International Journal of Smart Home
Vol. 9, No. 10, (2015), pp. 9-14
http://dx.doi.org/10.14257/ijsh.2015.9.10.02

Middleware-based Cooperative Context Dissemination for Smart Home Application

M. Robiul Hoque, M. Humayun Kabir, Dong-Hyuk Lim and Sung-Hyun Yang

Department of Electronic Engineering,
Kwangwoon University, 139-701 Seoul, Republic of Korea
{robiul, humayun, buldok0828, hyang}@kw.ac.kr

Abstract

A smart home service is provided by following several contexts of environment and user activities. Moreover, some contexts are common for different services. So, it is unwise to separately compute the same context for different services due to the processing time cost, here a context reusable mechanism can significantly reduce the context computing cost. For this purpose, middleware-based implementation is a good alternative. Middleware uses raw data through sensors and user profiles from database then generates context using these data, and finally conveys context to applications. In this paper, we propose a middleware architecture that shares context in a cooperative manner so that it can be reused among applications. Basically, this middleware generates a particular context at once for a time interval and disseminates among registered applications using a new mechanism. Measuring inequality of the context computing time over application-based implementation exposes the effectiveness of this middleware.

Keywords: *Middleware, Smart home, Context aware service, Cooperative middleware*

1. Introduction

The smart home is fitted with different types of sensors, actuators, home appliances, smart devices, and so forth; and can provide services to home occupants based on the current context. By context, we refer to any information that can be used to characterize the situation of an entity, where an entity can be a person, place, or physical or computational object [1]. To effectively realize a smart home service, we need to implement a software system, called smart home middleware, to support i) acquiring context information from various sources and producing context, ii) processing and reasoning of context, and iii) carrying out dissemination of context to interested parties. Every application needs a set of contexts to provide a service. Middleware computes those contexts and shares among applications without separately considering each application. To provide this facility, we present a middleware that supports cooperative context dissemination.

Several context-aware systems have been developed to demonstrate its usefulness. Refs [2] and [3] present application-based context-aware systems to provide personalized medical services. Extension of the application area of the above systems is difficult due to the lack of a separate middleware. Refs [4] and [5] propose middleware systems to provide context-aware services in the smart home. Another similar research work implements a smart classroom service [6]. But none of the above addresses cooperative characteristics of middleware systems. On the other hand, our middleware supports cooperative context sharing which reduces context computing time cost significantly, while the OSGi-based service oriented implementation helps system extension in multiple-domains. We have developed a context reasoned on a rich resource device e.g., a home server, which also assists in reducing context computing time. Moreover, the context is modeled based on ontology [7] using OWL- Web Ontology Language which enables logic inference and context consistency, presented in Ref. [8].

The rest of this paper is organized as follows. Section 2, middleware architecture is presented in details. In Section 3, implementation of this middleware architecture is presented. Then, in Section 4, cooperative characteristic of this middleware is discussed and finally, in Section 5 authors conclude this paper and indicate future plan.

2. Cooperative Middleware Architecture

Figure 1 shows our middleware architecture. The three main layers in the hierarchical design are the context provider layer, context processing layer and context-aware application layer. The context provider and application can reside on any computational device *e.g.*, smart phone, laptop, pc, or server. But the context processing layer resides only on a server.

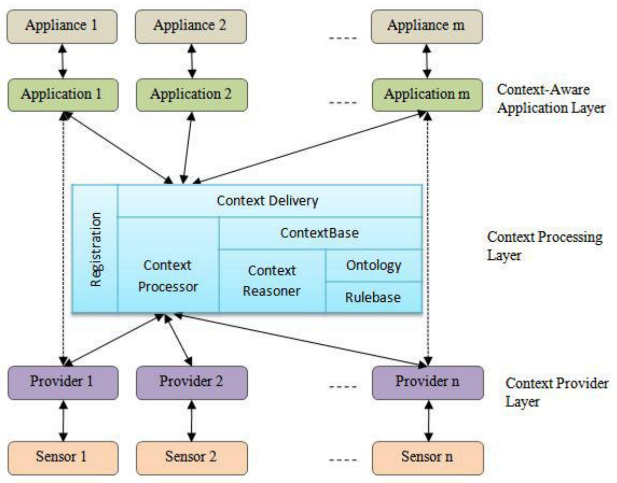

Figure 1. Three-layer Cooperative Middleware Architecture

2.1. Context Provider Layer

The context provider performs the task of context information acquisition, producing low-level context and conveying this to registered services i.e., application or context processor. Sensors may provide information with its own format e.g., a location sensor provides location using a specific coordinate system; but in the real world, we need more high level information. For example, instead of coordinates, we need information like bedroom, living room, home or office. To make this high level symbolic information, the context provider uses mapping techniques. Similarly, for temperature, we need to know whether it is cold, warm or hot. Here, a fuzzy function provides more precise information from several overlapping temperature groups. Using this symbolic information, the context provider generates a simple context with the help of context ontology where the ontology semantically represents domain knowledge. For example, the ontology provides a relation between location and temperature which is '*hasTemperature*'; and the context provider produces a simple context: *hasTemperature*(bedroom, cold).

