# RFID middleware design - addressing application requirements and RFID constraints

*Christian Floerkemeier and Matthias Lampe*

Institute for Pervasive Computing
Department of Computer Science
ETH Zurich, Switzerland
{floerkem|lampe}@inf.ethz.ch

## Abstract

Radio Frequency Identification (RFID) technology holds the promise to automatically and inexpensively track items as they move through the supply chain. The proliferation of RFID tags and readers will require dedicated middleware solutions that manage readers and process the vast amount of captured data. In this paper we analyze the requirements and propose a design for such an RFID middleware. We argue that an RFID middleware should not only focus on the application needs, but must also consider the constraints imposed by passive RFID technology.

## 1. Introduction

Radio Frequency Identification (RFID) systems have recently begun to find greater use in industrial automation and in supply chain management. In these domains RFID technology holds the promise to eliminate many existing business problems by bridging the economically costly gap between the virtual world of IT systems and the real world of products and logistical units [9]. Common benefits include more efficient material handling processes, elimination of manual inventory counts, and the automatic detection of empty shelves and expired products in retail stores [2, 15]. It is however not just the business community that can benefit from the use of RFID tags, but also the consumer. The magic medicine cabinet [23], the magic wardrobe [12], and the often-cited smart fridge are some of the applications in which the consumer would benefit from these "smart" products.

The widespread adoption of RFID requires not only low cost tags and readers, but also the appropriate networking infrastructure [20]. Such a supporting RFID infrastructure typically comprises a component – often referred to as RFID middleware or edgeware – that is application-agnostic, manages readers, filters and aggregates captured RFID data and delivers these to the appropriate consumers. To facilitate application development even further, an RFID infrastructure can also feature another component that consumes the events delivered by the middleware, combines the RFID data with application logic, and generates the appropriate application events. While the latter can be a stand-alone system that provides this service to an application, this functionality can also be integral part of an existing application as indicated in Figure 1.

In this paper we analyze the requirements the RFID middleware component should meet in order to manage large deployments of readers and the amount of data these readers capture. The main contribution of this paper is a middleware design that addresses both application needs and the constraints of passive RFID technology.

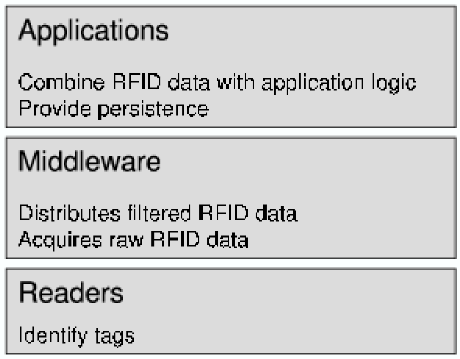


Figure 1: Overview of functional components of an RFID system.

The paper is organized as follows: In Section 2, we list related work. Section 3 discusses the application requirements RFID middleware should meet. Section 4 provides a brief overview of RFID technology and outlines the constraints imposed by the characteristics of RFID. In Section 5, we discuss the RFIDStack, an RFID middleware platform that addresses the requirements and constraints outlined in the two previous sections. The paper concludes with a summary of our contribution in Section 6.

## 2. Related Work

The concept of a distributed networking infrastructure for RFID was initially proposed by the Auto-ID Center, an industry-sponsored research program to foster RFID adoption [20], which coined the term EPC Network. Our work is thus closely related to the middleware component in the EPC Network called Savant [17]. While the Savant-Software features functionality for coping with the idiosyncracies of different kinds of readers and for cleaning the data, it has only limited built-in functionality that specifically addresses the constraints of passive RFID technology. The same holds for a number of other recent commercial and non-commercial efforts such as [19] that facilitate RFID application development.

The requirements, which an overall RFID network infrastructure should meet, have recently also been studied by [3, 5, 22]. Our work differs from the above because it exclusively focusses on the middleware component in an RFID system architecture. We are consequently not covering how RFID data can be interpreted in a given business context and turned into the corresponding application events. This paper focusses strictly on the constraints imposed by passive RFID and how these can be addressed in an appropriate design. We outline for example how the restricted bandwidth available to RFID systems can be efficiently utilized given the application needs for filtered and aggregated data.

Within the RFID community, there are currently various efforts to standardize the interface between reader and RFID middleware [10, 14]. This development, if successful, will have a positive impact on our work, since it makes the component in our RFIDStack superfluous that copes with the idiosyncracies of different reader types. In addition, these harmonization efforts will also help to integrate readers into conventional IT-service management concepts.

