Proceedings of the 2010 International Conference on Industrial Engineering and Operations Management
Dhaka, Bangladesh, January 9 – 10, 2010

Layered OSGi-based Reconfigurable Lightweight RFID Reader Protocol

Aniruddha Bhattacharjya and Rajat Kumar Pal
Dept. of Computer Science and Engineering
University of Calcutta
Kolkata 700 009, India

Abstract

Radio Frequency Identification (RFID) systems are emerging as a practical means of auto-identification in a wide variety of applications in the modern era and as one of the most pervasive computing technologies in history. A typical RFID system is usually composed of four parts: tag, RFID reader, RFID middleware and application system. The host system seeking between the hardware and the enterprise system should introduce intelligent operational environment using configurable rules and lightweight, event driven architecture for smoothing and restricting high volume of raw events injected from the physical world. Such implementation should be part and parcel of a generic middle-ware platform to ensure an open environment and adaptability. To make the RFID reader integration uniform and effective, and meet the low calculation requirement of low-cost readers, a layered OSGi-based Reconfigurable Lightweight RFID Reader Protocol is presented here. Bearing in mind that the EPC standard has become a virtual standard of RFID system due to its rapid development, to meet the demand of EPC Reader Protocol Standard (RPS), the presented protocol is specified in four layers, respectively the Discovery Layer, the Reader Layer, the Tag Layer and the Notify Layer. Differ from EPCRPS, in which different Messaging/Transport Bindings provide for different kinds of transport, the Messaging Layer and the Transport Layer are no longer designed to be working in pairs. This protocol makes the reader integration uniform and effective.

Keywords

Radio Frequency Identification (RFID), Reader, Reader Protocol, Electronic Product Code (EPC), service management framework (SMF) , OSGi, Java Message Service (JMS)

1. Introduction

RADIO frequency identification (RFID) technology is gaining attention both from academicians and from practitioners [11]. RFID has the potential to serve as a fundamental technology [13] for ubiquitous services [3] where both objects and people can be identified automatically via attached RFID tags [18-19]. An RFID system comprises five key components— RFID tag or transponder, reader/writer, encoder, middleware [2, 5, 7, 10], and application software [8]. An RFID tag consists of a microchip and an antenna. The RFID reader/writer requests the identifying information contained in the microchip by sending an RF signal to the tag that then uses its antenna to transmit that information to the reader/writer via wireless data communication. The reader then translates the received information into a digital form and sends it to the application software with the help of middleware. The encoder, often the RFID reader/writer itself, encodes the data for storage in the tag once or many times, depending upon whether the RFID tag [19] is a read-only tag [18] or a read-write tag [12]. RFID middleware [7, 10] is the interface between reader devices and enterprise applications. However, the RFID middleware and the readers are usually varied, which introduces new problems to the top level RFID applications [5]. ***1.*** The forepart protocols between readers and tags are dissimilar, which makes readers and their backend protocols (between readers and RFID middleware) different. ***2.*** The model and number of antennas supported by the readers are dissimilar, which leads to the disorder of the control on reader nodes. ***3.*** Backend protocols provided by different vendors are also different, and the integration between readers and the middleware is complicated. ***4.*** As we know, there are various physical interfaces of readers, such as RS-232, USB and TCP/IP. To support the different interface, the system architecture of the application will be accordingly complicated and expensive. It can be experiential that the interaction between reader and middleware offered by different vendors has some common characters despite of the variety of forepart protocol and antenna. That is to say, it's possible to establish a universal reader protocol**.** Integration of heterogeneous systems and development of upper level applications will become much more easy if different reader vendors and middleware vendors obey a universal protocol [6], and the adaptability of RFID [14] will be greatly improved. Considering that the computing performance of various readers may be varied, to meet the low calculation requirement of low-cost readers [6] those are controlled only by simple Single Chip Micyoco, the

basic characteristic of the universal protocol should be lightweight. To make the RFID reader integration uniform and effective, and meet the low calculation requirement of low-cost readers, layered OSGi-based [15, 16, 17] Reconfigurable [2] Lightweight [5, 6] RFID Reader Protocol [6] is presented here. Taking into consideration the EPC standard [18, 19, 20] has become a virtual standard of RFID system due to its rapid expansion, to meet the demand of EPC Reader Protocol Standard (RPS) [20, 21]; the presented protocol is specified in four layers, respectively the Discovery Layer, the Reader Layer, the Tag Layer and the Notify Layer.

