FlexRFID: A Flexible Middleware for RFID Applications Development

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

Abstract— **Nowadays, Radio Frequency Identification (RFID) is a popular automatic wireless identification technology, and provides promising benefits in a number of application areas such as supply chain management. Though RFID technology has attracted a significant attention due to the convergence of lower cost and increased hardware capabilities, there still exist major hurdles for the practical realization of its benefits. To achieve the maximum benefits of RFID technology, a dedicated middleware solution is required for managing and monitoring RFID readers or other types of sensing devices, as well as processing dynamically generated high volumes of noisy RFID data. Apart from these, such middleware should possess useful contextual characteristics such as implicit semantics, and support rapid RFID based application development. FlexRFID as reported in this paper is a simple and smart RFID middleware which satisfies the requirements mentioned earlier. 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.**

I. INTRODUCTION

Radio Frequency Identification (RFID) is a form of *Automatic Identification and Data Capture* (AIDC) [1] technique. RFID is recently being used in a wide range of areas such as Supply Chain Management (SCM), health care, retail, and access control [2]. The ability to store large amounts of data and identify items which are not in the line of sight has given RFID technology an edge over other automatic identification approaches such as the barcode based systems [1] and optical character recognition systems (OCR) [3]. As an example, RFID technology integration in the SCM systems has resulted in the reduced losses, and improved visibility in various stages of SCM [4].

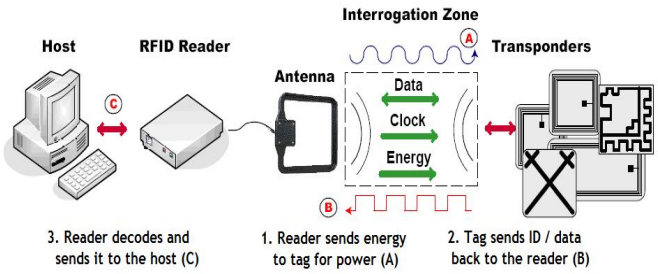

Fig. 1. RFID system components [5].

As shown in the Fig. 1, the RFID systems basically consist of three elements: a tag/transponder, a reader and a middleware deployed at a host computer. The *RFID tag* is a data carrier part of the RFID system which is placed on the objects to be uniquely identified. The *RFID reader* is a device that transmits and receives data through radio waves using the connected antennas. Its functions include powering the tag, and reading/writing data to the tag. As shown in the Fig. 1, the signals sent by the reader's antennas form an interrogation zone made up of an electromagnetic field. 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 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 technology will be ubiquitous in the near future as RFID systems have recently undergone significant improvements. The available RFID tags in today's market could be classified depending on different parameters. E.g. tag type could be one of *passive*, *semi-passive*, and *active* [4], [9]; the tag may have various types of memory e.g. *read-only*, *read-write*, *Electrically Erasable Programmable Read-Only Memory, Static Random Access Memory*, and *Write-once read-many* [9], [6] and also various sizes and shapes. RFID readers may be either fixed or handheld, and are now equipped with tag collision, and reader collision prevention techniques [5]. A variety of RFID tags and readers combined with decreasing RFID hardware prices, are making RFID deployment more attractive [5].

Since networking is not a concern in the traditional RFID applications like access control, there is barely a need for RFID middleware. However, in the novel application areas such as SCM, a number of RFID readers could be used to capture RFID data which need to be disseminated to a variety of applications. Hence, there is no longer a one-to-one relationship between reader and application [10].

The researchers in this area have reported a vast amount research (e.g. [8], [11], [12]) about the benefits, possible misuses and ethical issues (e.g. privacy) involved in the RFID technology. However, no significant attention has been paid to the issues involved in the RFID middleware that manages large deployments of readers producing high volumes of

978-1-4244-3474-9/09/$25.00 ©2009 IEEE

captured data, and encapsulates applications from the low level data by transforming them into more meaningful events [8]. Considering this void in the RFID middleware research, herewith we present a simple and robust design of a middleware called *FlexRFID* which addresses the above aspects. We analyze FlexRFID to the extent which it addresses applications' needs and allows an easy management of devices.

The rest of the paper is organized as follows. Section II showcases some research work related to RFID middleware designs. Section III describes the FlexRFID middleware architecture. Section IV presents a smart library application built using the support provided by the FlexRFID middleware, followed by conclusions and future work in Section V.

