

SmartRFLib - An RFID-Supported Library System

Cagri Balkesen, Gautier Boder  
Nihal Dindar, Florian Keusch  
Katinka Kromwijk, Ali Sengul  
Prof. Nesime Tatbul

ETH Zurich

February 11, 2008

# Overview

- 1 Introduction
- 2 Demo
- 3 Technical Details
  - Project Overview
  - Data Acquisition Layer
  - Query Processing Layer
  - Visualization Layer
- 4 Conclusions

# Project Goals

- to automate a library with the help of the RFID (Radio Frequency Identification) technology
  - RFID vs. Barcode Technology: line of sight and range
- to run higher-level event detection queries on RFID data streams
- to visualize important events and alerts in real time on SecondLife

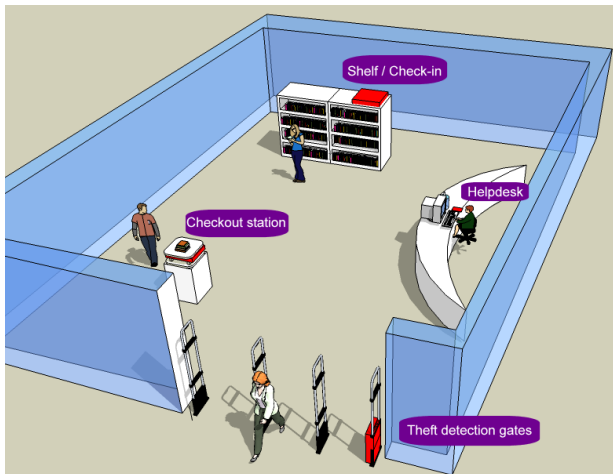


## Second Life

- SL is a virtual 3D world created by its users
- SL is not a game
- SL has its own economy and currency (Linden Dollars)
- ETH Zurich owns an island on SL
- Our library is located on the ETH Zurich Island



# Library Setup



# Events

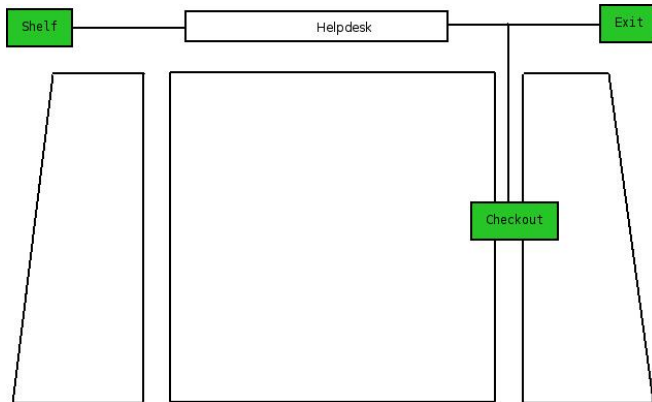
We have several types of events defined in our library:

- Book check-in
- Book checkout
- Illegal checkout
  - Reference books
  - Too many books
- Book theft

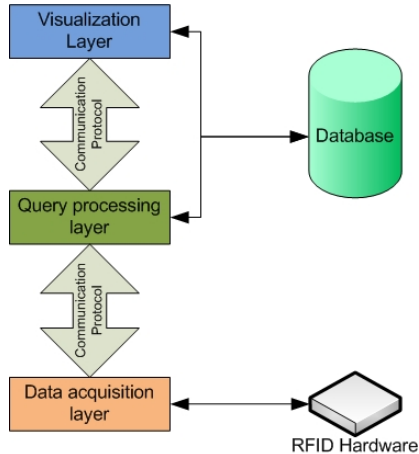
We will demonstrate the detection of these events during our demo.

