An Efficient Method to Create Business Level Events using Complex Event Processing based on RFID Standards*

Byung-Kook Son, Jun-Hwan Lee, Kyung-Lang Park, Cheong-Ghil Kim, Hie-Cheol Kim[1], and Shin-Dug Kim

Department of Computer Science, Yonsei University,
Seoul, Korea
{ssonkk, jhlee, lanx}@parallel.yonsei.ac.kr,
{Tetons, sdkim}@yonsei.ac.kr
[1]School of Computer and Communication Engineering, Daegu University,
Daegu, Korea
hckim@daegu.ac.kr

Abstract. RFID systems should be designed to process a large number of RFID data in real time. Therefore, there have been much research and company studies regarding RFID data processing. One of methods is CEP (Complex Event Processing), which can provide a method to process RFID data efficiently. However, previous work is just focused on raw RFID data processing, such as data filtering, the elimination of duplicated data, and the aggregation of data. Also, it creates primitive events based on just one physical or logical reader. Therefore, processing overhead for complex events may increase. And it cannot provide business level events. Therefore, we propose a method that can reduce processing overhead and create business level events by using CEP. Proposed method provides two primitive events that are defined by using the relationship of two readers. Thus, when any pattern of events is matched for a specific complex event, business level events can be generated and execution time can be reduced comparing with other mechanisms without those events. And, execution time can be reduced by about 57% as compared to others.

1 Introduction

RFID (Radio Frequency Identification) systems are constructed as four major components, i.e., the tags with unique ID, the RFID readers that identify any tags by using RF frequency, the middleware that can process raw RFID data, and the applications. Recently, EPCglobal [1] leads to standards for the components of RFID systems such as, tag data, air interface for communication, RFID middleware, and so on.

RFID systems should be designed to process many data in real time. Those data is processed by the RFID middleware.

*This research was supported by the UPLUS project of Ministry of Commerce, Industry and Energy.

Therefore, the role of RFID middleware is important in RFID systems. A standard of RFID middleware is proposed by EPCglobal as ALE (Application Level Event) [2]. However, RFID middleware based on ALE just provides pure RFID data. As a result, RFID applications have to include some logics to process business events. Although EPCglobal proposes the EPCIS (Electronic Product Code Information Service) [3] to provide business level events, that is incomplete standard that does not define many specific parts. And to apply the EPCIS in RFID systems, agreement is required for partners.

Therefore, we propose a CEP based on ALE to provide business level events. Especially, primitive event types in our proposed method are defined based on relationship of two readers. That is, most of business events occur when a certain tag moves between any two or more readers. For example, we can assume a situation that a customer pays for some items in a counter with RFID reader after taking out those from a shelf with RFID reader in the market. In this situation, when a customer takes out an item from the shelf, a business event for "SELL" can be started. And when a customer pays for an item at the counter, a business event for "SELL" can be completed. Our proposed event model can be expressed as "RelativeObservation" by considering relationship of two readers. And that event model can specify the starting of business level event (take out an item from the shelf) as "PredictedObservation". Thus, when any pattern of events matched for complex event to generate business level events, execution time can be reduced comparing with other mechanisms without those events and many effective business level events can be provided. When we use our method, execution time can be reduced by about 57% comparing with other mechanisms.

2 Related Works

Event processing technology has been studied in active database [4 - 6] for long time. Especially, it has been studied to process real-time stream data including sensor and RFID data. Also it is important to improve the performance of RFID systems. Therefore, in [7], several methods for RFID data management are introduced. In [9], temporal-based data model and rule-based RFID data transformation are introduced for RFID data processing. However, it does not provide any simple query to get high level events. And in [9], to get high level information, complex query has to be used, because it is not suited for handing RFID events. Therefore, CEP is introduced to process RFID data efficiently. Especially, in [10], to process RFID data, CEP is considered in RFID systems. However, it does not provide any information about detailed architecture of CEP. In [8], CEP is also applied to process RFID data. In that work, components of CEP are discussed including primitive event types, event operators, RFID rules, and complex event detection algorithm. It also introduces the algorithm that can detect non-spontaneous events. However, it is only focused on raw RFID data processing, such as the elimination of duplicated data, and the aggregation of data.

3 Proposed Complex Event Processing

Basically, we can consider two kinds of events in CEP. One is primitive event and another is complex event. Primitive events are basically composed to create complex events. Also, complex event can be created when one or more primitives are matched with any special pattern or when one or more complex events show any special pattern.

