RFID Emulation in Rifidi Environment

Claudio E. Palazzi, Alessandro Ceriali, Marco Dal Monte

Dipartimento di Matematica Pura e Applicata

Università degli Studi di Padova

Via Trieste 63, 35131 Padova, Italy

cpalazzi@math.unipd.it, {acerialilmdalm}@studenti.math.unipd.it

Abstract RFID is a technology for radio frequency identification that enables software systems to automatically detect and identify objects in the real world. Thanks to this ability, RFID is of interest for numerous companies; unfortunately, the evaluation of a new business scenario involving RFID demands significant investments in time, hardware, and infrastructure. Instead, Rifidi is a tool that quickly and realistically emulate RFID scenarios in order to explore their possibilities before investing in them. We present here an analysis of Rifidi, showing how to practically exploit it to design a new business process, and discuss pros and cons in its use. Furthermore, as a case study, we consider a new application that we have devised and that could be of interest for companies: an RFID-based, virtual shop assistant.

1 Introduction

Radio Frequency Identification (RFID) is a technology that has attracted great attentions both from researchers and industry practitioners. Indeed, thanks to its ability to allow software systems to automatically detect and identify objects in the real world, RFID has emerged as a technology able to bridge the gaps between the digital and physical worlds. Truly innovative context-based applications and business processes can be developed thanks to this technology, providing a new interaction paradigm between humans and technology.

With the term RFID we identify all the technologies that permit the distance recognition of objects, animals, and people using radio waves. A radio waves identification system is a composition of a transponder (or tag) and a reader. The tag is the label set on the object and contains a microchip and an antenna. The microchip is used to store data about the object: object's features or just an identification

code; in the latter case, to the code is generally associated a certain amount of data that are remotely stored in a database. Instead, the antenna is utilized to transmit the information stored in the microchip. Tags can be active or passive; active tags can actively transmit the information stored in their microchips, whereas passive tags need to be irradiated by an electromagnetic field (generated by a reader) to transmit their data. A reader is a device, fixed or mobile, used to read the RFID tag, and it can convert the radio waves emitted by a tag into a digital signal that can be transferred to a computer; readers and tags have to utilize the same radio frequency to be able to communicate.

Currently, the use of RFID has been tested in several scenarios, e.g., passports, toll payments, product tracking, animal identification, inventory, libraries, and museums [1]. RFID's potentialities have been noticed by many companies that are now wondering whether the benefits that they could achieve from RFID implementation justify the investment required by this technology. Unfortunately, the evaluation of a new business scenario involving RFID demands significant investments in time, hardware, and infrastructure; this complexity is also exacerbated by the vast heterogeneity (in terms of frequency, range, memory, cost, etc.) of available tags, antennas, and readers. It is hence evident how companies would like to have available a tool to quickly and realistically emulate an RFID scenario in order to explore its possibilities before investing in the technology.

To this aim, an interesting tool is represented by Rifidi, which allows to generate hybrid testbed scenarios combining emulated RFID features with real software systems. We have hence devised a new context-aware application based on RFID, i.e., a virtual shop assistant, and analyzed its business process through the use of Rifidi as a real company could do.

The contribution of this paper is twofold. First, analyze the Rifidi tool, showing how to practically exploit it to design a new business process, and discuss pros and cons in its use. Second, we present a new RFID-based application that we have developed and that could be of interest for companies: a virtual shop assistant. This application allows a software system to be aware of the product that a customer is evaluating for purchase and provide suggestions that are based on the company's choices rather than the peculiar tastes of a human clerk. This application has been implemented through a hybrid approach that combines Rifidi with other software components.

The rest of the paper is organized as follows. Section 2 provides background information on RFID technology. In Section 3, we discuss the main components and features of the Rifidi tool. Section 4 presents the RFID-based application that we have designed and developed as our case study: the virtual shop assistant. In Section 5, we explain how we have integrated our application with Rifidi for the emulation of the RFID scenario, and provide a critical analysis on the use of the Rifidi tool. Finally, Section 6 concludes this paper.

2 RFID Background

RFID systems use a varying number of frequencies, that can be classified as:

- low frequencies (LF, 125-134 kHz);
- high frequencies (HF, about 15 MHz);
- very high frequencies (UHF, 860-960 MHz);
- microwaves (over 2.45 GHz).

