Journal of Physics: Conference Series **219** (2010) 022036 doi:10.1088/1742-6596/219/2/022036

A DAQ System for CAMAC Controller CC/NET Using DAQ-Middleware

E Inoue[1,2], Y Yasu[1], K Nakayoshi[1] and H Sendai[1]

[1]High Energy Accelerator Research Organization, 1-1 Oho, Tsukuba, Ibaraki 305-0801 Japan

E-mail: eiji.inoue@kek.jp

Abstract. DAQ-Middleware is a framework for the DAQ system which is based on RT-Middleware (Robot Technology Middleware) and dedicated to making DAQ systems. DAQ-Middleware has come into use as a one of the DAQ system framework for the next generation Particle Physics experiment at KEK in recent years. DAQ-Middleware comprises DAQ-Components with all necessary basic functions of the DAQ and is easily extensible. So, using DAQ-Middleware, you are able to construct easily your own DAQ system by combining these components. As an example, we have developed a DAQ system for a CC/NET [1] using DAQ-Middleware by the addition of GUI part and CAMAC readout part. The CC/NET, the CAMAC controller was developed to accomplish high speed read-out of CAMAC data. The basic design concept of CC/NET is to realize data taking through networks. So, it is consistent with the DAQ-Middleware concept. We show how it is convenient to use DAQ-Middleware.

1. Introduction

We have been working to develop DAQ-Middleware in recent years. DAQ-Middleware is a framework based on RT-Middleware (Robot Technology Middleware) and dedicated to making DAQ systems. The Specifications of the RT-Component on RT-Middleware is the international standard for robot systems. Each of the preceding DAQ software was dedicated to a particular use because of poor environment suitable for a framework based on network DAQ. For this reason a very little progress has been made on reusing of the software. Since DAQ-Middleware has implemented DAQ functions as separate DAQ-Components, it is easy to add a new feature of the DAQ. As a DAQ component can run on a separate computer, it is also capable of offering a distributed data acquisition system. DAQ-Middleware includes components with basic functions for the DAQ, thus you are able to construct a DAQ system using DAQ-Middleware very easily. It is only necessary to add a minimal software program needed for you. DAQ-Middleware has an user-friendly interface using HTTP protocol and GUI software can be implemented using the latest Web techniques. We have developed the DAQ system for the CC/NET by adding a GUI part and a CAMAC readout part. The CC/NET is the CAMAC controller module developed to take advantage of many CAMAC assets which we have been using for many years. The CC/NET is well matched with DAQ-Middleware. We expect to show the usefulness of DAQ-Middleware by constructing the DAQ system for the CC/NET.

[2] To whom any correspondence should be addressed.

2. DAQ-Middleware

We proposed DAQ-Middleware [2], which is a framework based on RT-Middleware (Robot Technology Middleware) and dedicated to making DAQ systems. RT-Middleware is a software framework for robot systems using the RT-Component which is the international standard. National Institute of Advanced Industrial Science and Technology (AIST) contributed to set a standard of RT-Component. The Specifications of the RT-Component was adopted at OMG's working committee on September, 2006. This standard specification was officially announced from OMG (Object Management Group). Robotic system does not mean solid type robots, mobile robots or humanoid robots, it refers to systems which have robotic functions. We are now promoting the project, the joint research between AIST and KEK, in order to improve of functions and performances of DAQ-Middleware system [3].

DAQ-Middleware is constructed from uniform components, DAQ-Component, communicating through well-defined interfaces and provides a common platform on RT-Middleware. Every DAQ-Component has a specific function of DAQ. When the system of DAQ-Middleware is running, there are the DAQ-Components sequentially-actuated in accordance with the prescribed activity. DAQ-Middleware is organized as a distributed network of uniform components connected by standard interfaces. Currently one can readily use components such as DAQ Operator, Dispatcher, Logger, Monitor and Reader [4], [5]. The DAQ-Component has a state machine such as configured state and pause state and so on. For communication between DAQ-Middleware and the Web clients, the specific DAQ-Component called DAQ Operator coupled with the Web server, controls other DAQ-Components running in whole system.