3.1 Primitive Event

In RFID systems, previous CEP techniques are focused on raw RFID data processing in the RFID middleware, elimination of duplicated data, and aggregation of data. For that reason, a primitive event in previous work [8] is defined based on the tag identified by just one reader as physical or logical. However, we propose primitive events based on the tags identified between two readers.

Observation(r, o, t). Observation event type occurs when a reader identifies a tag. This primitive event type is equal to that in previous work. However, our event type can be considered unique RFID tag data because based on ALE. In the format of Observation(r, o, t), where r represents logical reader, o represents the EPC of object, t represents timestamp when the object is identified by reader.

PredictedObservation(r, o, t). The PredictedObservation event occurs when a tag disappears from any reader. That is, a tag disappeared from any reader may appear again to other readers some time. In the format of PredictedObservation(r, o, t), r represents a logical reader, the o represents the EPC of object, t represents timestamp when this object disappears from the reader.

RelativeObservation(o, r1, t1, r2, t2). RelativeObservation event occurs when a tag disappeared from any reader is identified by another reader. RelativeObservation means that an object is currently identified by r2 after it is identified by r1 first. In the format of RelativeObservation(o, r1, t1, r2, t2), o represents the EPC of object, r1, r2 represents logical reader, t1 represents relative time(after/before) between two readers, and t2 represents timestamp when this object is identified by reader2.

3.2 Event Operators

As mentioned above, a complex event can be created through detecting any specific pattern of primitive events or complex events by using event operators. In CEP, the role of event operators is to define any pattern of complex events. Event operators can be classified into temporal and non-temporal operators [8]. However in this paper, the definition of temporal operator is not required, because we cannot predicate the time when any business event occurs.

So, we have to define new operators for creating the business level events. These are causal operators that can express any casual relationship. In the market, if we assume “SELL” event by considering only events occurred at the counter reader, this may cause any misleading situation. Because, when a worker takes an item to the counter, this case may predicate a “SELL” event. Therefore, to predicate a correct business event, we have to check any causing event activated first for this business event. After then, we can predicate correct event by checking the result activated. For that reason, causal operators and the relationship of events are important for creating business level event.

3.2.1 Proposed operator for Business Level Event

Table 1 shows operators for creating business level events in this proposed method.

Table 1. Event operator for complex event processing.

Operator	Meaning	Usage
AND(∧)	Conjunction of two events E1 and E2 occur when both E1 and E2 occur without occurrence order	(E1∧E2)
OR(∨)	Disjunction of two events E1 or E2 occur when either E1 or E2 occur without occurrence order	(E1∨E2)
NOT(¬)	Negation of E1 event occur when E1 never occur	(¬E1)
SEQ (;)	Sequence of two events E1 and E2 occur when both E1 and E2 occur with occurrence order	(E1;E2)
CASUAL(→)	Cause of two event E1 and E2 occur when E1 is cause E2	(E1→E2)

3.3 Event Rule

We have to detect any complex event by using any given event pattern.

```
ON PredictedObservation(r, o, t)
IF Condition
        DEFINE RelativeObservation(o, r1, t1, r2, t2)
        ON Event
        IF Condition
                DO Action
```

Fig. 1. Event rule definition for complex event

So a method that detects the complex event is required. To detect any complex event efficiently, a rule should be defined. Traditionally, rule definition to composite events has been studied based on ECA (Event Condition Action) model. ECA model can used to detect any event easily and simply. Therefore, we use modified ECA model to express PredictedObservation and RelativeObservation as shown in Figure 1.

4 CEP Architecture and Operation Flow

Until now, we explain about the components of CEP. In this section, the architecture to process complex events will be presented. And we will explain the operation flow of architecture to create complex events.

4.1 Architecture and Operation Flow

Figure 2 shows the proposed architecture for CEP. Proposed method is based on ALE. Application will request monitoring about business events defined in section 3.3 according to rule definition. Query analyzer receives application requests and analyzes them. And query analyzer sends those to the rule matcher. Rule matcher tries to search its rule repository to check whether same rule exist or not. If there is no match, rule matcher needs to register that in the rule repository.

