A Policy Based Event Management Middleware for Implementing RFID Applications

M. E. Ajana[1], M. Boulmalf[1], H. Harroud[1], and H. Hamam[2]
1: Alakhawayn University in Ifrane, Morocco
2: Moncton University, New Brunzwick, Canada
M.Ajana@aui.ma

Abstract— **Radio Frequency Identification (RFID) has become a popular identification technology in a number of application areas such as supply chain management. A dedicated middleware solution is required to achieve the maximum benefits of RFID technology. The middleware components serve to abstract the communication between the middleware and the different types of sensing devices on one hand, and the middleware and backend applications on the other hand. FlexRFID is a simple and smart RFID middleware which provides device management and monitoring, data processing, filtering, and aggregation, rapid application development, as well as a policy based business rules layer which helps applying rules for accessing and configuring the services provided by the middleware. This layer serves to process some business intelligence rules locally so that the host system is offloaded from those mundane tasks. The paper shows that FlexRFID is a highly scalable and easily deployable middleware in the heterogeneous sites based on different standards and consisting of different hardware. Apart from these, FlexRFID incorporates the mechanisms for supervision, testing, and control of its components, plus handles the security and privacy issues that inhibit the adoption of RFID technology by applying the privacy policies in the business rules layer. FlexRFID middleware is controlling the data flows in and out as well as locally controlling some intelligence.**

Keywords: **RFID, middleware, FlexRFID, policy-based event management**

I. Introduction

Radio Frequency Identification (RFID) is a form of *Automatic Identification and Data Capture* (AIDC) [1] technique that uses radio waves to automatically identify people or other objects. In contrast to other automatic identification approaches such as *barcode* based systems [1] and *Optical Character Recognition* systems (OCR) [2], RFID technology does not require line of sight readings, can read multiple tags simultaneously, and store large amounts of data in addition to the ID of the object tracked. RFID has been commonly used in a wide range of areas such as healthcare, access control, pet identification, traffic monitoring, passports, human implants and supply chain management (SCM) [3]. The integration of RFID technology in the SCM systems has helped in monitoring and controlling inventory, and resulted in the reduced losses, increased Return on Investment (ROI), and improved visibility in various stages of SCM [4].

Depending on the scope of various mandates and the degree of compliance required, the benefits of using RFID technology will begin to accrue in phases. An obvious payback is expected in the form of increased labor, facility/equipment productivity, process improvements throughout the supply chain, reduction in theft and reduced inventory, and other benefits.

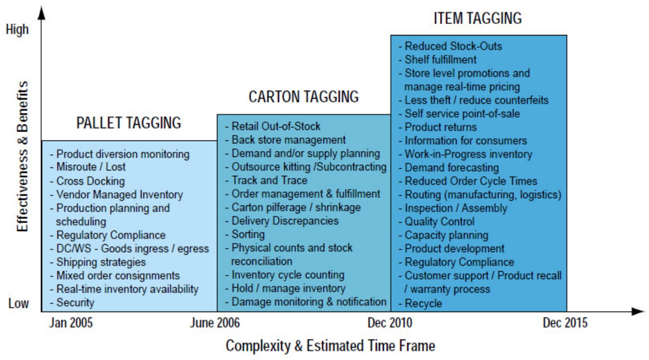


Fig. 1. RFID benefits across supply chain activities [12]

The Figure 1 illustrates the various stages and the time line during which a number of enterprise activities will be benefited [12].

A typical RFID system basically consists of three elements: a tag/transponder, a reader and a middleware deployed at a host computer as shown in Figure 2. The *RFID tag* is a data carrier part of the RFID system which is placed on the objects to be uniquely identified. Unique identification or electronic data stored in RFID tags can consist of serial numbers, security codes, product codes and other object specific data. The available RFID tags in today's market could be classified with respect to different parameters. For example with respect to powering, tags may be *passive*, *semi-passive*, and *active* [4], [9]. In terms of access to memory, the tags may be *read-only*, *read-write*, *Electrically Erasable Programmable Read-Only Memory*, *Static Random Access Memory*, and *Write-once read-many* [9], [6]. Tags have also various sizes and shapes and may be classified with respect to these geometrical parameters. The *RFID reader* is a device that transmits and receives data through radio waves using the connected antennas. RFID reader can read multiple tags simultaneously without line-of-sight requirement, even when tagged objects are embedded inside packaging, or even when the tag is embedded inside an object itself. RFID readers may be either *fixed* or *handheld*, and are now equipped with *tag collision*, *reader collision* prevention and tag-reader authentication techniques [5]. A variety of RFID tags and readers combined with decreasing RFID hardware prices, are

