Complex Event Processing in EPC Sensor Network Middleware for Both RFID and WSN

Weixin Wang, Jongwoo Sung, Daeyoung Kim
Auto-ID Lab Korea
Information and Communication University
{weixin.wang, jwsung, kimd}@icu.ac.kr

Abstract

In an integration system of RFID and wireless sensor network (WSN), RFID is used to identify objects while WSN can provide context environment information of these objects. Thus, it increases system intelligent in pervasive computing. We propose the EPC Sensor Network (ESN) architecture as an integration system of RFID and WSN. This ESN architecture is based on EPCglobal architecture, the de facto international standard for RFID. The core of ESN is the middleware part which is also implemented in our work. In this paper, complex event processing (CEP) technology is used in our ESN middleware which can handle large volume of events from distributed RFID and sensor readers in real time. Through filtering, grouping, aggregating and constructing complex event, ESN middleware provides a more meaningful report for the clients and increases system automation.

1. Introduction

Today's pervasive computing widely uses radio frequency identification (RFID) and wireless sensor network (WSN). In RFID technology, a unique ID EPC (Electronic Product Code) is assigned to a RFID tag which sticks to a real world object. With the help of this technology, real world objects can be easily mapped into the virtual world in an information system. RFID applications can be found in many areas such as logistics, industry automation, healthcare and security [1]. As for WSN technology, in which hundreds to thousands sensor nodes form one wireless sensor network, sensors sense the physical environment and transfer the sensing data back to the base station by multi-hops. A wide range of WSN applications were also developed, such as environment monitoring, military applications, healthcare and object tracking.

However, these two technologies are actually not so separated from each other, and more importantly the integration of RFID and WSN will bring us many advantages in future pervasive computing. On one hand, the current information system needs to be extended to support active tags, which can be implemented by sensor nodes. On the other hand, there is lack of standards for different sensor networks to share their data with each other. The existing EPCglobal architecture [2] for RFID can be used to process and share information from sensor nodes [3]. The integration of RFID and WSN can both track objects in the world and provide the context environment information to the objects. Hence, the reality and automation of an information system can be increased.

The EPC Sensor Network (ESN) architecture [3], an extension of the RFID EPCglobal architecture, is a global architecture for capturing, filtering and sharing both RFID and sensor data. In this paper we provide a design of middleware for ESN architecture. The middleware in this architecture plays an import role. It collects RFID and sensor data from distributed readers and processes this large volume of data in real time. After filtering, grouping and aggregating, well-formatted data reports are sent to the upper layer. In the implementation of our middleware, we adopted complex event processing (CEP) technology [4], which can discover the information among multiple events. CEP is proved to be a powerful tool in describing relationship between different events such as timing, causality, and membership in a real-time stream of data or events. As a result, it enhances the meaning of data report to upper layer and increases the automation of the system.

The rest of this paper is organized as follows: section 2 provides a literature review on the related works. In section 3, ESN architecture is briefly introduced. Section 4 discusses the events in ESN middleware. Later, section 5 describes the architecture of ESN middleware in detail. Section 6 describes our implementation, and section 7 concludes the paper.

2. Related works

ESN middleware relates with many current work, such as RFID and WSN middleware.

In RFID field, the Savant middleware [15] is an early successful implementation of EPC network. Currently, several of major IT companies already provide commercial RFID software, such as SUN EPC Network [20] and IBM WebSphere RFID Premises Server [21]. More recently, CEP technology is used in several RFID middleware systems [5-8]. In paper [5] Event Processing Language was used to define complex events while in paper [6] relationship set which origins from active database was adopted to define complex events. In this paper, we also follow relationship set way.

As for WSN part, several sensor data collecting and sharing architecture already stand out [22-24]. Global Sensor Network (GSN) [22] middleware that provides virtual sensor abstraction and powerful query tools makes the access to the heterogeneous wireless sensor nodes easier. Hifi architecture [9] is a hierarchical architecture to process distributed data from RFID and sensor network. It includes many components, such as data listener, data stream processor, data disseminator, resource manger, query listener etc. This paper proposes a novel approach. Instead of building our architecture from scratch, ESN middleware design is based on existing well-developed RFID standard. It leads to the framework fit for RFID and WSN integration applications.

