Development of a User-Oriented IoT Middleware Architecture Based on Users' Context Data

Taehyun Ha, Sangwon Lee(✉), and Narae Kim

Department of Interaction Science, Sungkyunkwan University, Seoul, Korea
ontophilla@gmail.com, upcircle@skku.edu,
narae0113@naver.com

Abstract. How to manage the connections of things efficiently with heterogeneous things is one of the important issues for IoT middleware development. Many researches have been focused on this issue but still no one accepted as the common model in the IoT environment. In this sense, we aim to develop a new IoT middleware architecture containing simple key-value model based and no-model based context-awareness function. The suggested middleware represents the context data without strictly defined data structure. Rather, it processes the context more focusing on the other technical aspects. We build the middleware architecture based on the basic structure of GSN (Global Sensor Networks). Also, by adapting no-model based context representation method suggested by Habit, we added the context-awareness function to the GSN. Through the middleware, many heterogeneous things not integrated on the standard structure can be managed effectively. We expect the suggested middleware can provide a flexible solution in current IoT development situation.

Keywords: IoT middleware architecture · GSN · Context-awareness

1 Introduction

Through the evolution of Internet, the concept of IoT (Internet of Things) that everyday objects are connected to the Internet has been newly devised. Ashton (2009) [2] firstly coined the definition in a presentation in 1998, and through the MIT Auto-ID centre, ITU(International Telecommunication Union), and other researchers, the IoT researches have been widely discussed [8]. Nowadays, over the range from household to industry, many IoT applications are being developed by researchers and engineers.

The IoT bundles many technology together and needs to be supported by middleware solutions. Many kinds of the middleware solutions have been suggested (e.g., GSN [1], Hydra [9], Ubiroad [10]), but not yet any of these solutions are not accepted as the standard solution. Especially, as Perera et al. (2014) [8] mentioned, even the context-awareness function is important in the IoT paradigm, many of these solutions have been not focused on the context-awareness. In this sense, it can be the meaningful research that developing IoT middleware architecture containing effective context-awareness functions.

N. Streitz and P. Markopoulos (Eds.): DAPI 2015, LNCS 9189, pp. 287–295, 2015.
DOI: 10.1007/978-3-319-20804-6_26

Besides, several researchers have been studied for the context-awareness. Especially, it is important that how to represent these complex users' context. Bellavista, et al. (2012) [4] categorized the context representation models into the three models: general models (e.g., key-value, markup scheme, and object oriented models), domain-specific models, and no model. We focused on the no model based methods because it can covers wide scopes of user context despite its some limitations.

In this paper, we introduce a user-oriented middleware architecture based on the users' context data. The basic middleware architecture is based on the GSN. Also we adapt the context awareness function which is not supported by the GSN devised by Habit [6]. The suggested middleware analyze the users' context data and utilizes it to manage the connections of things (relationship management), find the optimal path of the nodes (service management), and decide where the new things are added to the sensor network (service discovery).

2 Backgrounds

2.1 IoT Middleware Architecture

Middleware is the software that facilitates communication by connecting heterogeneous devices, hardware, and protocols in various environments. Among the several studies reviewing different kinds of IoT middleware, Bandyopadhyay, et al. [3] presented the overall features of the existing middleware in the IoT domain. In their study, functional components of IoT-middleware were discussed in the five sections: interoperation, context detection, device discovery and management, security and privacy, and managing data volumes. Based on these functional aspects, they classified the some existing middleware. Table 1 shows the results.

As shown in Table 1, Hydra [9] and Ubiroad [10] cover the all functional components of IoT-middleware. The Hydra (Officially, the Hydra project was ended in 2010, and now its name has been changed as "LinkSmart" because of the problem of intellectual rights to the name "Hydra") adapted a context-awareness function based on SWRL (Semantic Web Rule Language) and OWL (Web Ontology Language). Thus it can process the users' contexts with more expressive power [11]. On the other hand, the Ubiroad's solution is basically based on the concept of GUN (Global Understanding Environment). Its layers consist of semantic adapters, behavioral, coordination layer, and based on the users' scenario, the middleware manages the connections of things. However, still not enough cases of these applications have been reported, and more precisely no one covers the full set of the functionalities to meet the requirement of IoT-middleware [3].

