RT MODULE DESIGN FOR HOME AND BUILDING AUTOMATION

Kenichi Ohara, Shinji Yonesaka, Tomohito Takubo, Yasushi Mae, Tatsuo Arai
Osaka University, Osaka, Japan
{ k-oohara, yonesaka, takubo}@arai-lab.sys.es.osaka-u.ac.jp
{mae,arai}@sys.es.osaka-u.ac.jp

Abstract

The design of distributed modules using robot technologies (RT) is proposed to advance the ubiquitous robot environment. Module design involves analyzing the needed functions of a device and designing the architectures of the software and hardware. Modules have to work independently and communicate with each other. Moreover, they need to effectively accept the various requirements of all users. To meet these demands, we developed a module based on our proposed module design by using RTC-Lite, which is an RT-Middleware. framework for computers with limited resources. By using modules based on the proposed module design, a simple home automation system is constructed. We then verify the replaceability and reusability of these modules.

KEYWORDS: Ubiquitous Robotics, RT-Middleware, RTC-Lite, RT Module

INTRODUCTION

Recently, due to the increase of research in the robotics field, many different robot technologies (RT) have been developed, although most have been developed independently. As a result, each technology is rarely reusable for other systems or modules. If a common platform was available, people could use these technologies more easily. In our research, we are examining ways to advance ubiquitous robotics (Ohara et al., 2005), an environmental system in which modules providing RT are distributed.

RT modules make up a network system that enables people and robots to perform tasks such as navigating and interacting within this environment. The system developed by RT modules serves as a common platform for the development of robot technologies. By using one platform, people can apply RT in many different fields. Such modules need a control unit, communication interface and RT element (such as a sensor device). In this paper, an initial RT module design is proposed for development of a home and building automation system as one example of a ubiquitous robot environment. The module design includes analysis of the functions needed and design of the architectures in which these functions are mounted. This approach is important for the development of an RT module that can be used in many different ways. Finally, we present an RT module based on our proposed design and perform a demonstration verifying its capabilities.

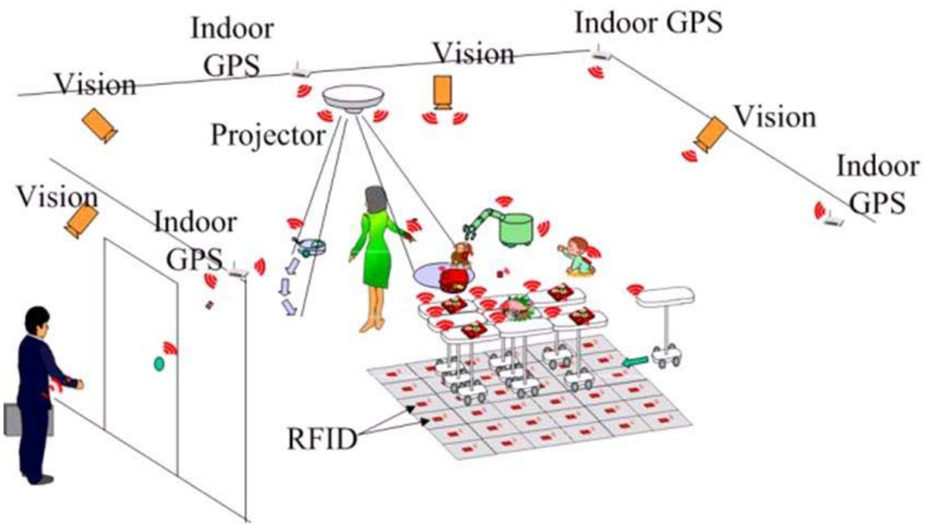

Fig. 1 Conceptual image of ubiquitous robot

MODULE DESIGN

In this paper, "module design" means analyzing the functions that modules should have and designing the architectures of the software and hardware to mount the functions. It is important to consider how to divide RT so that the module operates independently and for a broad variety of implementations. Moreover, people with various levels of expertise, such as novice users and device developers, should be able to use the modules without problems. For example, an ordinary person might want only to change the network connections or function of a module, but a device developer might want to attach a product to the modules so that the product can operate in the ubiquitous robot environment.