The rest of this paper is well thought-out as follows. Section 2 introduces the related literature survey. Section 3 analyzes the requirement of reader protocol. Section 4 presents the formulation for the lightweight RFID reader protocol. Section 5 describes the results of the experiments. Finally, section 6 concludes the paper.

2. Literature Survey

Numerous venders and organizations have proposed related criteria or protocols to universalize and standardize the communication protocol between readers and RFID middleware. The most typical one is the Simple Lightweight RFID Reader Protocol (SLRRP) [6]. SLRRP is enacted by Reva Company and its cooperation partners. The fundamentality of the protocol is TCP over IP. Self-defined functions are transported over SLRRP channel.
The main benefit of adapting SLRRP is that different readers and middleware systems can be incorporated easily with efficiency improving and error decreasing. However, SLRRP comes from manufacturer proprietary protocols, which entails following drawbacks.

- ✓ The reader logical function is defined in hard code, which makes the system almost impossible to be expanded.
- ✓ The parameters of commands are fixed, and dynamic parameters are not supported, which decrease the flexibility of the whole system.
- ✓ The physical interface of readers only supports TCP/IP, and the bottom protocol lacks of reliability restriction.

EPC Network [10] developed by MIT Auto-ID Center is a not-for-profit global research project. The EPC Network uses radio frequency in blend with a network system to allow items or products to be identified. EPCglobal [18, 19, 20] is the commercial heir of the Auto-ID Center, a global business initiative and academic research program with its roots at MIT, USA. The Auto-ID Center [12] ended its work in 2003, licensing its research results to the newly born EPCglobal, Inc. EPC Reader Protocol Standard (EPC RPS) specifies the interaction between a device capable of reading (and possibly writing) tags, and application software [7]. These two parties are herein referred to as the Reader and the Host. EPC RPS is specified in three distinct layers, respectively the Reader Layer, the Messaging Layer and the Transport Layer. Taking into consideration that the EPC standard has become a virtual standard of RFID system due to its fast development, EPC RPS is adopted as the basis in the paper to describe the working process and function of RFID reader protocol.

3. Requirement Analysis

The main inspiration of a universal reader protocol is to make the reader equipment integration uniform and effective. To fulfill this goal, the following principles should be obeyed.
1) Lightweight: To meet the low calculation requirement of some kinds of readers particularly those are controlled only by simple Single Chip Micyoco, the basic characteristic of the universal protocol should be lightweight.
2) Integrality: The reader protocol not only associates with data transmission from reader to the host, it should also cover the whole data period, which consist of tag data collection, filtration, buffering and transmission.
3) Fundamentality: To meet the requirement of auto recognition, and ensure that readers can be monitored by the host, the reader protocol should support several basic functions like antenna assign, tag data enquiring, tag data filtration, command sending, remote configure download (including user configuration data, communication speed, port, etc.), tag data transmission and etc.
3) Transparency: The change of forepart protocol should not influence backend protocol. For example, to support multiple forepart protocol readers, when different forepart protocol tags are detected, the communication rule between host and readers should remain the same.
4) Hardware independence: This is the core character of a universal protocol. Traditional communication protocols are usually designed by venders according to their own reader products, and are closely associated with the device hardware structure. This kind of protocols can not be compatible with each other. A universal protocol must have the capacity to be migrated to different standards of readers.
5) Simplicity: The communication module of an open communication protocol can be developed independently by other venders with public technical documents, which means that it is simple and can be easily implement.

6) Safety: Many RFID applications have high requirement of safety. For example, when it is used in personal identification area, privacy information should be prevented from filching. However, RFID readers usually have limited data processing ability. Hence, safety protection method should be effective and simple, to reduce the dependence of memory and performance of CPU (or MCU), and meet the real-time requirement.
7) Reliability: During the communication between the host and the reader, long distance data transfer is usually influenced by external interference, such as lightning interference and etc. Therefore, Communication protocol should offer the mechanism of error detection and control to improve the reliability.