## 3. Application Requirements

RFID technology promises to help automate many processes in supply chains, but also in other domains such as aircraft maintenance, baggage handling, and in hospitals. Common to all those applications is that they benefit from dedicated RFID middleware that provides data processing, routing, and reader management functionality. Such a middleware can for example preprocess the raw data captured by the readers before these applications interpret the data and turn them into more meaningful information. Based on an analysis of many different RFID applications, we identified the following requirements an RFID middleware should meet:

*RFID data dissemination.* The information captured by a reader is usually of interest not only to a single application, but to a diverse set of applications across an organization and its business partners. The captured RFID data must thus be broadcasted to the entities that indicated an interest in the data. Different latencies need to be supported, since the desired notification latency depends upon the application type. Applications that need to respond immediately to local interaction with the physical objects require a short notification latency that is comparable to the observation latency. Legacy applications that are not designed to handle streaming data might need to receive batched updates on a daily schedule.

*Data filtering and aggregation.* Common to all applications that make use of the captured data is the desire to receive filtered and aggregated RFID events rather than raw streams of RFID data. Different applications are however interested in a different subset of the total data captured based on the reader and the tag involved. Since RFID permits the identification at the instance-level rather than at the class-level, the fine-grained RFID data need to be aggregated for applications that cannot deal with the increased granularity.

*Reading from and writing to a tag.* Some tags feature not only memory space for an identifier, but for additional data. Middleware solutions should thus provide means to write to and read from this additional memory. This additional memory can then be used to store application data such as expiry dates in order to facilitate data exchange, where no network access is available.

*Reader integration in IT-service management.* The proliferation of readers mandates their integration in an existing IT-service management concept that performs incident, change, and configuration management.

*Privacy* The intended deployment of RFID-based tracking solutions in today's retail environments epitomizes for many the dangers of an Orwellian future: Unnoticed by consumers, embedded RFID tags in our personal devices, clothes, and groceries can unknowingly be triggered to reply with their ID and other information, potentially allowing for a finegrained yet unobtrusive surveillance mechanism that would pervade large parts of our lives. An RFID middleware should consider these consumer fears and the legal guidelines that apply for data collections [11].

Other application requirements that relate to security, scalability, and performance are not discussed here in detail, since they are not unique to RFID and have already been mentioned in [3, 22].

## 4. Constraints imposed by the characteristics of RFID

Before describing how the application requirements listed in the previous section can be met, we will outline the constraints imposed by the characteristics of RFID. We believe that these constraints have a significant impact on the design of an RFID middleware and introduce aspects that are unique to the RFID domain. Any RFID middleware design that fails to include these will result in inefficient data capture and consequently low quality data.

While all passive RFID systems are made up of readers and tags, a wide variety of different RFID systems exist that address the requirements of individual applications, e.g., with respect to range, transmission speed, susceptibility to environmental interferences and cost. Different passive RFID systems can be distinguished by the frequency band they operate in, the coding, modulation, and medium-access techniques used and the supported command set [8]. Since RFID design is generally driven by tradeoffs between different properties - e.g. data rates, read range, susceptibility to environmental interferences, tag form factor - there is no single RFID technology that proves to be superior in all possible application domains.

For the purpose of this paper, the characteristics common to all passive RFID systems are most important, since they have a strong impact on RFID middleware design. These include:

*Limited communication bandwidth.* RFID systems rely on the availability of unlicensed frequency bands. In the UHF frequency band that is particularly suitable for supply chain applications due to its superior read range, the European radio regulations permit the use of fifteen 200 kHz-wide channels between 865.0 MHz and 868.0 MHz by RFID readers [1]. Readers need to listen for other transmitters using the channel before beginning to communicate with the tags. The sensitivity level set for this Listen-Before-Talk scheme (-96dB in the worst case) implies that it is unlikely that two readers will be able to share a channel within one facility [7]. Since large distribution centers might need to run as many as 100 readers, it is evident that readers need to co-ordinate their activities somehow to avoid missing tags that pass by, while the reader is not operating. Another constraint is the bandwidth available per channel which limits data transmission rate between readers and tags. It restricts the number of tags that can typically be identified per second to the order of tenth or hundreds. A tag labeling a shipment that arrives on a pallet carrying more than thousand tagged items can thus easily be missed, unless the identification of the "shipment" tag is prioritized. To facilitate the latter, some RFID protocols permit the selection of a certain group of tags based on data stored on the tag [6, 13]. These bandwidth restrictions do not only apply to systems operating at UHF, but also to other frequency bands. The 13.56 MHz ISM band for example provides a single frequency channel only, but at the same time features more favorable propagation characteristics that result in a shorter reuse distance.

