A Multi-Agent-based RFID Framework for Smart-object Applications

Chi-Kong Chan, Harry K. H. Chow, Winson S. H. Siu, Hung Lam Ng, Terry H. S. Chu, and Henry C. B. Chan

***Abstract* — Coupled with software agent technology, RFID can transform everyday objects into smart objects. Currently, in most applications, agent definitions are not encoded directly on the tags due to tag memory limitations, and RFID technology is used purely for identification. Such approaches cannot provide the benefits of flexibility and modularization supplied by smart object systems. In this paper, we present a system called A-FRED for effectively encoding smart object data and agent definitions onto RFID tags. An XML-based memory-efficient encoding method is implemented and a behavior-based encoding scheme is adopted. The system provides an alternative framework for representing smart objects on passive RFID tags in general, and on memory limited low-cost UHF tags in particular.**

***Index Terms* — RFID, Software Agents, Smart Objects**

I. INTRODUCTION

It is envisioned that in the future everyday objects ranging from consumer electronic products to company assets will be tracked and processed automatically using RFID tags. Under notions such as the Internet of Things (IoT), tagged objects can be automatically represented, tracked, and queried over a network, particularly since the proposal of the EPCglobal architecture standard [1]. Since then, numerous RFID-based applications have appeared, many of which were made possible by the widespread availability of low-cost UHF tags,[1] despite their limited on-tag memory.

A natural approach to dealing with the large quantity of RFID data is to combine RFID with software agent technology [2]. Basically, software agent systems are used to monitor tag-reading events generated from RFID readers, and appropriate actions are taken accordingly [3-5]. For instance, an agent may trigger some actions in the local environment, establish a connection with a remote server [6], or communicate with other agents in the network to accomplish a task [7]. Many innovative applications have been developed. For example, in warehouses, stock volumes can be monitored automatically and exceptional conditions such as expired products or low stock volumes can be handled without delay [8]. In a library, books can be checked out automatically [9].

So far, many of these agent-based RFID systems utilize a static (non-mobile) system agent approach. That is, the processing agents and RFID modules of these systems are loosely coupled. The agents are not represented in the RFID tags and they function as an autonomous set of programs for tag processing, perhaps communicating using some standardized software agent languages and protocols. In such systems, one can theoretically replace the RFID components with other identification methods (e.g., barcodes, wireless location estimation, or even manual inputs), and the agent-based modules would require only small changes in basic design.

However, there is also a drawback to such static agent systems. As mentioned, the RFID technology in these approaches functions more or less as automated barcodes supported by elegant agent-based frameworks for data-processing. These frameworks, although promising in their own right, do not fully utilize the combined potential of RFID and software agent technology. For example, there is little intelligence in the tag level, and tags of all types are processed by the same set of static agents, making it hard to apply the modular design concepts often emphasized in agent systems.

Such potential is explored by smart object applications [10][11]. Smart objects are the virtual representations of individual everyday objects that are capable of sensing the environment, making on-site autonomous decisions, and communicating with humans or other system components. A smart RFID object consists of two components, namely, object processing logics and object data. The former is self-explanatory. The object data contain a unique identifier (for example, but not limited to, an EPC code), other object-related data such as the objects' current processing states, current and past sensor readings, and other static object data required during processing.

An RFID-based smart object requires a substantial amount of memory space to store object logics and data. This can be problematic for applications that depend on the more economical low-cost UHF tags with limited memory (typically 96 bits for the lower-end tags). Thus, we need some flexible and efficient ways to encode the required agent definitions and corresponding smart object data onto RFID tags. Recently, two possible approaches were identified in [12]. In the first approach, called identification-centric RFID systems (IRS), only the identifier of an object is stored on a tag, whereas the remaining object data and processing logics

Part of this work is related to the project "Enhancing the Competitiveness of the Hong Kong Air Freight Forwarding Industry Using RFID and Software Agent Technologies," which is funded by the Innovation and Technology Fund via the Hong Kong R&D Centre for Logistics and Supply Chain Management Enabling Technologies. Any opinions, findings, conclusions, or recommendations expressed in this material/event (or by members of the project team) do not reflect the views of the Government of the Hong Kong Special Administrative Region, the Innovation and Technology Commission, or the Panel of Assessors for the Innovation and Technology Support Programme of the Innovation and Technology Fund.