Meanwhile, GSN (Global Sensor Networks) [1] has been developed to address the sensor networks in general. Thus, it can be adapted to the various situations. Also, as the connections of things are depicted as a network, we can catch the connections intuitively and apply some network analysis techniques to the network (e.g., complex network analysis). Figure 1 shows the overall structure of GSN.

Table 1. IoT-middleware comparison

IoT middleware	Features of middleware				
	Device management	Interoperation	Platform portability	Context awareness	Security and privacy
HYDRA	✓	✓	✓	✓	✓
ISMB	✓	✗	✓	✗	✗
ASPIRE	✓	✗	✓	✗	✗
UBIWARE	✓	✗	✓	✓	✗
UBISOAP	✓	✓	✓	✗	✗
UBIROAD	✓	✓	✓	✓	✓
GSN	✓	✗	✓	✗	✓
SMEPP	✓	✗	✓	✓	✓
SOCRADES	✓	✓	✓	✗	✓
SIRENA	✓	✓	✓	✗	✓
WHEREX	✓	✓	✓	✗	✗

Fig. 1. Overall structure of GSN

The GSN covers several interface protocols such as Zigbee, RFID, WiFi, and Bluetooth. It uses basic unit named virtual sensor (node) and communicates with them using the wrappers. The official project website (https://github.com/LSIR/gsn) provides the packages and users can set up the environment easily.

However, as already shown in Table 1, the GSN does not cover the interoperation and context awareness functions. Also, as Perera, et al. (2013) [7] mentioned, when more and more sensors get connected to the Internet, the sensor search functionality becomes critical. Nevertheless, if we develop a model addressing these limitations, the middleware can be used as a basis for designing user-centered middleware. In other words, by adapting context awareness methods to the GSN, we would be able to gather the users' context information and utilize it for setting the network connections between users and things based on the GSN concept.

2.2 No Model Based Context Representation

To apply the context awareness function to the GSN, we need to consider how represent, process, and deliver the context, and support it in runtime. This is the concept of context data distribution. Bellavista et al. (2012) [4] have progressed the surveys of the context data distribution for mobile ubiquitous systems. As a part of the research, they classified the context data representation models as like Fig. 2.

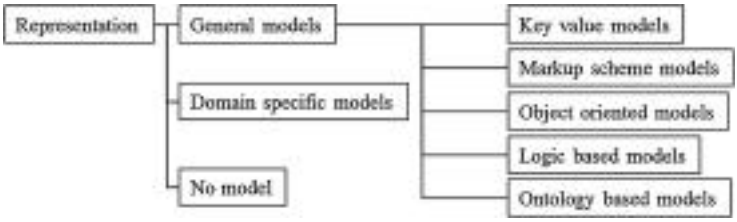

Fig. 2. The classification of the context data representation

As each of the data representation model has some limitations, some hybrid models based on two or even more have been suggested to overcome the limitations. Among the context representation models in Fig. 2, we need to focus on the no model based methods. The methods do not represent the data structure. Rather, it more focuses on the other technical aspects. As the no model based representation methods are not limited by data structures, it can describe more extensive context than the other methods. In other words, this method allows covering heterogeneous things in middleware more easily.

Among the related researches for the no model based methods, Habit [6] uses the user's context data to create data distribution routes. Figure 3 shows the example of content dissemination network configuration, and detail process descriptions for the Habit are below.

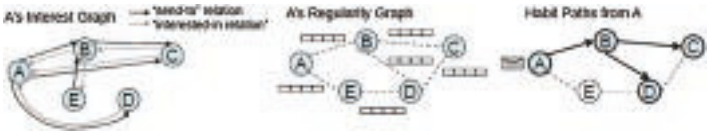

Fig. 3. Content dissemination networks from A's viewpoint

Building a Content Dissemination Networks: In Fig. 3, *A* is the one of the nodes in Habit. *A* maintains a list of nodes interested in receiving contents of *A*. The other nodes that encounter node *A* are called *familiar strangers* (*FS*). *A* maintains the *familiar strangers* in a certain number of hops (*maxHops*) and the number of *familiar strangers* is limited by the maximum number (*maxFS*). Every node in Habit maintains the interest graph. Besides, *A* also configures the *regularity graph* (i.e., regularity table). *A*'s