*Reliability issues.* Due to the field nulls, e.g., caused by multipath fading (at UHF) or absorption by objects in the range of the reader, there is no guarantee that a tag stays powered while in the assumed range of the reader. False negative reads can also be caused by collisions on the air interface and transmission errors [4]. The false negative reads result in the fact that a tag will not be continuously detected on consecutive scans by a reader (cf. Figure 2), although the tag remains in the assumed range of the reader.

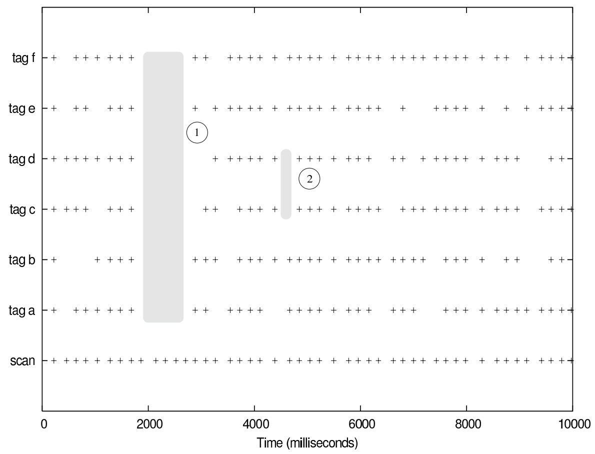


Figure 2: Example of a series of tag reads showing regular reads by an RFID reader (HF) of six tags which are present in its read range. The “missing” reads indicate that not all tags are detected on each scan (false negative reads). The figure also illustrates two different causes of false negative reads: interference problems (1) and collisions on the air interface (2).

*Tag memory.* The design of RFID middleware is also impacted by the memory structure on the tags. The memory on the microchip embedded in the tag usually contains a unique identifier. This can either be a random serial number or an identifier code that incorporates information about the tagged object, e.g., its manufacturer. Some microchips also feature small amounts of additional random access memory. Due to the increased power required to write to the EEPROM on the microchip, the maximum distance between reader and tag for a “write” operation is a fraction of that for a “read” operation.

*Heterogenous reader landscape.* The diverse computing and networking capabilities of readers is also characteristic for RFID. Low cost readers usually support only a single antenna and a serial RS232 interface. More sophisticated devices support several antennas, a TCP host interface, and ample computing resources for on-device data processing. All RFID readers can usually also be parameterized to some extent. This covers network interface parameters such as its ID and port, RF parameters such as transmit power, frequency hop sequences, noise levels, and air protocol specific parameters like data rate, coding type, and MAC properties.

# 5. RFIDStack

In this section we present the RFIDStack, a middleware platform which addresses the requirements and constraints described in the previous two sections. We show how the RFIDStack provides means to utilizes the restricted bandwidth available to RFID systems efficiently given the application needs for filtered and aggregated data. Specific RFID aggregate types are presented that reduce the flood of elementary tag detection events. Characteristics of the messaging component of our RFID middleware implementation are discussed and we outline how these help to address the limitations of RFID. There is also dedicated support for the heterogenous reader landscape and the different memory structures on RFID tags. At the end of the section we discuss the challenge of meeting the requirements to integrate RFID readers into IT-service management.

## 5.1 Filtering, Aggregation and Buffering

The removal of certain tag read events based on the reader which generated the event and the tag data captured is usually referred to as filtering. Table 1 shows the filter types supported by the RFIDStack. In our design, the filtering is carried out on the air interface for bandwidth considerations, whenever possible, or otherwise within the messaging service (cf. Figure 4).

| Filter by | Description |
|---|---|
| Reader Identifier | This filter type allows the application to specify that it is only interested data from a particular set of readers. |
| Tag Identifier and Data | The application can define the tag population that it is interested in, e.g., the restriction to tags attached to pallets. |

