CEng 490

DOOR SECURITY PROJECT

by C4-CORP

INITIAL 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 5

3 DESIGN METHODOLOGY 6

4 PROJECT SCHEDULING 11

5 DESCRIPTIONS AND DIAGRAMS OF SYSTEM MODULES 14

5.1 HARDWARE: 14

5.1.1 Card Design: *14*

5.1.1.1 C4 Control Card 14

5.1.1.2 C4 Control Card Specifications 15

5.1.1.3 RS-232 Standard 15

5.1.1.4 Internal Structure (Operating Principles) of C4 Control Card 16

5.1.1.5 Process and Data Flow Charts 19

5.1.2 Connections: *21*

5.1.2.1 Main Board, Control Program, Wireless Connection 21

5.1.2.2 Wiegand Compatible Proximity Reader 21

5.1.2.3 The 26 bit Standard Wiegand Output 21

5.2 SOFTWARE: 22

5.2.1 Operating System: *22*

5.2.1.1. Linux LiveCD Router Specification 23

5.2.2 Database: *23*

5.2.3 Administrator Program: *26*

6 FORMAL SPECIFICATION OF OUR SYSTEM SOLUTION 31

7 SYNTAX SPECIFICATION OF OUR SYSTEM SOLUTION 32

APPENDIX: DATA DICTIONARY 33

Table of Figures

FIGURE 1 – USE CASE DIAGRAM 8
FIGURE 2 – ACTIVITY DIAGRAM OF GRANTING AN ACCESS 9
FIGURE 3 – CLASS DIAGRAM OF THE SYSTEM 10
FIGURE 4 – PROJECT SCHEDULE, GANTT CHART PART I 12
FIGURE 5 – PROJECT SCHEDULE, GANTT CHART PART 2 13
FIGURE 6 – WIRE CONFIGURATION OF TWO-WAY (FULL DUPLEX) COMMUNICATIONS 16
FIGURE 7 - FLOW OF COMMUNICATION 16
FIGURE 8 – COMBINING TWO DATA LINES INTO ONE 17
FIGURE 9– SOURCES OF DATA 17
FIGURE 10 – FLOW CHART FOR MAIN BOARD 19
FIGURE 11 - FLOW CHART FOR MAIN BOARD (RESPOND TO C4 CONTROL CARD PART) 20
FIGURE 12 – PROXIMITY READER CONNECTIONS (HOST IN THIS TABLE: C4 CONTROL CARD) 21
FIGURE 13 – 26 BIT STANDARD WIEGAND OUTPUT 21
FIGURE 14 – ENTITY RELATIONSHIP DIAGRAM FOR SERVER 25
FIGURE 15 - ENTITY RELATIONSHIP DIAGRAM FOR MAIN BOARD 26
FIGURE 16 – LOGIN SCREEN 27
FIGURE 17 – CARD INSERTION SCREEN 27
FIGURE 18 – DOOR INSERTION SCREEN 28
FIGURE 19 – SCHEDULE INSERTION SCREEN 29
FIGURE 20 – PARAMETER INSERTION AND UPDATE SCREEN 30

Initial Design Report

C4-CORP

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

Executive Summary

Initial Design Report is written for the project that is the production of a card based wireless door security system. It is the initial phase of the design to make further 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 designs of modules (card design, connections, operating system, database, administrator program, protocols, and such.) are explained with drawings if necessary. Project scheduling with milestones is included, as well. Databases, data dictionaries, UML diagrams, administrator program with possible screenshots are specified. Moreover, the formal specification of our system solution and syntax specifications are included throughout the initial design report.

1 Introduction

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

Our project is software and hardware for establishment of a card based wireless security door control system. Project involves establishing a wireless secure network connection between the designed controllers and the master computer. Also 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 initially 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. 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 designed the hardware and software modules of the project. Card system, database, administration program and communication protocols are specified. We translated the requirements into an initial blueprint for consulting in the detailed design and constructing the project.

2 Objectives

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

- Accurately transfer users' requirements into an initial draft of the software and hardware product or 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- of the project.
- Define the appropriate design methodology so that the project can be done 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 detailed design.
- Providing a complete picture of the project, addressing data, functional, and behavioral domains for an implementation perspective.

Through the initial design report, we aimed to meet these objectives in order to establish a basement for later design 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 initial 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
   - Operating system
   - Database and
   - Administrator 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 initial 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:

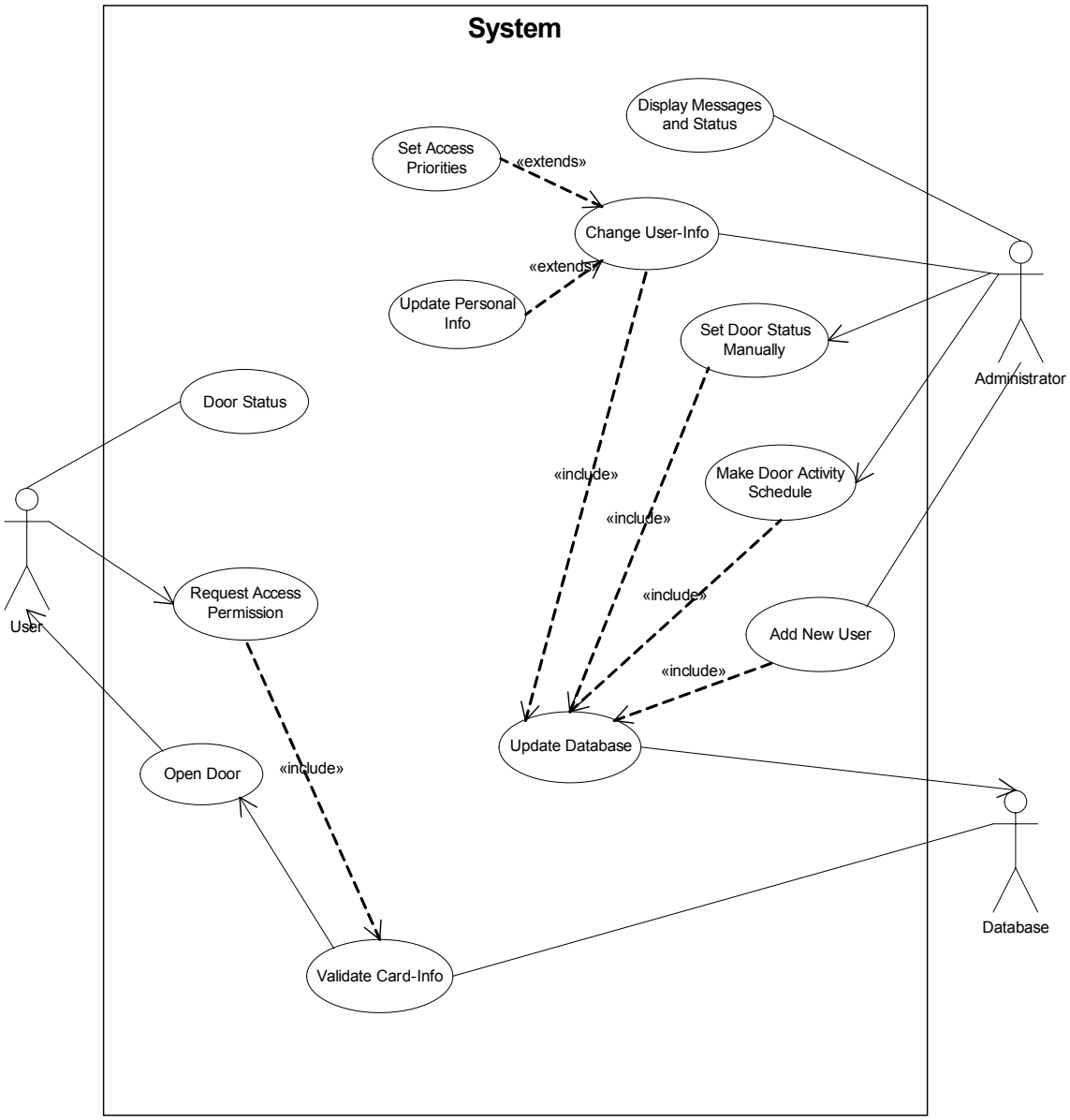

Figure 1 – Use Case Diagram

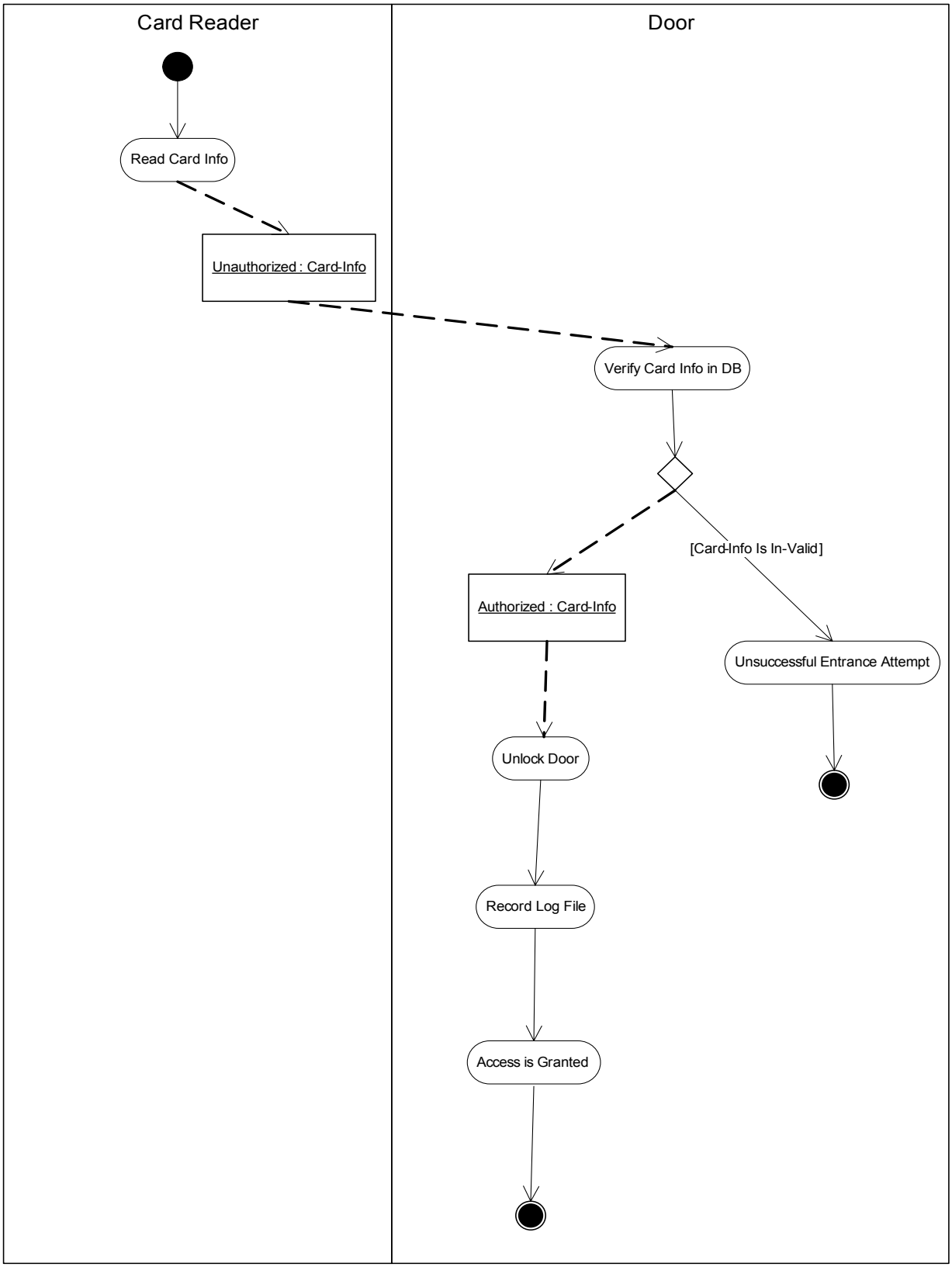

Figure 2 – Activity Diagram of Granting an Access

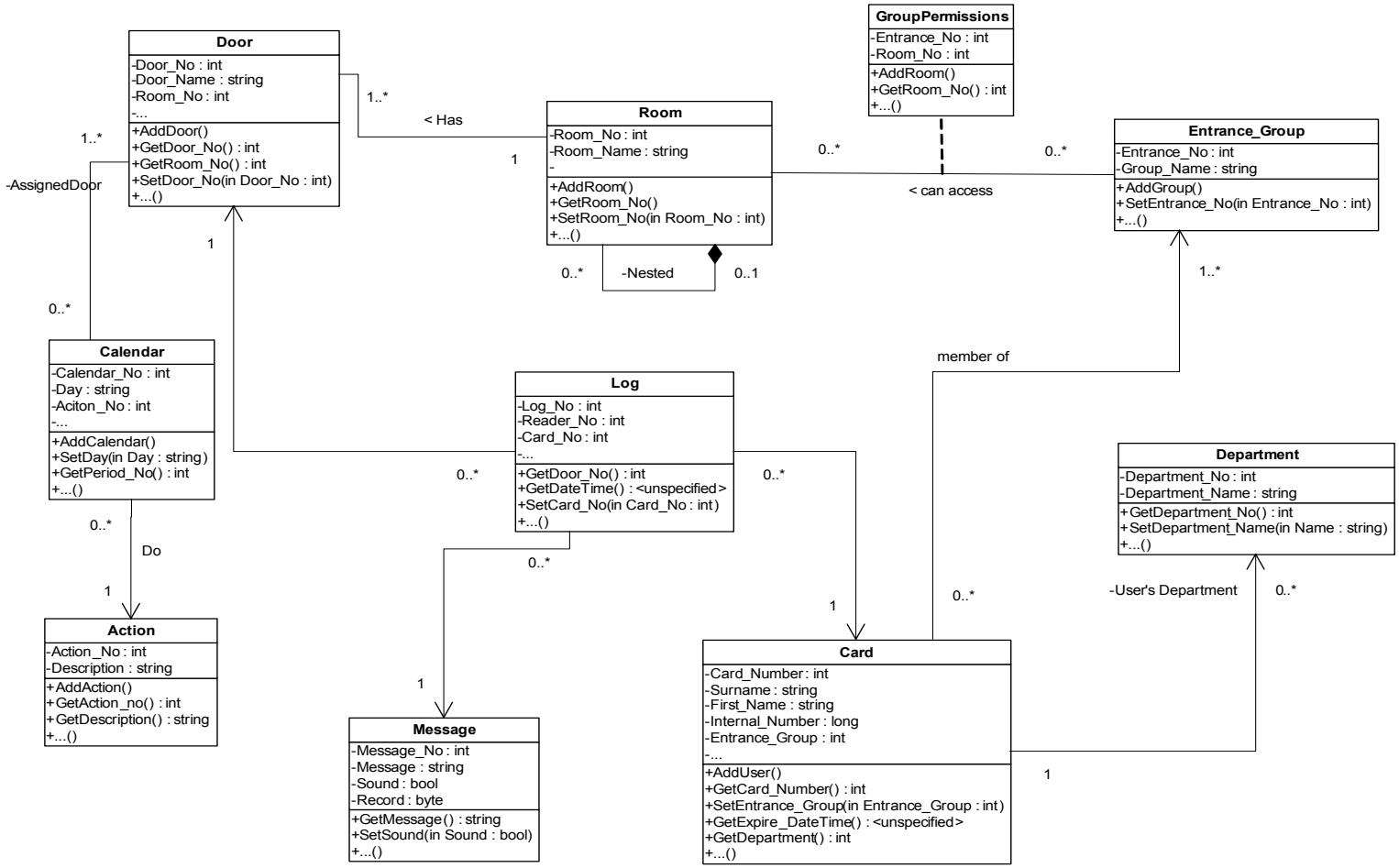

Figure 3 – Class Diagram of the System

4 Project Scheduling

We prepared our approximate 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
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	**- RequirementsAnalysis**	**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**	**100 days?**	**Thu 20.01.05**	**Wed 08.06.05**	**24**
26	Prototype review	5 days?	Thu 20.01.05	Wed 26.01.05	
27	Implementation Initials	16 days?	Thu 20.01.05	Thu 10.02.05	
28	**- System Database Modules**	**15 days?**	**Fri 11.02.05**	**Thu 03.03.05**	
29	DB Tables	6 days?	Fri 11.02.05	Fri 18.02.05	
30	DB Queries	10 days?	Fri 18.02.05	Thu 03.03.05	
31	**- Module**	**41 days?**	**Fri 04.03.05**	**Fri 29.04.05**	
32	Wireless Communication Modules	29 days?	Fri 04.03.05	Wed 13.04.05	