Frequency bands have different characteristics, thereby, they are used for different applications. In general, to a higher radio frequency corresponds also a wider transmission range and bandwidth, but also a higher sensitivity to interferences and cost.

RFID tags can be of three different types:

- passive: they get the power needed for transmissions from the waves sent by the reader which sends questions to the tag and induces electrical power to the antenna;
- semi-active: they have a power source which is needed for temperature or movement sensors;
- active: they are powered with a battery, which offers a bigger range for the radio signal and the reading distance.

The labels can be:

- read-only;
- write-once & read-many;
- read & write.

For the first two cases, the RFID tag represents a technological evolution of the bar code, because the information saved in the microchip cannot be modified. The read & write type is the more flexible one; the tag can be used as a dynamic memory because the information in the chip can be updated in every moment.

The frequency collisions on tags can be avoided with systems that use algorithms which divide the signals coming from the tag. So they regulate the time intervals in which they have to be read.

Yet, collisions may happen, for instance in a setting with multiple tags attempting to simultaneously send reply to a reader or when a tag is in the radio range of two or more readers [2]. To deal with this issue, various proposals have come from researchers in order to ameliorate the efficiency of tags identification [3-5].

There exist different architecture and data exchange protocol standards for RFID, and conformity standards for emitted radio frequencies. These emissions have not to lay on frequency bands previously allocated for other purposes such as Wi-Fi, Radio, analogical TV, etc. The problem in standards is that there has not yet been a convergence toward an international unification between the principal emerging or de facto standards (i.e., ISO and EPCglobal [6]). The situation in RFID frequency standards is more complex than architecture standards and protocol standards. The rules for the radio waves grants can vary from country to country (Europe, USA and Japan). So it seems complicated to find a frequency or a frequency band to reserve to RFID globally. The only frequency that can be now considered unified in the world is the HF one, fixed at 13,56 MHz, whereas at low frequencies the largest part of the world employs the range 125-134 kHz.

This heterogeneity represents a crucial issue for companies that would like to test RFID technology in their productive chain. The evaluation of different types of RFID tags/readers could involve the practical experimentation with different hardware, thus requiring consistent investment in time and money.

Instead, having a tool that enables quick emulation of RFID tags and readers would allow companies to focus on how such a technology could be integrated with the existing software system, leverage on the company's database to produce new services, and deal with customers' concern for their privacy by design an appropriate software architecture [7-11]. All these issues are crucial for the success of the RFID utilization; through an RFID emulative tool, they could all be appropriately addressed before having to buy the RFID hardware thus allowing companies to ponder their options with a minimal cost.

3 The Rifidi Tool

Rifidi is a software tool able to emulate RFID systems [12]. It allows the virtual creation of an RFID-based scenario while being sure that the software created for this purpose will run as is also in the real world. Indeed, Rifidi is a program that emulates the reader/client interface of an RFID reader. This means that a client communicates with the Rifidi reader in the same way that it would communicate with a real reader. For example, with the Alien reader, a client would send telnet messages to retrieve tag reads. The virtual Alien reader in Rifidi Emulator responds to telnet messages in the same way a real one does. Furthermore, Rifidi is implemented in Java and it is based on the ECLIPSE environment; it is possible to download the source code and modify it.

More in detail, Rifidi has two main components:

- a *Designer*, which permits to create a 3D scenario with graphic elements (e.g., readers, packages, push-arms, etc.) and animations so that tags can be read by emulated RFID readers;

- an *Emulator*, a development tool which permits to emulate the reading of a RFID tag and the correlated events (it is a simpler and faster tool than the Rifidi Designer, as it makes use of just a textual interface).

For our purposes we have used both of them; we elaborate on these two components in the following.

3.1 Rifidi Designer

When opened, the Rifidi Designer is a window divided into four main parts (see Fig. 1):

- at the upper left part of the screen there is a frame which shows some elements that can be added in the 3D scenario;
- in the downer left part there is a mini-map about the 3D scenario;
- in the centre of the screen there is the 3D scenario itself, in which it is possible to add the elements;
- in the window under the 3D scenario there is a list of properties which values can be changed.

