Journal of Physics: Conference Series **664** (2015) 082016 doi:10.1088/1742-6596/664/8/082016

The Application of DAQ-Middleware to the J-PARC E16 Experiment

E Hamada[1], M Ikeno[1], D Kawama[2], Y Morino[1], W Nakai[3, 2], Y Obara[3], K Ozawa[1], H Sendai[1], T N Takahashi[4], M M Tanaka[1] and S Yokkaichi[2]
[1]High Energy Accelerator Research Organization (KEK), 1-1 Oho, Tsukuba, Ibaraki 305-0801, Japan
[2]RIKEN Nishina Center, 2-1 Hirosawa, Wako, Saitama 351-0198, Japan
[3]The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-0033, Japan
[4]Research Center for Nuclear Physics (RCNP), Osaka University, 10-1 Mihogaoka, Ibaraki, Osaka 567-0047, Japan

E-mail: ehamada@post.kek.jp

Abstract. DAQ-Middleware is a software framework for network-distributed data acquisition (DAQ) systems. We adopted the framework for the J-PARC E16 experiment, which requires a DAQ bandwidth of more than 660 MB/spill (two-second spill per six-second cycle). We developed a prototype for the DAQ software and confirmed that the prototype's total throughput satisfies the requirements of the experiment.

1. Introduction

The aim of the J-PARC E16 experiment [1, 2] is to study chiral symmetry restoration in dense nuclear matter. We measure the mass spectra of vector mesons in nuclei using electron pair decay and numerous statistics. For such purposes, a high intensity proton beam with an intensity of 1×10^{10} per spill (two-second spill per six-second cycle) is used.

The DAQ software must process data generated at a rate of 660MB/spill. To handle such a large quantity of data, network distributed software is required. DAQ-Middleware [3, 4] is a software framework for network distributed DAQ software. We developed the DAQ software by using DAQ-Middleware.

First, we provide an overview of DAQ-Middleware and describe its usefulness in developing the DAQ software. Next, we describe the design and implementation of the system in sections 3 and 4. The results of the total throughput evaluation are described in section 5.

2. DAQ-Middleware

2.1. Overview of DAQ-Middleware

DAQ-Middleware is a software framework for a network distributed data acquisition (DAQ) system. The framework consists of DAQ-Components and a DAQ-Operator. The DAQ-Component is a base unit for software. We can develop a data acquisition path by connecting DAQ-Components. The

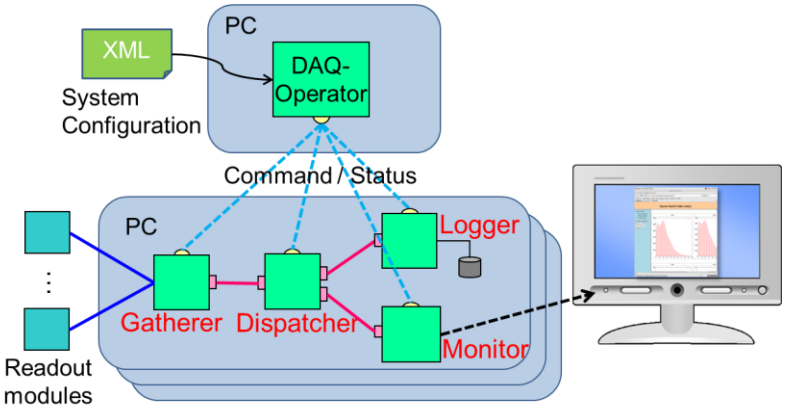

Figure 1. Typical example of software architecture based on DAQ-Middleware.

DAQ-Operator is a controller for DAQ-Components. The DAQ-Operator sends signals to each DAQ-Component, such as start and stop.

Figure 1 shows a typical example of software architecture based on DAQ-Middleware. The software consists of one DAQ-Operator and four DAQ-Components: a Gatherer, a Dispatcher, a Monitor, and a Logger. The Gatherer collects data from readout modules. The Dispatcher sends data to two DAQ-Components: the Logger and the Monitor. The Logger saves data on a hard disk drive. The Monitor analyses data and creates histograms as well as other types of graphs.