Table 1: *Filter types.*

| Aggregate types | Description |
|---|---|
| Entry & Exit | This aggregate type reduces a number of successful reads of a tag to the best estimate when the tag appeared and disappeared from the read range. |
| Count | Applications can prefer to receive information about the total number of items of a specific category detected rather than the individual ID of each object. |
| Passage | This event indicates the direction, in which a tagged object is moved, as a tag moves from one reader to another. Applications prefer receiving a passage event rather than being forced to interpret a sequence of entry and exit events from two individual readers. |
| Virtual readers | When an application does not distinguish between two readers, this aggregate type allows it to virtually join their read range. |

Table 2: *Aggregate types.*

Aggregation is desired to reduce the flood of raw tag reads to more meaningful events such as the first appearance of a tag in the read range and its subsequent disappearance (cf. Figure 3). Aggregation is also needed to address the problem of temporary false negative reads and to smooth the data accordingly [4]. The aggregation types that the RFIDStack supports are listed in Table 2. The aggregation functionality is currently realized via surrogates to which the readers are connected (cf. Figure 4). In the future more powerful readers can carry out

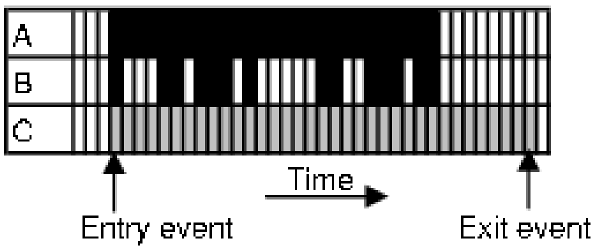


Figure 3: Entry&Exit event illustration. The row labeled A shows the frames, in which the tag under test was present in the read range and should ideally have been detected (dark boxes). The row below shows the frames in which the tag was actually detected by the HF reader. Row C shows the assumed presence of the tag and the point of time, where the entry&exit events are generated.

this functionality themselves, while less powerful readers will continue to rely on a surrogate to carry out the aggregation. Aggregates that are based on data captured by more than a single reader, e.g., passage events are computed by a separate aggregation entity within our architecture (cf. Figure 4).

A buffering component supports applications that require batch updates rather than notifications with a minimum latency. It collects raw tag reads or aggregates over the time period specified and delivers these in one batch via the messaging service at the desired point of time.

### 5.2 Messaging

Given the diverse set of applications that consume the captured RFID data and the networking limitations of readers, an event-based middleware that decouples readers and applications is appropriate for RFID. Readers produce RFID events, deliver them to the messaging system and it is the responsibility of the messaging system to get the messages to their intended destinations (cf. Figure 4). In such a publish/subscribe concept the producer, the reader, does not need to track which applications are supposed to receive a certain message. Likewise, applications consuming RFID data, do not need to maintain communication channels with individual readers, but can simply specify which events they are interested in by submitting subscriptions to the messaging system. The application requirements and constraints characteristic for the RFID domain mandate however a set of special features:

*Full content-based routing.* Applications are only interested in a subset of the total data captured. This subset can be specified using reader ID, tag ID, and possibly tag data (cf. with Table 1). In order to carry out the filtering within the messaging system itself, the nature of RFID events demands the use of a messaging system that provides full content-based routing rather than subject- or topic-based routing. Otherwise, the entire message content would need to be replicated in the subject. Alternatively, applications are forced to carry out some of the filtering locally. They would for example need to subscribe to a "reader" channel feed and discard the messages featuring tags of no interest.

*Subscription feedback mechanism.* While the decoupling of RFID event consumers and producers is desirable, the limited bandwidth available to RFID requires a feedback mechanism for readers to determine whether applications are interested in the RFID data they produce. Such feedback can then lead to an appropriate adaptation of the queries exercised by a reader over the air interface, e.g. targeting a particular tag population at a higher sampling rate or switching off completely to make the bandwidth available to another reader. The filtering of the RFID data is then no longer carried out in software, but over the air interface (cf. Figure 4). If such a feedback mechanism is missing and readers simply coordinate access to the radio channel independent of the application needs, the quality of the captured data will suffer. A reader configured to read any tag might miss a fast-moving pallet tag – potentially the only tag an application is interested in. Likewise, a reader listening for tag replies and occupying a radio channel though no application desires its data will potentially cause a dock door reader unable to find a free channel to miss an outgoing shipment. Such a subscription feedback mechanism is also beneficial from a privacy perspective. If an application does not require individual tag IDs, but rather the quantity of items of a certain product category, the RFID reader can adjust his interrogation accordingly [6, 15]. This permits better performance and the privacy-friendly anonymous monitoring mentioned in [11].

