Electronic ID for Inventory

Final Report
Fall Semester 2013

by
Justin Fritzler

Department of Electrical and Computer Engineering
Colorado State University
Fort Collins, CO 80526

Project Adviser:       Dr. Ali Pezeshki
Approved by:           Dr. Ali Pezeshki
Abstract

Lab Animal Resources (LAR) is a facility on the Colorado State University Campus, which houses and cares for animals that are part of scientific studies. LAR requires an efficient system for performing occupancy checks (“census”) across three facilities, comprising a total of 3500 cages, and 20,000 ft², as well as tracking the care history and location of each. Cages range in size from approximately 0.2 ft³ to approximately 9 ft³, and most frequently house rodents and small mammals.

A wireless system is sought to reduce the time required to perform the census, enable them to be conducted on a weekly bases, and further automate the process of checking for completeness. Each cage is identified with a small cage card, which is attached to the front of each cage, and to which a radio-frequency ID tag will be affixed.

Commercially available wireless solutions exist, but these are all a magnitude of order more expensive than the budget LAR has for this application. The best system configuration was determined to be passive RFID tags on the cages, with a mobile hand-held RFID reader with Wi-Fi capabilities plus hardwired connections available for direct transfer of information, and application software written in C#.

A variety of requirements exist for the RFID application software. The software must be able to associate cage cards with RFID tags and subsequently deactivate the relation if necessary. The software must be able to accurately identify the room for all cage cards during a census. In addition the software must be extensible, e.g., the software must be able to handle changes to personnel and real-estate.
# Table of Contents

1  Introduction [1] ....................................................................................................................... 3  
   1.1  Lab Function .................................................................................................................... 4  
   1.2  Facility Construction and Layout .................................................................................. 4  
   1.3  Database ........................................................................................................................... 4  
   1.4  Census and Billing .......................................................................................................... 5  
   1.5  Investigation Life Cycle .................................................................................................. 6  
2  Needs for LAR [1] .................................................................................................................. 7  
3  Requirements ....................................................................................................................... 8  
   3.1  Functional Requirements ............................................................................................... 8  
   3.2  Operational Requirements ........................................................................................... 9  
4  Software .................................................................................................................................. 9  
   4.1  Scope .................................................................................................................................. 9  
   4.2  Software Selection ......................................................................................................... 10  
   4.3  Program Design ............................................................................................................ 10  
   4.4  Software Enhancements ............................................................................................... 14  
5  Tag Filter .............................................................................................................................. 16  
   5.1  Filter Purpose .............................................................................................................. 16  
   5.2  Original Filter Design .................................................................................................... 16  
   5.3  Improved Filter Design ............................................................................................... 17  
6  Conclusion ............................................................................................................................. 18  
7  Appendix A: Abbreviations ................................................................................................. 19  
8  Appendix B: Budget [1] ....................................................................................................... 19  
9  Appendix C: Project Timeline ............................................................................................ 20  
10  Appendix D: Bibliography ................................................................................................. 21  
11  Acknowledgments ............................................................................................................... 21  

# List of Figures

Figure 1: Rack of Cages .......................................................................................................... 4  
Figure 2: FY 14 CSU Faculty Animal Care Per Diem Rates (U.S.) ...................................... 5  
Figure 3: Sample Cage Card in Holder ............................................................................... 6  
Figure 4: Software Flowchart - Census ........................................................................... 11
1 Introduction [1]

The Electronic ID for Inventory senior design project was introduced in Fall ‘12 to assist CSU Lab Animal Resources devise a wireless system for tracking animal cages. A solution was proposed in the first semester using a handheld RFID reader with passive RFID tags. An ultra high frequency reader was chosen, since the operating frequency is above the hearing range of any animals. The second semester was aimed at finding or altering tags to operate in a metal environment. In addition the handheld reader needed software developed for performing the census of cages. The project, now in its third and final semester, is focused on completing the software. The software design must allow for reusability and extensibility for change in personnel or addition of real estate. The software must also synchronize census data with a database and warn users of discrepancies.
1.1 Lab Function
Laboratory Animal Resources (LAR) strives “to use [their] expertise to provide the highest standard of care for laboratory animals in support of the Colorado State University research community.” They maintain over 3500 cages, housing animals such as calves, pigs, cats, dogs, rodents, etc. These cages are spread across two facilities – one at Colorado State University (CSU) main campus, and one at the CSU Foothills Campus. LAR maintains these animals while clients, referred to as private investigators (PI), perform research studies. This process includes the housing, hygiene, feeding, and veterinary care for animals under the direct care of LAR.