# SmartRFLib Demonstration

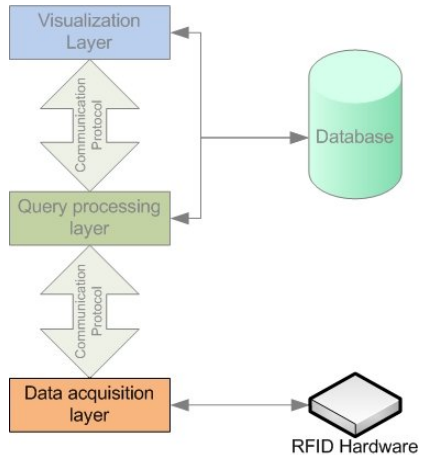
Library setup in this classroom:



# SmartRFLib System Architecture



# Data Acquisition Layer



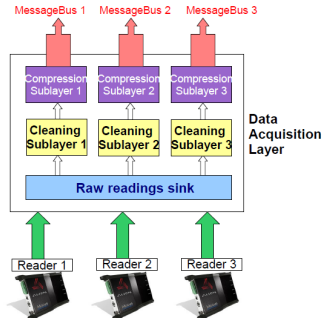
# Data Acquisition Layer Overview

## Data Acquisition Layer

- interacts with the physical layer, which consists of RFID tags, readers, and antennas
- collects large volumes of possibly “dirty” RFID readings from people and books
- produces streams of cleaned and compressed primitive event tuples
- plays a key role in the correct functioning of the whole system

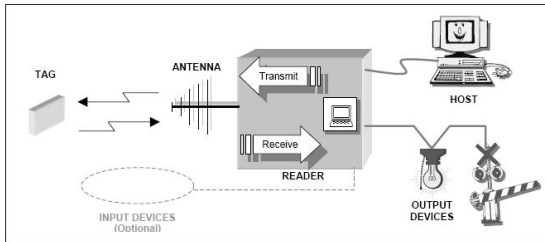
# Data Acquisition Sublayers

- Reader hardware: captures raw readings
- Data cleaning layer: cleans the dirty raw readings
- Data compression layer: groups and compresses the cleaned readings based on a time period or an action

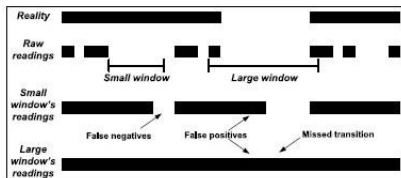


# RFID Hardware Components

- An RFID system is composed of readers, antennas, tags, and a host computing device
  - Readers: transmitter, receiver, microprocessor
  - Antennas: transmitter, receiver
  - Tags: identification and other information



# Data Cleaning Layer



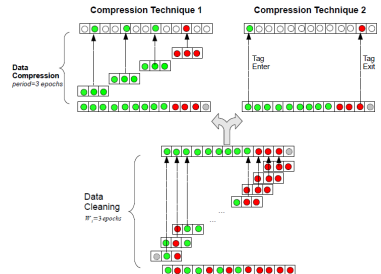
- Problem: RFID data is “dirty”.
- Data cleaning methods:
  - Fixed-window Data Cleaning: use a fixed size cleaning window that moves over the input data stream
  - Adaptive Data Cleaning: change the cleaning window size according to the input received

[JGF06] S. R. Jeffery, M. Garofalakis, M. J. Franklin, “Adaptive Cleaning for RFID Data Streams”, VLDB 2006.

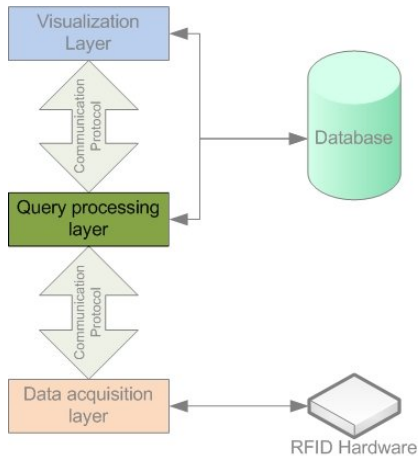


# Data Compression Layer

- Problem: RFID systems produce huge volumes of data.
- Solution: compress data by representing certain readings with a fewer number of tuples
- We developed two alternative techniques, depending on the application logic:
  - periodic responses from tags
  - momentary changes about their situation



# Query Processing Layer



# Query Processing Layer Overview

What does the QP Layer do?

- collects the incoming data streams from the Data Acquisition Layer
- runs complex event detection queries on the primitive event tuples
- triggers actions to react to the detected events
  - update the database
  - inform the Visualization Layer

# SRFLL Event Definition Language

- SRFLL (SmartRFLib Language) is an extension of SASE
- Example event definition in SRFLL:

```
NAME book_theft
PATTERN SEQ(Book+ b[ ]) ON 3
MATCH INCREMENTAL
WHERE notBorrowed(b[i])
ACTION alarm_theft(b), notifyTheft(b)
RETURN True
```

[GWC+07] D. Gyllstrom, E. Wu, H. Chae, Y. Diao, P. Stahlberg, G. Anderson, "SASE: Complex Event Processing over Streams", CIDR 2007.



