Generic and easy-to-deploy middleware for USN proliferation

Marie Kim, and HoyChan Bang

305-700 Gajeong-dong, Youseong-gu, Daejeon,
mariekim@etri.re.kr, bangs@etri.re.kr

Abstract. one of big characteristics of ubiquitous sensor network is the heterogeneity of resources and applications. Different types of sensor networks' capabilities enable applications to exploit sensor networks in their own ways. Consequently it leads to non-standardized and application-specific implementations. There have been many researches and developments on different sensor network middleware which are specialized to certain purposes e.g., context-aware, dynamic application linking, event-driven, web service supporting, DBMS-like, etc. Since specialized sensor network middleware tends to be heavy and biased to the certain purpose, it is not easy to be used by different types of applications. This paper introduces a generic and easy-to-deploy ubiquitous sensor network middleware which provides very basic functions such as heterogeneity abstraction (interface, data schema), transaction processing, security and resource management. The LightWeight-COSMOS aims at being positioned as a middleware for worldwide ubiquitous sensor network applications and services therefore it contributes significantly towards ubiquitous sensor network industry proliferation in a real business world. The design principle of *genericity* and *easy-to-deploy* comes from the experiences that application markets need compact and robust middleware and strive to have business logic within their own scopes.

Keywords: middleware, generic, easy-to-deployment, ubiquitous sensor networks

1 Introduction

Ubiquitous sensor network *(USN)* is an embracive concept of sensor networks which include wireless sensor network (WSN), wired sensor network, Radio Frequency Identification (RFID), legacy system, actuator network, sensor and actuation network, etc. An operation to get data is *sensing* (or *reading)* while an operation to act is *actuation* (or *writing*, *responding)*. In this sense, USN can cover all information and control exchanging system.

There have been many researches on sensor network middleware from academic interests [1][2][3] to industrial interests[4][5][6]. In academic fields, they were interested in specific purpose middleware such as context-aware middleware [7], semantic sensor network middleware [8], distributed sensor network middleware [9], etc. These researches show the possibility of intelligent ubiquitous sensor network applications and services provision, however not yet realized in real-world. Meanwhile, OGS Sensor Web Enablement (SWE) framework [10] is adapted in many pilot projects and tested in real-world. Despite these advancements in middleware, the USN industry is still struggling in the market because there is not enough recognition and needs to replace existing systems.

Based on this analysis, it is quite clear that there is a need of a generic, easy-to-deploy and robust standardized USN middleware to boost up USN industry in a cost effective way. This paper introduces LightWeight-COSMOS which is a refined version of USN middleware from COSMOS [11]. COSMOS is an intelligent USN middleware which provides not only basic functions such as query processing, sensor network abstraction, security and resource management, but also intelligent data processing functions such as event processing, context-aware processing, data mining, service brokering, etc.

2. Lightweight-COSMOS

LightWeight-COSMOS has been developed by ETRI funded by Korean government since 2011 as a part of the project COMUS (Common Open seMantic USN). The COMUS is an ongoing project and plans to provide easy-to-install, easy-to-use, easy-to-develop and easy-to-access environment for sensor network developers and USN application developers to boost USN industry in Korea. To pursue this goal, COMUS provides semantic resource discovery, semantic data processing and dynamic USN resource registration with standardized interface and data schema.

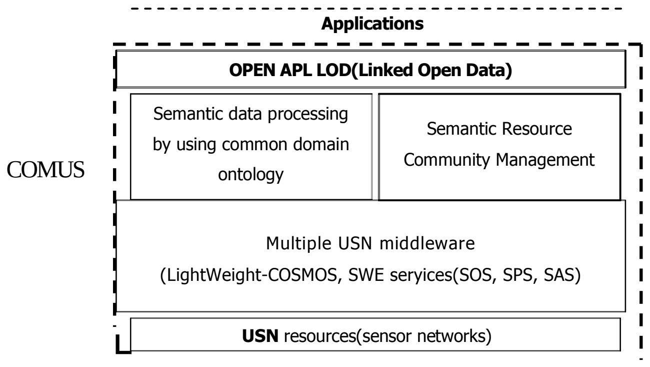
Applications
OPEN APL LOD(Linked Open Data)
Semantic data processing by using common domain ontology
Semantic Resource Community Management
COMUS
Multiple USN middleware (LightWeight-COSMOS, SWE seryices(SOS, SPS, SAS)
USN resources(sensor networks)

