

CEng 490

DOOR SECURITY PROJECT

by C4-CORP

DETAILED DESIGN REPORT

1190545 Özkan Kılıç
1248715 Kerim Şahin
1255603 Okan Bozkurt
1255637 M.Bilal Demirkan

Table of Contents

EXECUTIVE SUMMARY	3
1 INTRODUCTION	3
2 OBJECTIVES	6
3 DESIGN METHODOLOGY	7
4 PROJECT SCHEDULING	12
5 IMPLEMENTATION SCHEDULING	15
6 DESCRIPTIONS AND DIAGRAMS OF SYSTEM MODULES	21
6.1 HARDWARE:	21
6.1.1 <i>Card Design:</i>	21
6.1.1.1 C4 Control Card	21
6.1.1.2 C4 Control Card Specifications	22
6.1.1.3 RS-232 Standard	22
6.1.1.4 Internal Structure (Operating Principles) of C4 Control Card	23
6.1.1.5 Process and Data Flow Charts	26
6.1.2 <i>Connections:</i>	29
6.1.2.1 Main Board, Control Program, Wireless Connection	29
6.1.2.2 Wiegand Compatible Proximity Reader	30
6.1.2.3 The 26 bit Standard Wiegand Output	31
6.2 SOFTWARE:	31
6.2.1 <i>Operating System:</i>	32
6.2.2 <i>Database:</i>	32
6.2.3 <i>Administrator Program:</i>	36
6.2.3.1 User Management Sub-Module	37
6.2.3.2 Card Management Sub-Module	38
6.2.3.3 Door Management Sub-Module	39
6.2.3.4 Access Management Sub-Module	39
6.2.3.5 Reporting Sub-Module	40
6.2.3.6 Monitoring Sub-Module	41
6.2.4 <i>Main Board Program:</i>	46
7 FORMAL SPECIFICATION OF OUR SYSTEM SOLUTION	47
8 SYNTAX SPECIFICATION OF OUR SYSTEM SOLUTION	48
APPENDIX: DATA DICTIONARY	49

Table of Figures

FIGURE 1 – USE CASE DIAGRAM	9
FIGURE 2 – ACTIVITY DIAGRAM OF GRANTING AN ACCESS.....	10
FIGURE 3 – CLASS DIAGRAM OF THE SYSTEM	11
FIGURE 4 – PROJECT SCHEDULE, GANTT CHART PART I	13
FIGURE 5 – PROJECT SCHEDULE, GANTT CHART PART 2.....	14
FIGURE 6 – WIRE CONFIGURATION OF TWO-WAY (FULL DUPLEX) COMMUNICATIONS	23
FIGURE 7 - FLOW OF COMMUNICATION.....	23
FIGURE 8 – COMBINING TWO DATA LINES INTO ONE	24
FIGURE 9– SOURCES OF DATA	24
FIGURE 10 – FLOW CHART FOR MAIN BOARD	26
FIGURE 11 - LISTEN C4 CONTROL CARD	27
FIGURE 12 - LISTEN SERVER	27
FIGURE 13 - FLOW CHART FOR MAIN BOARD (RESPOND TO C4 CONTROL CARD PART).....	28
FIGURE 14 – DATA FORMAT	29
FIGURE 15 – MESSAGE TYPE CODES	30
FIGURE 16 – DATA FORMAT (DETAILED)	30
FIGURE 17 – PROXIMITY READER CONNECTIONS	31
FIGURE 18 – 26 BIT STANDARD WIEGAND OUTPUT.....	31
FIGURE 19 – ENTITY RELATIONSHIP DIAGRAM FOR SERVER.....	35
FIGURE 20 - ENTITY RELATIONSHIP DIAGRAM FOR MAIN BOARD	36
FIGURE 21 - STRUCTURE OF ADMINISTRATOR PROGRAM	37
FIGURE 22 - USER MANAGEMENT	38
FIGURE 23 - CARD MANAGEMENT	38
FIGURE 24 - DOOR MANAGEMENT	39
FIGURE 25 - ACCESS MANAGEMENT	40
FIGURE 26 – SCHEDULING	40
FIGURE 27 – REPORTING.....	41
FIGURE 28 - MONITORING.....	41
FIGURE 29 – LOGIN SCREEN.....	42
FIGURE 30 – MAIN SCREEN.....	42
FIGURE 31– CARD INSERTION SCREEN	43
FIGURE 32 – DOOR INSERTION SCREEN.....	44
FIGURE 33 – SCHEDULE INSERTION SCREEN.....	44
FIGURE 34 – PARAMETER INSERTION AND UPDATE SCREEN	45

Detailed Design Report

C4-CORP

Project members: 1190545 Özkan Kılıç
1248715 Kerim Şahin
1255603 Okan Bozkurt
1255637 Mustafa Bilal Demirkan

Executive Summary

Detailed Design Report is written for the project that is the production of a card based wireless door security system. It is the final phase of the design to make implementation and testing steps clearer. First the definition of the project and the goals of the report are specified. Then, design methodology is given in an explanatory way. The detailed designs of modules (card design, connections, database, administrator program, main board program, protocols, and such.) are explained in detailed ways with drawings if necessary. Databases, data dictionaries, UML diagrams, administrator program with maps and with possible screenshots are specified. Moreover, the formal specification of our system solution and syntax specifications are included throughout the detailed design report. Finally, project implementation scheduling with milestones is included, as well.

1 Introduction

The detailed design of the project is the backbone for the implementation phase. After analysis and system specification phase, we prepared an initial design report. The initial design report was prepared as a draft of the final design because design is very vital and it is difficult to prepare a good design at once. Therefore, throughout the detailed design, we will give our final designs, revision of the initial design, project implementation schedule and such.

In this report, we will mention about our detailed design process and focus on the work products- diagrams, and explanations- we obtained after realization of the static and dynamic aspects of our initial design.

Our project is software and hardware for establishment of a card based wireless security door control system. The project involves establishing a wireless secure network connection between the designed controllers and the master computer. Moreover, the master computer software with access control of the doors, card management, access group management, event and status reporting of the doors, and lock control are designed and will be implemented.

In case of multiple doors, accesses to the rooms will be determined by a master computer and all the information will be reported to the master computer by controllers. The communication between main computer and controllers will be based on wireless communication according to some protocols to improve security. Indeed, there will be software running on the master computer for card management, group management, control of the doors, locks, and reporting events and logs.

Hardware Specifications:

- VIA ITX Main Board (with 2xUSB + 1xSerial + 1xPCI + 1xParallel port)
- Power supply for Main Board
- 128MB-memory
- 128MB-Disk space
- Wi-Fi Ethernet Card
- 2 Wiegand Proximity Readers
- Door Status Sensor
- Lock
- C4 Control Card
- UPS

We formed the design way of the project with respect to the analysis period we completed beforehand and with respect to the reviews received from initial design. Unity of the analysis and design is very important because being stick to their interaction will decrease the failures and will lessen the probability of returning back and re-doing all the steps.

We received some feedback on our initial design. We are informed about strengths and weaknesses in the design. Therefore, in the detailed design, we reviewed the initial one and worked on our missing parts to make everything ready for implementation.

We designed the hardware and software modules of the project. Card system, database, protocol and communication details, administrator program and flow of this program are specified. The specifications of

reporting mechanism in administrator program and details of reports that can be gathered by means of that mechanism are given. We translated the requirements into an initial blueprint and they are reviewed in this design for the project implementation. Implementation schedule is given in this report.

2 Objectives

We perform our detailed design throughout the application of fundamental design principles, systematic methodologies, reviews of initial design and Unified Modeling Language. Therefore, our main objectives in this process to propose a detailed design are:

- Determine the final design basis for the implementation, after accurately transferring users' requirements into an initial draft of the software and hardware product of system.
- Determine the design specifications of the hardware mechanisms of the project so that we can easily integrate them into software part later.
- Specify the considerations of the software part – database and administration, main board programs- of the project.
- Define the appropriate design methodology so that the project can be implemented successfully.
- Outline for the implementation of all the explicit requirements of the analysis model, and accommodating all the requirements desired by customer.
- Prepare a readable and understandable guide for generating implementation and testing.
- Providing a complete picture of the project, addressing data, functional, and behavioral domains for an implementation perspective.

Through the detailed design report, we aimed to meet these objectives in order to establish a basement for later implementation and consequently set a plan for the implementation phase.

3 Design Methodology

In order to have a successful project, each phase must be planned, too. Design methodology should be appropriate to the project to carry out it in a determined way.

Since we have an integration of the hardware and software in the project, which grows up gradually, we chose the spiral model. By means of this model, we designed our system in a step-by-step manner. First we designed a primitive database for a single door to test data keeping and to control protocols. Then we gradually designed and extended the system so that we eventually reach to a large composite system.

Our main interest area in the detailed design phase of the project is to determine how to visualize, specify, construct and document the artifacts of the document flow mechanism aimed to be setup between hardware and software. We decided to make our design using methods defined in an object-oriented way. To accomplish this, we used modular approach.

We divided the whole system into modules:

1. Hardware
 - Card design
 - Connections
2. Software
 - Database and
 - Administrator program
 - Main Board Program

Then we gradually designed their combinations to see the whole picture.

We determined the operative objects covering the specific modules in specific components of our system. Then, we set up communication mechanisms co-operating between these modules in the system not only as software and also as hardware. This corresponds to the collaboration diagrams we presented below.

The combination of whole system should be designed so that the resulting program's flexibility and ease-of-usability will be efficient. Unnecessary operations and redundant storage space allocation can be eliminated by the effective design of the classes and coordination between hardware and software.

The design of the modules should contribute to the easy implementation of the use cases, as well.

The designing the whole system with respect to modules is our main methodology. It not only makes our labor division easier but also gives us the step-by-step picture of the main picture. Moreover, object oriented modular design is also easy to manage and implement. The detailed design specifications of each module are given later in this report in an explanatory way.

In order to visualize the whole picture of our system solution, following UML Diagrams will be explanatory:

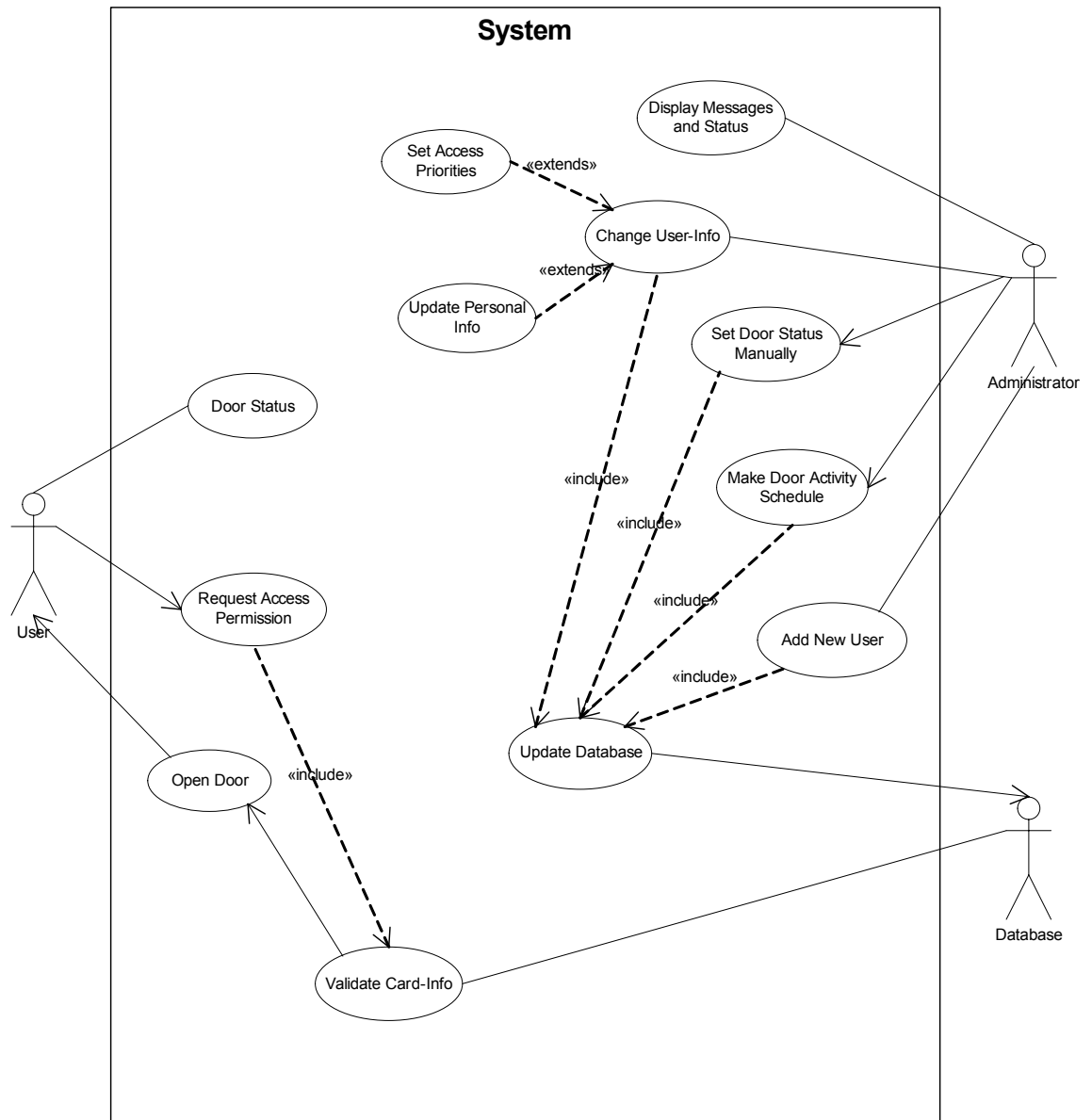


Figure 1 – Use Case Diagram

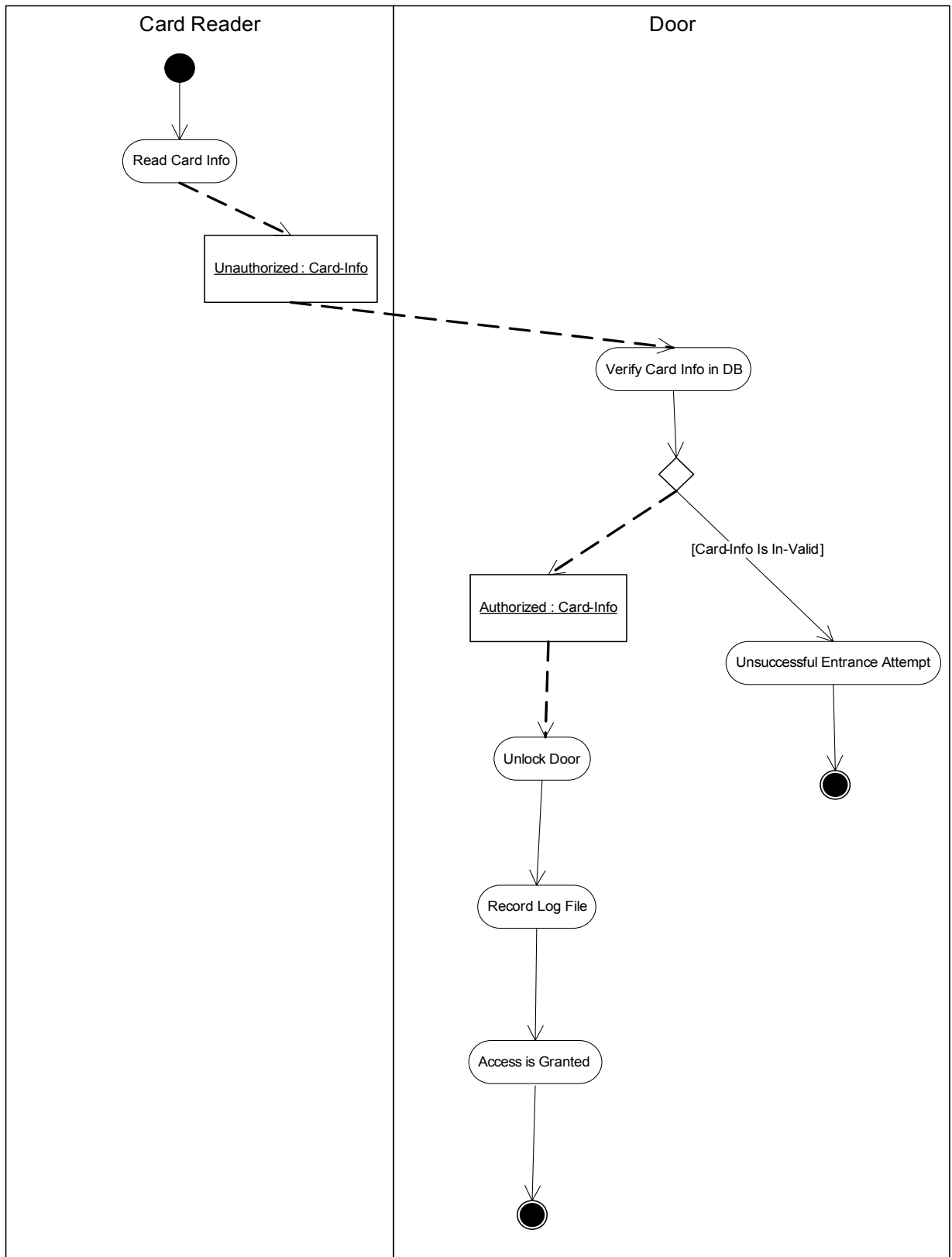


Figure 2 – Activity Diagram of Granting an Access

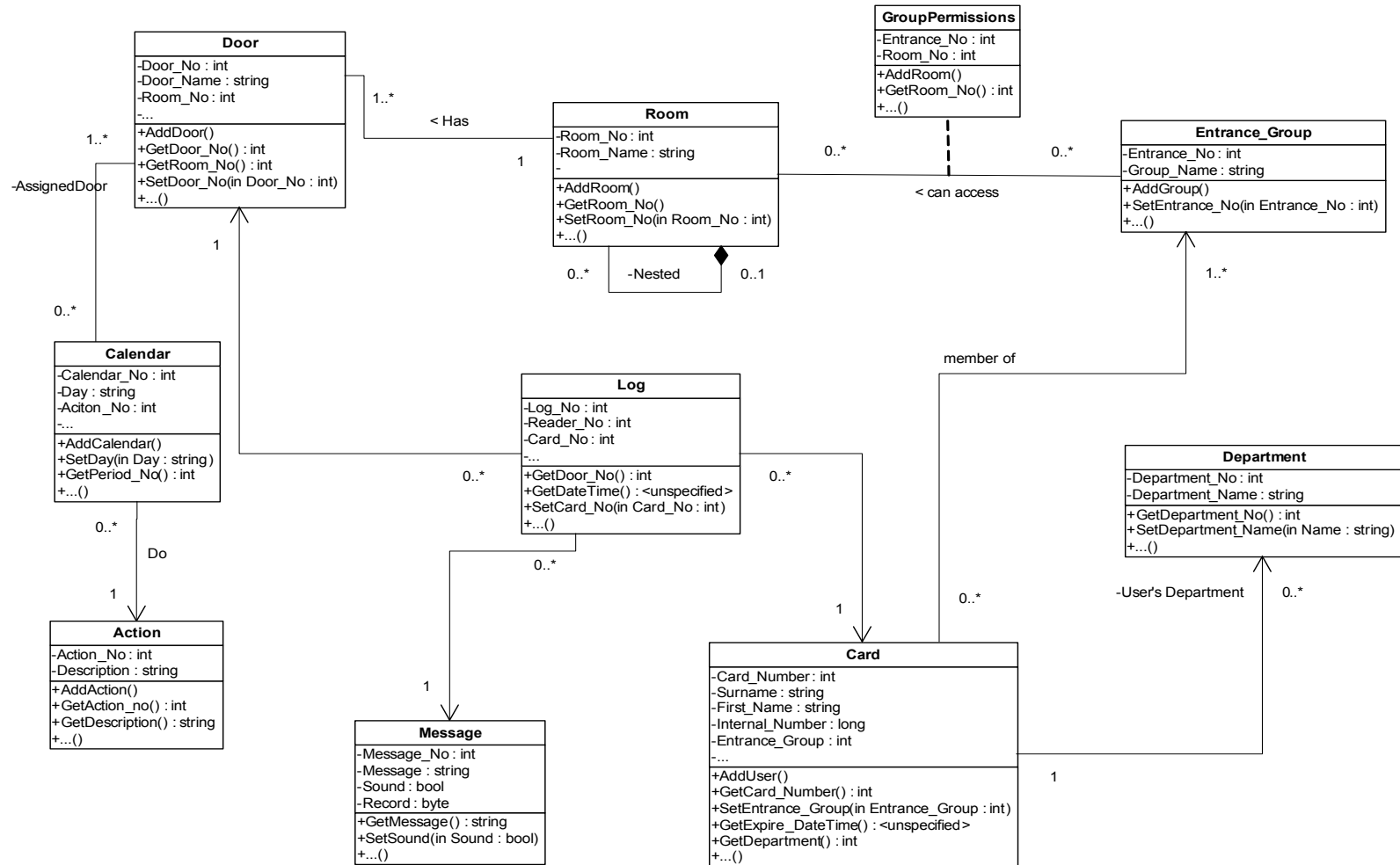


Figure 3 – Class Diagram of the System

4 Project Scheduling

We prepared our project scheduling. Having a timeline and schedule is important because the overall task gets complicated more and more while we design and implement modules by the time. Thus, outline and being stick to outline will aid us to have less risky and more productive project. Project schedules can be seen visually as Gantt chart representation in Figure 4 and Figure 5.

Apart from these, project meetings reports are given. Report dates are not regular because of holidays or their intersections with other reports' dates.

1st Meeting Report: 12 October 2004

2nd Meeting Report: 19 October 2004

Team Dinner: 22 October 2004

3rd Meeting Report: 27 October 2004

4th Meeting Report: 2 November 2004

5th Meeting Report: 20 November 2004

6th Meeting Report: 20 December 2004

And daily meetings are held on MSN Messenger.

Gantt Chart			Task Name	Duration	Start	Finish	Predecessors
	1		- Planning	12 days	Tue 28.09.04	Wed 13.10.04	
	2		Project Proposal	11 days	Tue 28.09.04	Tue 12.10.04	
	3		Milestone	0 days	Wed 13.10.04	Wed 13.10.04	
	4		- Requirements Analysis	32 days	Thu 14.10.04	Wed 24.11.04	3
	5		Specification Research	5 days	Thu 14.10.04	Tue 19.10.04	
	6		Web Questionnaire	29 days	Thu 14.10.04	Fri 19.11.04	
	7		Requirements Specification	18 days	Mon 01.11.04	Tue 23.11.04	
	8		Milestone	0 days	Wed 24.11.04	Wed 24.11.04	
	9		- Initial Design	10 days	Thu 25.11.04	Wed 08.12.04	8
	10		Design Drafts for Modules	4 days	Thu 25.11.04	Tue 30.11.04	
	11		Initial Design Specification	7 days	Mon 29.11.04	Tue 07.12.04	
	12		Milestone	0 days	Wed 08.12.04	Wed 08.12.04	
	13		- Detailed Design	24 days	Thu 09.12.04	Mon 10.01.05	12
	14		Draft Design Specification	6 days	Thu 09.12.04	Thu 16.12.04	
	15		Object Oriented Architectural & Procedural Design	13 days	Fri 10.12.04	Mon 27.12.04	
	16		Complete Class Diagrams	10 days	Wed 22.12.04	Mon 03.01.05	
	17		Hardware Design	22 days	Fri 10.12.04	Fri 07.01.05	
	18		Interface Design	5 days	Fri 31.12.04	Thu 06.01.05	
	19		Test Provisions	19 days	Thu 16.12.04	Mon 10.01.05	
	20		Design Specification	6 days	Mon 03.01.05	Mon 10.01.05	
	21		Milestone	0 days	Mon 10.01.05	Mon 10.01.05	
	22		- Pre - Implementation	23 days?	Mon 20.12.04	Wed 19.01.05	12
	23		Prototype	23 days?	Mon 20.12.04	Tue 18.01.05	
	24		Milestone	0 days	Wed 19.01.05	Wed 19.01.05	
	25		- Implementation	92 days?	Tue 01.02.05	Wed 08.06.05	24
	26		Prototype review	5 days?	Tue 01.02.05	Mon 07.02.05	
	27		Implementation Initials	8 days?	Tue 01.02.05	Thu 10.02.05	
	28		- System Database Modules	5 days?	Tue 01.02.05	Mon 07.02.05	
	29		DB Tables	4 days?	Tue 01.02.05	Fri 04.02.05	
	30		DB Queries	5 days?	Tue 01.02.05	Mon 07.02.05	
	31		- Module	58 days?	Tue 08.02.05	Thu 28.04.05	
	32		Wireless Communication Modules	24 days?	Tue 08.02.05	Fri 11.03.05	
	33		Card Implementation	27 days?	Thu 10.02.05	Fri 18.03.05	
	34		Main-Board Program	52 days	Tue 15.02.05	Wed 27.04.05	
	35		GUI Implementation	22 days?	Fri 04.03.05	Mon 04.04.05	
	36		Administrator Program	40 days?	Fri 04.03.05	Thu 28.04.05	
	37		First development shapshot-Milestone	0 days	Mon 14.03.05	Mon 14.03.05	
	38		Performance Improvement	15 days?	Mon 25.04.05	Fri 13.05.05	
	39		Test Reports	12 days?	Mon 02.05.05	Tue 17.05.05	
	40		First version Release - Milestone	0 days	Mon 02.05.05	Mon 02.05.05	
	41		Implementation (final)	8 days?	Thu 19.05.05	Mon 30.05.05	
	42		Documentation	92 days?	Tue 01.02.05	Wed 08.06.05	
	43		Milestone	0 days	Wed 08.06.05	Wed 08.06.05	
	44		- Customer Evaluation	2 days?	Thu 09.06.05	Sat 11.06.05	43
	45		Feedback	2 days?	Thu 09.06.05	Fri 10.06.05	
	46		Milestone	0 days	Sat 11.06.05	Sat 11.06.05	

