Service-Oriented Process Modelling for Device Control in Future Networks

Muhammad Khan and DoHyeun Kim
Computer Engineering Department, Jeju National University, South Korea

Abstract: *The recent advancements in the fields of electronics, information and communication technologies have paved a pathway towards a world-wide future network of connected smart devices. Researchers from industry and academia are taking even more interests in the realization of such an infrastructure where a seamless interaction of sensing and actuating devices will take place in order to provide valuable services to the human kind and other systems. So far the major focus of research is towards the connectivity, management and control of sensing devices and no major attention has been given to the control of actuating devices in such an environment. This paper presents a generic process model for actuating device control service in future networks. A prototype implementation of the proposed model based on the presented platform has been described along with the performance analysis of the proposed model.*

Keywords: *Process modelling, future networks, device profile, device control.*

Received October 16, 2014; accepted April 26, 2015

1. Introduction

In recent years, the number of devices (things) around us has exponentially increased. According to the European Commission report, the number of connected devices in the world will reach an estimated value of 50 billion by 2020 [9]. The era of these computing devices is outside the dominion of traditional desktop computers. For such an era, scientist have already introduced a conception of Internet of Things (IoT), where the things around us will be part of a huge network producing information to be used by humans or other related objects and vice versa. According to Gubbi et al., the vision of IoT is to connect anything, anytime and anywhere [3]. In such a scenario, all the daily life things around us such as home appliances heaters, televisions, cameras, ovens, refrigerators and even furniture [8] will all be connected to a huge network along with an array of sensors to collect the contextual and environmental data around these things and based on that data each appliance will be controlled.

The connectivity of industrial actuators and home appliances to internet in order to provide a remote access and control interface for these devices has been a long perceived concept. Such a scenario was always envisioned which would provide a communication and control interface for the daily used home appliances from anywhere in the world. World-Wide-Web (WWW) provided the initial realization of such visions. Effort were performed to connect and control multimedia devices such as cameras [2], telemedicine applications [5], Heating, Ventilation and Air Conditioning (HVAC) systems [6] etc. with the current developments in the electronic industry, daily life devices such as televisions, refrigerators, lighting equipment and microwave ovens etc. are being made smart and such devices are able to meet the varying needs of the users based on time and user activities [11, 13]. Some of the early efforts in this regards include the internet based control of manufacturing equipment [7] and personal robots [4, 12] among so many others. The main issue with these early attempts was that most of these implementations were based on the control of specific devices. Another issue was that most of the technologies used for such implementations were based on distributed object technologies such as Object Management Group (OMG) Common Object Request Broker Architecture (CORBA) [14], java Remote Method Invocation (RMI) [1] and Microsoft's Distributed Component Object Model (DCOM) [10]. These distributed objects based technologies were too tightly coupled which presented issues such as inflexible communication and implementations which could not be generalized.

The recent advances in the electronics and communication technologies have revolutionized human's lives. Smart devices in the form of sensors and actuators can be found all around us. The numbers of these devices are increasing with an exponential growth [9]. Miniature devices called the Micro-Electro-Mechanical Systems (MEMS) along with the wireless communications technologies over smaller distances have made possible the interconnection of devices into Wireless Sensor Networks (WSN). Such networks are being utilized widely in monitoring applications for environment, traffic and infrastructure etc., [15].

The concepts of ubiquitous computing by Mark Weiser [16] defines a smart environment as "the

physical world that is richly and invisibly interwoven with sensors, actuators, displays and computational elements, embedded seamlessly in the everyday objects of our lives, and connected through a continuous network". The more recent concepts of Cloud computing [17] and Internet of things [3] have the promise of reliable service provision based on next generation data centres, where the processing of data is hidden from users and provides the user with easy and understandable web based visualization.

Controlling devices from a remote location becomes very important in the above mentioned scenarios, where the devices are expected to provide intelligent services to the users. The major issue in solving this problem is mainly offered by the heterogeneous nature of the hardware of the devices that are needed to be connected and controlled. In past many efforts have been done to provide remote connectivity to devices and to provide remote control access via web and internet technologies but the solutions provided were too tightly coupled with the hardware and the implementation technologies. The existing solutions are mostly applications specific and so far no general model has been presented for controlling actuating devices in the future network scenarios. This paper presents a generic service-oriented process model for control service in future network paradigm. The process modelling notations have been used to model different processes that enable the control of devices and a prototype has been developed to test the performance of the proposed model.