Middleware Platform

To construct a robot system with RT, a working network is necessary. Thus, we need a middleware platform. Many kinds of middleware platforms have been developed. ORiN (Mizukawa et al., 2002), RoboLink (Narita et al., 2005), Miro (Utz et al., 2002), Orca (Bal et al., 1992), Microsoft Robotics Studio (Jackson, 2007), Open-R (Fujita and Kageyama, 1997), RT-Middleware (Ando et al., 2005), and so on, are proposed as robot middleware. ORiN and RoboLink support the networking technology for the robot system but do not support the components that control many types of RT modules. Miro is a distributed object-oriented framework for mobile robot control. However, these examples of middleware work only on the Windows OS, which means they are not compatible with other OSs such as Linux. As expected, Microsoft Robotics Studio only works on Windows OS. In contrast, Orca does not work on Windows OS and Open-R only works on Linux. RT-Middleware addresses these problems.

In addition, many kinds of middleware platforms are available for home and building automation systems. ECHONET (Hansen et al., 2007) and LonWorks (Alonso et al., 2000) are often used in this field. LonWorks is primarily used for building automation. In the

LonWorks network, each node has a "neuron chip" that controls each node and constructs the network. Echonet is used for home automation. Most nodes in the Echonnet network are home appliances, which include TVs, video recorders, refrigerators, air conditioners, and so on. This platform does not work in other fields such as building automation. We compare the RT-Middleware of these middleware platforms on the basis of requirements of the ubiquitous robot. Table 1 shows the availability of each requirement in these middleware platforms. ○ means available, △ means partly available and × means unavailable. As shown in Table 1, RT-Middleware is more useful for constructing such a system than other middlewares.

Table 1 Comparison in availability of these middleware platforms

	RT-Middleware	**LonWorks**	**ECHONET**
Open Architecture	○	△	△
Plug and Play	○	○	○
Diversity of distributed module	○	△	×
Cooperativeness of robots	○	×	△
Component of RT function	○	△	×
Participation of poor MPU	×	○	△

RT-Middleware

RT-Middleware (RTM) was developed as a middleware platform by AIST to develop robot systems efficiently. We use this package, which allows many kinds of software to be components, to develop RT modules that easily communicate with each other. We can change the status of the components and switch the network connections dynamically in the GUI (graphical user interface) of the RTM framework. RTM can work on many kinds of computers and OSs. In addition, the AIST web site offers free programs and developers' tools. However, there are a lot of distributed RT modules embedded in computers controlled by limited resources, such as the Microchip PIC in the ubiquitous robot environment. Since computer resources are not adequate for complex control processes, an RT component (RTC), which is a unit of components in RT middleware, does not run on these computers. The cooperation between each RT module and a robot working in this environment is very important for innovation in the field of robotics. To satisfy the demands for a home network system using RT, the RTM framework needs to expand to operate with the limited resources of embedded computers. The concept of RTC-Lite, which is a component design for realizing the concept of RTC on these computers, was thus proposed (Ando et al., 2009).

RTC-Lite framework

To enable computers with limited resources to work in the RTM network, the functions of an RTC are divided into two parts. One part is called the "proxy component" and the other is called the "RTC-Lite component" (Fig. 2). The functions of each component are as follows.

Proxy Component

The proxy component communicates with other components to send or receive data to or from other components. The role of this component is a bridge between other components and the RTC-Lite component.

RTC-Lite Component

The RTC-Lite component is responsible for the state machine of the component and actual implementation of the RT module.

The RTC-Lite component is mounted on computers with limited resources. Many kinds of communication protocols, such as Ethernet, serial communication, Zigbee (Kinney et al., 2003), Bluetooth (Haartsen et al., 2000), and so on, can be used to enable two components to communicate with each other. Therefore, many kinds of devices that have different communication protocols can be plugged into the RTM network. Moreover, the state transition of this component can shift to that of a general RTC, and developers can make a component in the same way. Even on computers with limited resources, we can use RTC by implementing the RTC-Lite framework. The RTC, composed of two components, can act as a general component. Thus, RTC has the potential to serve as many types of different components. So, we have to consider how to make these components for the RTC-Lite framework.

