International Journal of Smart Home
Vol. 10, No. 1, (2016), pp. 41-48
http://dx.doi.org/10.14257/ijsh.2016.10.1.05

Design and Implementation of Middleware Based on ID and IP Address for Actuator Networks

Chen Nan[1], YunJung Lee[2], Faiza Tila[3] and Do Hyeun Kim[3*]

[1]KETI, 25, Saenari-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, 463-816 Korea
[2]Dept. of Computer Science and Statistics, Jeju National University
690-756 Jeju-si, South Korea
[2]Dept. of Computer Engineering, Jeju National University
690-756 Jeju-si, South Korea
[1]xuehu001@gmail.com, [3] faizakhan797@gmail.com,
[2]rheeyj@jejunu.ac.kr, [32]kimdh@jejunu.ac.kr

Abstract

Recently, the intelligent services demands interactions with the sensor networks, actuator networks, and context aware applications. Actuator network supports an infrastructure to have an immediate access to information for controlling the physical world and its objects using sensor networks and context aware applications. Actuator network middleware supports the command message transfer for controlling the behavior of the environment or physical systems. In this paper, we design and implement the actuator network middleware based on ID(IDentification) and IP(Internet Protocol) address for control the environment using multiple actuator devices. Also, this middleware assigns ID for identifying each actuator nodes, and uses an IP address for transferring control messages between application and actuator networks. And, the presented middleware provides to connect between actuator ID and IP address using the mapping table. We design the middleware using sequence diagram and state diagram, and implement it based on .Net framework. The middleware supports to transfer command messages between application and actuator networks.

***Keywords**: Actuator networks, ID and IP address mapping, Actuator network middleware*

1. Introduction

Recently, there is a growing IoT(Internet of Things) using sensors for environmental monitoring and actuators for reacting to environmental changes. And, Actuators have been used various elements of automated control systems in industrial plant. Actuators were chosen to perform a specific task, inconsistent actuator performance, or the nature of the medium to be controlled [1].

Actuator network is in charge of providing knowledge from an environment to a non-expert user. Actuator network can be used in different environments, so it needs to be able to address many heterogeneous devices [2].

Actuator networks are a distributed system of actuator nodes that are interconnected. Actuators perform actions to change the behavior of the environment or physical systems. In many situations, actuator nodes typically have stronger computation and communication powers and more energy budget that allows longer battery life [3, 4].

There is the convergence of control of resources with communication and computation could be the next frontier in information technology [5, 6]. Middleware is software infrastructure that has been used to successfully integrate and manage software for complex distributed systems. Most middleware addresses a particular domain such as web services, and defines simple and uniform architectures for developing applications in the domain[7, 8].

Also, we address the issue of middleware for actuators network of the control systems which feature the convergence of control with communication and computation on Internet.

There are different actuator network middleware approaches for IoT(Internet of Things) service. These approaches help in offering important functions for different applications such as efficient control. The role of actuator network middleware is certainly to control devices and services. The action of the system can be done manually or automatically via a user control interface[9].

Actuator networks middleware supports the complexity and heterogeneity of the underlying hardware and network platforms, and application for transferring command and response message. Actuator network middleware is providing the controlling-based pervasive computing applications using the actuator nodes. Also, actuator network middleware supports the message transmissions for controlling the behavior of the environment or physical systems [1-4].

The Sentire is a representative middleware for sensor and actuator networks. The Sentire provides a framework to facilitate building extensible application middleware which includes logic to support resource and application management, sensor data processing *etc*. The Sentire supports the process of building such a middleware, rather than addressing a specific instance of a middleware in sensor and actuator networks. The Sentire consists of Interface manager, Data manager, Resource manager, Sensor manager, and Actuator manager. Interface manager provides initial layer of query/instruction filtering via developer-defined admit/reject policies. Resource manager supports other managers in their policy-based decisions. Sensor and actuator manager Influence both the quality of sensed information and adjustments to the environment. Data manager embodies develop-defined data processing routines.