33	Card Implementation	18 days?	Fri 25.03.05	Tue 19.04.05	
34	GUI Implementation	13 days?	Wed 13.04.05	Fri 29.04.05	
35	Administrator Program	6 days?	Fri 22.04.05	Fri 29.04.05	
36	Performance Improvement	15 days?	Mon 02.05.05	Fri 20.05.05	
37	Test Reports	12 days?	Thu 12.05.05	Fri 27.05.05	
38	Implementation (final)	8 days?	Thu 19.05.05	Mon 30.05.05	
39	Documentation	92 days?	Tue 01.02.05	Wed 08.06.05	
40	Milestone	0 days	Wed 08.06.05	Wed 08.06.05	
41	**- Customer Evaluation**	**2 days?**	**Thu 09.06.05**	**Sat 11.06.05**	**40**
42	Feedback	2 days?	Thu 09.06.05	Fri 10.06.05	
43	Milestone	0 days	Sat 11.06.05	Sat 11.06.05	

Figure 4 – Project Schedule, Gantt chart Part I

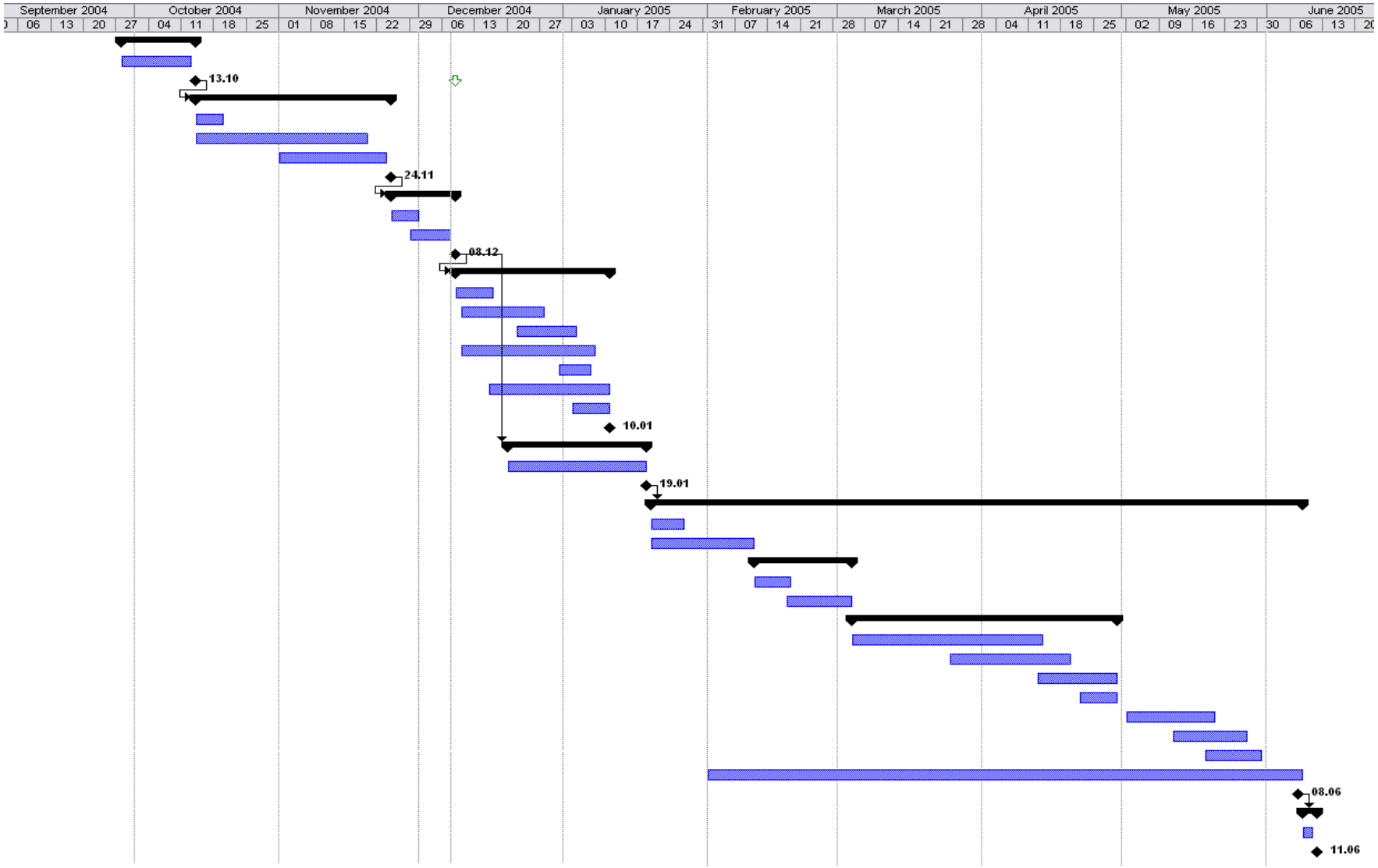

Figure 5 – Project Schedule, Gantt chart Part 2

5 Descriptions and Diagrams of System Modules

5.1 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:

5.1.1 Card Design:

5.1.1.1 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 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.