Chi-Kong Chan, Winson S. H. Siu, Terry H. S. Chu, and Henry C. B. Chan are with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong (e-mail: {csckchan, csshsiu, cshschu, cshchan}@comp.polyu.edu.hk).

Harry K. H. Chow was with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. He is currently with the Faculty of Management and Administration, Macau University of Science and Technology, Macao (e-mail: khchow@must.edu.mo).

Hung Lam Ng is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong (e-mail: 09658465g@connect.polyu.hk).

[1] In this paper, the term *low-cost UHF tags* refers to EPCglobal Class 1 Generation 2 tags with no more than 120 bits of available memory space.

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

have to be retrieved from databases over a network. This approach has the advantage of simplicity and compatibility with the EPCglobal standard. However, the network-based code and data storage leads to delays and limits its application to only those locations with network access. The second approach is called code-centric RFID systems (CRS). In this approach, source codes for defining agent-processing logics (i.e., agent actions) are stored on the tag using a compact coding scheme. However, as we shall see, agents encoded using this scheme can still easily exceed the severe memory limitations of most low-cost applications based on UHF tags.

Partially inspired by these previous works, we propose in this a paper a behavior-based encoding schema for smart objects on RFID tags. Instead of storing the entire (compact) code on a tag, we specify the desired behaviors that define an agent. To this end, we developed an Agent-enabled Flexible RFID Encoding and Decoding (A-FRED) system based on a novel RFID data compaction method for the efficient storage of both object logics and data. Our objective is to demonstrate a flexible and efficient approach to autonomous processing using smart objects. The efficiency of our system is demonstrated by experiments. An application example in logistics and a customer-oriented RFID application using low-cost UHF tags are discussed in this paper.

The remaining sections of this paper are organized as follows. Section II reviews some related works (including the aforementioned code-centric approach). Section III describes the A-FRED system. Section IV provides the results of some experiments to evaluate the efficiency of our system. Section V discusses two demonstration case study problems. Section VI compares the applicability of our system and that of some related approaches in different scenarios. Section VII gives the conclusion.

II. Related Works

A. Static RFID Agent Systems

A number of approaches to combining intelligent RFID systems with software agent technology have been proposed. Most of these applications employ a static agent approach, where software agents are employed in some kind of static environment to react to some RFID tag reading events. For instance, in manufacturing, an agent-based manufacturing control and coordination system (AMCC) for a manufacturing company in Taiwan is reported in [14] and, similarly, a manufacturing control system using RFID technology and multi-agent technology is documented in [15] for controlling material-handling tasks in packing and assembling. Examples can also be found in wireless sensor applications, for instance, for monitoring perishable items during transportation [16]. For cargo tracking in warehouses, an application example is reported in [17]. Similar applications based on static agents are also reported in the areas of healthcare [7], supply chain management [8], and library management [9], to name a few.

B. Smart Object Agent Systems

There have been several recent approaches investigating RFID-based smart objects. Unlike the static RFID agent approaches mentioned above, each RFID tagged item in a smart object based system is represented by its own mobile agent. Two major approaches can be identified, which Chen *et al*. called identification-centric RFID Systems (IRS) and code-centric RFID systems (CRS), respectively [12]. In the identification-centric approach, smart object logics or definitions are not stored on the RFID tags. Instead, each time a tag is identified by the RFID system, the relevant agent definition is retrieved from a database over the network and the corresponding agent is then created accordingly. Examples of this approach include the work of [11], who studied the use of smart objects for supply chain management, and the simulated IRS system implemented in [12]. However, the lack of local on-tag storage for smart object logics and data in this approach makes it inconsistent with smart object ideology, which emphasizes distributed intelligence and mobility, and modularity [13].