The first step with Rifidi is to create a new scenario giving it a name and choosing the dimension of the room where to put the objects. The tool permits to add some predefined objects such as producers of RFID-tagged objects, conveyors, reader gates, push-arms, boxes, etc.. Every object has some properties (e.g., sensitivity and speed) that depends on the type of object. Objects can be rotated, removed, and their behaviors can be made depending on another object's behavior by using the feature named GPIO (General Purpose Input Output). In particular, this property must be activated at the creation moment of the object to be used. A typical use of GPIO is the dependency of a push-arm by a gate (e.g., if a reader detects a certain RFID tag, a push-arm is activated to move the object labeled with that tag). Every object added to the 3D scenario can be enabled or disabled at run time.

Every gate is associated with an IP address and a port on which it listens for connections; this way, it is possible to monitor the RFID tags which pass through the gate. A console helps the user to analyze the read-process of tags for every gate under which a virtual box passes.

The usage of the Rifidi designer is very interesting; it permits to create complex configurations of box producers, push-arms, readers, etc. It also allows to set the transition velocity and responsivity of the various items. Unfortunately, there is only a very limited set of 3D objects that can be used for setting up a scenario. In

fact, the configuration prepared for this paper is similar to the intended scenario only for the number of steps, but not for the 3D object drawn on the screen.

In Fig. 1, we show a screenshot about an execution of the Rifidi Designer tool during one of our tests. Our tests were aimed at emulating a scenario corresponding with the virtual shop assistant application as described in Section 4.

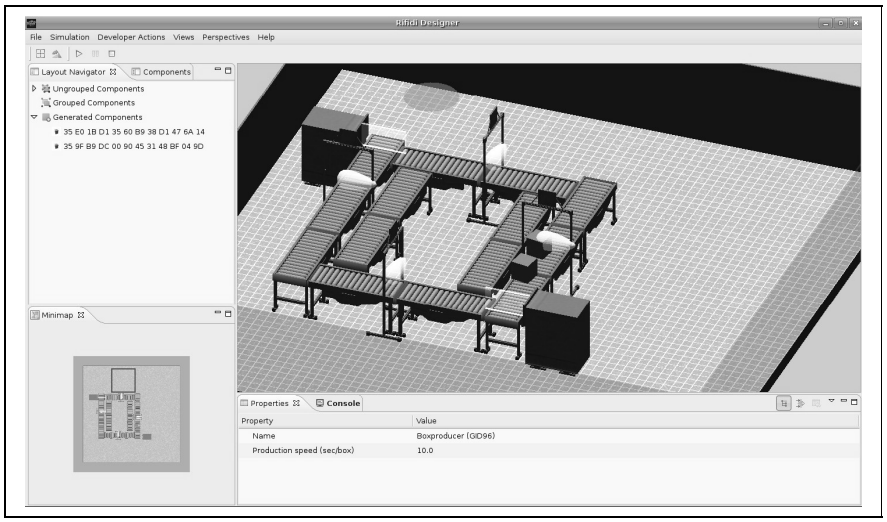

Fig. 1. Emulation with Rifidi Designer.

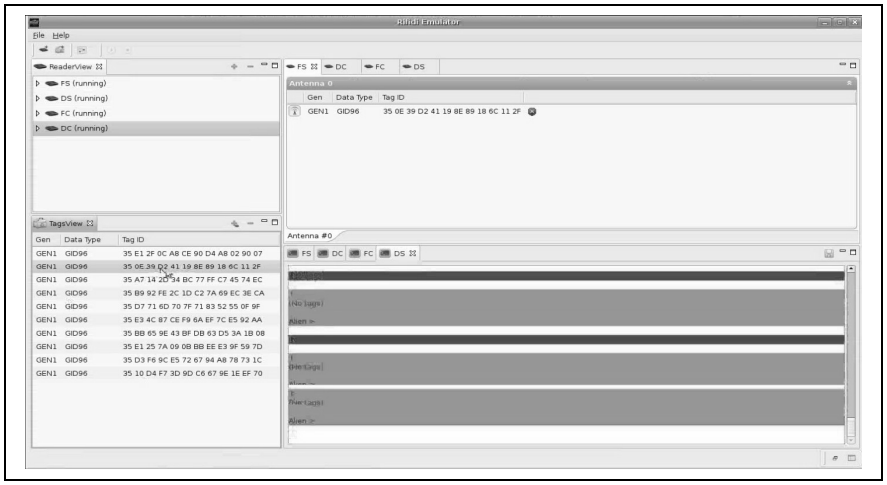

Fig. 2. Emulation with Rifidi Emulator.