978-0-7695-3841-9/09 $26.00 © 2009 IEEE
DOI 10.1109/WiMob.2009.75

making RFID deployment more attractive [5]. The RFID reader functions include powering the tag, and reading/writing data to the tag. As shown in Figure 2, the signals sent by the reader's antennas are conveyed by an electromagnetic field along the interrogation zone. When a tag enters this zone, it gets activated to exchange data with the reader [6]. Later, the identification data read by the RFID reader is processed by the software system, known as the RFID middleware. The *RFID middleware* refers to the software layer which resides between the physical layer components (hardware), operating systems or firmware, which deal with communication protocols and low level system calls, and the upper layer standalone or distributed enterprise applications generally interacting via the network. The RFID middleware manages readers, as well as filters and formats the RFID raw tag data so that they can be accessed by the various interested enterprise applications [7]. Hence, the middleware is a key component for managing the flow of information between tag readers and enterprise applications [8]. RFID tags and reader that work at different frequencies are currently available and their selection mainly depends on the requirements of the backend application.

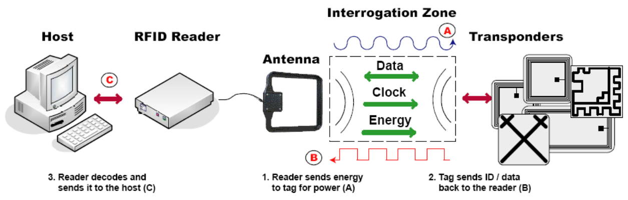

Interrogation Zone
Host
RFID Reader
Transponders
Antenna
A
C
Data
Clock
Energy
B
3. Reader decodes and sends it to the host (C)
1. Reader sends energy to tag for power (A)
2. Tag sends ID / data back to the reader (B)

Fig. 2. RFID system components [5].

There was barely a need for RFID middleware in the traditional RFID applications such as access control, because networking among RFID readers was not a concern. However, in the novel application areas such as SCM, a number of RFID readers are used to capture RFID data which is then disseminated to a variety of interested applications. Hence, there is no longer a one-to-one relationship between reader and application [10] and therefore a strong need for RFID middleware. RFID middleware gives better device control e.g. understanding how well the reader is reading, how a reader can be added to an existing host application without having to do anything at the host by doing it at the middleware layer, and knowing where data is coming from and how it is going to get to the output system. Good middleware architectures should enable the users and system designers to infer intelligent and informed decisions from the raw RFID data coming from the readers. The success of the middleware design depends on how well the different pieces of the middleware in different layers fit together and work to provide valuable processed data to the enterprise applications.

We developed a multi-layered middleware called FlexRFID using .Net framework. FlexRFID is a simple and robust design which addresses the above aspects. We described in [13] the FlexRFID middleware architecture, and analyzed the extent to which it satisfies applications' needs and allows for an easy management of devices. An application for library management called *Smart Library* was also developed using the support provided by the FlexRFID middleware. Herewith we give more details about the *Business Rules Layer* of the FlexRFID which is a policy based management service provided by the middleware, and show the benefits of using policies that enable applications to define their own rules for event detection.

The rest of the paper is organized as follows. Section II showcases research work related to middleware designs. Section III gives an overview of the FlexRFID middleware architecture. Section IV describes the FlexRFID middleware Business Rules Layer in details, followed by conclusions and future work in Section V.

II. Related Works

There has been some research work involving proposals for middleware design and RFID data processing. The *Auto-ID Center* has developed a middleware component called Savant [5] that is a framework to manage EPC data throughout the enterprise. The Savant is a data router that performs operations such as data capturing, monitoring and transmission. It adjusts multiple readings of a tag, and performs tasks such as archiving data, and inventory control [1]. Savant middleware architecture has three key elements: *Event Management System* (EMS), *real-time in-memory data structure* (RIED), and *Task Management System* (TMS) [1]. The EMS enables adapters to be written for various types of readers, collects EPC data from readers in a standard format, enables filters to be written to smooth or clean EPC data, enables various loggers to be written, such as database loggers to log EPC data into the database, and buffers events to enable loggers, filters and adapters to operate without blocking each other [11]. The RIED is an in-memory database that can be used to store event information by EMS. The EMS provides a framework to filter and log events. The loggers can log events in a database. However, databases cannot handle more than a few hundred transactions per second. RIED provides the same interface as a database, but offers much better performance. RIED can also maintain "snapshots" of the database at different timestamps. The maintenance of old snapshots is required for some applications. The TMS provides an external interface to schedule tasks, a platform-independent (Java) virtual machine with uniform libraries loaded on-demand from redundant class servers, and a robust scheduler maintaining persistent information about tasks, and capable of restarting tasks on a Savant crash or task crash. While the Savant middleware architecture provides features for cleaning the data and interfacing with different kinds of RF readers, it has limited built-in functionality for addressing business rules management, dealing with all types of sensor devices and providing data dissemination, filtering, and aggregation [15].