The DAQ-Component is the basic structure of DAQ-Middleware and it offers easy and robust features of activity, state transition, some In-Port/Out-Port and of command interface. State transitions of DAQ-Middleware are illustrated in figure 1.

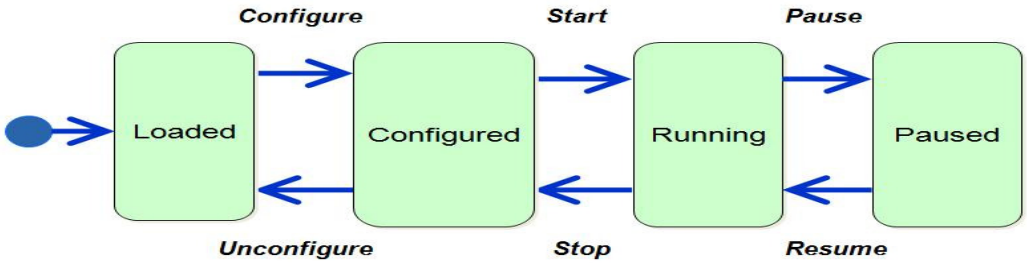

Figure 1. State transition diagram of the DAQ-Middleware system.

The basic activities of the DAQ-Components are assigned to single processes independently of the In-Port/Out-Port and perform asynchronously. We call an In-Port which performs service for process of received data from Out-Port of the other DAQ-Component. The DAQ-Component that requires receiving data stream has the In-Port. We call the Out-Port which performs service for transfer of processed data to the Out-Port of the other DAQ-Component. There are two types of method, push and pull, concerning the data passing between the In-Port and the Out-Port.

The functions of DAQ-Middleware are separated into DAQ Operator component, Reader component, Dispatcher component, Monitor component and Logger component. The block diagram is illustrated in Figure 2.

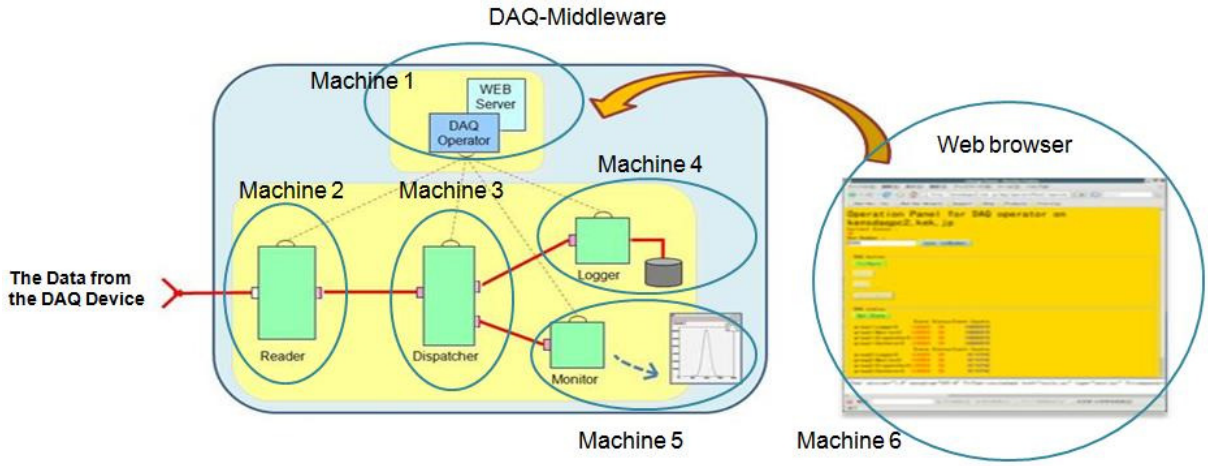

Figure 2. Configuration diagram of the DAQ-Middleware system.

The Reader component forwards the read out data stream which has been received from the detector of the data acquisition system, to its own Out-Port. The Dispatcher component forwards the data stream which has been received from one In-Port, to two Out-Port. The Logger Component stores the data-stream which has been received from the In-Port, to the disk. The Monitor Component can display the data as the histogram using the ROOT framework[6]. The DAQ Operator component can issue a command to control whole DAQ-Middleware system. The shell script for start-up attempt to obtain system configuration information from various name service files, for example, config.xml file, and this shell script will configure DAQ-Middleware system. These files for the system configuration are written by XML syntax. The DAQ Operator component works in cooperation with the Web server. Therefore, you can issue commands to control the entire DAQ-Middleware system via the network with Web browser.

