Managing RFID Data

(Extended Abstract)*

Sudarshan S. Chawathe*, Venkat Krishnamurthy†, Sridhar Ramachandran†, and Sanjay Sarma‡

*Computer Science Department
University of Maryland
College Park, MD 20742, USA
chaw@cs.umd.edu

†OAT Systems
265 Winter Street
Waltham MA 02451, USA
{venkat,sridhar}@oatsystems.com

‡Department of Mechanical Engineering
35–010 Massachusetts Inst. of Technology
Cambridge, MA 02139, USA
sesarma@mit.edu

Abstract

Radio-Frequency Identification (RFID) technology enables sensors to efficiently and inexpensively track merchandise and other objects. The vast amount of data resulting from the proliferation of RFID readers and tags poses some interesting challenges for data management. We present a brief introduction to RFID technology and highlight a few of the data management challenges.

1 Introduction

RFID technology has gained significant momentum in the past few years, with several high-profile adoptions (e.g., Walmart.) In addition to applications in retail and distribution, RFID technology holds the promise to simplify aircraft maintenance, baggage handling, laboratory procedures, and other tasks. RFID tags have recently been used to monitor patients in their homes, in order to alert medical workers when abnormal conditions are observed [19]. Standardization efforts such as EPC-IS and PML Core [7] provide the beginnings of a framework for using this technology that spans industries.

Nevertheless, there are some significant challenges that must be overcome before these benefits are realized. Below, we first present an overview of RFID technology, with a focus on major recent applications, such as retail and distribution. We present a high-level system architecture of a distributed RFID system, followed by a discussion of some key data management challenges. We hope to expose some of the issues raised by this technology and to stimulate further work in the area.

The core RFID technology is not new, and its roots can be traced back to World War II where it was used to distinguish between friendly and enemy aircrafts [18]. Technological improvements over the years have led to smaller and cheaper RFID devices. A single-chip design led to the *RFID tag*, a small device composed of a chip, an antenna, and an optional power source, that carries a unique identifier. The 1990s witnessed the use of such tags for card-keys, fuel-station payment systems, and automated toll payment. Such tags were typically specialized for a certain class of applications and cost a few dollars each. The tags typically stored application-specific data and were capable of modest processing on-tag.

Recent years have seen the emergence of a different kind of tag: one that is extremely limited in its abilities and does little more than provide a unique identifier. This approach has two key advantages: First, the simplicity of the tags makes it possible to manufacture them at very low cost. (The target number quoted is 5 cents per tag, a price achievable when the volumes reach the billions.) As a result, it is economically feasible to attach such tags to a large number of objects, even very inexpensive ones (e.g., razors in a retail store). Second, the tags are not application-specific and can be used across application domains. As a result, standards developed for managing RFID data are likely to see a wide cross-industry adoption.

The ability to inexpensively [16] tag and thus monitor a large number of items does raise some serious privacy concerns, especially when the tags are small enough to be unobtrusively attached [20]. Many RFID tags accept a *kill* command that permanently disables the tag. Conceivably, this feature may be used to protect consumers by disabling the tags when the items are purchased at checkout. Another alternative is the

*This work was partly supported by the National Science Foundation with grants IIS-9984296 and IIS-0081860.

