A Temporal RFID Data Model for Querying Physical Objects

Shaorong Liu, Fusheng Wang and Peiya Liu

February 8, 2007

TR-88

A TIMECENTER Technical Report

Title	A Temporal RFID Data Model for Querying Physical Objects
Author(s)	Shaorong Liu, Fusheng Wang and Peiya Liu
Publication History	February 2007, A TIMECENTER Technical Report

TIMECENTER Participants

Aalborg University, Denmark
Christian S. Jensen (codirector), Simonas Šaltenis, Janne Skyt, Giedrius Slivinskas, Kristian Torp

University of Arizona, USA
Richard T. Snodgrass (codirector), Bongki Moon, Sudha Ram

Individual participants
Yun Ae Ahn, Chungbuk National University, Korea; Michael H. Böhlen, Free University of Bolzano, Italy; Curtis E. Dyreson, Washington State University, USA; Dengfeng Gao, Indiana University South Bend, USA; Fabio Grandi, University of Bologna, Italy; Vijay Khatri, Indiana University, USA; Nick Kline, Microsoft, USA; Gerhard Knolmayer, University of Bern, Switzerland; Carme Martín, Technical University of Catalonia, Spain; Thomas Myrach, University of Bern, Switzerland; Kwang W. Nam, Chungbuk National University, Korea; Mario A. Nascimento, University of Alberta, Canada; John F. Roddick, Flinders University, Australia; Keun H. Ryu, Chungbuk National University, Korea; Dennis Shasha, New York University, USA; Paolo Terenziani, University of Torino, Italy; Vassilis Tsotras, University of California, Riverside, USA; Fusheng Wang, Siemens, USA; Jef Wijsen, University of Mons-Hainaut, Belgium; and Carlo Zaniolo, University of California, Los Angeles, USA

For additional information, see The TIMECENTER Homepage:
URL: <http://www.cs.aau.dk/TimeCenter>

The TIMECENTER icon on the cover combines two "arrows." These "arrows" are letters in the so-called *Rune* alphabet used one millennium ago by the Vikings, as well as by their precedessors and successors. The Rune alphabet (second phase) has 16 letters, all of which have angular shapes and lack horizontal lines because the primary storage medium was wood. Runes may also be found on jewelry, tools, and weapons and were perceived by many as having magic, hidden powers.

The two Rune arrows in the icon denote "T" and "C," respectively.

Abstract

RFID holds the promise of real-time identifying, locating, tracking and monitoring physical objects without line of sight, and can be used for a wide range of pervasive computing applications. To achieve these goals, RFID data have to be collected, transformed and expressively modeled as their virtual counterparts in the virtual world. RFID data, however, have their own unique characteristics – including aggregation, location, temporal and history-oriented – which have to be fully considered and integrated into the data model. The diversity of RFID applications pose further challenges to a generalized framework for RFID data modeling. In this paper, we explore the fundamental characteristics of RFID applications, and classify applications into a set of basic scenarios based on these characteristics. We then develop constructs for modeling each scenario, which then can be integrated to model most complex RFID applications in real world. We further demonstrate that our model provides powerful support on querying physical objects in RFID-based pervasive computing environment.

1 Introduction

RFID (radio frequency identification) technology uses radio-frequency waves to transfer data between readers and movable tagged objects. Thus it is possible to create a physically linked world in which every object can be numbered, identified, cataloged, and tracked. RFID is automatic and fast, and does not require line of sight or contact between readers and tagged objects. With such significant technology advantages, RFID has been gradually adopted and deployed in a wide area of applications, such as access control, library checkin and checkout, document tracking, smart box [1], highway tolls, supply chain and logistics, security, and healthcare [2].

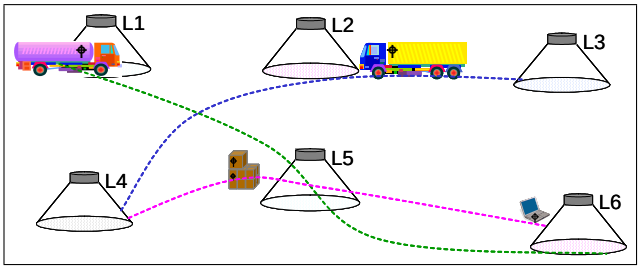

Figure 1: Pervasive Computing with RFID

One major challenge in pervasive computing is to automatically identify, locate, track and monitor physical objects in real time. RFID technology is a perfect fit to address the challenge. Physical objects can be uniquely identified by tagging them with RFID tags and virtually representing them as EPC. (Electronic Product Code is an identification scheme for universally identifying physical objects, defined by EPCGlobal [3].) Readers can be deployed at different locations and networked together providing a RFID-based pervasive computing environment as illustrated in Figure 1, where L1 – L6 denote locations mounted with readers. Tagged objects moving in this environment will then be automatically sensed and observed with their identifications, locations and movement paths.

Readers' observations, however, are raw data and provide no explicit semantic meanings. They have to be transformed into semantic data properly represented with their own data models before they can be integrated into applications. Thus, RFID data modeling, which translates a physical world into its corresponding virtual world, becomes the first step for managing RFID data and supporting business applications. The diversity of RFID applications and the unique characteristics of RFID data, however, pose new challenges to RFID data modeling.

The first challenge is the diversity of RFID applications. i) *Diversity of tags*. Five classes of tags have been defined by EPCglobal[4], ranging from read-only to read-write. ii) *Mobility of readers*. Readers can be either mounted at fixed locations and read tags within their scopes, or move around and approach tags to read them. iii) *Entity representations*. While tags can be attached to objects and serve as surrogates of the objects, handheld or wearable readers can also be associated with objects such as operators. These diversities enable a wide spectrum of RFID applications and make it difficult to generalize a unified data model.

The second challenge is the unique characteristics of RFID data. First, RFID data are temporal in nature. RFID applications dynamically generate observations, which may carry state changes, such as change of locations, and change of containment. Second, RFID data have implicit semantics, which have to be inferred and explicitly

expressed in the data model. RFID tags (and even readers) may act as surrogates of real world objects, and the activities/movements of these tags represent the application logic. These characteristics introduce the challenge of effective mapping RFID observations into the application logic in real world.

A temporal RFID data model is proposed in [5] to manage RFID applications. This data model captures fundamental characteristics of RFID data and provides effective query support. However, it only works for a special case of RFID applications with read-only RFID tags and fixed readers.

In this paper, we significantly extend the model from [5] and provide a comprehensive and general data modeling framework for a large spectrum of RFID applications, ranging from read-only to read-write RFID tags, and from fixed readers to movable readers. The data model provides a cornerstone and a general guideline for managing RFID data.

This paper is organized as follows. First, we overview RFID technology and underline the characteristics and scenarios that affect the modeling of RFID data in Section 2. We then propose a data model to support general RFID applications in Section 3. In Section 4 and 5, we discuss data modeling for each RFID application scenario. In Section 6, we study comprehensive RFID user cases, and demonstrate how we can model such real-world applications using our data model constructs. We then demonstrate how tracking and monitoring can be effectively supported using our data model. Lastly, we discuss related work followed by conclusion.

2 Overview of RFID Technology

2.1 How RFID Works

A RFID system consists of a host computer, RFID reader, antenna (which is often combined with readers), transponders or RF (Radio Frequency) tags, as shown in Figure 2.