3. CC/NET

CAMAC is the international standard for measurement bus to make data acquisition and control [7,8]. CAMAC is the coined acronym consisting of the initial letters of Computer Automated Measurement and Control. CAMAC has been used for data acquisition in high energy physics and nuclear fusion for many years. So, there are lots of CAMAC resources.

The CC/NET [8], the CAMAC controller was developed by KEK and TOYO Corporation from a joint research to accomplish high speed read-out of CAMAC data. The CC/NET was developed to take advantage of a lot of CAMAC assets which we have been used for many years. The read-out processes of CAMAC data at the CC/NET was improved by using the pipeline techniques. CC/NET has ability to perform read-out of CAMAC data at 2.9MB/s (1.04μs/cycle), this read-out speed of CAMAC data is almost the maximum CAMAC specification speed. In the pipeline read-out method, the multiple command frames and multiple reply frames are processed asynchronously, independently or concurrently. The time from when interrupt signal is received till when CAMAC read-out start cycle is done in less time, 400μs. There is very little dead time associated with the interrupt processing. This controller has a quick response and fast readout of data[9]. The CC/NET has two functions, CAMAC function and DAQ function. The CAMAC function performs to read out data on the CAMAC data-way when LAM interrupt signal is detected. Meanwhile, the DAQ function performs to read out data on the DAQ register on the CC/NET when DAQ interrupt signal had detected. There is a connector on the front panel of the CC/NET to input trigger interrupt signal. We used RAM disk technique to avoid destruction of compact flash memory as there is a limited number of writes the

compact flash memory can take[10]. We wrote a utility to restore the Linux operating system on the compact flash memory to the normal operating status if it was damaged accidentally. This tool is based on 1CD Linux, KNOPPIX[11]. The recovery tool was constructed to incorporate the compressed compact flash Linux system into the KNOPPIX system. The CC/NET user can recover the system of original compact flash Linux by using the 1CD KNOPPIX tool.

The CAMAC driver and library program developed at KEK are available to a wide range of CAMAC users through our Web site [9,10]. Many software examples are included in the standard CAMAC software, the CAMAC user has a good starting point to develop a new application software. There is a basic example program to read out of CAMAC data from remote machine using the TCP/IP socket programming and using the Java RMI. These example programs shows that the CC/NET has a great benefit for construction of the DAQ system at the network level.

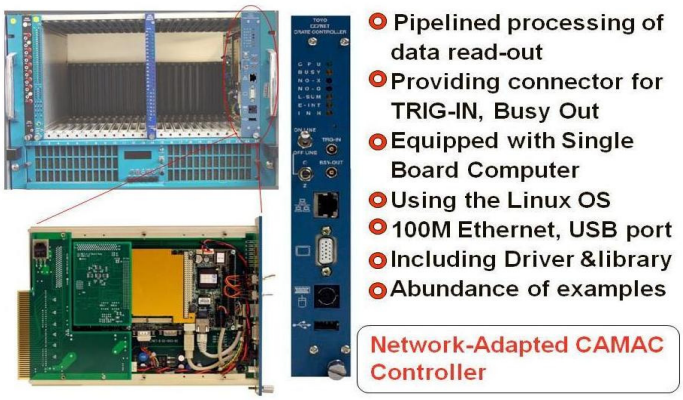

Figure3. Brief overview of the CC/NET.

The CC/NET has a SBC (Single Board Computer) which provide a compact flash memory that Linux operating system ran on. Various Linux boot systems are available such as a standalone Linux system and diskless Linux system. There is a Linux driver program to perform read-out of CAMAC data. It is easy to incorporate the DAQ-Component for CC/NET read out in the DAQ-Middleware system. The single board computer on the CC/NET plays an important role in this case. On the other hand, the SBC have an Ethernet port, it allows easy to access to the internet. The CC/NET has already been on the market at some company and has been used by a lot of user. The basic design concept of CC/NET is to realize data taking through networks. So, it is perfectly consistent with the DAQ-Middleware concept and can be used CAMAC resources effectively.