5.1.1.2 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)

5.1.1.3 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.

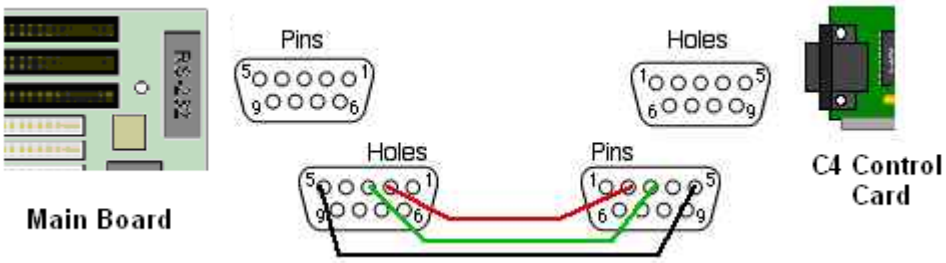

Figure 6 – Wire configuration of two-way (full duplex) communications

5.1.1.4 Internal Structure (Operating Principles) of C4 Control Card

Flow of communication can be seen in Figure 7.

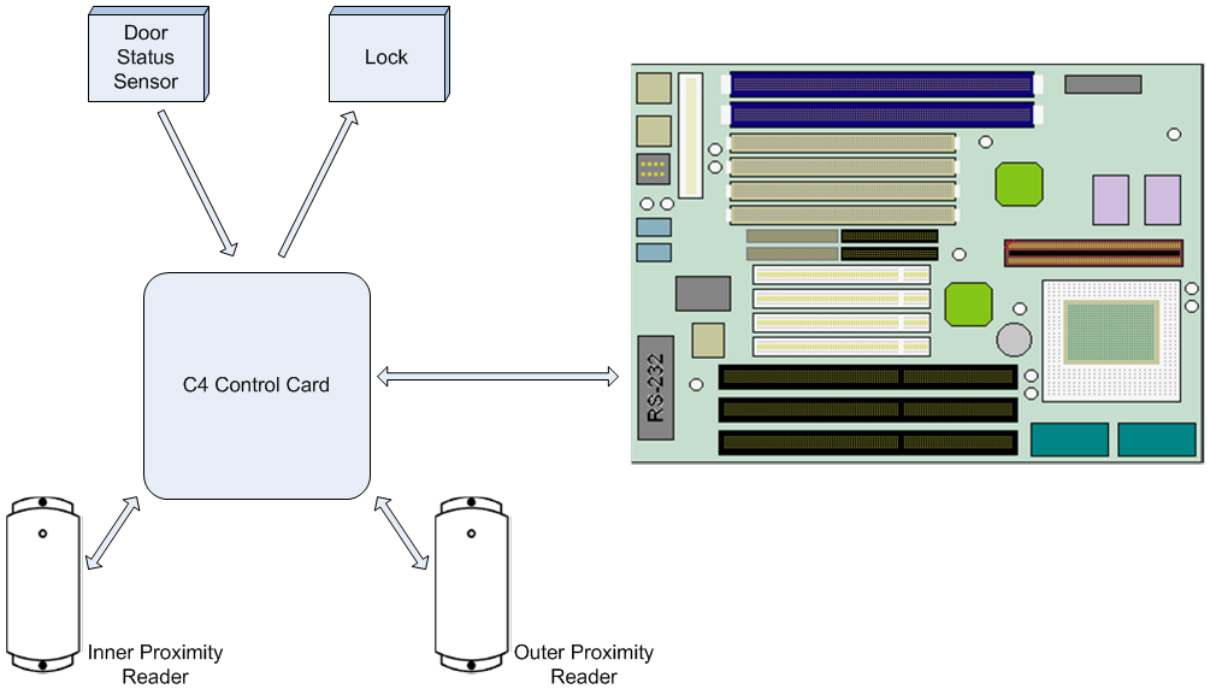

Figure 7 - Flow of communication

Main purpose of C4 Control Card is establishing a connection between Wiegand Proximity Readers and the main board. Wiegand Proximity Readers send 0's through data0 wire and 1's 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.

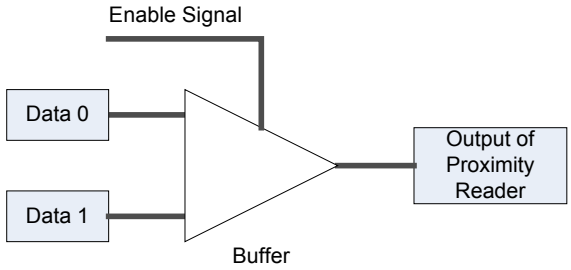

Figure 8 – Combining two data lines into one

Our hardware sends data between a start bit and a stop bit. Before starting to send data, it first sends a binary code (**Hata! Başvuru kaynağı bulunamadı.**), 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 **Hata! Başvuru kaynağı bulunamadı.**, 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 mainboard 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 mainboard 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 mainboard 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 mainboard 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 mainboard 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 mainboard 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.