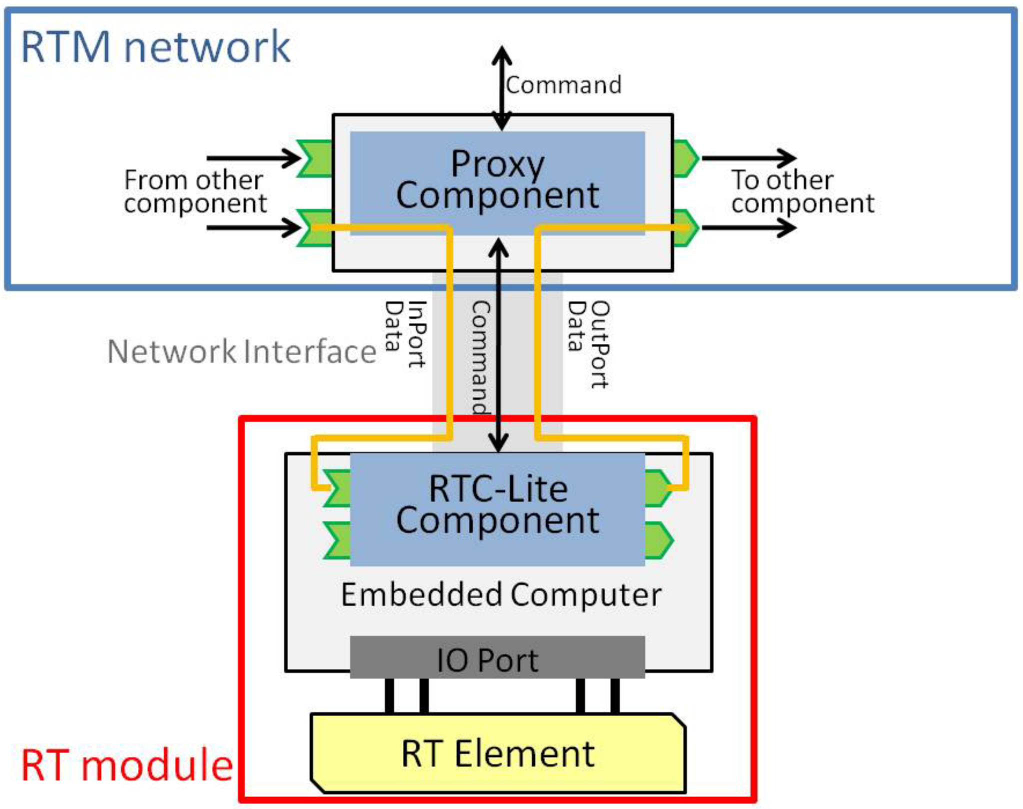

Fig. 2 Architecture of RTC-Lite

Hardware Architecture

To perform its role in the system, the RT module has to have a communication part, control part and RT element part. There are many possible methods for constructing the communication part, including wired or wireless methods. In the control part, various MPUs such as a PIC or H8, which is an 8-bit micro controller, can be used. In the RT element part, sensors, actuators and many devices are used as RT elements. Each part has many patterns, and so users can replace devices for each part. To realize this replaceability of RT modules, the three parts should be separate. With this scheme, users can easily replace a part of the RT module arbitrarily for corresponding devices.

Software Architecture

Software can accept the replaceability of RT module hardware because software is considered to be a component in the RT-Middleware framework. RTC-Lite components are implemented in the control part. The RTC-Lite component is composed of two parts: a communication control software part and an RT element control software part. These two parts are based on different files. Thus, software can be replaced in different ways. Since the RT control part software does not depend on the attached RT element part, one component can be used for different RT element parts.