To this end, a code-centric approach was proposed and studied in [12]. In this work, compact action codes were used to represent smart object processing logics on RFID tags. To make their code as compact as possible, identifiers, variables, and operators of one to three characters in length were used throughout the language for specifying agent actions. For instance, in their implemented example, the command `l$n` instructs an agent to switch on an LED light, whereas the command `#3` instructs the agent to migrate to another processing node. It was demonstrated in [12] by simulation experiments that the CRS approach is more efficient than the IRS approach in terms of system processing time. However, there is a drawback to the code-centric approach. The issue here is that even with its compact coding scheme, the memory limit on many low-cost UHF tags could still easily be exceeded. For example, the 31 character-long action code used in the abovementioned experiment would require over 150 bits to store, exceeding the capacity of many low-end UHF tags. Moreover, the expressive power of this relatively simple language also limits its applicability in some applications.

In the next few sections, we shall present an alternative behavior-based encoding approach called A-FRED. Partially inspired by [12], A-FRED utilizes a compact encoding scheme for storing smart object data and the required behavior on an RFID tag, but without the necessity of storing the entire processing code.

III. System Architecture

A. System Overview

We developed a system called the Agent-based Flexible RFID Encoder and Decoder (A-FRED), which we first describe in the following way (Fig. 1). The RFID-related functionality of the system is provided by an agent called the FRED-Agent, which accesses the services of an Encoder and Decoder module. This module provides generic RFID tag reading and writing functions according to user-specified tag schemas, as well as data compaction functionality for saving tag memory space. The FRED-Agent passes the decoded tag data to a System Agent, which extracts the smart object data and behavior codes, and uses this information to create worker agents in the designated platforms. The worker agents are the agents that execute the actions specified by the behavior code. For instance, a worker agent may access some backend

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

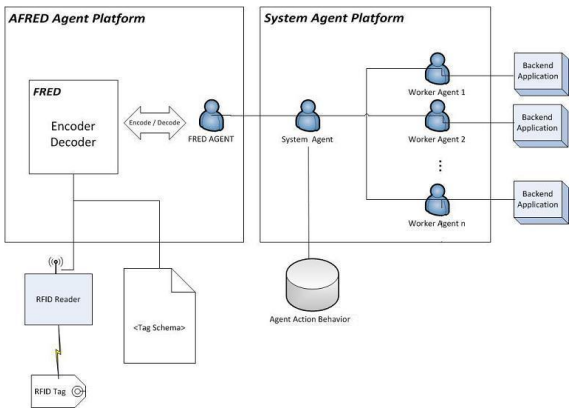

Fig. 1. A-FRED System Architecture

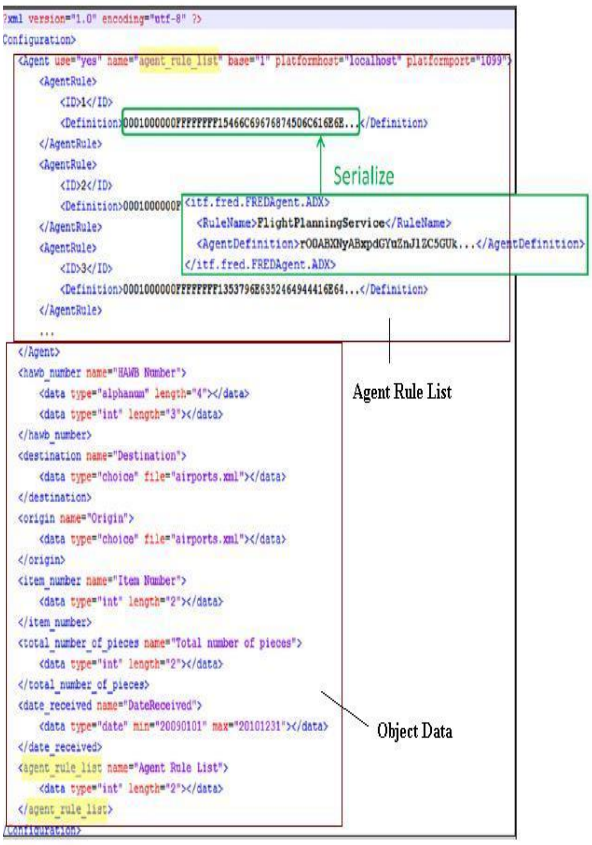

Fig. 2. Example of the A-FRED schema

applications or databases, or coordinate with other worker agents to accomplish certain tasks.