4. DAQ System for the CC/NET

We developed the DAQ system for the CC/NET using DAQ-Middleware. We added a GUI part to control DAQ-Middleware and a read-out part to access the CC/NET. We modified the Reader component, the software of DAQ-Middleware, to access the CC/NET. This DAQ system is controlled by sending the messages of command from GUI application program to DAQ-Middleware program. The whole DAQ system configuration of this DAQ system is shown in Figure 4. This is an all-in-one DAQ system which includes all functions when you are going to construct the DAQ system for the CC/NET. DAQ system configuration can be changed by modifying the content of the configuration file. This DAQ system is capable of storing and displaying the data from the CC/NET by manipulating the button on GUI screen.

When you issue a DAQ command by clicking a mouse on GUI screen, that message of command is received the DAQ Operator component on DAQ-Middleware through the Web server. The DAQ Operator component transfers a message of command to each DAQ-Component. Then the Reader component perform a task like Configure operation, Start operation, Pause operation, Resume operation, Stop operation and Log operation depending on the incoming command from the DAQ Operator component. The Reader component starts up to read the CC/NET when this component

receives a message to start data taking. The Reader component on DAQ-Middleware accepts to read the data from the CC/NET. The data from the CC/NET is sent to appropriate component through the Reader component to store and display.

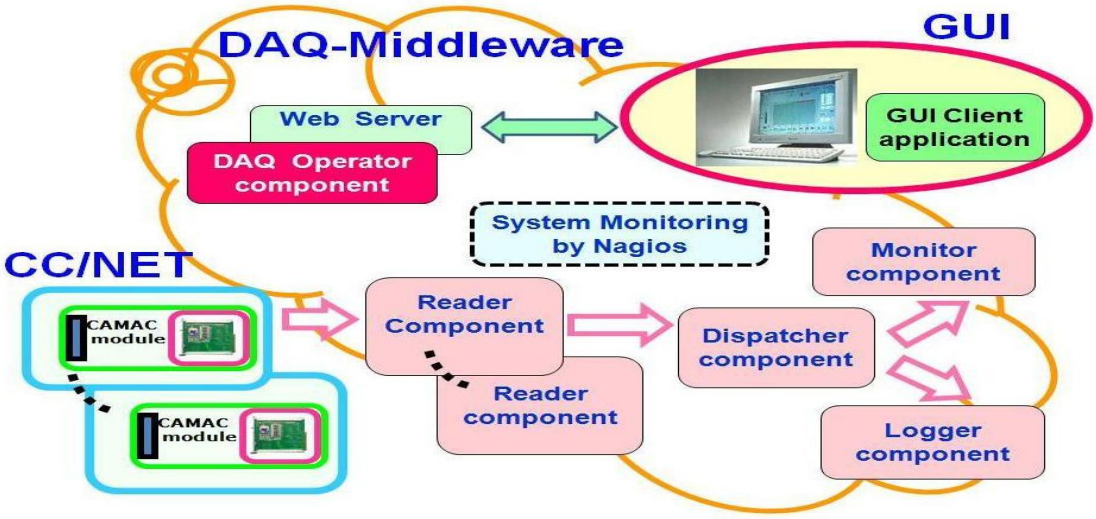

Figure 4. An entire block diagram of the DAQ System for the CC/NET.

4.1. Software Development Environment

We used the VMware [12] for development of the DAQ-Middleware software as we had to develop this DAQ system software together with a remote collaborator. There was a necessity to develop the software program at a place behind a firewall. We included packages necessary for development and execution of the application program into VMWare image and we were able to standardize our development environment using VMWare.