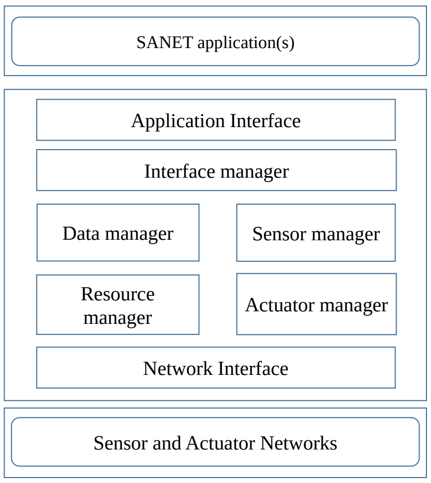


Figure 1. Sentire: A Framework for Building Middleware for Sensor and Actuator Networks

This paper designs the actuator network middleware mapping between actuator ID(IDentification) and IP(Internet Protocol) address for controlling the multiple actuators using sequence diagram and state diagram. Also we implement the actuator network middleware to connect between actuator ID and IP address using the mapping table based on .Net framework. And, we assigns ID to identify each nodes in actuator networks, and uses an IP address for transferring control messages between application and actuator networks. The middleware interconnects between application and actuator networks using the ID and IP address mapping table.

The rest of the paper is as organized follows. A middleware design for actuator networks are introduced in Section II. Section III illustrates a middleware implementation of actuator networks, and Section IV concludes.

2. A Middleware Design for Actuator Networks

A middleware of actuator networks is common software infrastructure that has been used to connect between applications and networks. The middleware addresses a part of control system such as common platform, and supports to develop intelligent applications.

The designed middleware is a part of a larger IoT system, understanding the actuator network middleware would describe various functionality. Figure 1 shows the actuator network middleware the detailed configuration. Its basic function is to transfer control messages between the application and the actuator networks. First function is the two way communication using TCP sockets. The figure shows how this communication is managed and carried out with the help of the modules inside the actuator network middleware. Second function have two management and processing for transferring messages between the application and the actuator networks.

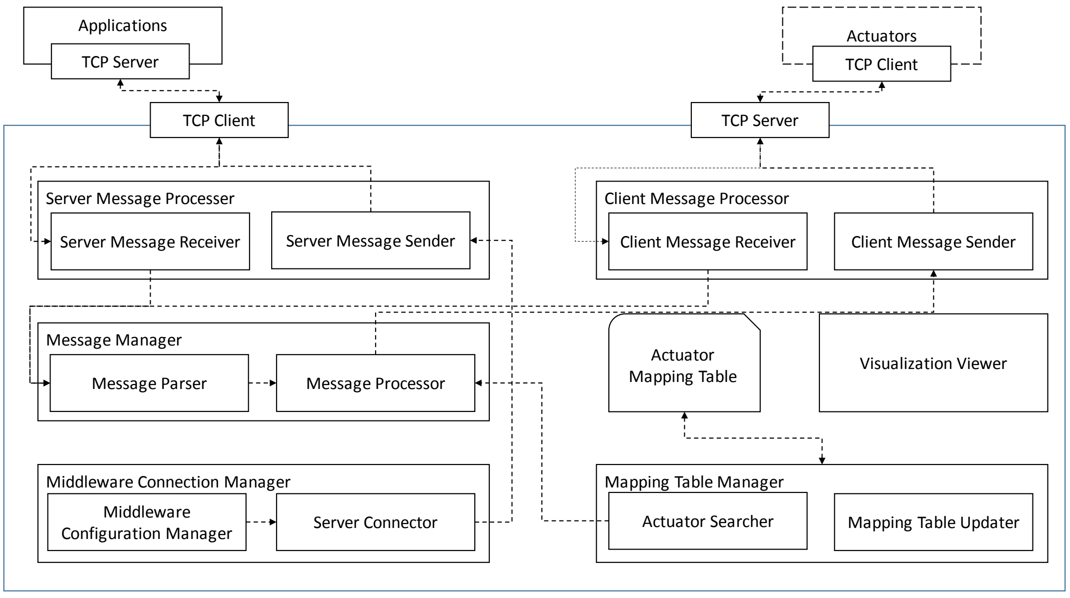


Figure 1. Proposed Middleware Configuration for Actuator Network

The middleware of actuator network has a server message processor controls the sending and retrieving of messages. It consists of a server message receiver, and a server message sender, the former controls messages received by the application. But the later retrieves a message from the middleware for actuator networks and forwards it to the middleware for actuator networks. Message manager has the message parser and message processor, the former parses the messages received from the application and actuator. The processing type includes connection request, control request, mapping table renewal request, control response and connect response.