Proceedings of the 30th VLDB Conference, Toronto, Canada, 2004

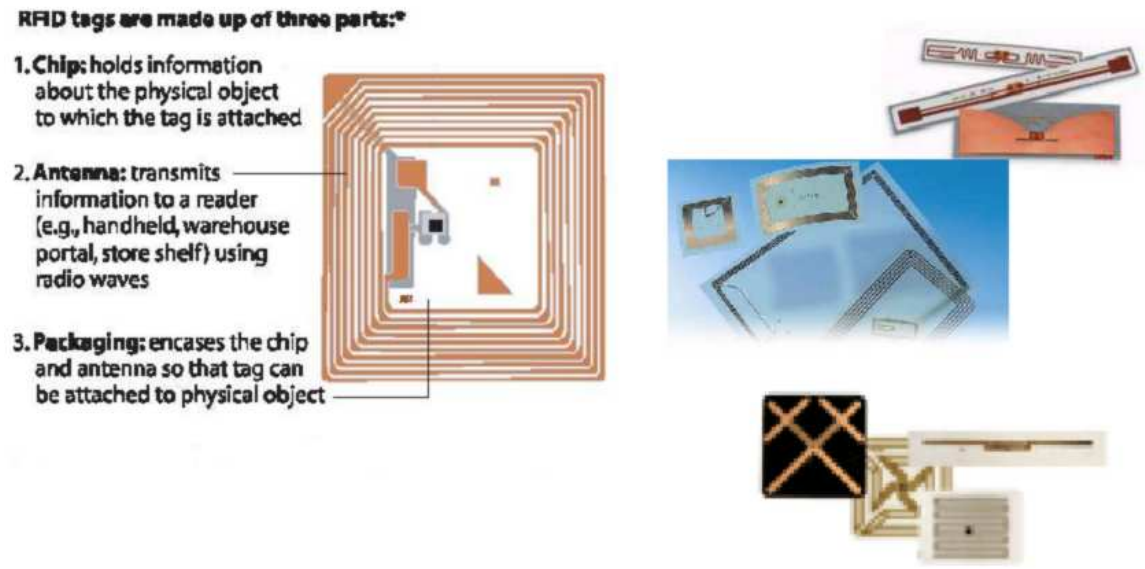

Figure 1: RFID tags

use of techniques such as blocker tags [10].

RFID devices are capable of operating on frequencies ranging from 100 Hz to beyond 2.5 GHz. However, due to regulatory restrictions (typically, country-specific) on use of the radio-frequency spectrum, only a few frequencies are commonly used. The two most common are 13.56 MHz in the HF band and some frequencies around 900 MHz in the UHF band. The HF frequency is usable world-wide, while the UHF frequencies are usable only in the U.S., E.U., and Japan (and vary among them). As in other transmissions, the frequency affects the characteristics of the resulting sensing environment. For example, HF signals propagate more easily through plastic, paper, and moisture than do UHF signals. HF tags are therefore a good choice for applications such as tagging bottles for the pharmaceutical industry. On the other hand, UHF signals have a longer range in the absence of obstructions. Therefore, UHF tags are a better choice for tagging items for the retail industry.

RFID tags may derive the energy to operate either from an on-tag battery or by *scavenging* power from the electromagnetic radiation emitted by tag readers. Further, tags may respond to signals from a reader by either passively reflecting or actively transmitting a signal. The features of tags resulting from different combinations are summarized in Figure 3. (The missing fourth combination is not currently used.)

RFID tags (and readers) have been categorized into five classes based on their capabilities, as summarized by Figure 4. Devices in the higher classes are, in general, larger, more expensive, and more capable than those in lower ones. For example, a class I device scavenges power and provides only a simple identifier using reflective transmission. It may be powered using a class V device, which is essentially a tag reader that can communicate with devices of several other classes. A class II device provides a larger data store than a class I device. In a shipping application, this additional memory may be used to store the electronic packing slip and billing information. (Further details are available elsewhere [6, 17].) An infrastructure for RFID needs methods to cope with such a diversity of devices.

The simplest RFID tag stores only a 96-bit identifier called the EPC. Such tags typically operate on the UHF band and are popular in retail and distribution environments (e.g., Walmart) due to their low cost. Other applications demand tags with enhanced capabilities. For example, the airline industry is using HF tags that can operate in the environmental extremes on an aircraft (including inside the engine) [11, 1]. These tags store not only an EPC but also supplementary data such as a the repair and service history of a part. RFID systems generate data at a high rate. For example, both UPS and FedEx are investigating the use of RFID to further streamline their transportation and delivery systems [8]. Every day, UPS handles 13.6 million packages, amounting to roughly 1.3 Gb/day from this source alone, even assuming the simplest tag (100 bits) and only one read per tag per day. In practice, the data rate is likely to be much higher because a package is scanned at several locations by several sensors.

2 System Architecture

Figure 5 suggests a layered architecture for managing RFID data. The lowest layer consists of RFID tags (located on objects such as cases and pallets). The next

Figure 2: RFID tag readers

Transmission mode	Power source	Name	Range	Life
reflective	scavenging	passive	3 meters	unlimited
reflective	battery	semi-passive	10 meters	5–10 years
active	battery	active	100 meters	1–5 years