WinRFID [12] developed at the *University of California Los Angeles* (UCLA), is another middleware architecture that uses web services and enables rapid RFID application development. It is a multi-layered middleware that consists of five main layers. The *physical layer* deals with the hardware consisting of readers, and tags. It deals with abstraction of three elements of the RFID infrastructure, readers, tags and the

I/O module of the readers. The *protocol layer* abstracts the reader-tag protocols. It is also abstracted to wrap the command syntax and semantics of a variety of published protocols such as ISO 15693, ISO 14443, ISO18000–6A/B, ICode, EPC Class 0 and EPC Class 1. It deals with protocol specifics such as byte based, block or even page reading and writing, structure and length of the command frames, partitioning of the tag memory space, checksums, etc. The *data processing* layer deals with processing data streams generated by the network of readers. The *XML framework* layer formats the cleaned tag data in a variety of ways to a higher level XML based representation. The *data presentation* layer presents this data as per the requirements of end-users or different applications requirements [12]. WinRFID exploits the .Net framework's runtime plug-in feature to support the addition of new readers, protocols, and data transformation rules with minimum disruption of the existing infrastructure [4].

The *IBM WebSphere* RFID solution is a sensor-enabled product enabling the aggregation and analysis of sensor data, identification of insights from that data and integration of those insights with the business processes implemented in a *Service Oriented Architecture* (SOA) environment. This solution consists of three components: *RFID devices*, *WebSphere Premises Server*, and *Websphere Business Integration Server* [14]. It expands device services allowing a single platform to support multiple sensor types, and supports workflow tooling for sensor data integration with business processes [14]. None of WinRFID and IBM WebSphere RFID solution considers business rules policies implementation, especially the ones concerned with security and privacy.

There are still many open issues concerning middleware designs. The reliability of RFID data needs to be improved since inaccurate data could misguide the application users. The accumulation of RFID data generated in high volumes, may lead to slower queries and updates, therefore efficient RFID data management solutions such as data transformation, aggregation, and dissemination should be investigated. Raw RFID data is not of significant value until it is aggregated with other data to obtain appropriate inferences and transformed into a suitable form for application level interaction. The applications with high security requirements are increasingly using RFID. Therefore support for data security and confidentiality is needed. However, such support should maintain a desirable system performance. RFID also raises the privacy concerns because of its potential to leak proprietary information and ability to track private information such as spending history of a consumer [4].

As compared to the related work described herewith, the FlexRFID middleware addresses the issues mentioned above, ensures that private data is not compromised thanks to a policy based business rules layer, and provides the following distinguishing aspects: the FlexRFID design aims at providing the applications with a device neutral interface to communicate simultaneously with many different hardware devices, creating an intelligent RFID network. It also provides an interface to access the hardware for the management and monitoring purposes. The FlexRFID provides all data processing capabilities along with the security and privacy features included in the data processing layer and enforced by a policy based management module for the business events, referred to as the Business Rules layer. The modular and layered design of FlexRFID allows integrating new features e.g. policies with little effort. The design also enables seamless integration of different types of enterprise applications. Finally, the utility of the FlexRFID middleware architecture is shown by providing examples of rules based on policies.

III. Overview of FlexRFID Middleware Architecture

As shown in Figure 3, FlexRFID is organized as a three-tier architecture consisting of *backend applications layer*, *FlexRFID middleware layer*, and *hardware layer* consisting of diverse types of sensors and devices.