## SRFLL - Book Theft Example

NAME book\_theft

PATTERN **SEQ(Book+ b[ ])** ON 3

MATCH INCREMENTAL

defines a sequence for this event

WHERE notBorrowed(b[i])

describes how to match the  
patterns

ACTION alarm\_theft(b), notifyTheft(b)

RETURN True

## SRFLL - Book Theft Example

NAME book\_theft

PATTERN SEQ(Book+ b[ ]) **ON 3**

MATCH INCREMENTAL

WHERE notBorrowed(b[i])

this event takes data from reader  
number 3

ACTION alarm\_theft(b), notifyTheft(b)

RETURN True

## SRFLL - Book Theft Example

NAME book\_theft

PATTERN SEQ(Book+ b[ ]) ON 3

MATCH **INCREMENTAL**

WHERE notBorrowed(b[i])

ACTION alarm\_theft(b), notifyTheft(b)

RETURN True

we have 2 different match types:

- match incremental: trigger after every received data element
- match longest: try to find longest sequence

## SRFLL - Book Theft Example

NAME book\_theft

PATTERN SEQ(Book+ b[ ]) ON 3

MATCH INCREMENTAL

WHERE **notBorrowed(b[i])**

ACTION alarm\_theft(b), notifyTheft(b)

RETURN True

event conditions defined by:

- user-defined functions
- subevents

## SRFLL - Book Theft Example

NAME book\_theft

PATTERN SEQ(Book+ b[ ]) ON 3

MATCH INCREMENTAL

WHERE notBorrowed(b[i])

ACTION **alarm\_theft(b), notifyTheft(b)**

RETURN True

user-defined functions that  
are triggered when an  
event occurs

## SRFLL - Book Theft Example

NAME book\_theft

PATTERN SEQ(Book+ b[ ]) ON 3

MATCH INCREMENTAL

WHERE notBorrowed(b[i])

boolean return value

ACTION alarm\_theft(b), notifyTheft(b)

RETURN **True**

# Parser, Lexer, Finite State Automata

## Parser, Lexer:

- We used jcup and jlex to parse the event definition language.
- Parser generates FSA.

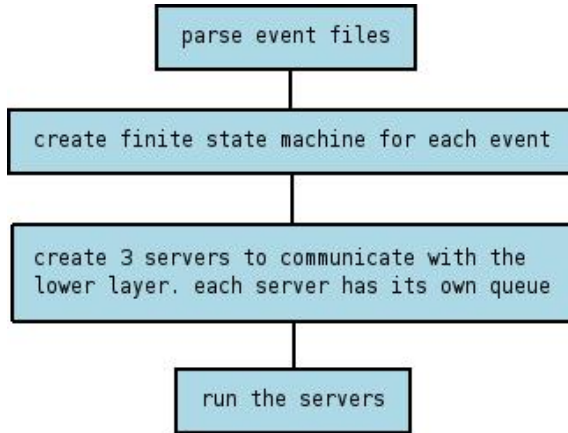
## Finite State Automata:

- FSA is the internal representation of an event.
- FSA is used for event evaluation together with:
  - parse tree of expressions
  - match buffer
  - user-defined functions called with Java reflection

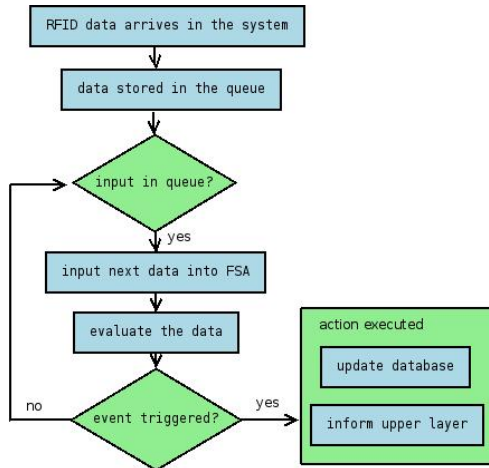
## Other Technical Issues

- Communication to both layers
- Database: store all the data (Books, People, Library status)
- Timers: detect end of events
- Composite events: build an event out of subevents