B. Tag Encoding and FRED-Agent

In order to effectively utilize the limited memory on RFID tags (and on low-cost UHF tags in particular), all tag data in our system are encoded, decoded, and compacted using a tag encoding and compaction scheme, as discussed below.

The content of each type of tag processed by our system is defined by a corresponding XML-based tag schema file that is accessible to the Encoder and Decoder module and the FRED-Agent. An example of a tag-schema file is given in Fig. 2, which illustrates the encoding schema of RFID tags in an air-cargo tracking application. Each schema is comprised of two parts: the *Agent Rule List* (ARL) and the *Object Data Definition* (ODD). Note that the schema file is not stored on the tags. Instead, it is located in an A-FRED Agent Platform where the Encoder and Decoder and the FRED-Agent are also located.

The ARL part of the tag-schema file contains information that deals with the definition of the smart object's software agents. This includes the host platform, where the worker agents are to be created, and a set of Agent Rules. Note that, unlike other fields defined in the tag-schema, the name of the host platform is directly specified in the tag-schema instead of being stored on the tags. The Agent Rules part of the ARL specifies the actual behaviors of the agent to be created. In order to achieve maximal savings of memory, each RFID tag only stores a bitmap to the set of all possible agent behaviors. Each agent behavior consists of one or more agent actions and is defined in a separate database that resides on the System Agent platform.

The ODD part of the schema defines the various fields of the object data part of the tag data. Each field definition contains the data type and the length of a field and, optionally, a valid value range. By requiring the user to specify the length and/or value range, the system is able to optimize on memory space by using just the required number of bits for storing each field according to the provided specification. Five types of data are currently supported: *alphabetic*, *alphanumeric*, *integer*, *date*, and *choices*. The first four are self-explanatory. The last type allows the user to specify an enumerated list of commonly used values that are defined either explicitly on the tag-schema file, or at an off-file location. The saving of tag memory space is achieved by storing an index to an element on the enumerated list. Similarly, data of the other four types are also stored as enumerated fields of all possible values of data in the valid data range.

C. Data Tag Compaction

During the reading and writing of tags, corresponding tag-schemas are used to encode the data into a compacted enumerated format so that the memory space usage of each field is minimized according to its declared type and data range. Additional compaction is then achieved by filling up unused value slots by further combining all encoded fields into one enumerated field.

D. System Agent

The decoded agent rules and object data are sent to a *System Agent* via an *inform-tag-read* message. The job of the System Agent is to create the *Worker Agents* that perform the tasks of the smart objects. In our implementation, agent actions are grouped into a number of agent behaviors. When an *inform-tag-read* message is received, the System Agent maps the decoded Agent Rules against the set of agent behaviors defined in an Agent Action Behavior Database. The required worker agents are then created in the specified agent platform using the retrieved agent behavior and the decoded object data. Depending on the system requirement, the System Agent and the Agent Action Behavior Database can reside on a different platform or on the same platform as the FRED-Agent.

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

E. Worker Agent

A Worker Agent is the agent for handling a single task on behalf of a single smart object. The Worker Agents in A-FRED are mobile agents, which means that it is possible for them to migrate to different platforms at run time. For example, a Worker Agent with an "Update Database Behavior" can be created at a "Main Agent Platform," and then migrate to a "Database Agent Platform" to perform the required updating actions, and finally report back to the "Main Agent Platform" to complete the remaining tasks. Note, however, that the Worker Agents for the same type of tags will initially be created on the same platform as that defined on the corresponding tag-schema file, which is not necessarily the System Agent Platform. All agent platforms in our system are implemented using the Java Agent Development Framework (JADE). All Worker Agents for a given smart object are created simultaneously so that any coordination (or sub-ordination) amongst the Worker Agents will have to be defined explicitly in the agent behaviors.