3.2 Rifidi Emulator

The Rifidi Emulator is the other part we talked about at the beginning of this section and it is used for studying the behavior of the system when the reader catches a tag.

When opened, the Rifidi Emulator is a window composed by four main parts:

- at the upper left part of the screen there is a frame which shows the readers that are in the system;
- in the downer left part there is a list containing the tags created, if there are some;
- in the centre of the screen there is a list containing the situation at runtime of the queue of the selected reader;
- on the bottom of the screen there is a shell for visualizing the behavior of the tags readings.

The use of Rifidi Emulator is simple too. The first step to use the Rifidi Emulator is to add at least one reader through the same procedure already seen in Section 3.1: select a reader type, give a name to the reader, set the port, set the GPIO on, click on the Finish button and the reader is added. After this few steps the system is ready and by clicking on the Play button it is possible to start the emulation. After the start, it is possible to create new tags (by just clicking on the symbol '+') in the list situated in the downer left part of the Rifidi window. The tags are given to be read by the reader by dragging them to the queue of the selected reader; by connecting via telnet to the port of the reader, it is possible to see that the reader catches the tag when it becomes the first in the queue.

Therefore, the practical difference between the Rifidi Designer and the Rifidi Emulator is the way in which the queue can be managed. With the Rifidi Designer, tags are automatically and randomly generated by the tag generator and the queue is automatically managed by the system. Instead, with the Rifidi Emulator, tags can be generated by the user, who can also drag tags on the queue or remove tags from the queue.

In Fig. 2, we present a screenshot about an execution of the Rifidi Emulator tool during one of our tests for understanding how to replicate the same scenarios created with the Rifidi Designer.

4 Case Study: Virtual Shop Assistant

The case study we have chosen represents a common garment store (Fig. 3). Customers looks for the garments exposed on the various shelves; When a customer

decides to take a garment, usually she/he tries it on. At this moment, the person goes on the changing room. In a common garment store the only opportunity to understand how good garments put on are suitable for the person is to wear them in front of a mirror, or to ask to other persons for suggestions (for instance to a clerk). Instead, we have devised and implemented a Virtual Shop Assistant (VSA).

Our VSA is an RFID-based, context-aware application that is able to detect which garments the customer has brought into the changing room and perform suggestions about them (other garments that may correspond with the customer's tastes, similar items that are on sale, shoes that match a certain shirt, etc.). VSA would hence allow both a more joyful experience for the customer and the possibility for the store's owner to automatically specify certain selling policies. The last point is particularly useful for big brands that have many stores distributed all around the world: they would be able, for instance, to ensure that the promotion of a certain item is actually performed without relying on the willingness of each clerk.

To utilize VSA, every garment item has an RFID tag. The customer gets some garments from a shelf and goes to a changing room for trying them. The system modifies the state of the items chosen by the customer as busy, using an appropriate RFID reader situated next to the rank. This RFID reader monitors the garments taken or put back and administrates their states as shown in Fig. 4.

At the entrance of every changing room is situated another RFID reader. This reader detects the tags passing in and out the changing room (Fig. 5) and updates their states to keep track of their position within a certain changing room.

Inside the changing room there is a monitor (see Fig. 3) on which our application shows the information about the garments chosen by the customer along with other useful and related details. The monitor displays information such as the total cost of the garments in the room and whether a chosen item (or a similar one still on the shelf) is on sale. It is also possible to obtain more details about the garments (e.g., the size, the brand, the fabric) just selecting the garments on the monitor. Moreover, the system can suggest other garments that may be of interest of that customer. Even if providing an artificial intelligence algorithm is out of the scope of this work, yet, in our system, we have implemented a simple suggestion mechanism that is based on simple rules such as "other customers that have bought X have often bought also Y, so if a customer has brought the item X in the changing room, then suggest also the item Y", "the product X is an on-sale alternative of the product Y". Of course, some suggestions can also come from the staff, using some fashion or taste criterion, and implemented in the system. Finally, the system may be able to suggest only items that actually are present in the store with the appropriate size.