5.1.1.5 Process and Data Flow Charts

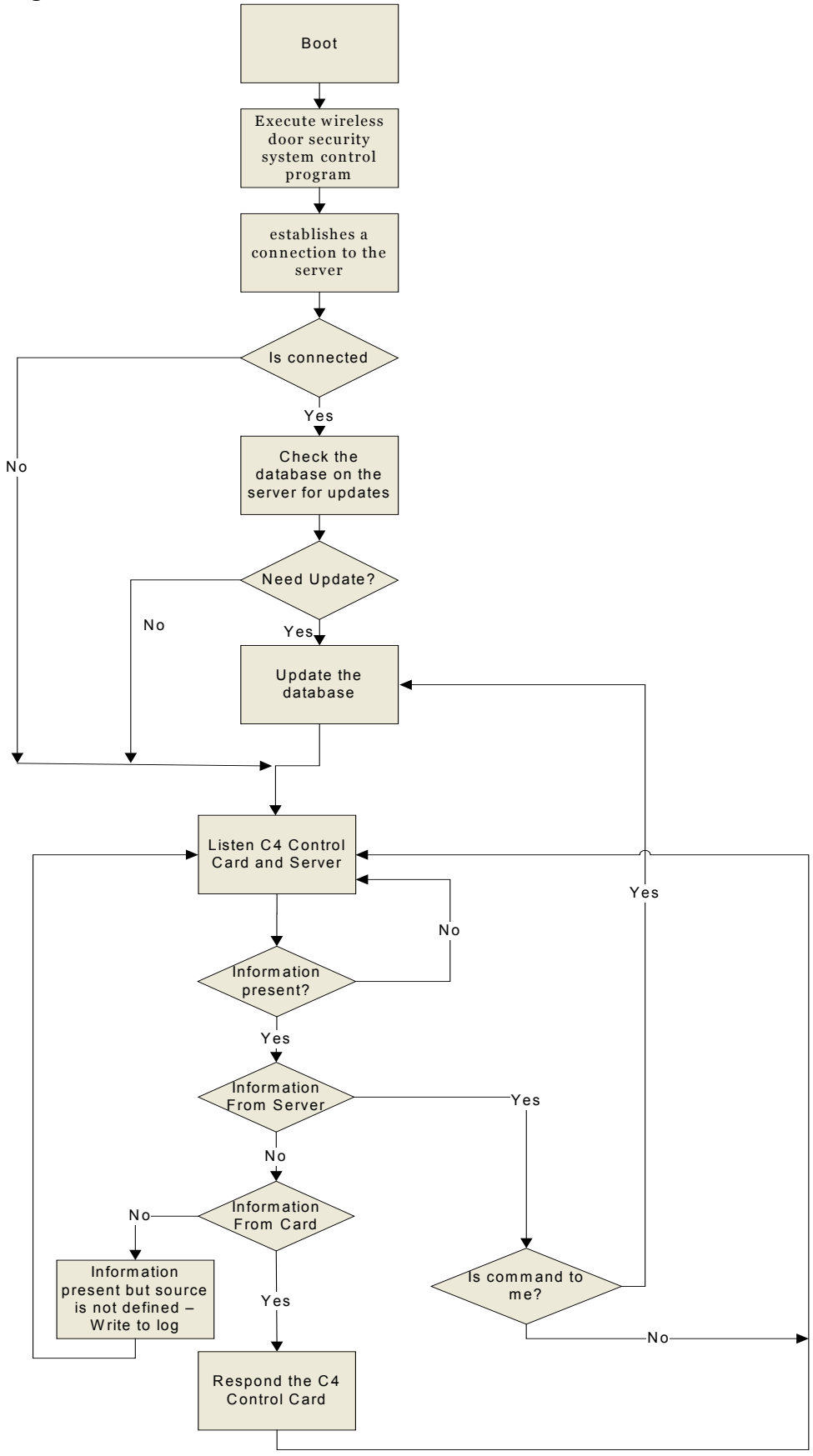

Figure 10 – Flow chart for main board

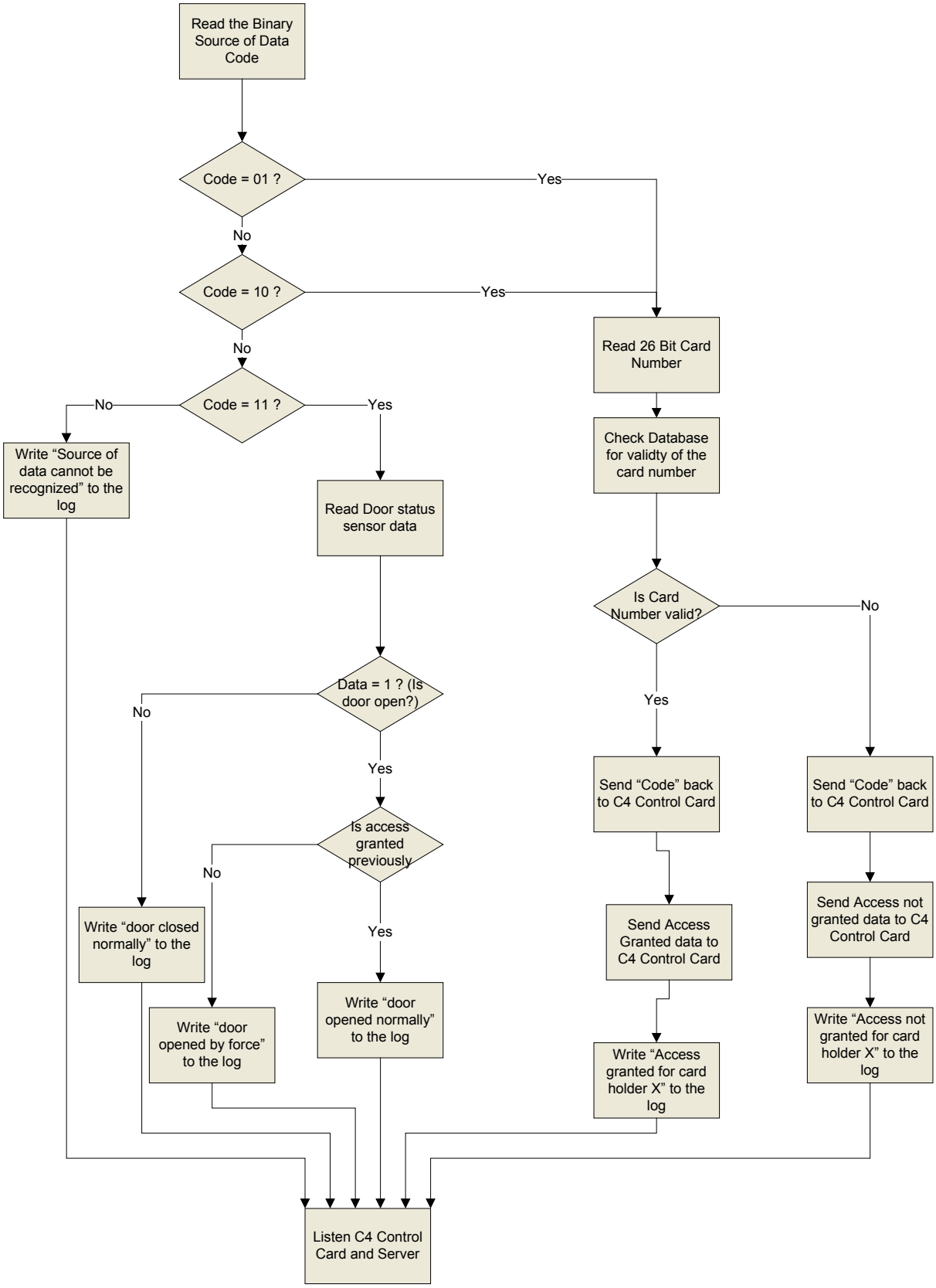

Figure 11 - Flow chart for main board (Respond to C4 Control Card part)

5.1.2 Connections:

5.1.2.1 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 that is needed 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. Wireless connection is based on TCP/IP protocol. Finally, the program starts listening to C4 Control Card for any card reading operation or any door status change.

5.1.2.2 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 an 7 core screened cable, which should be no longer than 150 meters. The following table (Figure 12) 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 12 – Proximity Reader Connections (Host in this table: C4 Control Card)

5.1.2.3 The 26 bit Standard Wiegand Output

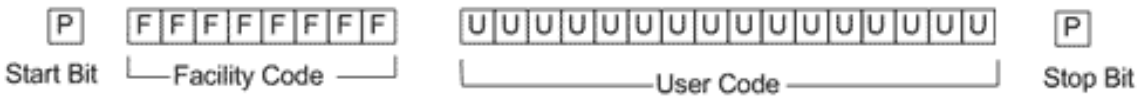

Figure 13 – 26 bit Standard Wiegand Output

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 13. 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.

Start bit = (first twelve bits) & 000000000001
Stop bit = (next twelve bits + 1) & 000000000001

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.

5.2 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:

5.2.1 Operating System:

We kept this part flexible as much as possible because we have not agreed on which operating system we should use. We will use either Windows CE or Linux. We will consider efficiency, speed and compatibility properties of each to determine the best later in detailed design. Fortunately, it seems that we will most probably use a Linux distribution called Linux Live-CD Router.
Here is the specification of Linux Live-CD Router that we are going to possible use.

5.2.1.1. Linux LiveCD Router Specification

The Linux Live-CD Router allows you to share and firewall your broadband connection and use WIFI. You can use DSL, ADSL, Cable Modem, T1, Fixed IPs, Dial-Up, and WIFI, build you own Access Point and more.

