

RFID Data Management: Challenges and Opportunities

Roozbeh Derakhshan, Maria E. Orlowska and Xue Li

Abstract—Radio-frequency Identification(RFID) technology promises to revolutionize the way we track items in supply chain, retail store, and asset management applications. The size and different characteristics of RFID data pose many interesting challenges in the current data management systems. In this paper, we provide a brief overview of RFID technology and highlight a few of the data management challenges that we believe are suitable topics for exploratory research.

I. INTRODUCTION

RFID technology uses radio-frequency waves to automatically identify people or objects. There are several methods of identification, but the most common is to store a serial number that identifies a person or object, and perhaps other information, on a microchip that is attached to an antenna (the chip and the antenna together are called an RFID transponder or an RFID tag). The antenna enables the chip to transmit the identification information to a reader. The reader converts the radio waves reflected back from the RFID tag into digital information that can then be passed on to computers that can make use of it. RFID is automatic and fast and will replace the barcode system in the near future. The big difference between RFID and barcodes is line-of-sight technology. That is, a scanner has to see the barcode to read it, which means people usually have to orient the barcode toward a scanner for it to be read, RFID by contrast, doesn't require line of sight. RFID tags can be read as long as they are within range of a reader. RFID is a proven technology that's been around since at least the 1970s. Up until now, it has been too expensive and too limited to be practical for commercial applications. But if tags can be made cheaply enough, they can solve many of the problems associated with barcodes and bring much more benefit. Both the size and cost of RFID tags have been continuously falling and the price of a tag might fall to below 5 cents by 2007. With potentially significant applications and the cheap price of RFID technology, it is predictable that every moving object could be tagged in the near future.

In recent years, since Wal-Mart originated and applied RFID technology in supply chain management, RFID has been widely used in many different fields such as defense and military[3], postal package tracking[4], aviation industry[5], health care[6] and baggage and passenger tracing in airports[7]. The wide usage of RFID will depict a scenario of “*an internet of things*”; a world in which billions of objects will report their location, identity, and history over wireless

connections[8]. However, such huge volume of data created from this immense network of tagged items will pose many interesting data management challenges which need to be resolved.

A. Motivation and Perspective

Before considering the research problems associated with RFID data management systems, we note that there has been a great deal of interest in the topic mainly within the retail industry over the last few years. Most of the leading vendors claim to provide at least some RFID systems. Based on a survey in[9], 70 percent of retailers with annual sales over 5 billion dollars will implement some kind of RFID system within the next 18 months. The obvious benefit of RFID for any company directly affected by Wal-Mart mandates[10] is that compliance will help ensure their survival as a supplier to the world's largest retailer. Beyond the obvious sort of “*slap and ship*” scenario[11], the technology has the potential to provide tremendous value to corporations; it will help companies squeeze inefficiencies from their logistic operations and give them better visibility into their supply chain. RFID will also help companies leverage real-time information about operations, aided by a menu of analytic tools, and help them improve their replenishment process. Here we give some of the direct benefits of using RFID technology in supply chain management and retailers[12]:

- **Automation:** The most direct benefits of RFID is an automatic version of the barcode. Using RFID can save a huge amount of labor because we do not need to use humans to scan the items in different parts of the organization
- **Inventory shrinking:** Retailers replenishment decisions are based on the inventory information kept in the inventory system which is assumed to be accurate. However, some times the count in the inventory system does not reflect the correct number of items in the actual inventory due to shrinkage or stock loss. Handling this type of problem is a very costly operation that requires a regular manual counting of stock. RFID technology will decrease the cost of inventory counting significantly and this process can be easily accomplished regularly.
- **Inventory replenishment:** Shelf inventory management is critical to the retail business. Quite often, items are out of stock on the store shelf while there is still plenty of stock available in the backroom of the store. This is because there is no automatic process for detecting the stock out and restocking the shelf once it becomes empty. With RFID technology, shelf inventory can be tracked automatically. For example, if a customer

R.Derakhshan, M.E.Orlowska and X.Li are with the school of Information Technology and Electrical Engineering, The University of Queensland, Brisbane, Australia Email: roozbeh,maria,xueli@itee.uq.edu.au

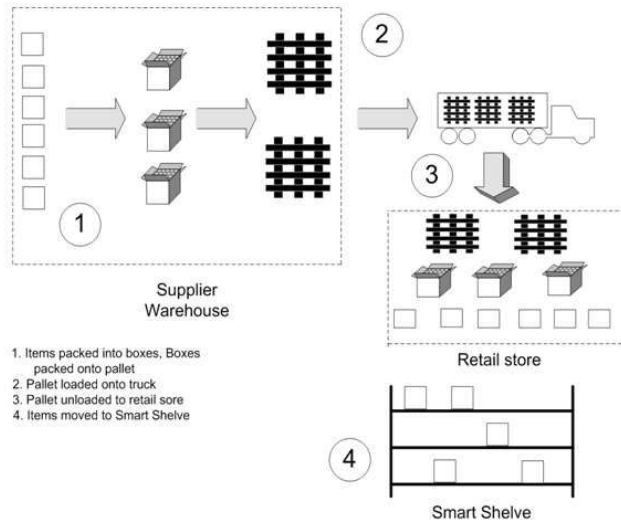


Fig. 1. A Sample RFID Supply Chain System

buy the item that makes the quantity move below the threshold available on the shelf, an automatic restock order will immediately be sent to the store manager for refilling.