II. Related Work

There have been some proposals and research work involving middleware design and RFID data processing. The Auto-ID Center has developed a middleware component called *Savant* [5] that collects, accumulates, and processes Electronic Product Code (EPC) data obtained from several RF readers. 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) that provides a Java API for different types of RF readers, real-time in-memory data structure (RIED) that manages event information and Task Management System (TMS) that provides an interface for task management [1]. 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.

WinRFID [13] 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, tags and other sensors. The protocol layer abstracts the reader-tag protocols. 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 [13]. 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 *WebSphere* RFID solution by IBM consists of three components: RFID devices, WebSphere premises server, and Websphere Business Integration Server [14]. This solution 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 the business rules policies implementation, especially the ones concerned with security and privacy.

All of these middleware designs aim to provide a scalable solution for gathering, filtering, and providing clean RFID data to the end-user. However, there are still many open issues. 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. Technical solutions must be implemented to ensure that private data is not compromised with [4].

As compared to the related work described herewith, the distinguishing aspects of the proposed FlexRFID middleware are as follows: the FlexRFID design aims to provide 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 integration of new features with little effort. The design also permits seamless integration of different types of enterprise applications. Finally, the utility of the FlexRFID middleware is shown by developing an application, the smart library.

III. FlexRFID Middleware Architecture

The FlexRFID middleware architecture takes into account the design issues discussed in the Section II. As shown in Fig. 2, 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 allows incredible flexibility in the selection of devices, lets companies build 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, which require a middleware layer that monitors, manages, coordinates, and obtains data from the

different devices. In FlexRFID, these functions are taken care of before processing the raw data and applying business logic to them. Our approach is to use a *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).

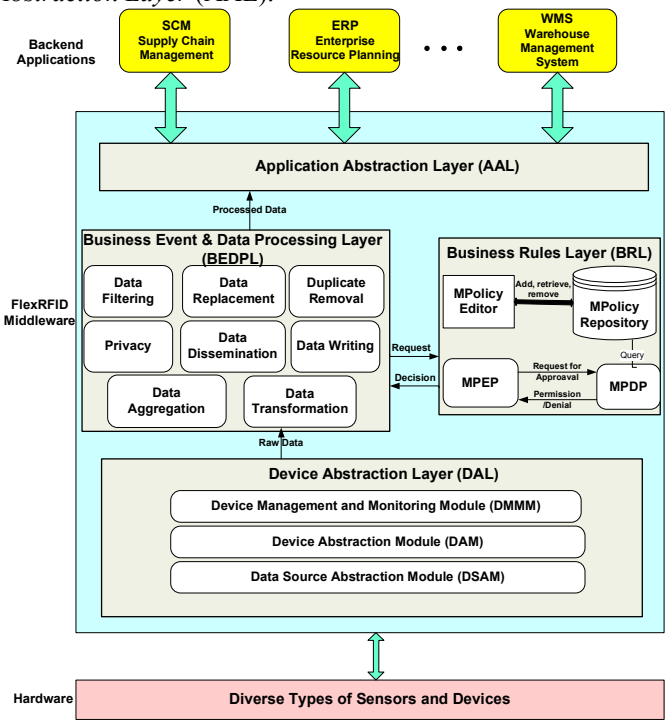

Fig. 2. FlexRFID middleware architecture.

A. *Device Abstraction Layer (DAL)*

The Device Abstraction Layer of the FlexRFID middleware is responsible for interaction with various devices and data sources independent of their characteristics. The *Data Source Abstraction Module* (DSAM) of the DAL provides a standard view of data regardless of the data source protocol (e.g. EPC Gen2, ISO 15693, ISO14443A), air interface (e.g. UHF, HF), power supply, type, and memory size of a device. The *Device Abstraction Module* (DAM) of the DAL provides a common interface to access hardware devices with different characteristics such as protocols, air interface, and host-side communication interface (e.g. USB, Serial Port, Ethernet port). The DAM exposes simple functions like open, close, read, write, etc. that trigger the complex operations of the devices. Both, the DSAM and the DAM allow the FlexRFID middleware to be extendable 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. This allows the FlexRFID middleware to be light weight as libraries are loaded based upon request. The DMMM configures the devices as specified by the upper layers, and also monitors and reports their status.

B. *Business Event and Data Processing Layer (BEDPL)*

The BEDPL acts as a mediator between the DAL and the AAL. The services accepted by the BEDPL are first authorized by the Business Rules Layer (BRL) and then allowed to issue commands to the DAL in order to get the raw data and process them accordingly. Similarly the raw data are carried from the DAL, processed, and passed on to the AAL by this layer. Services provided by the BEDPL are described in the following:

Data dissemination: A diverse set of applications across an organization are interested in the captured information. The captured data is therefore broadcasted by the data dissemination service to all the interested entities. In addition, different applications require different latencies. For example, low latency for the notifications is desired by the applications that need to respond immediately to objects' events. In contrast, some legacy applications need to receive batched updates on a daily schedule.