*Reliability* On the one hand, there are applications, e.g., a point of sale system or a magic medicine cabinet, that request the raw RFID data stream with a minimal notification latency. Since these applications need to respond to the detection of individual tags in "real-time", there is little need to retain messages matching their subscriptions, which accumulate, while the connection is being re-established. The situation is different for applications that receive a batched update comprising sometimes "a day's worth RFID events". These require the messaging service to guarantee that a message will be stored until the consumer is able to receive it.

The messaging component of the RFIDStack relies on the content-based router Elvin [21]. Through its quenching functionality it provides the feedback mechanism we require to address the bandwidth limitations. The RFIDStack also benefits from the built-in security features, which prevent unwanted eavesdropping, unauthorized applications receiving notifications and unauthorized readers feeding false information to applications. A single instance of the above event router can process more than 10000 messages per second [16]. In the worst case this corresponds to 100 readers detecting continuously 100 tags per second and feeding the raw data non-aggregated to the messaging system. In a more realistic scenario, a single non-federated event router should thus be able to deal with at least 1000 readers. Likewise, a federation of event routers will be capable of processing the captured data within an RFID-enabled enterprise featuring more than 10000 readers. The support for disconnected applications is currently not realized in the RFIDStack, but has been shown to work in conjunction with the particular event router used [18].

### 5.3 Reading from and writing to a tag

The RFID middleware should ideally make writing to an RFID tag as easy as writing data to a hard disk of a computer. The virtual tag memory service (VTMS) proposed in our system design facilitates this by shielding the application from the particularities of RFID tag memory: limited memory size, different memory organizations, reduced write range. Applications simply provide key-value pairs that should be written to a set of tags. The RFID middleware then checks with the VTMS for the appropriate tag memory block and page to write to given the key. If the write succeeds, the RFID middleware will ac-