IV. Experiment

Before we proceed to analyze the advantages of the proposed approach, we first evaluated the running time efficiency of our system in a series of experiments. In these experiments, a number of EPCglobal Class 1 Gen 2 UHF RFID tags with 240 bits of memory were encoded and processed using A-FRED, and the running time was recorded. The tags were arranged into batches of various sizes, and in each test case tags in the same batch were processed simultaneously by the system. There were eleven test cases, corresponding to the batch sizes of 1, 5, 10, 15, … , and 50 tags, respectively. One agent behavior was defined, which contained a single action for logging down the current system time when the worker agent was created. Thus, for each batch b and each tag $i \in b$, we recorded the RFID tag reading time t_1^i, which was logged by the Encoding and Decoding module when a tag was first read, and the worker agent registration time, t_2^i, which was recorded by the (single) Worker Agent created for a tag. Each test case was repeated 10 times, and for each test case we computed the average tag processing time $\bar{t} = \sum_{i \in b} (t_2^i - t_1^i) \Big/ |b|$ and average batch processing time $t_b = \max_{i \in b}(t_2^i) - \min_{i \in b}(t_1^i)$. All system components (the tag schema, all agents, and all databases) resided in a high-performance computer with 6Gs of memory. For the RFID equipment, a fixed UHF RFID reader and a single circularly polarized antenna were used. The tags were placed at a distance of 1.5 meters from the antenna. During the tests, the total processing time per tag and per batch was recorded. For comparison, we also implemented a static agent system and its average processing time per RFID tag was also recorded.

Fig. 5. RFID-enabled HAWB labels

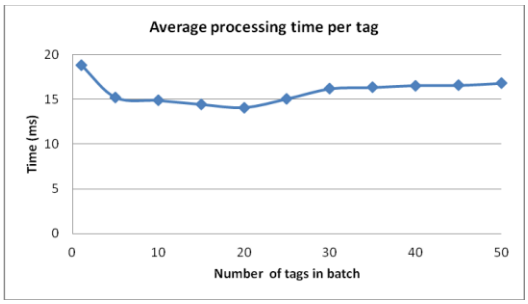

Fig. 3. Average processing time (per tag)

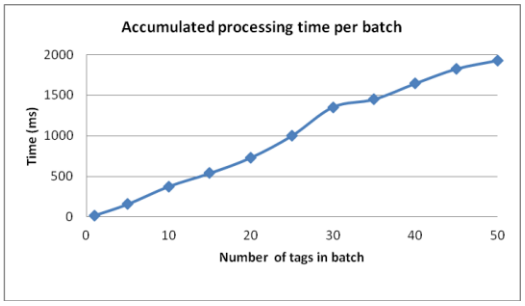

Fig. 4. Accumulated processing time (per batch)

The results are shown in Figs. 3 and 4. It can be seen that the per tag processing time remained approximately constant (at between 15 to 20 ms) for all of the batch sizes that we tested. In comparison, the processing time in the static agent system averaged 4.16 ms, and up to 15 ms in the worst case. Thus, the A-FRED system requires a longer running time than a static system, although this is anticipated. The overhead is mainly due to the extra work involved in creating the Worker Agents, a step that is not required in static systems. However, the results that were obtained are still within the acceptable range, as over 50 RFID tags were processed per second, a figure that falls within the typical reading rate of most UHF RFID readers in the majority of RFID applications.

V. Example Applications

A-FRED can be employed to support many advanced smart object applications. For the purpose of illustration and to facilitate further discussion, we implemented two case study applications, described as follows.

