

Smart RFLib: Project Proposal

Cagri Balkesen, Gautier Boder, Nihal Dindar, Florian Keusch,
Catharina Kromwijk, Ali Sengül

October 17, 2007

Contents

1	Introduction	2
1.1	Background Information	2
1.2	Motivation	2
2	General architecture	4
3	Data acquisition layer	4
3.1	Overall description of components in library application:	5
3.2	Hardware and Development Environment	6
3.3	Technical Description of Data Acquisition Layer: Architecture	6
4	Database, Query processing layer	7
4.1	Technical aspect	7
5	GUI: second life	8
5.1	Second Life	9
5.2	Web interface	9
6	Technical background	10
6.1	Second life	10
6.2	Web interface	11
6.2.1	Second life LSL	11
6.2.2	Second life viewer	11
6.2.3	Second life system	11
7	Timetable	12
8	Related Work	13

1 Introduction

The goal of this project is to explore the RFID technology through a practical application: a library management application. For this, we use RFID tags which we will put on books and people and create a full application which does everything from the data acquisition to a 3D graphical user interface.

The goal is to totally automate a library and be able to obtain, for example, the following information:

- Is a book currently in the shelf or not?
- Did the user check out all the books?
- Did the user not borrow too many books?
- Implement the library's policy (ex: a student can borrow only three books, certain books may not be taken out of the library, ...)

1.1 Background Information

RFID is a term which designates a system where people or objects transmit their identification over radio frequencies. In earlier technologies, the labels or tags had to be scanned manually to be able to capture the identity of the object/person. RFID, however, doesn't require manual scanning. An RFID system is generally made up of several readers with antennas which emit radio signals and capture back the signals emitted by the tag. Thanks to this technology, we are able to build systems which don't require manual reading, making it faster and easier than the older technologies.

The first RFID-type applications go back to the second World War where it was used by the Germans and English to track their planes. Since then, this technology has involved quite a bit from huge antennas and receptors to the small tags and antennas we have now. This is why RFID is applied in more and more fields. The most popular one being stock management, but RFID can be used to track anything. The goal of most implementations of RFID systems is to reduce the costs being caused by losses, thefts or other accidents which can happen. Since this technology is relatively cheap (a tag can be as cheap as 0.20) we can easily see how fast this could go.