Fig. 3. A garment store environment.

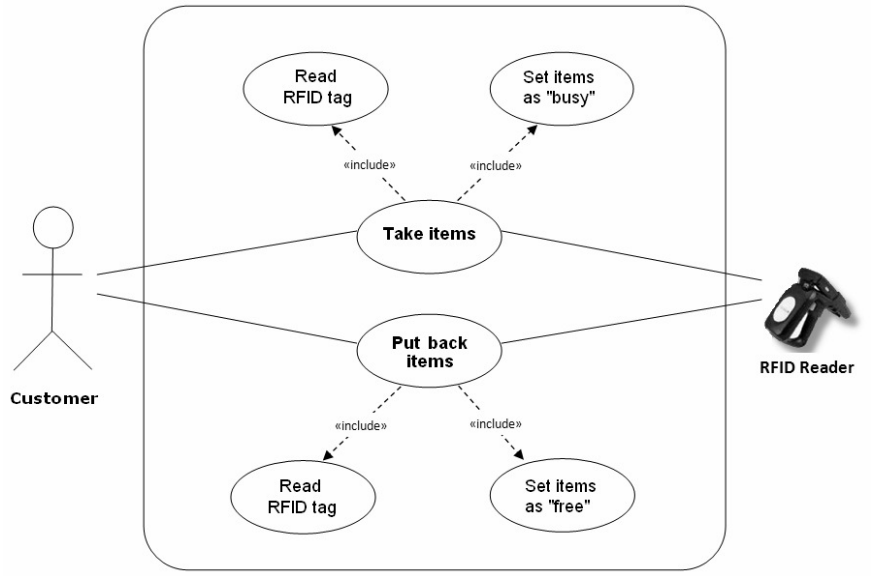

Fig. 4. Use case diagram of a system tracking garments taken from the shelves and put back.

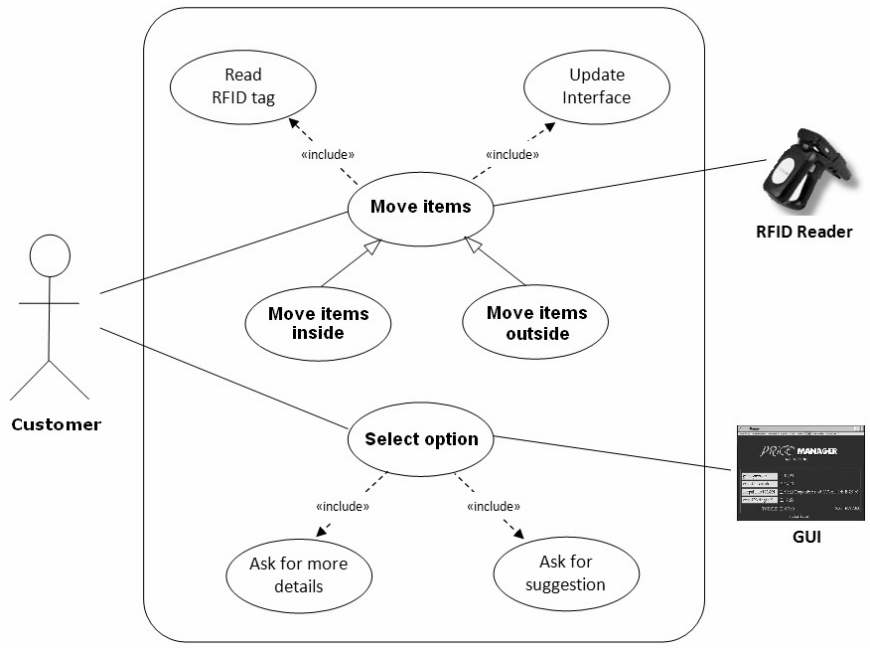

Fig. 5. Use case diagram of VSA functioning within the changing room.