LAR faces a challenge in periodically reviewing the count and location of occupied cages, tracking the care and location history in each of those cages, and maintaining all the collected information and history in a database format. Manual data collection is labor intensive and time consuming, as well as introducing a greater risk of human error. However, a complete census of the cage counts and locations must be completed prior to each billing. LAR therefore desires a remote sensing solution in order to reduce the labor and cost as well as reducing the risk of human transcription errors.

1.2 Facility Construction and Layout
LAR consists of two facilities which jointly comprise about 20,000 interior sq ft. These two facilities house cages in a variety of rooms. The standard rooms are approximately 16’ x 25’, situated along corridors with several rooms in each facility. Additionally, some rooms are small, approximately 3’ x 8’, which open via double-doors into hallway adjoining the main corridors.

The majority of rooms contain metal racks which hold the cages as shown in Figure 1: Rack of Cages. These cages may be of a plastic material, housing small animals as shown, or larger animals in racks holding 4-6 cages. Additionally, some of the large cages are entirely metal, and fit 4-6 on a rack.

1.3 Database
Granite is a database management system currently used by Laboratory Animal Resources to track all of their cage information and billing procedures. This includes contact information for the investigators, billing rates, animals associated to each investigator, protocol information, and cage information.

Support for the version of Granite currently being used by LAR will expire within the next 5 years. Upgrading to a newer version of Granite is expensive, so LAR is looking to alternate
solutions. In approximately 10 years, the Kuali project – a joint venture between academic institutions – is expected to reach fruition, and will alter or replace much of the system in use at that time. Therefore any system or suggested method should be designed with a 10-year working life in mind.

1.4 Census and Billing

1.4.1 Census

*Census*: The process of recording each occupied cage present in either of the LAR buildings, along with its room location, and comparing against a database record of expected occupancy.

Currently LAR performs a monthly census. This entails a staff member printing out the cage numbers that should be in each room based on the database information. With these printouts in hand, a staff member compares the cages in each room and manually reads each cage card number, ensuring it belongs in the room. After going through each room, the data then must be compiled by hand to identify any discrepancies. Staff then has to remedy any problems that arose. This process takes approximately 30 minutes per room.

Over the course of a month, many private investigators will move cages, start using new cages, and remove some cages. This means that the information LAR has in their database can differ drastically from what is actually in each room within a month. This then leads to significant loss in revenue for LAR.

1.4.2 Billing

Laboratory Animal Resources bills their customers on a per day basis. Rates differ based on the type of animal and other factors. Figure 2 shows these rates.

In order to bill a client accurately, LAR needs to have information on exactly how many days a client has used a cage. Presently, this information is not exact and LAR has experienced a loss in revenue.

Because LAR only does a monthly census and private investigators have the ability to add cages as needed, LAR does not have a precise way to determine when a given cage was added. Currently LAR starts billing from the date of the first census in which a given cage was recorded. This means LAR is unable to bill for
any days the cage was in use prior to the census. Thus, there is the potential loss of up to 30 days unpaid use between when a given cage is added to the racks, and when it is entered into the billing system, or – alternatively – a full loss if a cage is in use for less than 30 days. The lack of precision in the billing system gives rise to the need for a new system process.

### 1.5 Investigation Life Cycle

This sequence offers insight into the current process employed at Lab Animal Resources.

**New Protocol – Entry in the system**

1. The first step is a private investigator generates a new protocol (request).
2. A staff member from LAR then takes that information and creates a new Animal Order within the Granite database. The Animal Order details all the needs of the investigator such as the animal desired, for what purpose, special restrictions, etc. as well as details LAR uses such as facility location, room location, etc.
3. After the account information has been approved, LAR contacts certain animal vendors to purchase the animals necessary for the private investigator.
4. Cage cards are then generated to put on all the cages the private investigator will be using.
5. The animals arrive and are then put in the appropriate cages and put in the rooms that were assigned.

