University of Nevada, Reno
Computer Science and Engineering Department

CS425/625 Software Engineering

Design Document

Date: Nov 23, 2005.
Project Title: goSmart.

Team Name: JFMS.
Team Members:

- Jim Motta. (Project Manager).
- Fares Qeadan.
- Siraj Malik.
- Muhanna Muhanna.

Instructor and Advisor: Sergiu Dascalu, PhD.
External Advisor: Rasoul Baghal, PhD.
Fred Harris, PhD.

Table of Contents

Introduction

Smarthomes are homes which incorporate various types of technologies into their design. One such technology is called X10 [see Glossary]. X10 technology gives the user the ability to control electrical devices within the household via the electrical wiring in the house. This can be very useful for homeowners of homes that have already been built but would like to incorporate some new technology into their homes. Our project design will be to create an interface that will allow the user to wirelessly control these devices within the house. This will be accomplished using the X10 Firecracker wireless module [see Glossary]. This hardware is currently on the market with a software package. The existing software package, however, does not offer any type of scheduling or automation for these devices. Our software has been designed to be easily learnable by all levels of computer user. Command button interfaces with icons identifying their use are incorporated throughout the package. This makes the software easier to use by giving the user the ability to navigate by text or picture.

Several revisions have been made to our previous design. These revisions deal mainly with the user interfaces and their flow. As stated above, our primary concern with the first part of our project is to create an intuitive interface that will be easy to learn on all levels while still maintaining its functionality. As a result of this, we have decided to give the user various routes throughout the interface to achieve (essentially) the same outcome. This will allow users who want minimal interaction with the software the same control as those who would like to have full control of the environment. As an example of this we have decided to integrate "on the fly" control into our interface. This will allow users to simply control devices by typing in their unique ID and selecting the command. This will not require any type of device or zone setup. Additionally, we will be retaining the ability for the user to setup devices and assign them to zones.

We are currently in our first revision of software with an average amount of progress made to date. We have instantiated the "x10" class which will handle the control of our x10 devices, however, there are a few bugs we are still working out dealing with sending out the command packets. We still need to complete the portion of software that will control the information being implemented and stored. Discussion is ongoing as to how we should handle this. We are currently leaning towards storing the information in files that can be created, accessed and overwritten by the software as necessary. This may or may not require another class to be created to handle these operations.

After some additional research we have found several circumstances that can lead to mixed results using x10. x10 is reliant upon the electrical wiring in a house, therefore, it is also subject to any disturbances within the electrical system. Additionally, most homes do not have phase bridges [see Glossary] setup within the electrical system. Both of these should not affect the operation of our software, however, they could have a vital role in the testing of our software if we are using test beds (Homes) with electrical problems.

Product User

This product is intended for use by home owners who would like to add smarthome functionality to their existing home without incurring a large cost. There is no specific background or level of training the user must have to use the product. However, it is recommended that the user have some minimal prior knowledge of x10 technology and for safety and security reasons users must be of the age 10 or older. Finally, for operating purposes users should have a basic knowledge of home computers and how they work to be able to follow the operating instructions of the product.

Social Impact of the Project

goSmart simplifies life and contributes in fulfilling the growing interest in smart homes. The use of this product is expected to provide better and safer living conditions leading to an improved quality of life and independent living. Luxury and convenience are what most people seek. Going wireless in controlling some devices within the house using the home computer will provide the sense of security and ease of home management. The social impact of this product is very significant to the general public particularly to both elderly and disabled people.

User Accessibility Features

The main functionality and characteristic driving this project is control. The user interface will be designed to be fully functional with a mouse or text entry via keyboard. The view screen of goSmart provides a summary of all the defined devices within a house. From this screen, you can access all of your devices. Each device will have a device name and the Function buttons [see Glossary] (e.g., ON, OFF, DIM, BRIGHTEN, ALL ON, ALL OFF, and DISABLE). Commands are issued to the device by selecting the function and pressing OK. Additionally, the user will be able to enter the device ID by hand in order to control the device. This will allow for minimal setup in order to use the software.

Challenges

Some of our biggest challenges still facing us for this project will include the following:

1. Packetizing the command codes.[see Glossary]
2. Finding reliable test beds for our software.[see Glossary]
3. Integration of additional x10 components.
4. Creating a web portal.[see Glossary]
5. Interfacing the SMS text messaging to the software.[see Glossary]
6. Creating a reliable base module for scheduling events.

Professional Growth

This project provides a very viable market solution. We will be able to gain extensive knowledge on how the market works by utilizing and fulfilling current consumer requests of other similar products being sold on the market. This could be very beneficial for us to see how the actual consumer market reacts to various solutions so that we may anticipate them in the future.