2.2. Context Processing Layer

The context processor decides whether a context is needed to combine with others to make a high-level context. High-level context may be computed from combining only low-level contexts or both low-level and high-level contexts in a hierarchical fashion as shown in Figure 2.

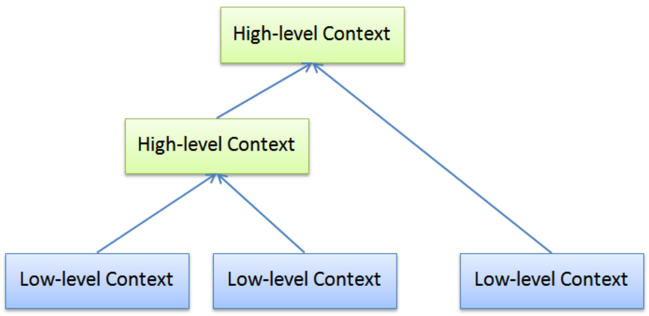

Figure 2. Context Combination Hierarchy

The context reasoner is an inference engine that is responsible for computing high-level context from low-level contexts using rules and domain ontology. It supports two types of reasoning: ontology reasoning and user-defined rule-based reasoning. Ontology reasoning retrieves the relations between context entities and maintains class consistency. On the other hand, rule-based reasoning asserts new contexts from other contexts. In this middleware, we implement rule-based reasoning using a forward chain algorithm based on the RETE algorithm [9]. Here a context processor with context reasoner acts as a context provider. Equation 1 shows a user-defined rule to deduce a person's current status.

$$Person(?x) \wedge BedRoom(?y) \wedge PIR(?r) \wedge BedPressure(?p) \wedge hasLocation(?x,?y) \wedge isPositionOf(?y,?r) \wedge isPositionOf(?y,?p) \wedge hasMotion(?r,false) \wedge hasStatus(?p,ON) \rightarrow engagedIn(?x,Sleeping) \quad \ldots\ldots\ \ldots\ldots. \quad (1)$$

A registration service provides facilities to declare the capabilities of context providers and the context processor. The registration service stores the entity name, context type and access point of each registered context provider in the context base. It is also able to track and adapt to the dynamic changes of context providers based on adding or removing sensors in the smart home.

2.3. Context-Aware Application Layer

An application generates smart home services using the required contexts. The application gets the access point of a context provider through context delivery. That means that the context delivery gets a query from applications and, after processing that query, returns the required context provider access point. Then the application can directly access the context provider. Based on the selected service, a set of commands is generated to control home appliances as a response to this service.

3. Implementation

We adopt the OSGi [10] service platform, which is a software stack as shown in Figure 3. It provides a bundle management mechanism to control and manage several services. We implement middleware components as services using Java. We develop the context provider and application as distributed independent services where we use Java RMI as the communication mechanism. At the time of system initialization, applications get the requested context providers’ access point to directly receive the context from those context providers.

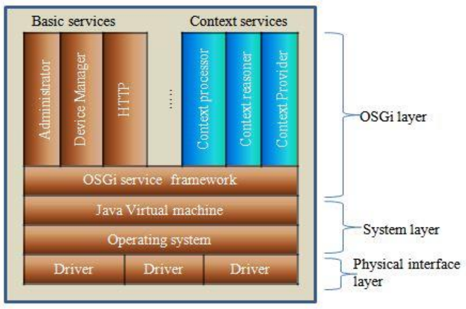

Figure 3. Software Architecture For The Osgi Service Platform

4. Cooperative Characteristic

The context provider produces a context which is used by applications. Different applications provide different services at different times based on the current context. Instead of separately producing a context for each application, the middleware produces the context at once for a time interval and shares among applications.

We can describe the cooperative characteristic with the example of smart home applications. Suppose a home user enters into his bedroom; an RFID tag will provide his location, and the light adjustment application turns on the room light based on the location context. The temperature control service needs the user location and current temperature context in that location to adjust the temperature based on that user’s preference. In the case of a sleeping service, the application needs some additional context such as a motion context provider that provides no motion in the room, and a pressure detection provider that provides the pressure sensor on the bed is ON. Then, the application provides a sleeping service (turn off light, close window curtain and adjust room temperature) that combines the previous location and the temperature context, and the newly received motion and pressure context. Here, the same location context is shared among three applications and the temperature context is shared between two different applications.

Consider that the separate context computing times of location, temperature, motion and pressure are l_t, t_t, m_t, and p_t, respectively. In the case of our middleware-based service implementation, the total computing time of these contexts for the mentioned three services is $(l_t+t_t+m_t+p_t)$ms. On the other hand, for application-based implementation, without middleware, the context computing time of the same three services is $\{l_t+(l_t+t_t)+(l_t+t_t+m_t+p_t)\}$ms or $\{(2l_t+t_t)+(l_t+t_t+m_t+p_t)\}$ms. Using inequality, we can compare these computing times as $(l_t+t_t+m_t+p_t)<\{(2l_t+t_t)+(l_t+t_t+m_t+p_t)\}$. In the worse case, where distinct contexts are required for different services, the context computing time for both implementations will be the same. Thus, we can finally revise the inequality as, $(l_t+t_t+m_t+p_t)\leq\{(2l_t+t_t)+(l_t+t_t+m_t+p_t)\}$.