**Upkeep – Census Capture**

6. Staff member prints out room reports from Granite database.
7. Staff member goes into a room and checks off all cages that are seen on the report, while recording all cages in the room that are not on the report as well as noting the cages that are missing from the room when they are listed on the report.
8. Staff repeats step 7 for every room.
9. Staff compiles data from reports and addresses all irregularities: moving cages where they need to go, adding new cages to database (beginning billing cycle), removing cages from database (see step 10)

**Cage Termination – Removal from the system**

10. Once a private investigator has finished using a cage, they put the cage card in a box designated for cages that need termination.
An LAR staff member collects cage cards from termination box and manually stops the billing cycle for that cage.

2 Needs for LAR [1]

This project arose from two different issues. One being that Granite will soon not be supported, and the other that LAR is losing money based on the inability of their current system process to facilitate weekly billing. This prompted the desire for a full system overhaul. Such an effort would include replacing the current database with a new updated database and incorporate a new census gathering system and new billing process.

The main need of LAR is to implement a new tracking system to quickly and reliably take a census of each room and compare the results with the database information. This new tracking system is meant to significantly reduce the census time so it can be done on a weekly basis, thereby cutting down on profit losses. A secondary need is to create a new database with the same basic functionality (with a few modifications) that Granite currently has, and also provide a web interface for this database. This additional need will be carried out by work and independent study students.

Due to complexity, this project has been split into two phases:

PHASE 1 will design a solution to quickly and reliably conducting a census in each room, and generating a list of all the cages, along with corresponding locations.

PHASE 2 will design a more comprehensive software element than is used in Phase 1, which can address the list of cages, identify discrepancies, and accept user input regarding how to handle each discrepancy.

Specific needs identified for or pertaining to this project:

1. LAR requires a solution that can function within the constraints placed on them as an operating animal laboratory.

2. LAR has a need to have a complete and correct census,

3. LAR, as part of the census, needs to have traceable cage locations,

4. LAR has a need to generate periodic billing statements, supported by the census,

5. LAR desires to confirm census on a weekly basis,

6. LAR requires a secure system facility,
7. LAR requires a solution to be maintainable,

8. LAR requires continuous database function with minimal interruption,

9. LAR seeks a system to facilitate maintaining adequate training of staff.

3 Requirements
The goals of this semester were to transform the proof of concept software into a professional product that can be easily maintained by personnel that did not write the software. Additionally, the accuracy and reliability of the system required significant improvement. Only the requirements pertaining to this semester are listed.

3.1 Functional Requirements
1. Cages must be uniquely identifiable by RFID scan.
   1.1. The RFID number associated with each card cage must be linked via software to the cage card number.
   1.2. At no time should two RFID tags with the same RFID number be present on the shelves.

2. Cage location must be identifiable.
   2.1. Room must be identifiable, either by user or by RFID scan.
   2.2. System should allocate tags to one room only.
       2.2.1. If tags are identified in multiple rooms, duplicates should be eliminated, either by filter or user input.

3. Database integrity must be maintained.
   3.1. Scans should not update the database until user indicates their acceptance of the changes. (This need not be incremental, acceptance of the census as a whole is appropriate.)
   3.2. The developed system will not preclude the continued use of Granite unless the system presents a validated, fully functional substitute Database (deemed jointly by Dr. Kendall and the EID team).
   3.3. The system will record which team member performs each census action.
4. Census must generate a complete and correct log of the occupied cages and locations upon completion.

4.1. A log of rooms and cage cars therein should be created from each successful census scan.

4.1.1. The log should not be deleted except by clear user intent.

4.1.2. The log should be able to handle up to 300 cages per room.

4.2. Any RFID tag scanned in the course of a census, but without an active database association, must generate an error.

4.3. Any RFID indicated by scan in multiple rooms must be addressed by an error-handling procedure.

4.3.1. Software procedures should be used for low-probability cases to minimize the requirement for user input.

4.3.2. Users must be able to confirm or override room locations if needed.

4.4. Log must be formatted to facilitate easy transfer to the database.

3.2 Operational Requirements

1. The system must facilitate a scan time of five minutes or less per room.

2. The system must function within the environment of the lab facilities, which include a significant presence of metal.

3. The system must function with the use of metal and plastic cage card holders.

4. System must be amenable to modular deployment – able to be used for all or part of the census, without disrupting the remaining census.

4 Software

4.1 Scope

Software for this project includes an application for reading and managing RFID tags, a network service installed on a client computer (or later to be run through a web interface) for uploading census files, a check-in/out application installed directly on the device for associating RFID tags with cage card numbers. Initial releases of the RFID reader application and network service
were developed for the proof-of-concept demonstration at the end of the spring '13 semester. The check-in/out application will be released prior to the end of fall semester '13.