Targets of DAQ-Middleware are medium-scale experiments and test benches for sensors and electronics. DAQ-Middleware has been used in 19 experiments, including the Material and Life Science Facility (MLF) [5] in J-PARC and CAlcium fluoride for the study of Neutrinos and Dark matters by Low Energy Spectrometer (CANDLES) [6]. DAQ-Middleware has also been used in nine test benches.

2.2. DAQ-Components and their features

Figure 2 shows the DAQ-Component architecture. The InPort and OutPort are used to create a data path. The Service port is used to create a command/status path. A DAQ-Component contains data transfer, run control, and software configuration functions; these elemental functions are provided by DAQ-Middleware. Users develop new DAQ-Components by implementing core logic that meets their specific requirements [7]. Examples of core logic include the Gatherer reading data from readout modules via Ethernet, and the Monitor generating histograms on the PC display. Figure 3 shows DAQ-Component features.

- Flexibility: Users can flexibly change DAQ component combinations with a minor modification to a configuration file.
- Scalability: Users can deploy DAQ components on multiple PCs in a network. This feature enhances scalability of performance. If the number of detector channels increases, and PCs with DAQ-Components are under a heavy workload, we can add PCs and deploy DAQ-Components on them for load sharing.
- Reusability: Users can employ DAQ components in various DAQ software modules. Users need not develop a completely new DAQ-Component if a similar component already exists.
- Ring buffer: A DAQ-Component with an InPort has a ring buffer. In this DAQ-Component, two threads execute — one thread saves received data in the ring buffer, while the other thread receives the data from the ring buffer and processes the data. The ring buffer is provided by the DAQ-Middleware. Therefore, users need not implement a buffer function to develop DAQ-Components.

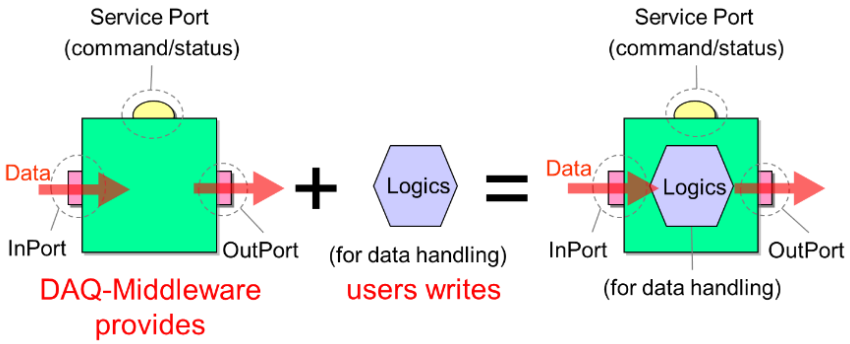

Figure 2. DAQ-Component architecture.

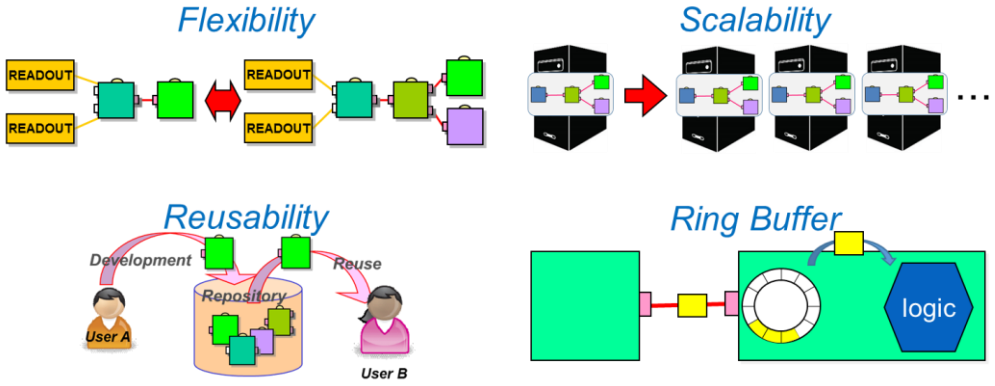

Figure 3. DAQ-Component features.

These features significantly improve the DAQ software created for the J-PARC E16 experiment.

