.modpython.org/*

[11] The ROOT Team, ROOT, *http://root.cern.ch/drupal/*

[12] J-PARC/MLF, *http://j-parc.jp/MatLife/en/index.html*

[13] Satoh S, Muto S, Uchida T, Tanaka M, Yasu Y, Nakayoshi K, Inoue E, Sendai H, Nakatani T, Otomo T, Development of a readout system employing high-speed network for J-PARC, *Nucl. Instr. And Meth.* A **600** (2009) 103-106

[14] Uchida T, Hardware-Based TCP Processor for Gigabit Ethernet, *IEEE Trans. Nucl. Sci. NS* **55** (2008) 1631.