4. Formulation

In this section, the formulation of the Layered OSGi-based Reconfigurable Lightweight RFID Reader Protocol is presented. To meet the demand of EPC RPS, the presented protocol is specified in four layers, respectively the Discovery Layer, the Reader Layer, the Tag Layer and the Notify Layer as shown in Figure 1.

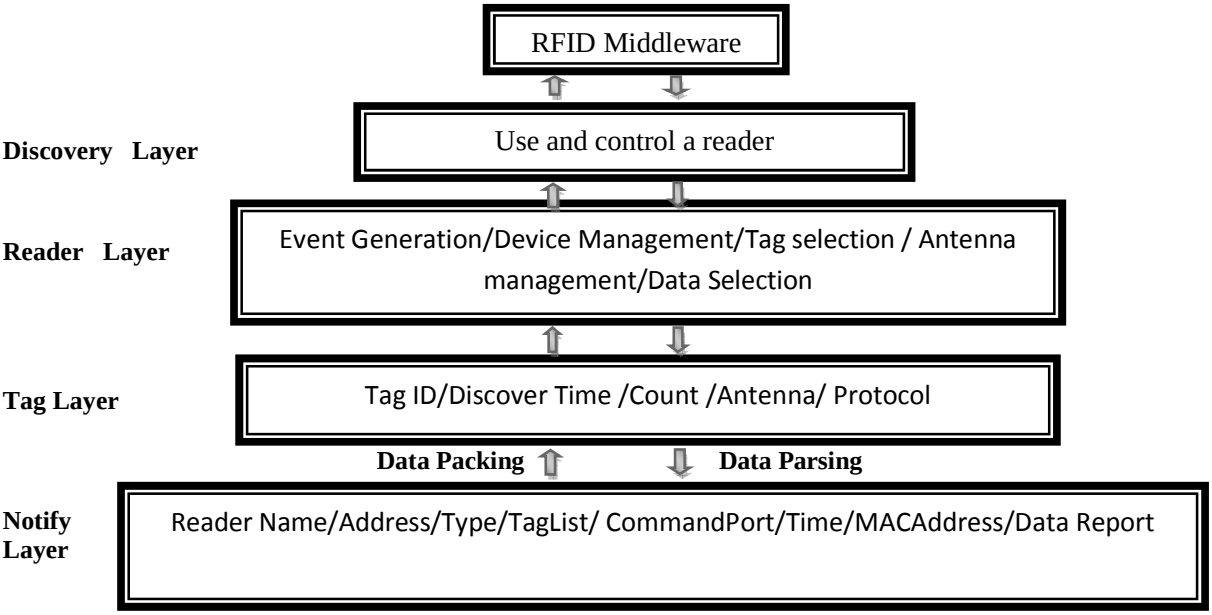

Figure 1: Framework of the presented protocol.

The upper objects that make use of this protocol can divided into three types, respectively the RFID middleware or application, collection nodes and the central control system. This protocol provides the communication and control channel between these objects and readers. Differ from EPC RPS (in which the function of transporting the encoded Reader Protocol commands between Readers and Hosts is performed by the Messaging/Transport Bindings and different MTBs provide for different kinds of transport), the Messaging Layer and the Transport Layer are no longer designed to be working in pairs. OSGi technology is a realization framework and technology of SOA thinking, compared to the traditional realization of SOA. OSGi framework [16, 17] is a more dynamic and scalable components framework. OSGi gives a definition of a standard, service-oriented computing environment, and provides with an open, service oriented, easy-to-deploy lightweight components Model. Based on OSGi framework, this work puts forward a reconfigurable RFID middleware service component architecture.

