Context Management for User-centric Context-aware Services over Pervasive Networks

Sin-seok Seo
Dept. of Computer Science and Engineering, POSTECH, Korea
Email: sesise@postech.ac.kr

Joon-Myung Kang
Dept. of Electrical and Computer Engineering, Univ. of Toronto, Canada
Email: joonmyung.kang@utoronto.ca

Yoonseon Han and James Won-Ki Hong
Division of IT Convergence Engineering, POSTECH, Korea
Email: {seon054, jwkhong}@postech.ac.kr

***Abstract*—Large and various amounts of context data related to a user's environment are available from different domains including mobile devices, smarthomes, wearable sensors, and social networking services. These context domains are interconnected and the context data from them can be shared thanks to mobile, pervasive, convergent, and ubiquitous technologies. We can provide user-centric context-aware services by aggregating and associating the diverse types of context data distributed over multiple domains around a user. However, state-of-the-art research efforts have been devoted to managing context only in a single domain. In this paper, for user-centric context management, we propose 1) a flexible and extensible technology-neutral information model that can represent generic concepts of context; 2) a hierarchical context management architecture that can understand and manage complex interactions among multiple contextual entities.**

***Index Terms*—Context-aware Service, Context Management, Information Model, Pervasive Computing**

I. Introduction

The number of sources that can produce various types of context data related to a user's environment is rapidly increasing thanks to the improvements of pervasive networking, sensor, and mobile device technologies. Domains[1] that contain the context sources include smartphones, tablets, laptops, smarthomes, U-Health sensors, and Social Networking Services (SNSs). In addition, the context data of other socially related users' domains also constitute very important and valuable information for defining the context of a user. These context data are being exploited widely for diverse kinds of context-aware services, including personalized handover decisions for heterogeneous mobile services [1] and health state monitoring using biological sensors [2].

Context management can be categorized into three levels of management coverage:

- **Domain level**: collects, infers, and provides context data from a specific domain (e.g., smartphone and smarthome).
- **User level**: collects, aggregates, infers, and provides context data from multiple domains that are related to a user.
- **Social level**: considers the social-level context that is obtained and generated by means of interactions between a user and socially related other users, in addition to user-level context.

Currently, most research efforts have been devoted to managing context only at a single domain level [3]–[5]. If these separated context domains are related to a specific user, however, we need user-level context management that aggregates and associates the diverse types of context data obtained from the multiple domains around a user. Furthermore, management at the social-level should be the ultimate goal of context management research, because it provides very important and valuable information for defining the context of a user [4].

In this paper, we propose, for user- and social-level context management, 1) a flexible and extensible technology-neutral information model that can represent multiple domains' contexts; and 2) a hierarchical context management architecture that can understand and manage complex interactions among multiple contextual entities. We name the proposed solution U-CoUDE (pronounced "you could"), which stands for "User-centric Context manager for Ubiquitous and Distributed Environments."

II. Related Work

A context model is essential for the automatic computerized representation and storing of context data. Therefore, much research work aimed at designing flexible and useful context models has been done [6], [7]. Sinderen *et al.* [6] proposed a general context model to facilitate interoperability using Unified Modeling Language (UML) class diagrams. The components of context management infrastructure will specialize and extend their general model to represent contexts using specific technologies such as programming languages or databases. Dobslaw *et al.* [7] proposed a more flexible and sophisticated context information model that comprises *Sensed Context*, *Internet of Things*, and *Context Meta Data*. However, these modeling approaches overlooked some core context-related concepts including quality of context, relevance to a related contextual entity, and security. In Section III of this

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2012-H0301-12-3002) and by World Class University program funded by the Ministry of Education, Science and Technology through the National Research Foundation of Korea (R31-10100).

[1]In this paper, we define a domain as a collection of sensors that are located in a specific device or an area to collect related context data.