regularity graph consists of *regularity weights* between *A* and *familiar strangers*. The *regularity weights* mean the number of times specific node (*A*) meets another node in given regularity interval of the week. For example, if *A* meets the *B* three times in the hour slot Monday 10AM-11AM (this day/time slot can be adjusted to the human-meaningful time slot such as commute item slot, working time slot) in last five weeks, the *regularity weight* is 0.6. Each node can build the content dissemination networks by: (1) changing the direction of each edge in the *interest graph* (transforming the relation "interest in" into the relation "send content to"), and (2) overlaying it onto the *regularity graph*.

Reasoning on the Content Dissemination Network: After Habit builds the content dissemination network, Habit finds the optimal route for node *A*. This is the reasoning of the content dissemination network, and consists of four steps as follows.

(1) **Determine Recipients:** the reverted *interest graph* is consulted to determine what nodes *R* are interested in receiving content from *A* ($R = \{B, C, D\}$).
(2) **Find Cheapest Paths:** Find the paths that can reach the node of destination. In Fig. 3, if *A* try to reach node *D*, there are three routes that *A-B-D*, *A-B-C-D*, and *A-E-D*. In this case, the first and second routes are chosen because their costs are 0. On the other hand, the third route's cost is 1 because of *E* (not the member of *R*), and this route will be not chosen. Besides, the Habit also calculates the regular weights (i.e., delivery probabilities) of the chosen routes. To avoid the computational complexity, Habit uses simple heuristic method: only the first non-zero regularity weight between a pair of nodes is considered before moving on to the next edge, and set the minimum *regularity weight* as the delivery probability of the route. Let assume the *regularity weight* between *A* and *B* is 0.7, and *B* and *D* is 0.3. Then the delivery probability of the route *A-B-D* is 0.3 ($min\{0.7,0.3\}$).
(3) **Select Paths:** If more than one route is chosen, the route has the maximum delivery probability is selected.
(4) **Optimizations:** As *A* covers nodes within *maxFS* and *maxHops* range, *A* may not always be aware of all nodes interested in receiving *A*'s messages. In this case, the intermediary may then follow the same steps that *A* did, and discover the new paths.

To describe sensor networks as an intuitive model and make a basis for applying network analysis techniques into the sensor network, we develop a new IoT middleware architecture basically based on GSN architecture. Also, to address the GSN's limitation for context awareness functions, we additionally adapt simple key-value context model and the Habit's content dissemination method. Similar approach has been studied by Perera et al. (2013) [7]. They developed IoT middleware architecture containing context awareness function. Their middleware aimed to search the sensors more efficiently in huge networks environment by using the ontology based context awareness function. In this model, the context property consists of such as accuracy, reliability, latency. As this model aims to reduce the sensor networks size, it is more appropriate to the users who have to deal with tremendous items. On the other hand, we aim to develop the user-centered middleware identifying the context in individual dimension (e.g., user preferences) [5]. Our approach more actively utilizes the users' usage patterns into the middleware, and thus can provide the practical solutions.

3 Development of a User-Oriented IoT Middleware Architecture

In this section, we describe the user-oriented IoT middleware architecture. As the middleware is basically based on the GSN, basic structure of our middleware follows the GSN structure. We describe our middleware architecture by the three conceptual parts of modules. Generally, the middleware architecture consists of some other sub modules, but we would more focus on the three parts of our model originally suggested. Figure 4 shows the overall architecture of our middleware.