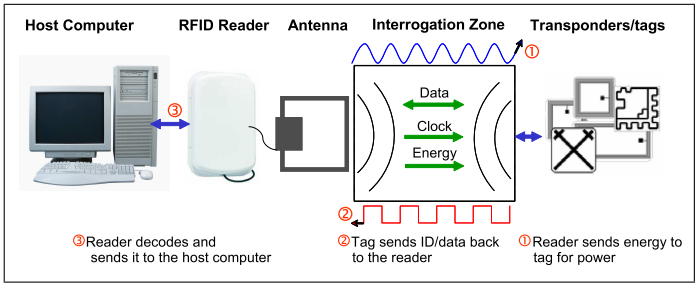

Figure 2: How RFID Works

Tags Tags can be read-only or read-write, and tag memory can be factory or field programmed. A RFID tag is uniquely identified by an ID stored in its memory. EPCglobal classifies RFID tags into 5 classes as shown in Table 1: Class 0 and 1 tags are read-only; Class 2 tags are read-write by readers; Class 3 tags come with onboard sensors and sensors can write data into tags; Class 4 tags are actually wireless sensors used for sensor networks; and Class 5 are readers. (EPCGlobal has released a new EPC standard (Gen 2) for RFID tags, which are slightly different from Table 1. These will not affect our discussions next.)

Readers Readers can be mounted at a fixed point such as warehouse entrance/exit or point of sale. They can also be mobile: tethered, hand-held or wireless. A reader comes with one or more antennas that communicate with tags using RF signals.

Observations As shown in Figure 2, a reader sends energy through RF signal to a tag for power, and the tag sends back modulated signal with ID and data. The reader then decodes and sends the data to the host computer.

2.2 Characteristics of RFID Applications

While there are a wide spectrum of RFID applications, these applications share common characteristics.

Class	Description
0	*Read-only.* Factory programmed ID.
1	*Write once read only.* Factory or user programmed ID.
2	*Read-write.* Users can read and write to tags' memory from a writer(also a reader).
3	*Read-write with onboard sensors.* Sensors write measurement to tags' memory.
4	*Read-write with integrated transmitters.* Active tags that can communicate with other class 4 tags or readers.
5	*Readers.* Wireless networked readers that can communicate with each other.

Table 1: EPC Classes of RFID Tags

- *Identification.* A major difference between RFID and other AIDC (Automatic Identification and Data Capture) technology (e.g., barcode) is its unique ID. Thus, RFID tags are usually attached to or embedded into objects and act as their surrogates: observations of tags denote the observations of the corresponding objects.
- *Location.* One key concept in RFID applications is *location*. A location can be either a geographic location, such as a location from a GPS system, or a symbolic location, such as a warehouse, a shipping route, a surgery room or a smart box [1]. RFID applications track and trace objects by keeping track of their movements among different locations through RFID observations. While these movements are observed as sequences of observations, they signify processes/movements in applications, e.g., movement of products in a supply chain.
- *Aggregation or Association.* Another key concept in RFID applications is *aggregation*, i.e., formation of hierarchical relationships among objects. A common case is the containment relationship, such as a tagged case containing tagged items. Aggregation relationship often implies important semantics. For example, a case with its contained items shares the same location. Another similar concept is *association*, where tagged objects are associated with certain relationships. For example, a laptop may be associated with a list of authorized persons; and a surgery kit may include a list of tools belonging to this kit.
- *Temporal and Dynamic.* RFID applications are mostly temporal-oriented. All events (e.g., observations) are associated with timestamps when the events happen. Observations may represent different semantics, including: i) location changes; ii) aggregation/association relationship changes; iii) start (or end) of operations/processes; and iv) occurrences of new events. Thus a history-oriented data model is essential to preserve the event and state-change histories, which are the key to tracking, tracing, and monitoring physical objects in pervasive computing environment.

2.3 Spectrum of RFID Applications

Reader/Location/Operation \ Tag Type		Class 0,1 Read-only	Class 2 Reader-write	Class 3 Sensor-write (Semi-Passive)
Fixed Reader	Fixed Location	A	F	G
Moveable Reader	No Location	B	-	-
	Discrete Location	C	-	-
	Continuous Location	D	-	-
With Operation		E	-	-

Figure 3: RFID Application Scenarios

In last section, we identify several fundamental elements that determine the characteristics of RFID applications. These include RFID readers, locations, and RFID tags. In fact, different RFID applications may have

different location semantics, and also different associations of readers with locations. While a reader is often fixed at a location, many types of movable readers are also available, including handheld and wearable readers [6], which can be either operated by human or guided by motion control systems. Moreover, locations can not only be fixed symbolic locations but also be continuous physical locations detected by location sensors, such as GPS or local position radar(LPR) [7]. In some RFID applications, locations may be even ignored (refer to Section 4.2).

One special case in RFID applications with moveable readers is that readers may act as surrogates of operators, in which reader observations are used to track operations. *Operation* is an fundamental concept in such applications, which represents a process or an aggregated event in applications (refer to Section 4.5).

Based on the diversity of readers and location semantics as well as the wide spectrum of RFID tag classes (Table 1), we classify RFID application scenarios as shown in Figure 3. (Class 5 tags are actually readers. Class 4 tags can work as other tags with support of larger distance, or compose wireless sensor networks, which are out of the scope of this paper.) In this paper, we focus on the fundamental application scenarios A to G. Real RFID applications are usually combinations of the fundamental scenarios A-G, which can be seen from our case studies in Section 6. Case A represents the scenario in which tags are read-only and readers are fixed. Case B, C and D are all for read-only tags and moveable reader, but with different location semantics. In Case B, location is ignored and identification is the most important. In Case C, a location is a symbolic location while in Case D, a location is a continuous physical location. Case E captures the scenario in which a reader is attached to an operator and an observation signifies the start or end of an operation. Case F (G) is for reader-write (sensor-write) tags with fixed readers.

3 A General Temporal Data Model for RFID Applications: TMDR

3.1 An Example of RFID-Enabled Supply Chain

Figure 4 shows a simplified example of a RFID-enabled supply chain system, with the following steps:

1. Carriers containing items move along the production line for processing, where each item is associated with a read-only tag and each carrier has reader-write tag. A RFID reader is mounted at each station and writes the information about the performed steps onto the tags of the carriers.

2. Readers at the end of the production line check the completeness of the production steps performed on the items. If complete, items are scanned and packed into cases.

3. Cases are packed into pallets in a warehouse.

4. Pallets are loaded onto a truck, which contains a sensor-write tag and a reader with GPS on board.

5. When the truck is en route, its reader together with the GPS sensor periodically sends the exact location information plus environment information by radio (e.g., cellphone-based networks) to a central computer.

6. Pallets are unloaded from the truck and unpacked into cases, which are scanned and moved into a retail store.

7. Items are scanned by RFID readers and checked out at registers of the retail store.

In this sample RFID application, we observe that while objects carrying RFID tags move around, data are exchanged between the tags and readers. Such raw data, however, provides no explicit meaning and must be transformed into a semantically explicit data model before they can be used for identifying, locating, tracking and monitoring objects. To this end, we propose a general data model for RFID applications (DMRA), which is based on ER data model [8] with some significant extensions. Before we present our data model, we first discuss the fundamental entities and relationships in RFID applications as follows.