Figure 4 – Project Schedule, Gantt chart Part I

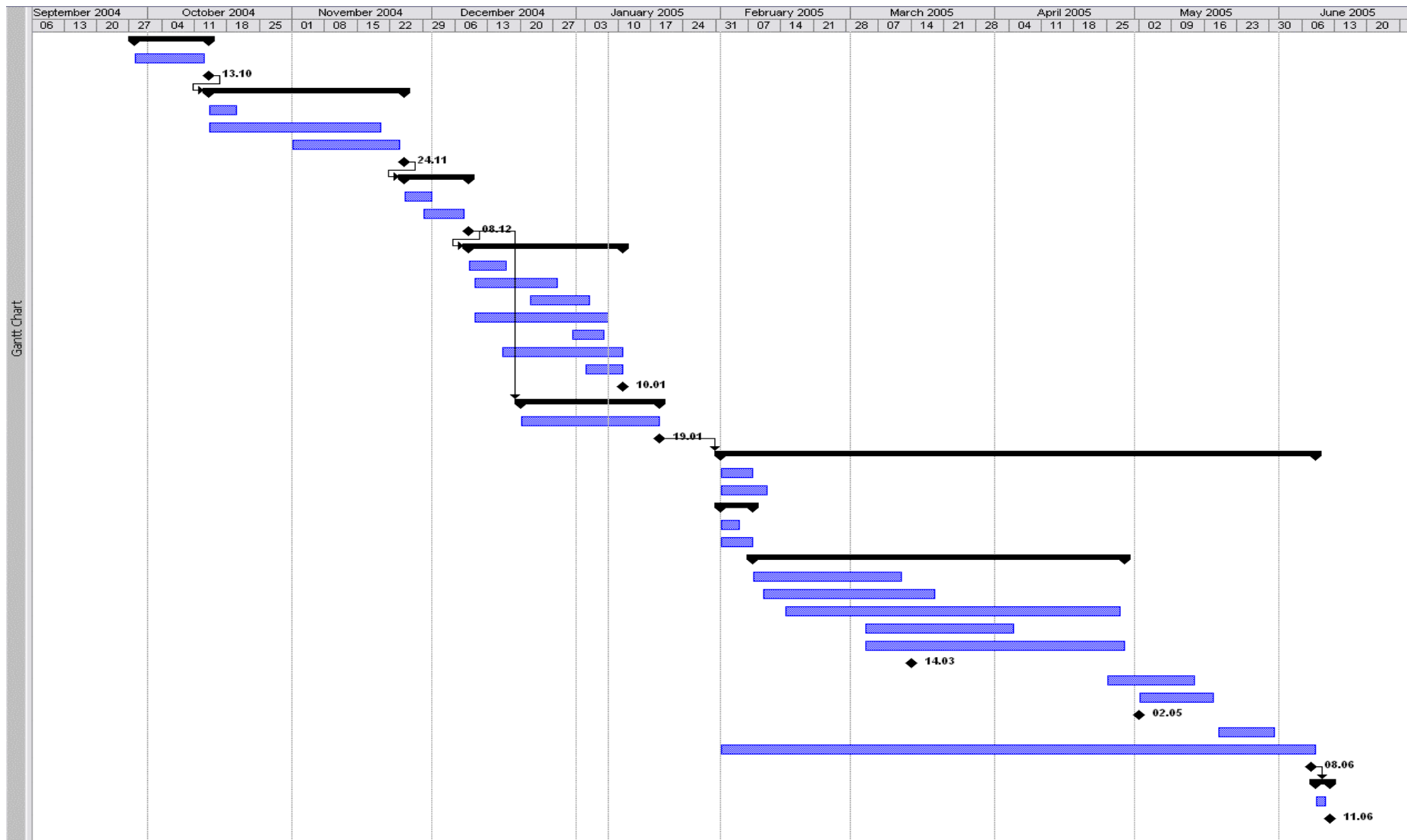


Figure 5 – Project Schedule, Gantt chart Part 2

5 Implementation Scheduling

Since the whole system is designed in a detailed way, we are ready to offer an optimum implementation schedule. The implementation schedule is given in Gantt chart. The actions to be completed in implementation are as follow:

- **Pre-implementation (estimated time:19.01.05):**

Pre-implementation is actually a discovery for us to get to know hardware, and to make us determine a path to implement the initials of the project. We have already started for pre-implementations. We have experienced failures and successes. Furthermore, pre-implementation is important because it enabled us to see the difference between analysis-design and implementation. Since we had limited knowledge in hardware, this phase helped us to replenish what we know and learn more.

- **Prototype (estimated time:18.01.05):**

Prototyping is the premature version of the program. It is an inevitable part of the spiral development because it helps us to test our design ideas, to decrease some risks, and to understand what the deficiencies are. For prototyping, we have been implementing some key components of our solution, some visualization and some simple integration. Our control card design and implementation is one of the most challenging parts of the project. We will not implement it in this prototype but we will simulate, instead. We will represent some data communication throughout RS232 and boards. This communication will be simulated in and responded by the prototype of administrator program. The prototype of administrator program will respond to this pseudo data as if it were sent by the door hardware. Furthermore, prototype of administrator program will send some commands, like lock or unlock, as if there were door hardware communicating via RS232.

- **Implementation (estimated time:08.06.05):**

In the following semester, we will implement our system by means of the design we have been making. Therefore, it is important to have a successful design. Prototype will lead us in further implementation considerations. For the implementation, prototype reviews, the results of these reviews, modules, final reviews, performance improvements, tests and documentations will be done dependently.

- **Prototype Reviews (estimated time:07.02.05):**

After the presentation of prototype, we will review it to see how successful we can transfer our design keys into implementation. The reviews of prototype will help us to see our weak and strong points because it will be the first draft of the implementation, like initial switching from the abstract level to the concrete level.

- **Implementation Initials (estimated time:10.02.05):**

The reviews following the prototype will set our initials, the concepts we must consider throughout the project. Actually, the prior jobs have already shown us some keys to consider; for example, we have learnt how important is to have discipline, regular working, schedule, and communication. By means of prototype reviews, we expect to have more concrete initials, such as, some hardware usage principles, integration concerns, and implementation priorities. After setting the initials, we will have more concrete basis to stick to while moving into details of implementation.

- **System Database Modules (estimated time:07.02.05):**

We will overlap implementations of modules in order to make the implementation faster. Thanks to modular design, database modules were designed initially and we can implement it separately. We will be able to combine the modules because of well design.

- **DB Tables (estimated time:04.02.05):**

In the appendix section and in ER diagrams given in section 6.2.2 Database, we specified database tables. We considered database design carefully so that we will implement it successfully.

- **DB Queries (estimated time:07.02.05):**

There are two sections where the queries will be in charge: One separate section in reporting module of the administrator program and the other one in query builders of each module. In the administrator program, it is important to have effective queries in order to monitor the system. Monitoring can be done throughout the comprehensive observation of logs. We designed the reporting system in such a detailed manner that the administrator can get various kinds of effective queries from the logs. In the query builders of each module, the builder will be specialized with regard to the module it belongs to. For example, in “User Management Module”, the query builder will operate on data related to users only.

- **Modules (estimated time:28.04.05):**

Apart from database module, we will implement the remaining closely related modules, review, test, and combine them consequently. We planned to make improvements on prototype, to evolve the prototype step by step to get eventually the whole system.

- **Wireless Communication Modules (estimated time:11.03.05):**

In this module, we will implement client-server communication specifications. This module will serve to the system for data transfer by means of wireless network. When the connection is set, the

peers will listen to each other in order to get commands, tables, actions and such.

- **Card Implementation (estimated time: 18.03.05):**

As we specified all the details of the C4 control card, we will implement this hardware with respect to the specifications given above. This implementation seems to be the one of the most difficult modules because of our being inexperienced.

- **Administrator Program (estimated time: 28.04.05):**

This module will be the main software running the system. The capabilities, specifications and functions of administrator program are given above in Administrator Program Design Module. Administrator program will run in Linux environment.

- **GUI Implementation (estimated time: 24.04.05):**

Our administrator program will be web-based and it will be programmed by PHP. We will prepare a user friendly Graphical User Interface implementation of it. This will help administrators to manage the system more easily.

- **Main Board Program (estimated time: 27.04.05):**

Main Board Program will be communicating with administrator program by means of wireless communication and it will be commanding C4 Control Card to order what to do. This program will keep its own door's database and ask / give updates from Administrator Program if necessary. It will run in Linux environment.

- **Performance Improvement (estimated time:13.05.05):**
The work is not of course finished when the implementation is finalized. The following issues are performance, testing, reviews, documentation and customer feedback. In the performance improvement stage, the reliability and speed concerns come into view. After the implementation, we will work on performance improvements.
- **Test Reports (estimated time:17.05.05):**
Test reports will be prepared for the system. Crisis cases, working specifications and such will shape our way of test. Test reports will give us a plan to create final version of the implementation.
- **Final Implementation (estimated time:30.05.05):**
In Initial Phases, the main concern is whether we will be able to combine the modules and have a working product. Yet, in final implementation, the aim is to have a final product with improved performance. The final implementation is the whole working system and ready to be handed to the customers.
- **Documentation (estimated time:08.06.05):**
Documentation is very important and it is one of the main deficiencies that engineers usually lack. In the design and analysis phases, we unfortunately suffer the consequences of poor documentation. Now, we are aware of its importance. Since we have been keeping track of everything and project versions, it will be easy for us to prepare documentations for customer and other technicians. It will include technical directions for further improvements.
- **Customer Evaluation and Feedback (estimated time: 11.06.05):**
This stage is rewarding for our hard work and sacrifices. It will show us how successful our project is. The feedback

gathered from the user will lead us in our further projects or in professional carriers.

6 Descriptions and Diagrams of System Modules

6.1 Hardware:

Our project extensively involves integration of hardware mechanisms as we specified above. Here in this module, we prepared initial design of the tools. We specified hardware as two sub-modules, Card design and connections. Their designs and responsibilities are as follow:

6.1.1 Card Design:

6.1.1.1 C4 Control Card

In this project, Wiegand Proximity Readers that use 26 bit Standard Wiegand Protocol to communicate outer world will be used. However, this protocol cannot be recognized directly by any computer. To provide an interface for communication between the proximity readers and main board (computer), we design interface hardware that we named C4 Control Card. Furthermore, an electronically controlled lock and door status (open or close) sensor will exist on the door. C4 Control Card that we will develop should also control them.

In addition, the controller provides power to proximity readers, so the readers are powered on when the controller is powered on. The reader's normal state is to display constantly on amber led as it waits for a card to be presented. When a card passed within a few centimeters of the reader, the reader will beep and either the green or red led will flash (depending upon whether or not the card has been enrolled at the controller, in fact, controller requests a check for card's enrollment information from the main board) and then return to color of led to steady amber.

As we stated later in this section, Wiegand output cannot be recognized directly by a computer. C4 Control Card is mainly designed for that reason. However, that time a new problem occurs. How C4 Control Card can communicate with a computer? We cope with this problem by choosing RS-232 standard in order to establish a connection between C4 Control Card and main board.