The *Diverse Types of Sensors and Devices layer* comprises RFID readers, sensors and other industrial automation devices. Such approach offers incredible flexibility in the selection of devices, enables companies building their enterprise solutions without handling low-level programming, and allows creating an intelligent sensor network, where RFID readers are choreographed with other devices. There are diverse makes and models of devices requiring a middleware layer that monitors, manages, coordinates and obtains data from different devices. In FlexRFID, these functions are carried out before processing raw data and applying business logic to them. Our approach consists in using Device Abstraction Layer (DAL) that abstracts the interaction with the physical network of devices. The FlexRFID middleware incorporates three other layers which are: Business Event and Data Processing Layer (BEDPL), Business Rules Layer (BRL), and Application Abstraction Layer (AAL) [13].

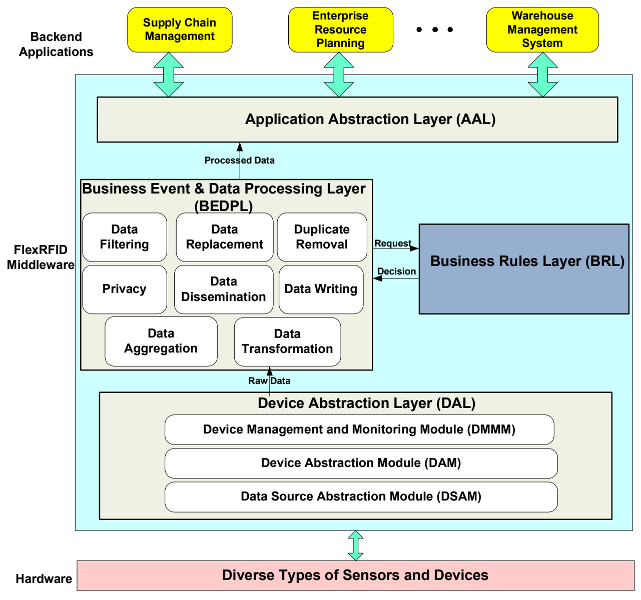

Backend Applications
Supply Chain Management
Enterprise Resource Planning
• • •
Warehouse Management System
Application Abstraction Layer (AAL)
Processed Data
Business Event & Data Processing Layer (BEDPL)
Data Filtering
Data Replacement
Duplicate Removal
Privacy
Data Dissemination
Data Writing
Data Aggregation
Data Transformation
Request
Decision
Business Rules Layer (BRL)
FlexRFID Middleware
Raw Data
Device Abstraction Layer (DAL)
Device Management and Monitoring Module (DMMM)
Device Abstraction Module (DAM)
Data Source Abstraction Module (DSAM)
Hardware
Diverse Types of Sensors and Devices

Fig. 3. FlexRFID middleware architecture [13].

The *Device Abstraction Layer* (DAL) of the FlexRFID middleware is responsible for interaction with various data sources and devices independent of their characteristics through the *Data Source Abstraction Module* (DSAM) and the *Device Abstraction Module* (DAM) respectively. Both, the DSAM and the DAM offer extendibility to FlexRFID middleware to support various data sources and devices. The *Device Management and Monitoring* Module (DMMM) of the DAL is responsible for dynamic loading and unloading of the driver libraries or device adaptors, as well as configuration, monitoring and status reporting of the devices.

The *Business Event and Data Processing Layer* (BEDPL) acts as a mediator between the DAL and the AAL. Services provided by the BEDPL include: data dissemination, data aggregation, data transformation, data filtering, duplicate removal, data replacement, data writing and privacy management [protection]. The services accepted by the BEDPL are first authorized by the *Business Rules Layer* (BRL) and then permitted to issue commands to the DAL to access raw data and process it accordingly. Similarly raw data is carried from the DAL, processed, and passed on to the AAL by this layer.

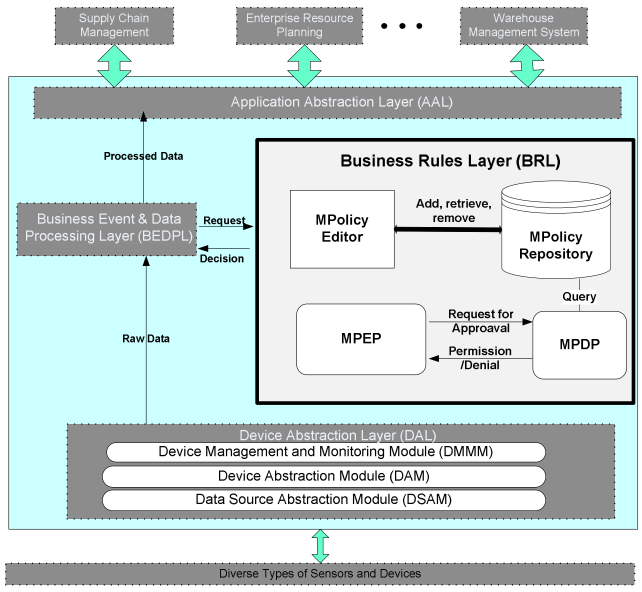
The BRL is a policy-based management engine that defines the rules granting or denying access to resources and services of the FlexRFID middleware. This is achieved by determining the policies to apply when an application requests access to a service in the BEDPL. Components of this layer are described in detail in the Section IV.