3.2 Fundamental Entities in RFID Applications

Many entities may exist in RFID applications. Only some of them, however, are directly related to RFID. We consider such entities as fundamental entities in RFID applications.

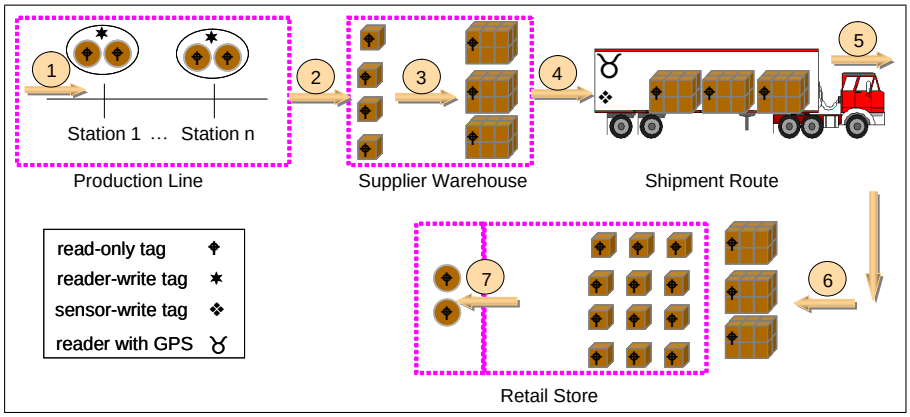

Figure 4: A Sample RFID-Enabled Supply Chain

- *Objects.* These are EPC-tagged objects, such as items, carriers, cases, pallets and trucks in Figure 4. Objects are uniquely identified by their surrogates – EPC tags.
- *Readers.* These refer to readers that use radio-frequency signals to communicate with EPC tags and read data from tags, or write data to read-write tags. A reader can also have multiple antennas. Each RFID reader (or its antenna) is also uniquely identified by its EPC code.
- *Sensors.* These are onboard sensors of RFID tags, such as temperature sensors, motion sensors, or location sensors. A sensor measures a target and then writes the measurement to its master RFID tag.
- *Operators.* These refer to humans who operate with a reader. For example, a nurse wearing a wearable reader may interact with syringes, medicines and patients. The interactions between the operator (e.g., the nurse) and objects represent certain operations. In such cases, readers are surrogates of operators.
- *Locations.* A location represents where an object is (or was), and can be either a physical location (such as a point detected by GPS), or a symbolic location. The granularity of locations can be defined according to application needs. Moreover, one location may contain another. For example, in Figure 4, the locations include Station 1 to Station n, production line end checkpoint, supplier warehouse, route from the warehouse to a retail store, retail store, and customers. In a smart box application [1], locations are simply "in-box" and "out-of-box."
- *Transactions.* These refer to business transactions in which EPC is involved. For example, a checkout involves a credit card transaction with many EPC readings. (Transactions are not considered in many RFID applications, and here we consider them for completeness.)

3.3 Relationships between Entities

While the entities in RFID applications are static in general, relationships among these entities can be either static or dynamic. Static entity relationships are similar to traditional relationships in ER model. For example, the relationship between an object and its on-board sensors – *OBJECTSENSOR* – is a static relationship. Due to the temporal nature of RFID data, most relationships in RFID applications, however, are dynamic and history-oriented. Entities interact with each other and generate movement, workflow, operations, and business logic. The interactions are in two forms: *event* and *state* changes.

Events are generated when entities interact, which include:

- *Observations.* These are generated when readers interact with objects.
- *Sensor Measurements.* These are generated when a sensor on a tag senses a target (e.g., temperature).
- *Property Values.* These are generated when a writer (i.e., a reader) writes the value of an object property into the object's tag. For example, a reader may write the processing steps performed on an object to the tag attached to the object. The property here is a processing step.

- *Transacted items.* These are generated when an object participates in a transaction.

Besides event changes, there are also state changes when entities interact. State change history, i.e., the information about during which period an object is in a certain state, is essential to tracking and monitoring applications and has to be well captured in RFID data models. In RFID applications, state changes include:

- *Change of Object Locations.* For instance, the truck and its loaded pallets leave a warehouse.
- *Change of Object Aggregation Relationships.* For example, cases are packed onto pallets, as shown in Figure 4.
- *Start/End of an Operations.* An operation represents a process or a history-oriented events. An RFID observation can be used to signify the start or end of an operation.
- *Change of Reader Locations.* For instance, a reader is deployed at a new location; or a moveable reader enters/leaves a location.

3.4 Data Model for RFID: DMRA

Based on the discussions above, we summarize a general data model for RFID applications (DMRA). This model is based on ER model and inherits the modeling of static entities and static relationships. The new features extended are discussed as follows.

- **Temporal Relationships** There are two types of temporal relationships among RFID entities: relationships that generate events and relationships that generate state histories. (We use dash-dot lines and dash lines to represent these two relationships respectively.) For an event-based relationship, we use an attribute `timestamp` to represent the occurrence timestamp of the event. For a state-based temporal relationship, we use attributes `tstart` and `tend` to represent the lifespan of the state.
- **Nested Relations** Nested relationship is a new characteristic in read-write RFID applications. For example, for an application with sensor-write tags, an onboard sensor records the temperature measurement history in the tag. Thus a reader observation contains both the EPC of the tag and the measurement history. That is, the observation relation includes a nested relation: the sensor measurement history. While a nested relation can be directly supported in some commercial databases, here we expand such relations into flat tables for generality.

DMRA much differs from DRER [5] which only covers a special case of RFID applications with read-only tags and fixed readers. This model here is much more general and can support all the cases of RFID applications.

Based on the above discussion on fundamental entities and relationships in RFID applications, we next propose basic data model constructs for each scenario in Figure 3. These basic data model constructs can then be integrated to model most complex RFID applications in real world.

4 Data Model for Read-only RFID Applications

4.1 Case A: Fixed Reader and Read-only Tags

We first start from a common scenario: fixed readers and read-only tags. An example of this case can be simplified from Figure 4 by removing production line part and the reader inside the truck. That is, items are packed onto cases, which are further packed into onto pallets and then loaded onto a truck. The truck delivers the products to a retail store, where the products are unpacked and finally sold to customers. At each location, a fixed reader is used to scan tagged objects.

The data model for this case is illustrated in Figure 5. The relations used for this case are described as follows.

Static Entity Tables

`READER(reader_epc,name,description)`

The READER table records the EPC, name and description of a reader.