6.1.1.2 C4 Control Card Specifications

Input Devices

- ➔ Two proximity readers (26 bit Standard Wiegand Protocol)
- ➔ Door status sensor
- ➔ Main board (Serial port, RS-232 standard)

Output Devices

- ⬅ Door lock
- ⬅ Two proximity readers (26 bit Standard Wiegand Protocol)
- ⬅ Main board (Serial port, RS-232 standard)

6.1.1.3 RS-232 Standard

The serial port on a main board is a full-duplex device meaning that it can send and receive data at the same time. In order to be able to do this, it uses separate lines for transmitting and receiving data. The TD (transmit data) wire is the one through which data from a DTE (Data Terminal Equipment) device is transmitted to a DCE (Data Communications Equipment) device. The TD line is kept in a mark condition by the DTE device when it is idle. The RD (receive data) wire is the one on which data is received by a DTE device, and the DCE device keeps this line in a mark condition when idle. Other wires are used for handshaking purposes. In fact, two-way (full duplex) communications is possible with only three separate wires - one to send, one to receive, and a common signal ground wire. These three wires can be connected as in Figure 6. DTE is main board and DCE is C4 Control Card in our system. The serial port sends data one bit at a time over one wire. It is also valid for receiving data, that is, the serial port receives data one bit at a time over one wire. Communication starts with a start bit and ends with a stop bit. After the start bit has been sent, the transmitter begins to send actual data bits. After the data has been transmitted, a stop bit is sent. A stop bit has a value of 1 (or a mark state) and it can be detected correctly even if the previous data bit also had a value of 1. This is accomplished by the stop bit's pulse duration. Stop bits can be 1, 1.5, or 2 bit periods in length.

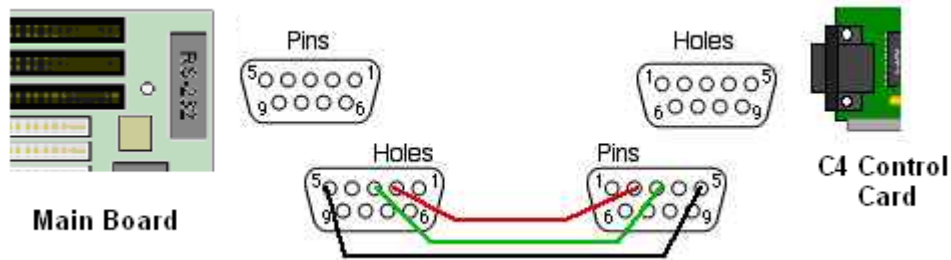


Figure 6 – Wire configuration of two-way (full duplex) communications

6.1.1.4 Internal Structure (Operating Principles) of C4 Control Card

Flow of communication can be seen in Figure 7.

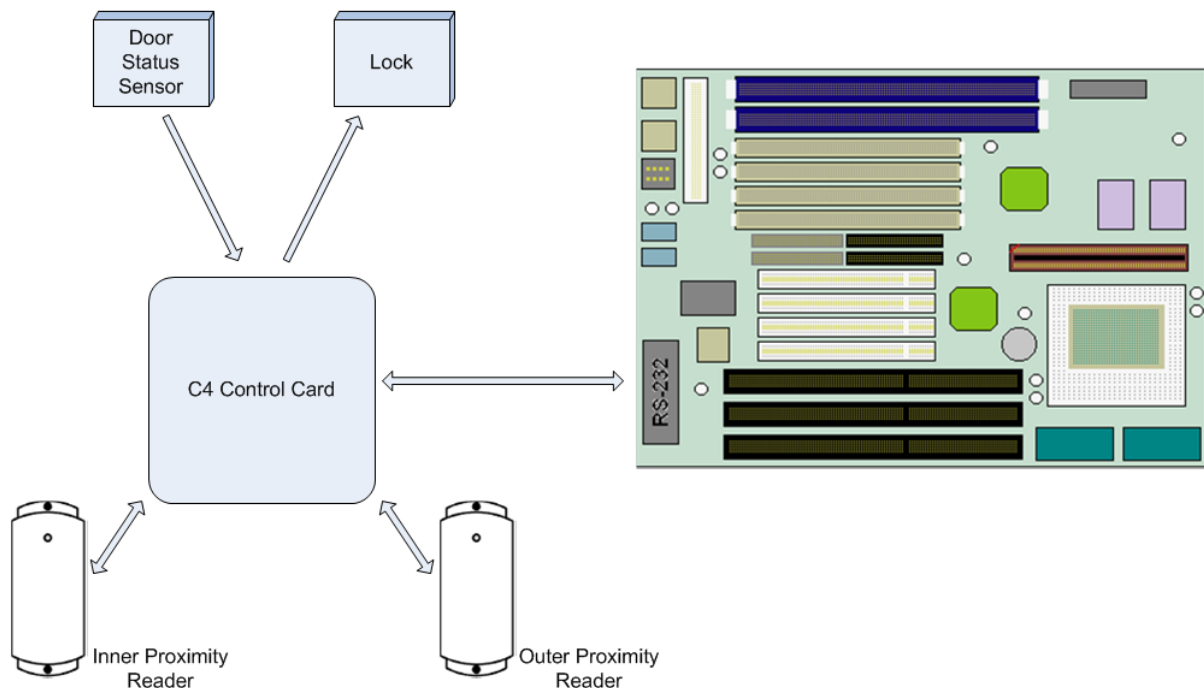


Figure 7 - Flow of communication

Main purpose of C4 Control Card is establishing a connection between Wiegand Proximity Readers and the main board. Wiegand Proximity Readers send (0's) zeros through data0 wire and (1's) ones through data1 wire. It sends 26 bit standard Wiegand data one bit at a time by using the principle mentioned in the preceding sentence. Since C4 Control Card sends data using only one wire on RS-232, we should combine data0 and data1 wires of Wiegand Proximity Reader into one wire by using some electronic component. We can combine these wires

using component called buffer because these data wires are mutually exclusive (Figure 8). In other words, data0 and data1 do not go low at the same time.

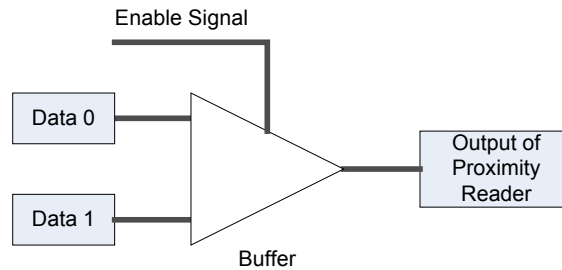


Figure 8 – Combining two data lines into one

Our hardware sends data between a start bit and a stop bit. Before starting to send data, it first sends a binary code (Figure 9), which describes the source of data. The data source can be either first reader, second reader or the door status sensor. However, the data comes to our C4 Control Card on the fly. Therefore, we have to gain some time to send source of data code to main board. It is only possible with using some memory unit in the C4 Control Card to preserve the data that came to card at that time.

Code	Source of Data
00	Main Board
01	Inner Wiegand Proximity Reader
10	Outer Wiegand Proximity Reader
11	Door Status Sensor

Figure 9– Sources of Data

As can be seen in the Figure 9, there is an additional code '00'. It is sent by main board when main board needs to report an error to C4 Control Card.

After sending the binary code of source of data, depending on the situation, C4 Control Card sends either 26 bit card number or door status sensor information.

- i) If C4 Control Card sends 26 bit card number to the main board, the program we implemented that run on the main board checks the access information of that card number on the door that the reader is present at this time. After checking, main board sends some information that consists of source of data code and response data, which is access granted or not, back C4 Control Card. If access is granted, the main board adds “access granted for card holder X” message to the local log file and C4 Control Card unlocks the door and sends a signal to the proximity reader to change the color of its led to green. After sending access granted information, the main board waits some specified time interval and then sends a signal to control card to lock the door. Between enrolled card information sent and lock door signal sent, if the door is opened, the main board adds “door opened normally” message to the local log file. After this message, when the door is closed, door status sensor sends a signal to the main board through C4 Control Card for door closing. The main board adds “door closed normally” message to the local log file. If main board does not receive door closed signal message within specified time interval, it adds “door stay opened long time” message to the local log file, it sends back source of data, which is zero, and door stay opened long time data to C4 Control Card. When Control Card receives these data, it activates the beepers of both proximity readers. If access is not granted, C4 Control Card changes color of led of corresponding reader to red and produces a beep at that reader. The main board adds “access is not granted for card holder X” message to the local log file.
- ii) It also may send door status sensor data, independent from the readers. That is, when none of the proximity readers has read any card. If the data of door status sensor is 1 (That means the door is open) and access is not granted, main board sends back source of data, which is zero, and door opened by force data because it seems that door is opened by an external force and main board adds “door opened by force” message to the local log file. In this case, C4 Control Card sends a signal to the proximity readers to change the color of their leds to red and activates the beepers of both proximity readers.