The rest of the paper is organized as follows. Section 2 presents the conceptual model of the platform on which the control model has been based. The section describes the platform in terms of the layers and major responsibilities of each layer. Section 3 provides a detailed description of the proposed process model for device control service in IoT scenario. The section illustrates each process in step by step detail along with graphical representation of the processes in the form of business process model diagrams. In order to test the proposed model, a simple prototype of the proposed model was developed. Section 4 provides a brief description of the implementation tools and technologies for the development of the prototype.

Section 5 presents the experimental setup, the network deployment, hardware and software platform specifications and the results of the performance analysis experiment. Finally, Section 6 concludes the paper.

2. Control Platform

The proposed model for device control is based on a generic platform shown in the Figure 1. The physical devices indicate the hardware which is intended to be controlled using the system. The middleware is responsible for the connection and communication with the heterogeneous hardware devices and implements the necessary drivers and communication technologies. Service provider represents the sink node which maintains a database to store the information defining the connected devices and also the data produced by the devices. Normally, control is associated with actuating devices so the data produced by such devices is normally the operational states of the devices. The proposed scheme considers more than one service providers in order to keep the system simple and flexible.

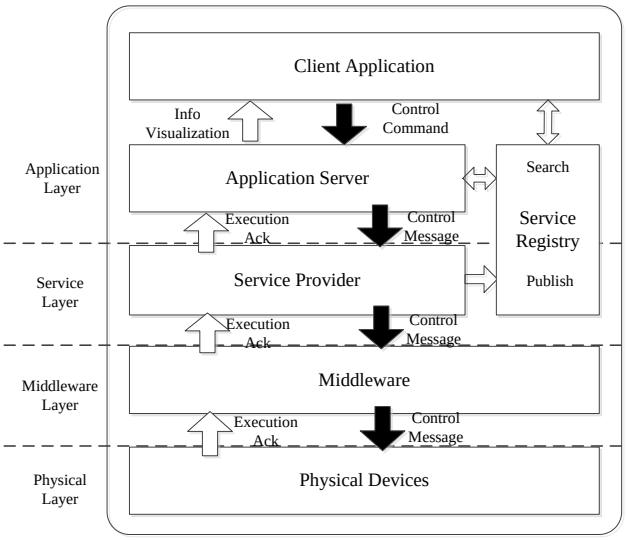

Figure 1. IoT paltform for device control.

The state data collected from the physical devices by the middleware is forwarded to the service providers. This is made possible by the control service exposed by the service provider. The service provider exposes provider service to enable the clients or the application server to access the data and information stored at the service provider's database. The provider services are published to the service registry in order to be discovered and utilized by application server or the clients. The service registry is a module with Search and Publish interfaces and maintains a database to store the service information. This information is utilized by the service searching entities (clients/application server) to utilize the published services.

Application server is the processing and binding point for data and information offered by one or more service providers. It also presents a connection point for the clients to get integrated services based on the provider services offered by individual service providers. The binding of services (integration) is performed by getting the service information for multiple service providers from the service registry using the search interface. The data offered by each service provider is then combined according to a predefined schema. In order to make the clients utilize the combined data, the applications server exposes (publishes) application server provider service. These

services are also published to the service registry from where the clients may search and consume a service.

3. Proposed Model

This section presents the illustration of the proposed model for device control process in future networks. The model has been described in the form of individual processes at each layer. Each process is step by step illustrated in the form for business process model diagrams with various tasks at different layers. There are four major processes and each one is explained as follows.

3.1. Setup Process

The setup process configures the various components of the system in order to initialize the system.
The setup process is mandatory so that all the components (objects, connections, services etc.) are initialized for the whole system to work properly. The setup process involves the application layer and service layer. The steps describe the process in details and the steps are illustrated in Figure 2. Here devices mean the actuating devices unless otherwise mentioned.