The middleware connection manager also has two other modules inside it the middleware configuration manager and the server connector. The middleware configuration manager sets up the environment by providing ID and IP information to the middleware for actuator networks, and the server connector verifies the middleware access right and decides whether actuator server can or cannot connect. The client message processor consists of the client message receiver and the client message sender. And the client message receiver receives the actuator message, the client message sender sends message from the middleware for actuator networks to the actuator.

The purpose of addressing is mainly to identify the device and to indicate its virtual location within the domain. We use ID(Identification) for identifying each nodes, and apply an IP address for transferring control messages. The middleware supports IP address each nodes so that other nodes know where to reach it. The IP address is usually allocated and given automatically in the beginning when the device is plugged into the network. Also the middleware assigns an ID each nodes for identifying manually.

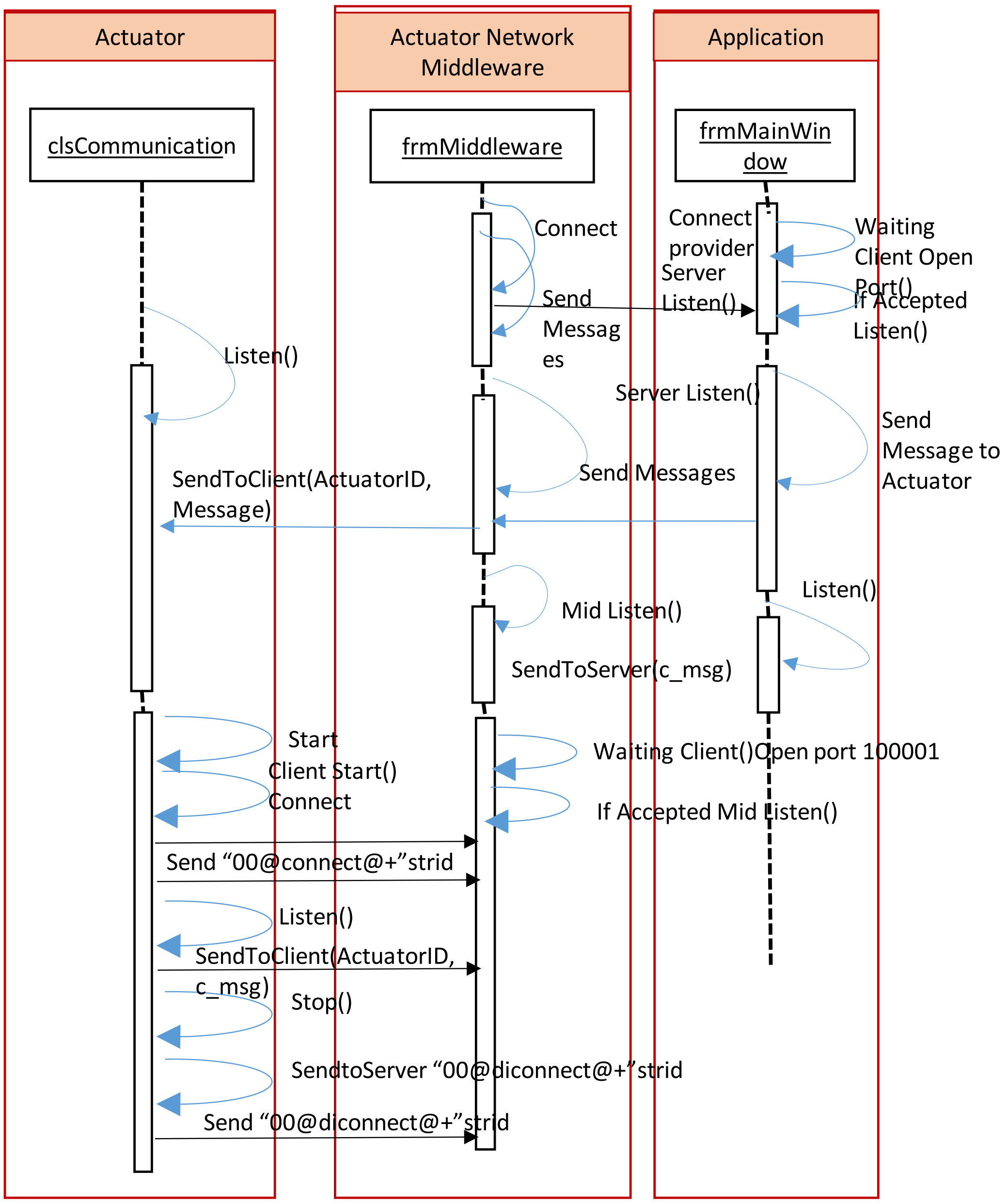


Figure 4. Sequence Diagram of the Middleware for Actuator Networks

The mapping table manager consists of the actuator searcher and the mapping table updater. The actuator searcher provides functionality for finding an actuator ID and an actuator IP address, the mapping table updater modifies a mapping table between actuator ID and an actuator IP address with mapping information which is saved in middleware memory. It is used for storing the actuator IP address and actuator ID information. Addressing. The server message receiver receives the control message sent by application and client, and forward it to the message parser. The parsed message is then forwarded to the message processor. The message processor searches the actuator connection information using the message contents and transfers the message to the intended