Final Program & Digest Book

IEEE CAA AA

SICE-ICCAS 2006

SICE - ICASE International Joint Conference 2006

October 18 - 21, 2006

BEXCO(Busan Exhibition & Convention Center), Busan, KOREA

The Society of Instrument and Control Engineers (SICE), Japan (http://www.sice.or.jp)

The Institute of Control, Automation and Systems Engineers (ICASE), Korea (http://icase.or.kr)

SICE-ICASE International Joint Conference 2006
Oct. 18-21, 2006 in Bexco, Busan, Korea

Development of Light-Weight RT-Component (LwRTC) on Embedded Processor -Application to Crawler Control Subsystem in the Physical Agent System-

Yutaka TSUCHIYA[†], Makoto MIZUKAWA[†], Takashi SUEHIRO[††]
Noriaki ANDO[††], Hiroyuki NAKAMOTO[†††], Akihiro IKEZOE[†††]
[†] Department of Electrical Engineering, Shibaura-I.T., Tokyo, Japan
[††] Task Intelligence Research Group in Institute of Intelligent Systems, AIST, Ibaraki, Japan
[†††] Space Technology and Newer System Business Field, SEC Co., Ltd., Tokyo, Japan

Abstract: Because the RT middleware is only supported by ORBs implemented on PCs, application to a small-scale system and the small-size robot seems to be difficult. To reduce power consumption and developing distributed RT-components on micro processors, application of the RT middleware to the embedded processing system and arbitrary network system are important. In this paper, we report the LwRTC that is the implementation of the RT-component on embedded micro-processors, and the prototype system for the mobile robot using LwRTC.

Keywords: RT-Middleware, Embedded Computer, Controller Area Network (CAN), Distributed Control System, Physical Agent System

1. INTRODUCTION

RT-Middleware (RTM)[1][2] is developed as the middleware for next generation platform of RT systems in AIST supported by NEDO (New Energy and Industrial Technology Development Organization).

The robot system which uses the RTM is consist of distributed object software components called RT-Component (RTC). The robot system is constructed as distributed RTCs on the network.

This research purposes implementation method of RTCs on embedded computer systems to achieve reusability of program and to introduce decentralized and embedded processor system into robot controllers with less power consumption and computing power.

2. PHYSICAL AGENT SYSTEM (PAS)

In the future, robots are expected to spread more widely in society. At that time, robot must be easy to use and to be developed. The design and configuration of robot system, however, depends on its hardware and software. This makes it difficult to develop various robot systems to provide services using independently-designed robot-components.

Physical Agent System (PAS)[3][4] is one of solutions for this request. PAS is tele-operated and semi-autonomous robot. It performs tasks as an agent of a remote operator to support our daily-life.

2.1. PAS Overview

Figure 1 is Outline of PAS. PAS has many devices to control robot such as a joystick, a voice interpreter, sensory glove to detect hand pose, and so on. The commands from those devices are trancemitted to the agent robot via the Internet using the middleware, ORiN (Open Robot/Resource Interface for the Network)[5][6].

2.2. Physical Agent Robot (PAR)

The Physical Agent Robot (PAR) is the robot used in the PAS. Figure2 shows the component-structure of the PAR. The PAR adopts distributed processor control system. This robot consists of functional modules called subsystems that control functional devices (ex. Crawlers, vision unit, etc) in the PAR. Those subsystems are