Figure 1 COMUS Architecture

As a part of COMUS, LightWeight-COSMOS plays a role of data collector and control transmitter between upper layer and heterogeneous USN resources. Intelligent information processing is provided by an upper layer and LightWeight-COSMOS focuses on basic middleware functions. On the other hand, LightWeight-COSMOS also runs independently from other parts of COMUS. Any application can use LightWeight-COSMOS directly without use of COMUS upper layer.

2.1 Architecture

Following is the architecture of LightWeight-COSMOS, given in Figure 2. It consists of Platform Manager, Connection Manager, Data Translator, User Session Manager, Sensor Data Repository, Governance Manager, Query Manager, Request Manager, Sensing Data Manager, Monitoring Manager, Resource Manager, and Communication Manager. Platform Manager processes user profile management, retrieves LocalRegistry, and configures platform. Connection Manager processes requests from users and responses from middleware. Data Translator is for format-converting from COSMOS-formatted USN resource description and data into SensorML and O&M. Query Manager processes sensing data queries and Request Manager processes requests from applications and sensor networks. Monitoring Manager monitors the status of USN resources, and if not-normal USN resource is detected, Request Manager will try to reset USN resource to return it into normal status. Resource Manager manages USN resources and these USN resources' information is used by Query Manager to validate and process the input queries. USN resource registers at LightWeight-COSMOS anytime when it is ready, and registration information (description of USN resource) is kept at LocalRegistry. Communication Manager handles connections between middleware and USN resources. User Session Manager authenticates applications to prevent unauthorized access to middleware.

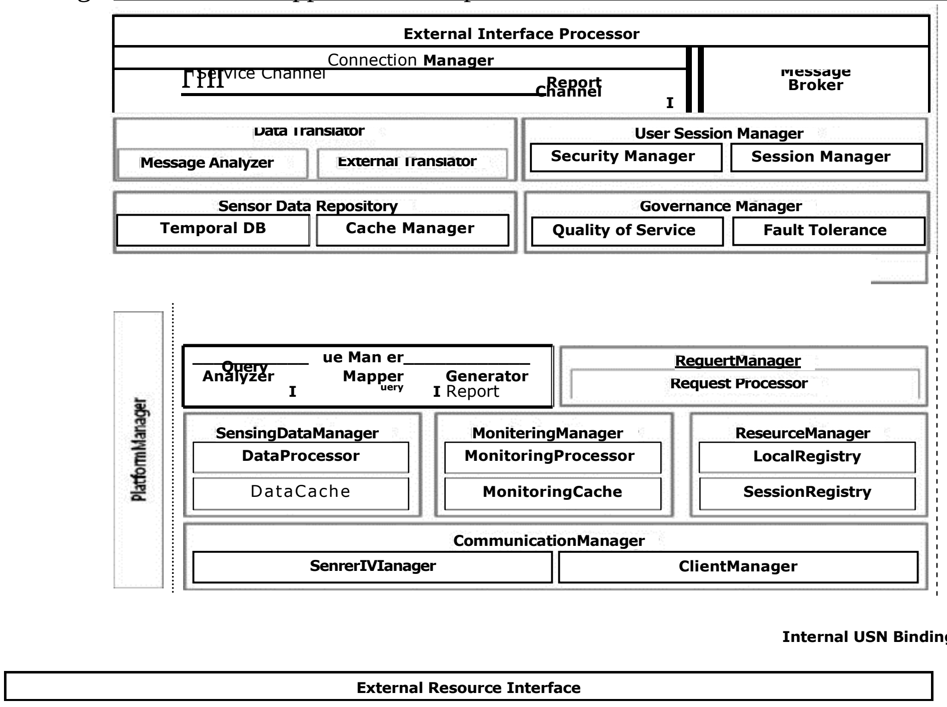

Figure 2 LightWeight-COSMOS Architecture

2.2 Characteristics

First, LightWeight-COSMOS provides only basic middleware functions such as transaction processing, interfacing between applications and heterogeneous sensor networks. Intelligent data processing is within applications' scope.