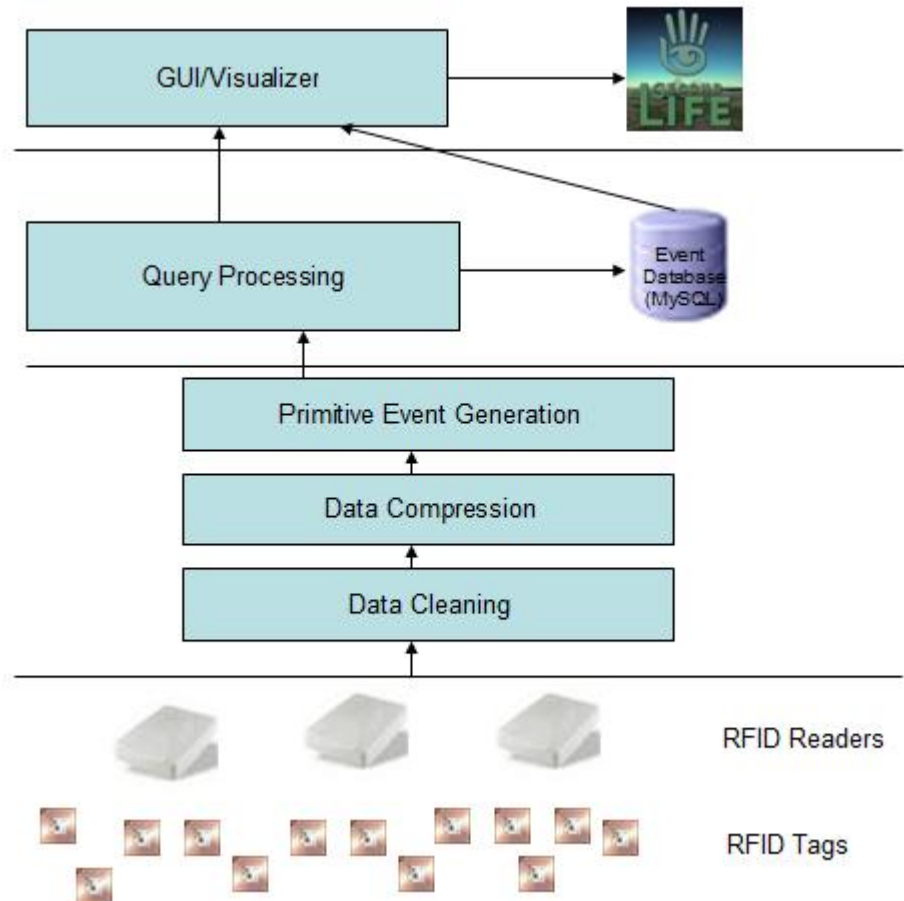
1.2 Motivation

Stock management is a very predictable application, and there are not a lot of uncertainties. Most of the time, the transport company can predict the exact path a lot of products will take. Lately, however, there has been an interest in trying to apply this technology in less certain contexts. Uncertainties can be of many natures: it could be the tracking of animals, the intervention of people or any other uncertain context. These contexts give some extra challenges and

we would like to explore them in our project. We want to try to find out how these situations can be handled, and what the difficulties are exactly. For this we will implement a general RFID system which will be applied in a library management system, where people and books will be tagged and handled.

A library is a place which is very hard to handle without an appropriate system. RFID provides some new possibilities for this management. One problem in a library is that an inventory might take several days, weeks or even months when doing it by hand. Because of this, when a book is misplaced on a shelf it becomes very hard to find it back. Other problems include thefts, non-returns and workplace injuries because of RSI [1]. With an RFID-based application in the library, these problems are mostly going to be resolved. Our challenge is to make a full application which does everything from the data acquisition to an intuitive user interface.

2 General architecture



3 Data acquisition layer

Data Acquisition Layer is the lowest layer of our RFID management system design. Its main task is to interact with the physical layer and provide high level primitive event tuples to the upper layer. The physical layer consists of RFID hardware namely RFID readers, their antennas and passive RFID tags. The physical layer is the part of our system where our system physically interacts with the real world by receiving simple tag read events. This layer deals with the RFID data which is large in volume and inaccurate. This inaccurate data is called dirty data. In RFID management dirty data can be due to missed reading, unreliable reading and data redundancy. This dirty data should be cleaned and processed stream of simple events which will be send to upper layer. The readers

in the hardware level have definite position that we know in advance and when some tags enter their read range we get tag notifications in the form of their approximate place, their unique identifiers and the time of the read from the readers. By nature these notifications are simple in that they only say that some object is somewhere at sometime, but if we think of these notifications could be continuous or in some pattern we can conclude that getting the semantic of the aggregation of these events can require doing complex processing over them.

3.1 Overall description of components in library application:

In the context of our library application we will label books and library users with appropriate RFID tags. The shelves of our library will be equipped with RFID readers to monitor shelves. It will enable us to keep track of the library items; whether they are missing or in some wrong place. As another component we will have self check-in and check-out desks. This will also be equipped with RFID readers and may be a simple informative screen. This component will make it library users life easier for borrowing library items and returning them. On the other hand it will also diminish the workload on library counters. One of the most important components of our library system will be security gates at the exit of the library. This will avoid people from taking out library items without checking-out them. It will signal an alarm when an item is taken out without being checked-out. Our aim it to create a smart library as pictured here:



3.2 Hardware and Development Environment

Data Acquisition Layer will get readings from the hardware level. These readings could be received either by the readers on the shelves, at the exit or at the self check-in, check-out desks. So this layer will enormously interact with the RFID readers. We have chosen Alien Technology ALR 8800 RFID readers to use in our project. The supplier provided us the technical documentation on this hardware. Also they provided us the Java API to interact with the devices using a high level programming language Java. As a result we will design and implement our data acquisition layer in Java programming environment. We will connect the readers to a pc both through local area network and serial communication. As the development environment we will make use of the rich open source development facilities of Linux operating system. Also as the integrated development environment (IDE), we can make use of open source Eclipse IDE. In the following days we will delve more into the developers documentation to learn how to use Java API and we will do some hands on experience with the hardware.