- **Visibility of inventory across the supply chain:** The RFID technology provides a comprehensive visibility of inventory throughout the entire supply chain. Figure 1 shows a simplified example of an RFID-enabled supply chain system. Each product item is tagged. So, items that moving around a business can be monitored the from supplier warehouse to the shelves. This comprehensive view of the supply chain allows businesses to trim inventory, streamline logistics, and optimize the efficiency of their workforces as they gain a competitive advantage [12].

B. Characteristics of RFID Data

Since in this work we are interested in data management issues of RFID technology, we must first point out the different characteristics of RFID data. Due to the nature of RFID data, which is fundamentally different to traditional relational and data warehouse technologies, we believe that these differences pose great research challenges and they need to be fully considered in RFID data management systems. We list these characteristics here:

Simple Data: Data generated from an RFID application can be seen as a stream of RFID tuples of the form $(EPC, location, time)$ [13], where *EPC* is an Electronic Product Code which universally identifies an item[14]. *Location* is the place where the RFID reader scans the item, and *time* is the time when the reading took place. As it shows, RFID data does not carry much information. In order to transform this raw data into a form that enterprise applications can use, several levels of inferences must be done. We will later explain more about inferencing in section III-B.

Large in-flood: One of the biggest issues in RFID is to

deal with the in-flood of data. As an example, Wal-Mart will generate as much data in three days as is contained in the entire U.S Library of Congress[15], and it is not just a problem for companies the size of Wal-Mart. Even modest RFID deployment will generate gigabytes of data a day since each item is tagged and will send the data continuously about its EPC, location and time.

Inaccuracy: One of the primary factors limiting the widespread adoption of RFID technology is the inaccuracy of the data stream produced by RFID readers. The observed read rate in real world RFID deployments is often in the 60-70 % range[16], [17]. Unfortunately, such error rates render raw RFID data essentially useless for the purpose of higher-level applications. Thus, we need to clean this data before feeding our system with such unreliable data. As a result, the data in RFID are generally *inaccurate*.

Spatial and Temporal: RFID applications dynamically generate observation (e.g regular changing of the location of tagged items) and the data carry state changes[18]. Thus, in RFID data management, it is essential to model all such data by an expressive data model suitable for application level interaction including tracking and monitoring. Given the facts that not only RFID readers but also the tags can now be built into PDAs, Cell Phones, and moving objects, both tagged items and readers are in constant movement.

In this paper, we review recent work in RFID data management and address some of the issues raised by RFID technology followed by stimulating further work in the area. To the best of our knowledge, this is the first survey that studies the research challenges in RFID data management. The remainder of this paper is organized as follows: section II suggests a layered system architecture followed by a brief explanation about each layer. Section III mentions the research problems which are associated with each layer separately within three subsections. Section IV concludes the paper.

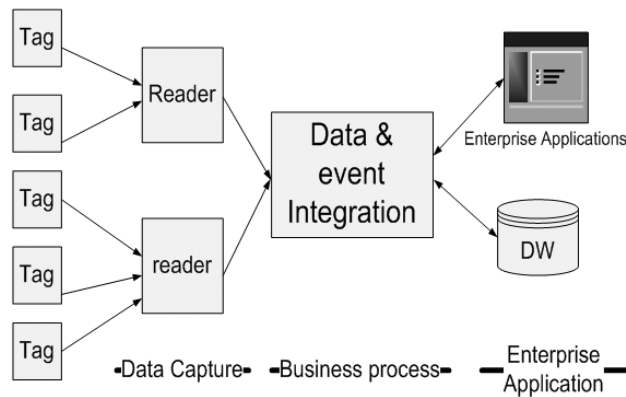


Fig. 2. System Architecture

II. SYSTEM ARCHITECTURE

Fig. 2 suggests a layered architecture for managing RFID data. The lowest layer consists of RFID tags, located on objects such as individual items and pallets. The next layer consists of tag readers. This layer, called *Data capture*, deals with the stream of tuples of the form of $(EPC, location, time)$. Due to the volume and inaccuracy of RFID data, using a low level data process before sending the data to the next layer is inevitable. This low-level data processing would consist of data cleaning and aggregation. In addition, one primary role of this layer is to process a stream of simple events which are generated during the interaction between readers and tagged objects[36]. We will explain more about RFID event processing in section III-C.

Business Process is the second layer of the architecture. This layer is responsible for mapping the low-level data streams from readers to a more manageable form that is suitable for application level interactions. Perhaps the most interesting and challenging tasks in this layer are those that combine business logic with the stream of data emerging from the reader layer. Also detecting more complex events is one of the major responsibilities of business processes.