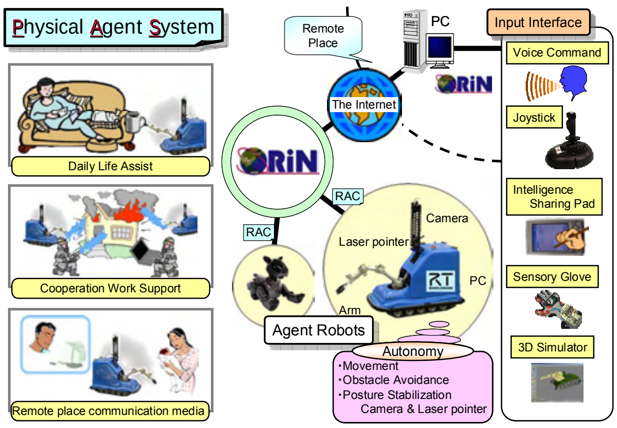

Fig. 1 Outline of Physical Agent System

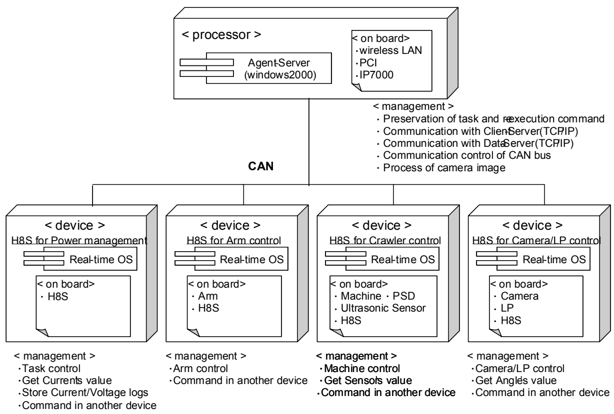

Fig. 2 Structure of Physical Agent Robot

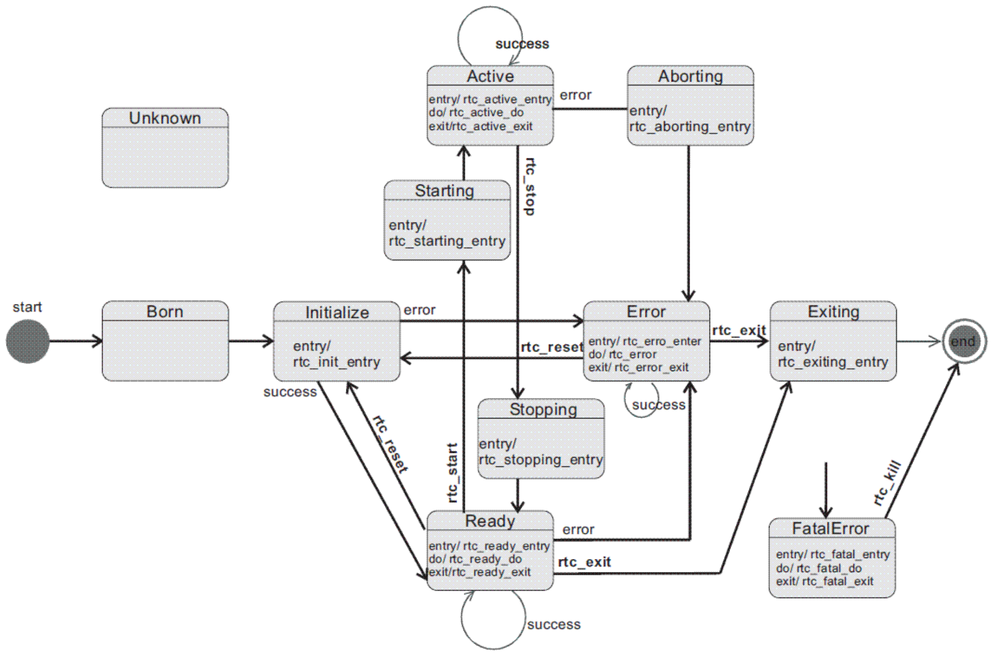

Fig. 3 State-Chart of RT-Component

connected each other via Controller Area Network (CAN). This architecture makes it easy to add and modify subsystems.

3. THE PROBLEM OF DEVELOPMENT TO DESIGNING SERVICES OF COMPONENTS

Development of service for the modules that construct the PAR becomes more and more difficult as extending systems by adding new function modules. In such cases, it sometimes happen that the software for other systems cannot applied and the engineer is forced to develop it from scratch when develop the new modules. Therefore, the robot development cost seems to be increasing.

We develop PAR system as easily adding or customizing subsystems to adapt any requests by connecting functionally distributed subsystems with the CAN. But the efficient method is not exist that combine with subsystems easily. It becomes burden of the developer. Therefore, we have to develop the framework to develop robot service more efficiency.

4. OpenRTM-aist

OpenRTM-aist is one kind of the RT-Middleware that is developed by AIST to reduce system developing costs. It provides common framework of RT component and interfaces as shown in Fig.3. It's based on CORBA (Common Object Request Broker Architecture)[7] technology to construct distributed modules on network. CORBA is standardized by OMG (Object Management Group). The robot system, which uses the RTM, consist of distributed object software components called RT-Component (RTC). Figure 3 shows state chart of RTC. The robot system is constructed as distributed RTCs on network.

On the CORBA environment, it's little problem to adapt software code to new environment. Thus, OpenRTM-aist can take advantages to develop robot systems by providing interoperability and reusability of software.