actuator through client message sender. Also, the message processor forwards the message to the server message sender, and it sends the connection request message created by middleware. And, the message processor transfers the control response message with actuator state information. Then, the client message receiver forwards it to application and client.

Figure 4 shows the sequence diagram to describe each steps for the middleware of actuator networks. First step, the actuator connects to the application of actuator networks using an open port. After established connection the middleware for actuator network, we scan receive control messages from the application of actuator networks. Next step, the middleware communicates with the actuators thus connecting between the application and actuators in actuator networks. The actuator can send its status messages to the application and receive the control messages from the application through the middleware.

Figure 5 shows the state chart diagram for the middleware of actuator networks. The application waits to be connected to the actuator through the middleware for actuator networks. After connecting to the actuator, it can send the control message and receives the response message from the actuators.

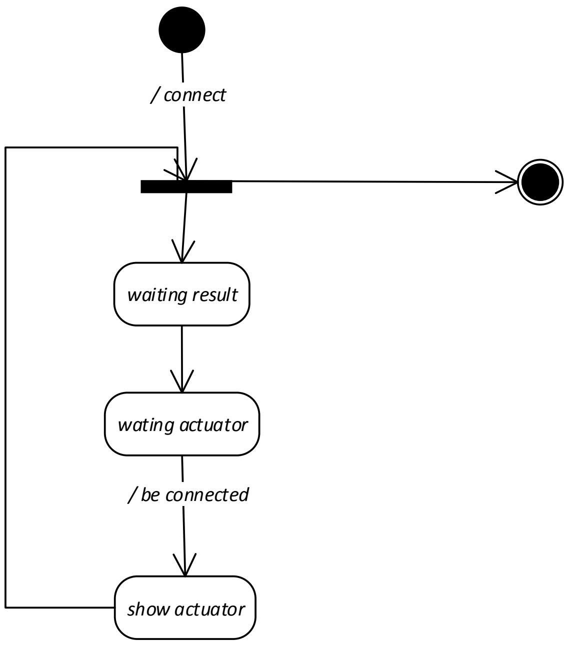


Figure 5. State Diagram for the Middleware of Actuator Networks

Figure 6 shows the class diagram for the middleware of actuator networks. It includes four classes, frm Middleware, Mapping Table Item, Socket, and IPEndPoint. The frmMiddleware is the main class inherited from the window class. It provides interface for visualizing middleware connection, operational state and message communication. The MappingTableItem class saves a list of the actuator IP information that connects to the middleware. The main window class uses the IPEndPoint class, socket class and the MappingTableItem class. The IPEndPoint class saves the network address of the endpoint that connects to the middleware or the application of actuator networks.