The third level of the architecture, *Enterprise Application*, supports the business process of enterprise applications such as Supply Chain Management(SCM), Customer Relationship Management(CRM), or Asset Management running on SAP or non-SAP back-end systems[19].

III. RESEARCH PROBLEMS

Based on the general architecture for RFID data management described in section 2, we now outline a number of research problems which are associated with the different layers of system architecture.

A. Data Capture Layer

The data capture layer is responsible for coordinating multiple tagged objects and cleaning incoming data before sending to the next layer, as well as detecting some simple events and reporting them to the management systems.

1) *RFID Data Cleaning*: As we mentioned earlier, one of the primary factors limiting the widespread adoption of RFID technology is the inaccuracy of the data stream produced by RFID readers. So as a result of that we have big load of unreliable data which is useless for the purpose of higher-level processing. Therefore, we need to clean this unreliable data which we call *dirty data*.

Data cleaning is a common problem in most of the data management systems. For instance, in data warehouses it is usually an off-line, centralized, iterative, and sometimes interactive process that focuses on a small set of well-defined tasks[20]. In contrast, RFID data are time sensitive, e.g., inventory control. This demands that the data must be cleaned online before they are streamed to applications. In RFID data management, dirty data appears in three general forms: 1) missed readings 2) unreliable readings 3) data redundancy.

Missed and unreliable readings: These problems are very common in RFID applications and often happen in situations of low-cost, low-power hardware and wireless communications, which lead to frequently dropped readings or with faulty individual readers. Thus, the challenge is to 1) directly address the particular error characteristics (i.e., missed and unreliable readings) and 2) provide a solution that is easy to deploy and configure. *Missed and unreliable readings* problems have been already discussed in different research works[21][16][22].

In general, receptor-based applications are not interested in individual readings or individual devices in terms of time and space, but rather in an application-level concept of *temporal granules* and *spatial granules*[21]. The concepts of *temporal granule* and *spatial granule* have been defined in[21]. Jeffery *et al* introduce *temporal granule* as the smallest unit of time that the application operates. For instance, in a retail scenario when an application continuously monitors the count of items on each shelf, the temporal granule would be each 5 seconds. *Spatial granule* groups items based on some spatial categories which will be the lowest level of spatial granule at which the application operates, such as a shelf in a retail scenario. ESP (Extensible Sensor Stream Processing)

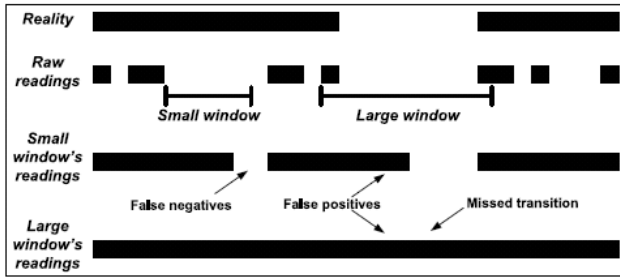


Fig. 3. Tension in setting the smoothing-window size for tracking a single tag (dark bars indicate the tag is present/read): small windows fail to fill in dropped readings (false negatives); large windows fail to capture tag movements (false positives)[16].

is a general technique for sensor stream data cleaning[21]. ESP has five stages each of which is responsible for a different logical aspect of the data. However, ESP is a general technique for sensor data stream and we believe ESP may fail in the following two cases:

- It may fail to find the spatial granule for a supplier warehouse where the location of items are changing regularly (e.g., stage 1 and 3 in Fig. 1). So it is difficult to group the items in order to define coarser spatial granule.
- It may fail when the size of the temporal granule (sliding-window) has not been chosen correctly. The window size must be large enough to smooth the lost readings (False-negative) but small enough to accurately capture tag movements (False-positive). Fig. 3 [16] shows the tension in setting the smoothing-window(temporal granule) size for tracking a single tag.

The first failure refers to the evolving spatial and temporal nature of RFID data when the location of items is constantly changing. For example, in a supplier warehouse where the tagged items are moving on the conveyor belt, it is hard to find the correct unit of spatial granule, which also, increases the chance of missing readings for readers. However, in general, items are not always seen in isolation, but often travel together. A common example is that of pallets and cases. Two cases on the same pallet will tend to both be detected by RFID readers at the same time. Similarly, the pallet will be detected along with the two cases. All together they form an *aggregate* or *containment*. Item aggregation provides an opportunity to enhance the reliability of RFID information by referencing the existence of individual items in terms of the location of their container.

The second failure has already been addressed in[16]. Jeffery *et al* proposed SMURF(*statistical sMoothing for Unreliable RFID data*), the first declarative, adaptive smoothing filter for cleaning raw RFID data streams. SMURF uses a sampling-based approach to find the right temporal window size for the applications. Unlike traditional techniques, SMURF does not expose the smoothing window parameter to the application; instead, it determines the right window size automatically and continuously adapts it over the lifetime of

the system based on observed readings.