3.3 Technical Description of Data Acquisition Layer: Architecture

After getting raw RFID reading data we have to process it firstly because of the known problems related with RFID technology. These problems are also mentioned in the related works. Namely it is the unreliability of the readings provided by the readers. In real life applications, actual read rate of tags in proximity of readers is seldom complete. Most of the times, there are some missing readings as well as sometimes redundant ones. As a result this is a challenging problem sourcing from the physical layers unreliability. So we have to deal with this issue in Data Acquisition Layer. The first thing to do should be trying to improve the quality of the raw data received from the RFID readers. We should clean the data to compensate for the missing readings. Then we should filter out the duplicated readings coming from several readers at the same time. Afterwards we can smooth our data to fix the corrupted parts and recreate our missing or incomplete data. Furthermore we want to make sure that all tags in the readers range are read. This can be achieved by making smoothing window too large. At the same time we would like to capture tag dynamic due to tag motion. Large windows may fail to capture tag movement. To achieve these tasks we can employ some novel techniques from the related works (such as adaptive window sizes) or even we may think of some new specific techniques for our system. Another important task in our layer is the compression of raw data created by the readers. Since the readers will continuously create readings, it will be huge in amount to deal with. Our compression techniques must be lossless in that we have to preserve the reliability of our readings. On the other hand we can make use of some aspects of our application area to do data compression. One of them is that most of our items will be stationary for a long time, i.e. the books will stay in the shelves for a period of time. So the items

stationary at a place will generate no new events until they will be moved. It is a fact that we can combine the readings until an item moved into one reading thus making a reasonable compression at the same time not losing precision. We will also think about other ways to do compression on our readings. After our data is cleaned and compressed we will generate primitive event tuples to the upper layer which are more reliable than the raw readings.

4 Database, Query processing layer

The main goal of the Event and Query layer is inference of clean and compressed RFID data detection of events and storage of the necessary information for tradition database queries. Defined actions in case of event detection are transferred into upper layer in order to inform the user and visualize the situation. In addition to that, layer provides a clear interface with upper layer in order to get the traditional defined queries from the GUI and transfer the result of the queries. Some examples of possible query might be How many books are stored in library?, Which books are most popular?, Which books are borrowed by the users? In addition to the traditional queries, the following events are planned to be detected by the system:

- too much books (a user wants to take books more than defined book limit)
- book theft (a book that is not yet borrowed leaves the library)
- checkout of book (just a normal event when a user borrows a book)
- return of book (a user returns his borrowed books)
- borrowing timeout (a book is borrowed by a user although the lending time is up)
- missing of book (a book has been taken out from the shelf, but is not taken to the checkout or back to the shelf)

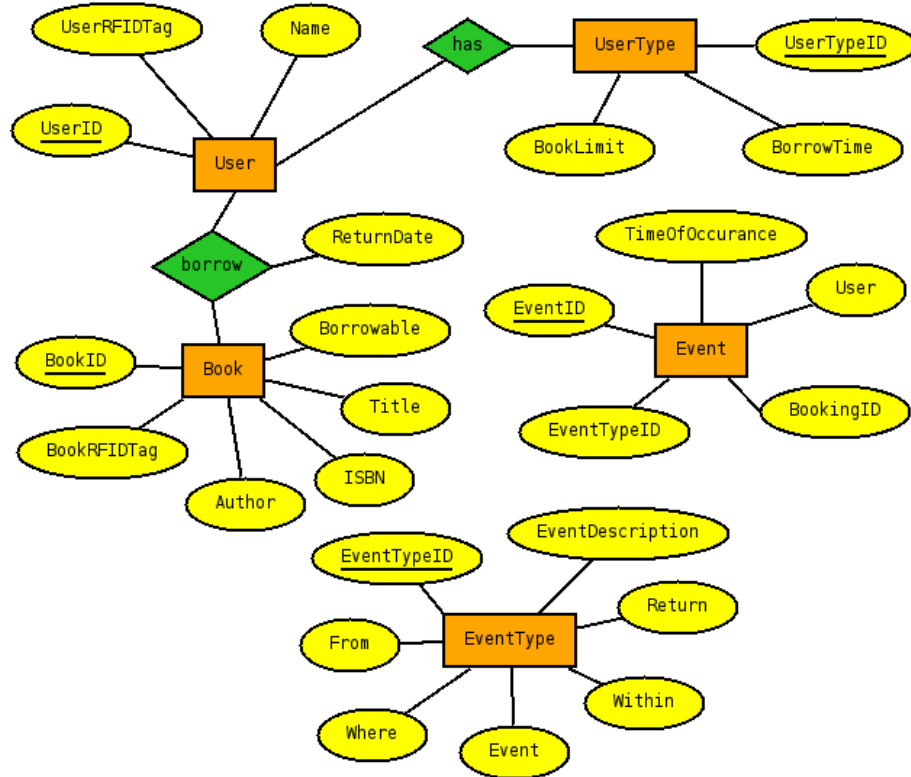
All these events are handled in the Query/Event Processor. This component takes the RFID tag stream that comes from the lower layer and checks for an event, stores it in the database and does the according reaction. For example if user #10 borrows book #15 then this is stored in the database and logged as an event.

