FEASIBILITY OF DATA ACQUISITION MIDDLEWARE BASED ON ROBOT TECHNOLOGY

Y.Yasu#, E.Inoue, K.Nakayoshi, H.Fujii, Y.Igarashi, H.Kodama, KEK, Tsukuba, Japan,
N.Ando, T.Kotoku, T.Suehiro, Intelligent Systems Research Institute, AIST*, Tsukuba, Japan,
S.Hirano, Information Technology Research Institute, AIST*, Tsukuba, Japan

Abstract

In the situation that many programming languages and communication protocols are flooded, it is necessary to design DAQ system independent of programming languages, operating systems and communication protocols, because the rapidly growing Information Technology and the usage of many PCs in DAQ system lead us to increase necessary DAQ experts and the workload to make the software. In robot technology field, they also had same situation. It was necessary to make an infrastructure of robot systems. Then, Robot Technology Middleware (RTM) was born. AIST* led the project and then developed a software package called OpenRTM-aist. From software point of view, the basic technologies are similar to that of DAQ and the model of software development process is also powerful for DAQ. Thus, we studied the feasibility of Data Acquisition Middleware based on RTM to make a DAQ software framework.

In this paper, DAQ demonstrators based on the RTM were shown and then the feasibility is discussed.

MOTIVATION

In the situation that many programming languages and communication protocols are flooded, it is necessary to design DAQ system independent of programming languages, operating systems and communication protocols. The experiment groups construct their own DAQ software for their own requirements. In particular, the rapidly growing IT technology and the usage of many PCs in DAQ system lead us to increase necessary DAQ experts and the workload to make the software. The software is strictly limited to be used by other experimental groups for their DAQ. As the result, the DAQ software is not interchangeable.

In field of robot technology, there were many robot systems with their own software while their software was not compatible. The robot technology has a huge market and grows rapidly. In this situation, Japan Robot Association drew up a new strategy to make an infrastructure of the robot systems [1]. As the result, RTM was born. AIST led the project and then developed a software package called OpenRTM-aist [2], [3]. The aim of RTM is to establish basic technologies for integrating robot systems.

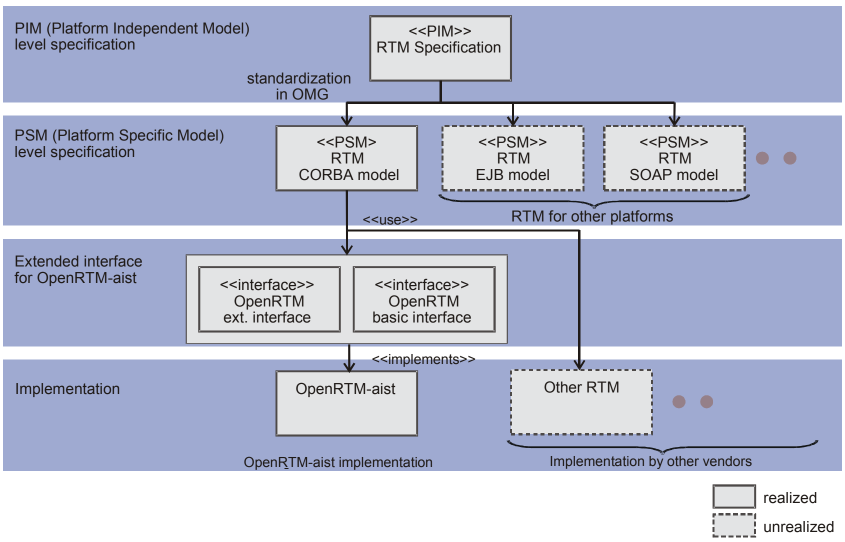

Figure 1: RTM development process model.

#Yoshiji.Yasu@kek.jp
* National Institute of Advanced Industrial Science and Technology