Data Redundancy: The redundancy problem is recognized as a serious issue in RFID and sensor networks. We discuss this issue at two different levels :

- Redundancy at readers level
- Redundancy at data level

Redundancy at readers level: This problem occurs when an item is in the vicinity of more than one reader which are simultaneously sending signals to the item. For example, as Fig. 4 shows, readers R1, R2, R3 and R4 are redundant since the tags covered by each is covered by at least one other reader. The optimal solution requires only R2 to be active while the other readers may be turned off. However the problem of finding the optimal solution for the redundant reader elimination is NP-hard [23]. In [23], Carbanar *et al* proposed an algorithm called RRE which is a randomized, decentralized, and localized approximation algorithm for the redundant reader elimination problem. The RRE algorithm has three different steps. First, it detects the set of RFID tags placed in the vicinity of a reader. Second, each RFID reader attempts to write its tag count (number of covered tags) on to all its covered tags. A tag placed in the vicinity of several readers will overwrite the count stored on behalf of a reader only if the new value is larger. The reader that issued the highest count for a tag, locks the tag. In the final step of RRE, each reader sequentially queries all its covered tags to discover the ones it has locked. A reader that has not locked any of its covered tags is declared redundant.

The RRE algorithm assumes that the position of readers will be constant for a long period of time. This assumption may not be held in applications such as the supply chain where the position of readers may change in order to optimize the business process[24].

Redundancy at data level: RFID data are usually regarded as an example of streaming data, and as result, the redundancy on the data level has always been handled in the general way of dealing with data streams. However, we believe, RFID data has its own peculiarities which have been largely ignored. We can compare RFID data and general data streams to illustrate some special features of RFID data. First, the data volume for RFID is usually larger (some readers can perform over 100 readings each second). Second, the RFID data on average is less useful than other data streams. For example, in traffic monitoring and financial applications, every record might be useful for further analysis. On the other hand, in the case of RFID data, we should be able to identify those data that have been repeatedly read multiple times. The less useful part of RFID data is the data that are continuously reported after the initial reading. For instance, in supply chain management, a tagged item can move to the shelf and sit the whole day on the shelf and send the data to the RFID management system constantly. But, from the management point of view, the most useful information for event detection is when the tagged item moves to the shelf and when the item is removed by a customer. Therefore, it is necessary to reduce RFID data redundancy before processing.

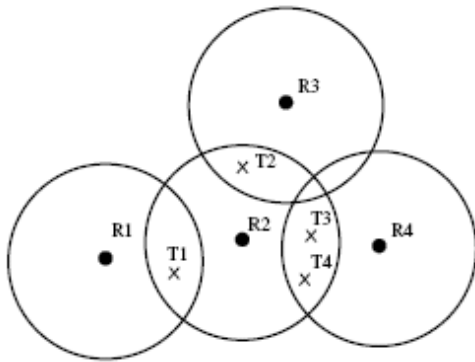


Fig. 4. Redundant reader Example [23]

One simple solution would be for each reader to keep the initial snapshot of the existing items in the vicinity of that specific reader and after each reading compare the number of items at the time with the number of items in the snapshot and report the primitive events if items have been moved or removed. Here we assume that the movement of items from one point to other points in the vicinity of that specific reader is not considered as an event.

B. Business Process Layer

The business process layer is responsible for mapping the incoming observation messages from the data capture layer to a more manageable form that is suitable for the enterprise application level. This layer is also responsible for complex event detection. The business process layer is subject to a standardization effort under the new name “middleware”, such as Siemens middleware[18] and SAP middleware[19].

Perhaps one of the most interesting and challenging issues in RFID data management systems is to transform the low level data into more sophisticated data which can be useful for the enterprise application. In order to achieve this, different levels of inferences must be done on the raw data[24]. The integration of the inference rule and raw data turns the RFID data management system into a real-time and interactive system. For instance, in supplier chain management, we will be able to track the items more accurately in real time and also allow the system to take the appropriate action automatically when some inference rules have been satisfied.

The volume and necessity of having a fast and online query processing in RFID data management systems will bring up an interesting issue called *Inference Rule Materialization*.

1) *Inference Rule Materialization*: The task of gathering a stream of raw data to the centralized database shares many features with analogous tasks in data warehousing[24]. In data warehouses, we select a set of views to materialize in order to facilitate the query processing time, but in RFID data management systems, we must decide which set of inference rules should be materialized beforehand (eager approach) and which set should materialize at the query time (lazy approach). The lazy approach has been discussed in terms

of the RFID cleaning rule in [48]. However, for business rules which are in bigger number than cleansing rules there is also the necessity of having a real time system to take the appropriate action automatically when some inference rules have been satisfied. Here we suggest the eager approach and call it *Inference Rule Materialization*.

There are two major differences due to the nature of RFID which makes *inference rule materialization* different from materialized-view maintenance in data warehouses[25] and materialization of rule-derived data in traditional EDS (Expert Database Systems)[26]. These differences are

- Uncertainty of inference rules
- Regular update on data

Uncertainty of inference rules: The materialization of rule-derived data in EDS has been discussed in[26]. Segev *et al* proposed an algorithm to select derived relations for materialization so that the overall cost of processing the inference rules is minimized while satisfying requirements on query response time. However, rule-derived data in RFID may not be constant due to the uncertainty of inference rules. We use an example to explain the uncertainty of inference rules in the RFID environment.