Data aggregation: The fine-grained data has implicit meanings and associated relationships with other data, and need to be aggregated into summaries and/or proper inferences for applications that can not deal with the increased granularity. For example, it is common that an application is only interested in an event when an object enters or leaves a certain area. Other applications may only need a total count of objects belonging to a specific category rather than a serial number of each object detected. The data aggregation service provides such kind of functionality.

Data transformation: Raw data present little value until they are transformed into a form suitable for application-level interactions. So, from an application perspective, it is desirable to provide a mechanism that turns the low-level captured data into the corresponding business event. For example, a detection of a number of tagged books at the exit door of a library can be automatically translated into a books checked out event. This requirement is taken care by the data transformation service.

Data filtering: The volumes of data generated by the different devices require significant data filtering to extract the most important information. Also, different applications are interested in different subsets of data captured. There are filtering policies available in the FlexRFID middleware policy repository of the BRL, therefore the data filtering service filters data depending on the filter characteristics provided by the application. This offers flexibility in handling multiple filtering formats.

Duplicate removal: Multiple devices may generate duplicate readings of the data, for example tags in the vicinity of a RFID reader are read continuously. Since this result in a large amount of repeated data, duplicate removal service prevents reporting of duplicate data. The application specifies a time window, so that the same data read within it are only reported once.

Data replacement: Usually the rate at which the devices insert data in the channel buffer is slower than the read rate of the applications. However, in case the application is not responsive enough or not executing, the channel buffer gets full, and leads to buffer overflow problem. The data replacement service allows the application to specify the action to be taken in case of channel buffer overflow. The application specifies the data replacement policy stored in the BRL policies repository, which will be executed by the data replacement service.

Data writing: Certain special data sources like RFID tags provision additional memory space for both ID and additional data. The FlexRFID middleware handles both the reading and writing of data to this additional memory.

Privacy: RFID based tracking solutions could trigger RFID tags attached to the personal belongings to reply with their ID and other private information, therefore increasing the potential of unauthorized surveillance mechanism that would pervade large parts of our lives. FlexRFID design supports dedicated privacy enhancing feature through the privacy module. The business rules of this module are stated in the privacy policy of the BRL.

C. Business Rules Layer (BRL)

The BRL is a policy-based management engine that defines the rules that grant or deny 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. The *Middleware Policy Editor* (MPE) allows storing, retrieving, and removing policies from the *Middleware Policy Repository* database. When an application needs to access a service that is protected by the Business Rules Layer, the request passes 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 when there is an error in processing the request and Not Applicable when 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 that should be available in the Business Rules Layer are: *Access policy*, *data replacement policy*, *quality of service policy*, and *privacy policy*.

D. Application Abstraction Layer (AAL)

The Application Abstraction Layer 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 communication among the enterprise applications and the FlexRFID middleware.

IV. APPLICATION: SMART LIBRARY

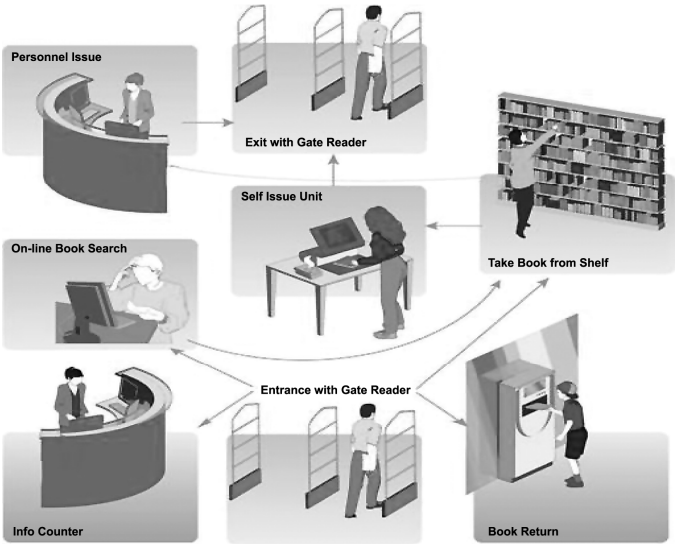

Fig. 3. RFID use in library [15].