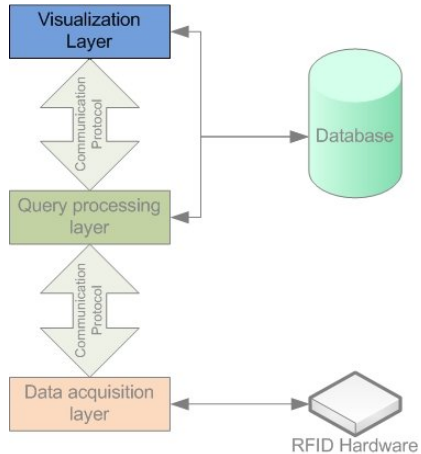
## QP Layer - System Initialization



# QP Layer - Event Detection



# Visualization Layer



# Visualization Layer Overview

- Visualization of detected events in a 3D world
- System administration
- Library users interrogation

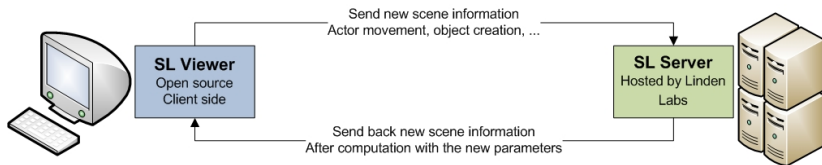
## Second Life - Visualization in a 3D World

Second Life imposed itself as 3D world, though we had to solve some problems

- How to create the virtual library?
- How to link the Query Processing Layer with Second Life?
- How to receive and process outside events?

## Second Life - Communication and Development

- Second Life has a client server architecture:



- The objects 'poll' from inside Second Life to see if their states or positions have changed
- Each object has an associated script written in LSL (the Second Life programming language, similar to Java)

## Web Interface - System Administration

Since some tasks were not possible in Second Life, we also designed a web interface to support our 3D SL visualizer.

- Management of
  - Books
  - People
  - Library Policies
- System monitoring

## Web Interface - User Interrogation

Users may ask the librarian or query the system to retrieve information

- about books
- about their own current status
- about the system

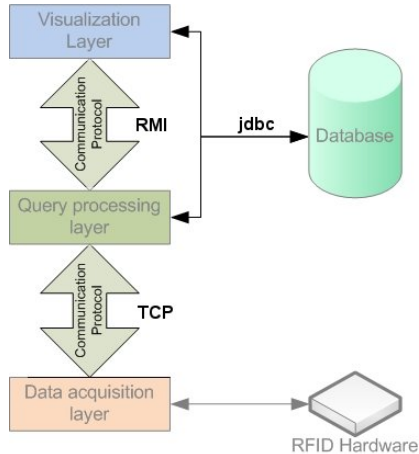


# Visualization System Structure

We used multiple platforms and systems:

- **VLCore:** receives and processes events from lower layer
- **Database:** stores events and system information
- **Web Interface:** is used for management, querying, and monitoring
- **Second Life:** shows the virtual library and real-time events
- **Live Client:** receives events directly from VLCore

# Communication between the System Layers



## Open Issues

If we had more time, we would

- work on reader range intersection
- find a more sophisticated solution to the Person and Books match during checkout
- assessing the limits of the system

# Conclusions

- Most parts of SmartRFLib are general-purpose and can be applied to other RFID-based data management applications.
- Working with RFID data is difficult.
- Extending Metaphysical Data Independence idea of SMURF
- Challenging project: many problems to solve, weekly meetings
- We learned a lot and had fun thanks to good teamwork :)



# Acknowledgements

Special thanks to:

- Prof. Nesime Tatbul
- Dr. Michele de Lorenzi
- Dr. Peter Fischer and ISG/D-INFK
- Roozbeh Derakhshan
- The builders of the Second Life Island

## Discussion

# Questions?

For more information, please see our project webpage:

[http://www.dbis.ethz.ch/education/ws0708/infysyst\\_lab/rfid](http://www.dbis.ethz.ch/education/ws0708/infysyst_lab/rfid)