3. EPC sensor network architecture

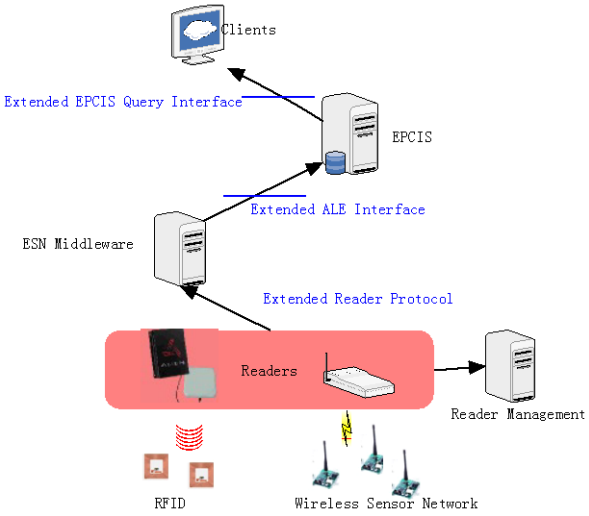

Figure 1. ESN architecture

ESN architecture, shown in Figure 1, is extended from EPCglobal architecture and supports standard RFID, WSN and their integration.

The reader layer includes both a RFID reader and a sensor reader (base station). In RFID, a reader reads tags in a non-line-of-sight way while in WSN, sensor data are sent to readers by different types of protocols, such as Zigbee [8] and IP-USN [10]. Reader management servers are responsible to monitor the health of the readers and manage reader functions. The middleware manipulates large volume of data from multiple readers. Then those data are filtered, grouped and reported to EPCIS (EPC Information Services). RFID and sensor data, together with their business logic are stored in the EPCIS repository. Finally, applications can query data in EPCIS. To meet this goal, we also extend the standard interface among the parts of the system.

4. Events in the ESN middleware

4.1 RFID events and sensor event

In ESN middleware, two kinds of event objects exist, RFID event object and sensor event object. As is described in [11] and [12], RFID event object is defined as a tuple (ID, L, T) where ID represents EPC code of the RFID tag, L location and T time stamp. The sensor event object, which is more complex than RFID, is defined in a hierarchical way. The first layer is still a tuple (ID, L, T, D), where ID is the identification of a sensor node, L location, T time stamp, and D sensor data. The ID includes both the reader (base station) ID and the sensor node ID. To achieve the ID uniqueness property, EPC codes of the reader and the sensor node can be used as their IDs. L is location; T is time stamp and D is sensor data. While in the second layer of event object in D, a tuple of sensing type such as (humidity, temperature, pressure) can be included.

4.2 Simple and complex events

Simple event is the above RFID or sensor event with constrain. For example, a RFID event happens in the test lab is a simple RFID event.

S_1 = (ID, L{L="test lab"}, T)

And the temperature of a sensor event is above 40 is also a simple sensor event.

S_2 = (ID, L, T, D{D.temperature.value>40})

Complex event is a combination of simple events or complex events with the following set [13]:

AND(∧): $E_1 \wedge E_2$ represents two events, where E1 and E2 occur together.

OR(∨): $E_1 \vee E_2$ means either E_1or E_2 occurs.

NOT(!): $!E_1$ means that E_1 does not happen.

SEQ(→): $E_1 \rightarrow E_2$ means that E_1 is followed by E_2.

Relative periodic (R_p): $R_p(E_1,E_2,E_3)$ means that E_2 occurs between E_1 and E_3, possibly several times.

4.3 Rule

The rule, the description of method to process events, includes three parts: events, conditions and actions. To get better understanding on the rule, we illustrate the rule using an example of a warehouse application (figure 2).