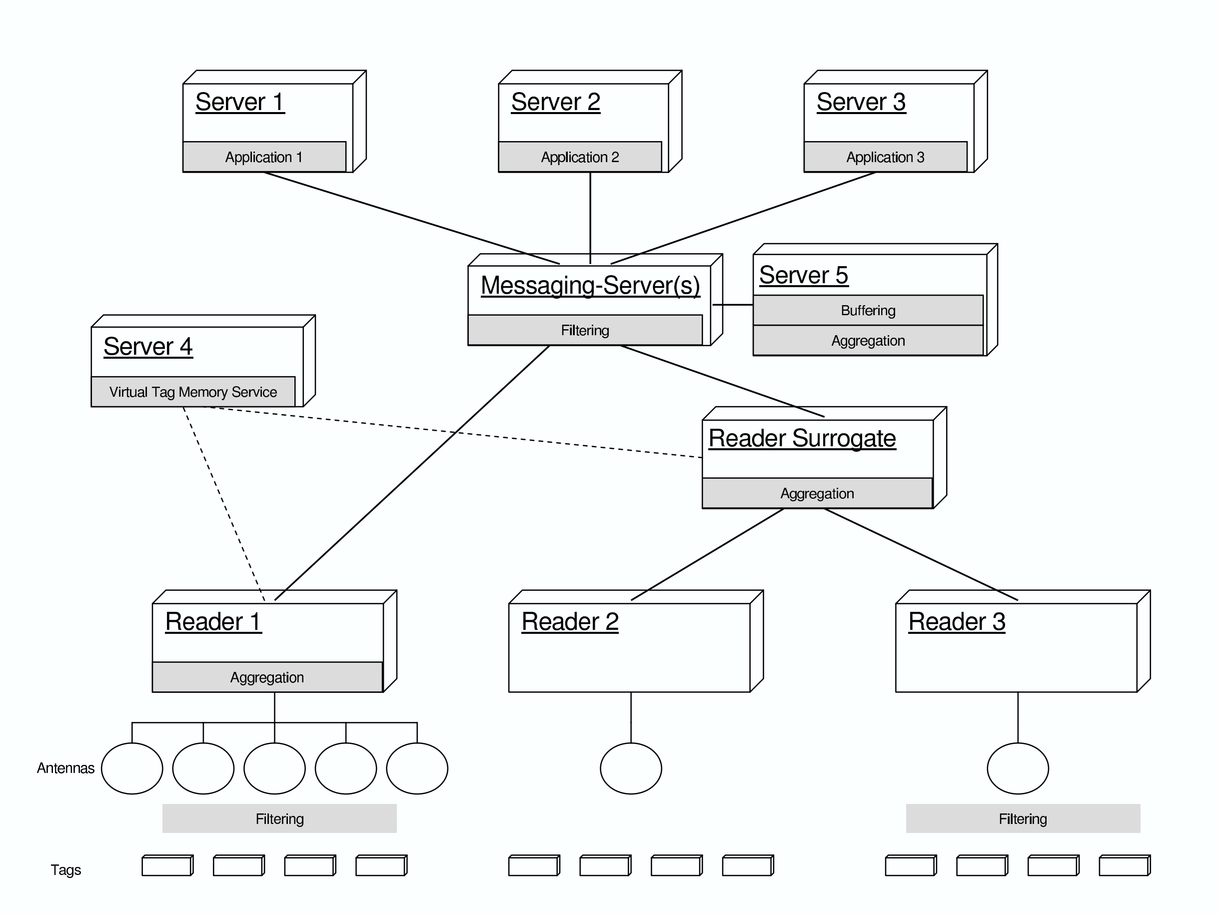


Figure 4: Deployment diagram of the proposed RFID middleware design. The diagram shows how an event-based messaging system decouples applications and readers. It also features the virtual tag memory system and the surrogate concept that address the limitations of tag memory and the heterogeneous reader landscape respectively. To utilize the scarce bandwidth effectively, the filtering is done on the air interface where possible or otherwise directly in the messaging system.

knowledge this to the application and will store a backup copy of the data in the virtual representation of the tag in the VTMS. If the memory gets corrupted at a later stage or the application wants to access the tag's memory, while the tag is outside the range of any reader, the RFID middleware can make the data available via this virtual memory. If the write to the tag fails due to insufficient power, the key-value pair will be stored in the VTMS and flagged as "open". The RFID middleware will retry the write command at a later point of time. If there is insufficient memory space, the application will receive the appropriate error message and the key-value will be stored in the virtual tag memory only. The application can also indicate that the virtual memory of a tag can only be accessed, once the tag is in the read range of the particular reader. The VTMS service is a distributed infrastructure itself (cf. Figure 4).

### 5.4 Reader integration in IT-Service Management

The desirable integration of RFID readers in an existing IT- service management concept that performs incident, change, release, and configuration management is straightforward from a technical perspective. It requires methods to query and modify the existing configuration of a reader, mechanisms to remotely update the software on a reader, and exception reporting functionality. The absence of a de-facto standard to date that fulfills these requirements seems to be more a matter of the reader vendors not agreeing on a common approach rather than technical challenges. In the RFIDStack there is currently only limited support for configuration management of the seven different reader types we support.

## 6. Conclusion

This paper analyzes the requirements RFID middleware solutions should meet in order to manage large deployments of readers and the amount of data these readers capture. We argue that the characteristics of passive RFID technology introduce constraints that are unique to the development of middleware for the RFID domain and present the RFIDStack, a middleware solution that addresses both application needs and the technology constraints.

## 7. References


[1] Radio frequency identification equipment operating in the band 865 MHz to 868 MHz. Technical Report EN 302 208-1, European Telecommunications Standards Institute (ETSI), 2004.

[2] K. Alexander, T. Gilliam, K. Gramling, M. Kindy, D. Moogimane, M. Schultz, and M. Woods. Focus on the supply chain: Applying Auto-ID within the distribution center. Technical Report IBM-AUTOID-BC-002, Auto-ID Center, 2002.

[3] C. Bornhövd, T. Lin, S. Haller, and J. Schaper. Integrating automatic data acquisition with business processes - experiences with SAP's Auto-ID infrastructure. In *Proceedings of 30th VLDB Conference, Toronto, Canada*, pages 1182–1188.

[4] J. Brusey, C. Floerkemeier, M. Harrison, and M. Fletcher. Reasoning about uncertainty in location identification with RFID. In *Workshop on Reasoning with Uncertainty in Robotics at IJCAI-2003, Acapulco, Mexico*, 2003.

[5] S. Chawathe, V. Krishnamurthyy, S. Ramachandrany, and S. Sarma. Managing RFID data. In *Proceedings of 30th VLDB Conference, Toronto, Canada*, pages 1189–1195.