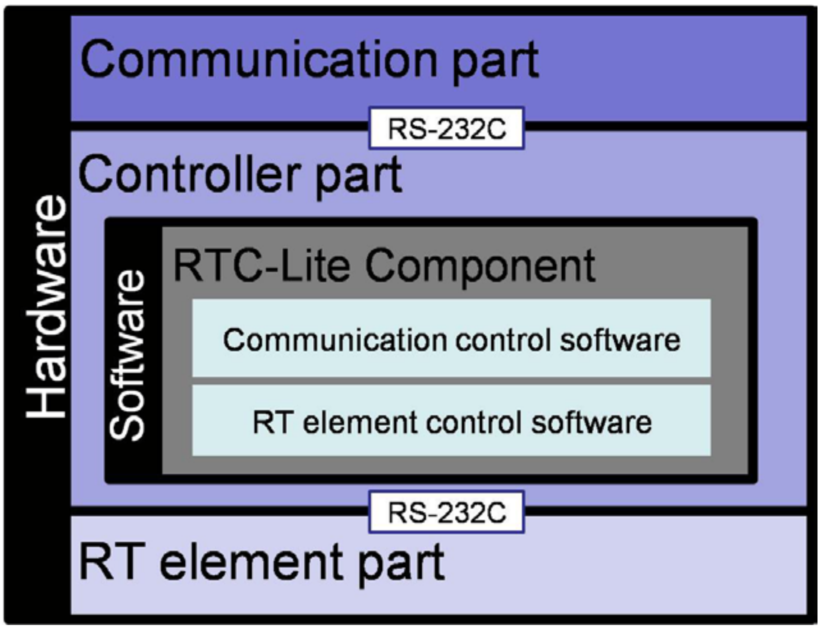

Fig. 3 Hardware and software architectures

IMPLEMENTATION

Implementation of the RT Module

We developed an RT module based on our design. Fig. 4 shows one of the RT modules. This module in Fig.4 provides switch function, being implemented DI component.

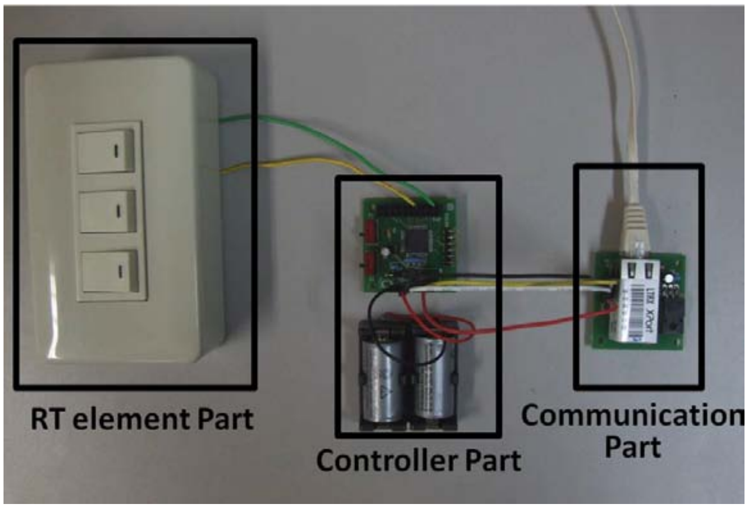

Fig. 4 RT module

Table 2 Specification of a RT module

Microprocessor	dsPIC30F3014
Communication part	LANTRONIX XPort
RT element part	Panasonic WV7013W & WN5311
Power	DC 3.3V

This module is made of three parts: the communication part, controller part, and RT element part. Each part is separated from the other, so that users can replace individual parts easily. The RTC-Lite component is implemented in the controller part. In Fig. 4, in order to control a switch as an RT element, a component provides a digital input function in the control part. Implemented software is used as the component. Thus, the user can replace implemented software easily because software is handled as a component. Moreover, by using RT-Middleware as a platform, the communication network is constructed easily. When components are connected on the GUI, as shown in Fig. 5, a network is built between the components. By using these modules, a simple home automation system is developed. Next, we show the replaceability and reusability of the RT module.