In a warehouse application, in which some products needs to be stored in cold temperature (below 0 centigrade), sensor nodes are deployed both in the Truck A and Warehouse B to detect the surrounding temperature. RFID readers C and D are located at the doors of warehouse and truck separately. A report of successful delivery requires the following conditions: 1) temperature in the truck below 0 degree centigrade; 2) temperature in the warehouse below 0 degree centigrade; 3) product delivery time from truck to warehouse less than 5 minutes.

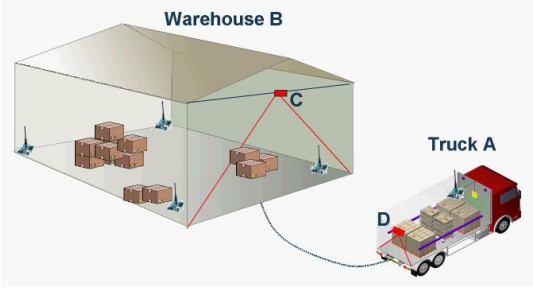

Figure 2. Warehouse Rule Example

Warehouse rule definition:

Events:

S1 = (ID, L{L="Truck A"}, T, D{D.temperature.value<0}) (1)

S2 = (ID, L{L="Warehouse B"}, T, D{D.temperature.value<0}) (2)

S3 = (ID, L{L="reader D"}, T) (3)

S4 = (ID, L{L="reader C"}, T) (4)

C1 = (S3 →S4) ∧S1 ∧S2 (5)

Conditions:

S3 .ID = S4 .ID (6)

S4 .timeStamp - S3 .timeStamp < 5min (7)

Actions:

Generating qualified product entrance report to upper layer.

In this example rule, (1) and (2) are simple sensor events; (3) and (4) are simple RFID events. The event in (5) is a complex event which simultaneously satisfies events S_1, S_2, S_3 and S_4, while at the same time, S_3 should happen before S_4. The condition in (6) represents the same product passes through both reader D and reader C. The condition in (7) defines the moving time to be less than 5 minutes. If all the above events and conditions are satisfied, the middleware system will generate a successful delivery report to the upper layer.

4.4 Event granularity

In addition to content differences between RFID and sensor events, the event granularity is different too [14]. In general, the sampling time of a RFID reader is about every several milliseconds. While in sensor networks, the reading time of sensor data might be several orders different among different applications. However, in spite of the wide range in sensor reading time, few reading frequencies can be as comparable as RFID applications because of the power and bandwidth limitations in sensor network.

To solve this problem, the life time of sensor events is extended. A new variable defined as duration is added in the sensor data. This duration variable refers to the effective life time of sensor data. Thus in the sensor value is always held as last sensing data during sensor reading interval.

5. ESN middleware architecture

ESN middleware, which is based on event-driven architecture, uses complex event processing to meet the needs of advanced users. It is able to trigger more sophisticated reports and enhance automation and efficiency in ubiquitous environment. Furthermore, ESN follows the standard of RFID, thus there is no problem to connect with existing RFID systems. By extending the RFID standard, our middleware can support RFID, sensor network and the interaction between them. The middleware design is illustrated in figure 3. As is shown, four basic components are included in ESN middleware: event handler, event database, event processing engine and event action unit. The event handler collects events from different kinds of event resources (RFID and sensor readers). The event database temporarily stores events which are not processed immediately. In the event processing engine, events are filtered, grouped, aggregated and finally formed to be complex events. The last part, the event action unit, generates actions according to the received events.