The implementation of a new custom database to include billing was also originally within the scope of the project. However, the customer determined a web interface was most appropriate. This was deemed beyond the scope of this project due to the scale and skills needed. Two work study students with the desired skills were hired to fulfill this requirement. The use of a code repository should be strongly considered for the ongoing success of this project.

4.2 Software Selection
Software for this project was written using Visual Studio 2008 in C#. Mobile software used .NET Compact Framework 3.5 and Motorola EMDK v2.5. The EMDK (Enterprise Mobility Development Kit) included libraries for making calls to Motorola hardware. This package supported Visual Studio 2008 as the latest IDE; C, C#, and Java as programming languages; and required the installation of Windows Mobile 6 SDK, Windows Mobile 6.5 Dev Tool Kit, and Windows Mobile Device Center to connect to the RFID reader.

C# was selected out of the available programming languages, because it included tools to quickly build a GUI using the .NET Framework. In addition it was the easiest of the languages to use, which allowed for minimal programming errors with the shortest development time. Compact Framework was used on mobile software to save storage space. The Full Framework was used for the client application as it provided significantly more functionality. The Visual Studio IDE was selected for its robust set of debugging tools and interface that actively aids the programmer while writing software.

Oracle Database Lite was chosen for communicating with current Granite database. The Granite client software actually uses an Oracle 10g database on the backend. Oracle Database Lite is the only free software for a mobile device designed to synchronize with an Oracle 10g database. In addition this software handles connection issues, while 3rd party applications require the programmer to handle connections to the database.

4.3 Program Design
The RFID census application needed to be lightweight and operate efficiently on mobile hardware. Functional requirements included the capability of identifying a room number with each RFID tag inventoried and generating a file to be input to Granite indicating the staff member that performed the census. The flow of the census program is show in Figure 4: Software Flowchart - Census.
Figure 4: Software Flowchart - Census
4.3.1 Census session

Upon launching the census application software, the user will be asked to select the current user. After selecting the user, the main screen as shown in Figure 5: Census Application - main screen will be displayed. The user must first scan for a room tag. Once a room tag has been identified it will appear in the drop down box at the bottom of the main screen. Only RFID tags associated with a room in the database will be displayed in the drop down box. The button labeled “Scan Room” will also turn green and label changed to “Enter Room”. Once the user has selected the correct room and clicked on the “Enter Room” button the census session will commence for that particular room. Once the “Enter Room” button is clicked, it will turn red and transform into the “Exit Room” button.

During the census session the user must pull the trigger to scan for RFID tags. When tags are read, they will be added to a hash table if the RSSI is above the filter threshold. A hash table was used to store tags during the census process due to the fact that it takes constant time $O(1)$ to store and access the contents. The quick access time was essential to allowing users to quickly wave the RFID reader across all tags and successfully read all of the tags. When a tag is read, if it is found in the hash table the meta-data is updated appropriately. If the read tag is not found, the hash table creates a new entry of the tag ID and its meta-data.

The user can identify whether all tags have been read in a room by looking at the “Tags Read:” field shown in large font on the read tab (see Figure 5: Census Application - main screen). Once the user has read all the tags in a room, the “Exit Room” button is pressed, which adds the tags from the hash table to a list that holds all of the information from the census and then clears the hash table.

When data from the hash table gets exported to the tag list after exiting a room, the software compares the new items to the existing and looks for duplicates. If a duplicate is found, it means the same tag was read in multiple rooms, which is an error. These instances are then put in a sorted list called the “duplicate list.” The sorted list offers many advantages essential to this mobile application. The first being the space efficiency of a sorted list, since resources are limited on the mobile platform. The second advantage is that each tag is only sorted once, minimizing computations, hence minimizing power consumption. The sorted list is grouped by tags with same ID and sorted by Max RSSI. This allows the program to identify the first tag in each group as the most likely candidate for having the correct room attached.
4.3.2 Manage Tab

When the user has completed their census, they may enter a management tab on the reader to address any preliminary errors with the reads (owing to a given tag's response being recorded in multiple rooms). The strongest signal strength is assumed to correspond to the room physically containing the tag. This room is indicated in green for the user. The other room(s) in which the tag's response was read are indicated in red. This sequence is presented in list format, sorted by tag ID.