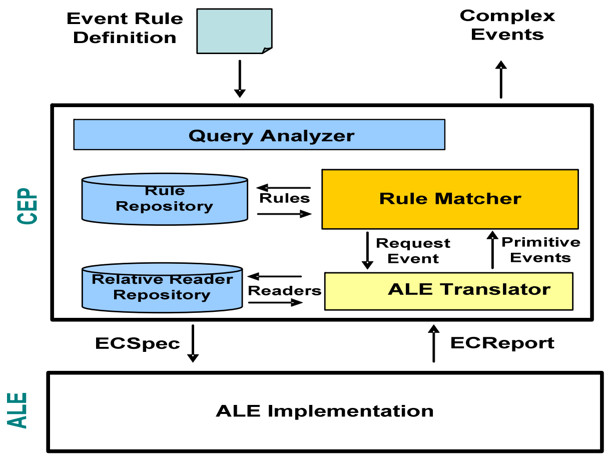


Fig. 2. Proposed architecture of complex event processing.

After that, the rule matcher sends a message to ALE translator. The ALE translator makes the ECSpec to creating primitive event. And the ALE translator requests ECReport message by the ECSpec through ALE API. ALE translator waits for the ECReport. If ALE translator receives the ECReport from the ALE, it may generate primitive events based on ECReport. ALE translator may search the relative reader repository for checking relationship of reader registered to crate the RelativeObservation event. If the incoming event is not related with the registered reader, ALE translator may generate the Observation or PredictionObservation events depending on whether the tag is disappeared from the readers or not. The rule matcher checks whether primitive events generated by ALE translator matches with any complex event pattern or not. If primitive events match with any complex event pattern, the rule matcher may generate complex events.

5 Examples of Complex Event Detection

In this chapter, an example that creates the business level events by using primitive events, event operators, and event rules is shown. We assume two scenarios. One is that a customer is shopping as shown in Figure 3. Another one is that an object is moving along fixed path as shown in Figure 6.

5.1 Scenario 1 (Shopping in the Market)

Figure 3 shows shopping scenario in the market. A customer with shopping cart enters the gateway with RFID reader. When he passes the gateway, the RFID reader identifies the tag on shopping cart. After that, he is walking around shelves to find some items. When he finds item needed, he takes out the item from shelf. After he finishes shopping, he pays for some items at the counter with RFID reader. On the other hand, a reader located on shelf reports the message that a particular item has to be supplemented because the number of items lacks. Therefore, an administrator supplies the item from the warehouse to the shelf in the market.

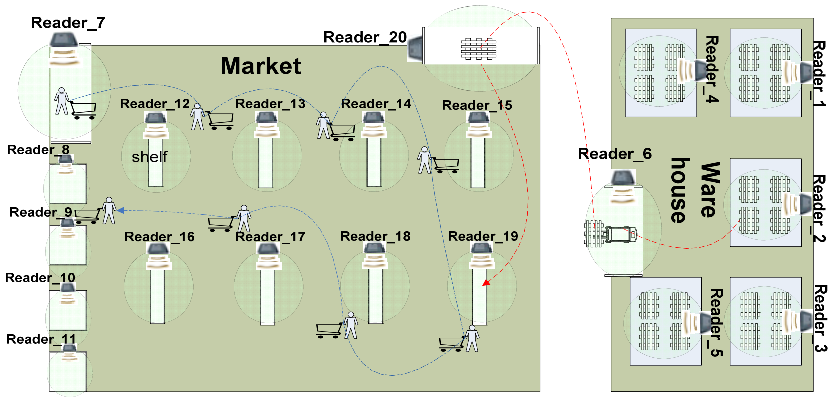


Fig. 3. A customer pays for items, after taking out some items.

Table 2 shows the logical readers named by the role of physical readers in the market.

Table 2. Logical readers definition in avobe scenario.

Logical Reader	Physical Readers	Role of Readers
BSectionReader	Reader_1 ~ Reader_5	Identify items that exist in the warehouse
BGatewayReader	Reader_6	Identify items that take out from warehouse
SEnteranceReader	Reader_7	Identify the people that enter in the market
SGatewayReader	Reader_20	Identify the items that income in the market
SCalculationReader	Reader_8 ~ Reader_11	Identify the items at the counter
ShelfReader	Reader_12 ~ Reader_19	Identify the items that exist at the shelves

When considering the business events occurred by a customer in the market, those events may be caused by shopping, selling, robbery, and refund. And events occurred by a administrator may be caused by incoming of products, outgoing of products, and supply of products in the shelf. All business events defined above can be expressed by our proposed method. Figure 4 shows complex event definition about “SELL”. When an item on the shelf with RFID readers disappears, PredictedObservation event may occur as soon as possible. At that time, another reader waits for a tag disappeared from that reader. If the tag is identified by readers in the counter, system may create the action as “SELL”.

```
ON PredictedObservation(ShelfReader, EPC, T1)
IF True
          DEFINE E1 = RelativeObservation(EPC, ShelfReader, AFTER,
                                     SCalculationReader, T2)
          ON E1
          IF True
                    DO “SELL”
```

Fig. 4. The complex event definition of “SELL”.