3. DAQ software for the J-PARC E16 experiment

3.1. Requirements for the DAQ software

The requirements for the DAQ software are as follows:

- Required functions: DAQ software must contain a function to save all data on a hard disk drive, and a function to analyze all data from one of multiple events and display the result.
- Required data rate: The proton beam-on time (spill length) is two seconds in every six-second cycle. During one cycle, DAQ PCs receive data for two seconds. The expected data rate is 660 MB/spill with a trigger rate of 2 k/spill. Because of the two-second spill, the average data rate is 330 MB/s with an average trigger rate of 1 kHz. The DAQ software must process data at this rate. The instantaneous trigger rate fluctuates because of the beam rate variation. We estimate the maximum instantaneous trigger rate as 2 kHz and the maximum instantaneous data rate as 660 MB/s. The DAQ software must manage this data even though the trigger rate instantaneously increases.
- Adaptation to the data rate increase for the detector upgrade: We plan to begin the experiment with a few detectors; we will then add detectors in a step-by-step manner to implement the detectors upgrade. Therefore, the number of readout modules and the data rate increase in a step-by-step manner. The DAQ software must adapt to the detector upgrade.

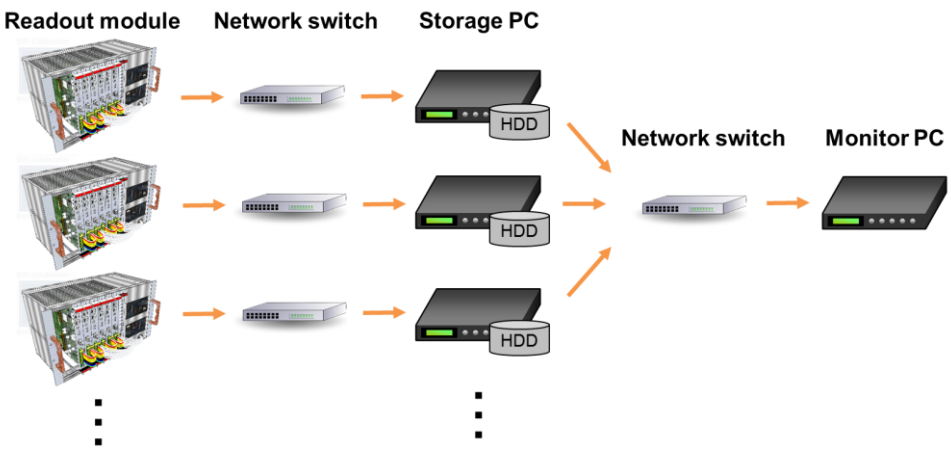

Figure 4. Design of the DAQ software for the J-PARC E16 Experiment.

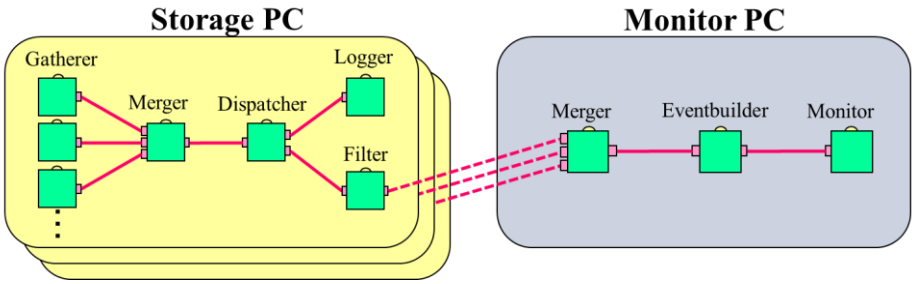

Figure 5. DAQ-Component configuration.

3.2. Architecture of the DAQ software

A single PC cannot handle the required data rate. Therefore, the DAQ software must be installed on multiple PCs. Figure 4 shows that DAQ software is installed on two types of PCs: Storage PCs and a Monitor PC. There are multiple Storage PCs to save all event data. There is one Monitor PC to confirm that the data acquisition executes properly. By using these PCs, we can develop DAQ software that satisfies the required functions.

The Storage PCs have the following functions:

- Read data from the readout modules
- Save data on each hard disk drive
- Send data from one of multiple events to the Monitor PC

The Monitor PC has following functions:

- Build events
- Analyze data and show the result

3.3. DAQ-Component implementation into the DAQ software for the J-PARC E16 experiment

Figure 5 shows the DAQ-Component configuration for the J-PARC E16 experiment. The green boxes represent DAQ-Components.