5. OBJECTIVES OF THIS RESEARCH

We introduce OpenRTM-aist into the development of PAR in order to reduce the development cost. However, two specific issues should be solved to take advantage of RTM to apply it to embedded systems. The issues of this research are discussed next.

5.1. Supporting RTC on the field bus

Since the OpenRTM-aist is based on CORBA, it is effective to develop the robot system which uses TCP/IP communication. But, this means that it is put restriction on robot systems to use TCP/IP communication. It is desirable to apply RTM to arbitrary communication protocols used in various field buses.

➢ Supporting field bus on RTC

Ethernet needs more wiring compared with bus type wiring, because Ethernet is star type wiring. This is serious problem to develop robot system which has to make it compact. Therefore, small volume wiring provided by field buses such as CAN is preferable as an internal network within a robot body.

➢ Reliability of communication

Inside of the robot, the serious EMC noises from the various electronic devises exist. Such noises cause missing of the control data. In such environment, the reliable field bus like CAN would achieve highly reliable communication in the internal network.

5.2. Sipporting RTC on Embedded processors

The RTM only support Linux PC. However, conventional PCs require large electric power consumption than embedded processing processor. And It's impractical to use PCs to provide a simple function like motor–control module even if merits of RTC are took into account.

Therefore, we set these two objectives mentioned above. These can reduce the robot volume as well as suppressing electrical power consumption of the robot.

6. LwRTC ON RTC-CAN SYSTEM

In the PAR, subsystems communicate with each other using CAN. However, the openRTM-aist supports only Ethernet interface. Therefore, we need to develop new system to support CAN without any changes in the openRTM-aist. As the simpler RTC on low level MPU, the RTC-Lite[8] has being developed in AIST. Extending this research, we developed so called RTC-CAN System that is able to pass RTM messages on CAN. Figure 4 is overview of RTC-CAN System. Each subsystem runs their proxy-RTC on the master controller. The mater controller has performance enough to run Linux and CORBA. All CORBA messages from and to both subsystem and external RTC modules are passed through these proxy-RTCs. Each subsystem controller (embedded processor) has software (called CAN-RTC) which supports internal state of the RTC. CAN connects each proxy-RTC and CAN-RTC. This system can provide RTC functions to CAN nodes.

Thus, with this mechanism, it is possible to run RTM on low performance embedded processing system of where CORBA is not supported. In addition, the LwRTC architecture of proxy RTC and CAN-RTC can be ported easily to other device interface networks.

6.1. Supporting CAN interface on RTM

In the RTC-Lite, the interface which connects between proxy-RTC and CAN-RTC were not defined. Therefore, we introduced the gateway which converts Ethernet and CAN protocol each other. We call this "TCP/IP-CAN Gateway". The H8S MPU board which has both interfaces of Ethernet and CAN is adopted as the Gateway. Conversion of protocol is done with the software on TCP-IP/CAN Gateway.

6.2. Software of RTC for subsystem control

There are two problems to make software of RTC for subsystem control.

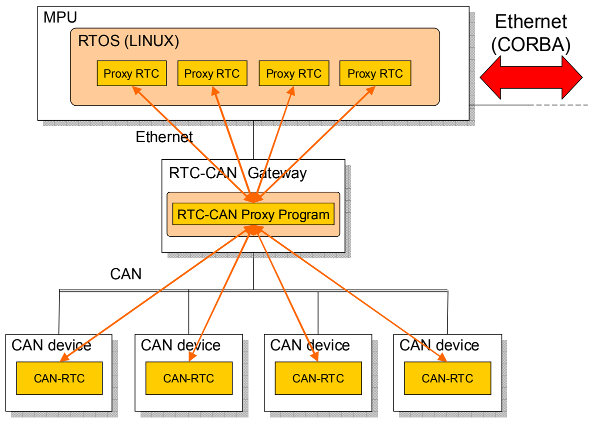

Fig. 4 Overview of RTC-CAN System

- CORBA framework is not supported on H8S MPU
- No support of CAN interface in CORBA

In the RTC-CAN system, RTC is divided into two parts, CAN-RTC and Proxy-RTC. The CAN-RTC takes charge of subsystems' original function. Proxy-RTC takes charge of CORBA communication function. CAN-RTC does not have the communication function using CORBA. But, internal state and the transition are made as specified by the specification of RTM. The Proxy-RTC is allocated on embedded processing system, SH4 which can run CORBA, and CAN-RTC is allocated on subsystem controller, H8S. And we connect between these two with CAN.