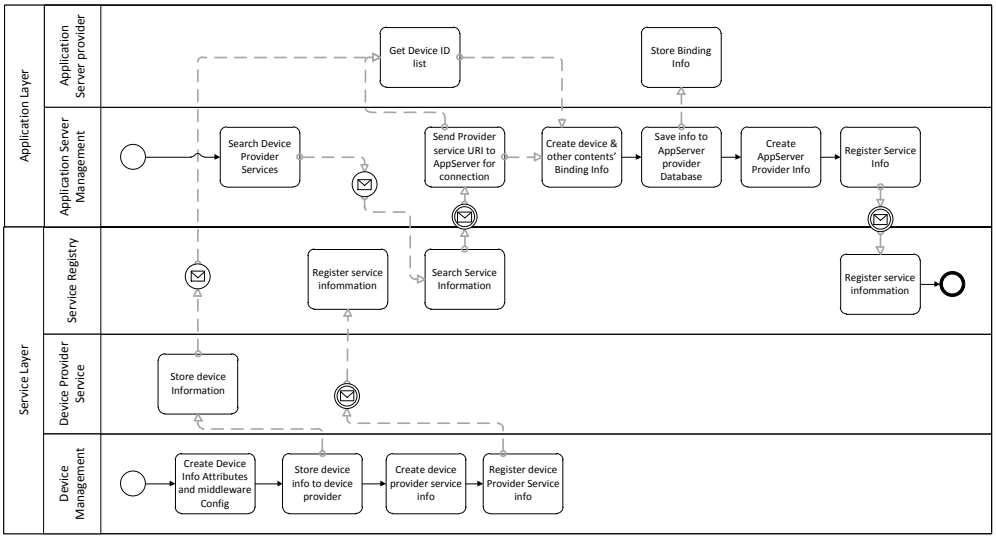

Figure 2. Process model for setup of device control environment.

Device management module of the service layer creates Actuator information (ID, Type etc.,) and saves the info to the service provider's database at the service layer. This information is stored in the provider's data repository. The device management module also creates the service provider's service information and registers it to the service registry module at the service layer. This enables the provider service to be searchable by the clients which have a link to the service registry module.

As the device control platform is proposed for an indoor environment scenario, the location information for the devices is provided by an information service provider called as the GIS service provider. The GIS Management module at the Service layer creates map contents based on the map image (building area, floors and rooms) and saves the info to the GIS Provider's database at the Service layer. The stored GIS contents are made available for the clients by exposing a GIS provider service. GIS Management module creates the GIS Provider's service information as an xml document and registers it to the service registry module at the Service layer so that other modules may search and access the service.

In order to bind the information from different providers, the Application Server Management module searches the intended actuator provider service information in the service registry module and sends it to application server provider to get actuator information from the respective service provider. the application server management module also acquires the intended GIS Provider service from the Service Registry and binds the service to a map based on the administrator's choice. The GIS service must be associated with a map because the GIS service provider contains multiple maps and map contents.

Thus, each utilization of the service must associate a map object with the service in order to use the GIS contents related to that map. The application server management module then creates binding information for map coordinates and actuator locations based on the administrator choice. The application server Management module finally saves the binding information to the application server provider database, Creates application server provider service information and registers it to the Service Registry Module.

3.2. Search Process

Service registry module exposes the search service interface in order to enable the clients to search services information in its service information repository. The search service interface helps the client

to find the required service information using keyword search and returns the service information through which client may consume and utilize the respective service. The search process involves the interaction between application layer and the service layer. The steps describe the process in details and the process is illustrated in Figure 3.

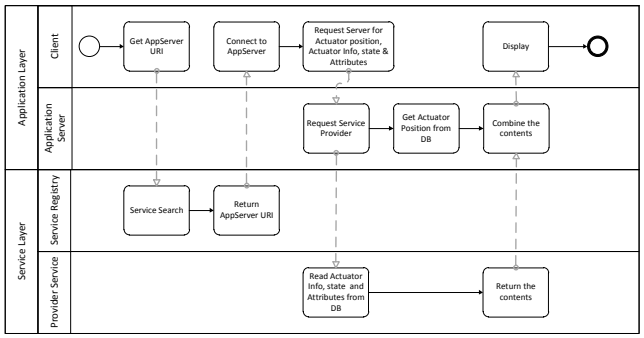

Figure 3. Process model for search process.