As Fig. 5 shows, readers at the shipping center indicate that Box A is now contained in Pallet B. An application may query for Box A’s location and the system may respond based on the location of the Pallet B, e.g., on truck C. Now if the readers at the receiving center (warehouse) that unpacked Pallet B fail to read Box A, we must consider various possible explanations. These different interpretations are the results of uncertain inference rules. The first interpretation would be the malfunction of readers in the receiving center, in which case, we can still infer the location of Box A on Pallet B (based on *aggregate* or *containment*). The second alternative is that Box A simply fell off the pallet at some location away from readers or was stolen. We may wish to examine the history of the inference rules and the specific items in order to detect the problem as well as selecting the best set of inference rules to materialize.

The process of tracing back the history of inference rules and individual items is closely related to the problem of lineage tracing in data warehouses [24], and more precisely, the usage of lineage tracing in order to find the source of uncertainty of data has been discussed in[27]. Thus, an interesting open research problem in this area would be using lineage tracing to find the appropriate set of inference rules to materialize under the consideration of uncertainty of inference rules. *Inference Rule Materialization* shares many features with *Probabilistic Views* [28]. However, the granularity of the probabilistic information are key parameters here, and this granularity could be different to RFID cases. We will explain more about lineage tracing in section III-B.2.

Regular update on data: Currency of data is typically not a major requirement for data warehouses. In data warehouses it is often desirable that a warehouse is updated only infrequently (daily, weekly) and at designated times. In contrast, currency is very important in a typical RFID

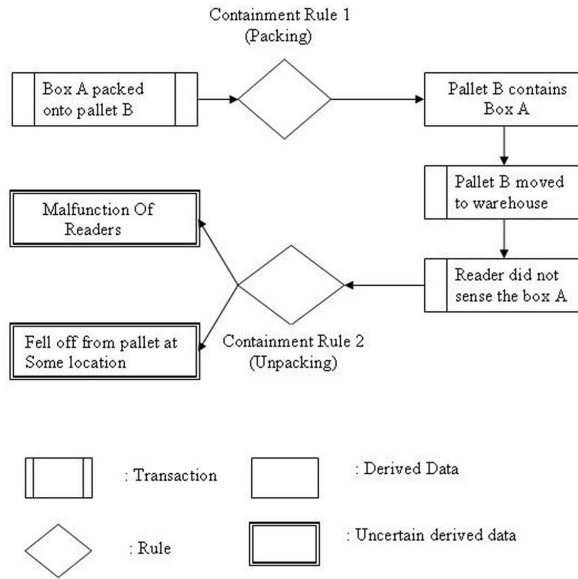


Fig. 5. Uncertainty of inference rule in RFID

deployment. For example, it is important for the supply chain management to know about the arrival of pallets at a distribution center without a significant delay so that it may act immediately and inventory can turn over quickly. Therefore, we need online methods for data propagation in an RFID infrastructure.

2) *Lineage Tracing in RFID*: In general, the **lineage** of a datum consists of its entire processing history. This includes its origin as well as all subsequent processing steps applied to it[29]. Data lineage is sometimes called "lineage tracing", "data provenance" or "pedigree". The *lineage tracing problem* has already been discussed in many different areas such as: data warehousing[30], scientific databases[31], network monitoring, view updates and database visualization [29].

In traditional data warehousing, we collect data from different sources as materialized views in local databases, and keep the view contents up-to-date when the source changes. Most of user's queries are based on high-level aggregated data (views). However, the view content alone sometimes does not provide sufficient information for in-depth analysis. So, sometimes we have to drill from interesting view data all the way down through the original source data that derived the view [32]. This problem called *view data lineage*[33]. In [32], they used *auxiliary information* representing certain intermediate results in the view definition. By using this auxiliary information, they can reduce or entirely avoid source accesses for lineage tracing because querying the sources for lineage information can be difficult or even impossible; the source may be inaccessible, expensive to access, or expensive to transfer data from.

In RFID warehousing[13], [34], [35], due to the nature of RFID applications which demand track and trace queries for individual items, as well as having standard OLAP queries, we need to handle such low level (fine grained) queries

in a reasonable amount of time. Therefore, in RFID data management, the lineage tracing problem is more critical and challenging than traditional data warehousing. To the best of our knowledge, the problem of drilling and digging through the large volume of data, in order to trace an individual tagged item, has not been discussed in literature yet.

One possible solution for this problem would be using a set of *auxiliary information*[32] to find the source of requested individuals rather than drill through the big load of RFID data. However, defining a set of *auxiliary information* in the RFID scenario will pose some research problems such as how much of that auxiliary information is needed to support such fine-grained data lineage in an RFID data management system.