For example:

<table>
<thead>
<tr>
<th>Tag</th>
<th>Room</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>389</td>
<td>1</td>
<td>-37</td>
</tr>
<tr>
<td>389</td>
<td>7</td>
<td>-55</td>
</tr>
<tr>
<td>412</td>
<td>6</td>
<td>-34</td>
</tr>
<tr>
<td>412</td>
<td>3</td>
<td>-48</td>
</tr>
<tr>
<td>412</td>
<td>4</td>
<td>-55</td>
</tr>
<tr>
<td>412</td>
<td>8</td>
<td>-57</td>
</tr>
</tbody>
</table>

This is presented within the viewing screen displayed in Figure 6: Census Application - manage tab. The user has the ability to select the correct room location manually. When the census is uploaded to the database, the tag(s) indicated in the management tab will be recorded as having been located in whichever room the user selects. If no selection is made, the control software reports the tag as being present in the room with the strongest response signal strength (indicated in green).

4.3.3 PDT File

A PDT file is written containing all the information obtained from the census session. It writes this file by first adding the date and time the application started along with the staff member. Then the program sequentially writes the room number and each tag id associated with that room by first printing the attached date/time field and then the 8 digit identifier. This procedure results in a file that can be read by Granite. Figure 7: Sample PDT file shows a sample PDT file that Granite can read. The user may choose to save the census to a file and upload it to the server at a later time if a connection to the server cannot be established; otherwise the file can be saved and uploaded at the end of the census.
4.3.4 Other Software
The network service to be installed on the client computer only accepts connections from the census application. It simply receives a file and stores it appropriately to be later fed into Granite by a user. Due to the need to associate an RFID tag with a cage card number, a table linking the two items must be inserted into the database. In order to create and maintain this table a check-in/out application is needed. The check-in/out application will allow the user to associate a tag with a cage card number, most likely with a generated file, and then later remove it once an investigator no longer requires the use of a cage. This application will remotely connect to Oracle database using Oracle Database Lite to read and store this table. In addition it would make sense to include the table for associating RFID tags with room numbers at this point, thereby keeping all this data in a centralized location instead of manually distributing it to each RFID reader.

4.4 Software Enhancements
A goal of this semester was to transform the software developed for the proof-of-concept demonstration into a professional product. In order to obtain a professional level, code refactoring, improved error-handling, bug fixes, and additional features were required.

Refactoring was essential to any further development of code. The two major refactoring items included refactoring methods and classes. Methods were broken up into multiple smaller methods with a clear cohesive goal. Void methods were altered to contain only side effects as described by the method name. Non-void methods were stripped of any side effects and only returned the desired result. Classes were divided up to only contain attributes and methods encompassing a single idea. Additionally, methods and attributes were renamed to fit the C# coding standards. By breaking up the classes and methods into cohesive units, the modularity of the program was drastically improved. Now classes were able to be reused for other programs, such as the check-in/out software with no alteration.

The refactoring greatly improved the readability of the code. Readability allows other users to quickly understand the code, so that they can understand, maintain, and add extensions to it if necessary. A side effect of this refactoring also allowed, for easy spotting of bugs, just by reading the code. Many bugs were fixed by just rereading the code after the refactor.

Improved error-handling was implemented by different means. First users were restricted to be able to only input valid inputs. Validating text boxes were added to indicate to the user if an input was valid and decline the input until valid. Try-catch statements were added around any exception throwing calls. The exceptions were generally handled by sending a notification to the user of the error, but allowing the program to continue instead of crashing.
Additional extra features were also implemented to increase usability and extensibility. A record of uploaded PDT files was created to remember any files that had been uploaded. The files are marked as uploaded during the upload selection. This simple extra feature allows the user to see any PDT files that may not have been uploaded. A user selection screen was also added. Originally the user could be selected from a drop down box. However, this proved difficult to use with even a handful of users in the system.