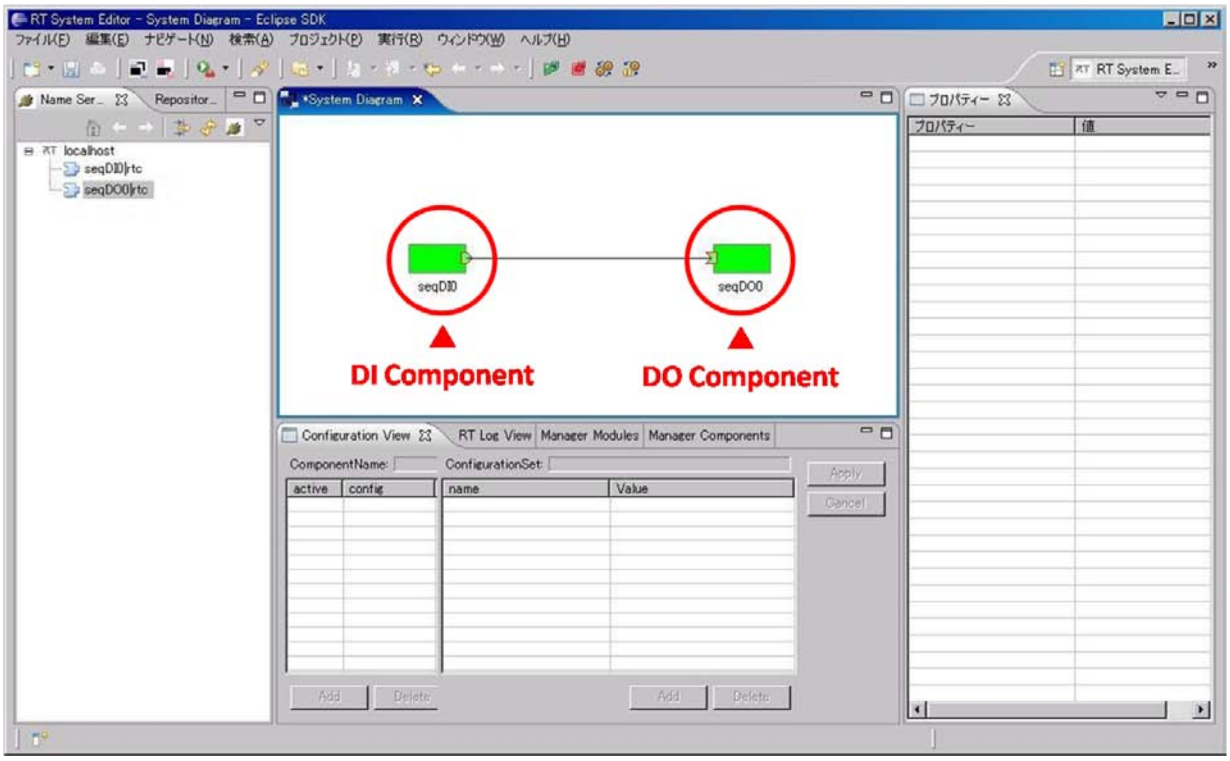

Fig. 5 GUI showing example of the RT-Middleware network

Replaceablity of the RT element part

There are many kinds of simple home appliances in a home network, such as a switch, light, and fan. The purpose of the RT module is described here. To control appliances, we extract their functions and categorize them into function classes. As noted in Table 3, simple home appliances as RT element parts are categorized into the following functions: Digital input, Analog input, Digital output, and Analog output.

Table 3 Extracted and categorized functions

[Functions Classes]	Digital input	Analog input	Digital output	Analog output
[Simple home appliances]	Switch	Sensors	Light	Electric fan

By considering the appliance's use and replaceability, four types of controllers are needed to control the home appliances and build a simple home automation system. Thus, to control these devices as RT elements, four components are constructed as shown in Table 3: digital input, digital output, analog input, analog output. Due to these categorized components, one component can be applied to different types of RT element parts and used effectively without changing resources of RT modules much. This means these RT modules have easy replaceability and flexibility. These RT modules can be used in various fields. By implementing these components, we can let various home and building appliances work in our proposed network easily.