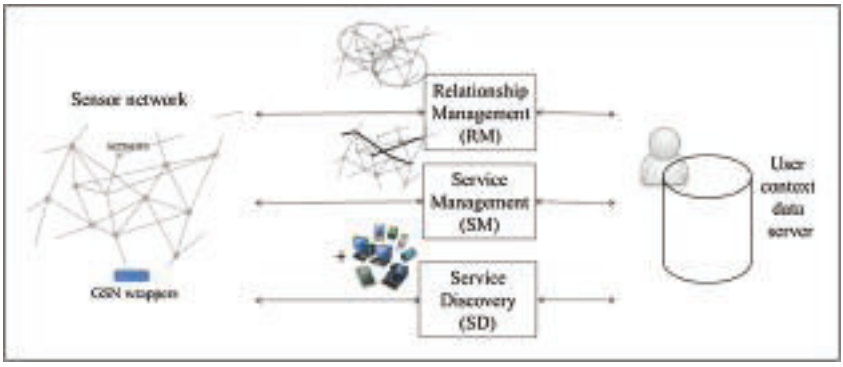

Fig. 4. The IoT middleware architecture based on users' context data

3.1 Relationship Management Module

To manage the connections of things efficiently, clustering methods can be used as a useful solution in aspect of it reduces the computational complexity. In our middleware, the relationship management module clusters the things by either local first strategy or content (thing's usage) first strategy. For example, let assume the situation there are four things in the sensor network: A (television located in house 1), B (air heater located in house 2), C (television located in house 3), and D (robot vacuum cleaner located in house 1), and house 1 and 2 are placed in near distance. In this situation, the relationship management module makes the two clusters of the things: {A, B, D} and {C} if the module uses local first strategy. On the other hand, if the relationship management module uses content first strategy, the module makes the three clusters of things: {A, C}, {B}, and {D}. These clustering strategies are decided referring the user's pattern, and refreshed at the defined time interval. Thus it is related to the service management module below (the relationship management module checks what types of the contextual properties (i.e., location or usage) are in close relationship between the things in optimal paths configured by the service management module). Figure 5 shows the relationship management strategies.

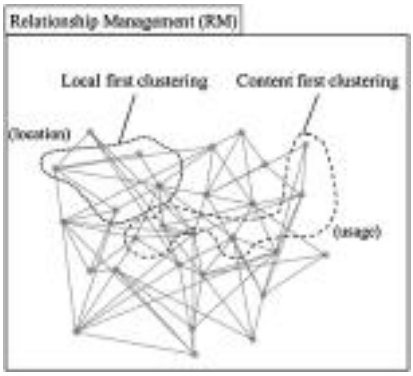

Fig. 5. Relationship management strategies

3.2 Service Management Module

Based on the concept of content dissemination network in Habit, the service management module analyzes the user's pattern and selects the optimal path. In other words, based on the user's transaction history for the things in networks, the service management module configure the content dissemination network containing send to- and interested in- graphs, and regularity weights. In this content dissemination network, routes from start node (things) to destination node are listed and calculated their costs. After the calculations, the module chooses cheapest routes. If the more than one route is chosen, then select the route which has the maximum delivery probability. Figure 6 shows the example for how the service management module selects the optimal path.

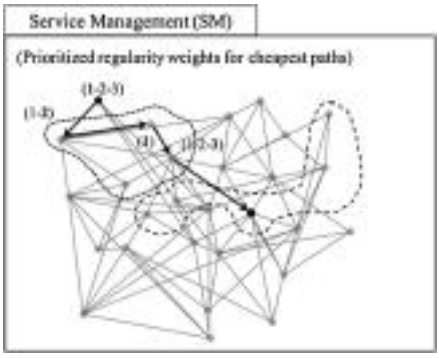

Fig. 6. Example of selecting processes in service management module

3.3 Service Discovery Module

Service discovery module covers new things that are not added to the middleware. When a new thing tried to communicate with the things in the sensor network, the service discovery module requests the new thing's location and usage. Then add it to the network by using GSN wrapper. According to the strategy currently used by relationship management module, the service discovery module decides a position where the new thing should be added, and what nodes would be prepared to the communications with the new node. For example, if the relationship management module uses the location first strategy, the new node is placed based on its location. At the same time, the service discovery module informs other nodes placed nearby the new node to prepare the communications with the new node. On the other hand, if the relationship management module follows the content first strategy, the new node is placed nearby the nodes which have similar usages. Figure 7 shows the process in service discovery module.