Each Storage PC contains five types of DAQ-Components. The Gatherer reads data from one readout module. The Merger receives data from multiple Gatherers and sends the data to the Dispatcher. The Dispatcher sends data to the Logger and the Filter. The Logger saves data on a hard disk drive. The Filter sends data satisfying specific conditions to the Merger in the Monitor PC.

The Monitor PC contains three DAQ-Components. The Merger receives data from multiple Filters in the Storage PCs and sends the data to the Eventbuilder. The Eventbuilder builds event data from the received data. The Monitor analyzes the event data and shows the results.

4. Usefulness of DAQ-Component features for the J-PARC E16 experiment

We described DAQ-Component features in section 2.2. These features significantly improve the DAQ software.

When the number of readout modules increases during the detector upgrade, the data rate also increases. The DAQ software can adapt to the detector upgrade, because we can increase the DAQ software's throughput by adding new Storage PCs and modifying the configuration file. This is possible because of Flexibility and Scalability.

Because the DAQ-Middleware package provides sample components, we were able to save time and effort during development. The Dispatcher is the same as the sample component. The Gatherer, Logger, and Monitor were developed by modifying the sample components. This was possible because of Reusability.

The DAQ software can improve throughput by employing the ring buffer. As we described in section 3.1, DAQ PCs receive data for 2 seconds and do not receive data for 4 seconds. This cycle is repeated. While receiving data, DAQ components save the data on the ring buffer temporarily. DAQ components obtain the data from the ring buffer and process the data continuously. Therefore, the DAQ software can use no-event time effectively.

5. The DAQ software Throughput

Using DAQ-Middleware, we developed a prototype and evaluated its total throughput, to confirm whether the required data rate would be satisfied. The prototype consists of two Storage PCs and one Monitor PC. Table 1 shows the specifications of the prototype PCs.

5.1. Evaluation Method

Figure 6 shows the evaluation environment. We developed emulators to use in place of the readout modules. These emulators transmit test data to two Storage PCs. The test data format matches that used by readout modules in the J-PARC E16 experiment. These emulators run one cycle every 6 seconds. During one cycle, the emulators send test data for 2 seconds and do not send test data for 4 seconds. Two Storage PCs receive the data, then save the data on their hard disk drives and send it to the Monitor PC once every ten events. The Monitor PC regularly shows values from portions of data. Under this evaluation environment, we measured the total throughput.

5.2. Measurement Parameters

We changed the number of emulators to change the transfer data rate. The trigger signal was constant and the trigger rate used two patterns: 1 kHz and 2 kHz. When the trigger rate was 1 kHz, the DAQ software always received data at the average trigger rate. Required throughput is 660 MB/spill, as we described in section 3.1. When the trigger rate was 2 kHz, we expected the DAQ software to *always* receive data at the maximum instantaneous trigger rate. As described in section 3.1, the instantaneous trigger rate fluctuates, and the maximum instantaneous trigger rate is 2 kHz. If the DAQ software satisfies the required throughput of 2 kHz, it can manage the data volume even with trigger rate fluctuations. The required throughput is 1320MB/spill (= 660MB/s × 2 second spill).

Table 1. Specifications of the prototype PCs. These specifications affect evaluation result.

	Storage PC 1	Storage PC 2	Monitor PC
CPU	Intel® Xeon® X5650@ 2.67GHz 6Cores	Intel(R) Xeon(R) CPU E5-2640 0 @ 2.50GHz 6Cores	Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz
Memory	24GB	32GB	8GB
OS	Scientific Linux 6.4	Scientific Linux 6.6	Scientific Linux 6.6
Storage	Hitachi HDS724040ALE6 4TB	Hitachi HDS724040ALE6 4TB	

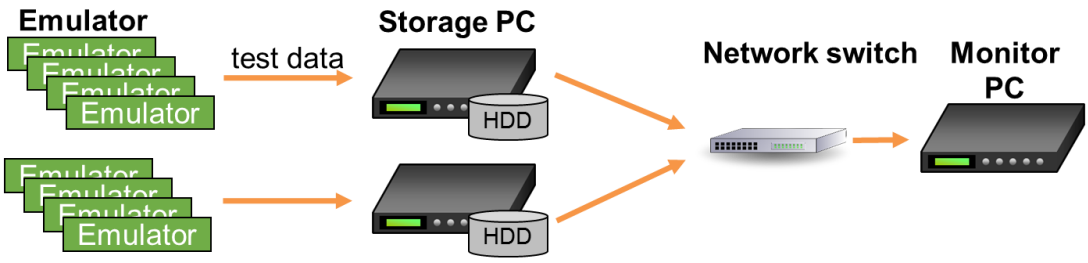

Figure 6. Evaluation environment.