The *Discovery Layer* used in order to use and control a reader, its network address or the serial port number on the host where it is connected first is known by it. Dynamic Host Configuration Protocol (DHCP) mode of configuration eliminates the need for the user to perform network configuration for the device. This layer provides the classes, SerialDiscoveryListenerService and NetworkDiscoveryListenerService, needed to automatically search for and discover readers using both of these connection modes. In both cases, the service is created, an object is registered as the recipient of discovery events, and the service is started. Once started, the service runs on its own thread until it stops automatically (for serial discovery) or is told to stop (for network discovery). Here NetworkDiscovery-ListenerService class is developed along with SerialDiscoveryListenerService class. Each Alien reader is configured,

by default, to broadcast heartbeat messages over its local subnet. These messages are UDP (User Datagram Protocol) packets containing small XML documents which detail the reader's type, name, and contact information. The class that performs these listening duties is called NetworkDiscoveryListenerService. Once this class is instantiated and started, it will run in its own thread until it is stopped. While running, it listens for reader heartbeats on the listener port (which is specified in the constructor), calling either the readerAdded() or readerRenewed() methods of a registered DiscoveryListener when it detects a reader. Part of the heartbeat sent out by the reader indicates the time until the next heartbeat is expected. If this time expires before the next heartbeat is received, then the service assumes the reader has gone offline and will call the readerRemoved() method.

The *Reader Layer* consists of primary classes for communication between readers and the host (RFID middleware or applications) either over the network or a serial port. Typically the reader object will be obtained from a DiscoveryItem object, however, if the location (either serial port or network address) is known, a reader object can be instantiated directly without the need of any discovery classes.

The *Tag Layer* is one of the most important layers as tags play a very important part in the RFID reader and tag system. For this reason there is a single class devoted to storing and manipulating tag information: the Tag class. Additional classes and an interface are helpful for managing raw tag data and tag lists within the applications. The Tag class has the following members, each of which is accessible through getters and setters in the API: *Tag ID* – a string representing the tag's ID, *Discover Time* – the time the tag was first seen by the reader, *Last Seen Time* – the time the tag was last seen by the reader, *Count* – the number of times the reader has read the tag since it was first seen, *Antenna* – the (transmit) antenna number where the tag was last seen, *Protocol* – the air protocol used to acquire the tag's ID.

The *Notify Layer* consists of classes work in conjunction with a reader running in autonomous mode. In autonomous mode the reader is configured to read tags over and over again without the need for human interaction. The reader can be configured to send messages to listening services on the network when specific events occur, such as a timer expiring, tags added/removed from the taglist, successful/unsuccessful programming, etc. The notify classes implement such a listening services, constantly waiting and listening for notification messages from readers, and converting these messages into Java objects which are then available to the developed application. The key class in the notify layer is MessageListenerService. This is a service that listens at a specified port for incoming reader notification messages. A Message object encapsulates a collection of metadata about the notification message itself, and an array of Tag objects extracted from the taglist portion of the notification message. It contains the following members, all of which are available through getter and setter accessor methods: *ReaderName* – the name of the reader, *ReaderType* – the type of reader, *IPAddress* – the IP address of the reader, *MACAddress* – the MAC address of the reader, if provided, *CommandPort* – the port number on which to send commands to the reader, *Time* – the date and time the message was sent out, *StartTriggerLines* – indicates which external inputs triggered the reader to start, *StopTriggerLines* – indicates which external inputs triggered the reader to stop, *TagList* - an array of Tag objects extracted from the notification.

5. Results Obtained

In the experiment, one PC is served as the host. Alien Multi-Port General Purpose RFID Reader with four antennas are used. In the approach I have considered the RFID system (reader, antenna(s), tags, and host computing device,) as an "Edge Service Unit" for tracking data. Here one edge service unit is present consisting one Alien RFID Reader 8800, four antennas, two tags with Ethernet connectivity. The lightweight *OSGi-based Reconfigurable* RFID reader protocol is implemented here in JAVA, the run-time platform is service management framework (SMF) which is Open Services Gateway initiative (OSGi) compliant. Therefore the run-time environment is supported by embedded O.S like WINDOWS CE, Embedded LINUX as well as 32/64 Bit O.S environment (SUSE LINUX, WINDOWS XP/2000). Each layer consists of one or more OSGi bundles delivering specific set of services. The protocol uses publish/subscribe Java Message Service (JMS) infrastructure. Therefore, the architecture ensures a lightweight dynamic modularity for many applications.

```
<Alien-RFID-Reader-Auto-Notification>
<ReaderName>Alien Reader</ReaderName>
<ReaderType><Alien RFID Tag Reader (Class 1 / 915MHz)</ReaderType>
<IPAddress>10.1.70.13</IPAddress>
```

```
<CommandPort>23</CommandPort>
<MACAddress>00:80:66:10:11:6A</MACAddress>
<Time>2008/08/21 12:49:22</Time>
<Reason>TEST MESSAGE</Reason>
<Alien-RFID-Tag-List>
<TagID>8000 8004 0000 003B </TagID>
<DiscoveryTime>2003/12/04 15:08:59</DiscoveryTime>
<LastSeenTime>2008/06/04 15:08:59</LastSeenTime>
<Antenna>0</Antenna>
<ReadCount>4</ReadCount>
<Protocol>1</Protocol>
</Alien-RFID-Tag>
<Alien-RFID-Tag>
<TagID>8000 8004 9999 0004</TagID>
<DiscoveryTime>2008/12/04 15:08:59</DiscoveryTime>
<LastSeenTime>2008/12/04 15:08:59</LastSeenTime>
<Antenna>0</Antenna>
<ReadCount>3</ReadCount>
<Protocol>1</Protocol>
</Alien-RFID-Tag-List>
</Alien-RFID-Reader-Auto-Notification>
```