Replaceability of the communication part

In this section, we show the replaceability of the communication part of the RT module. The RTC-Lite framework can accept various kinds of communication protocols between the RTC-Lite component and the proxy component. Therefore, this framework can accept Ethernet, Zigbee (XBee in Digi Int. Inc.) and so on between these two components. The communication protocol between them has flexibility. RT modules can be used in wired and wireless network. Users can select the communication protocol which they want. Moreover, the communication protocol can be replaced without changing resources drastically due to RTC-Lite framework. RT modules can be replaces only the part which they want to change. Thus, we can say RT modules have reusability too.

Extension of the communication protocol with RT element

There are many kinds of home and building appliances which are controlled by infrared ray(IR) remote controllers such as TV, air conditioner and so on. These appliances are controlled by IR-based signal data from IR remote controllers. RT modules based on above four functions can control these appliances. However, these appliances can be more easily controlled by using such remote controllers. RT modules are expected to have flexibility. In order to allow RT module control these appliances by using IR protocol, we extend the capability of RT modules. Communication way of RT module in Fig.6 is wireless by using XBee. Of course, if the communication device is replaced to device for RS232C communication, the communication way can be wired. Users can choose either way according to their needs. We implemented RT module which can control infrared-ray-controlled home appliances.

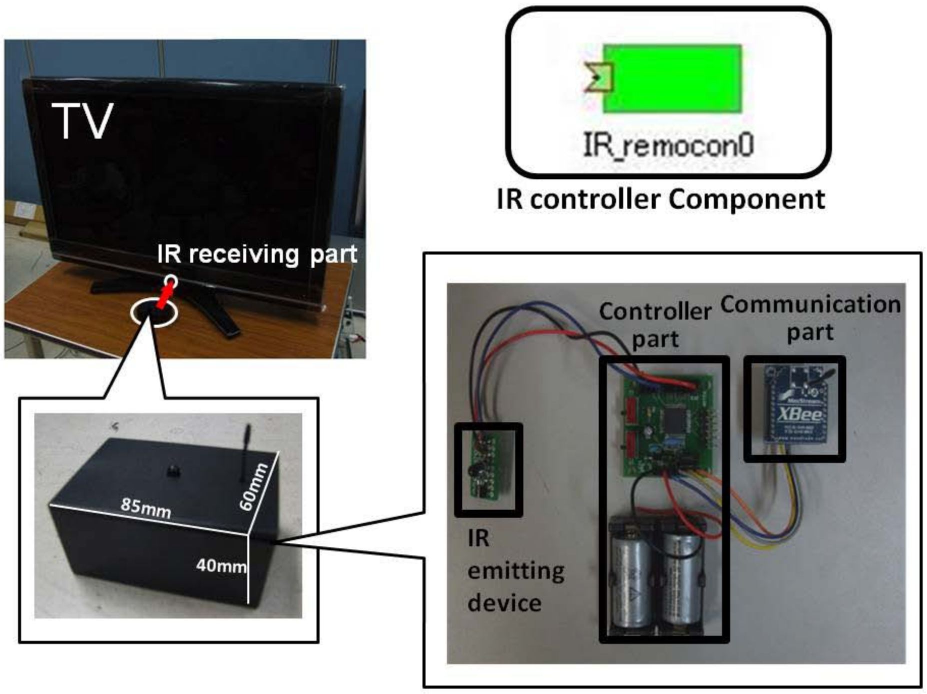

Fig. 6 IR controller module

In this case, this RT module can control the TV in [power ON/OFF], [switch the channel 1] - [switch the channel 12], [volume UP/DOWN], [channel UP/DOWN] and so on. This module is regarded as one RT component in RTM network because it is developed based on our proposed module design. This can emit IR signal based on data from another RT component in order to control IR-controlled appliances.