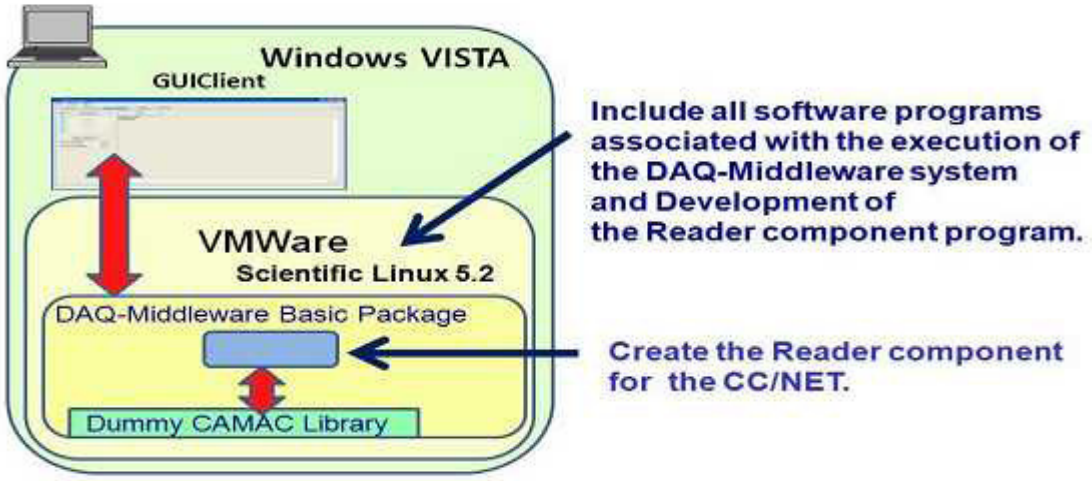

Figure 5. Schematic overview of Software Development Environment.

We decided to use Scientific Linux 5.2 as the operating system on the VMWare image and Windows OS as the base OS on which VMWare runs. The program for read out of the CC/NET is running on Linux OS. On the other hand, the program for GUI software is a program running on Windows OS. We established a communication between the two programs. The VMWare image based on Scientific Linux 5.2 includes all software such as basal DAQ-Middleware packages, RT-Middleware packages and associated packages, ROOT utility, CAMAC library and other required software to construct the DAQ system for CC/NET.

During the software development we used a dummy CAMAC library program which had no access to a CAMAC device and returned a dummy data. Later it was replaced by a real CAMAC library and tested to work correctly. The network capabilities of CC/NET are very convenient because one can access a CAMAC device through the CC/NET connected to the network and it was also used to develop the DAQ system software for the CC/NET after verification that the DAQ-Middleware runs correctly.

4.2. GUI for the CC/NET DAQ System

We developed that GUI software for the CC/NET DAQ system by using Eclipse RCP. There is the RTC-Link tool to provide an easy-to-use, graphical interface to RT-Middleware packages which is being standardised by AIST. The RTC-Link tool is created by using Eclipse. A similar approach was adopted to make the GUI application program for CC/NET.

The view of GUI screen of this DAQ system is selectable by using a tab. On GUI screen, you are able to control DAQ operation and to monitor various aspects of machine status. This GUI application is made up of a Windows program, so we used Windows VISTA capable personal computer.

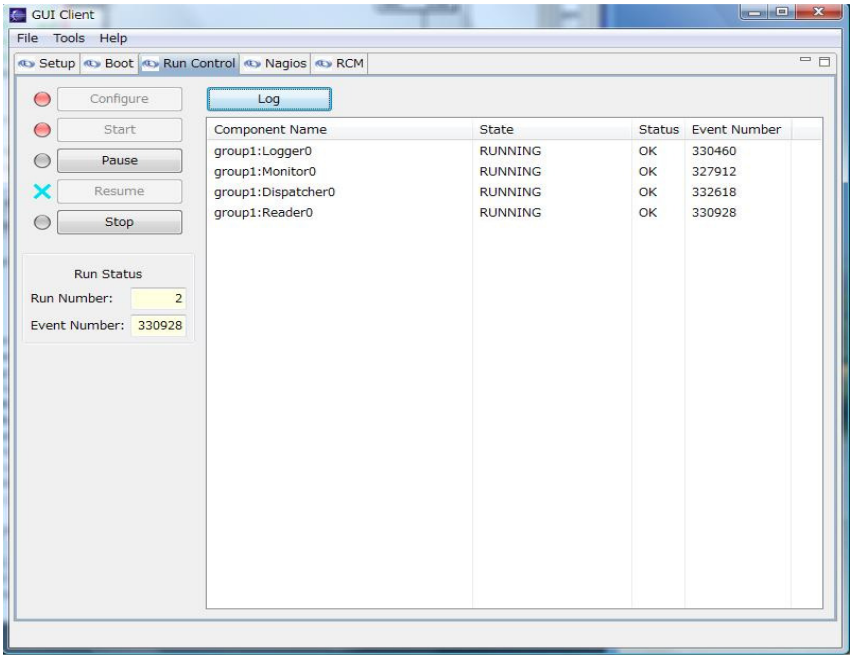

Figure 6. A screen shot of Run Control tab of the GUI.