`OBJECT(epc,name,description)`

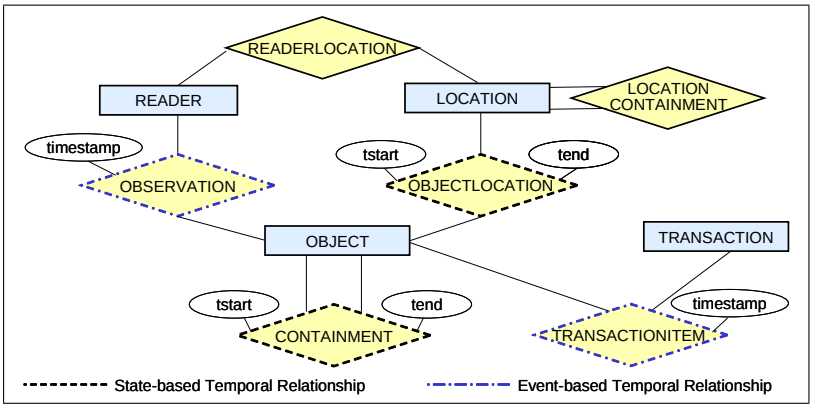

Figure 5: Case A: Fixed Reader and Read-only Tags

The OBJECT table includes the EPC, name, and description of an EPC-tagged object.

`LOCATION(location_id,name,owner)`

The location table defines symbolic locations used for tracking, including id, name, and owner of a location. Table 2 shows a sample location table.

location_id	name	owner
LOC0	Production line E	Manufacturer D
LOC1	Warehouse A	Supplier A
LOC2	Route to retail C	Carrier B
LOC3	Retailer store C	Retailer C
LOC4	Customer	Customer

Table 2: Sample LOCATION Table

`TRANSACTION(transaction_id, transaction_type)`

Since transaction data are business specific, here we simplify a transaction as a record with transaction id and a transaction type.

Static Relationship Tables

`READERLOCATION(reader_epc,location_id)`

This table keeps the location of a fixed reader including EPC of a reader, and the location id.

`LOCATIONCONTAINMENT(location_id, parent_location_id)`

This table records the containment relationship among locations. For example, a warehouse may contain a loading zone and a departure exit.

Temporal Relationship Tables

`OBSERVATION(reader_epc,value,timestamp)`

This table records the raw reading data generated from readers, including reader's (or antenna's) EPC, tag's EPC value, and the reading timestamp.

`CONTAINMENT(epc, parent_epc, tstart, tend)`

This table records in what period `[tstart, tend]` an object (identified by its EPC) is contained in its parent object (identified by the EPC of its parent).

`OBJECTLOCATION(epc, location_id, tstart, tend)`

This table preserves the location history of each object, including an object's EPC, location id, and the period `[tstart, tend]` during which the object stays in that location. For example, Table 3 shows a sample OBJECTLOCATION table, where *UC* denotes 'now'.

`TRANSACTIONITEM(transaction_id, epc, timestamp)`

epc	location_id	tstart	tend
1.2.3.4	LOC0	2004-10-10 12:33:00.000	2004-10-30 17:33:00.000
1.2.3.4	LOC1	2004-10-30 17:33:00.001	2004-11-01 10:35:00.000
1.2.3.4	LOC2	2004-11-01 10:35:00.001	2004-11-07 11:00:00.000
1.2.3.4	LOC3	2004-11-07 11:00:00.001	2004-11-08 15:30:00.009
1.2.3.4	LOC4	2004-11-08 15:30:00.010	UC

Table 3: Sample OBJECTLOCATION Table

This table records information about a transaction including the transaction id, EPC of the object in the transaction, and the timestamp when the transaction occurs.

In the remainder of this paper, when we describe data models for other scenarios, we only present entities and relationships specific to those scenarios. For example, the `TRANSACTION` entity and its associated relationship as well as the `OBJECTCONTAINMENT` relationship are skipped in the models for other cases.

4.2 Case B: Moveable Reader without Location, and Read-only Tags

In many RFID applications, readers are movable with different location semantics. One special case is that location information is unimportant. In such case, identification may be the only interest. For example, a nurse may use a movable reader to identify a patient in order to retrieve the patient's medical history.

The data model for this case, illustrated in Figure 6, can be derived from that of Case A by removing LOCATION entity and location-related relationships.

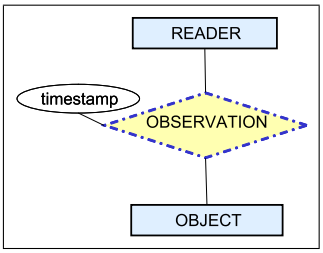

Figure 6: Case B: Moveable Reader without Location, and Read-only Tags

4.3 Case C: Moveable Reader with Discrete Location, and Read-only Tags

In Case A, a reader's location is fixed; a tagged object is movable; and an object's location is identified by a reader's EPC. In contrast, in this scenario, a reader is movable; some special tags are fixed to represent locations; and a reader's location is identified by the observations of such tags. For example, when a nurse with a movable reader enters a room, the reader will scan a tag on the door of the room, from which we can identify the nurse's location. Then the nurse with the reader will interact with other objects in the room to track or identify these objects. The data model for Case C is illustrated in Figure 7, and the relations are discussed next.

Tables

This case has similar tables to those of Case A but differs from Case A in two ways. First, readers' locations are temporal in this case but static in Case A. Moreover, locations in this case are identified by tag EPCs but identified by readers in case A.

`T-READERLOCATION(reader_epc,location_id, tstart, tend)`

This table records in what period [`tstart, tend`] a reader, identified by its EPC, is at a specific location represented by `location_id`.

`E-LOCATION(location_id,name,owner,epc)`

The E-LOCATION table is similar to the one in Case A with one additional attribute `epc` (from the tag used to identify this location).

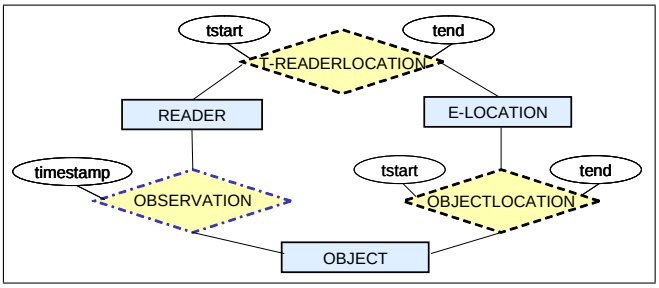

Figure 7: Case C: Moveable Reader with Discrete Location, and Read-only Tags

4.4 Case D: Moveable Reader with Continuous Location and Read-only Tags

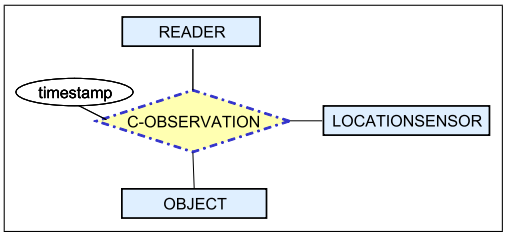

Figure 8: Case D: Moveable Reader with Continuous Location and Read-Only Tags

In Case D, a reader is movable and locations are continuous physical locations. Locations can be identified either by a location sensor (e.g., GPS) or local position radar combined with a reader. A sample application of this case is discussed in [9] for continuous cargo management. Figure 8 illustrates the data model for Case D.

Static Entity Tables

`READER(reader_epc,name,description)`

`OBJECT(object_epc,name,description)`

The above two tables in this case are the same as those in Case A. One additional table is included for describing location sensors:

`LOCATIONSENSOR(sensor_id,description)`

Temporal Relationship Tables

`C-OBSERVATION(reader_epc,object_epc, sensor_id, x, y, z, timestamp)`

This relation is generated by combining the interaction among the READER, the OBJECT, and the LOCATIONSENSOR. It captures the observation of an object (identified by the object's EPC) by a reader (identified by its EPC) at the physical location `(x, y, z)` provided by the location sensor at time `timestamp`.