4.1 Technical aspect

Business logic of the layer is developed by using Java language in Linux operating system and MySQL is used as database technology. Events are defined in complex event language SASE and processed in business logic rather than extension of MySQL. This decision is made by considering the ease of application and time constraint. The modifications, additions of events are planned by updating event text file. The data needed for RFID library project is stored in database.

It stores books, users, borrowings information. Besides basic library data, event abstractions and detected events are also stored in the MySQL database to keep a history.

The data contained in the database is described by the ER-model below. This is only a current sketch and could also easily be extended.



5 GUI: second life

The visualization interface of the project has been split into two different parts. One based on a virtual library in second life and the other based on a web application. The main aims of these interfaces are to provide the vision of the current books location, to localize books in the library by using a querying interface, list books that mistakenly located or stolen... The following points detail the functionalities offered by each part.

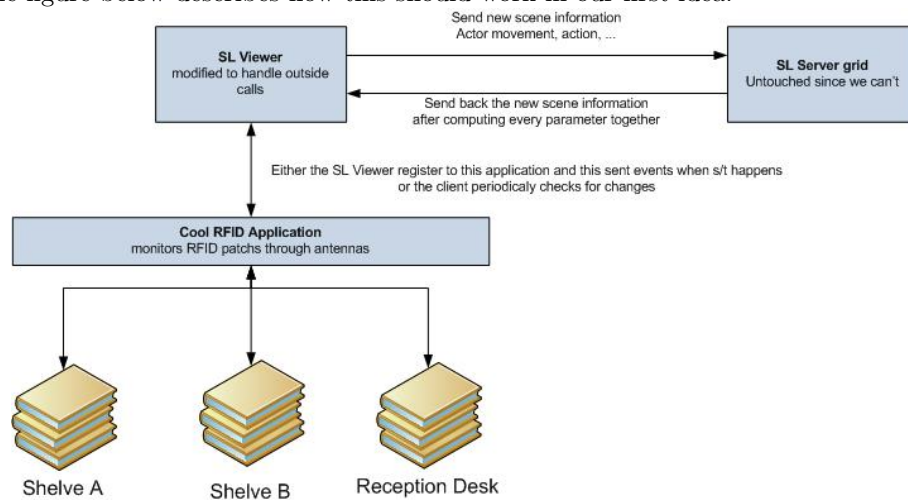
Maybe both interfaces could use the same background model. As the web interface and SL client will listen to the same background application.

5.1 Second Life

Second Life is used to display the Virtual Library corresponding to our Lab library. This library will have actors that move books around when they are moved in the reality. Books movement are described by the instant when their RFid patch is living antennas cell and entering another. Therefore virtual books movement will not be as the real one like when you are using trackers. Instead it will be a generic move from one position described by the source antenna (like shelf) to the destination antenna (like the reception desk). The virtual library has also to display in a different manner mistakenly placed books and books with unauthorized moves than the regular books.

The structure used to communicate with Second Life (SL) is not sure yet since some feasibility checks still needs to be done. However we are thinking about using Linden Script Language (LSL) to alter books and to move them. These scripts need to be launched by an event generated on the client side after a modification inside the real library. The way how to link events from the rfid monitoring application with the LSL scripts is not clearly defined yet. It will probably go through a modification of the client allowing it to receive events from another extern application or making it checking every now and then a DB or webservice spotting for books changes.

The figure below describes how this should work in our first idea:



5.2 Web interface

The web interface is used to allow user inputs and queries. It allows users to request for a certain books and displays their location graphically (using bitmaps). If a searched books should be on shelve A, but it is located on the shelve B it will be marked as mistakenly placed. Books that are taken out of the library without authorisation or that arent authorized to leave the library

need also to be listed.

Nota: we need to expand this part but we need to discuss a little more before

6 Technical background

6.1 Second life

Second Life is based on client-server architecture. Each of them has its defined tasks and they communicate using a dedicated protocol. The information exchanged is as minimal as possible to ensure a strong efficiency and a low network usage. This chapter doesn't pretend to teach you the whole structure of Second Life but only the main concepts that will help you to understand the project.

Another point differentiates strongly these parts. Server side must be seen as a black-box without having any idea about how it works or programmed. On the other hand client side (viewer) code is open source and ready to be edited by anybody that wants to add new functionalities. However Linden Labs edited some rules asking to edit the viewer as less as possible to keep one version around the world and deeply recommend using LSL to program SL environment.

Now study a little more deeply the specific tasks of each part. This enumeration is extracted from this page on the Second Lives Wiki
http://wiki.secondlife.com/wiki/Server_architecture (Simulator vs. Viewer).