C. RFID event processing

As we discussed earlier, simple and complex event detection is one of the primary roles played by the layers of data capture and business processes respectively. Simple events in RFID applications are those which are generated during the interactions between readers and tagged objects[36]. Generally an RFID reading can be considered as a simple event. Wide deployment of reader devices will soon be able to generate a huge number of events. In order to detect more complex events that are at a semantic level appropriate to end-user applications we need to filter and correlate such huge number of simple events. To do so, we consider these complex events as a group of continuous queries that needs to be run against incoming simple events.

In order to model the process of complex events, we need to define a language that would be able to filter and correlate the events. Complex event languages have been discussed in different contexts such as *publish/subscribe*[38], *stream processing*[39] and *complex event languages*[40]. However most currently proposed languages have some shortcomings in detecting the RFID events. Here, we use shoplifting detection[37] as an example to show those shortcomings. Shoplifting consists of a sequence of events that describe the scenario where an item was picked up from a shelf and then taken out of the store without being checked out. So as we can see in this example, we need to address both occurrence and non-occurrence of events and impose temporal constraints (sliding window query) as well as value-based constraints over these events. In[37], Wu *et al* proposed **SASE** as an expressive language to support queries for complex event processing.

We also need to implement the complex event languages. In order to implement such languages, there is a need for having a data structure in the underlying language model. The issue of using different data structures had been discussed in different areas such as: XML filtering, publish/subscriber and active databases extensively. We can categorize them in three different categories: Automata-based[41][42][43], Petri-Nets[44] and tree and graph based approaches[40][36].

In[41], Altinel *et al* proposed XFilter which used a separate Finite State Machine (FSM) per query and an indexing method to allow all of the FSMs to be executed

simultaneously during the processing the documents. They used SAX[45] in order to encode each documents to the series of events as an input to the XFilter system. In[42], Diao *et al* proposed YFilter which improved the XFilter by combining all of the FSMs into the one Nondeterministic Finite Automata(NFA). YFilter improved the quality of XFilter in terms of time and memory usages.

In SASE, they extend the proposed model on YFilter in order to execute complex event queries over real-time streams of RFID readings. SASE proposes a query plan-based approach which uses native operators at the bottom and then tries to pipeline them in a similar way to relational query optimization in order to decrease the size of intermediate results, however the underlying methods use an NFA (similar to YFilter) in order to detect the individual query.

One of the key concepts that has been ignored in SASE is the notion of time. In many RFID applications, an event is considered as *valid-event* if it happens within or after a time limit. Also SASE assumed that all the events are totally ordered by their timestamps. Such assumption would not be true for all different kinds of RFID scenarios[37]. In order to address these issues above, we suggest to use a *time-based* filter, based on timed-automata[46]. Timed-automata assigns a timer (clock) to every transition in an automata machine. By using time-automata the transition from one state to another would only happen if the value of the clock assigned to each transition compared to the time of the incoming event, satisfies the timing constraint. After each transition, a timed automaton may reassign a new value to the clocks (i.e., reset the clocks). So in timed-automata a system should control the reassignment of its clocks[47]. As it can be seen, this reassignment in RFID applications, with a large number of clocks, could be very complicated.

Here, we suggest to use a sub-class of timed-automata called *Event-clock Automata(ECA)*[47]. Unlike, timed-automata, ECA does not control the reassignment of the clocks, and, the value of a clock is fixed and associated with the time value of incoming events. ECA can have two types of clocks, *event-recording clock* and *event-predicting clock*. We explain these two types of clocks and show how ECA can address the above issues more efficiently as follows:

- *Event-recording clock*: The value in this clock always equals the time of the last occurrence of an incoming event relative to the current time. Fig. 6 shows an example of using an ECA with an event-recording clock for a simple query with a sequence of events (SEQ(a,b,c)). At the bottom, there are a stream of events and the number below the arrow line is the timestamp. Unlike SASE the events are not totally ordered here. On top of that is an ECA with two event-recording clocks: $X_a < X_b$, $X_b < X_c$. This time-constraint can check the order of events and discard the unordered events which was ignored in SASE. As the output shows, only two series of events as respond to our query and discarded the unordered events(e.g: a9, b6, c7).
- *Event-predicting clock*: The value in this clock is a prophecy variable whose value always equals the time

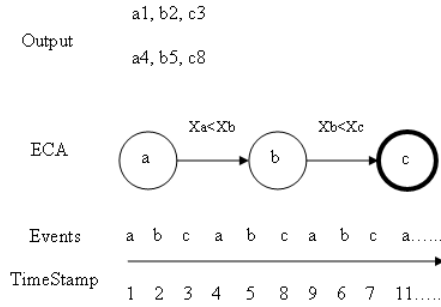


Fig. 6. An example of ECA with event-recording clock

of the next occurrence of an event relative to the current time. So the time-constraint in this clock will satisfy only if the incoming event happens after or within a certain period of time. For instance, in our example, event *b* is valid if only happens after 4 seconds from event *a* then we can simply change the time-constraint on ECA to $X_b < 4$ so the output for this case will then be a1, b5, c8. In other cases, when event *b* should happen within 4 seconds from event *a*, we have to then change the time-constraint to $X_b < 4$.