5.3. Evaluation Result

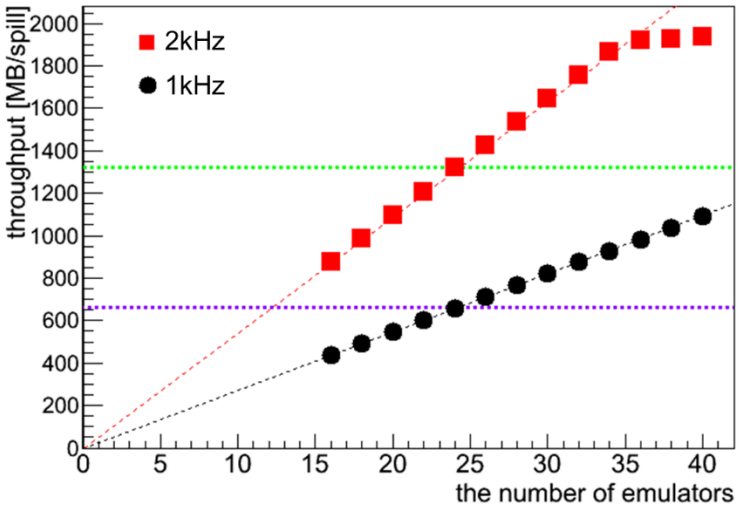

Figure 7. Evaluation results of the total throughput. The x axis indicates the number of emulators. The y axis indicates the total throughput.

Figure 7 shows evaluation results of the total throughput. The black circle points show the total throughput when the trigger rate was set to 1 kHz. The lower horizontal line shows the required throughput. The red square points show the total throughput when the trigger rate was set to 2 kHz. The upper horizontal line shows the required throughput. The saturation is due to a write limit on a hard disk drive. In the case of either 1 kHz or 2 kHz, the total throughput is higher than the required throughput. Therefore, we were able to confirm that the results satisfy the data rate requirements. The results indicate that we can apply the DAQ software to the experiment.

6. Summary

We have developed DAQ software for the J-PARC E16 experiment by using DAQ-Middleware.

DAQ-Components have the following features: Flexibility, Scalability, Reusability, and Ring Buffers. Using these features, the DAQ software can adapt to the detector upgrade, improve throughput, and reduce development time and effort.

We evaluated the total throughput by using DAQ-Middleware with three PCs, and the evaluation results satisfy the requirements. Thus, we can apply the DAQ software to the J-PARC E16 experiment.

References

[1] Kawama D *et al* 2014 Experimental investigation for Mass Modification Effect in Nuclei at J-PARC *Proceedings of Science* PoS(Hadron 2013)178

[2] Takahashi T N *et al*, in this proceedings.

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

[4] Yasu Y, Nakayoshi K, Sendai H, Inoue E, Tanaka M, Suzuki S, Satoh S, Muto S, Otomo T, Nakatani T, Uchida T, Ando N, Kotoku T and Hirano S 2010 Development of DAQ-Middleware *J. Phys.: Conf. Ser.* **219** 022025

[5] Nakayoshi K, Yasu Y, Inoue E, Sendai H, Tanaka M, Satoh S, Muto S, Suzuki J, Otomo T, Nakatani T, Ito T, Inamura Y, Yonemura M, Hosoya T and Uchida T 2010 DAQ-Middleware for MLF/J-PARC *Nucl. Instr. Meth. A* **623** 537–539

[6] Suzuki K, Ajimura S, Nomachi M, Ogawa I and Yoshizawa M 2014 New DAQ system for the CANDLES experiment *R15th IEEE NPSS Real Time Conf.* doi:10.1109/RTC.2014.7097521

[7] Sendai H, Nakayoshi, Yasu Y and Inoue E 2011 Performance measurement of DAQ-Middleware J. Phys.: Conf. Ser. **331** 022039