A screen shot of the Run Control tab of GUI developed by us is shown in Figure 6. The developed GUI software was allowed to select the screen depending on feature using tabbed browsing. Currently, "DAQ Run Control tab" and "Nagios execution tab" are available.

4.3. CAMAC Read-out part for the CC/NET DAQ System

Basal software of this DAQ system, DAQ-Middleware is already in active. At this time what we have to do is, to modify the Reader component on the DAQ-Middleware, and to fit the function of read out of the CC/NET into the Reader component. DAQ-Middleware can be operated under Scientific Linux 5.2. On the other hand, the software for the CC/NET can be operated under Scientific Linux CERN

4.6. The machine for DAQ-Middleware and the machine for the CC/NET, SBC, have been connected to each other via the network. Data passing from the CC/NET to the Reader component is carried out through the network. The CC/NET provides the function of read-out of the CAMAC device remotely using the standard CAMAC library program. This is a function that allows you to control the CC/NET by socket connection through the networks and to read the CAMAC device. We used this function to read the CAMAC device on the Reader component. The conceptual diagram of the CAMAC read out part is shown in Figure.7.

The configuration data file in DAQ-Middleware is written in XML. The description of the condition data file in XML is ideal, but it is too much for the DAQ-Component to it directly. The parameters of DAQ-Middleware are used in the form of NVList which combine name and value. If NVList data type is also used in description on the condition data file, it would be a good idea to simplify the DAQ-Middleware. RT-Middleware is the basis of DAQ-Middleware, and on the framework of RT-Middleware is using the data type of NVList to describe the system parameter on the configuration data file. So we were interested in using this data type on DAQ-Middleware. We used the condition data for the device parameters. We decided to use JSON (JavaScript Object Notation) which was a lightweight data interchange format, as condition data format on DAQ-Middleware. JSON is based on a subset of the JavaScript Programming Language. XML has a lot of flexibility, but sometimes it is redundant. On the other hand JSON can easily be used. We also implemented JSON on this DAQ system software for the CC/NET by following DAQ-Middleware's policy. The contents of the CAMAC command file to be carried out on the CC/NET are written by JSON format. When the DAQ system software is initialized, the contents of the CAMAC command file is loaded into command buffer register after conversion into NVList format. The Reader component program executes read-out operation of the CC/NET using the NVList data of a command buffer register.