[6] EPCglobal. Class 1 Generation 2 UHF Air Interface Protocol Standard Version 1.0.9, 2005. Available from: `www.epcglobalinc.org/`.

[7] G. Falcke. *The New RFID Standard in Europe*. Available from: `www.rfidjournal.com/`.

[8] K. Finkenzeller. *RFID Handbook: Radio-Frequency Identification Fundamentals and Applications*. John Wiley & Sons, 2000.

[9] E. Fleisch and M. Dierkes. Ubiquitous computing: Why Auto-ID is the logical next step in enterprise automation. Technical Report STG-AUTOID-WH-004, Auto-ID Center, 2003. Available from: `http://www.autoidcenter.org/research/STG-AUTOID-WH-004.pdf`.

[10] C. Floerkemeier. EPC-Technologie – vom Auto-ID Center zu EPCglobal. In E. Fleisch and F. Mattern, editors, *Das Internet der Dinge – Ubiquitous Computing und RFID in der Praxis*. Springer-Verlag, 2005.

[11] C. Floerkemeier, R. Schneider, and M. Langheinrich. Scanning with a purpose – supporting the fair information principles in rfid protocols. In H. Murakami, H. Nakashima, H. Tokuda, and M. Yasumura, editors, *Ubiquitious Computing Systems. Revised Selected Papers from the 2nd International Symposium on Ubiquitous Computing Systems (UCS 2004), November 8-9, 2004, Tokyo, Japan*, volume 3598 of *Lecture Notes in Computer Science*, Berlin, Germany, June 2005. Springer-Verlag.

[12] A. Gershman and A. Fano. *A wireless world: The Internet sheds its chains*. Available from: `www.accenture.com/`.

[13] International Organization for Standardization. ISO/IEC 18000 Part 6: Information technology automatic identification and data capture techniques - Radio frequency identification for item management air interface, 2003.

[14] P. Krishna and D. J. Husak. Simple lightweight RFID reader protocol – working draft, 2005. Available from: `http://www.ietf.org/internet-drafts/draft-krishna-slrrp-03.txt`.

[15] A. Krohn, T. Zimmer, M. Beigl, and C. Decker. Collaborative sensing in a retail store using synchronous distributed jam signallin. In H. W. Gellersen, R. Want, and A. Schmidt, editors, *3rd International Conference on Pervasive Computing*, volume 3468 of *Lecture Notes in Computer Science*, pages 237–254, Munich, May 2002. Springer-Verlag.

[16] Mantara. Elvin router product datasheet, 2005. Available from: `www.mantara.com/`.

[17] Oat Systems and MIT Auto-ID Center. The savant version 0.1. Technical Report MIT-AUTOID-TM-003, Auto-ID Center, 2002.

[18] R. A. Peter Sutton and B. Segall. Supporting disconnectedness - transparent information delivery for mobile and invisible computing. *CCGrid 2001 IEEE International Symposium on Cluster Computing and the Grid, Brisbane, Australia*, May 2001.

[19] B. S. Prabhu, X. Su, H. Ramamurthy, C.-C. Chu, and R. Gadh. WinRFID Ű- A Middleware for the enablement of Radio Frequency Identification (RFID) based Applications. In R. Shorey and C. M. Choon, editors, *Mobile, Wireless and Sensor Networks: Technology, Applications and Future Directions*. Wiley, 2005.

[20] S. Sarma, D. L. Brock, and K. Ashton. The networked physical world – proposals for engineering the next generation of computing, commerce & automatic identification. Technical Report MIT-AUTOID-WH-001, MIT Auto-ID Center, 2000. Available from: `http://www.autoidcenter.org/research/MIT-AUTOID-WH-001.pdf`.

[21] B. Segall, D. Arnold, J. Boot, M. Henderson, and T. Phelps. Content based routing with elvin4. In *Proceedings AUUG2K*, Canberra, Australia, June 2000.

[22] F. Thiesse. Architektur und Integration von RFID-Systemen. In E. Fleisch and F. Mattern, editors, *Das Internet der Dinge – Ubiquitous Computing und RFID in der Praxis*. Springer-Verlag, 2005.

[23] D. Wan. Magic medicine cabinet: A situated portal for consumer healthcare. In *Proceedings of the International Symposium on Handheld and Ubiquitous Computing, Karlsruhe, Germany*, September 1999.