4.5 Case E: Operations and Read-only Tags

In Case E, an operator with a RFID reader (e.g., by handholding or wearing the reader) performs certain operations or processes to tagged target objects. In this case, a reader acts as the surrogate of an operator and a reading signifies the start or end of an operation. A major difference from Case A, C and D is that, instead of tracking locations, Case E uses RFID interactions to track operations.

One example of this case is that in a hospital, nurses wearing a wearable reader operate on certain targets. Motion sensors trigger the readings at the beginning and ending of an operation. By effectively tracking the sequence of these operations, safer medical procedures can be achieved.

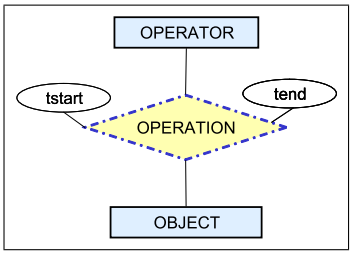

Figure 9: Case E: Operations and Read-Only Tags

The data model for this case is illustrated in Figure 9, and relations in this case include:

Static Entity Tables

`OPERATOR(operator_id, reader_epc, name, description)`
This table describes the operator with its id, name and description as well as the EPC of the reader the operator holds/wears.

`OBJECT(object_epc,name,description)`
This table describes the object being operated, including its unique EPC code, name and description.

Temporal Relationship Table

`OPERATION(operator_id, object_epc, tstart, tend)`
This table records the operations performed by an operator with the operator's IDs, observed object EPC, and the lifespan [`tstart, tend`] of this operation.

5 Data Models for Read-Write RFID Applications

In read-write RFID applications, besides readings of tags, there are also writings to tags. Writings can be performed by a reader on class 2 tags, or performed by onboard sensors on class 3 tags. Writings may be independent of reading and be more frequent. For example, while the sensor in the truck (Figure 5) periodically writes to the sensor-write tag, the observations of the tag may be less frequent. Thus, when the reader observes the tag, the observation contains not only the EPC value of the tag but also a history of records, either sensor measurement history or application-specific data.

To model these data, data writing can be modeled as an individual aggregation, and this aggregation then forms relationships with other entities.

5.1 Case F: Fixed Reader and Reader-Write Tags

Case F describes the scenario in which readers are mounted at fixed locations and tags are of Class 2 – reader-write tags. In this case, applications may predefine some properties for objects, such as "processing steps." Property values are written through a writer (which is actually a reader) as timestamped values. When a reader observes a tag, it reads not only the tag ID, but also the history of the object property values. (Note that a "writing" reader that writes data to a tag may be a different one from a reading reader that reads these data.) Thus an observation here forms a nested relation. The model for this case is shown in Figure 10, and the tables for this case are listed below. (We omit the `LOCATION` entity, `READERLOCATION` and `OBJECTLOCATION` relationships in Case F and G for brevity, which are the same as those in Case A.)

Static Entity Tables

`READER(reader_epc, name, description)`

`OBJECT(object_epc, name, description)`

Static Relationship Tables

`OBJECTPROPERTY(object_epc,property_id,name)`

This table describes the properties defined for an object, including the object's EPC, property's id and name, which is a weak entity dependent on the `OBJECT` entity.

Temporal Relationship Tables

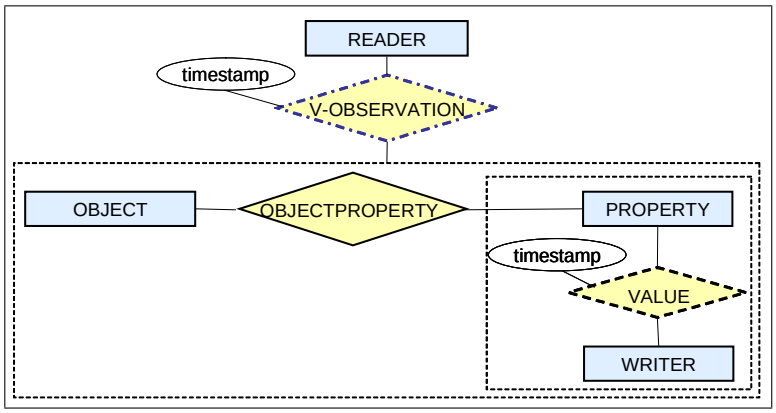

Figure 10: Case F: Fixed Reader and Reader-Write Tags

`N-V-OBSERVATION(reader_epc,object_epc, timestamp, N-PROPERTYVALUE(property_id, writer_id, value, timestamp))`

This nested relation represents the observation of a tag, including the reader EPC, the object EPC, and observation timestamp, together with the nested history of property values written by writers. While nested relations are supported in some RDBMS, this relation can be flattened into two separate tables as follows.

`V-OBSERVATION(vo_id, reader_epc, object_epc, timestamp)`

This table represents reader observations, where `vo_id` is the unique ID for each observation. `vo_id` forms the foreign key linking V-OBSERVATION and PROPERTYVALUE table in the following:

`PROPERTYVALUE(vo_id, writer_id, property_id, value, timestamp)`

This table represents the history of property values with parent key vo_id, property_id, value, the id of the writer that writes the value and the timestamp when the writing occurs.

Note that the `WRITER` entity is omitted here, since it is a reader. Its ID is represented as writer_id in the `PROPERTYVALUE` relation.

5.2 Case G: Fixed Reader and Sensor-Write Tags

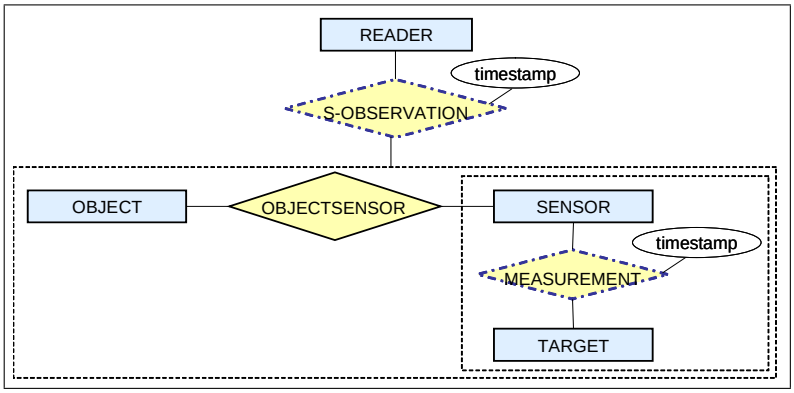

Figure 11: Case G: Fixed Reader and Sensor-Write Tags

Case G is similar to Case F, except that writing is performed by on board sensors. Sensors detect targets

independent of readers and periodically write sensor measurement to tags. When a reader interacts with an object, the reader observes both the ID and the logged sensor measurement history.

Static Entity Tables

READER(reader_epc, name, description)

OBJECT(object_epc, name, description)

Static Relationship Tables

OBJECTSENSOR(object_epc, sensor_id, name, type,measurement_unit,description)

This table describes the information about a sensor, including the EPC of the object containing the sensor, sensor ID, name, type, description and measurement unit.