The client utilizes the search service exposed by the registry service and using the search interface and search keyword, the client queries the service repository for the required provider or application server services. It is important to note that client can search and consume services exposed by service providers for using the raw data from the underlying hardware or may search and consume application server's services to get integrated data from multiple service providers. In the current scenario, the control is being provided by the application server so the client utilizes the integrated service for location based control of devices. Once the connection with the application server is established, the client requests the Server for actuator information and location data. Application server requests the application server provider to get the Actuator Information and GIS contents from the actuator provider database and GIS contents repository. Application Server then integrates the information from the two service providers and sends it to client. The client then receives the information and displays it to the user.

3.3. Service Registration Process

Registration service is the interface provided by Service Registry module for service providers to make their service searchable to the clients. The registration service is utilized by the service providers to register their service information with the Service Registry. Using the registration service, the provider's service information (xml) is stored at the service repository at the Service registry from where the clients can select and utilize it using the search service. The process is illustrated in Figure 4.

The service provider's management module creates the service information in the form of an xml file. This information is then sent to the service registry module using the registration service. The registration service has the necessary implementation to send and store the provider service information at the information repository at the service registry module. The clients may then search and utilize the registered services using the search service interface.

3.4. Device Control Process

The actuator control process involves physical layer, service layer and application layer. The process has been illustrated in the Figure 5 with the division of tasks at each layer. In order to control an actuator, the actuator and the associated middleware must first be configured.

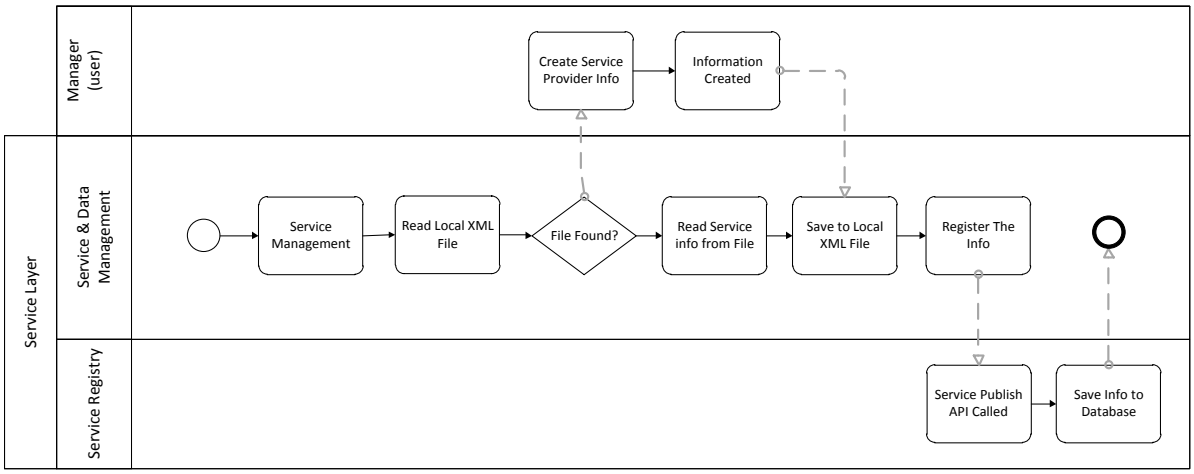

Figure 4. Service process model for device control service.