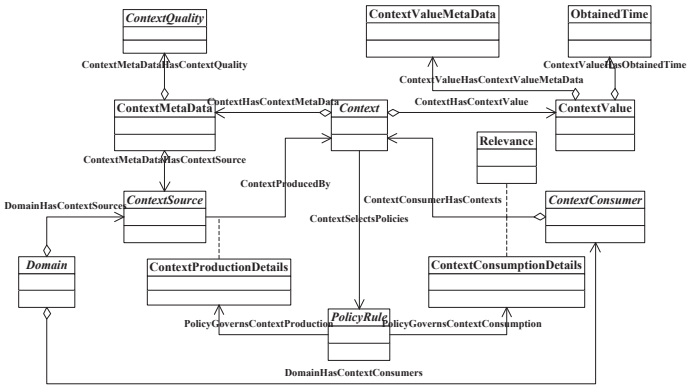

Fig. 1. U-CoUDE information model for representing a context and its related concepts.

paper, we propose a technology-neutral context information model that embraces the aforementioned concepts.

A comprehensive and efficient context management architecture is essential for understanding complex interactions among different types of contextual entities [3], [5]. Chen *et al.* [8] proposed a broker-centric agent architecture for supporting context-aware systems in smart spaces. The broker maintains and manages a shared contextual model on the behalf of a community of agents. Korpipää *et al.* [9] suggested a context management framework which has four main functional entities: *Resource Server*, *Context Recognition Service*, *Change Detection Service*, and *Security*. The *Context Manager* manages the functional entities as a centralized server. Note that the most of the existing context management architectures do not consider the user-centric nature of contexts. In other words, they lack processes that deal with the different types of contexts obtained from the multiple domains around a user. In Section IV of this paper, we propose a novel context management architecture that manages contexts from the perspective of a user.

III. U-CoUDE Information Model

A model that can represent any form of context data from multiple domains is essential for user- and social-level context management. We have studied an extensive number of research articles that deal with the concepts of context and their modeling approaches [3]–[7]. Based on the survey, we have drawn the conclusion that existing context modeling approaches are insufficiently flexible or extensible for representing various types of context data from multiple domains. In this Section, we propose a technology-neutral context information model that addresses the drawbacks of existing modeling approaches.

Fig. 1 shows the proposed U-CoUDE information model that depicts a context and its related concepts. A *Context* is produced by a *ContextSource* and consumed by a number of *ContextConsumers*. The *ContextSource* is one of various types of sensors or direct input from a user or an administrator. The *ContextSource* is located in a *Domain*; the *Domain* can have multiple *ContextSources*. For example, a smartphone *Domain* has many *ContextSources* including an accelerometer, a digital compass, a WiFi network interface, and GPS.

ContextConsumer provides context-aware services or generates abstract contexts by inference. *Relevance* represents a degree of relatedness between *ContextConsumer* and the consumed *Context*. It is calculated by considering the *Context*'s weight, importance, temporal and regional proximity, applicability, reciprocity, influence, usefulness, and priority for the *ContextConsumer*. For example, a *ContextConsumer* that provides context-aware services for a user, Alice, would assign more degrees of *Relevance* to *Contexts* that are up-to-date and obtained from a *ContextSource* near to Alice. The *ContextConsumer* is located in a *Domain*; the *Domain* can have multiple *ContextConsumers*.

ContextMetaData contains the associated *Context*'s important meta-data including a type of context, authentication-related information, a list of authorized *ContextConsumers*, and descriptions of the *Context*. *ContextMetaData* also contains *ContextSource* and *ContextQuality* as separate classes. *ContextQuality*, also known as Quality of Context (QoC), is defined by Buchholz *et al.* [10] as *"any information that describes the quality of information that is used as context information."* As can be seen, *ContextQuality* is closely related to *Relevance*. A difference between them is that *ContextQuality* represents the quality of *Context* itself, and is therefore somewhat static, while *Relevance* is influenced and determined by dynamic interactions between the *Context* and its *ContextConsumer*. As a result, *Relevance* uses *ContextQuality* (not shown in Fig. 1) as well as the relationship between *Context* and *ContextConsumer* to calculate its value.

ContextValue contains the actual value of a *Context*. For example, smartphone's GPS *Context* might have a *ContextValue* of latitude and longitude coordinates such as 37.77, -122.41. *ContextValue* has *ContextValueMetaData* that contains the associated *ContextValue*'s meta-data including a unit, sampling interval, and descriptions of the *ContextValue*. *ContextValue* also has *ObtainedTime* that is the time at which a *ContextSource* produces the *ContextValue*. We separated *ContextValueMetaData* and *ObtainedTime* because the former is static regardless of *ContextValue*'s updates, whereas the latter is changed whenever *ContextValue* is updated.