In the late 1990s, libraries began using RFID systems to replace their electro-magnetic and barcode systems. In North America approximately 130 libraries are using RFID systems, and hundreds more are considering it [16]. Fig. 3 shows RFID uses in a library management system. The RFID self-check systems are increasingly becoming popular since they allow patrons to check-in or check-out many items, rather than one at a time. This reduces the number of library staff needed at the circulation desk. Inventory related tasks could also be done in a fraction of the time, as a portable reader can read a whole shelf of books, and then report which are missing or misplaced. Moreover, as books are dropped in the book return station, the reader reads the tag and uses the automatic sorting system to return the book back to the shelves. A RFID tag can be used for both identifying items and securing them, and there is no need to purchase additional tags for security or use security strips separately. As patrons leave the library, the tags are read to ensure that the items have been checked out [16]. If the item is not checked-out, the RFID readers placed near the exit detect the presence of the tag and sound an alarm.

A significant impediment to library use of RFID is privacy concerns associated with an item-level tagging. The tag contains static information that can be easily accessed by unauthorized readers. The privacy issues are generally described as *tracking* and *hotlisting*. Tracking refers to the ability to track the item movement or the person carrying the item by correlating multiple observations of the item's RFID tag. Hotlisting allows building a database listing the items and their corresponding tag numbers and then using an unauthorized reader to get who is checking out items on the list [16]. Therefore, libraries implementing RFID should use and configure the technology to maintain the privacy of patrons.

Smart library management applications require data to be automatically read, analyzed and written back. Every patron is issued a RFID tagged library card that stores both personal

information and information of the library items borrowed. Upon borrowing an item, the patron card is checked if he/she is permitted to borrow. Then, depending on the permissions, the application updates the borrowing status of the patron and the internal library database or rejects the request.

We are currently developing a smart library RFID prototype using FlexRFID that provides services to borrowers without having to go through an employee at the library. This prototype also is aimed at helping library staff to track items placed at the wrong places, and identify most read documents in the library. This will allow the visualization of important events and alerts in real time. The most important events are: *item check-in, item check-out, shelf management, and item theft.*

In order to illustrate the value and maturity of the FlexRFID middleware, the smart library prototype makes use of its services such as filtering, duplicate removal, transformation, aggregation and is tested with different devices such as bar code readers, RFID readers, and sensors. A solution to the security and privacy concerns is also provided by the FlexRFID's security and privacy modules managed by policies.

The smart library prototype is developed using Microsoft Visual Studio .Net 2005. The prototype is coded using C# as a language and up to now it uses only the Data Writing, Data Replacement, and Duplicate Removal modules of the FlexRFID BEDPL module. Further modules of FlexRFID are still under development. The hardware used in testing the prototype consists of Intermec IF4 fixed RFID reader, Intermec 915 MHz ID Card, Intermec passive tags, and sensors used to initiate and stop the reading of tags at the entry/exit point of the library.

V. CONCLUSION AND FUTURE WORK

A number of enterprise applications using RFID technique introduce a need for an infrastructure that hides proprietary device interfaces, facilitates configuration and monitoring of the devices, and processes the captured data. In this paper we presented the design of the FlexRFID middleware framework that addresses these application requirements. FlexRFID has four important layers: the Device Abstraction Layer (DAL), the Business Event and Data Processing Layer (BEDPL), and the Application Abstraction Layer (AAL). FlexRFID enables the following: communication with different types of devices; implementation of functionalities by ensuring the business rules using policy-based management; and seamless integration of various enterprise applications. The ongoing development of a smart library application is presented to show the usefulness of the designed middleware solution.

In the future work, we propose the following extensions to the FlexRFID design: performance improvement of the middleware by reducing the response time of the application interface and improving memory utilization; inclusion of additional DAL device drivers to increase the range of supported devices; and extension of the middleware to support EPC standard. We also plan to develop a complete system using various devices and evaluate it extensively with multiple hardware configurations, and applications requirements.

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

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

[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] D. Molnar, and D. Wagner, "Privacy and Security in Library RFID: issues, practices, and architectures," in proc. Of 11th ACM conference on Computer and Communication Security, Washington, DC, USA, Oct. 2004.

[12] Parliament Office of Science and Technology, "RADIO FREQUENCY IDENTIFICATION (RFID)", July 2004, Number 225, pp. 1-4. [Online]. Available: http://www.parliament.uk/documents/upload/postpn225.pdf

[13] 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.

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

[15] Bibliotheca RFID Library Systems, 2008. [Online]. Availabe: http://www.bibliothecarfid.com/

[16] L.B. Ayre, "Position Paper: RFID and Libraries," Galecia Group, Petaluma, CA, 2004 pp.1-21.