Features

- ✓ Share and Firewall your broadband or dedicated Internet connection
- ✓ Includes Firewall Shorewall and Masquerading (NAT)
- ✓ Does not require any installation. It is a LiveCD, your computer simply boots straight from the CD. Does not require a hard disk
- ✓ Supports DSL, Cablemodem, Fixed IP and Dial-Up
- ✓ Use standard and low cost computer, networking and wifi hardware (also USB)
- ✓ Can use multiple ethernet and Wi-Fi 802.11a/b/g cards. Supports PCI, USB and PCMCIA cards (can replace external Access Points!)
- ✓ Traffic Control, QoS
- ✓ DHCP Client and Server
- ✓ Remote SSH administration
- ✓ Includes DNS Cache to accelerate surfing
- ✓ Includes SNMP Monitoring, MRTG graphical statistics
- ✓ Linux Software compatible with Windows and Mac Networks

Hardware Requirements

One dedicated computer with the following minimum specifications: 486 Processor, 16 MBytes of RAM, 2X CD-ROM Reader, Floppy Drive, 1 or 2 ethernet cards. NO hard disk! Optionally a Wi-Fi card.

5.2.2 Database:

We design our database considering possible cases. We thought about some scenarios and dependencies to determine fields and tables. Because 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 7 tables and 54 data fields for server database and 4 tables and 25 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 14 that is namely the Entity Relationship Diagram and those for main board side are presented in Figure 15. For more detailed description of database, namely data dictionary is also included on appendix part.

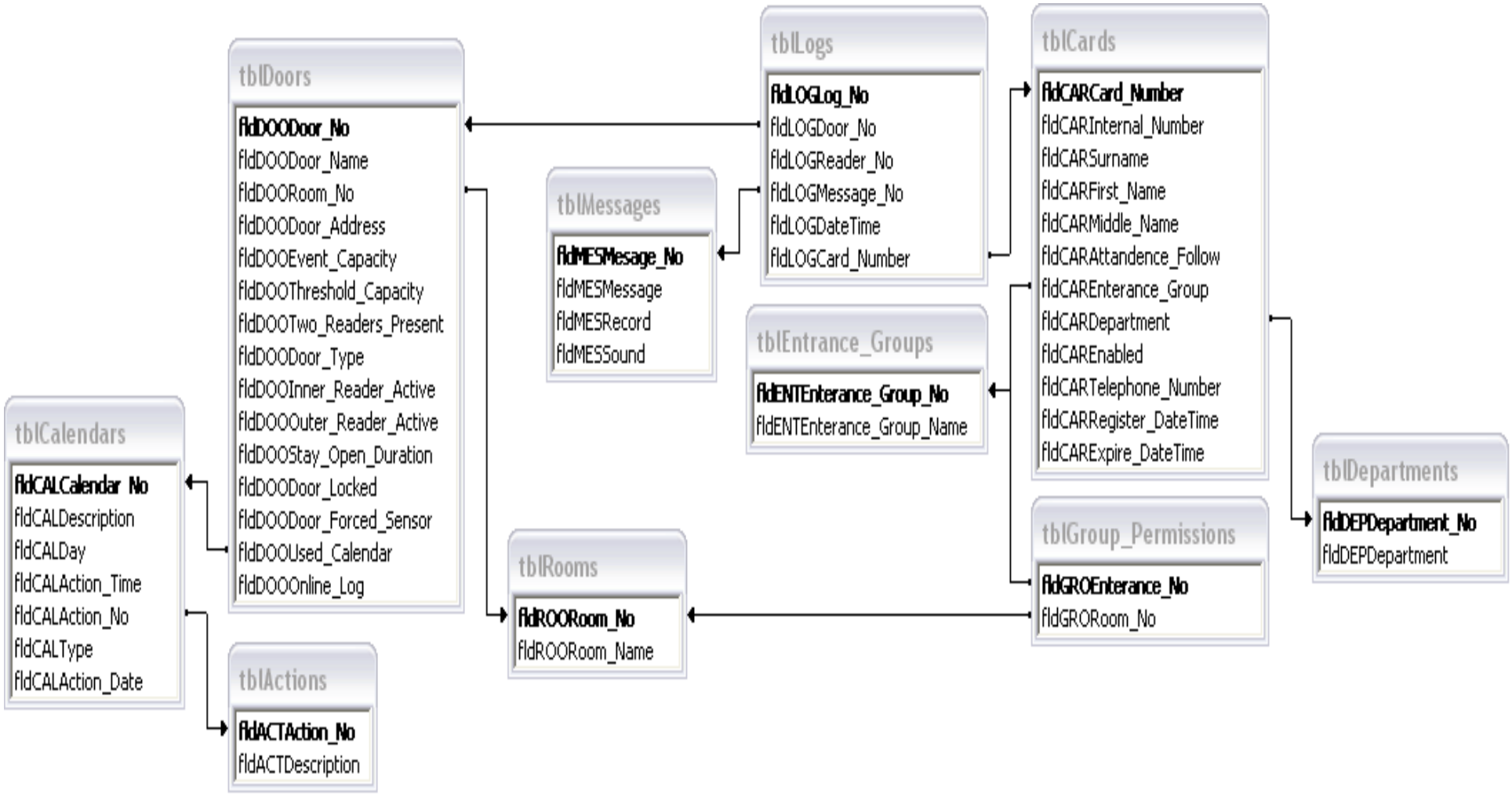

Figure 14 – Entity Relationship Diagram for Server

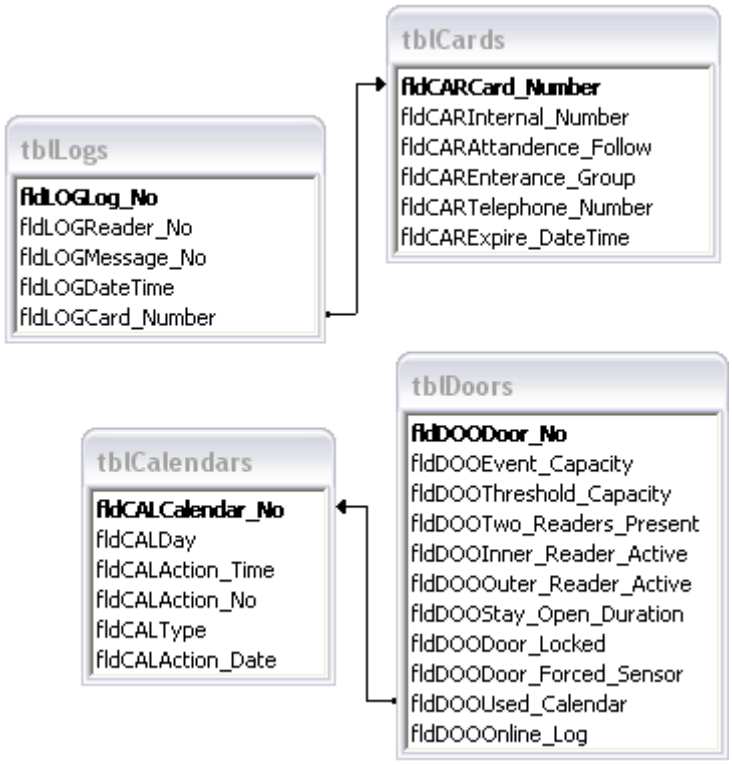

Figure 15 - Entity Relationship Diagram for Main Board

5.2.3 Administrator Program:

We designed a web-based administrator program that will run on the server. 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 initial design report. We created initial interfaces for the administrator program, which is in the center of control. Here is the some screen shots of initial administrator program:

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 16, and the menu screen will come for supervision (Figure 16).