Temporal Relationship Tables

N-S-OBSERVATION(reader_epc, object_epc, timestamp,
N-SENSORMEASUREMENT(sensor_id, target, value, timestamp))

This nested relation represents the observation of a tag, including the reader EPC, object EPC, and observation timestamp, together with the nested history of sensor measurement written by sensors. This relation can be flattened into the following two tables.

S-OBSERVATION(so_id, reader_epc, object_epc, timestamp)

This table represents reader observations, where so_id is the unique ID for each observation. so_id forms the foreign key linking S-OBSERVATION and SENSORMEASUREMENT table as follows:

SENSORMEASUREMENT(so_id, sensor_id, target, value, timestamp)

This table preserves the history of sensor measurements including the parent key so_id, sensor id, the target (e.g., temperature), value, and the sensing timestamp.

Here the TARGET entity is omitted and the target information is included in the SENSORMEASUREMENT relation.

Since RFID readers are a special type of sensors, readers together with sensors attached to objects form nested sensors. Nested sensors are discussed in PML (Physical Markup Language) [10]. Here we show that nested sensors can be well supported in our data model in consistent with PML.

6 Case Study

With the discussion of the data models for the fundamental application scenarios A-G, we can now demonstrate how RFID-based applications can be expressively modeled by integrating constructs for Case A-G. In this Section, we discuss two examples of RFID-based pervasive computing. We show that all these examples can be properly modeled with the combination of the cases discussed above.

6.1 Example 1: RFID-Enabled Supply Chain

The RFID-enabled supply chain is explained in Section 3.1. In this example, we have three types of tags: read-only tags, reader-write tags, and sensor-write tags. Read-only tags attached to items, cases and pallets are observed by readers fixed at the production stations, warehouse and retail store (Case A). Readers on the production line write the processing steps performed into the reader-write tags attached to the carriers (Case F). The reader on the truck reads the temperature history logged in the truck's tag written by onboard sensors (Case G). The truck reader together with a GPS device observes the continuous locations of the truck (Case D). Thus, this example is a combination of Case A, D, F and G.

Figure 12 illustrates the data model for this example. Note that the left bottom polygon block represents the aggregation of reader-write property values from the production line, and V-OBSERVATION is the corresponding observation of the tags. The right bottom polygon represents the aggregation of sensor-write measurements in the truck, and S-OBSERVATION is the corresponding observation of the tag. C-OBSERVATION is the observation of the truck and its location from the GPS-combined reader. OBJECTLOCATION is the location history of objects (items, cases, and pallets).

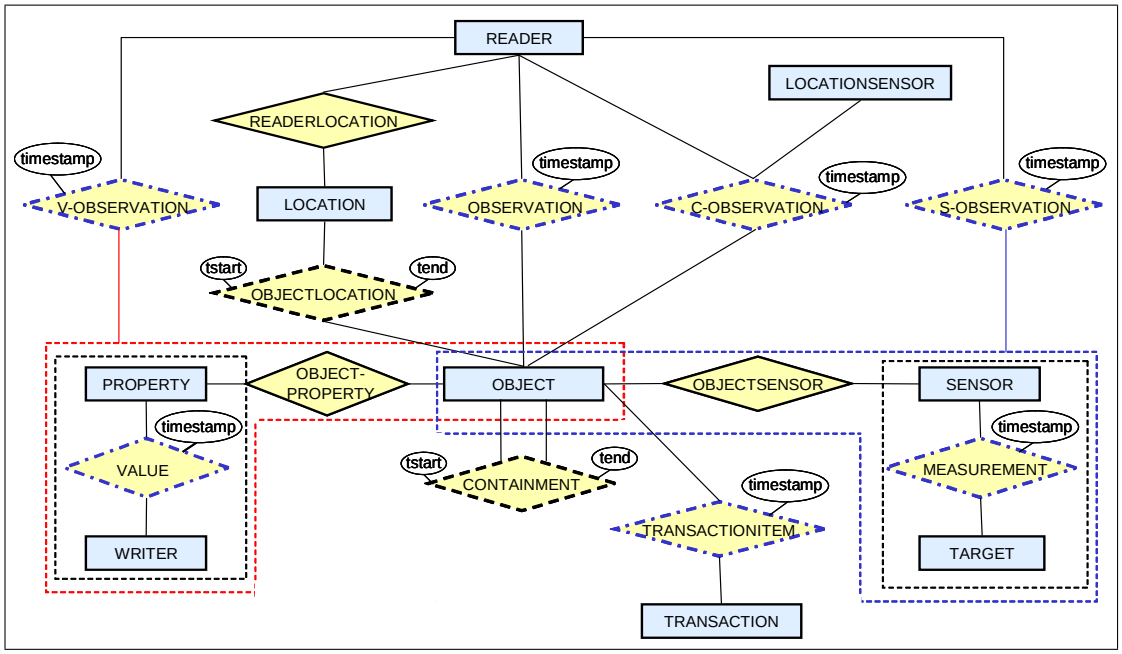

Figure 12: Data Model for RFID-Enabled Supply Chain

The complete list of tables for this example is listed in Figure 13, where 'X' denotes updates to the tables. When objects move around the supply chain, data are automatically collected, transformed based on the model in Figure 12, and stored into the tables as shown in Figure 13. Next, we present how locating, tracking and monitoring can be effectively supported in RFID applications based on our data model.

Next, we show that based on the data model, we can effectively support localization and tracking and monitoring in RFID applications.

6.2 Example 2 – Locating Items in Warehouses

Figure 14 shows a sample application to find locations of some specific objects. For example, a store staff is locating more PDAs with certain tag IDs in a large warehouse which is divided into several divisions. Each warehouse division is tagged with RFID tags at its entrance and exit. The staff holds a wireless handheld reader and walks through the warehouses, and the reader is equipped with a location sensor (e.g., a GPS device or a local position radar). When the staff is close to the tags he is seeking for, the reader will observe that tag and the location sensor will detect the location as well.

In this example, while each warehouse has a fixed location which can be identified by its entrance readers (Case C), finer location of an object is detected by the movable reader together with its location sensor (Case D). Thus this example is a combination of Case C & D.

The data model for this example is shown in Figure 15. Note that `OBJECTLOCATION`, the symbolic locations associated with the tags at the entrance, is predefined at deployment. The `OBSERVATION` table keeps the handheld reader's interaction with the tags on entrances and exits; the `T-READERLOCATION` table records the large-grained location history of the reader, which is changed while the reader enters or exits a warehouse. The `C-OBSERVATION` table logs the observations of seeking-after objects with fine-grained locations through location sensors. Figure 14 (b) describes when a table is updated in the locating process.

7 Effective Support of Tracking and Monitoring in RFID Applications

In this section, we first present an example to demonstrate the expressive of our data model as compared to other event-based models. We then use examples to show the effectiveness of our data model in locating, monitoring, tracking and tracing physical objects in RFID applications.