Figure 3: RFID tags classified by transmission mode and power source

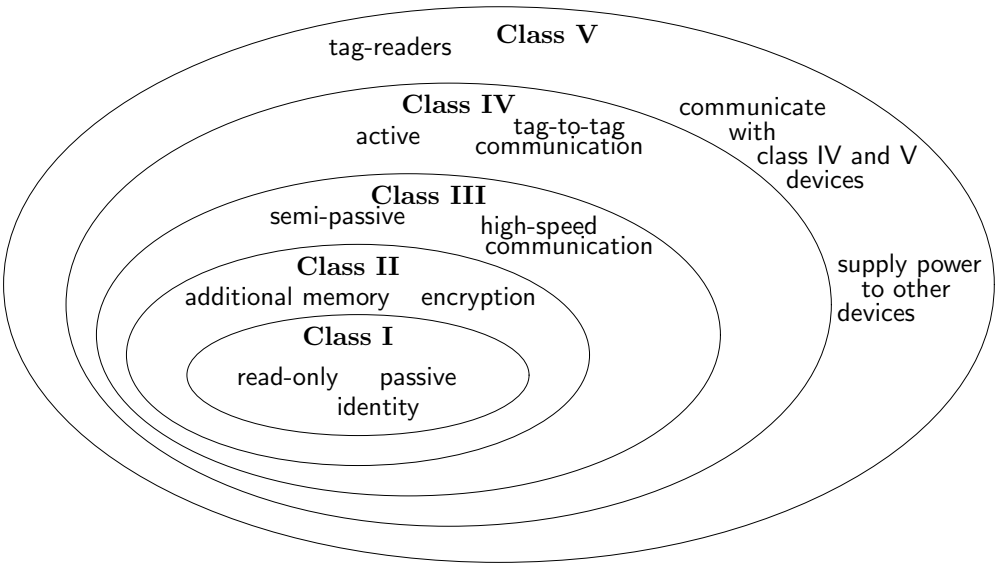

Figure 4: RFID classes

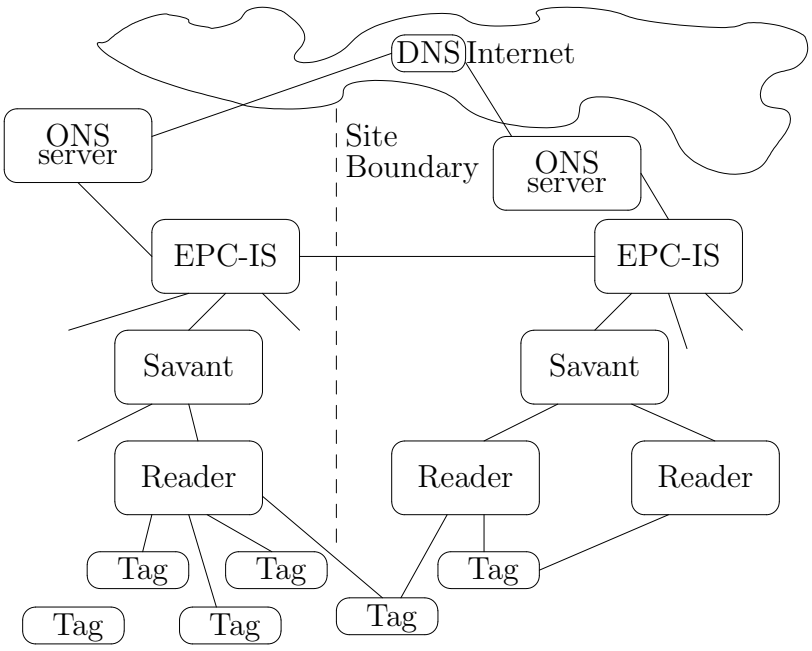

Figure 5: System Architecture

layer consists of tag readers. The interface between these two layers is the so-called *RFID Air Interface* and the RFID protocols for this interface specify the low-level details such as anti-collision techniques (similar to those used by other networking technologies). The focus of this paper is the part of the system that lies above the second layer. The data emerging from the second layer may be regarded as a stream of tuples of the form (r, s, t), denoting that reader r read tag s at time t. Both reader and tag are identified using a global naming scheme called EPC (for Electronic Product Code, by analogy with the UPC standard used for bar-codes). Readers emit such tuples typically in response to some event, such as a timer expiring or a motion-sensor signalling that a new pallet has arrived at a dock door in a warehouse. A reader's mode of operation, and the resulting frequency with which it emits data, is often configurable. A reader may also be capable of filtering the stream of tuples in simple ways.