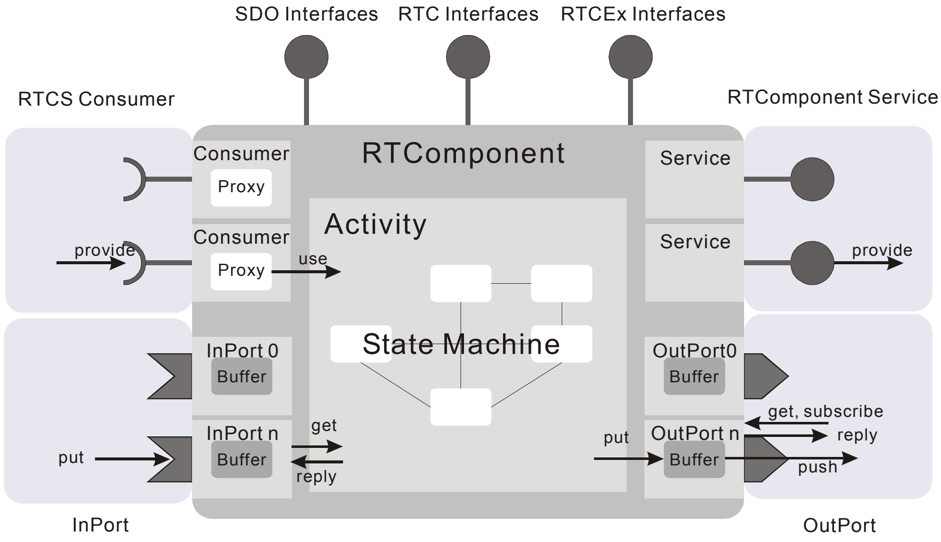

Figure 2: RT-Component Architecture.

Data Acquisition Middleware

From software point of view, the basic technologies are similar to that of DAQ and the model of software development process is also powerful for DAQ. Thus, it is meaningful to study the feasibility of Data Acquisition Middleware based on RTM to make a DAQ software framework.

WHAT IS ROBOT TECHNOLOGY MIDDLEWARE

In the modeling flow of robot system, the system design was dependent of person's experience and know-how so far. However, RT system should be modeled and designed through systematic design flow independent of the person's ones. Object Management Group (OMG) [4] has a model of software development process called Model Driven Architecture (MDA) [5]. It has a hierarchy including Platform Independent Model (PIM) and Platform Specific Model (PSM) layers. RTM adopted the model.

RT specific features are as follows;

- Granularity of modules - various granularity size of module can be considered from fine level such as a sensor to rough level such as an arm of humanoid robot.
- Active module – RT-Component is an autonomous object and works as a task, against a passive object such as a general distributed object.
- Platform independence - RTM does not depend on programming languages, operation systems and distributed object middleware.
- Network independence - RTM has to support

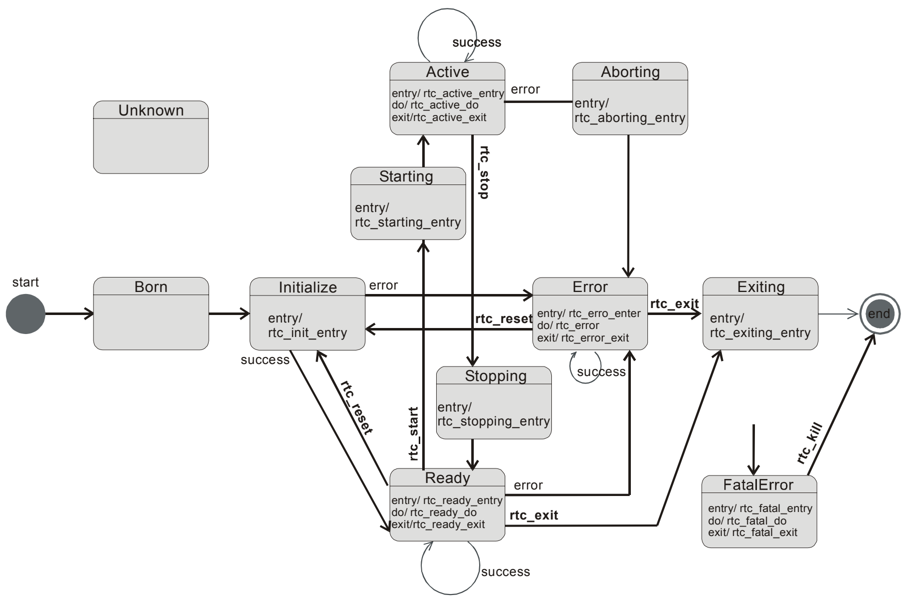

Figure 3: State Machine Diagram of RT-Component Activity.

various communication media and its model should have independent structure from them.

AIST now proposes a RTM development process model shown in Figure 1. It has four layers, namely, PIM, PSM, Extended interface for OpenRTM-aist and implementation [6]. The PIM, PSM and implementation correspond to the layers of development process model of OMG called MDA. The PIM defines resource data model and the interfaces to access and manipulate resource data, described in Unified Modeling Language (UML). Thus, it does not depend on programming languages, operating systems and communication media&protocol. In the PSM, the interfaces and data structures used in the individual methods are mapped according to a CORBA IDL specification. It is still independent of programming languages and operating systems. In the OpenRTM-aist implementation, C++ and Python were adopted as programming language.