6.1.1.5 Process and Data Flow Charts

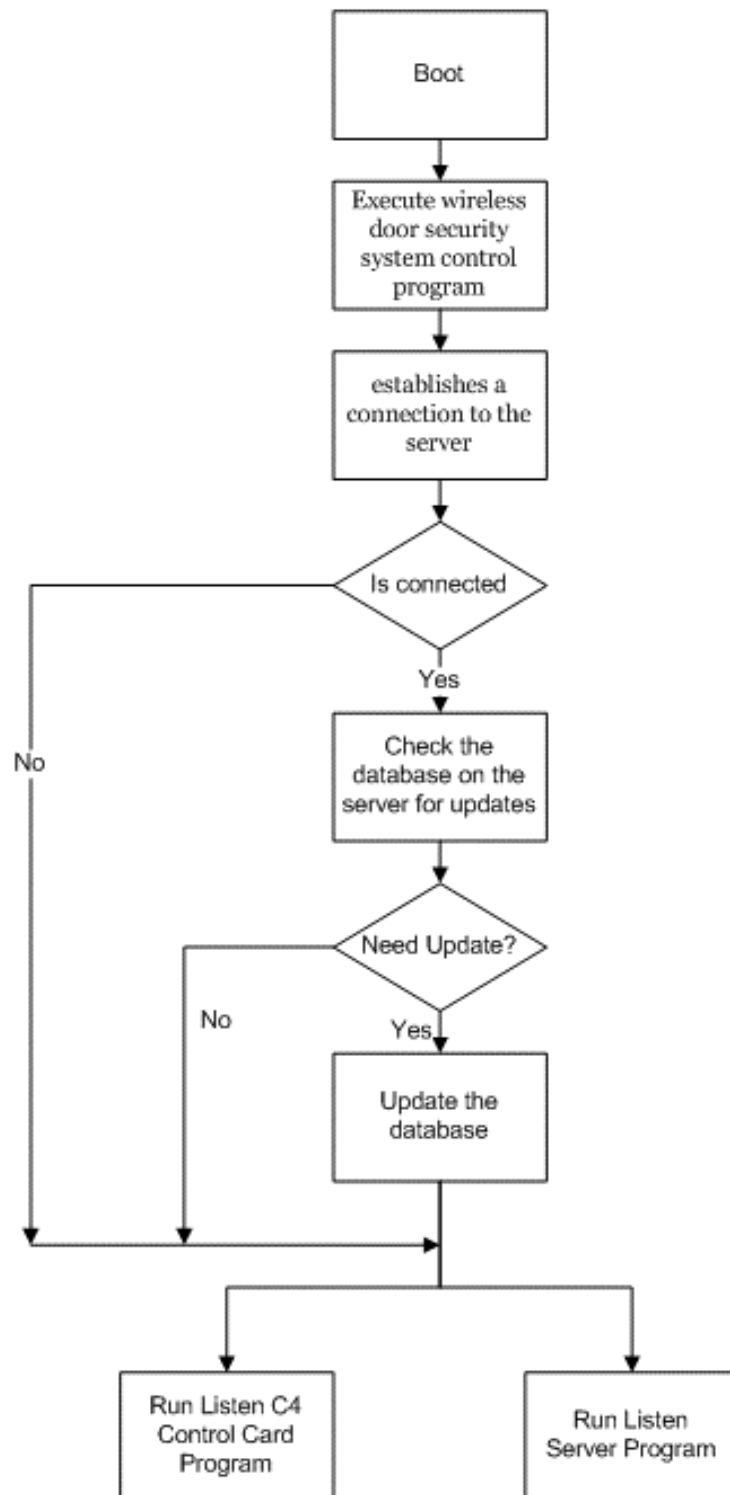


Figure 10 – Flow chart for main board

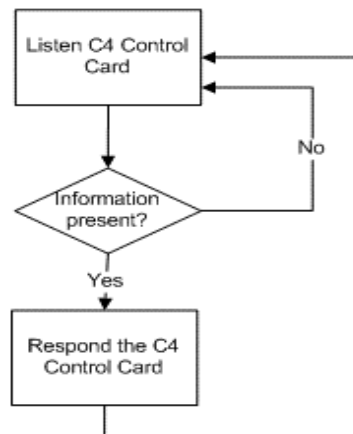


Figure 11 - Listen C4 Control Card

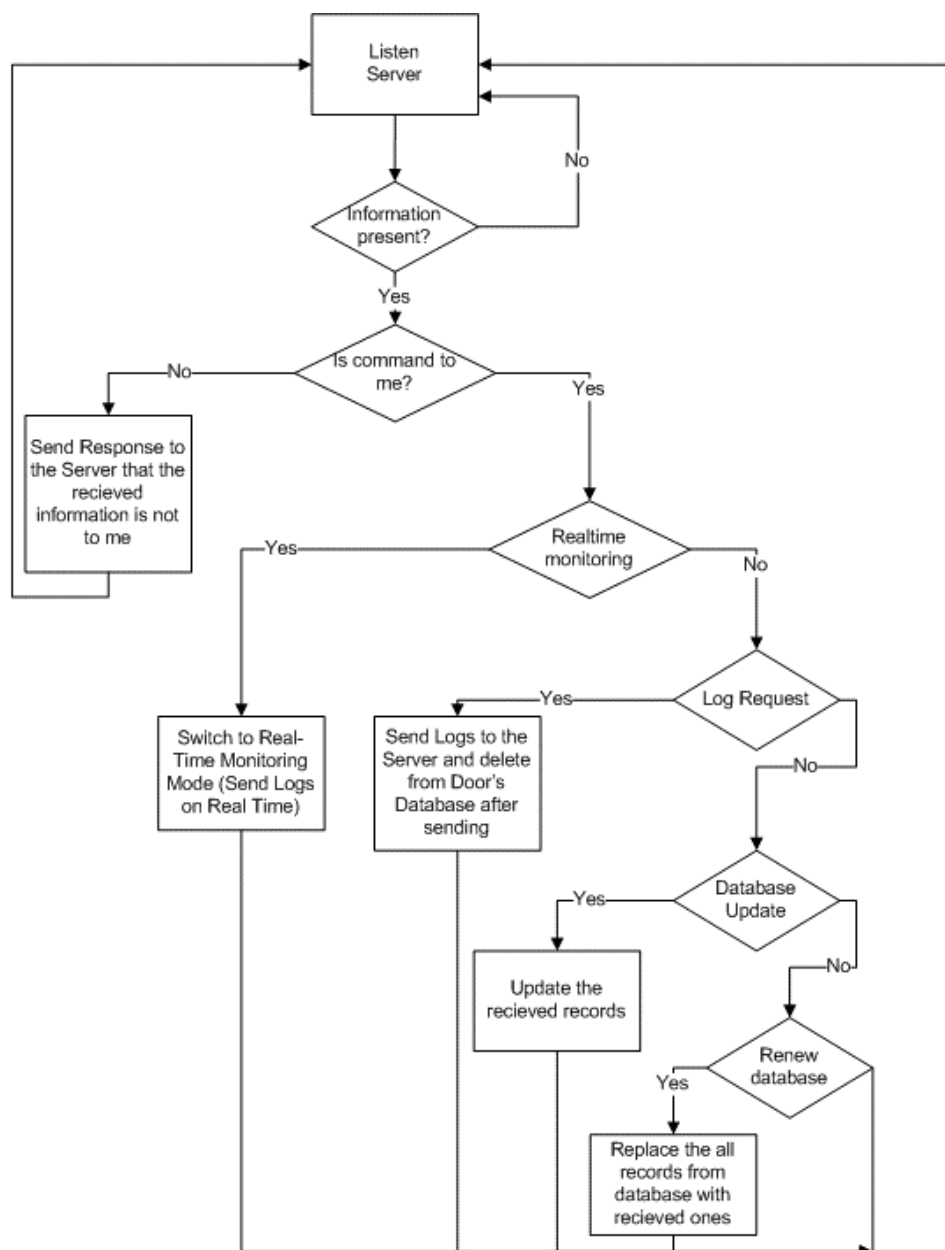


Figure 12 - Listen Server

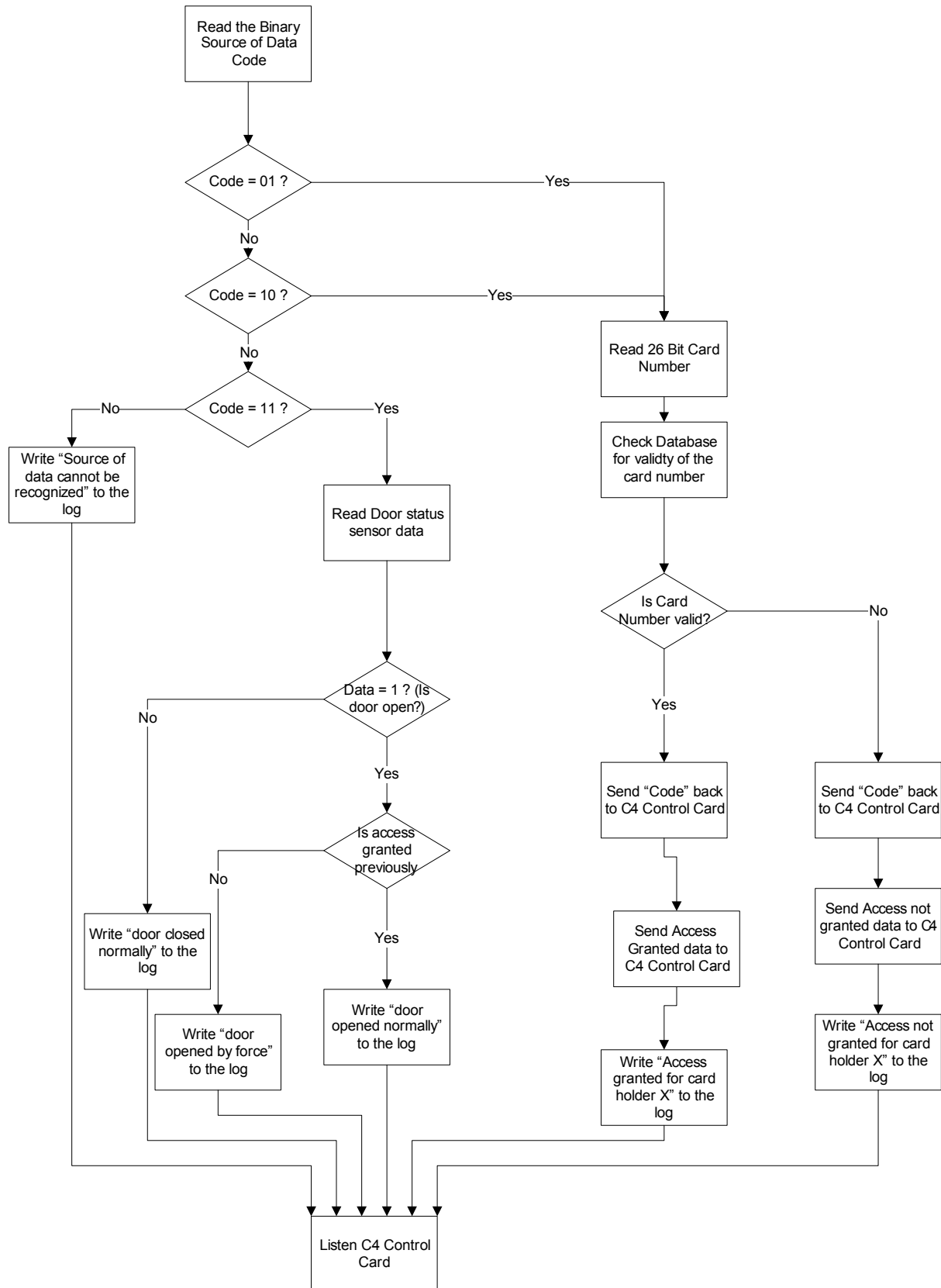


Figure 13 - Flow chart for main board (Respond to C4 Control Card part)

6.1.2 Connections:

6.1.2.1 Main Board, Control Program, Wireless Connection

In this project, 128MB USB pen-disk is used as disk space. Main board is booted by means of this disk. Firstly, it operates an operating system and then runs the program, which is written in C/C++, for controlling wireless door security system. Firstly, this program establishes a connection to the server with using wireless Ethernet card and it updates its database from the server if it is necessary. Finally, the program starts listening to C4 Control Card for any card reading operation or any door status change.

- Wireless connection is based on TCP/IP protocol. The program running on main board and master computer will communicate with each other. We will use socket programming written in C/C++ for packet communication.

The data format that will be sent among main board programs and administrator program will be as following:

DoorID	MessageTypeCode	DataLength	Data
--------	-----------------	------------	------

Figure 14 – Data format

The first field will be ID number of the door. The administrator program writes the ID number of the door to which it is sending the data at this field. The control program running on main board will verify that the data has been sent for its door by checking DoorID field of the data.

The second field will be message type code. The control and administrator program will check this field and understand what kind of data it is and read the remaining data accordingly. The following table shows the codes, their meanings and the direction of messages.

The third field will be length of data. The control and administrator program will check this field and learn the length of the remaining data and reads as many bits as the number written on this field.

The fourth field will be data itself. It may be a log data, which recorded card ID, reader no, time and etc. or it may be database table records.

Message type codes are as following:

MessageTypeCode	MessageType	Direction
001	Log Data	MB → Master computer
010	Database Update Request	MB → Master computer
011	Wrong DoorID Notification	MB → Master computer
100	Real-time Monitoring Request	Master computer → MB
101	Log Data Request	Master computer → MB
110	Related Door-Database Data	Master computer → MB
111	All Door-Database Data	Master computer → MB

Figure 15 – Message type codes

*If the Master computer sends the all door-database data (Message type code=111), then the data format will be as:

DoorID	MessageTypeCode	DataLength	Data	DataLength	Data	DataLength	Data
		For tblDoors table		For tblCards table		For tblCalendars table	

Figure 16 – Data format (detailed)

6.1.2.2 Wiegand Compatible Proximity Reader

Proximity readers must send data according to the Security Industry Association's Wiegand Reader Interface Standard. Readers have a single multi-color LED and an internal beeper to indicate status and error information. The recommended cabling type for the Wiegand interface is a 7 core screened cable, which should be no longer than 150 meters.

The following table (Figure 17) describes the signals for the cable:

Color	Wiegand Signal Description	Signal Direction
Yellow	Beeper Control (optional)	From Host
Purple	Green LED Control (optional)	From Host
Brown	Red LED Control (optional)	From Host
White	Data 1	To Host
Green	Data 0	To Host
Red	Power Supply	From Host
Black	Ground	From Host

Figure 17 – Proximity Reader Connections
(Host is C4 Control Card in our design)

6.1.2.3 The 26 bit Standard Wiegand Output

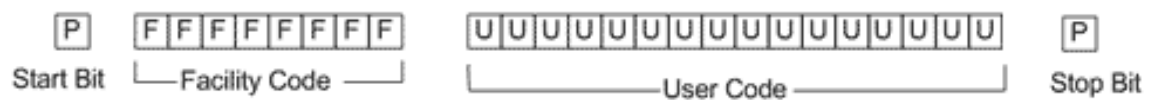


Figure 18 – 26 bit Standard Wiegand Output

The 26 bit Standard Wiegand output is composed of a start bit, 8 bit facility code, 16 bit user code and a stop bit as shown in Figure 18. Start bit is used for even parity of first twelve bits after the start bit. Similarly, stop bit is used for odd parity of next twelve bits.

$$\begin{aligned}\text{Start bit} &= (\text{first twelve bits}) \& 000000000001 \\ \text{Stop bit} &= (\text{next twelve bits} + 1) \& 000000000001\end{aligned}$$

Special cards may be used which comprise a start sentinel followed by 3 digits (000...255), a field separator, a further 5 digits (000000...65535) and an end sentinel. For cards of this type, the 3 digits preceding the field separator are taken as the site code and the 5 digits after the field separator are taken as user code.

6.2 Software:

The second main module of our system solution is software module. It will be the coordinator module of hardware. It is going to control hardware module with its sub-modules. We have 3 sub-modules as Operating System, Database and Administrator Program. Their designs, specifications, diagrams, possible screenshots and responsibilities are as follow:

6.2.1 Operating System:

We considered efficiency, speed and compatibility properties to determine the best operating system. We compare Linux and Windows and ultimately, decided to use a Linux distribution.