The third layer of the architecture is responsible for mapping the low-level data stream from readers to a more manageable form that is suitable for application-level interactions. The modules responsible for this mapping were called *Savants* in the original EPC work. Savants may be likened to the wrappers used in data integration systems. In addition to cleaning data and coping with the idiosyncrasies of different kinds of readers, Savants performed further filtering and cleaning of data. Savants were also responsible for setting up the readers (initialization, firmware configurations, etc.). Today, the Savant concept is subject to a standardization effort under the new names "middleware" and "edge systems."

Applications may interact with Savants by issuing simple queries on the state maintained at the Savant (typically, small) as well as by installing standing queries (subscriptions) that result in a stream of matching data. The fourth layer consists of provides higher-level services that are easier for applications to use. For example, this level maps EPC codes to the type of object it represents (individual item, case, pallet) and provides information such as product names and manufacturers. It is also responsible for providing instance-specific information, such as the expiration date of the frozen meat represented by an EPC code.

Perhaps the most interesting and challenging tasks in this layer are those that combine business logic (obtained from other enterprise data systems) with the stream of data emerging from the sensing framework below them. For example, a common task is tracking an item as it travels in the distribution channel (the so-called *track and trace* query). At first glance, this query appears to be nothing more than a simple selection on the tag-ID attribute. However, we need to address problems due to incomplete sensing data and make inferences based on physical axioms or business rules. For example, if a case of razors and a pallet are sensed together in a packing station, a desirable inference is that the case has been packed in the pallet. As a result of such packing, it is possible that readers may fail to sense the tag on the case as the pallet moves through the distribution channel. An implementation of the track-and-trace query must implement such containment scenarios, and must operate correctly when items are unpacked and repackaged in a distribution center.

The final component of the architecture in Figure 5

is part of the *Object Name Service (ONS)*. The ONS is essentially a global lookup service that maps an EPC to a URL that describes the item represented by the EPC.

The design of the ONS service [14] uses the NAPTR facility of the standard Domain Name Service (DNS) [12, 13, 15] to rewrite EPCs into URLs. The mapping may be dynamic. For example, as a case of meat products moves from manufacturer to distributor and further down the supply chain, the ONS mapping changes to reflect the current custodian of the product.

3 Inferences

As noted earlier, the base data emerging from a network of tag readers consists of triples of the form (r, s, t) indicating that reader r read tag s at time t. In order to transform this raw data into a form that enterprise applications such as inventory tracking and resource planning can use, several levels of inference must be made. As a simple example, we may combine join a relation $R_1(r, s, t)$ representing the stream of data from tag-reader R_1 with a relation $L(r, l)$ providing the locations of readers and a relation $N(s, n)$ providing the names of items associated with EPCs to yield a data stream which provides information such as the existence of 27 cases of Gillette razors in aisle 7. However, as noted in Section 2, we also need to make inferences based on containment of items. Further, if a reader (say, reader 5) that has been continually reading the tag of a case of razors (say, case 73) suddenly stops reading it, but if none of the neighboring readers reports any reads for case 73, we may not wish to infer that case 73 has disappeared. Rather, when asked for the location of case 73 (or for an inventory count), we my wish to assume that case 73 is still in the vicinity of reader 5 but is not being detected due to interference, or temporary or permanent malfunction of some component.

We may build a complex web of such inferences, yielding derived data with varying levels of confidence. The methods we use for making, storing, and using such inferences must handle negations of prior assumptions well. (In other words, some level of nonmonotonic reasoning is required.) For example, suppose the operation of readers at a shipping center indicates that case c_1 is now contained in pallet p_1. An application may query for c_1's location and the system may respond based on the location of the pallet p_1 on truck t_1. Now if the readers at the receiving center that unpacks pallet p_1 fail to read c_1, we must consider various possible explanations. Readers may have simply missed c_1 at the receiving site, in which case, we continue to infer its location based on p_1. Alternatively, readers at the shipping site may have incorrectly read c_1. (Such false positive reads, while less common than false negatives, do occur.) Another possibility is that c_1 simply fell off the pallet at some location away from readers or was stolen. False positives may also lead to a database state that indicates an item's presence at two incompatible locations.

At a higher level, an application may wish to examine the history of inferences and contradicting tag reads in order to detect problems or improve the inference procedure. For example, if we observe a high correlation between pallets shipped using truck t_1 and those with missing cases, an investigation may be in order.