Context selects *PolicyRules* to control the input of context data produced by a *ContextSource*, and to control the output of its data in reply to a request from *ContextConsumers*. *PolicyRule* governs the production of context data from a *ContextSource* via *ContextProductionDetails*, i.e., only an authenticated *ContextSource* can update the *Context*, and *PolicyRule* controls it. *PolicyRule* also governs the consumption of context data by *ContextConsumers* via *ContextConsumptionDetails*, i.e., only an authorized *ContextConsumer* can retrieve the *Context*, and *PolicyRule* controls it. Context data have to be protected and secured from unauthenticated updates by data from sensors and unauthorized requests from context-aware applications, because most of the context data contain private and sensitive information about a user. However, despite its importance, most of the existing context modeling approaches

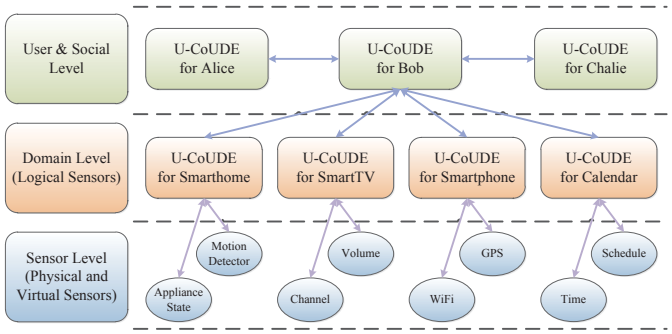

Fig. 2. Three-levels of hierarchy for managing multiple domains' contexts.

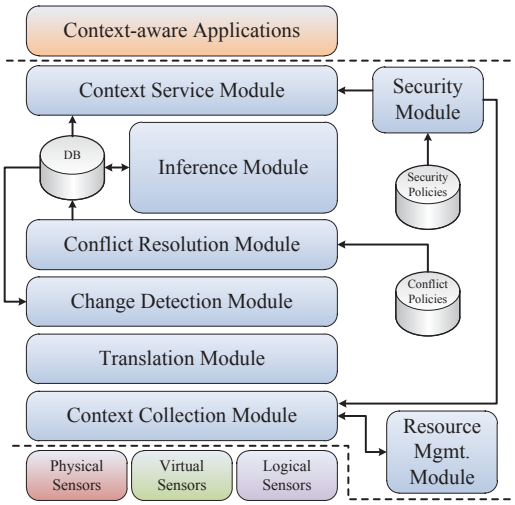

Fig. 3. U-CoUDE architecture.

do not consider this kind of security aspect at all.

IV. U-CoUDE Hierarchy and Architecture

This Section describes three levels of hierarchical relationships among contextual entities. A high-level architecture of U-CoUDE and detailed explanations of each module in the architecture then follow.

A. U-CoUDE Hierarchy

Fig. 2 illustrates three levels of hierarchical interactions among contextual entities including U-CoUDEs. At the bottom, a sensor level represents various kinds of physical or virtual sensors including motion detectors for a smarthome, volume for a smartTV, and a schedule for an electronic calendar. A domain-level U-CoUDE aggregates the sensor-level context data coming from each domain, and infers abstract contexts using only the domain's contexts. The inference function of the domain-level U-CoUDE acts as logical sensors for domain-level context-aware services or higher level U-CoUDEs, i.e., user & social-level. A user & social-level U-CoUDE aggregates all the contexts coming from the various domains that are related to the user and infers an abstract context by using contexts from multiple domains. It also interacts with other users' U-CoUDEs to obtain and provide social-level contexts.

B. U-CoUDE Architecture

Fig. 3 illustrates the architecture of a single U-CoUDE that corresponds to each U-CoUDE box at either the domain or user & social-level, shown in Fig. 2. A U-CoUDE is located in between three types of sensors and context-aware applications. It is possible for a domain-level U-CoUDE to become a logical sensor for a user & social-level U-CoUDE. In other words, the user & social-level U-CoUDE could be a context-aware application to the domain-level U-CoUDE. The followings are detailed descriptions of each module in the architecture.