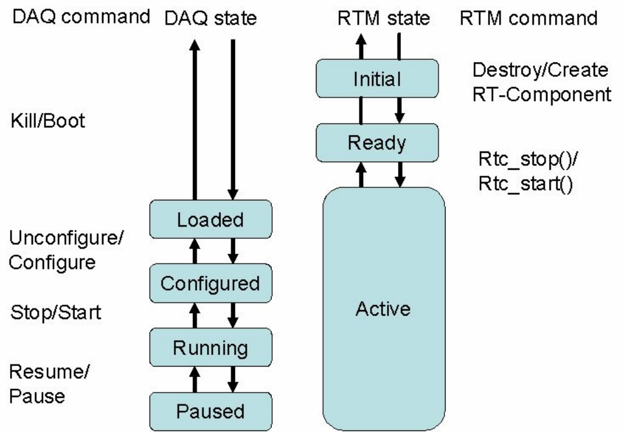

Figure 4: DAQ/RTM state mapping

RT-Component Architecture

Figure 2 shows RT-Component Architecture. RT-Component is basic functional unit of RT-Middleware based systems. RT-Component consists of several objects. InPort and OutPort are data stream input and output, respectively. RT-Component service (RTCS) and RTCS consumer are service interfaces for user command and reply to change internal attributes from outside without re-compiling. SDO interfaces, RTC interfaces and RTCEx interfaces are service interfaces for common command and reply to manage RT-Components.

DAQ/RTM State Mapping

A RT-Component is an autonomous object. It has an activity which always continues processing its task, and the activity serves as a subject of a device control, such as a robot. The State Machine Diagram of RT-Component's Activity and the DAQ/RTM state mapping are shown in Figure 3 and Figure 4, respectively. The "entry" methods and "exit" methods in Figure 3 are called only when changing the states while the "do" methods are continuously called in the steady states. DAQ component also has its own states in Figure 4. A configuration method is called when a configuration command is issued while run method is continuously called after a start command is issued. The state mapping between DAQ and RTM was successfully done in this case.

OpenRTM-aist System Tools

The OpenRTM-aist has some system tools. One is "rtc-template" command, which automatically generates C++ or Python codes by specifying parameters such as InPort and OutPort profiles. Another is rtc-link command, which is a GUI tool to manage RT-Components. Figure 5 shows the snapshot of rtc-link managing the components related to a DAQ demonstrator. It can put the components into the window, connect InPorts to OutPorts by drawing the line and run the components. The InPort and OutPort can be dynamically connected while RT-Component can be dynamically created.