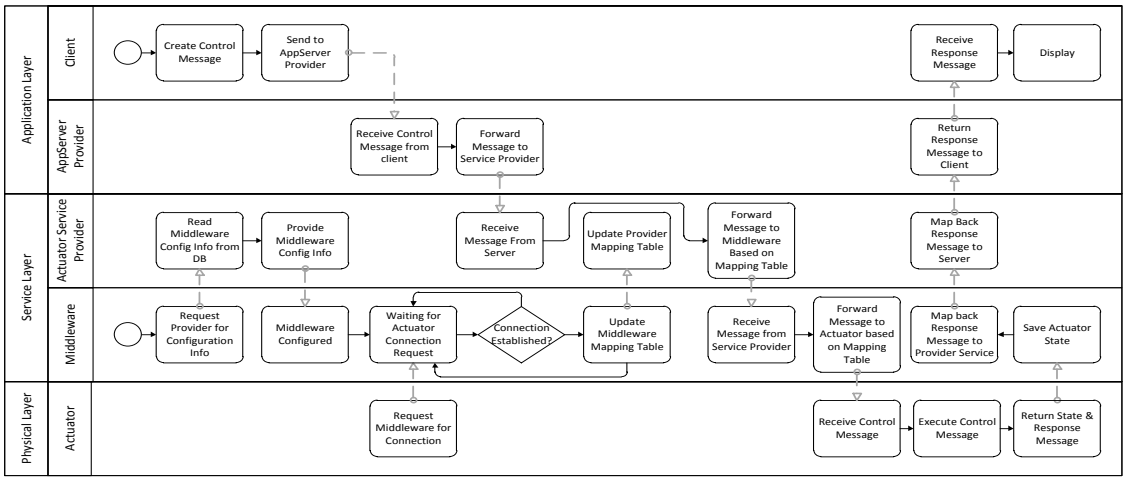

Figure 5. Process model for service registration process

Thus the first step in the process is the configuration of actuator middleware. the middleware requests the configuration information from the actuator provider and based on that information it creates a configuration for the actuator network to be created. The middleware then waits for a connection from the actuators at the physical layer. Once actuators successfully connect to the middleware, the middleware mapping table (actuator, middleware mapping) is updated and the same information is sent to actuator provider to update the mapping table (middleware, provider mapping) at the provider.

When the client creates a control message for changing the operational state of an actuator, it sends the message to the application server provider. The application server provider receives the message and forwards it to the appropriate actuator provider at the service layer based on the actuator information. Based on the provider's mapping table, the provider forwards the control message to the appropriate middleware.

Middleware receives the control message and based on middleware mapping table, it forwards the message to the appropriate actuator.

Actuator receives the message and executes it. After execution of the control command, it sends a response message with its current state information backwards to the middleware. Middleware receives the response message, saves the actuator state and based on the middleware mapping table sends the response message to the actuator provider. Actuator provider uses its mapping table to send back the response message to the application server provider. Finally, the application server returns the response message to the client, which displays the new state of the actuator.

4. Implementation

This section provides a brief description of the implementation technologies used for the development of the system. The services in the proposed scheme have been implemented by using Windows Communication Foundation (WCF) technology in the .Net framework. The data and service repositories are implemented using SQL Server 2010. The client application for the utilization of the services and data is implemented using Microsoft Silverlight 4.0.

5. Performance Analysis

5.1. Experimental Setup

Figure 6 shows the experimental setup for testing the performance of the proposed process model for device control in IoT. The figure shows the network setup for various modules of the system. For this experiment, an actuator emulator has been used to emulate the functionality of actuating devices such as fan, light, AC and boiler/heater. A specialized control Requester module has been developed for the sake of experiment.

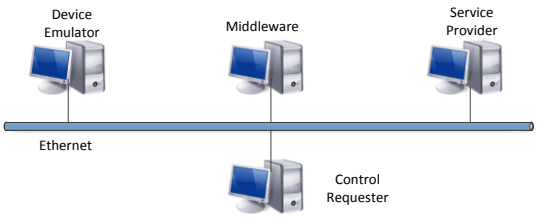

Figure 6. Network composition for experiment.

The control requester can be configured to simultaneously request the control of single or multiple devices attached to the system. For this experiment the actuator emulator uses TCP connection to connect to the middleware. The middleware in turn uses the control service exposed by the service provider. The Control requester module utilizes the provider service exposed by the service provider for evaluation at the service layer while for the application layer testing; the requester utilizes the provider service exposed by the application server.

Table 1 summarizes the experimental setup in terms of the hardware and software used. The table presents the hardware in terms of processor specification, memory and graphic adaptor used in each server. The

software includes the development environment and the backend database used at each station.

A control requester module was specially designed for the sake of this experiment which utilizes the control service exposed by the control service provider.