1) Context Collection Module: This module collects context data from various types of sensors by directly communicating with them over pervasive networks. The selection of a specific technology for providing communication between a sensor and the Context Collection Module is strongly dependent on the type of the sensor and the characteristics of the context provided by the sensor. The context collection process is controlled by the Security Module to obtain only authenticated context data from sensors. This module also collaborates with the Resource Management Module to discover new sensors or to check the availability of the registered sensors.

2) Resource Management Module: The primary role of this module is to provide a list of available sensors to help the context acquisition of the Context Collection Module. To achieve this, the module maintains a list of available sensors by discovering new sensors either with the help of the Context Collection Module or through the direct registration of a user or an administrator. It also checks the availability of the registered sensors regularly.

3) Translation Module: The method for representing context data that differ widely depends on the types of sensors and their vendors. This kind of diversity is more obvious in user & social-level context management, because it must deal with context data from multiple domains. Accordingly, we need a method that translates different forms of context data into a standardized common form to deal with the diversity. The most promising one is an ontology-based modeling and translation approach [11]. In U-CoUDE, the Translation Module maintains a common ontology model, which is derived from the U-CoUDE information model (Fig. 1), by translating different forms of context data from multiple domains into the common ontology. Ontology-based translation or mapping is one of the most popular research topics in the knowledge engineering area. Accordingly, we could have found some useful solutions for this module [12], [13]. In our research, we do not propose a new ontology translation method, but rather take advantage of the existing approaches.

4) Change Detection Module: It can burden the context management system if the Conflict Resolution and Inference Modules are triggered to perform whenever the context data are updated. The Change Detection Module prevents this kind of problem by triggering the upper modules only when there are meaningful context changes compared to the existing

context data in a database (DB) of U-CoUDE. The application of threshold-based approaches could be sufficient to help this module make the decision whether to trigger the upper modules or not in most context change cases. Obviously, urgent or sensitive context updates could bypass this module.

5) Conflict Resolution Module: It is possible for conflicts to occur among contexts in U-CoUDE, because it obtains the contexts from multiple domains. For example, two temperature values collected from sensors in the smarthome and the smartphone, which is in the smarthome, could be significantly different. The Conflict Resolution Module handles this kind of problem in two steps: detection and resolution. In the detection step, the same types of context data that were collected from multiple domains and normalized by Translation Module are grouped together. They are then compared to each other to discover whether a context conflict occurred or not. This module assumes that a context conflict has occurred when the differences among the contexts of the same type exceed a predefined threshold. Next, in the resolution step, this module selects one of the conflicted contexts or generates a new context by modifying (e.g., averaging) the conflicted contexts. The selection and modification processes take into account the values of *ContextQuality* and *Relevance* shown in Fig. 1. For instance, in the temperature conflict situation described above, if the smartphone's *ContextQuality* and *Relevance* were greater than smarthome's values, then this module might select smartphone's temperature or give more weight to smartphone's temperature than smarthome's when calculating an average temperature. All of these operations of the Conflict Resolution Module are controlled by Conflict Policies.

6) Inference Module: Abstract and high-level contexts can be inferred by various inference techniques using either raw context data obtained from physical and virtual sensors or other inferred context data obtained from logical sensors. While most of the existing context inference approaches consider only limited contexts from a single domain, the Inference Module in U-CoUDE utilizes various types of contexts from multiple domains by considering that they are related to a user. To achieve this, the capabilities of this module are not limited to a specific inference technique; it can utilize any type of inference method as a form of plug-in.

7) Context Service Module: This module provides collected and inferred context data to various types of context-aware applications, including another U-CoUDE. The Security Module controls this Context Service Module to force it to provide context data only to authorized context-aware applications.

8) Security Module: This module controls the input to U-CoUDE of context data produced from a sensor, and the output of context data from a U-CoUDE to a context-aware application, i.e., it protects private and sensitive context data from being updated with data from an unauthenticated source and from responding to an unauthorized request. Security Policies control the operations of this module. These processes correspond to *PolicyRule*, *ContextProductionDetails*, and *ContextConsumptionDetails* of the U-CoUDE information model in Fig. 1.