So by using ECA, we should not control the reassignment of the clock value in time-automata plus we can easily filter and detect unordered events as well as discarding those events which are valid only within or after a time limit.

IV. CONCLUSIONS

RFID technology promises to revolutionize the way we track items in supply-chain, retail store, and asset management applications. In this work, we have provided an overview of the history of this promising technology and highlighted the data management challenges that we believe are suitable topics for exploratory research. The main part of our work focused on discussing the research problems which are associated with the different layers of our suggested system architecture for RFID data management.

We started to discuss the research problems in Data Capture Layer, and discussed data cleaning issues in RFID data management systems. We surveyed the data cleaning literature, plus some of the weaknesses of the proposed methods, and gave suggestions in order to overcome these obstacles.

The second layer of system architecture is the Business Process Layer. We believe there are a number of important open research problems in this layer which have not been explored before. We discussed the *inference rule materialization* in order to achieve better query response time followed by exploring the uncertainty of inference rule. We suggest using the lineage tracing process to trace back the history of inference rules in order to find the source of uncertainties.

As wide deployment of RFID devices in retailers and supply chain management will generate a huge number of in-

flood events, it is expected that RFID event processing will play a major role in RFID researches. So we discussed some of the issues in RFID event processing and surveyed complex event languages and their implementation. We discovered that the notion of time has been ignored in SASE, in an important publication[37], which is the only proposed model for RFID event processing. SASE cannot process the unordered events and does not consider that some events in RFID applications are only valid within or after a period of time. We suggest a *time-based filter* in order to address these shortcomings in SASE.

The third layer of system architecture is Enterprise Application. We briefly discussed this layer; however, we believe the challenges in this area are more business-specific issues. Particularly, RFID data privacy and security issues are not addressed in this paper.