The *Application Abstraction Layer* (AAL) provides various applications with an interface to the hardware devices, through which the applications request the set of services provided by the FlexRFID middleware with hidden complexity. This layer provides a high level of software abstraction that allows for communication among the enterprise applications and the FlexRFID middleware.

IV. FLEXRFID POLICY BASED BUSINESS RULES LAYER

Figure 4 below shows the BRL components. The *Middleware Policy Editor* (MPE) allows storing, retrieving, and removing policies from the *Middleware Policy Repository database*. If an application needs to access a service that is protected by the Business Rules Layer, the request goes through the *Middleware Policy Enforcement Point* (MPEP) which asks the *Middleware Policy Decision Point* (MPDP) whether to permit or deny access to the service. The MPEP gives the MPDP the authority of decision making whether or not to grant the application access to the service based on the description of the application attributes. The MPDP makes its decision based on the applicable policies stored on the system. The returned decision is *Permit*, *Deny*, *Indeterminate* or *Not Applicable*. Indeterminate is returned if there is an error in processing the request and Not Applicable if no policy that applies to the request could be found.

Policies are operating rules used to maintain order, security, consistency, or other ways of successfully achieving a task. Examples of policies included in the Business Rules Layer are: Access policy, data replacement policy, quality of service policy, and privacy policy.



Supply Chain Management
Enterprise Resource Planning
Warehouse Management System
Application Abstraction Layer (AAL)
Processed Data
Business Rules Layer (BRL)
Business Event & Data Processing Layer (BEDPL)
Request
Decision
MPolicy Editor
Add, retrieve, remove
MPolicy Repository
Query
Raw Data
MPEP
Request for Approaval
Permission /Denial
MPDP
Device Abstraction Layer (DAL)
Device Management and Monitoring Module (DMMM)
Device Abstraction Module (DAM)
Data Source Abstraction Module (DSAM)
Diverse Types of Sensors and Devices

Fig. 4. FlexRFID Business Rules Layer (BRL) components.

Different types of applications using the FlexRFID middleware may define rules to detect events and process them using the services provided by the middleware. *Primitive events* such as observations from readers may lead to actions such as change of location. *Sequence events* consist of a sequence of primitive events of the same type, defined by the order and closeness of intervals. *Composite events* are a combination of primitive events and sequence events, and may lead to actions such as aggregation of data. Here we present some examples of rules enforced by their corresponding policies.

The *filtering rule* filters data according to predefined policies by the applications. For example, multiple readers may generate duplicate readings. To filter this, the filtering policy will scan data within a sliding window to find if there are duplicate RFID tag readings from multiple readers, and delete the duplicate if it exists. A policy for *duplicate removal* could specify that if readings from reader Rx and Ry have the same tag ID value within T, then one of them is dropped.

The *location transformation rule* serves to transform RFID readers' observations into location changes. For example, Reader R1 is mounted at a warehouse departure zone and will scan objects before their departure. A policy for this transformation could state that any observation generated from reader R1 will change the object's location to a value different from its current location.

The *data aggregation rule* is used to detect a sequence of ordered events and generate an aggregation relationship. For instance when pallets are loaded into a truck to depart, a sequence of readings on the pallets are done, followed by (with a distinctive distance) a separate reading of the truck's EPC. This sequence of events will aggregate as a *containment relationship* between the pallets and the truck.

Privacy threats in an RFID application can include covert reading, tracking over time, and individual profiling. The *privacy rule* specifies whether an application has the right to access RFID tag data, can track it over time, and use it to generate events. Applications can load into the FlexRFID middleware's Business Rules Layer privacy policies specifying how to use and configure the RFID technology to maintain the privacy of data and prevent it from tracking and hotlisting.