- Simulator's job (server):
 - Runs physics engine
 - Collision detection
 - keeps track of where everything is
 - Sends locations of stuff to viewer
 - Sends updates to viewers only when needed (only when collision occurs or other changes in direction, velocity etc.)
- Viewer's job (client):
 - Handle locations of objects
 - Gets velocities and other physics information, and does simple physics to keep track of what is moving where

Web interface For more information about this system and their components please refer to these websites from Linden Lab. At the beginning the information seems to be unstructured (though it's a little the case) but after a while you will discover how much information is hidden into these wiki pages.

6.2 Web interface

The background knowledge cant be defined yet because the web interface is not designed and we dont know which technology will be used. It will be PHP / JS or J2EE or anything to do with web application. References:

- General introduction to second life for people who havent heard from SL before this document: http://en.wikipedia.org/wiki/Second_Life

6.2.1 Second life LSL

- LSL Wikimedia site: http://wiki.secondlife.com/wiki/LSL_Portal
- LSL Community: <http://www.lslwiki.net/lslwiki/wakka.php?wakka=HomePage>

6.2.2 Second life viewer

Viewer open source project: http://wiki.secondlife.com/wiki/Open_Source_Portal

6.2.3 Second life system

Learn the system wiki pages: <http://wiki.secondlife.com/wiki/Documentation>

7 Timetable

Date	Data acquisition group	Query Processing groupe	GUI group
17.10.07	Proposal Report	Proposal Report	Proposal Report
24.10.07	Getting hands-on experience with the hardware, reading documentation	Database Sample data / db setup / db connections / SASE Event Parsing	Define how to start LSL scripts on our application's events, Write a script that moves objects as we would like, changes books colors and hides objects, define goals of the gui
31.10.07	Learning to use Java API of the RFID reader	Database Testing / SASE Implementation Design	Design the whole structure of communication with SL system, Find furnitures and books objects ready to use and verify if they suit our expectations, Find a place where to build on SL world, interaction with other layers
07.11.07	Implementing a prototype with single reader which handles read items from the reader	Progress report 1	Progress Report 1
11.11.07	Providing an interface to the upper layer to access data		
14.11.07		SASE Event processing implementation	
21.11.07	Extending the prototype to work with more readers at the same time	Event processing / Testing phase 1	
28.11.07	Design and implementation of Data Cleaning algorithms	Progress report 2	Progress Report 2
05.12.07	Design and implementation of Data Compression algorithms	Proper Interfaces / Testing phase 2	
12.12.07	Testing the layer with a simulation either by physical or virtual generated data	Working Demo 1	Working demo
17.12.07	Working Demo 2	Working Demo 2	Working Demo 2

8 Related Work

- **SPIRE:** Spire addresses key challenges that arise in large scale RFID based information system. Design of the SPIRE system is presented in order to manage enormous volumes of RFID data and provide fast data transformation. Paper describes the design levels: filtering, smoothing and data compression, and event and query processing.
- **SASE: Complex Event Processing over Streams.** SASE defines a new powerful complex event language and illustrates its usage. The implementation of the language is also explained in the paper.
- **Design Considerations for High Fan in Systems, the Hi-fi Approach:** Paper focuses on the key characteristics and data management challenges of high fan-in systems that are widely distributed and their edges consists of different devices like RFID readers or probes. Architecture is suggested to address these challenges.
- **An Adaptive RFID Middleware for Supporting Metaphysical Data Independence** This paper describes how the physical data one get from different RFID devices can be processed to be used for applications in the digital worlds. It shows certain techniques to clean the RFID data coming from the readers using sliding windows, sampling, smoothing filters and other statistical methods. Some experiments are giving a better understanding of the different parameters that one is free to choose and how they influence the methods.
- **Challenges for Pervasive RFID-based Infrastructure** Small scale pilot building-wide RFID-based application is described in the paper. Challenges encountered and lessons learned during system deployment are shared.
- **Design Considerations for High Fan in Systems, the Hi-fi Approach:** Paper focuses on the key characterists and data management challenges of high fan-in systems that are widely distributed and their edges consists of different devices like RFID readers or probes. A architecture is suggested to address the challenges.
- S. R. Jeffery, M. Garofalakis, M. J. Franklin, "Adaptive Cleaning for RFID Data Streams", VLDB Conference, 2006.
- J. Singh, N. Brar, C. Fong, "The State of RFID Applications in Libraries", Information Technology and Libraries, 2006.
- S. Rizvi, S. R. Jeffery, S. Krishnamurthy, M. J. Franklin, N. Burkhart, A. Edakkunni, L. Liang, "Events on the Edge (Demo)", SIGMOD Conference, 2005.