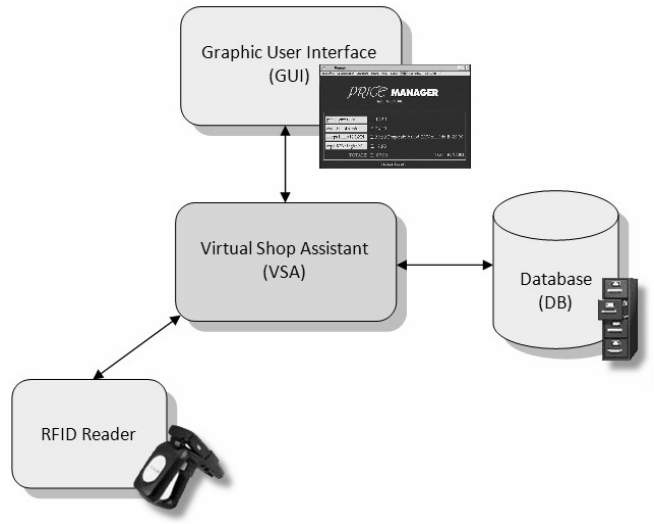

Fig. 6. System architecture.

4.1 VSA Software Architecture

In Fig. 6 we present the software architecture of our RFID-based application.

The core of our system is the Virtual Shop Assistant (VSA). This module allows all the different components to interact; in particular it interacts with three different objects:

- the database;
- the RFID reader;
- the Graphic User Interface (GUI).

We will show their functionalities in the following sections, whereas we focus here on the purpose of the VSA. In essence, VSA analyzes data read by the RFID Reader. It manages the RFID tag and asks the database for supplementary details. This information is used differently depending on the type of the RFID reader. For instance an RFID reader may be placed on a shelf and, in this case, our system will use the information coming from that reader to update the state of that shelf (e.g., which products are currently on it). Instead, if the reader is placed at the entrance of a certain changing room, then our system will pass all the information related to the product to the GUI in the changing room. Moreover, the VSA will interact with the customer through the GUI when the customer asks for more details or suggestions about garments. Finally, the VSA generates the appropriate query to retrieve the needed information, so that the GUI can be correctly updated. The VSA has been developed using Java.

4.2 The Database

The database of the store has been created using DBMS MYSQL.

The tables in the database are:

- *Garments*: it contains all the information about garments (e.g., size, color, brand, price, etc.);
- *Color*: it contains the colors to be associated with garments;
- *Brand*: it contains the description of the brands;
- *Type*: it contains the types of garments;
- *Location*: it contains the locations on the shelves;

- *Suggestion*: it contains couples of tags with the relation “if X chosen then suggest Y”, these relations are currently fixed but they could be automatically updated based on actual customers’ choices.

- *Type of Suggestion*: it indicates the type of suggestion, by the staff, by the customers, or a discount.

Inside the VSA there is a module which manages the interface with the database, using the JDBC driver to MYSQL [13]. This module allows the manipulation of the data with the typical SQL instruction SELECT, INSERT, UPDATE, and DELETE on the tables.

4.3 The RFID Reader

The interface to an RFID reader is managed through a client-server connection. Every RFID reader has an IP address and a specific port. The reader works like a server. It puts on the socket the information about a tag, which will be read by the application client. Thereby, the VSA establishes a client connection to a RFID reader with the appropriate parameters previously discussed; when reading a tag information are communicated via the socket to the VSA client. Information may regard:

- the advice that an item is busy when it is taken away from the shelf;

- the visualization of the information related to the item when it is brought into the changing room.

Obviously these two operations depend on the RFID reader type. For our case study, we have considered low frequency RFID tags and readers (LF from 125 to 134 kHz), as we need a short transmission range to avoid interferences between tags on garments in adjacent changing rooms; moreover, this kind of RFID technology is the least expensive one.

4.4 The Graphic User Interface

The GUI has been prepared to show the information about the garments chosen by the customer. A first screenshot visualizes data about the garments in the room, identifying their prices, discounts, and the total amount (see Fig. 3). From this screenshot the customer can select which garments are more interesting and have further details on them and alternative suggestions. The VSA is responsible to provide all the needed information to the GUI.

In our experiments, the GUI has been developed in a very simple way (as a real company would do just to have a proof-of-concept prototype). In a real system, the GUI could be much more advanced, visualizing also images about the various garments and making use, for instance, of a touch screen.