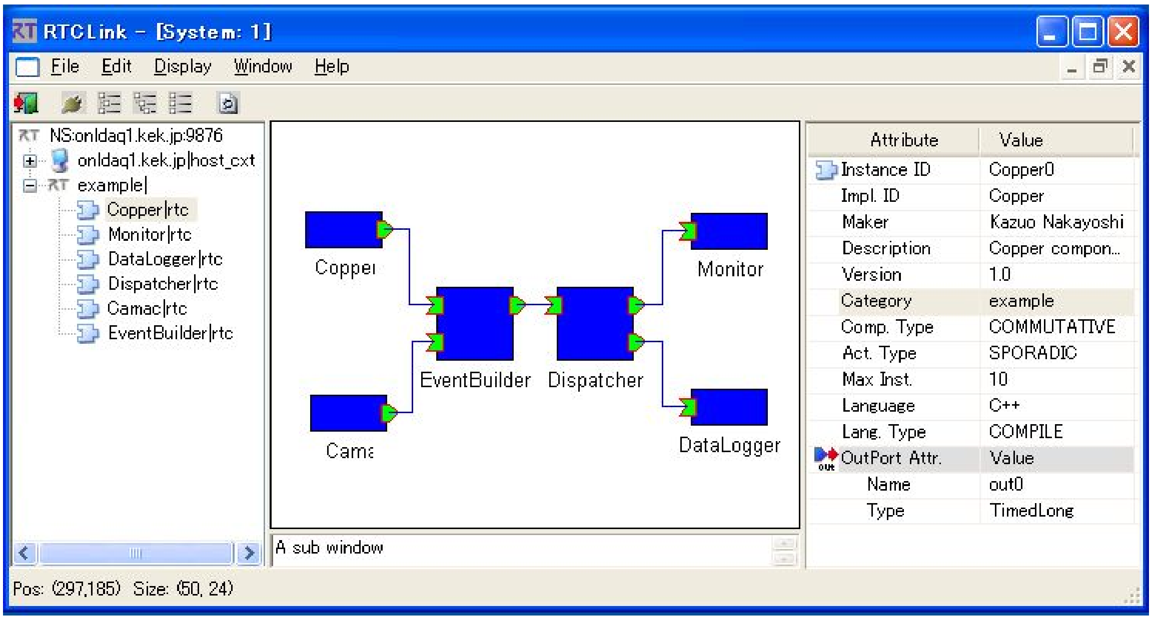

Figure 5: A snapshot of rtc-link

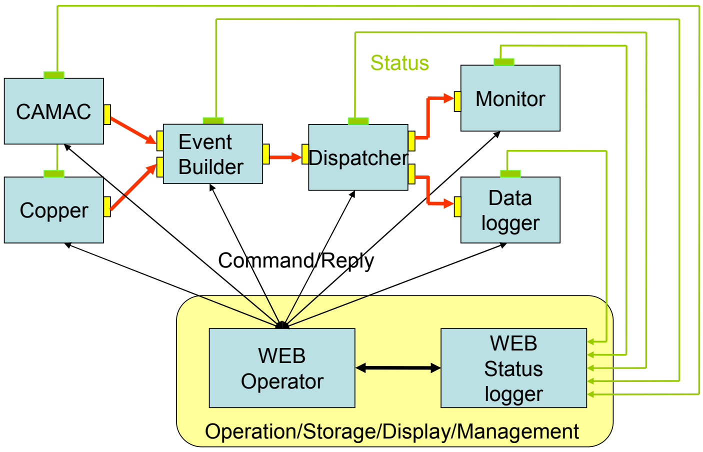

Figure 6: An example of DAQ Demonstrator

DAQ DEMONSTRATOR

We made DAQ demonstrators to evaluate the OpenRTM-aist. A DAQ demonstrator is shown in Figure 6. It consists of data collectors gathering data from CAMAC [7] and Copper [8], a 2x1 event builder, a data dispatcher, a data logger writing data to a file system, a monitor (analyzer with ROOT[9]), WEB operator and WEB status logger. The WEB operator can issue DAQ commands such as "start", via RTC service while the WEB status logger can get the status from the DAQ components via OutPorts. For developing the components, rtc-template to generate the C++ codes was used. Thus, we could concentrate on coding the contents of the DAQ function. As the result, the development time could be reduced.

DISCUSSION

Our aim is to study the feasibility of Data Acquisition Middleware. The viewpoints of the discussion are;

- Model of software development process - RTM adopted the MDA as the model. It is a good way to design a platform independent DAQ system. RTM also adopted component-oriented approach. It is excellent from the reusability point of view.
- Mapping DAQ components to RT components - DAQ states could be mapped to the RTC states in the demonstrators.
- Development time - RTM has the system tools and the template program of RT-Components is also provided. It is useful to reduce the development time to construct DAQ system.
- Dynamic creation and binding - InPort and OutPort can be dynamically connected while RT-Component can be dynamically created. It makes dynamic DAQ configuration possible.

CONCLUSION

The feasibility of Data Acquisition Middleware was studied and discussed. Robot Technology Middleware was introduced, and the RTM-based DAQ demonstrator was made and evaluated. In the demonstrators, the DAQ states were successfully mapped to the RT states. The study showed that RTM had good features for DAQ and then Data Acquisition Middleware was feasible. More investigation is expected.

ACKNOWLEDGEMENTS

The study was supported by Accelerator Science and Technology Support Fund at KEK.

REFERENCES

[1] Japan Robot Association, Summary Report Technology Strategy for Creating a "Robot Society" in the 21st Century, May 2001, http://www.jara.jp/e/dl/report0105.pdf

[2] N.Ando et al., 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

[3] OpenRTM homepage, http://www.is.aist.go.jp/rt/

[4] OMG homepage, http://www.omg.org/

[5] OMG MDA page, http://www.omg.org/mda/

[6] OMG document mars/2006-01-05, "PIM (Platform Independent Model) and PSM (Platform Specific Model) for RTComponent (Robot Technology Component)", 01/23/2006.

[7] Y.Yasu et al. The 13th IEEE-NPSS Real Time Conference 2003, Montreal, Canada, May 18-23, 2003

[8] Y.Igarashi et al., The Data Acquisition System Based on PMC bus, CHEP03, La Jolla, California, March 24-28, 2003

[9] ROOT system homepage, http://root.cern.ch/