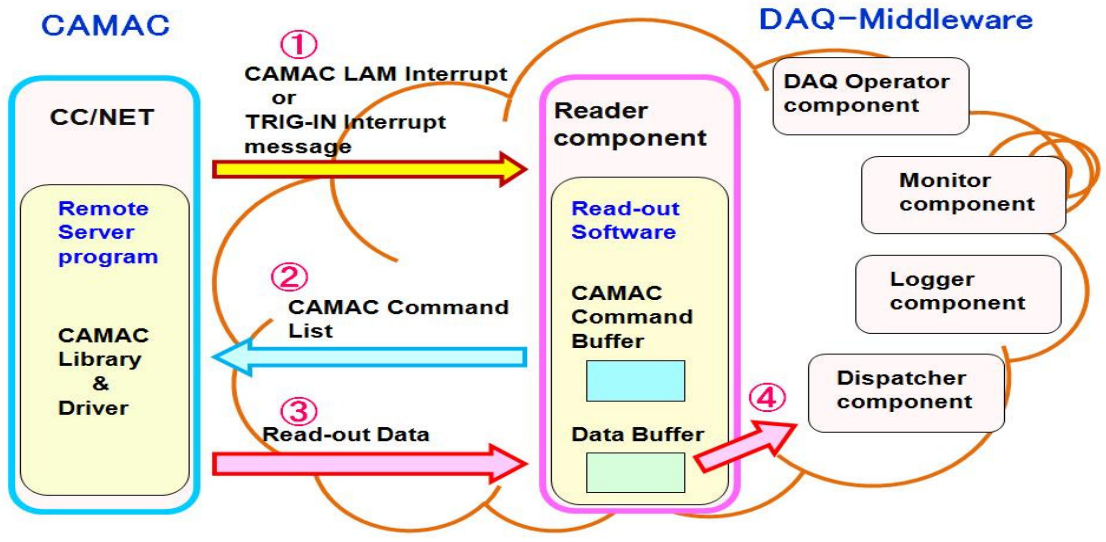

Figure 7. Data passing between CAMAC and the DAQ-Middleware.

The functions of the Reader component in this DAQ system are as follows. The Reader component sets up a list of the CAMAC command sequence and executes it. The Reader component initializes the CC/NET. When interrupt message is received from the CC/NET, the Reader component handles it immediately. The Reader component stores the data from the CC/NET into data-buffer. The Reader component transfers the data on the data-buffer to the Dispatcher component. The data buffer size can

be set when this DAQ system is starting. These are the basic operations of the Reader component. These operations are repeated until Stop command is received from the DAQ Operator component by manipulation on the GUI screen.

When a particular interrupt occurs from a CAMAC module to the CC/NET, the CC/NET informs the Reader component accordingly after handling this interrupt event in an appropriate manner. Then Reader component receives this interrupt message through the network and begins to start read-out process from the CC/NET. There are two types of interrupt sources which can be received by the CC/NET. These are a CAMAC LAM interrupt signal source and a TRIG-IN interrupt signal source. The Reader component is capable of responding appropriately to both of these interrupt sources.

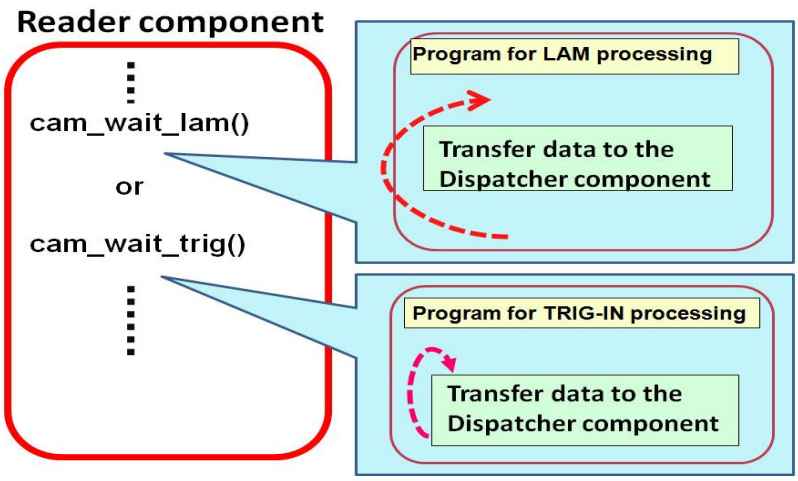

Figure 8. The Reader component software design.

The Reader component is able to execute a block of command line on the command buffer in linear sequence to read the data from the CC/NET. This method is too efficient in taking the data from the CC/NET. The data block from the CC/NET is temporarily stored into the data buffer. When read-out process from the CC/NET has completed, the block of data on the data buffer is transferred to the Dispatcher component.

The process of a series of CAMAC read-out is completed on termination of data transfer to the Dispatcher component. After that, the Reader component program wait next interrupt event to occur. This processing sequence is executed repeatedly till Stop instruction or Pause instruction is received, by clicking the command button on the GUI screen.