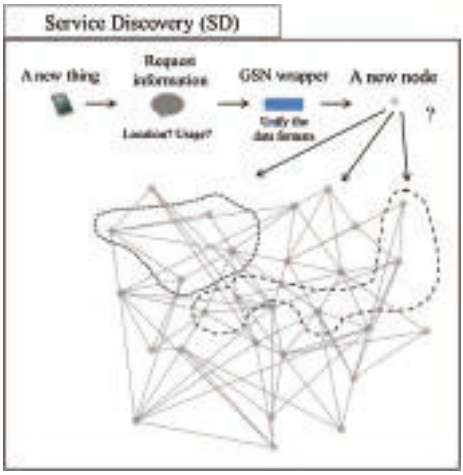

Fig. 7. Process in service discovery module

4 Conclusion

In this study, we proposed a user-oriented IoT middleware architecture based on the GSN and Habit. The middleware adds the things in the sensor network using the GSN wrapper, and each module in the middleware manages the connection of things using the users' context. As the architecture does not define the strict context structures, it is relatively free to analyze context, and thus can cover more wide range of the context. Even the middleware requires context information of things that location and usage, it is quite little things comparing other middleware requires.

However, in aspect of accuracy, it can be pointed out that other context representation models such as ontology based models, logic based models can be better to

distribute and analyze the user's context. Moreover, it is another matter to pick the specific method in the context representation methods. It requires many times and costs for making the systems based on various context distribution methods. We need to provide a practical solution for combining heterogeneous things into the IoT environment. In this sense, our study tried to develop the middleware architecture which can be used more broadly with less effort.

In future study, we will conduct the validation test for the suggested middleware in user scenario based environments. Some validation criterions can be used such as accuracy and latency. Also, comparative analyses for the other types of context distribution methods can be considered. Through these validation tests, some limitations of the suggested user-oriented middleware architecture would be supplemented.

Acknowledgements. This research was supported by the Ministry of Education, South Korea, under the Brain Korea 21 Plus Project (No. 10Z20130000013) and Basic Science Research Program (No. NRF-2014R 1A 1A2054531).

References

1. Aberer, K., Hauswirth, M., Salehi, A.: Global sensor networks. Technical report (2006)
2. Ashton, K.: That 'internet of things' thing. RFiD J. **22**(7), 97–114 (2009)
3. Bandyopadhyay, S., Sengupta, M., Maiti, S., Dutta, S.: Role of middleware for internet of things: a study. Int. J. Comput. Sci. Eng. Surv. **2**(3), 94–105 (2011)
4. Bellavista, P., Corradi, A., Fanelli, M., Foschini, L.: A survey of context data distribution for mobile ubiquitous systems. ACM Comput. Surv. (CSUR) **44**(4), 24 (2012)
5. Eugster, P.T., Garbinato, B., Holzer, A.: Middleware support for context-aware applications. In: Garbinato, B., Miranda, H., Rodrigues, L. (eds.) Middleware for Network Eccentric and Mobile Applications, pp. 305–322. Springer, Heidelberg (2009)
6. Mashhadi, A.J., Ben Mokhtar, S., Capra, L.: Habit: leveraging human mobility and social network for efficient content dissemination in delay tolerant networks. In: World of Wireless, Mobile and Multimedia Networks & Workshops (WoWMoM), pp. 1–6. IEEE (2009)
7. Perera, C., Zaslavsky, A., Christen, P., Compton, M., Georgakopoulos, D.: Context-aware sensor search, selection and ranking model for internet of things middleware. In: IEEE 14th International Conference on Mobile Data Management (MDM), pp. 314–322. IEEE (2013)
8. Perera, C., Zaslavsky, A., Christen, P., Georgakopoulos, D.: Context aware computing for the internet of things: a survey. Commun. Surv. Tutorials **16**(1), 414–454 (2014)
9. Sarnovský, M., Kostelník, P., Butka, P., Hreňo, J., Lacková, D.: First demonstrator of hydra middleware architecture for building automation. In: Proceedings of the Scientific Conference Znalosti (2008)
10. Terziyan, V., Kaykova, O., Zhovtobryukh, D.: Ubiroad: semantic middleware for context-aware smart road environments. In: Fifth International Conference on Internet and Web Applications and Services (ICIW), pp. 295–302. IEEE (2010)
11. Zhang, W., Hansen, K.M.: Towards self-managed pervasive middleware using owl/swrl ontologies. In Fifth International Workshop on Modelling and Reasoning in Context (MRC), pp. 1–12. TELECOM Bretagne (2008)