6.3. Location of Proxy-RTC

The robot system which consists of embedded processing system with LwRTC, it is necessary to use the server outside. However, in this way, the external server processes all of the communication that cooperation subsystem each other.
Because of that there is a possibility communication bandwidth being wasted with sending and receiving of the control data.

As a result, the communication with the outside can be suppressed to the minimum, because the control of the robot is concluded in the robot. And only the robot makes operation possible without external server.

6.4. RTC-CAN protocol

Communication protocol that using RTC-CAN system is provided originally based on the specification of RTM.

The data communication and the command communication are supported in the RTC-CAN. The command communication is the communication to control internal states of RTC, and the data communication is a communication of the robot to transfer the control data. In control communication, it is

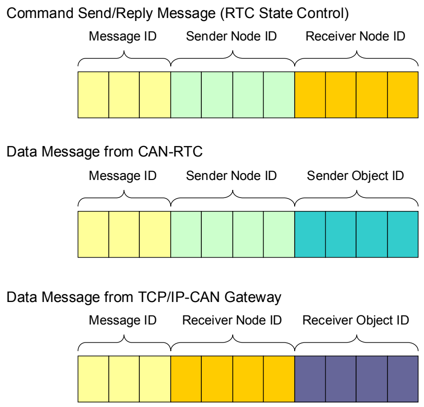

Fig. 8 Using ID field of CAN Message on RTC-CAN

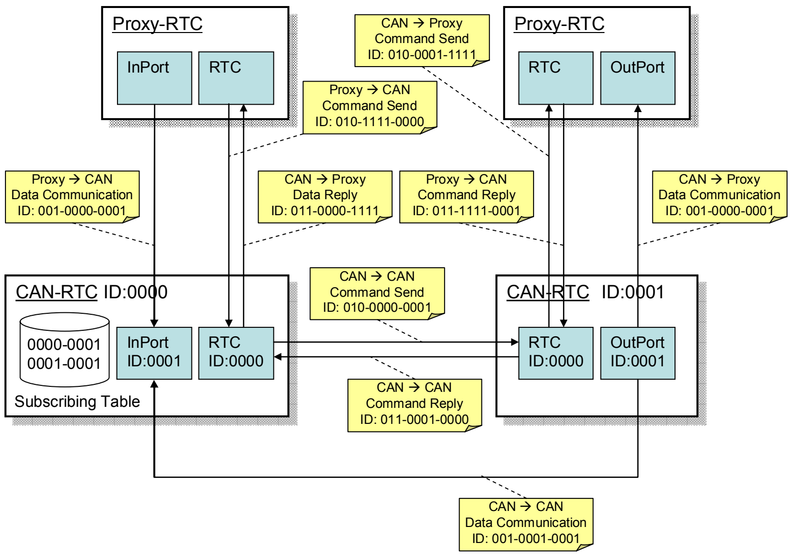

Fig. 7 Example of RTC-CAN Communication

also possible to transmit control messages that cannot be transmitted in one message by dividing into two or more messages.

The RTC-CAN system divides the ID field of the CAN message as shown in Figure 5. And example of RTC-CAN communication is shown in Figure 6.

7. RTC-CAN TEST SYSTEM --CRAWLER CONTROL--

We designed crawler control system of the PAR as a test system to verify performances of RTC-CAN System.

7.1. System Configuration

Figure 7 shows the test system configuration. Table 1 shows hardware we used. This test system was implemented to the crawler subsystem in PAR04R.

Because there's no available micro-CPU-board equipped with CORBA that supports both Ethernet and CAN. Therefore, we adopted the RTC-CAN Gateway using H8S MPU between the master controller and subsystems to convert Ethernet and CAN.

7.2. LwRTC Operation Experiments

LwRTC operation experiments are as follows:

Confirmation of basic crawler operation (forward and back, turns right and left), measurement of communication time

7.3. Experimental Results

We confirmed that the crawler-subsystem is recognized as the RTC, and verified the execution of basic operations (forward and back, turns right and left). Thus, the functions of LwRTC were achieved. Figure 8 shows crawler motion. However, it takes too much time --20ms/one-way, 40ms/roundtrip (practical bound) -- to communicate control data.

Latency of the gateway to convert data is up to 16ms.