5 Integration with Rifidi

The RFID reader module is emulated using Rifidi. It is important to understand that this virtual environment allows to use the real protocol of the RFID tag and reader. As said, Rifidi emulates the reader/client interface of actual RFID readers; a client communicates with the Rifidi reader in the same way that it would communicate with a real reader. Therefore, the virtual module can be substituted with a physical module which manages the physical RFID objects and the whole system will still work.

In our tests, we have considered four RFID readers AlienALR9008, which have the characteristics to interact with other objects, such as push-arms. We have also created some RFID tags which are represented as boxes in the 3D environment. These boxes with tags are moved in the 3D scenario proceeding on conveyors with the help of push-arms. The RFID readers are situated at different locations that are passed by the boxes to recreate different moments: i) the customer gets one or more items from the shelf, ii) the customer goes to the changing room, iii) the customer exits from the changing room, and iv) the customer puts the items back on the shelf.

The operations with Rifidi can be divided into two parts. In the first one, we start the connections to a shelf reader from the console. We suppose, in fact, the existence of a central computer which will manage these readers and will set the connections. In a simple way, the software part about the shelf reader sets as busy the taken garment. If the garment is put again on the shelf, it will be set as free in the database. We can write the following command to start a reader connection

java MainGarmentRank 127.0.0.1 20000

where the host of the server is 127.0.0.1 and the port is 20000.

The second part manages the interface with the changing room reader. In this case we suppose an elaboration unit in every room. The developed software manages the visualization on the monitor of the data about the garments in the room. The application updates the visualization when the customer will substitute some garments. In this case, the user can write

java MainChangingRoom 127.0.0.1 21000

to start the connection.

In addition the user has to start the GUI; this is done by simply loading a web page with an applet that we have created to this aim.

5.1 Critical Analysis

During our tests we have encountered some bugs, both in the Rifidi Designer and in the Rifidi Emulator.

In particular, the Rifidi Designer has the following problems:

- the user can save the scenario configuration only when she/he is sure that it is definitive as Rifidi does not allow to save a modified configuration after the first one;
- if there are many object of the same type, in the GPIO perspective only the first one can be renamed;
- if there are more than one GPIO connection, in the GPIO perspective only the first one is correctly visualized, whereas the others, even if existing after their creation, are not correctly visualized;
- gates' properties such as IP addresses and ports can be defined only at the very first creation of the scenario;
- the program is computationally heavy: it takes a lot of CPU time and memory resources.

Instead, the Rifidi Emulator has two main problems:

- the user cannot generate the same set of tags in two different emulations: this because the system adds some extra-digits after the initial pattern requested by the user, so it is quite impossible to have the same set of patterns during a new generation;
- in part deriving from the previous point, the user cannot generate the specific tag he wants, even if providing all the digits of the tag: this depends from the fact that the system utilizes only a certain number of digits provided by the user and, after those, it randomly assigns the remaining digits.

In general, the provided documentation was useful to learn the basics of the Rifidi software, but it is not enough for users that would like to fully exploit the platform or even integrate it to address its problems.

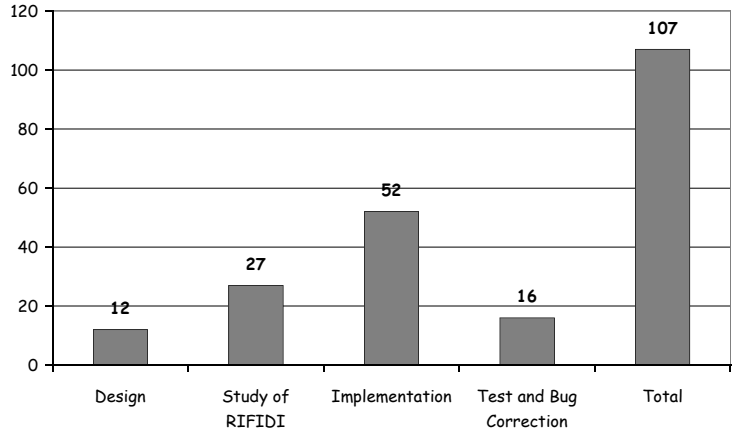

Fig. 7. Work hours dedicated to the implementation of our case study.

Yet, even in its current version, Rifidi is a useful tool with great potentialities for companies that would like to perform a quick and non expensive evaluation of RFID-based processes before investing in the technology. To this aim, we report in Fig. 7 the number of work hours (Fig. 7) globally invested by two junior programmers in getting acquainted with Rifidi and use it to create the VSA application as explained in this paper.