Table 1. Hardware and software sepcifications

	Server 1 (mcl-12)	Server 2 (mcl-16)	Client 1 (mcl-11)
Operating System	Microsoft Windows 7(X64)	Microsoft Windows 7(X64)	Microsoft Windows 7(X64)
Development Environment	.Net Framework 3.5, 4.0	.Net Framework 3.5, 4.0	.Net Framework 3.5, 4.0
DBMS	Microsoft SQL Server 2012 express	Microsoft SQL Server 2012 express	-
Hardware	CPU: Intel® Core™ W3503 @ 2.39GHz RAM: 4GB Graphics: NVIDIA GeForce FX 580	CPU: AMD® Athlon™64 X2 Dual Core Process 6000+ @ 3.1GHz RAM: 5GB Graphics: Standard VGA Graphics Adapter	CPU: Intel® Core™ i5-3570 @ 3.40GHz RAM: 4GB Graphics: NVIDIA GeForce GT 440

The requester is designed in such a way that it can select anyone of the available devices and then may issue a control command for that specific device using the control service. The command message is then processed at the provider and forwarded to the intended middleware. The middleware forwards the control message to the specific device and takes back the response. This response message is routed back to the control requester. The time between the issuance of a command message and the reception of the response message by the control requester is measured as the delay.

5.2. Results

Performance analysis has been performed based on the level of device control service provision and the measurements taken at two different levels e.g. the Service Provider level and the Application Server level have been compared. Figure 7 shows the graph of minimum, maximum and average delay for performance measurement at the service layer. As mentioned in the previous section, at the service layer resides the service providers. Clients or other applications may directly consume the control service exposed by the provider. The graph is a depiction of the control delay in terms of minimum, maximum and average delay in three different scenarios. In the first scenario, control requester is configured to control 1 device using the control service. The time for control requester to issue a control message and getting back the response message is recorded for 20 iterations. In the second scenario, control requester is configured to simultaneous control 5 devices using the control service and in the third scenario, 25 devices are simultaneously issued control commands by the control requester module.

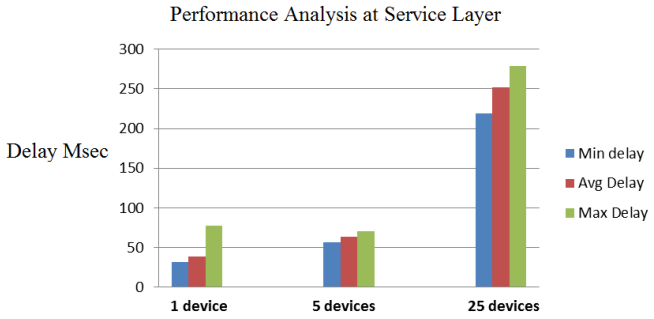

Figure 7. Control service performance at service layer.

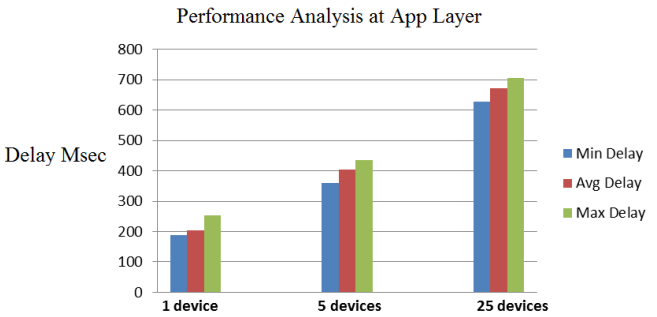

Figure 8. Control service performance at application layer.

6. Conclusions

In this paper we presented a general model for device control service in actuator web as part of the IoT system. The proposed model is based on service-oriented platform, which has been illustrated in the paper. The processes for system configuration, service registration, service search and device control service have been illustrated step by step using business process model diagrams. The paper also presents a prototype implementation of the control service based on the service-oriented platform. The prototype has been evaluated for performance in terms of control service provision at service as well as application layer. The results shows that the model can be adopted as a generic service-oriented solution in scenarios where actuating devices connected in a network need to be controlled. Further, the prototype system presented here will be developed into a larger system for autonomous control system based on the integration of various service providers like the one presented in this article. We also intend to include data semantics based on ontology for improved automated reasoning and decision making capabilities in the future system.