Figure 5 shows the complex event definition about “THEFT”. When an item on the shelf with RFID readers disappears, PredictedObservation event may occur as soon as possible. If the PredictedObservation event occurs in one reader, another reader in the store waits for a tag disappeared from that reader. If the tag is identified by reader in the gateway rather than reader at the counter, the system creates this action as “THEFT”.

```
ON PredictedObservation(ShelfReader, EPC, T1)
IF True
          DEFINE E1 = RelativeObservation(EPC, ShelfReader, AFTER,
                                     SCalculationReader, T2)
          DEFINE E2 = RelativeObservation(EPC, ShelfReader, AFTER,
                                     SEnteranceReader, T3)
          ON (¬E1 AND E3)
          IF True
                    DO “THEFT”
```

Fig. 5. The complex event definition of “THEFT”.

5.2 Scenario 2 (Recognition of Right Path)

Another example, recognition of right path, can be performed in our proposed method as follows. This scenario shows that a tag is moving along any fixed path formed by many readers. Figure 6 shows the situation that checks whether any tag X is moving along its correct fixed path or not. We amuse that the fixed path is specified as A->C->D->E->G->H. In this case, if we use any event model that do not define any relationship among readers, although tag X arrives at the reader H after reader A, C, D, E, F, G , and H, the system may create a complex event that tag X was moved within the correct fixed path. However, tag X is moved incorrectly, because tag X passes through the reader F undefined in the path. If we use our event model, tag X has to pass through the reader G after reader E without any other reader between them.

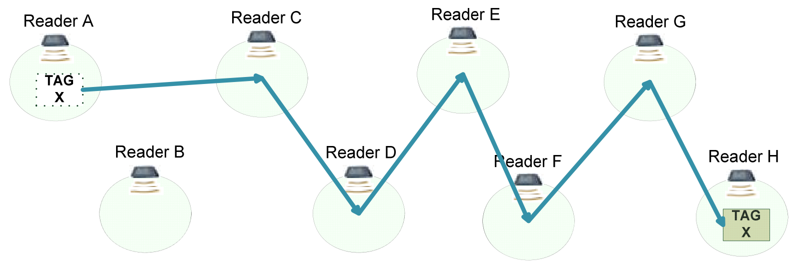


Fig. 6. The recognition of right path.

If the path to move is defined as A->C->D->E->G->H, the system may create complex event that tag X is moved along the incorrect path. Figure 7 shows the complex event definition about "RIGHT PATH".

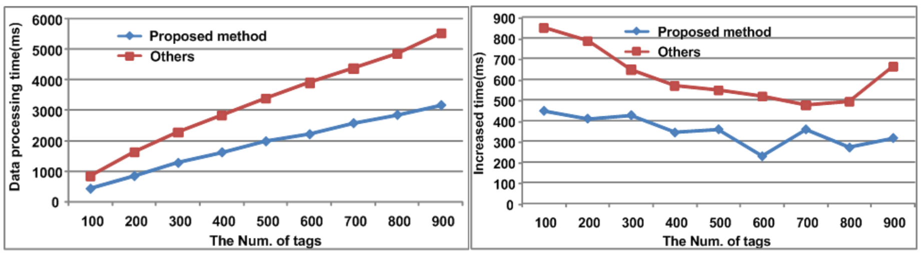
```
ON PredictedObservation(Reader A, X, T1)
IF True
        DEFINE E1=RelativeObservation(X, Reader A, AFTER, Reader C, T2)
        DEFINE E2=RelativeObservation(X, Reader C, AFTER, Reader D, T2)
        DEFINE E3=RelativeObservation(X, Reader D, AFTER, Reader E, T2)
        DEFINE E4=RelativeObservation(X, Reader E, AFTER, Reader G, T2)
        DEFINE E5=RelativeObservation(X, Reader G, AFTER, Reader H, T2)
        ON (E1 ➔ E2 ➔ E3 ➔ E4 ➔ E5)
        IF True
                DO "RIGHT PATH"
```

Fig. 7. The complex event definition of "RIGHT PATH".