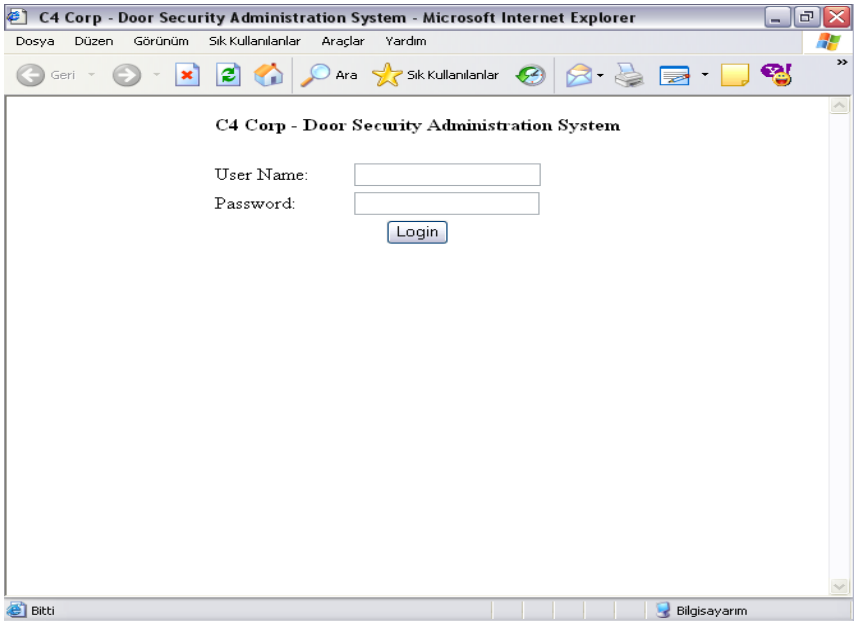

Figure 16 – Login Screen

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 17).

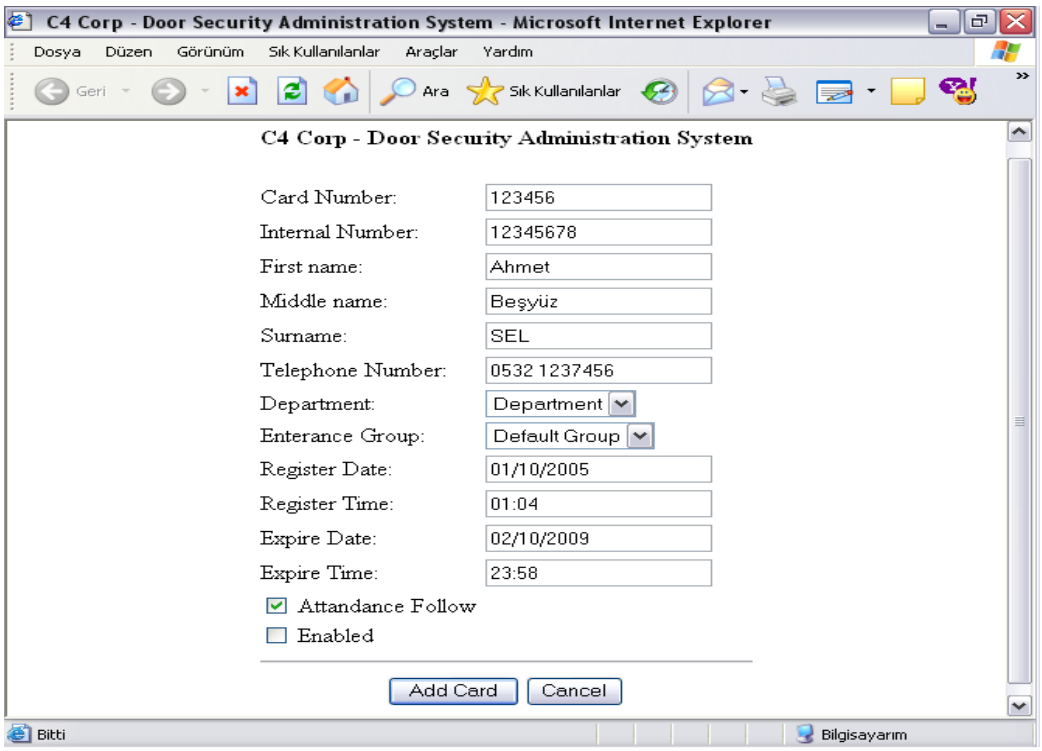

Figure 17 – Card Insertion Screen

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 18).

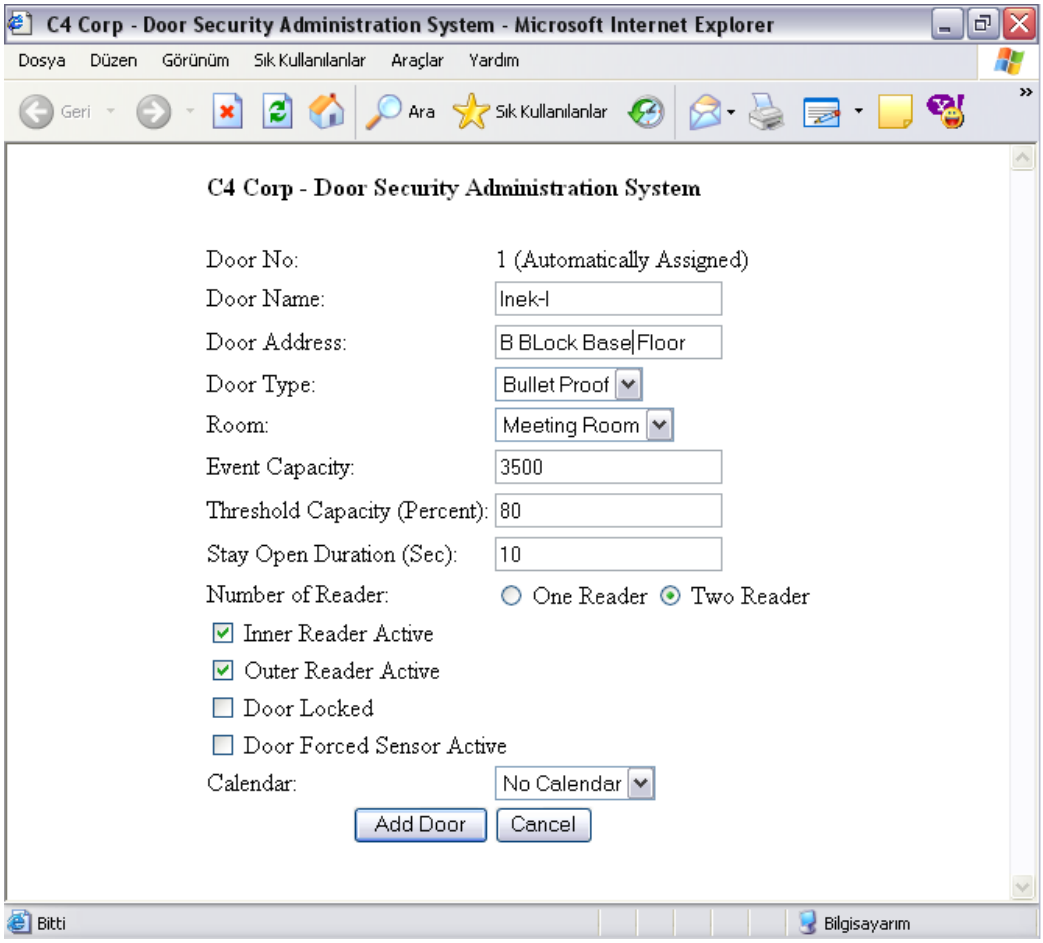

Figure 18 – Door Insertion Screen

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 19).