6.2.2 Database:

We will use MySQL as a SQL Server on both server side and main board side. For data retrieval, the following pseudo codes will be used:

```
TimeStamp=  SELECT fldDOOUpdate_Time_Stamp
              FROM tblDoors
              WHERE fldDOODoor_NO= DoorNumber

              SELECT RelatedFields
              FROM tblDoors
              WHERE fldDOODoor_NO= DoorNumber AND
                    fldDOOTime_Stamp>TimeStamp
```

RelatedFields: This code will retrieve the door-related fields form the records of tblDoors.

Door Number
Event Capacity
Threshold Capacity
Two Readers Present
Inner Reader Active
Outer Reader Active
Stay Open Duration
Door Locked
Door Forced Sensor
Used Calendar
Online Log

```
RoomNo=      SELECT fldDOORoom_No
              FROM tblDoors
              WHERE fldDOODoor_NO= DoorNumber
```

```
EntranceGroupNo=  SELECT fldGROEntrance_No
                    FROM tblGroup_Permissions
                    WHERE fldGRORoom_NO= RoomNo
```

```

SELECT RelatedFields
FROM tblCards
WHERE fldCAREnterance_Group=EntranceGroupNo AND
      fldDOOTime_Stamp>TimeStamp

```

RelatedFields: This code will retrieve the entrance group- related fields form the records of tblCards.

Card Number
Internal number
Attendance Follow
Expire Date-Time

```

CalenderNo= SELECT fldDOOUsed_Calender
             FROM tblDoors
             WHERE fldDOODoor_NO= DoorNumber

```

```

SELECT RelatedFields
FROM tblCalenders
WHERE fldCALCalender_No=CalenderNo AND
      fldDOOTime_Stamp>TimeStamp

```

RelatedFields: This code will retrieve the calendar - related fields form the records of tblCalenders.

Calendar No
Day
Action Time
Action No
Type
Action Date

These are for selecting data from door, card and calendar tables.

In the administrator program, there will be reporting option to get information from the logs. Some sample reporting sentences are as follow:

Retrieve the list of the users whose names are YYY that entered to room XXX.

Retrieve the phone numbers of the users that entered to room XXX between the dates YY.YY.YY and ZZ.ZZ.ZZ

Report the open doors.

Retrieve the list of the doors that are scheduled to OPEN in the date ZZ.ZZ.ZZ

We design our database considering possible cases. We thought about some scenarios and dependencies to determine fields and tables. In the syntax specification section we specified our table and field naming convention.

Since there can be connection failure between server and main board due to some external reasons, we decided to hold a subset of main database, which is on the server, on the main board. There exist 10 tables and 58 data fields for server database and 4 tables and 26 data fields for main board. To be able to complete the data dictionary, we designed the relations between tables and also fields. Database fields and relations for server side are shown in

Figure 19 that is namely the Entity Relationship Diagram and those for main board side are presented in Figure 20. For more detailed description of database, namely data dictionary is also included on appendix part.

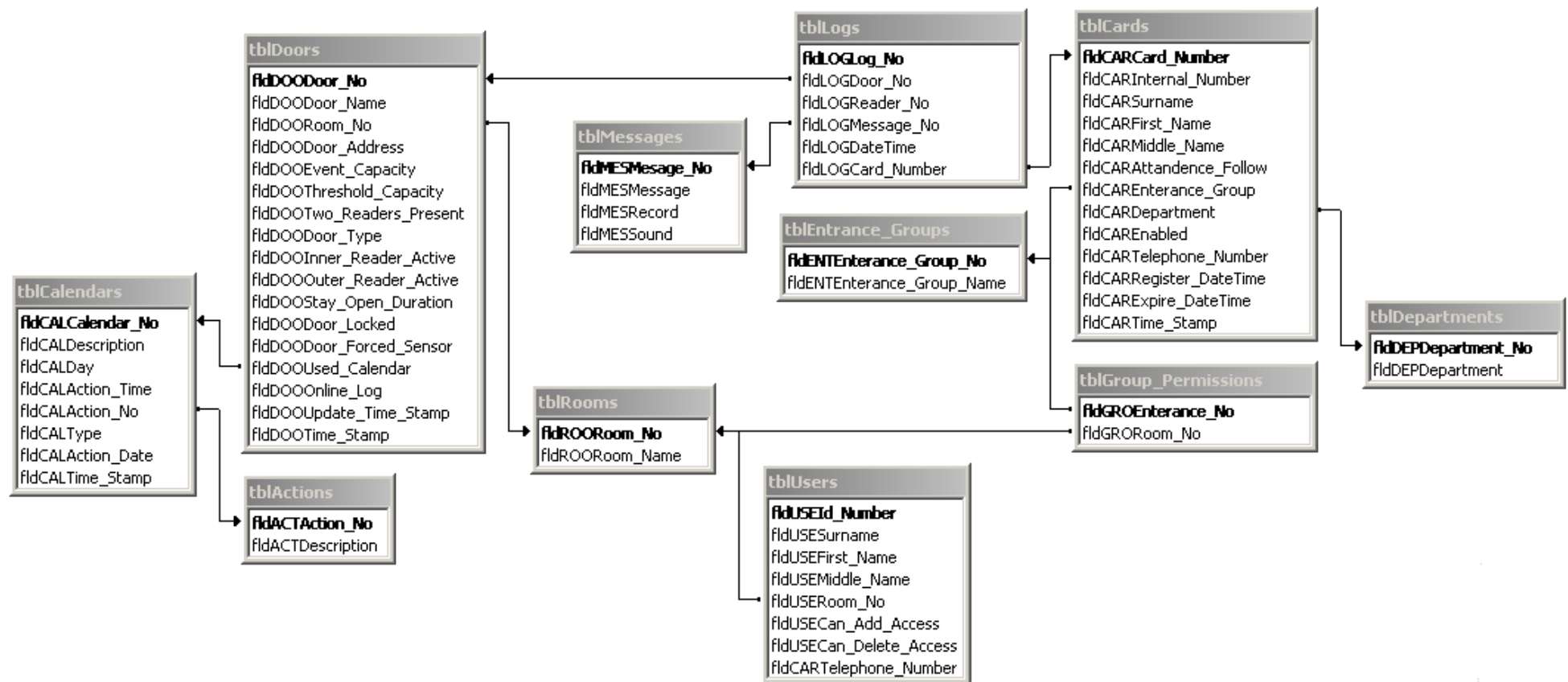


Figure 19 – Entity Relationship Diagram for Server

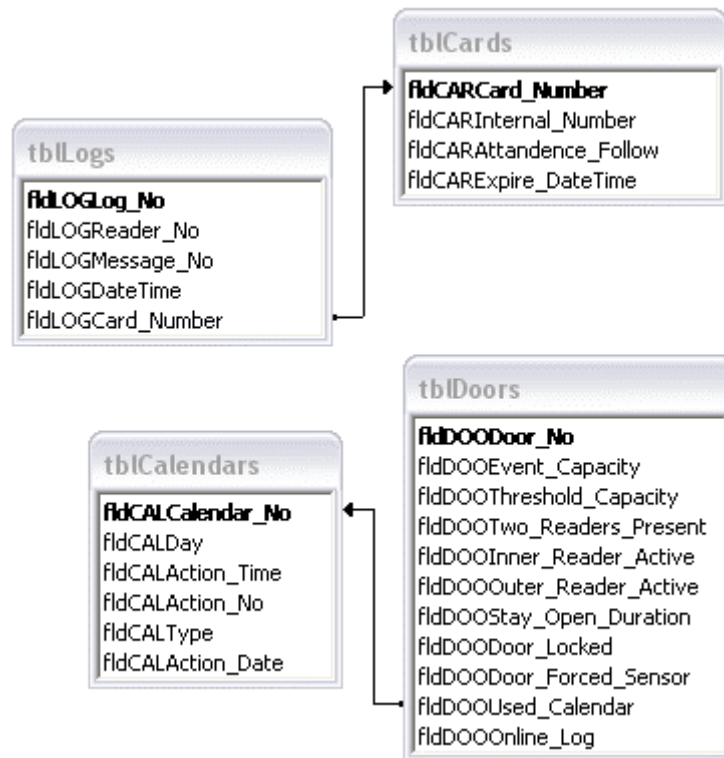


Figure 20 - Entity Relationship Diagram for Main Board

6.2.3 Administrator Program:

We designed a web-based administrator program that will run on the server. It will be programmed by PHP. This program will control the hardware and database. Moreover, it is responsible for creating reports from logs on demand. It will be based on a user friendly Graphical User Interface (GUI).

All functions with adding property will follow the syntax and formal specifications we described later in detailed design report. We created initial interfaces for the administrator program, which is in the center of control.

The architectural structure of the administrator program is as follows:

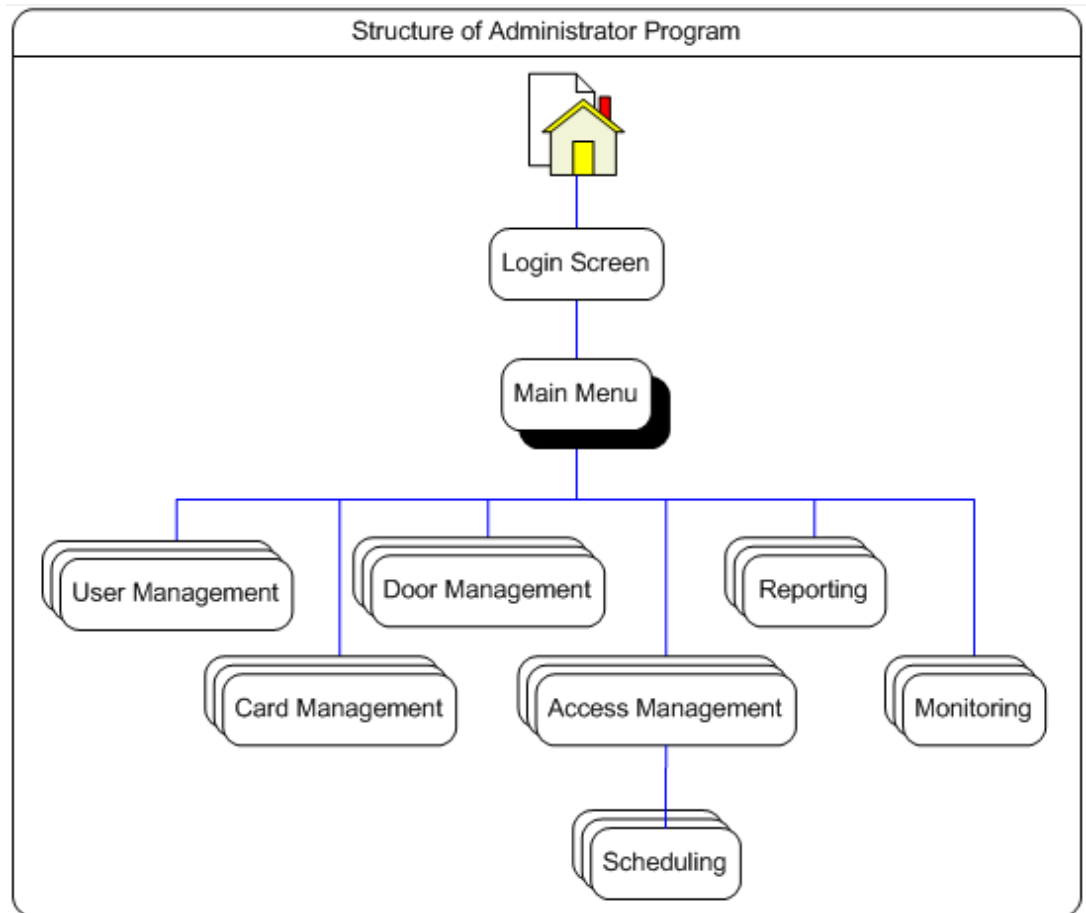


Figure 21 - Structure of Administrator Program

The architecture of the sub-modules of the administrator program is as follow:

6.2.3.1. User Management Sub-Module

We make the dividing of the management of whole system into subparts possible and distribute these to operators. For instance, administrator can give right of locking a specific door to a specific operator. In order to do this, we add an extra user management part (Figure 22) into the administrator program. Administrator can add a new user, change user information, delete users or list all users or list users based on a query.