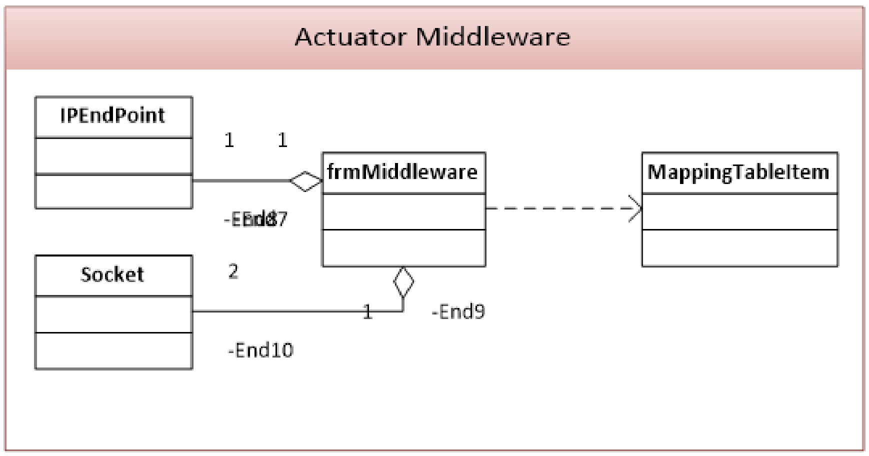


Figure 6. Class Diagram of the Middleware for Actuator Networks

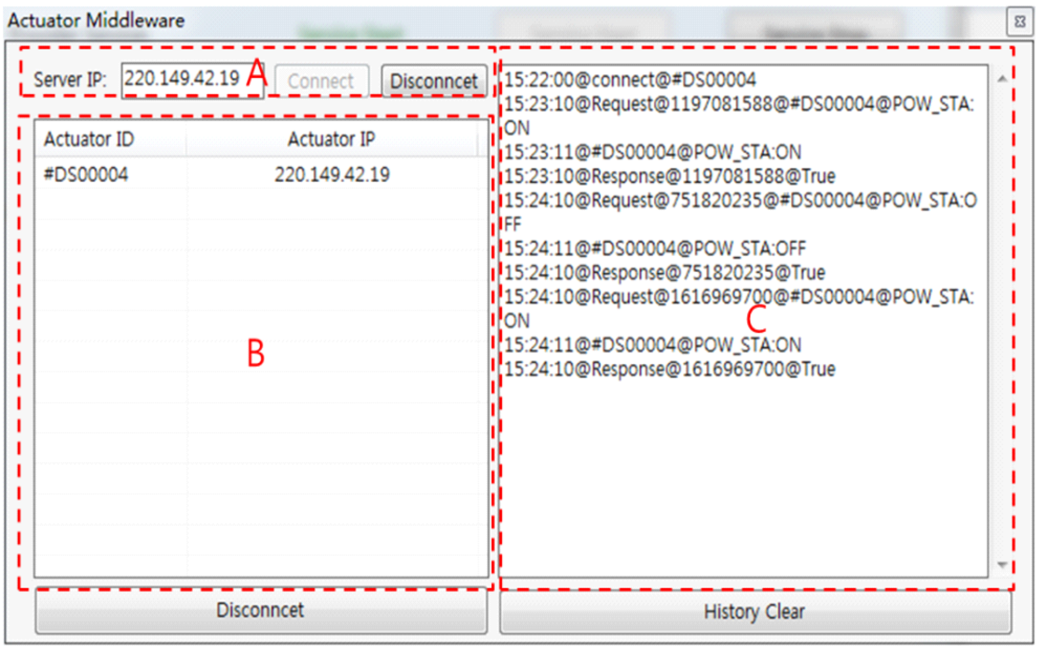
3. A Middleware Implementation for Actuator Networks

The middleware implementation environment is described in Table 1 for actuator networks. We use Microsoft Windows 7(X64) for the operating system. Developing is performed in visual studio .net framework version 3.5 or 4. We use Microsoft SQL Server 2008 R2 for database management. Hardware computer has Intel core i 3 with 3.30 Ghz for CPU, 4 GB ram and NVIDIA Geforce GT 440 graphics card.

Table 1. Implementation Environment of the Middleware for Actuator Networks

Operating System	Microsoft Windows 7(X64)
Development Environment	.Net Framework 3.5, 4.0
Development Tool	Visual Studio .Net 2010
Programming Language	C#, XMAL, XML
DBMS	Microsoft SQL Server 2008 R2
Hardware	CPU: Intel® Core™ i3-2125 @ 3.30GHz RAM: 4GB Graphics: NVIDIA GeForce GT 440

Figure 7 illustrates implementation result for the middleware of actuator networks. In Figure 7, area A provides to set the server IP address for application in the middleware of actuator networks. Server IP assigns to connect between the middleware for actuator networks and application. Area B shows the connection state of the connected actuators and middleware in Figure 7. Actuator ID represents actuator's unique identifier and actuator IP is the current actuator's IP address to connect between the actuator and the middleware. An actuator node has an IP address, and one or more actuator ID. Area C shows the message received from the application and the actuator in the middleware for actuator networks.