V. Concluding Remarks and Future Work

With the increase in the number of sources that provide various types of context data related to a user's environment, we are confronted with research challenges related to managing the data in a user-centric manner. In this paper, to meet the challenges, we proposed: 1) a technology-neutral context information model that is sufficiently flexible and extensible to represent heterogeneous contexts; and 2) a hierarchical context management architecture that can manage interactions among multiple contextual entities in a user-centric way.

In the future, we will develop a more concrete design as well as specific algorithms and methods for each module in the U-CoUDE architecture (Fig. 3). We will then implement prototype of the proposed context-aware services with real context sources. Obtaining and exploiting contexts from socially related users' contexts will be the most interesting and challenging aspect of our future research [4].

References

[1] J.-M. Kang, J. Strassner, S. Seo, and J. W.-K. Hong, "Autonomic personalized handover decisions for mobile services in heterogeneous wireless networks," *Computer Networks*, vol. 55, no. 7, pp. 1520–1532, May 2011.

[2] N. Roy, S. K. Das, and C. Julien, "Resource-optimized quality-assured ambiguous context mediation framework in pervasive environments," *IEEE Transactions on Mobile Computing*, vol. 11, no. 2, pp. 218–229, Feb. 2012.

[3] M. Baldauf, S. Dustdar, and F. Rosenberg, "A survey on context-aware systems," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 2, no. 4, pp. 263–277, 2007.

[4] P. Lukowicz, A. S. Pentland, and A. Ferscha, "From context awareness to socially aware computing," *IEEE Pervasive Computing*, vol. 11, no. 1, pp. 32–41, Jan. 2012.

[5] Y.-B. Kang and Y. Pisan, "A survey of major challenges and future directions for next generation pervasive computing," in *Proc. 21st International Symposium on Computer and Information Sciences (ISCIS '06)*, ser. LNCS, vol. 4263, Istanbul, Turkey, Nov. 1–3, 2006, pp. 775–764.

[6] M. J. van Sinderen, A. T. van Halteren, M. Wegdam, H. B. Meeuwissen, and E. H. Eertink, "Supporting context-aware mobile applications: An infrastructure approach," *IEEE Communications Magazine*, vol. 44, no. 9, pp. 96–104, Sep. 2006.

[7] F. Dobslaw, A. Larsson, T. Kanter, and J. Walters, "An object-oriented model in support of context-aware mobile applications," in *Proc. 3rd International ICST Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications (Mobilware '10)*, ser. LNICST, vol. 48, Chicago, USA, Jun. 30–Jul. 2, 2010, pp. 205–220.

[8] H. Chen, T. Finin, and A. Joshi, "An ontology for context-aware pervasive computing environments," *The Knowledge Engineering Review*, vol. 18, no. 3, pp. 197–207, Sep. 2003.

[9] P. Korpipää, J. Mäntyjärvi, J. Kela, H. Keränen, and E.-J. Malm, "Managing context information in mobile devices," *IEEE Pervasive Computing*, vol. 2, no. 3, pp. 42–51, 2003.

[10] T. Buchholz, A. Küpper, and M. Schiffers, "Quality of context: What it is and why we need it," in *Proc. 10th International Workshop on the HP OpenView University Association (HPOVUA '03)*, Geneva, Switzerland, Jul. 2003, pp. 1–14.

[11] T. Strang and C. Linnhoff-Popien, "A context modeling survey," in *Proc. 6th International Conference on Ubiquitous Computing (UbiComp '04)*, Nottingham, England, Sep. 7–10, 2004, pp. 1–8.

[12] A. K. Y. Wong, P. Ray, N. Parameswaran, and J. Strassner, "Ontology mapping for the interoperability problem in network management," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 10, pp. 2058–2068, Oct. 2005.

[13] C. E. Kaed, Y. Denneulin, and F.-G. Ottogalli, "Dynamic service adaptation for plug and play device interoperability," in *Proc. 7th International Conference on Network and Service Management (CNSM '11)*, Paris, France, Oct. 24–28, 2011.