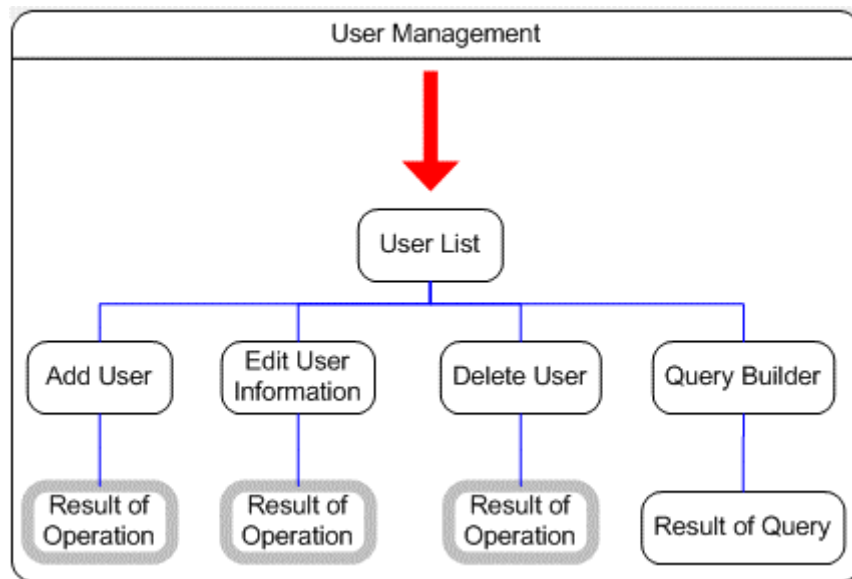


Figure 22 - User Management

6.2.3.2. Card Management Sub-Module

In this module (Figure 23), administrator can add a new card, change card information, delete cards or list all cards or list cards based on a query. After addition, changing and deleting operation, administrator will be informed by displaying result of operation. That is, administrator will see either the operation has been completed successfully or an error occurred.

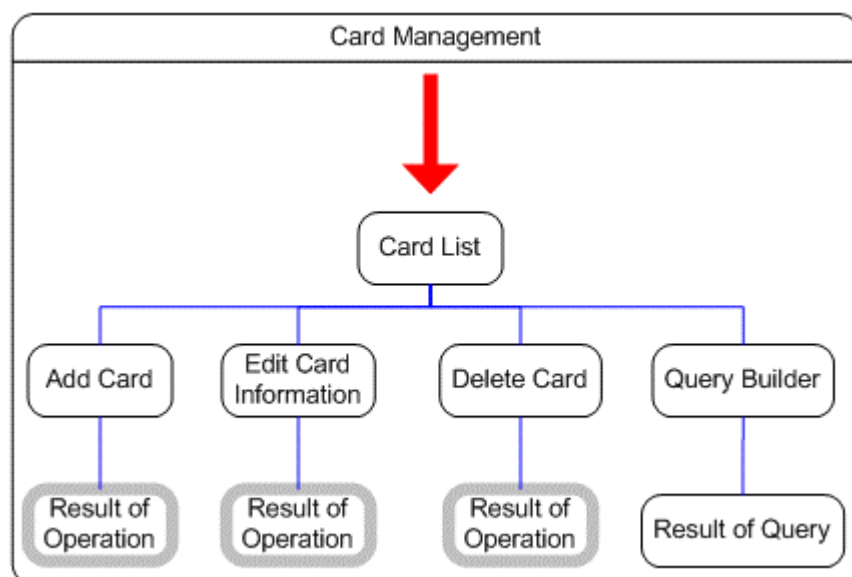


Figure 23 - Card Management

6.2.3.3. Door Management Sub-Module

In this module (Figure 24), administrator can add a new door into the system, change door information, delete doors or list all doors or list doors based on a query. After addition, changing and deleting operation, administrator will be informed by displaying result of operation. That is, administrator will see either the operation has been completed successfully or an error occurred.

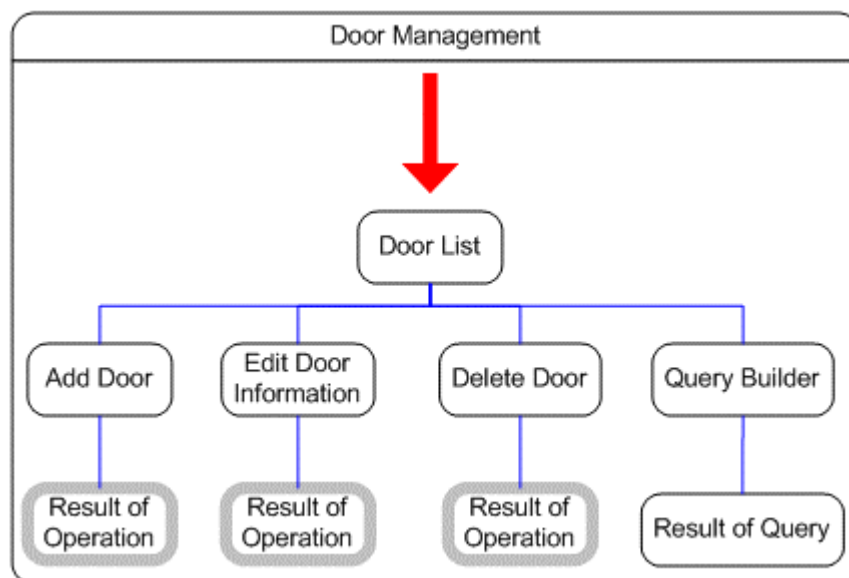


Figure 24 - Door Management

6.2.3.4. Access Management Sub-Module

In this module (Figure 25), administrator can define access groups, open or close door(s), scheduling and use some queries based on these.

In the scheduling part (Figure 26), administrator can add a new entry for specific actions, that is, either a calendar or a specific date and time, change the schedules, delete schedules or use some queries based on these. After addition, changing and deleting operation, administrator will be informed by displaying result of operation. That is, administrator will see either the operation has been completed successfully or an error occurred.

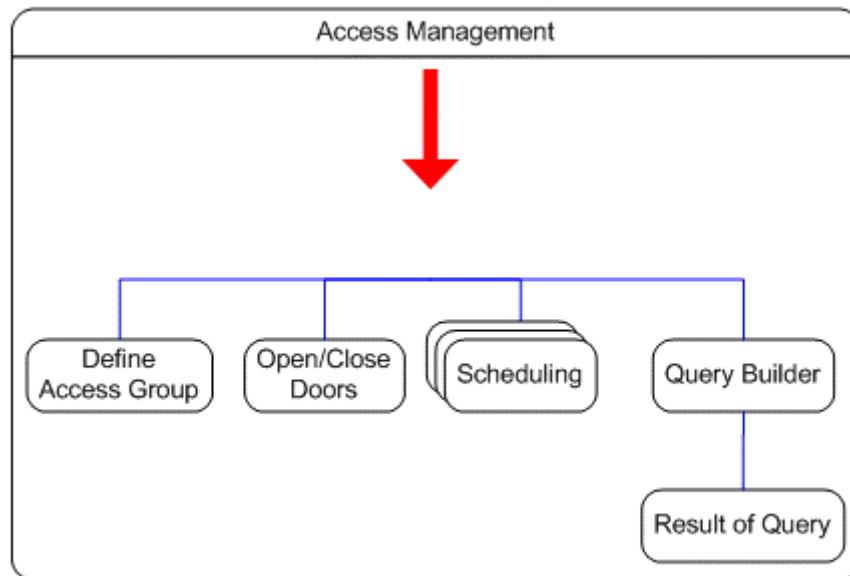


Figure 25 - Access Management

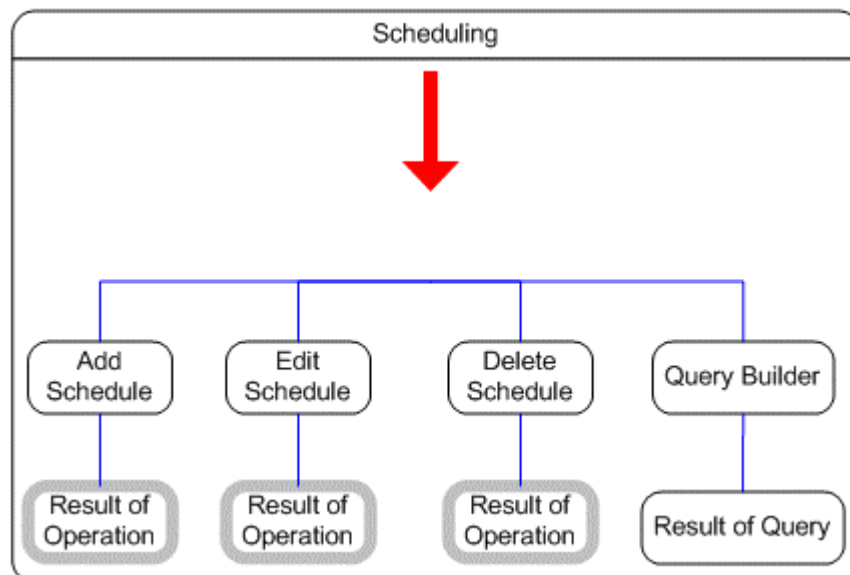


Figure 26 – Scheduling

6.2.3.5. Reporting Sub-Module

In this module (Figure 27), administrator can back-up whole database and logs, restore database from previously taken back-ups, Create reports using logs taken use some queries based on these.

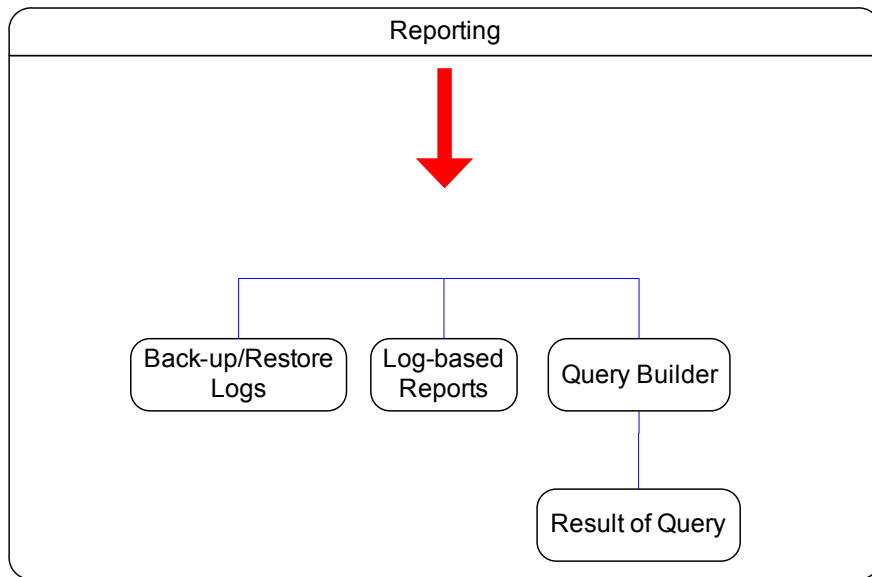


Figure 27 – Reporting

6.2.3.6. Monitoring Sub-Module

In this module (Figure 28), administrator can see the status of door, that is, if the door open or close, the capacity of disc space of main board of doors, etc. Administrator also can monitor the whole system on real-time. And similar to other sub-modules, he/she can use some queries based on these.

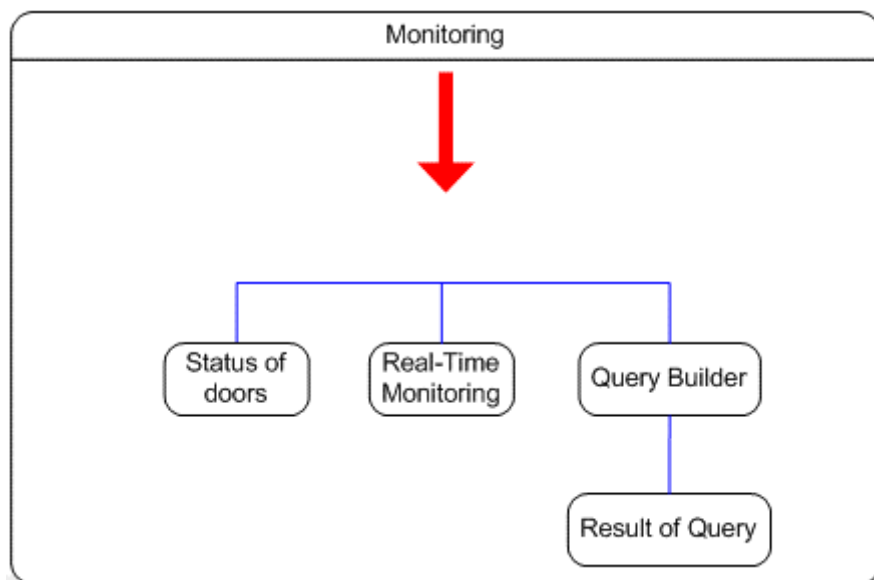


Figure 28 - Monitoring

Here is the some screen shots of initial administrator program with respect to the architectural design provided above:

Login Screen: The administrator program will have a login screen for security. Username and admin password will be asked in this screen, which is shown on Figure 29, and the menu screen will come for supervision (Figure 30).