In order to let RT modules have TV function, RT modules can accept various communication ways such as DIO, AD/DA or IR communication. Due to this IR controller module, we can propose various ways of the communications protocols. Thus, users can choose the way they want to use depending on their situation.

CONCLUSION

In this paper, we propose the module design of an RT module in a ubiquitous robot environment. We discussed important aspects of the robot technology so that it can adapt to the various needs of users. Considering the role of RT modules, we analyzed the functions they should have. Then, hardware and software architectures were proposed to enable the RT module to have reusability and replaceability. Moreover, based on our RT module design, we implemented an RT module and showed the replaceability of each part. RT can accept various communication protocols between each part. From this implementation, it is demonstrated that the RT module can accommodate the various needs of users. By using such modules, users will be able to apply RT in many different fields

In future work, we intend to improve the usability of RT modules. The replaceability of RT modules should become higher. We want to reduce difficulties for users in the case of their constructing RT modules or replace each part of them. Moreover, RT modules will be able to accept more various communication protocols between each part. Thus, RT modules will be used in various fields and applied to other fields.

ACKNOWLEDGMENT

The authors acknowledge the New Energy and Industrial Technology Development Organization (NEDO) of Japan for funding this research.

REFERENCES

Alonso.J.M, Ribas.J, Coz.J.J, Calleja.A.J, Corominas.E.L, and Rico.M (2000) Deveopment of a distributed control scheme for fluorescent lighting based on LonWorks technology. IEEE Trans. Ind. Electron, 47(6), 1253-1262.

Ando.N, Ohara.K, Suzuki.T, and Ohba.K (2009) RTC-Lite: Lightweight RT-Component for Distributed Embedded Systems. SICE Journal of Control, Measurement, and System Integration (SICE JCMSI), Vol. 2(6), 328-333.

Ando.N, Suehiro.T, Kitagawa.K, Koutoku.T and Yoon.W.K (2005) RT-Middlewere:Distributed Component middlewere for RT(Robot Technology). Proceedings of 2005 IEEE/RSJ Int.Conf.Intelligent Robots and Systems, 3555-3560.

Bal,H.E., Kaashoek.M.F, and Tanenbaum.A.S (1992) Orca: a language for parallel programming of distributed systems. IEEE Transactions on Software Engineering 18(3), 190-205.

Fujita.M and Kageyama.K (1997) An Open Architecture for Robot Entertainment. Proceedings of the First International Conference on Autonomous Agents, 435-442.

Haartsen.J.C (2000) The Bluetooth radio system. IEEE Personal Communications February, pp. 28-36.

Hansen.S, Robertson.T, and Wilson.L.S (2007) ECHONET: designing for flexible use. Proceedings of the 10th European Conference on Computer Supported Co-operated Work.

Jackson.J (2007) Microsoft robotics studio: A technical introduction. IEEE Robotics and Automation Magazine, vol.14(4), 82-87.

Kinney.P, ZigBee Technology (2003) Wireless Control that Simply Works. White Paper.

Mizukawa.M, Matsuka.M, Koyama.H, Inukai.T, Noda.T, Tezuka.A, Noguchi.H, Otera.Y, and Shibaura.N (2002) ORiN: open robot interface for the network - the standard and unified network interface for industrial robot applications. Proceedings of the 41st SICE Annual Conference, vol.2, 925-928.

Narita.M, Shimamura.M, and Oya.M (2005) Reliable protocol for robot communication on Web services. Proceedings of the 2005 International Conference on Cyberworlds, 8pp. -220.

Ohara.K, Ohba.K, Kim.B.K, Tanikawa.T, Hirai.S, and Tanie.K (2005) Ubiquitous robotics with ubiquitous functions activate module. Proceedings of INSS2005, 97-102.

Utz.H, Sablatnog.S, Enderle.S, and Kraetzschmar.G (2002) Miro-middleware for mobile robot applications. IEEE Trans. Robot. Automat., vol.18, 493-497.