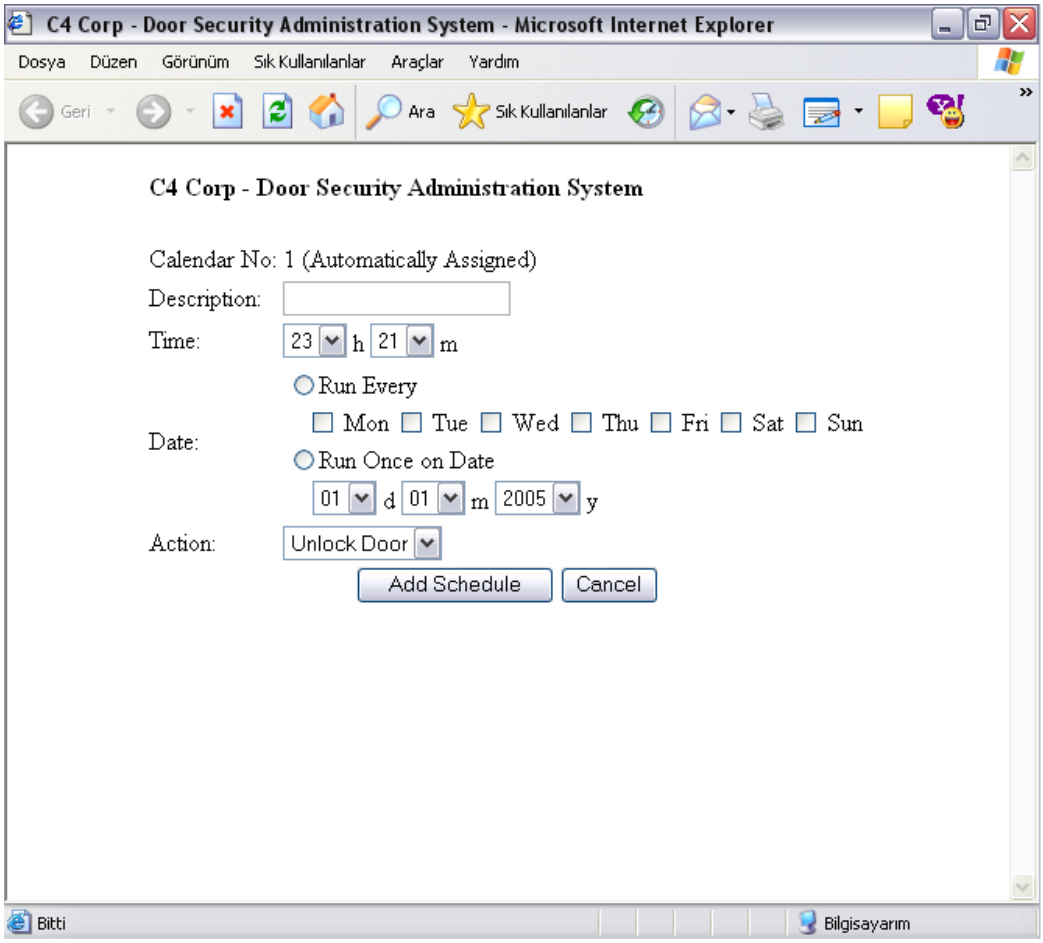

Figure 19 – Schedule Insertion Screen

Room, Entrance Groups and Department Creating and Editing Screen: This screen will be used for creating rooms and user groups. This screen will also 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 20).

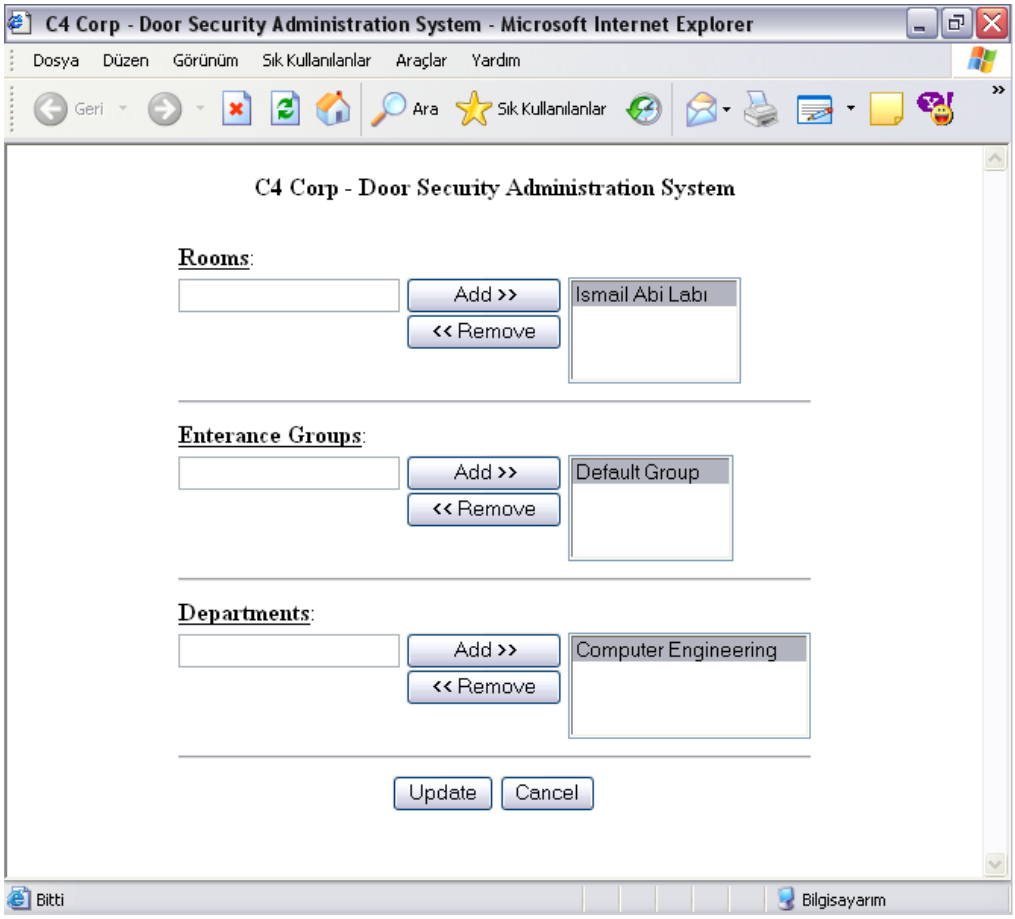

Figure 20 – Parameter Insertion and Update Screen

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. Its further specifications will be clearer in detailed design phase.

6 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.
- ✓ Recoveries for failures: Since we keep all the logs of all actions one bye 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. Their detailed explanations will be in detailed design report because they highly dependent on final designs hardware and software modules.

7 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