5 Experience Results

In this chapter, we will show experiment results when using our CEP to create complex events. We examine the performance of proposed method by using CEP simulator. Performance of CEP is compared with those without any PredictedObservation and RelativeObservation events for equal test conditions. As a test condition, the number of tags is increased by 100 each time. And we assume that 50% of tags are moved fro one reader to another reader. After that, event processing time is measured required to create complex events.

Figure 8(a) shows total event processing time to create complex events when the number of tags increases by 100 each time. Most of time is spent to check whether primitive events satisfy any predefined complex event pattern or not. Therefore, if we can reduce the number of primitive events used in pattern match processing, we may reduce the overall event processing time. Actually, when we reduce the number of primitive events by using PredictedObservation and RelativeObservation events, we can obtain the result as shown in Figure 8(a).



(a) (b)

Fig. 8. (a) Data processing time when the number of tags increase, (b) Increased data processing time when the number of tags increase.

Figure 8(b) shows the increased time required to process events, when the number of tags increases by 100 each time. As mentioned above, to create complex events, pattern matching is needed. Therefore, the more tags increase, the more event processing times increase. However, the number of events is increased in the proposed method less than others because of using two primitive events. Therefore, the proposed method can reduce the time to process events when the number of tags increases by 100 each time.

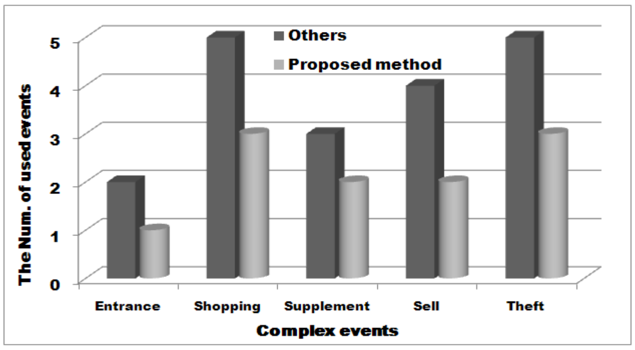


Fig. 9. The number of used events for creating complex events.

Figure 9 shows the number of used primitive events to create complex events in the proposed method and other CEP. Proposed method generates a primitive event based on the relationship of two readers. Also, it can express the direction of moving tags. Therefore, when proposed method and other define equal complex events, the number of used primitive events can be reduced significantly.

6 Conclusion

In this paper, we propose a method that can apply CEP based on RFID standards to RFID systems. Especially, we introduced new primitive events that are PredictedObservation and RelativeObservation, because most of business level events occur when any tag moves between any two or more readers. And we propose a method that can create business level events by using those. And we could confirm the facts that when any pattern of events matched for complex event to generate business level events, execution time can be reduced comparing with other mechanisms without those events and many effective business level events can be provided. When we use our method, execution time can be reduced by about 57% comparing with other mechanisms. In addition, we can check whether any tag moves along the correct route in fixed path or not.

However, in the paper, we just consider RFID data for complex event processing. Therefore we will consider not only RFID data but also other sensor data in future work. Also we will design new primitive event type, event operator, and event rule definition for use these data in complex event processing.

References

1. EPCglobal, http://www.epcglobalinc.org/.
2. EPCglobal Proposed Specification, The Application Level Events (ALE) Specification Version 1.0, EPCglobal (2005)
3. Working Draft Version, EPC Information Services (EPCIS) Version 1.0 Specification, EPCglobal (2005)
4. Stella Gatziu, Klaus R. Dittrich.: SAMOS: an Active Object-Oriented Database System. IEEE Quarterly Bulletin on Data Engineering, Special Issue on Active Databases (1992)
5. S.Chakravathy, V.Krishnaprasad, E.Anwar, and S.-k. Kim.: Composite Events for Active Databases: Semantics, Contexts and Detection. In VLDB (1994), pages 606-617
6. S. Gatziu and K. R. Dirtrich.: Detecting Composite Events in Active Databases Systems Using Petri Nets. In Workshop on Research Issues in Data Engineering: Active Database Systems (1994), pages 2-9
7. Mark Palmer.: Seven Principles of Effective RFID data Management. Progress Software (2004)
8. Fusheng Wang, Shaorong Liu, Peiya Liu, Yijian Bai.: Bridging Physical and Virtual World: Complex Event Processing for RFID data Streams. In Proc. of the 10th International Conference on Extending Database Technology, Munich Germany (2006)
9. Fusheng Wang, Peiya.: Temporal Management of RFID data. 31st VLDB Conference, Tondheim Norway (2005)
10. John B. Trigg: Progress for RFID: An Architectural Overview and Use Case Review, Progress Software (2005)