A. A-FRED-Logistics

The first demonstration application that we implemented is a business-oriented application called A-FRED-*Logistics*. Currently, many consumer products are transported as air freight to ensure timely delivery. A-FRED-*Logistics* is a system for supporting automated cargo tracking and processing at various stages of the cargo supply chain. During the process, the cargo items are transported in cartons arranged into pallets. Each carton is identified by an RFID

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

```
<?xml version="1.0" encoding="utf-8" ?>
<Configuration>
    <Agent use="yes" name="A-FRED-Logistics" base="1"
      platformhost="localhost"  platformport="1099">
      <AgentRule>
        <ID>1</ID> <Definition></Definition>
      </AgentRule>
    </Agent>
    <hawb name="HAWB">
       <data type="alphanum" length="7"></data>
    </hawb>
    <origin name="Origin">
       <data type="choice" ref="IATA"></data>
    </origin>
    <destination name="Destination">
       <data type="choice" ref="IATA"></data>
    </destination>
    <Shipmentdate name="Shipment Date">
      <data type="Date" min="20110101" max="20121231"></data>
    </Shipmentdate>
</Configuration>
```

Fig. 6. Tag schema for A-FRED-*Logistics*

enabled *waybill label* (Fig. 5), which contains an identity number for the carton (called the HAWB number), as well as other information required during the transportation process, such as the item's destination, place of origin, and shipment date. Two types of tasks are performed at each checkpoint: i) cargo verification, which means verification that the cartons are being processed at the correct location, and ii) status updates, which refers to updates of the carton's status on system servers. However, some types of cargo items require specific status update strategies. For example, some types of cartons require the system to connect to the backend system of both the logistics company and the shipping company for status updates, while this step of updating can be skipped altogether for some other types of cargo.

Our implementation of A-FRED-*Logistics* follows the architecture as specified in Fig. 1, and the tag schema is shown in Fig. 6. Three action behaviors are defined for the actions involved in verifying the location of a carton, and for the two cargo status updating strategies, respectively. For each behavior specified on the tag, a corresponding worker is created on the System Agent Platform after the tag is read. Overall, the outcome of the system was satisfactory. The object data and agent behavior required only 61 bits to encode, which fit well into the low-cost UHF RFID tags and larger tags alike.

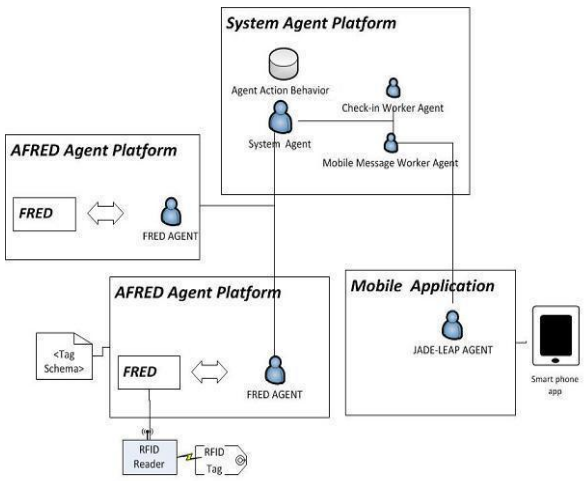

Fig. 7. System Overview of the *Shopper Information Agent System*

```
<?xml version="1.0" encoding="utf-8" ?>
<Configuration>
    <Agent use="yes" name="A-FRED-Mobile" base="1"
      platformhost="10.0.18.200"  platformport="1099">
      <AgentRule>
        <ID>1</ID> <Definition></Definition>
      </AgentRule>
    </Agent>
    <customer_id name="Customer ID">
       <data type="alphanum" length="5"></data>
    </customer_id>
    <surname name="Surname">
       <data type="char" length="10"></data>
    </surname>
    <gender name="Gender">
       <data type="choice" ref="sex"></data>
    </gender>
    <expiry_date name="Expiry Date">
       <data type="date" min="20110920" max="20111231"></data>
    </expiry_date>

</Configuration>
```

Fig. 8. Tag schema for *SIAS*

B. An RFID-Mobile-Phone Application for shop customers

Another example of an application that we implemented is called the *Shopper Information Agent System* (SIAS). This system is a customer-oriented application that allows the user to check in at various checkpoints located in retail stores. Low-cost UHF RFID tags are issued to the customers, with each tag storing the following information: a customer's ID, gender, the customer's surname, and an expiry date. Two behaviors are defined, namely a check-in behavior, which records the user's presence at a checkpoint, and a mobile-message behavior, which retrieves and sends a user-specific welcome message to a registered smart phone application to facilitate the user's shopping experience.