Server IP: 220.149.42.19

Actuator ID	Actuator IP
#DS00004	220.149.42.19

```
15:22:00@connect@#DS00004
15:23:10@Request@1197081588@#DS00004@POW_STA:
ON
15:23:11@#DS00004@POW_STA:ON
15:23:10@Response@1197081588@True
15:24:10@Request@751820235@#DS00004@POW_STA:O
FF
15:24:11@#DS00004@POW_STA:OFF
15:24:10@Response@751820235@True
15:24:10@Request@1616969700@#DS00004@POW_STA:
ON
15:24:11@#DS00004@POW_STA:ON
15:24:10@Response@1616969700@True
```

Figure 7. Implementation Result of the Middleware for Actuator Networks

4. Conclusion

In this paper, we present the actuator network middleware based on ID and IP address for controlling the multiple actuators. Also, we design this middleware assigned ID for identifying each actuator nodes, and used an IP address for transferring control messages between application and actuator networks. Also we implement the actuator network middleware to connect between actuator ID and IP address using the mapping table based on .Net framework. The middleware provides to transfer control messages between application and actuator networks.

Acknowledgments

This work was partly supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) (No.10043907, Development of high performance IoT device and Open Platform with Intelligent Software) and This research was supported by the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2015-H8501-15-1017) supervised by the IITP(Institute for Information & communications Technology Promotion) Corresponding author; DoHyeun Kim (e-mail: kimdh@jejunu.ac.kr).

References

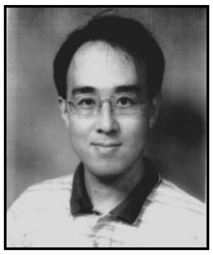
[1] B. Dong, "Design and Implementation of Middleware for Wireless Sensor Networks", Applied Mechanics and Materials, vol. 530, **(2014)**.
[2] S. Zarghami, “Middleware for Internet of Things”, University of Twente, **(2013)**.
[3] J. Schneider, A. Klein, C. Mannweiler and H. D. Schotten,”An efficient architecture for the integration of sensor and actuator networks into the future internet”, Adv. Radio Sci., vol. 9, (2011).
[4] G. Baliga, S. Graham and P. R. Kumar, “Middleware and Abstractions in the Convergence of Control with Communication and Computation”, Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, (2005); Seville, Spain.
[5] T. Melodia, D. Pompili, V. C. Gungor and I. F. Akyildiz, “Communication and Coordination in Wireless Sensor and Actor Networks”, IEEE Transactions on Mobile Computing, vol. 6, no. 10, (2007).

[6] J. W. Branch, J. S. Davis II, D. M. Sow and C. Bisdikian,"Sentire: A Framework for Building Middleware for Sensor and Actuator Networks", Proceedings of the 3rd Int'l Conf. on Pervasive Computing and Communications Workshops, (2005).
[7] CORBA Success Stories, OMG Inc, http://www.corba.org/success.htm.
[8] H. Abangar, P. Barnaghi, K. Moessner, A. Nnaemego, K. Balaskandan and R. Tafazolli, "A service oriented middleware architecture for wireless sensor networks", Proceedings of future network and mobile summit conference, (2010).

Authors

Nan Chen, he received the Master degree in mobile computing from the Jeju National University, Jeju-si, Republic of Korea. Currently, he is a Researcher at Embedded Software Combination Research Center of the KETI(Korea Electronics Technology Institute). His research interests have IoT open platform and IoT Interworking.

Yunjung Lee, she did her Ph.D. in the Department of Computer Science from Korea University, Seoul, South Korea in 2002. Professor, Dept. of Computer Science and Statistics of Jeju National University, South Korea



Do-Hyeun Kim, he received the B.S., M.S. and P.D degrees in Electronics Engineering from Kyungpook National University, Taegu, Korea, in 1988 and 1990, 2000 respectively. He joined the Agency of Defense Development (ADD), Korea, in 1990. Since 2004, he is currently a professor at the Department of Computer Engineering at Jeju National University, Korea. His research interests include sensor web, and context prediction.

Faiza Khan, she is studying in the Department of Computer Engineering of Jeju National University, South Korea. Her research interests included an IoT and Ontology System.