Location / Table	Deployment	Production Line	Warehouse	Shipment Route	Retail Store (back-store)	Retail Store (front-store)
Static Entity Table:						
READER	X					
OBJECT		X	X			
LOCATION	X					
TRANSACTION						X
Static Relationship Table:						
READERLOCATION	X					
OBJECTSENSOR	X					
OBJECTPROPERTY	X					
Dynamic Event-based Relationship Table:						
OBSERVATION		X	X	X	X	X
C-OBSERVATION				X		
S-OBSERVATION				X		
SENSORMEASUREMENT				X		
V-OBSERVATION		X				
PROPERTYVALUE		X				
TRANSACTIONITEM						X
Dynamic State-based Relationship Table:						
OBJECTLOCATION		X	X	X	X	X
CONTAINMENT		X	X		X	

Figure 13: Tables for the RFID-Enabled Supply Chain

7.1 An Example of the Expressive Power of DMRA

Our data model provides significant advantage on expressive power, as demonstrated by the following example.

"This case (with epc 'e') of meat is tainted. Where has it been and which other products crossed paths with it?"

QUERY A: The Query Based on Event-based Modeling Most past RFID data models are event-based, which make it very difficult to express complex queries for tracking and monitoring RFID objects. For example, the following is the query based on the data model proposed in [11]. It takes eight steps to get the queries done, which is very cumbersome and inefficient.

1. For the EPCe of the tainted case of meat, use ONS together with the sequence registry (dynamic ONS) to obtain a list of addresses of custodians who have handled the object

   => Set of addresses of custodians' EPCIS services a

   May also provide alternative EPCs to track within certain time ranges (e.g. super-tag of pallet)

2. For each address a of an EPCIS service of a party on that supply chain, request a trace of the path taken by EPCe while in their custody / premises

   SELECT * FROM reader_data WHERE tagEPC = e

   => Hash-table (r, t) consisting of ReaderIDs and corresponding timestamps

3. For each reader r within the trace, request all other EPCs read within a time range t-dt to t+dt

   SELECT * FROM reader_data WHERE readerID = r AND timestamp > (t-delta) AND timestamp < (t+delta) AND tagEPC!= e

   => Recordset of observations (e', r, t')

4. Iterate steps 2 and 3 over all custodian EPCIS services, a

   => Hash-table of custodians and set of EPCs read at similar times, (a,e')

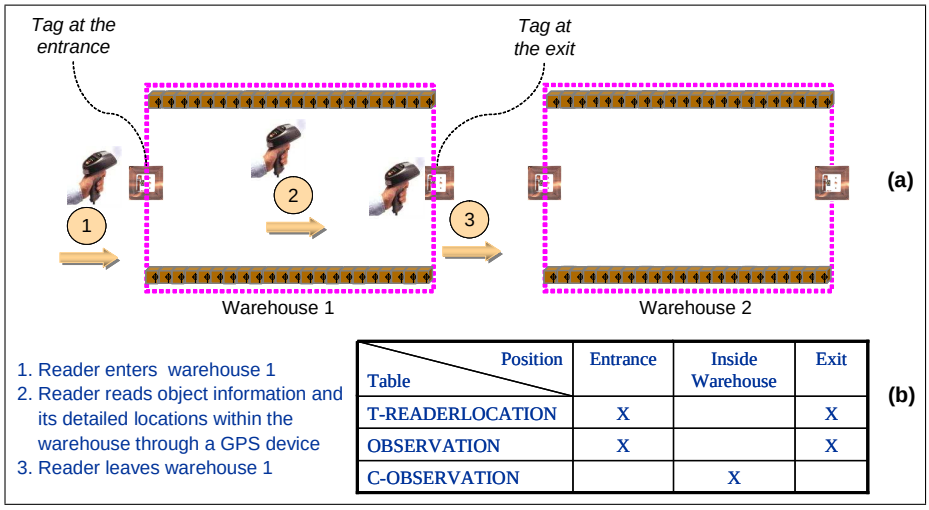

Figure 14: Locating Items in Warehouses

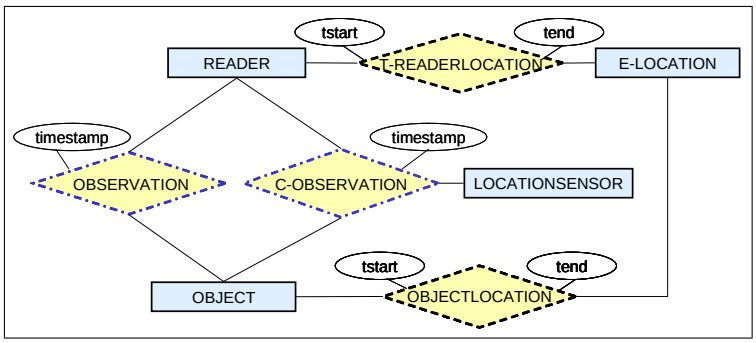

Figure 15: Data Model for Item Finding in Warehouses

5. For each EPCe', perform an ONS lookup to locate the manufacturer's EPCIS service => Hash-table of addresses of custodians' EPCIS services (e',a')
6. Query the EPCIS of each manufacturer, a' for the description/name of EPC e'

   use XPath to query static attribute data for data keyed on e' and node corresponding to the product description

   => Hash-table of product descriptions (e',description)
7. Off-line human evaluation of all product descriptions to determine which products may have caused tainting

   => List of potential contaminant EPCs contaminantEPCs
8. Re-use data collected at steps 2-4 to contact each custodian EPCIS where contamination may have occurred.

   => Name and contact details for those custodians

QUERY B: The Query Based on DMRA On the other hand, with our DMRA data model, the query can be expressed as a single SQL query as follows.

```
SELECT l.location_id, l.epc FROM OBJECTLOCATION l WHERE overlaps(
 l.tstart, l.tend, c2.tstart, c2.tend) AND
 l.epc in  (
  SELECT DISTINCT c2.epc
```

```
  FROM CONTAINMENT c1, CONTAINMENT c2
  WHERE c1.parent_epc = c2.parent_epc
   AND c1.epc = 'e' AND overlaps(
   c1.tstart, c1.tend, c2.tstart, c2.tend)
```

7.2 More Query Examples

Our data model integrates RFID application characteristics into the data model itself, and thus is highly expressive. In the following, we present some examples using the data model for identifying and tracking physical objects in RFID applications.

Q1. Identification: find if an item with EPC '002' is in the store.

```
SELECT *  FROM  OBJECTLOCATION
WHERE epc='002' AND location_id='LOC3'
  AND tend = 'UC'
```

Q2. Real time tracking: get the latest physical location of the truck with EPC '123'.

```
SELECT x, y, z, timestamp
FROM C-OBSERVATION
WHERE object_epc='123'
and timestamp=(
     SELECT max(timestamp)
     FROM C-OBSERVATION
     WHERE object_epc = '123')
```

Q3. Real time tracking: get the latest temperature of the truck with EPC '123'.

```
SELECT m.value, m.timestamp
FROM SENSORMEASUREMENT m,S-OBSERVATION o
WHERE  m.target='temperature'
AND o.object_epc='123'
AND m.so_id = o.so_id AND m.timestamp=(
        SELECT max(m2.timestamp)
        FROM SENSORMEASUREMENT m2
        WHERE m2.so_id = m.so_id)
```

Q4. History tracking: find the location history of an object with EPC value '002'.

```
SELECT * FROM OBJECTLOCATION
WHERE epc='002'
```