The architecture and tag-schema file of the system are illustrated in Figs. 7 and 8, respectively. Note that in this case, unlike in the previous example, more than one FRED-Agent (representing multiple store locations) can be connected to an external System Agent. Two types of Worker Agents will be generated: the Check-in-Worker-Agents and the Mobile-Message-Worker-Agents. The Mobile-Message-Worker-Agents in turn communicate with user-agents located on the users' mobile phone platform, and user-specific messages are delivered accordingly. The mobile phone based user agents are implemented using JADE-LEAP.

The overall performance of SIAS was also satisfactory. The smart objects behavior and object data required only 86 bits to encode, which again fit well into low-cost UHF RFID tags.

VI. Discussion

Earlier in this paper, we discussed three types of RFID-based software agent systems: 1) static RFID agent systems, which do not support smart objects, 2) identification-centric systems, where no smart object actions or processing logics are specified on the tag, and 3) code-centric systems, which store a smart object's action code on a tag using a compact encoding scheme. In this paper, we proposed a fourth approach, the behavior-based encoding scheme, which is utilized by our current system, A-FRED. Kiviat diagrams illustrating various properties of the four approaches are presented in Fig. 9. It should be noted that all four approaches have their respective advantages and

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

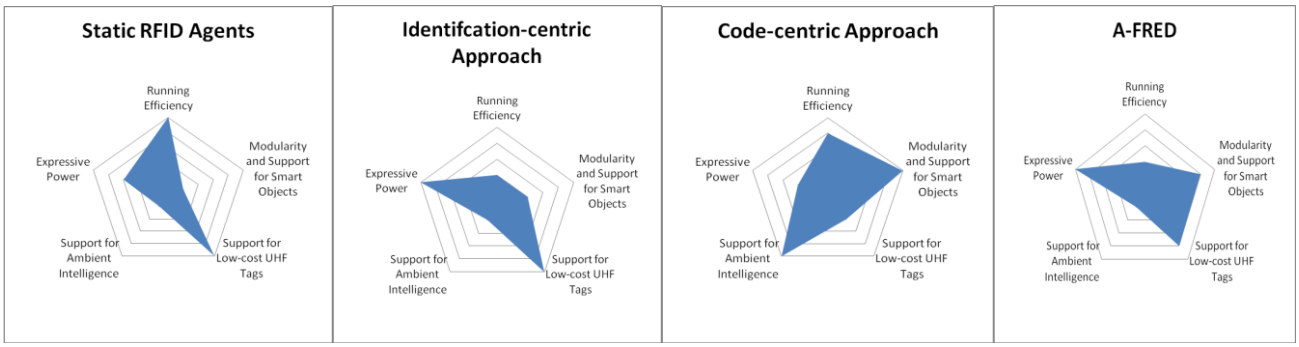

Fig. 9. Kiviat diagrams for the four types of RFID-based software agent systems

disadvantages. For instance, for supporting system modularity, which is an important concept in software agent design, the code-centric approach and A-FRED are superior due to their ability to specify agent behaviors on individual RFID tag levels. With regard to running time efficiency, the code-centric approach and the static agents outperform the identification-centric approach and A-FRED, because the latter two need to retrieve agent code definitions from the server during the creation of agents. By contrast, the relatively simple action encoding language of the code-centric approach has limited its applicability to applications with relatively simple tasks, and may not be easily applied in low-cost UHF tag based applications due to the larger memory size that it requires. A-FRED, on the other hand, does not have these limitations, which makes it a good candidate for implementing smart objects on low-cost UHF tags in general.

VII. Conclusion

The concept of smart objects is leading to many innovative applications, with RFID and software agent technologies being two of the main components. In this paper, we presented a flexible RFID data encoding and decoding mechanism called A-FRED. By using an efficient data representation scheme and a behavior-based encoding scheme, our system can be used to encode not only object information but also the required agent actions onto RFID tags. The approach that we have presented can complement existing approaches to RFID-based smart object systems, particularly for applications that are dependent on low-cost UHF tags.