This problem is closely related to the problems of data provenance and lineage tracing in data warehouses [2, 5, 4]. An interesting difference is that while most data warehouses do not directly control their inputs (which are determined, for example, by customer actions at a cash register), in an RFID scenario, we have the option of probing readers and changing configurations (often electronically, by adjusting sensitivity parameters).

4 Online Warehousing

The task of funneling a stream of data from tag readers to a centralized database (real or virtual) shares many features with the analogous task in data warehousing [9]. We need methods for collecting data, cleaning it, shipping it, installing it at the warehouse, and updating derived data. As with materialized-view maintenance in data warehouses, we must decide which inferences (Section 3) are made eagerly and which are made lazily, at query time, with the usual tradeoffs. However, there are important differences (in addition to the tentative nature of some inferences in an RFID system): Currency of data is typically not a major requirement for data warehouses. To the contrary, it is often desirable that the warehouse be updated only infrequently (daily, weekly) and at predictable times, so that an analyst is guaranteed a consistent working set over the duration of her study, which may last several days. In contrast, currency is very important in a typical RFID deployment. We wish to learn of the arrival of pallets at a distribution center without significant delay so that it may be acted upon (unpacked, repackaged, shipped) and the inventory turned over quickly. Thus, we need online methods for data propagation in the RFID infrastructure.

Another feature that distinguishes an RFID infrastructure from a warehousing infrastructure is the much greater emphasis on station-local activities in the former. In a typical data warehousing setup (e.g., for a department store), it is uncommon for the data to be queried at its source (e.g., the point-of-sale terminal). The emphasis is on aggregating data at a central warehouse, where it can be indexed and queried using grouping and aggregation. In contrast, data generated by the tag readers at a dock door in a distribution center is more likely to be used within the distribution center than away from it. For example, the arrival

of pallets as they are unloaded from a truck triggers tag reads that, in turn, trigger the activation of business rules and workflows directing personnel to take action on the new items. Thus, while warehousing systems can use a store-and-forward approach to data generated at point-of-sale terminals and other sources, an RFID system must enable efficient transformation (cleaning, filtering, correlation) and querying of data at the data source (as well as querying from foreign locations). For this purpose, a carefully crafted data replication and migration policy is required: A simple policy is to perform immediate (online) updates to the local database in response to tag-read and other event and to push the data to the central infrastructure (which may be composed of several distributed servers) using persistent queues [3].

5 Configuration Design

The above discussion has implicitly assumed a preexisting placement and configuration of tag readers and other hardware. In practice, determining number, type, and placement of readers, and the manner in which they are connected to other sensors (e.g., motion detectors) and actuators (e.g., conveyor belt speed controls) is part of a large design problem.

As an example, suppose we wish to use RFID tags to keep track of rare books in a large bookstore. Perhaps the most straightforward design is to assign a reader to each bookshelf in order to determine the books in its vicinity. However, the number of readers required by this design, and the implied size of higher-level infrastructure to support the data rate from them, may not be economically feasible. An alternate design is to assign readers to the points of entry and exit from aisles between bookshelves. In this case, we can infer the current location of a book based on the location of the reader that read its tag most recently. In the former case, tag readers provide state information (book x is at location y) while in the latter case, readers provide change-of-state (event) information (book x just entered aisle z). This design choice at the lower layers of the architecture (Figure 5) affects the amount and nature of data that must be stored at other layers. In the state-based design, if all past sensor readings for book x are somehow lost (perhaps due to a system malfunction) the book can still be very easily located by simply issuing a query for its EPC. In the event-based design, this option may not be available because the current location of x is out of the range of all sensors.

As another example, suppose we wish to monitor the operation of a distribution center that receives shipments of pallets from manufacturers, unpacks them, and repackages their contents into new pallets to be shipped down the distribution channel (thus regrouping items from sender-based groups to receiver-based groups). We may need to use multiple tag reader (perhaps of different classes, Figure 4) to read the tags on the pallets and those on the cases within them. We also need a method to determine when one pallet has been completed and a new one started. A method that relies purely on tag readers would be error-prone. It is preferable to use a more reliable indicator of when a pallet is done (say, an optical sensor that detects its passage beyond a certain point on the conveyor belt). If items move too fast on a conveyor belt, the readers may miss tags or produce erroneous reads. Thus, the system must be coupled with an actuator that controls the speed of the conveyor (which is also subject to other constraints, such as the rate at which a human operator can load or unload items).