Alien-RFID-Reader-Auto-Notification TagLists in XML format

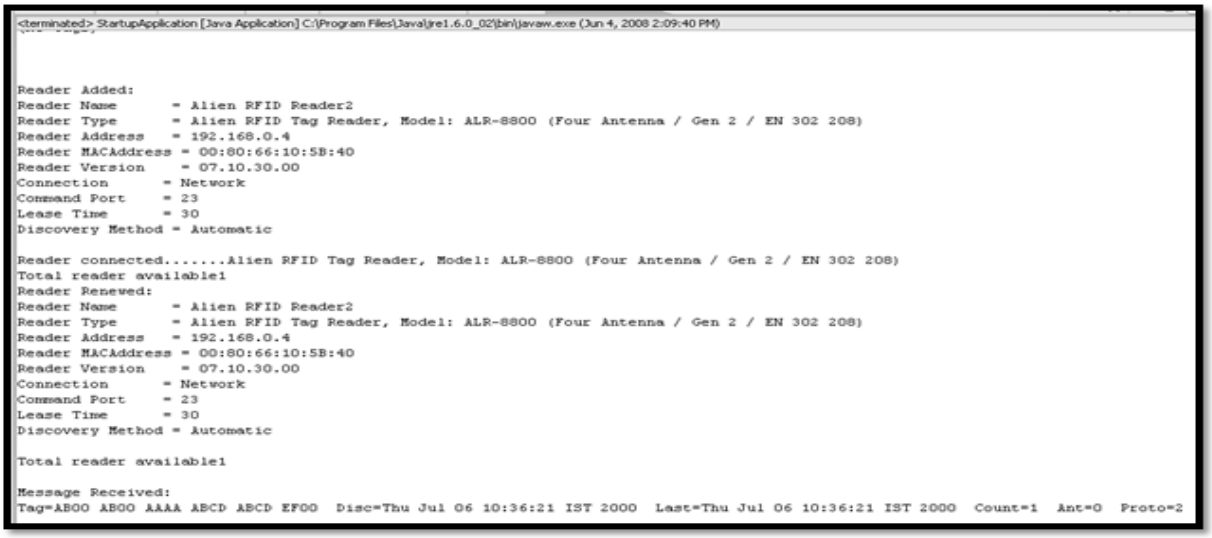

Figure 2: Depicting the console results which shows that protocol is lightweight and reconfigurable.

This approach also ensures scalability within the distributed environment where services can be partitioned and decentralized to reduce single point of failure. As there is no probability of aggregating all data traffic in a particular repository server in the event processing network so it is optimized automatically. The ability to quickly adapt to rapidly changing environments proves that the system is dynamic and flexible. Suppose there is only one "edge service unit" connected, then our system will show that reader available is one and corresponding tag-id, time and count number etc. The Figure 2 below shows the console results when only one reader is tracking the event data. The interoperability comes from the use of XML based which is shown below interactions that facilitate implementations in different languages, running on different platforms and over multiple transports.