Q5. History tracking: find the processing step history for the items in a carrier with EPC value '003'.

```
SELECT * FROM V-OBSERVATION v, PROPERTYVALUE p WHERE
v.object_epc='003' AND v.vo_id = p.vo_id
```

Q6. Aggregation: this item (EPC '005') is defected. Find all items that were processed at the same carrier in the production line.

```
SELECT c2.epc
FROM CONTAINMENT c1, CONTAINMENT c2
WHERE c1.parent_epc = c2.parent_epc
 AND c1.epc = '005' AND overlaps(
 c1.tstart,c1.tend,c2.tstart,c2.tend)
```

8 Related Work

RFID technology has emerged for years and poses new challenges for data management [12, 13, 14, 15]. However, little research has been done on how to effectively data modeling RFID data. Harrison et al [11, 16] summarize the data characteristics of RFID data, and provide some reference relations to represent the data. In their model, RFID data are modeled as events, thus the state history and the temporal semantics of business processes are implicit. This data model is not effective in supporting complex queries such as RFID object tracking and monitoring. A query often needs to be divided into numerous steps [11], which is indirect and inefficient, and not nature for users. [5] develops a data model for RFID applications, which, however, only supports fixed readers with read-only tags. [17] proposes a general data model for RFID applications, which, however, does not provide constructs for modeling basic scenarios.

EPCglobal [4], the current EPC standard group, defines the networks for RFID data and product data [18, 19], but RFID data modeling is not a task in its working group.

There are much work on location management in pervasive computing. Römer et al [20] emphasize the importance of location management, but no systematic model is proposed. In [1], a smart box application model is generalized. Indeed, this application model can be well supported by our model with the definition of several locations, such as "inbox" and "outbox" locations. Location models are studied in [21, 22], which can be combined with our data model for tracking and monitoring. Nexus [23] focuses on positioning and spatial data management.

Recently, major IT vendors are providing sophisticated RFID platforms, including the Sun EPC Network [24], SAP Auto-ID Infrastructure [25], Oracle Sensor Edge Server [26], IBM WebSphere RFID Premises Server [27], Sybase RFID Solutions [28], and UCLA's WinRFID Middleware[29]. These platforms serve as the bridges between the RFID physical world and the rest of the software infrastructure. RFID data are acquired, filtered and normalized through the platforms, and then dispatched to applications. High level RFID data modeling, however, is up to applications.

9 Conclusions and Future Work

In this paper, we develop a comprehensive data model for RFID applications for pervasive computing, by integrating the semantics of RFID applications into the data model itself. We analyze the large spectrum of RFID applications, and generalize them into several typical scenarios, and propose the data model constructs specific to each of them. We show that these generalized constructs can then be integrated to model any RFID applications. Such a general RFID data model provides a bridge between the virtual world and the physical world, and serves as the foundation for RFID data management and RFID middleware. Thus, by building RFID applications on top of this data model, application and data integration cost can be greatly reduced. One significant benefit of our data model is that tracking and locating are made very effective, which are among the fundamental goals of RFID-enabled applications.

References

[1] M. Lampe and C. Flrkemeier. The Smart Box Application Model. In *PerCom*, 2004.

[2] Siemens to Pilot RFID Bracelets for Health Care. http://www.infoworld.com/article/04/07/23/HNrfid implants_1.html, July 2004.

[3] EPC Tag Data Standards Version 1.1. Technical report, EPCGlobal Inc, April 2004.

[4] Epcglobal. http://www.epcglobalinc.org.

[5] F. Wang and P. Liu. Temporal Management of RFID Data. In *VLDB*, 2005.

[6] A. Schmidt, H. W. Gellersen, and C. Merz. Enabling Implicit Human Computer Interaction A Wearable RFID-Tag Reader. In *ISWC*, 2000.

[7] System Description of Local Positioning Radar System. http://www.3d-positioning.com/data/lpr_description.pdf, 2003.

[8] P.P.-S. Chen. The Entity-Relationship Model - Towards a Unified View of Data. *ACM Trans. on Database Systems*, 1(1):9–36, 1976.

[9] Continuous Cargo Management System Links RFID, GPS, Satellite. http://www.foodprocessing.com/vendors/products/ 2005/28.html, 2005.

[10] C. Floerkemeier, D. Anarkat, T. Osinski, and M. Harrison. PML Core Specification 1.0. Technical report, Auto-ID Center, September 2003.

[11] M. Harrison. EPC Information Service - Data Model and Queries. Technical report, Auto-ID Center, 2003.

[12] S. S. Chawathe, V. Krishnamurthy, S. Ramachandrany, and S. Sarma. Managing RFID Data. In *VLDB*, 2004.

[13] S. Sarma. Integrating RFID. *ACM Queue*, 2(7):50–57, 2004.

[14] R. Want. The Magic of RFID. *ACM Queue*, 2(7):40–48, 2004.

[15] M. Palmer. Seven Principles of Effective RFID Data Management. www.objectstore.com/docs/ articles/7principles_rfid_mgmnt.pdf, Aug. 2004.

[16] M. Harrison, J. Brusey, H. Moran, and D. McFarlane. PML Server Developments. Technical report, Auto-ID Center, October 2003.

[17] S. Liu, F. Wang, and P. Liu. Integrated RFID Data Modeling: An Approach for Querying Physical Objects in Pervasive Computing. In *CIKM*, 2006.

[18] S. Clark, K. Traub, D. Anarkat, and T. Osinski. Auto-ID Savant Specification 1.0. Technical report, Auto-ID Center, September 2003.

[19] The EPCglobal Network and The Global Data Synchronization Network (GDSN): Understanding the Information & the Information Networks. http://www.epcglobalinc.org/news/position_papers. html, October 2004.

[20] K. Römer, T. Schoch, F. Mattern, and T. Dübendorfer. Smart Identification Frameworks for Ubiquitous Computing Applications. *Wireless Networks*, 10(6):689–700, December 2004.

[21] C. Becker and F. Drr. On Location Models for Ubiquitous Computing. *Personal and Ubiquitous Computing*, 9(1):20–31, 2005.

[22] I. Satoh. A Location Model for Pervasive Computing Environments. In *PerCom*, 2005.

[23] D. Fritsch, D. Klinec, and S. Volz. NEXUS Positioning and Data Management Concepts for Location Aware Applications. In *Proc. Intl. Sym. on Telegeoprocessing*, 2000.

[24] A. Gupta and M. Srivastava. Developing Auto-ID Solutions using Sun Java System RFID Software. http://java.sun.com/developer /technicalArticles/Ecommerce/rfid/sjsrfid/RFID.html, Oct 2004.

[25] C. Bornhoevd, T. Lin, S. Haller, and J. Schaper. Integrating Automatic Data Acquisition with Business Processes - Experiences with SAP's Auto-ID Infrastructure. In *VLDB*, pages 1182–1188, 2004.

[26] Oracle Sensor Edge Server. http://www.oracle.com /technology/products/iaswe/edge_server.

[27] WebSphere RFID Premises Server. http://www-306.ibm.com/software/pervasive/ws_rfid_premises _server/, December 2004.

[28] Sybase RFID Solutions. http://www.sybase.com/rfid, 2005.

[29] UCLA WinRFID Middleware. http://www.wireless. ucla.edu/rfid/winrfid/.