A major feature added was the use of XML files and parsers. Using XML files, configuration settings for the device could be written to persistent memory instead of being hard coded. Configuration files allow for the device settings to be set permanently instead of resetting every time the application software is restarted.

```
<?xml version="1.0" encoding="utf-8"?>
<settings>
  <type>
    <last>0</last>
    <metal>
      <power>22</power>
      <filter>-50</filter>
    </metal>
    <plastic>
      <power>18</power>
      <filter>-39</filter>
    </plastic>
  </type>
  <server>
    <IP>192.168.1.12</IP>
    <port>4000</port>
  </server>
  <deviceID>000000000000000007A6C2</deviceID>
</settings>
```

*Figure 8: Configuration File*

The use of XML also allows for additional extensibility of the program. As new staff members are hired or quit, an xml file can be synchronized with the staff table of the Granite database. The XML file simply contains XML elements for the user’s id as pulled from the database as well as the user’s first and last name. An example is shown in Figure 9: User Profile.

```
<staff id="1">
  <first>Justin</first>
  <last>Fritzler</last>
</staff>
```

*Figure 9: User Profile*
The same configuration can be used to support room tags as shown in Figure 10. Room tags may need to be periodically replaced due to damage or loss. The addition of new real-estate, or change in the current use of rooms would also require a change to the XML file. A feature has been implemented to replace or add new room tags. The feature is run from the device, since the new RFID tag must be scanned. Therefore, a “dirty” attribute has been added so the synchronization process knows that the entry must be updated in the database.

5 Tag Filter

5.1 Filter Purpose
Due to the nature of RFID tags to read through walls, a filter was needed to limit the read range. Even though software can place tags in the correct room with 100% accuracy, the software can not identify tags that are never read. To understand why tags may not be read when tags are inadvertently read in an adjacent room, we must first understand how a user determines if all tags are ready. A user will know the number of cages in any given room. Using this information the user knows how many tags should be read in a room. The number of tags read is displayed in large font on the main screen of the census application. If the read count displayed on the device is greater than or equal to the number of tags in the given room due to inadvertent reads from the adjacent room, the user no longer has an indication as to whether or not all of the tags have been read.

For example: the user accidently misses a tag in the current room, but gets a stray read from an adjacent room. The user completes scanning the room and checks the indicated count with the known count. The user believes that all the tags in the room have been accounted for when in reality, the user is missing one tag. It won’t be until the user completes the adjacent room and checks for duplicate tag reads, will the user know that a tag has been missed.

5.2 Original Filter Design
In order to understand how the filter should be designed a program had to be created to characterize the read performance of various tags attached to either metal or plastic cage card holders at varying power levels. This program recorded the min, max, mean, median, and standard deviation for the tag count and Max RSSI of test read sessions. 120 trial runs were performed using this software to collect sufficient data characterizing the tags. Tag count proved to be completely dependent on the user and was thrown out as a filtering option.
The minimum Max RSSI values fell nicely within 1 standard deviation of the median. However, in practice this proved to be an ideal, rather than practical result. Instead, a threshold of 2 standard deviations proved to be more practical. The standard deviation turned out to be about 5 for all tested power levels for both metal and plastic cage card holders. At a power level of 15 dBm for plastic cage card holders, the median Max RSSI value was -35 dBm. Therefore, a filter threshold was placed 2 standard deviations away at -45 dBm. For metal cage card holders at a power level of 20 dBm, the median Max RSSI was -48 dBm, hence a filter threshold of -58 dBm.

The original filter design was never used extensively in practice. However, by performing additional proof-of-concept trials, it was clear that some tweaking could be done. The use a dynamic filter, based on the current read session was also considered for implementation. The dynamic filter would record RSSI statistics while the user was reading a room. When the user exits a room, all tags with a Max RSSI 3 standard deviations below the median would be marked as possibly being in another room. Once the census is complete, tags marked will be added to the room originally found in, if they do not exist in another room. This idea proved to be complicated and prone to error. Instead a simpler solution had to be devised.

5.3 Improved Filter Design

In order to create a more precise filter, experiments were run profiling the effects of power and distance on the RSSI. By performing a read at every power interval at a fixed distance a profile could be created showing the impact of power on the RSSI. 3 trials were run at a 6” distance for 5 second intervals at every power interval in the range 5 to 30 dBm on both metal* and plastic cage card holders. (*tags placed in metal cage card holders are affixed to rubber insulation.) The produced profile is shown in Figure 11.

At first glance the power profile does not provide much useful information. In fact it gives no real indication as to what power should be selected. The RSSI is close to constant with the exception that it slightly tapers down at lower power levels and that a minimum power is required to read tags in metal cage card holders. What can be derived from this profile is that the RSSI is higher for tags placed in plastic cage card holders than those placed in metal holders. This information was already known from the trial runs performed last semester.
In order to determine a suitable power level a different approach had to be taken. From the same trial runs used to create the power profile, data on the number of tag reads was also collected. Creating a profile of power with respect to tag reads over a 5 second interval produced an interesting result.