Second, LightWeight-COSMOS provides dynamic USN resource management. Contrasting to COSMOS, any USN resource connects to LightWeight-COSMOS any time when it is ready to operate. Whenever USN resource doesn't want to provide services to LightWeight-COSMOS, then it deregisters easily. In addition, if USN resource's status is changed, it is reported by an USN resource immediately.

Table 1 shows interfaces related with USN resource registration information.

Table 1 USN resource management interface

category	Interface	explanation
USN resource registration	registerResourceReq	Registration request (R—>A4)
	registerResourceRes	Registration response (R—M)
	deRegisterResourceReq	Deregistration request(R—>M)
	deRegisterResourceRes	Deregistration response(R<—M)
USN resource update	updateResourceReq	Update request(R—*M)
	updateResourceRes	Update response (R—M)

Third, mapping relation between middleware and USN resource is M:N. One LightWeight-COSMOS can use several USN resources and one USN resource can serve to several LightWeight-COSMOS or other types of middleware. This mapping is based on that sensor network providers can pursue the benefit-maximization by providing sensing data to multiple different middleware providers. Therefore, LightWeight-COSMOS can't keep the load of each USN resource. It results in that any request from applications is delivered to the USN resource whenever the USN resource's capability (description) is fit to the request. For example, in case where, a USN resource can handle only three requests at a time, the USN resource should return a reject message with appropriate status code when it receives 4th request. In contrast, COSMOS doesn't deliver a request to an USN resource if the load of USN resource is full already.

Fourth, USN resources can be addressed as a gateway level, as a sensor node level or as a transducer level. Applications can use only sensor node identifiers or transducer identifiers without higher level of identifiers. This is because identification scheme includes hierarchical information within it: *gatenodeithpanid:nodeid;transducerid.* Whereas, when COSMOS applications need to address a specific transducer in a query, they should input sensor network identifier (gateway identifier), sensor node identifier and transducer identifier, altogether.

Fifth, LightWeight-COSMOS doesn't keep sensing data within its scope. That is because; most sensing data are sensitive for business perspective and middleware is a just intermediate node in whole business architecture. SensingDB in LightWeight-COSMOS is to provide sensing data reliably. LightWeight-COSMOS stores sensing data collected during accidental application's failure and it delivers those data when the application is reconnected. Especially, in medical fields, it is required to deliver whole sensed data even if there is temporary application failure.

Sixth, in contrast to COSMOS, LightWeight-COSMOS doesn't keep connection between USN resources and middleware. COSMOS provides connection interfaces such as connReqCtrl and ConnResCtrl to start a connection and DisConnReqCtrl to end the connection. But keeping connection between USN resource and middleware is not necessarily required. In some cases, keeping connection is a just unnecessary burden to both USN resource and middleware. Therefore, LightWeight-COSMOS is designed not to hold the session but to establish the session when it is needed. But if data exchange rate is high, then LightWeight-COSMOS provides a way to keep connection by using HTTP header. If *"Connection"* field is set to *"close"*, then connection is not kept, otherwise, connection is kept.

3.3 Development

LightWeight-COSMOS runs as a web application of Tomcat for easy deployment and easy running. And LightWeight-COSMOS is distributed as an Eclipse project so within Eclipse it can be run easily. LightWeight-COSMOS communicates with clients via RESTfull API and communicates with USN resources via xml over HTTP MTB (Message Transport Binding).

LightWeight-COSMOS is tested with USN resource simulator and real sensor network Gateway. Following is the test environment. The USN resource simulator operates as like one sensor network which consists of one gateway, multiple pans, multiple sensor nodes and multiple transducers (sensor or actuator). In Figure 3, middleware client(MW client) runs on a laptop, LightWeight-COSMOS runs on a separate desktop, 4 simulators run on a separate desktop and one gateway operates as a separate hardware module.