V. Conclusion and Future Work

FlexRFID is a RFID technology middleware addressing diverse applications requirements. It has four important layers: the Device Abstraction Layer (DAL), the Business Event and Data Processing Layer (BEDPL) and the Application Abstraction Layer (AAL). In this paper, we gave a brief overview of FlexRFID middleware components, and focused on the Business Rules Layer which uses policies to enforce access to the services provided by FlexRFID. FlexRFID uses an architecture hiding proprietary device interfaces, facilitating configuration and monitoring of the devices and processing the data captured through services that ensure the business rules using policy-based event management.

In the future work, we propose the following extensions to the FlexRFID design: evaluating the performance of FlexRFID middleware, and based on the results obtained, we intend finding a mechanism to reduce the response time and improve memory utilization, including additional device abstraction layer drivers in order to increase the range of supported devices, extending the FlexRFID middleware to support the EPC standard, developing a complete system using multiple devices and evaluating it with multiple hardware configurations and applications requirements and developing a prototype application for supply chain management using the features of FlexRFID middleware.

Acknowledgment

We would like to express our sincere appreciation to the Academics Affairs at Al Akhawayn University in Ifrane for the financial support of this research work.

References

[1] Ishikawa T., Yumoto, Y., Kurata, M., Endo, M., Kinoshita, S., Hoshino, F., Yagi, S., Nomachi, M. "Applying Auto-ID to the Japanese Publication Business to Deliver Advanced Supply Chain Management, Innovative Retail Applications, and Convenient and Safe Reader Services," Auto-ID Center, Keio University, Oct. 2003.

[2] Phoenix Software International. "Optical Character Recognition (OCR): What You Need to Know," 2006. Online. Available: http://www.phoenixsoftware.com/pdf/ocrdataentry.pdf

[3] S. Polniak, The RFID Case Study Book: RFID Application Stories from Around the Globe. Abhisam Software, 2007.

[4] Q. Sheng, X. Li, and S. Zeadally, "Enabling Next-Generation RFID Applications: Solutions and Challenges", IEEE Computer, Vol. 41, No. 9, September 2008.

[5] D. J. Glasser, K. W. Goodman, and N. G. Einspruch, "Chips, tags and scanners: Ethical challenges for radio frequency identification," Ethics and Information Technology, v.9 n.2, p.101-109, July 2007

[6] H. Al-Mousawi, "Performance and reliability of Radio Frequency Identification (RFID)", in Agder University College, June 2004, Faculty of Engineering and Science. [Online]. Available: http://student.grm.hia.no/master/ikt04/ikt6400/g28/Document/Master_Thesis.pdf

[7] C. Floerkemeier and M. Lampe, "RFID middleware design – addressing application requirements and RFID constraints," in Proceedings of SOC'2005 (Smart Objects Conference), Grenoble, France, Oct. 2005, pp. 219–224.

[8] J. Burnell, "What Is RFID Middleware and Where Is It Needed?," ALX Technologies, 2008. [Online]. Available: http://www.rfidupdate.com/articles/index.php?id=1176

[9] United States Government Accountability Office, "INFORMATON SECURITY Radio Frequency Identification Technology in the Federal Government," United States Government Accountability Office, May 2005. [Online]. Available: http://epic.org/privacy/surveillance/spotlight/0806/gao05551.pdf

[10] C. Floerkemeier, C. Roduner, and M. Lampe, "RFID Application Development with the Accada middleware Platform," IEEE Systems Journal, Vol. 1 No. 2, Dec. 2007, pp. 82-94.

[11] Sun Microsystems. "Sun's Auto-ID Architecture, "June, 2006. Online. Available: http://www.fda.gov/ohrms/dockets/dailys/03/Nov03/110503/03N-0361-emc-000025-03.pdf

[12] B. S. Prabhu, X. Su, H. Ramamurthy, C. Chu, and R. Gadh, "WinRFID – A Middleware for the enablement of Radio Frequency Identification (RFID) based Applications," Wireless Internet for the Mobile Enterprise Consortium (WINMEC), Los Angeles, Dec. 2005.

[13] M. E. Ajana, H. Harroud, M. Boulmalf, and H. Hamam, "FlexRFID: A Flexible Middleware for RFID Applications Development," 6th International Wireless and Optical Networks Communications (WOCN) Conference, Cairo, Egypt, Apr. 2009.

[14] IBM, "WebSphere Premises Server". [Online]. Available : http://www-01.ibm.com/software/integration/premises_server/index.html

[15] Oat Systems & MIT Auto-ID Center, "Technical Manual: The Savant version 0.1 (Alpha)," May 2002. [Online]. Available: http://www.autoidlabs.org/uploads/media/MIT-AUTOID-TM-003.pdf