We have developed the DAQ system for the CC/NET by using DAQ-Middleware. DAQ-Middleware is operating properly now and is offered to users as a software package. When users attempt to make their own DAQ system, they add a desired function and modify the DAQ-Component for some minimum parts as described above. It will usually be completed the

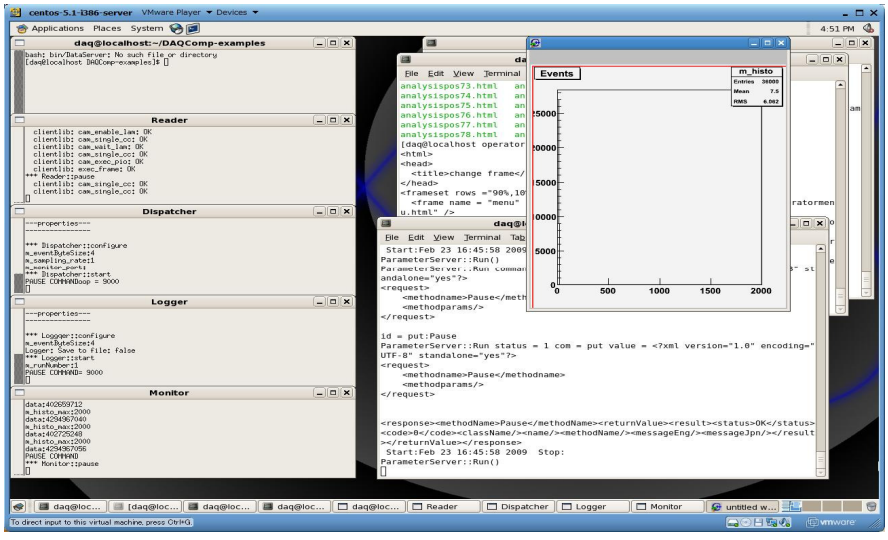

Figure 9. An execution example of the DAQ system for the CC/NET

modifying the Reader component. In addition, it is easy to modify because DAQ-Middleware is constructed from uniform component communicating through well-defined interfaces. There is no side effect on the basic function on DAQ-Middleware or other components by modifying a specific DAQ-Component.

5. Conclusions

With DAQ-Middleware, you can easily construct your own DAQ system. As DAQ-Component is organized by DAQ functions and there is a standardized data exchange between components, it is easy to reuse the software and the expansion of a DAQ function is simple too. Because the system configuration performs in accordance with the description of XML file, it is easy to change the configuration of the DAQ system. It is easy to construct your own DAQ system using DAQ-Middleware and you can get a highly-reliable DAQ system. DAQ-Middleware provides a user-friendly Web interface. We were able to construct the DAQ system for the CC/NET using DAQ-Middleware only by modifying one component and adding a GUI part.

6. Acknowledgments

The authors would like 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 to members of the Electronics System Group at KEK for their contributions to this project.

References

[1] Y.Yasu et al. “Development of a Pipeline CAMAC Controller with PC/104 Plus Single Board Computer”, The 13th IEEE-NPSS Real Time Conference 2003, Montreal, Canada, May 18-23, 2003
[2] Y.Yasu et al. “Data Acquisition Middleware”, IEEE Real Time Conference 2007, Fermilab, Batavia IL, April 29- May 4, 2007
[3] Y.Yasu et al. “Feasibility of Data Acquisition Middleware based on Robot Technology”, CHEP2006, Mumbai, Maharashtra, India, 13-17 Feb 2006
[4] K.Nakayoshi et al. “Development of a Data Aquisition Sub-System using DAQ-Middleware”, International Symposium on Pulsed Neutron and Muon Sciences, March 5-8, 2008, Mito
[5] S.Satoh et al. “Development of a readout system using high-speed network for J-PARC”, International Symposium on Pulsed Neutron and Muon Sciences, March 5-8, 2008, Mito
[6] ROOT framework, http://root.cern.ch/drupal/
[7] IEEE, "IEEE Standard Modular Instrumentation and Digital Interface System (CAMAC) (Computer Automated Measurement and Control) IEEE Std 583-1975
[8] Y.Yasu et al. “Development of a Pipeline CAMAC Controller with PC/104 Plus Single Board Computer”, International Conference on Computing in High Energy and Nuclear Physics CHEP03, La Jolla, California, USA, March 24-28, 2003
[9] CAMAC software for CC/NET, http://www-online.kek.jp/~yasu/Parallel-CAMAC/
[10] CAMAC software for CC/NET, http://www-online.kek.jp/~inoue/Parallel-CAMAC/
[11] KNOPPIX, http://www.knopper.net/knoppix/
[12] VMWare, http://www.vmware.com/jp/?