Read performance can be defined as the number of tag reads over a given time interval. Figure 12 shows the new result, now defined as the performance profile. From this graph it is easy to see that performance does not improve after a given power level. Therefore, we can pick a minimal power level for maximum performance.

A power level of 18 dBm for plastic holders and 22 dBm for metal holders has been chosen for the continuation of this filter design. Now a distance profile needs to be made for plastic and metal holders respectively at the chosen power levels. The distance profile created in Figure 13 was created in a similar manner to the previous profiles. This time distance was varied from 2 to 18 inches for the chosen power levels. This time the Max RSSI was recorded. Therefore, we can determine a cutoff point for a desired distance. For this project the desired cutoff distance is 12 inches. Hence, a filter threshold of -39dBm was chosen for plastic holders and a filter threshold of -50dBm for metal holders. The new values are a vast difference from the original power and filter levels as determined in section 5.2.

6 Conclusion
The project has been very successful in terms of the work completed so far. More work is desired; however, this work does not fit a senior design project well. The desired skillset for the continuation of this project better fits a CS or CIS major. Continued maintenance will also be required, but this would not be enough to fill a senior design project. However, it is now time
to turn management of this project back over to LAR. The project also came in significantly below budget; see Appendix B for the budget. The project has been so successful, that LAR is looking at offering this solution to other Universities.

7 Appendix A: Abbreviations

CSU – Colorado State University
LAR – Laboratory Animal Resources
PDT – File extension associated with census files read by Granite
PI – Private Investigator (Client performing research on lab animals)
RFID – Radio Frequency Identification
RSSI – Received Signal Strength indication

8 Appendix B: Budget [1]

All funding was provided by Laboratory Animal Resources. A $100 allowance from the CSU ECE department to use for miscellaneous items was available as well. Proof of Concept (delivered in May 2013) had a $5000 limit.

<table>
<thead>
<tr>
<th>Item</th>
<th>Item Cost</th>
<th>Quantity</th>
<th>Total Estimated Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFID Reader</td>
<td>$3,000.00</td>
<td>1</td>
<td>$3,000.00</td>
</tr>
<tr>
<td>RFID Tag for Testing</td>
<td>$100.00</td>
<td>1</td>
<td>$100.00</td>
</tr>
<tr>
<td>RFID Tags for Proof</td>
<td>$240.00</td>
<td>1</td>
<td>$240.00</td>
</tr>
<tr>
<td>Testing Materials (misc)</td>
<td>$16.00</td>
<td>1</td>
<td>$16.00</td>
</tr>
<tr>
<td>E-Days Materials</td>
<td>$25.00</td>
<td>1</td>
<td>$25.00</td>
</tr>
</tbody>
</table>

**Final Cost** | $3,381.00

Figure 14: Cost - Proof of Concept
Appendix C: Project Timeline

Jun 2012 - Dec 2012
Planning

Jun 2012 - May 2013
Hardware Development

Jan 2013 - Dec 2013
Software Development

Jun 2013 - May 2014
Operation and Maintenance

June 2012

May 2014

12/13
Written Report

12/11
Oral Presentation

Dec 2013 (End of semester)

Aug 26 (Start of semester)

8/26 - 9/5
Semester planning

9/2 - 11/7
RFID census application development

9/5 - 11/20
Census application testing

11/7 - 11/30
User manual

11/14 - 12/20
Database synchronization software

11/21 - 12/20
Synchronization testing

11/25 - 12/9
Presentation prep

11/25 - 12/13
Written report

12/9 - 12/20
Turnover

9/18
Project Website

10/26
Testing Plan

August 2013

20
10 Appendix D: Bibliography


11 Acknowledgments

I would like to the individuals that have helped support this project:

Dr. Lon Kendall for coordinating with the ECE department to create this senior design project, and his feedback throughout the development of this project.

Dr. Ali Pezeshki for advising this project and his additional guidance.

All the staff at Lab Animal Resources for providing user feedback, and support throughout the whole development process.

Tyler Wilson for Granite software and database support.

Olivera Notaros for leading the Senior Design Program at Colorado State University.