Acknowledgments

This work was partly supported by Institute for Information and 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 and communications Technology Promotion).

References

[1] Cattell R., Inscore J., and Partners E., *J2EE in Practice: Building Business Applications with the Java 2 Platform Enterprise*, Amazon, 2001.

[2] Chang E., "Internet based Remote Camera Control System," *in Proceeding of the IEEE Intelligent Vehicles Symposium*, Tokyo, pp. 126-129, 1996.

[3] Gubbi J., Buyya R., Marusic S., and Palaniswami M., "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645-1660, 2013.

[4] Han H., Kim S., Kim J., Lee E., and Kim H., "Implementation of Internet-based Personal Robot with Internet Control Architecture," *in Proceedings of International Conference on Robotics and Automation*, Seoul, pp. 217-222, 2001.

[5] Hernández A., Mora F., Villegas G., Passariello G., and Carrault G., "Real-time ECG Transmission via Internet for Nonclinical Applications," *IEEE Transactions on Information Technology in Biomedicine*, vol. 5, no. 3, pp. 253-257, 2001.

[6] Lin P. and Broberg H., "Internet-based Monitoring and Controls for HVAC Applications," *IEEE Industry Applications Magazine*, vol. 8, no. 1, pp. 49-54, 2002.

[7] Luo R., Tzou J., and Chang Y., "An Internet-Based Remote Control and Monitoring Rapid Prototyping System," *in proceeding of 27th Annual Conference of the IEEE Industrial Electronics Society*, Denver, pp. 159-164, 2001.

[8] Maskeliunas R. and Raudonis V., "ROBOSOFA-Low Cost Multimodal I/O Fusion for Smart Furniture," *The International Arab Journal of Information Technology*, vol. 10, no. 4, pp. 317-328, 2013.

[9] Perera C., Zaslavsky A., Christen P., and Georgakopoulos D., "Sensing as a Service Model for Smart Cities Supported by Internet of Things," *Transactions on Emerging Telecommunications Technologies*, vol. 25, no. 1, pp. 81-93, 2014.

[10] Sessions R., *COM and DCOM: Microsoft's Vision for Distributed Objects*, Amazon, 1997.

[11] Sou K., Weimer J., Sandberg H., and Johansson K., "Scheduling Smart Home Appliances using Mixed Integer Linear Programming," *in proceeding of 50th IEEE Conference on Decision and Control and European Control*, Orlando, pp. 5144-5149, 2011.

[12] Taylor K. and Dalton B. "Internet Robots: a New Robotics Niche," *IEEE Robotics and Automation Magazine*, vol. 7, no. 1, pp. 27-34, 2000.

[13] Wang Z., Ding H., Han J., and Zhao J., "Secure and Efficient Control Transfer for IoT Devices," *International Journal of Distributed Sensor Networks*, vol. 2013, pp. 1-8, 2013.

[14] Yang Z. and Duddy K., *CORBA: a platform for distributed object computing*, SIGOPS Operating Systems Review, 1996.

[15] Yick J., Mukherjee B., and Ghosal D., "Wireless Sensor Network Survey," *Computer Networks*, vol. 52, no. 12, pp. 2292-2330, 2008.

[16] Weiser M., Gold R., and Brown J., "The Origins of Ubiquitous Computing Research at PARC in the late 1980s," *IBM Systems Journal*, vol. 38, no. 4, pp. 693-696, 1999.

[17] Zhang Q., Cheng L., and Boutaba R., "Cloud Computing: State-of-the-Art and Research Challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7-18, 2010.

Muhammad Khan received his B.S. and M.S. degrees from Computer Software Engineering Department, University of Engineering and Technology Peshawar in 2008 and 2012 respectively. Meanwhile, he had been a part of the software development industry in Pakistan as a designer and developer. From 2010 onwards, he has been working as a faculty member at his parent department. Currently, he is pursuing his Ph.D. studies from Jeju National University, South Korea and is associated with the Mobile Computing lab at JNU. The major focus of his work is the application of software design strategies towards the design and development of Sensor-Actuator Networks and Internet of Things.

DoHyeun Kim 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, optimization algorithm and context prediction.