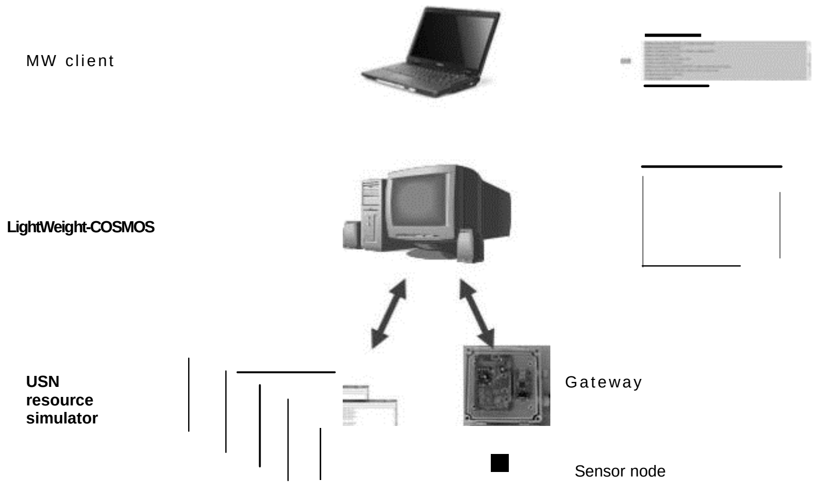

Figure 3 LightWeight-COSMOS test environment

4. Further Works

There are some issues which need to be considered more deeply. First, there is not enough consideration on the powerful mobile sensor node. If a mobile sensor node can communicate with middleware by itself, then it does not need a gateway to interpret interface between sensor node and middleware. Nowadays there are many approaches to exploit smartphones as a sensor node such as participatory sensing. Therefore, support on powerful sensor nodes is needed. Second, granularity of interface issue is not easy. If the interface is designed too generically, then utilizing different sensor nodes' capabilities are impossible. Meanwhile, if the interface is designed too much detailed, then there should be too many interface and some are not used by most applications. The interface between middleware and sensor networks was proposed as a New Proposal to ISO/IEC JTC1 and it starts as ISO/IEC 30128(Generic Sensor Network Application Interface). Lastly, security issue needs to be studied more deeply. When it comes to USN or IoT, the data are very sensitive from people's privacy point of view or from business interest's point of view. Therefore, security consideration is not optional but essential.

References

1. K. Henricksen, and R.Robinson, "A Survey of Middleware for Sensor Networks:StateoftheArt and Future Directions," MidSens'06, November 27December 1, 2006
2. M. M. Molla and S I Ahamed, "A survey of middleware for sensor network and challenges," in Proceedings of the International Conference on Parallel Processing Workshops, pp. 223-228, 2006.
3. Karen Henricksen, Ricky Robinson. "A survey of middleware for sensor networks: State-of-the-art and future directions," In Proceedings of the Int. Workshop on Middleware for Sensor Networks Table of Contents, Melbourne, Australia, 2006, pp.60 {65.
4. OGC Implementation Specification 06-009r6: OpenGIS Sensor Observation Service (SOS); Open Geospatial Consortium: Wayland, MA, USA, 2007.
5. OGC Implementation Specification 07-014r3: OpenGIS Sensor Planning Service; Open Geospatial Consortium: Wayland, MA, USA, 2007.
6. OGC Best Practices 06-028r3: OGC Sensor Alert Service Candidate Implementation Specification; Open Geospatial Consortium: Wayland, MA, USA, 2006.
7. Viterbo, J.; Sacramento, V.; Rocha, R.; Baptista, G.; Malcher, M.; Endler, M, "A Middleware Architecture for Context-Aware and Location-Based Mobile Applications," Software Engineering Workshop, 2008, pp.52-61.
8. Huang, v., Javed, M.K., "Semantic Sensor Information Description and Processing," Sensor Technologies and Applications, 2008, pp.456-461.
9. T. Abdelzaher , et al., "EnviroTrack: Towards an Environmental Computing Paradigm for Distributed Sensor Networks," Proc. 24th Int'l Conf. Distributed Computing Systems (ICDCS 04), IEEE CS Press, 2004, pp. 582-589.
10. http://www.opengeospatial.org/projects/groups/sensorwebdwg
11. M. Kim, J.W. Lee, Y.J. Lee and J.C. Ryou, "COSMOS: A Middleware for Integrated Processing over Heterogeneous Sensor Networks," ETRI Journal, Vol.30. 2008, pp. 696-706.