REFERENCES

- [1] L. Sullivan. Target Meets With Suppliers About RFID Plans. *Information Week*, August 2004.
- [2] F. S. I. Metro Group. <http://www.future-store.org/>.
- [3] J. Collins. DOD Tries Tags That Phone Home. *RFID Journal*, 2006.
- [4] P. Harrop. RFID in the Postal Service. *MoreRFID*, 2005.
- [5] J. Collins. Boeing Outlines Tagging Timetable. *RFID Journal*, 2006.
- [6] C. Swedberg. Hospital Uses RFID for Surgical Patients. *RFID Journal*, 2005.
- [7] R. B. Ferguson. Logan Airport to Demonstrate Baggage, Passenger RFID Tracking. *eWeek*, May 2006.
- [8] I. I. Reports. The internet of things. <http://www.itu.int/pub/S-POL-IR.IT-2005/en>, 2005.
- [9] C. Swedberg. Survey says manufacturers drive RFID uptake. *RFID Journal*, <http://www.rfidjournal.com/article/articleview/23711>, May 2006.
- [10] Walmart supplier information: Radio frequency identification usage. <http://www.walmartstores.com>, 2005.
- [11] C. Heinrich. RFID and Beyond: Growing Your Business Through Real World Awareness. Wiley Publishing, Inc, 2005.
- [12] M. Lee, F. Cheng, and Y. Leung. A quantitative view on how RFID will improve a supply chain. *Technical Report RC23789(W0511-065)*, IBM Research Center, November 2005.
- [13] H. Gonzalez, J. Han, X. Li, and D. Klabjan. Warehousing and analyzing massive RFID data sets. In *Proc of the International Conference on Data Engineering(ICDE06)*, pages 1-10, 2006.
- [14] EPC tag data standard version 1.1., technical report. *EPCGlobal Inc*, April 2004.
- [15] M. Palmer. Principles of effective RFID data management. *Enterprise Systems*, March 2004.
- [16] S. Jeffery, M. Garofalakis, and M. Franklin. Adaptive cleaning for RFID data streams. *Proceedings of the 32nd international conference on Very large data bases(VLDB)*, Page 163–174, 2006.
- [17] C. Floerkemeier and M. Lampe. Issues with RFID usage in ubiquitous computing applications. *Pervasive Computing: Second International Conference, PERVASIVE*, 2004.
- [18] F. Wang and P. Liu. Temporal management of RFID data. *Proceeding of the VLDB05*, pages 1128-1139, 2005.
- [19] C. Bornhord, T. Lin, S. Haller, and J. Schaper. Integrating automatic data acquisition with business processes experiences with saps auto-id infrastructure. *Proceedings of the 30th VLDB Conference*, pages 1182-1188, 2004.
- [20] E. Rahm and H. Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3-13, 2000.
- [21] S. Jeffery, G. Alonso, M. Franklin, W. Hong, and J. Widom. A pipelined framework for online cleaning of sensor data streams. *International Conference on Data Engineering (ICDE 06)*, page 140, April 2006.
- [22] S. Jeffery, G. Alonso, M. Franklin, W. Hong, and J. Widom. Declarative support for sensor data cleaning. *Pervasive*, May 2006.
- [23] B. Carbanar, M. Ramanathan, M. Koyuturk, C. Hoffmann, and A. Grama. Redundant reader elimination in RFID systems. pages 176-184. *Sensor and Ad Hoc Communications and Networks*, 2005.
- [24] S. Chawathe, V. Krishnamurthy, S. Ramachandran, and S. Sarma. Managing RFID data. *Proceedings of the 30th VLDB Conference*, pages 1189-1195, 2004.
- [25] Y. Zhuge, H. Garcia-Molina, J. Hammer, and J. Widom. View maintenance in warehousing environment. *ACM SIGMOD*, pages 316-327, 1995.
- [26] A. Segev and J. L. Zhao. Data management for large rule systems. *Proceeding Of the 17th International Conference on Very Large Data Base(VLDB91)*, pages 297-307, 1991.
- [27] O. Benjelloun, A. Sarma, A. Halevy, and J. Widom. Uldbs: Databases with uncertainty and lineage. *Proceeding Of the 32th International Conference on Very Large Data Base VLDB06*, 2006.
- [28] N. Dalvi and D. Suciu. Answering queries from statistics and probabilistic views. *Proceedings of 31th VLDB Conference*, pages 805-816, 2005.
- [29] A. Woodruff and M. Stonebraker. Supporting fine-grained data lineage in a database visualization environment. *Technical Report UCB/CSD-97-932*, Computer Science Division (EECS), University of California, Berkeley, January 1997.
- [30] A. Gupta, I. Mumick, and K. Ross. Adapting materialized views after redefinitions. In *Proc. of the ACM SIGMOD(95)*, pages 211-222, 1995.
- [31] N. I. Hachem, K. Qiu, M. Gennert, and M. Ward. Managing derived data in the gaea scientific dbms. In *Proc of VLDB(93)*, pages 1-12, 1993.
- [32] Y. Cui and J. Widom. Tracing the lineage of view data in a warehousing environment. *ACM Transactions on Database Systems*, 25(2):179-227, June 2000.
- [33] Y. Cui, J. Widom, and J. Wiener. Tracing the lineage of view data in a warehousing environment. *Technical report, Stanford University Database Group*, November 1997.
- [34] H. Gonzalez, J. Han, and X. Li. Flowcube: Constructing RFID flowcubes for multi-dimensional analysis of commodity flows. *Int. Conf. on Very Large Data Bases (VLDB06)*, September 2006.
- [35] J. Han, H. Gonzalez, X. Li, and D. Klabjan. Warehousing and mining rfid data sets. *International Conference on Advance Data Mining and Application(ADMA06)*, LNAI 4093, Springer, pages 1-18, 2006.
- [36] F. Wang, S. Liu, P. Liu, and Y. Bai. Bridging physical and virtual worlds complex event processing for RFID data streams. *International Conference on Extending Database Technology (EDBT)*, LNCS 3896, pages 588-607, 2006.
- [37] E. Wu, Y. Diao and S. Rizvi. High-Performance Complex Event Processing Over Streams. *ACM SIGMOD*, Page 407–418, 2006.
- [38] F. Fabret, H. Jacobsen, P. Lirbeta, K. Ross and D. Shasha. Filtering Algorithm and Implementation for very Fast Publish/subscribe System. *SIGMOD*, Page 115–126, 2001.
- [39] A. Arasu, S. Babu and J. Widom. CQL: A Language for continuous queries over streams and relations. *DBPL*, 1–19, 2003.
- [40] S. Chakravarthy, V. Krishnaprasad and S. Kim. Composite events for active databases: Semantic, Contexts and detection. *VLDB*, page 606–617, 1994.
- [41] M. Altinel and M. Franklin. Efficient filtering of XML documents for selective dissemination of information. *Proceedings of VLDB*, Page 53–64, 2000.
- [42] Y. Diao, M. Altinel, M. Franklin, H. Zhang and P. Fischer. Path Sharing and Predicate Evaluation for High-Performance XML Filtering. *ACM Transactions on Database Systems*. Vol. 28, No. 4, 2003, Page 467–516.
- [43] A. Demers, J. Gehrke, M. Hong and M. Riedewald. Towards expressive publish/subscribe systems. In *EDBT*. Page 627–644. 2006.
- [44] S. Gatzu and K. Dittrich. Events in an active object-oriented database system. *Proc of the 1st Intl Conference on Rules in Database Systems*. Page 23–39. 1993.
- [45] SAX Project Organization. SAX: Simple API for XML. <http://www.saxproject.org/>
- [46] R. Alur and D. Dill. A Theory of timed automata. *Theoretical Computer Science*, Volume. 126, Issue 2, 1994. Page 183–235.
- [47] R. Alur, L. Fix and T. A. Henzinger. Event-Clock Automata: A Determinizable Class of Timed Automata. *Proceeding of the 6th Annual Conference on Computer aided Verification(CAV)*. Lecture notes in computer science 818, Springer-Verlag, Page 1–13. 1994.
- [48] J. Rao, S. Doraiswamy, H. Thakkar and L. S. Colby. A Deferred Cleansing Method For RFID Data Analytics. *Int. Conf. on Very Large Data Bases (VLDB06)*, Page 175–186. September 2006.