Obviously, our implementation is more efficient compared with application-based implementation. Moreover, if the number of applications increases, the shared contexts will also increase, while the context computation cost will be significantly reduced. Hence, applications and context providers of the middleware cooperate among each other. So, we call our middleware supports context sharing in a cooperative manner.

5. Conclusions

In this paper we have presented context-aware middleware design and its cooperative characteristic. The cooperative property facilitates middleware to provide service in time by reducing the context computation cost. This middleware demonstrates acceptable rates of correct service for users by inferring the current context in the smart home. In the future, we will extend our middleware for multiple-domains.

Acknowledgements

This work was supported by the Industrial Strategic Technology Development Program (10041788, Development of Smart Home Service based on Advanced Context-Awareness), funded by the Ministry of Trade, Industry & Energy (MI, Korea). The present research has been conducted by Research Grant of Kwangwoon University in 2015.

References

[1] A. K. Dey and G. D. Abowd, "Towards a better understanding of context and context-awareness", Proceedings of Workshop on the what, who, where, when and how of context-awareness, **(2000)**.
[2] S. -J. Shin, "Development of life management system for elderly and people with disabilities", The Journal of IIBC, **(2014)**.
[3] J. -S. Seo and S. -C. Park, "Design and Implementation of support system for personalized medical service based on mobile", the journal of IIBC, **(2013)**.
[4] T. Gu, H. K. Pung, and D. Q. Zhang, "A service-oriented middleware for building context-aware services", Journal of Network and Computer Applications, **(2005)**.
[5] M. R. A. Jose and W. Johnny, "Service-oriented middleware for smart home applications", Proceedings of Wireless HIVE Networks Conference, August, **(2008)**.
[6] W. Win, Y. Shi and Y. Suo, "Ontology-based context-aware middleware for smart spaces", Tsinghua Science and Technology, **(2007)**.
[7] D. Fensel, "Ontologies, A silver bullet for knowledge management and electronic commerce", Springer, **(2003)**.
[8] M. R. Hoque, M. H. Kabir, T. Minami and S. -H. Yang, "Ontology-based Context Modeling for Smart Home Domain", Proceedings of the 1st International Joint Conference on Convergence Vietnam, **(2015)** February 2015, pp. 5-7.
[9] C. L. Forgy, "RETE, a fast algorithm for the many pattern/many object pattern match problem", Artificial intelligence **(1982)**.
[10] "Open Services Gateway Initiative", OSGi Service Platform, http://www.osgi.org

Authors

M. Robiul Hoque, He has received B.Sc (Hon's) and M.Sc. degree in Computer Science and Engineering from Islamic University, Kushtia, Bangladesh, in 2003 and 2004 respectively. He is an Assistant Professor at Islamic University, Kushtia, Bangladesh. Now, he is pursuing his Doctoral degree at Kwangwoon University, Republic of Korea. His main research interests include Context-Aware System, Ubiquitous Computing, and Smart Home, Sensor Networks, Image and Speech processing.

M. Humayun Kabir, He has received his B.Sc (Hon's) and M.Sc degree in Applied Physics, Electronics and Communication Engineering from Islamic University, Kushtia, Bangladesh, in 2001 and 2003 respectively. He is an Assistant Professor at Islamic University, Kushtia Bangladesh. Now, he is pursuing his Doctoral degree at Kwangwoon University, Seoul, Republic of Korea. His main research interests are Context-Aware Systems for the Smart Home, Embedded Systems, M2M, Sensor Networks, Image and Speech processing.

Dong-Hyuk Lim, He has received his Bachelor's degree in Department of Display and Electronic Engineering from Doowon Technical University, Gyeonggi-do, and Republic of Korea in 2014. Now he is pursuing his Master's degree at Kwangwoon University, Seoul, and Republic of Korea. His main research interests are M2M, Sensor Networks.

Sung-Hyun Yang, He has received his B.S. and M.S. degree in Electrical Engineering from Kwangwoon University, Seoul, Republic of Korea, in 1983 and 1987 respectively. He completed his Ph.D. from Kwangwoon University in 1993. He is a Professor in Electronic Engineering at Kwangwoon University, Seoul, and Republic of Korea. He is a Director of the Ubiquitous Home Network Center, Kwangwoon University. He was a Research Scientist at Boston University from 1996 to 1998. He was Chairman of the Home Network Market Activation Section, Korean Association for Smart Home from 2007 to 2008. His main research interests are Digital Logic, Embedded Systems, M2M, Next Generation Ubiquitous Home Networks, and Context-Aware System.