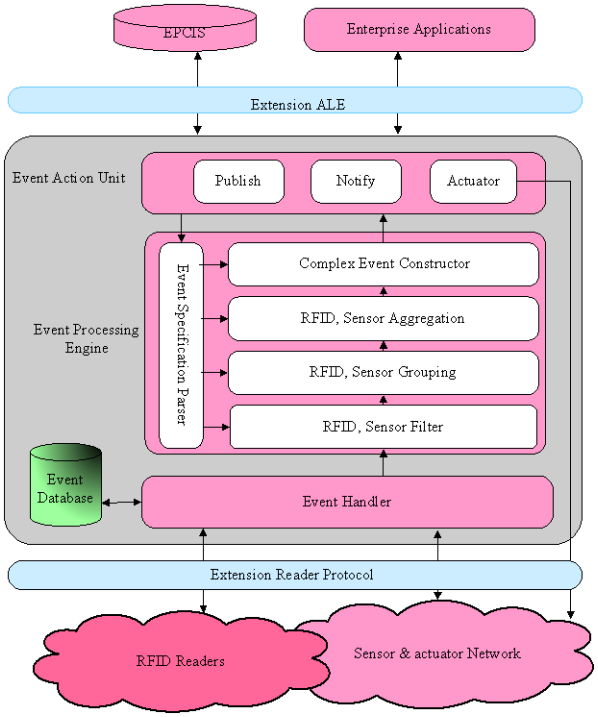

Figure 3. ESN middleware architecture

5.1 Event handler

Events from distributed RFID and sensor readers are sent to event handler first. In event handler, an event queue is adopted to manage the processing sequence of received events. In addition, the event handler can also send events to temporary event database according to event duration.

5.2 Event database

In our middleware, not all the events will be processed immediately. Thus a temporarily storage mechanism for events is necessary to solve the incompatibility problem between RFID and sensor events. Moreover, the operations like aggregation need a collection of events for a period of time. For example, instead of sending the temperature data of every hour to the customer who needs only maximum and average temperature in one day, the event of every hour can be temperately stored in event database, and then they will be processed together in event processing engine.

5.3 Event processing engine

The event processing engine is the kernel of ESN middleware. It filters out uninteresting data, formats the remaining useful data and constructs complex events according to specifications in real-time.

The event specification parser interprets and transforms event specifications into four processing steps: filtering, grouping, aggregation and complex event construction.

The volume of event data is very large in the ESN middleware system. The filter selects only those interested events to the upper layer, thus reducing the report data dramatically.

In the report to upper layer, event data are separated into several groups for clear demonstration.

Aggregation provides statistic information among event data. Besides the "count" operation described in RFID standard [18], we have "SUM, MIN, MAX, AVG" for sensor event data. By aggregating, the volume of reported data can be reduced again.

Later, simple events are generated to a complex event as described in the section 4. Complex events provide more meaningful reports and enhance the system automation.

5.4 Event action unit

The event action unit connects to the upper layer EPCIS or other enterprise applications. It describes when a complex event will be generated and what actions will be performed. It can also send report to EPCIS by subscribing/publishing, notifying enterprise applications and triggering actuators in the sensor network.

6. Implementation

ESN middleware is a part of our ongoing EPC sensor network project. The basic extensions of ALE and reader protocol have been defined in this work. In event handler, an asynchronous event queue handles events from distributed readers. In event database, a real time in-memory event database technique which originates from Savant [15] improves real time performance. A part of Accada [16], an open source project of RFID EPC network, acts as part of our RFID event processing. The complex event constructor is implemented as described in section 4. Finally, for EPCIS or enterprise applications to get event reports in the publish/subscribe way, Axis [17] and Tomcat [19] are used to provide web service.

Figure 4. Smart shelf application

A demo, a smart shelf application (figure 4), was built to test of our ESN middleware. Bread and cookies with RFID tags were placed on the shelf to be read by an Alien reader nearby. Meanwhile, a sensor node equipped with an Atmelga 128L microcontroller, a CC2420 RF transceiver, a temperature sensor and a humidity sensor senses the temperature and humidity of the smart shelf environment. Sensor nodes in different shelves compose a WSN and transfer data to reader using Zigbee.