Figure 29 – Login Screen

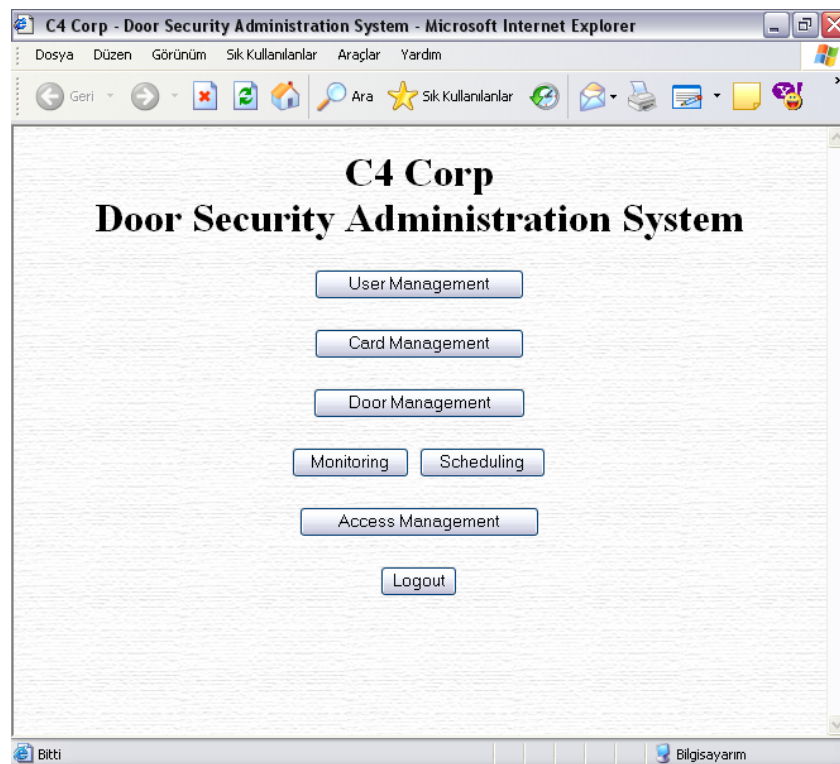
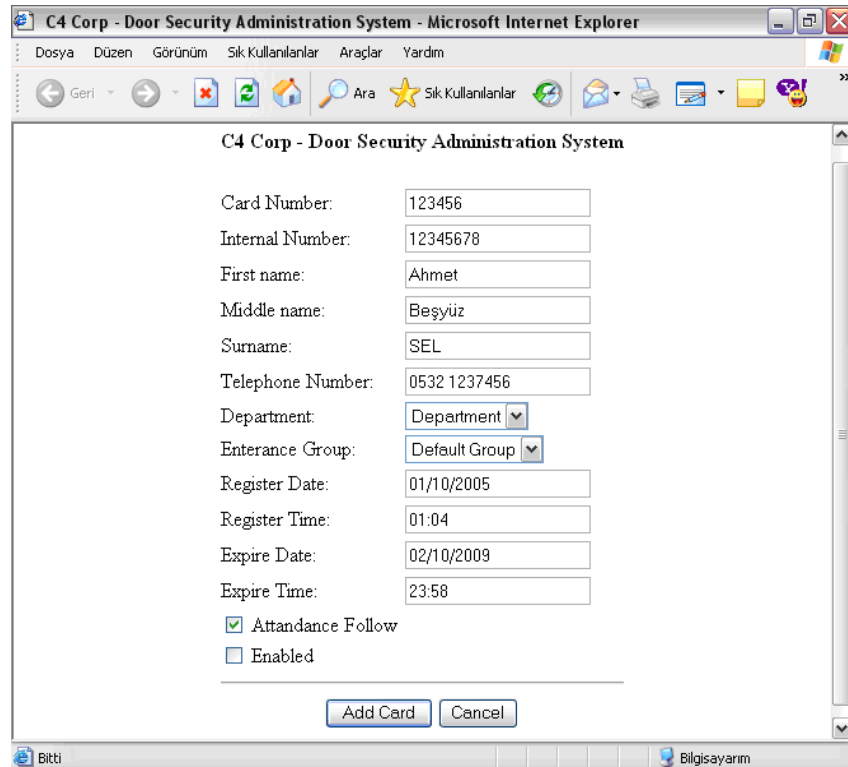


Figure 30 – Main Screen

Main Screen: When the administrator log in to the system, he will see a screen like this one with possible actions he can perform.

Card Adding: This screen will function as extending the system with new cards with their numbers, owners, etc. It will update the tblCards table with the appropriate fields (Figure 31).



C4 Corp - Door Security Administration System - Microsoft Internet Explorer

Dosya Düzen Görünüm Sık Kullanılanlar Araçlar Yardım

← Geri → Ara ★ Sık Kullanılanlar

C4 Corp - Door Security Administration System

Card Number: 123456

Internal Number: 12345678

First name: Ahmet

Middle name: Beşyüz

Surname: SEL

Telephone Number: 0532 1237456

Department: Department

Entrance Group: Default Group

Register Date: 01/10/2005

Register Time: 01:04

Expire Date: 02/10/2009

Expire Time: 23:58

☒ Attendance Follow

☐ Enabled

Add Card Cancel

Figure 31– Card Insertion Screen

Door Adding: This screen will be used to add new doors to the systems. This will update the doors table of the database with the necessary entries (Figure 32).

Calendar Screen: This screen will enable the administrator to create a calendar. In this calendar, the administrator can determine a schedule for the doors to specify time, and also day of week or a date to perform some actions like opening or locking a door, automatically (Figure 33).

C4 Corp - Door Security Administration System - Microsoft Internet Explorer

Dosya Düzen Görünüm Sık Kullanılanlar Araçlar Yardım

← Geri → Ara ★ Sık Kullanılanlar ↻ İletim Kutusu 🖨️

C4 Corp - Door Security Administration System

Door No: 1 (Automatically Assigned)

Door Name:

Door Address:

Door Type:

Room:

Event Capacity:

Threshold Capacity (Percent):

Stay Open Duration (Sec):

Number of Reader: ☐ One Reader ☒ Two Reader

☒ Inner Reader Active

☒ Outer Reader Active

☐ Door Locked

☐ Door Forced Sensor Active

Calendar:

Bitti Bilgisayarım

Figure 32 – Door Insertion Screen

C4 Corp - Door Security Administration System - Microsoft Internet Explorer

Dosya Düzen Görünüm Sık Kullanılanlar Araçlar Yardım

← Geri → Ara ★ Sık Kullanılanlar ↻ İletim Kutusu 🖨️

C4 Corp - Door Security Administration System

Calendar No: 1 (Automatically Assigned)

Description:

Time: h m

☐ Run Every

☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri ☐ Sat ☐ Sun

Date: ☐ Run Once on Date

d m y

Action:

Bitti Bilgisayarım

Figure 33 – Schedule Insertion Screen

Room, Entrance Groups and Department Creating and Editing Screen:

This screen will be used for creating rooms and user groups. This screen will also be used to edit existing groups to change their current settings. You can also add departments to the systems. The added departments will be consistent with the cardholders to be added (Figure 34).

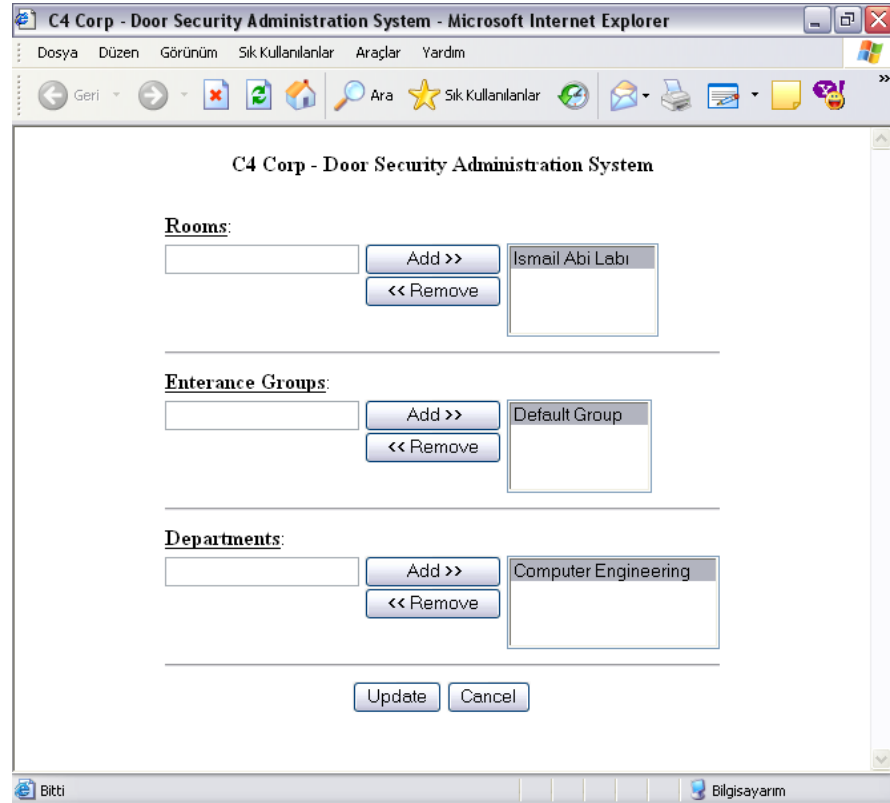


Figure 34 – Parameter Insertion and Update Screen

Log Reporting and Query: This screen will have reporting option for logs kept. The administrator can create queries in logs. For example, the administrator can need to find a weekly progress of a user or he may need to observe daily actions of a specific door. Then, this screen will form a query and request it from the logs for reporting.

We plan to integrate other parts of our system with this web-based prototype. As we specified before, we will design it in an evolutionary way growing up gradually as the components and databases extends, too.

6.2.4 Main Board Program:

The main board program will run on the main board in Linux environment. Firstly, this program establishes a connection to the server with using wireless Ethernet card and it updates its database from the server if it is necessary. Finally, the program starts listening to C4 Control Card for any card reading operation or any door status change; and it listens to master computer for commands. This program will hold the database of its own door. The communication specifications and systems are given above in wireless connection section.

7 Formal Specification of Our System Solution

Our system solution considers different cases and conditions. The hardware, administrator program, database and log procedure are our basis. We designed our hardware and software considering different situations; for example:

- ✓ Users with multiple hierarchies and duties will be handled in administrator program and database. We will resolve the existence of users with multiple duties considering the duty with highest priority. The logs will be arranged with respect to it, too.
- ✓ Electricity Cut: Our system solution requires UPS in order to make our system safe in case of electricity cut. It will be connected to the main board to provide card readers and other hardware with energy.
- ✓ Urgency: Administrator can lock and unlock specified doors at any time if needed. Failures, fire alarms, thieves and such are examples of such urgencies.
- ✓ In the presence of a secondary server, whole database synchronized with servers and the main board of doors will try to connect secondary server in the case of the crash of the server.
- ✓ Recoveries for failures: Since we keep all the logs of all actions one by one and then we update them to the server, it will be easy to recover the system. Only way to examine the logs is via administrator program.
- ✓ ...

These are example additional solutions for the system because we need to add more options to have a something more than a simple door opening and closing system. The project becomes better as we include additional details after considering special use cases as we described above.

And such are considered in our system solution as described in modules.

8 Syntax Specification of Our System Solution

In the syntax specification of our database, we kept the naming conventions agreed in actual English names of data elements. We omitted abbreviations as much as possible because we know that such abbreviations become meaningless later as the program gets complicated.

In the database, we gave “tbl-” prefix to the tables. For the fields, first we added the “fld-” prefix to the field names. Then the first 3 letters of the table to which that field belongs are added in capital letters. The rest of the naming for both tables and fields is in full English, starting with capital letters. In case of multiple words, each word is separated by an underscore and each name starts with capital letter. Methods in programs will also be in full English.

In the administrator program, we will obey these naming conventions too. The actions of administrator program are explained above. Their functions will be named according to the actions in English. It will also aid us to make documentations easier because as the project gets complicated, the readability decreases and the need for good documentation increases.

For example:

```
fldCARCard_Number  
fldCARSurname  
fldDOOStay_Open_Duration  
tblCards  
tblDoors  
createLog()  
updateTable()  
getUserInfo()
```

APPENDIX: Data Dictionary