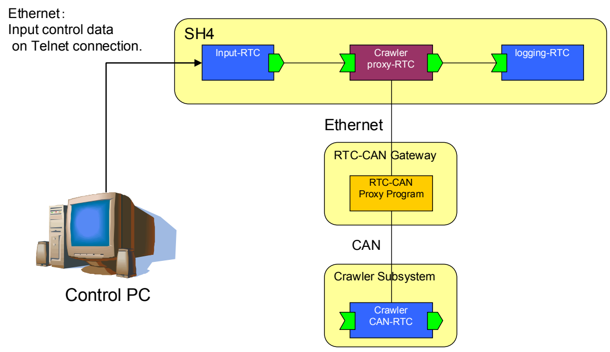

Fig. 5 RTC-CAN Test System(Crawler Control)

Table 1 Hardware List

Embedded System Controller (execute Proxy-RTC)	HRP-3P-CN (General Robotix, Inc.) CPU: SH4 7751R (HD6417751RBP240M)
TCP/IP-CAN Gateway Controller	H8S2638F evaluation kit (hokuto denshi Co.Ltd) CPU: H8S/2638F (HD6417751RBP240M)
Crawler Subsystem Controller	HSB8S2638Q (hokuto denshi Co.Ltd) CPU: H8S/2638WF (HD64F2638WF20 FP-128)

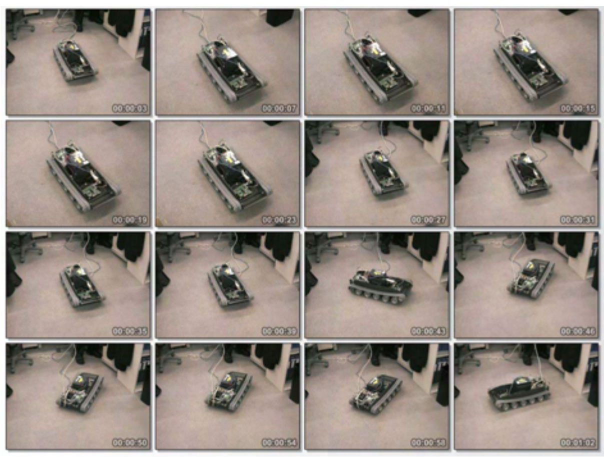

Fig. 6 Crawler control test

This is because low performance of H8S used in the gateway. At this latency, the Gateway cannot accept RTC control command because converting-process consumes whole computing resources. Communication performance is also far from satisfaction. It's difficult to achieve short-cycle and feedback-control under this specific gateway configuration.

8. DISCUSSION

Experimental results shows a possibility to apply RTM to PAR04R by using RTC-CAN system that is the implementation of LwRTC. The proposal of LwRTC confirmed possibility to port RTC in any other network interface without Ethernet.

We confirmed that the proposed LwRTC can achieve RTM functions. However, the performance was too low because of low CPU performance in the gateway.

8.1. Future Works

We should improve performances of the gateway. We are going to propose to stop using this Gateway with higher MPU as well as to apply this system to SH4 board which equipped both Ethernet and CAN.

9. ACKNOWLEDGMENTS

This research is being conducted as a part of the "Project for the Practical Application of Next-Generation Robots" consigned by New Energy and Industrial Technology Development Organization (NEDO).

REFERENCES

[1] N.ANDO, "RT-Middleware Distributed Component Middleware for RT (Robot Technology)", *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3555 – 3560

[2] RT-Middleware Official Web: http://www.is.aist.go.jp/rt/

[3] K.OGAWA, et al.," Physical Agent System: System Concept and Remote Control Service using ORiN", *ISR2005 IEEE 36th International Symposium on Robotics*,TH4H4

[4] "Research of distributed control system for the Physical Agent System(PAS)", Yosuke MATSUNO, Makoto MIZUKAWA, Yoshinobu ANDO, *SICE SI2004*

[5] M.MIZUKAWA, et al.," Implementation and applications of open data network interface 'ORiN'", SICE 2004 Annual Conference, pages 1340-1343,vol.2,2004

[6] Open Robot/Resource interface for the Network (ORiN) Official Web: http://www.orin.jp/

[7] Common Object Request Broker Architecture (CORBA) Official Web: http://www.corba.org/

[8] "Lightweight RT-Component for Embedded systems : RTComponent-Lite", Noriaki ANDO, Takashi SUZUKI, Manabu INAGAKI, Kenichi OHARA, Koutaro OHBA, Kazuo TANIE, *SICE SI2005*