While the hardware configuration (placement of readers, interconnections, etc.), is difficult to change on a frequent basis, the software configuration (manner in which readings are interpreted and routed) can be changed without much labor. This possibility provides the opportunity to rapidly incorporate new business processes into the RFID infrastructure. For example, if a batch of oversized items cannot be processed at the warehouse in the usual manner, a different physical workflow may be used (perhaps bypassing narrow hallways and expediting outbound shipping for the oversized items). A corresponding change to the electronic (RFID) workflow must also be made. Such changes affect not only the interpretation of the base data from readers but also higher-level inferences. As an example of the former, a state-based tag read for regular items may need to be interpreted as an event-based tag-read for the oversized items. As an example of the latter, the presence of oversized items in the warehouse may increase the likelihood of errors in reading tags (due to obstruction of signals).

6 Conclusion

We have described the new enterprise applications and architectures emerging as a result of RFID technology and illustrated the nature of data management problems they pose: There is a need for methods that can cope with the large variety of RFID tags and readers and their differing capabilities. An architecture for efficiently cleaning, filtering, and augmenting the raw data generated by tag readers is essential for the data to provide any real value. We discussed specific problems in inferencing, online warehousing, and configuration design.

References

[1] B. Brewin. Delta, boeing to test RFID on engine parts. *Computerworld*, pages 1,61, June 2004.

[2] P. Buneman, S. Khanna, and W.-C. Tan. Why and where: A characterization of data prove-

nance. In *Proceedings of the International Conference on Database Theory*, 2001.

[3] S. Ceri and J. Widom. Managing semantic heterogeneity with production rules and persistent queues. In *Proceedings of the Nineteenth International Conference on Very Large Data Bases*, pages 108–119, Dublin, Ireland, Aug. 1993.

[4] S. Chaudhuri and U. Dayal. An overview of data warehouse and OLAP technology. *ACM SIGMOD Record*, Mar. 1997.

[5] C. Cui and J. Widom. Practical lineage tracing in a data warehouse. In *Proceedings of the International Conference on Data Engineering*, pages 367–378, 2000.

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

[7] C. Floerkemeier, D. Anarkat, T. Osinski, and M. Harrison. PML Core specification 1.0. Auto-ID Center Recommendation. `http://develop.autoidcenter.org/`, Sept. 2003.

[8] G. Gruman. UPS v. FedEx: Head-to-head on wireless. *CIO*, pages 66–71, June 2004.

[9] A. Gupta and I. Mumick. Maintenance of materialized views: Problems, techniques, and applications. *IEEE Data Engineering Bulletin, Special Issue on Materialized Views and Data Warehousing*, 18(2):3–18, June 1995.

[10] A. Juels, R. L. Rivest, and M. Szydlo. The blocker tag: Selective blocking of RFID tags for consumer privacy. In *Proceedings of the ACM Conference on Computer and Communications Security*, Washington, D.C., Oct. 2004.

[11] T. Kontzer. RFID flies high with airlplane makers. *Information Week*, page 28, June 2004.

[12] M. Mealling. The naming authority pointer (NAPTR) DNS resource record. IETF Network Working Group Request for Comments 2915, Sept. 2000.

[13] M. Mealling. Dynamic delegation discovery system (DDDS) part one: The comprehensive DDDS. IETF Network Working Group Request for Comments 3401, Oct. 2002.

[14] M. Mealling. Auto-ID Object Name Service (ONS) 1.0. Auto-ID Center Working Draft. `http://develop.autoidcenter.org/`, Aug. 2003.

[15] P. V. Mockapetris and K. J. Dunlap. Development of the domain name system. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, pages 123–133. ACM Press, 1988.

[16] S. Sarma. Towards the five-cent tag. Auto-ID Center Technical Report. `http://develop.autoidcenter.org/`, 2001.

[17] S. Sarma and D. W. Engels. On the future of RFID tags and protocols. Auto-ID Center Technical Report. `http://develop.autoidcenter.org/`, June 2003.

[18] H. Stockman. Communication by means of reflected power. In *Proceedings of the IRE*, pages 1196–1204, Oct. 1948.

[19] L. Sullivan. Alzheimer's patients get help at home. *Information Week*, page 32, June 2004.

[20] K. Takaragi, M. Usami, R. Imura, R. Itsuki, and T. Satoh. An ultra small individual recognition security chip. *IEEE Micro*, 21(6):43–49, 2001.