6 Conclusion

RFID is a technology that is of great interest for researchers and industry practitioners. In particular, companies can take advantage of this technology to automate their business processes. Unfortunately, the evaluation of a new business scenario involving RFID is complex both for the heterogeneity of this technology and for the cost of a real testbed.

To this aim we have evaluated the use of an emulative tool, Rifidi, that can be used to quickly and realistically generate hybrid testbed scenarios combining emulated RFID features with real software systems. To perform this evaluation we have implemented a new RFID-based application that we have devised: a virtual shop assistant.

The contribution of this paper is hence twofold: i) a critical analysis of Rifidi, an emulative tool to test RFID-based business processes, and ii) a new RFID-based application, named Virtual Shop Assistant (VSA), that may be of great interest for companies.

In particular, we have shown practical limitations of Rifidi; this emulative platform still needs some work to unveil all its potentialities. Still, through it, we have been able to generate a testbed scenario to evaluate our application with only 107

hours of total work split on a couple of junior programmers (1 week and a half of parallel work by the two programmers) and without the need to buy and install real RFID technology during this first evaluation. A technology manager could hence very quickly have available an evaluation of our business scenario and decide, for instance: i) to perform more emulative tests with different radio frequencies or ii) to buy real RFID tags/readers and implement a prototype using directly all the software that was developed during this first evaluation.

References

[1] K. Michael, L. McCathie, "The Pros and Cons of RFID in Supply Chain Management", in Proc. of the IEEE Proceedings of the International Conference on Mobile Business (ICMB'05), Sydney, Australia, Jul 2005.

[2] D. W. Engels S. E. Sarma, "The Reader Collision Problem," in Proc. of the IEEE International Conference on Systems, Man and Cybernetics (SMC 2002), Hammamet, Tunisia, Oct 2002.

[3] H. Vogt, "Efficient Object Identification with Passive RFID Tags", Pervasive Computing 2002, LNCS 2414, pp. 98-113, Zurich, Switzerland, Aug 2002.

[4] H. Vogt, "Multiple Object Identification with Passive RFID Tags", in Proc. of the IEEE International Conference on Systems, Man and Cybernetics (SMC 2002), Hammamet, Tunisia, Oct 2002.

[5] P. Popovski, "Tree Protocols for RFID Tags with Generalized Arbitration Spaces", in Proc. of the IEEE 10th International Symposium on Spread Spectrum Techniques and Applications (ISSSTA '08), Bologna, Italy, Aug 2008.

[6] "EPCglobal: EPCTM Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz-960 MHz ," vol. 2005, version 1.1.0, http://www.epcglobalinc.org/standards/uhfc1g2/

[7] K. Rhee, J. Kwak, S. Kim, D. Won, "Challenge-Response Based RFID Authentication Protocol for Distributed Database Environment", in Proc. of Security in Pervasive Computing (SPC 2005), LNCS 3450, pp. 70-84, Boppard, Germany, Apr 2005.

[8] C. Flörkemeier, M. Lampe, "RFID Middleware Design - Addressing Application Requirements and RFID Constraints", in Proc. of the ACM Joint Conference on Smart Objects and Ambient Intelligence: Innovative Context-aware Services: Usages and Technologies, Grenoble, France, Oct 2005.

[9] M. Ohkubo, K. Suzuki, S. Kinoshita, "RFID Privacy Issues and Technical Challenges", Communications of the ACM 48(9), 66-71, Sep 2005.

[10] O. Günther, S. Spiekermann, "RFID and the Perception of Control: The Consumer's View", Communications of the ACM 48(9), 73-76, Sep 2005.

[11] L. M. Ni, Y. Liu, Y. C. Lau, A. P. Patil "LANDMARC: Indoor Location Sensing Using Active RFID", Wireless Networks 10, 701-710, 2004.

[12] Pramari, LLC, Rifidi, Software Defined RFID http://www.rifidi.org, 2007.

[13] Sun Microsystems Inc., "Trail: JDBC™ Database Access (The Java™ Tutorials)", http://java.sun.com/docs/books/tutorial/jdbc/, 2008