6. Conclusion

To make the RFID reader integration uniform and effective and meet the low calculation requirement of low-cost readers, A Layered OSGi-based Reconfigurable Lightweight RFID Reader Protocol is presented in this paper. This protocol can be applied on the RFID middleware or application, the collection nodes and the central control system,

and can be easily deployed and expanded. The experiment console result shows that the protocol can meet the requirement of lightweight, integrity, fundamentality, transparency, hardware independence, simplicity, safety and reliability. Bearing in mind that the EPC standard has become a virtual standard of RFID system due to its rapid development, to meet the demand of EPC Reader Protocol Standard (RPS), the presented reader protocol is specified in four layers, respectively the Discovery Layer, the Reader Layer, the Tag Layer and the Notify Layer. Varying from EPCRPS, in which different Messaging/Transport Bindings provide for different kinds of transport, the Messaging Layer and the Transport Layer are no longer designed to be working in pairs. This reader protocol in the framework makes the reader integration uniform and effective.

References

1. Floerkemeier, C., Anarkat, D., Osinski, T.: PML Core Specification 1.0, AUTO-ID CENTER work paper (2003).
2. Fagui, L., Liu, H., LinKai, Ruan Yongxiong. R.: OSGi-based Reconfigurable RFID Middleware. Foundation: the National High-Tech Research and Development Plan of China under Grand No.: 2006AA04A115, the National Undergraduate Innovation Plan of China (2007).
3. Hossain, M.M., and Prybutok, V.R.: Consumer Acceptance of RFID Technology: An Exploratory Study. IEEE Transactions on Engineering Management, Vol. 55, No. 2 (May 2008).
4. Krishna, P.: Simple Lightweight RFID Reader Protocol, SLRRP Working Group.
5. Park, N., Lee, J., Kim, H., Chung, K., and Sohn, S.: A Layered Approach to Design of Light-Weight Middleware Systems for Mobile RFID Security. IEEE (2006).
6. Ouyang, Y., Hao, J., Zhang, T., Ren, Q., and Xiong, Z.: Research on Lightweight RFID Reader Protocol. IEEE (2008).
7. Song, J. and Kim, H.: The RFID Middleware System Supporting Context-Aware Access Control Service. In ICA0T (Feb.20-22, 2006).
8. Vue, D., Wu, X., Bai, J.: RFID Application Framework for Pharmaceutical Supply Chain. IEEE (2008).
9. Wu, J., Wang, D., Huanye Sheng, H.: Design an OSGi Extension Service for Mobile RFID Applications. In: IEEE International Conference on e-Business Engineering (2007).
10. Wu, J., Wang, D., and Sheng, H., Department of Computer Science & Engineering, Shanghai Jiaotong University: A Component-based Reconfigurable REID Middleware. In: The 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON 2007).
11. HighJump Software, A 3M Company: The true cost of radio frequency identification [Online]. Available: http://highjumpsoftware. com/promos/rfid-cost-report.asp (2004).
12. The History of RFID, Association for Automatic Identification and Data Capture Technologies, www.aimglobal .org (Oct. 2001).
13. RFID Journal Frequently Asked Questions, www.rfidjournal.com.
14. Data from: Radio Frequency Identification A Basic Primer, Association for Automatic Identification and Data Capture Technologies, www.aimglobal.org and from Wikipedia Encyclopedia, http://en.wikipedia.org/wiki .RFID (Aug. 2001).
15. OSGi Alliance, http://www.osgi.org/.
16. OSGi Technical Whitepaper, http://www.osgi.org/documents/collateral/ OSGITechnicalWhitePaper.pdf.
17. OSGi Alliance. OSGi service platform release 3 [EB/OL]. http://www.osgi.org/osgi_technology /spec_download3.asp.
18. EPCglobal, EPCglobal Architecture Framework , Version 1.2 (Oct. 2007).
19. EPCglobal, EPCglobal Certificate Profile, Ratified Specification 1.0 (Mar. 2006).
20. EPCglobal, Reader Protocol Standard, Version1.1 Ratified Standard (Jun. 2006).
21. Auto-ID Savant Specification 1.0., Oat Systems & MITAuto-ID Center, http://www.epc.org.mx/contenido/WD-savant-10-20030911.pdf.