As an example, one of the complex events in the above application is generated when there is bread on the shelf and temperature is above 25 degrees centigrade. For this complex event, the corresponding event rules are as follows:

S_1 = (ID{ID= 'bread ID'}, L{L="reader A"}, T)
S_2 = (ID, L{L="shelf A"}, T,
D{D.temperature.value>25})
$C_1 = S_1 \wedge S_2$

When this complex event is generated, EPCIS gets a report. Figure 5 shows the middleware test client which interacts with ESN middleware.

7. Conclusion

In this paper, we introduced the use of CEP technology in the middleware of ESN, an integrated architecture between RFID and WSN. The events of RFID, WSN and their interaction were also analyzed. By adopting CEP technology, we built a middleware system that has the functions of filtering, grouping and aggregating event data in real-time. Additionally, this ESN middleware can construct complex events from simple events according to their timing, causality, and membership relationship, thus catching complex relationship in the real world and providing more meaningful report to its clients. Finally, we build a prototype to demonstrate the functionality of our proposed middleware in real business system.

Acknowledgement

This work was supported by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korea government (MOST) (No. R0A-2007-000 10038-0)"

References

[1] RFID journal, http://www.rfidjournal.com/
[2] EPCglobal, http://www.epcglobalinc.org/home
[3] Jongwoo Sung, Tomas L. Sanchez, Daeyoung Kim, "EPC Sensor Network for RFID and USN Integrated Infrastructure", Percom 2007.
[4]Complex event processing, www.complexevents.com
[5] Liang Dong, Dong Wang, Huanye Sheng, "Design of RFID Middleware Based on Complex Event Processing", IEEE conference on Cybernetics and Intelligent Systems, 2006.
[6] Kaushik Dutta, Krithi Ramamritham, Kamlesh Laddhad, Karthik B. "Real-Time Event Handling in an RFID Middleware System" Workshop on Databases in Networked Information Systems (DNIS) 2007, Japan
[7] Fusheng Wang, Shaorong Liu, Peiya Liu, Yijian Bai "Bridging physical and virtual worlds: complex event processing for RFID data streams" In EDBT, 2006.
[8] Zigbee alliance. http://www.zigbee.org/.
[9] Michael J.Franklin, et al, "Design considerations for high fan-in systems: the HiFi approach" CIDR 2005.
[10] http://www.ietf.org/html.charters/6lowpan-charter.html
[11] N.W.Paton, D´ıaz, "Active database systems" ACM Comput. Surv., 31(1):63-103,1999
[12] H.Gonzalez, J. Han, X. Li, and D. Klabjan, "Warehousing and analyzing massive rfid data sets". In ICDE'06
[13] S. Chakravarthy. Sentinel: an object-oriented DBMS with event-based rules. In SIGMOD '97: Proceedings of the 1997 ACM SIGMOD international conference on Management of data, pages 572–575, New York, NY, USA, 1997. ACM Press.
[14] Shawn R. Jeffery, Michael J. Franklin, Minos Garofalakis "An Adaptive RFID Middleware for Supporting Metaphysical Data Independence", VLDB 07, September 23-27 2007, Austria
[15] Oat Systems and MIT Auto-ID Center, "The Savant version 0.1 technical report".
[16] Accada project, http://www.accada.org/
[17] Axis, http://ws.apache.org/axis/
[18] EPCglobal Application Level Events (ALE) Specification Version 1.0, September 15,2005.
[19] Tomcat, http://tomcat.apache.org
[20] A. Gupta and M. Srivastava, "Developing Auto-ID solutions using Sun Java system RFID soltuion".
[21] WebSphere RFID Premises Server. http://www-306.ibm.com/software/integration/ws_rfid_premises_server/
[22] Karl Aberer, Manfred Hauswirth, Ali Salehi, "Infrastructure for data processing in large-scale interconnected sensor networks", MDM 2007
[23] P. B. Gibbons, et al , "IrisNet: An Architecture for a World-Wide Sensor Web", IEEE Pervasive Computing, 2(4), 2003.
[24] M. Sgroi, et al, "A service-based universal application interface for ad hoc wireless sensor and actuator networks". In Ambient Intelligence, 2005