References

[1] EPCglobal Inc., "EPC radio frequency identity protocols Class-1 Generation-2 UHF RFID protocol for communications at 860 MHz – 960 MHz Version 1.2.0", 2007.

[2] The Foundation for Intelligent Physical Agents(FIPA). http://www.fipa.org

[3] H. K. H. Chow, K. L. Choy, and W. B. Lee, "A dynamic logistics process knowledge-based system - An RFID multi-agent approach", *Knowledge-Based Systems*, vol. 20, pp. 357-372, 2007.

[4] A. J. C. Trappey, T. Lu, and L. Fu, "Development of an intelligent agent system for collaborative mold production with RFID technology", *Robotics and Computer-Integrated Manufacturing*, vol. 25, pp. 42-56, 2009.

[5] M. Tu, Jia-Hong Lin, Ruey-Shun Chen, Kai-Ying Chen, and Jung-Sing Jwo, "Agent-Based control framework for mass Customization Manufacturing With UHF RFID Technology", *IEEE Systems Journal*, vol. 3, no. 3, pp. 343-359, Sept 2009.

[6] D. G. Yun, J. M. Lee, M. J. Yu, S. G. Choi, and C. H. Seo, "Agent-based user mobility support mechanism in RFID networking environment", *IEEE Transactions on Consumer Electronics*, pp. 800-804, 2009.

[7] F. Gîză, C. Turcu and C. Turcu, "RFID technology and multi-agent approaches in healthcare", *Deploying RFID - Challenges, Solutions, and Open Issues*", Cristina Turcu (Ed.), ISBN: 978-953-307-380-4, InTech, pp. 127-140, 2011.

[8] S. Wang, S. Liu and W. Wang, "The simulated impact of RFID-enabled supply chain on pull-based inventory replenishment in TFT-LCD industry", *International Journal of Production Economics*, vol. 112, pp. 570–586, 2007.

[9] T. Minami, "Library services as multi agent system", *Agent and Multi-Agent Systems: Technologies and Applications*, Lecture Notes in Computer Science, vol. 4953, pp. 222-231, 2008.

[10] G. Kortuem, F. Kawsar, V. Sundramoorthy, and D. Fitton, "Smart objects as building blocks for the internet of things", *IEEE Internet Computing*, pp. 44-51, 2010.

[11] E. Bajic, "A service-based methodology for RFID-smart object interactions in supply chain", *International Journal of Multimedia and Ubiquitous Engineering*, vol. 4, no. 3, 2009.

[12] M. Chen, S. Gonzalez-Valenzuela, Q. Zhang, and V. Leung, "Software agent-based intelligence for code-centric RFID Systems", *IEEE Intelligent Systems*, vol. 99, 2010.

[13] The European technology platform on smart systems Integration (EPoSS), *Internet of Things in 2020: ROADMAP FOR THE FUTURE* (2008), Version 1.1, 2008. http://ec.europa.eu/information_society/policy/rfid/documents/iotprague2009.pdf

[14] R. S. Chen and M. Tu, "Development of an agent-based system for manufacturing control and coordination with ontology and RFID Technology", *Expert Systems with Applications*, vol. 36, pp. 7581-7593, 2009.

[15] P. Vrba, F. Macurek and V. Marik, "Using Radio-frequency identification in agent-based control systems for industrial applications", *Engineering Applications of Artificial Intelligence*, vol. 21, pp. 331-342, 2008.

[16] R. Jedermann, C. Behrens, D. Westphal, and W. Lang, "Applying autonomous sensor systems in logistics—combining sensor networks, RFIDs and Software Agents", *Sensors and Actuators A*, vol. 132, pp. 370-375, 2006.

[17] H. K. H. Chow, K. L. Choy, and W. B. Lee, "A dynamic logistics process knowledge-based system – An RFID Multi-Agent Approach", *Knowledge-Based Systems*, vol. 20, no. 4, pp. 357-372, 2007.

ISBN: 978-988-19251-1-4
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)