Project Potential

This project has a very large potential within itself. We would like to extend the architecture to include an SMS server that can accept text messages which will control the X10 devices. For example, a user will be able to text message that they want a particular light in the house turned on/off. This project can also be extended into the full X10 home automation market to control devices for heating/cooling the house, webcams, audio/video equipment, and much more.

Ideally, we think that we can modularize the software so that packages can be included later on. For instance, we can create a module that could control TV access/recording, or we could create a module that could control an audio/video system to be used for things like setting wake up alarms, etc. The home automation market is very large and diverse so the possibilities for the software can become quite large.

High Level and Medium-Level Design

Context Model

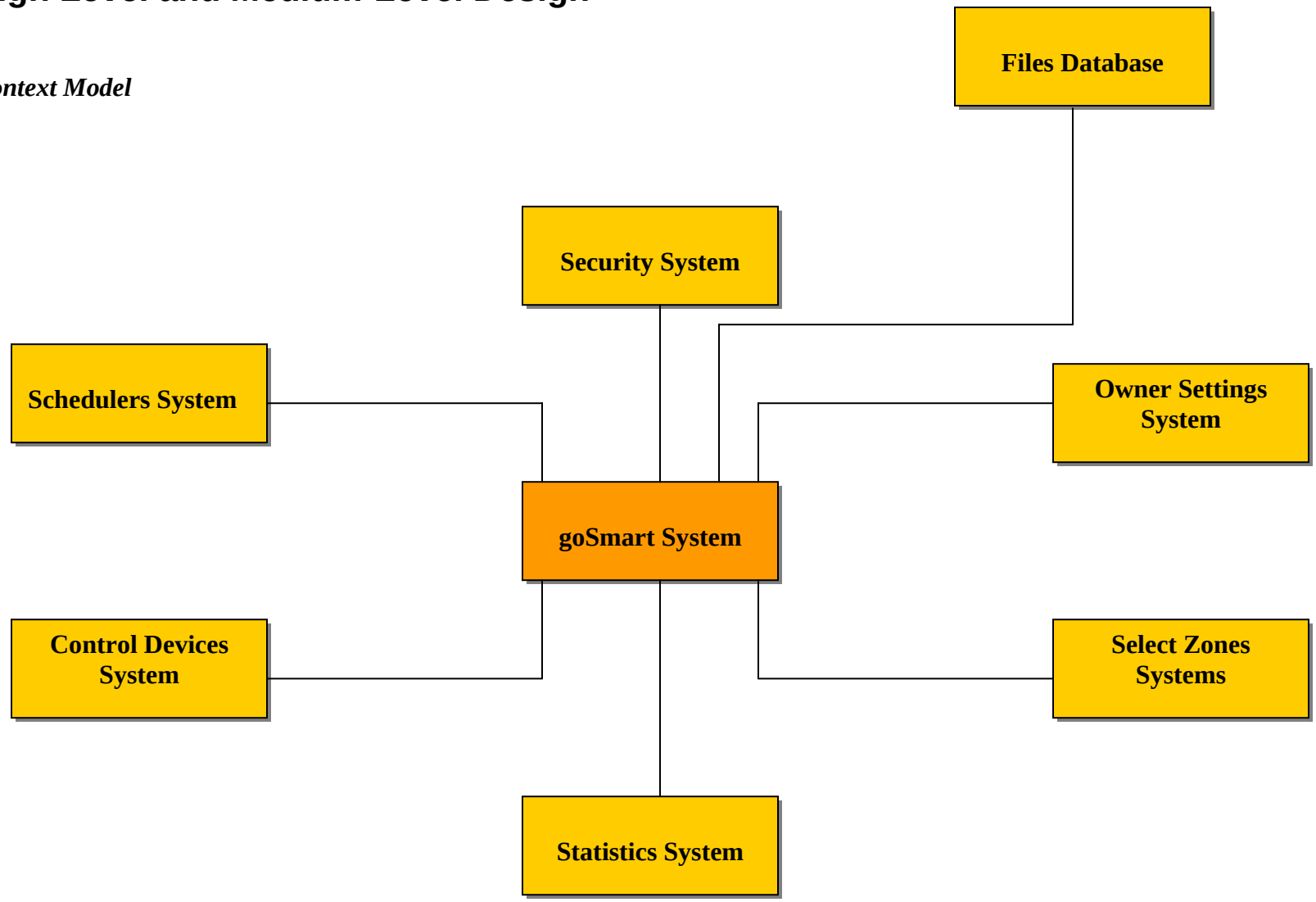
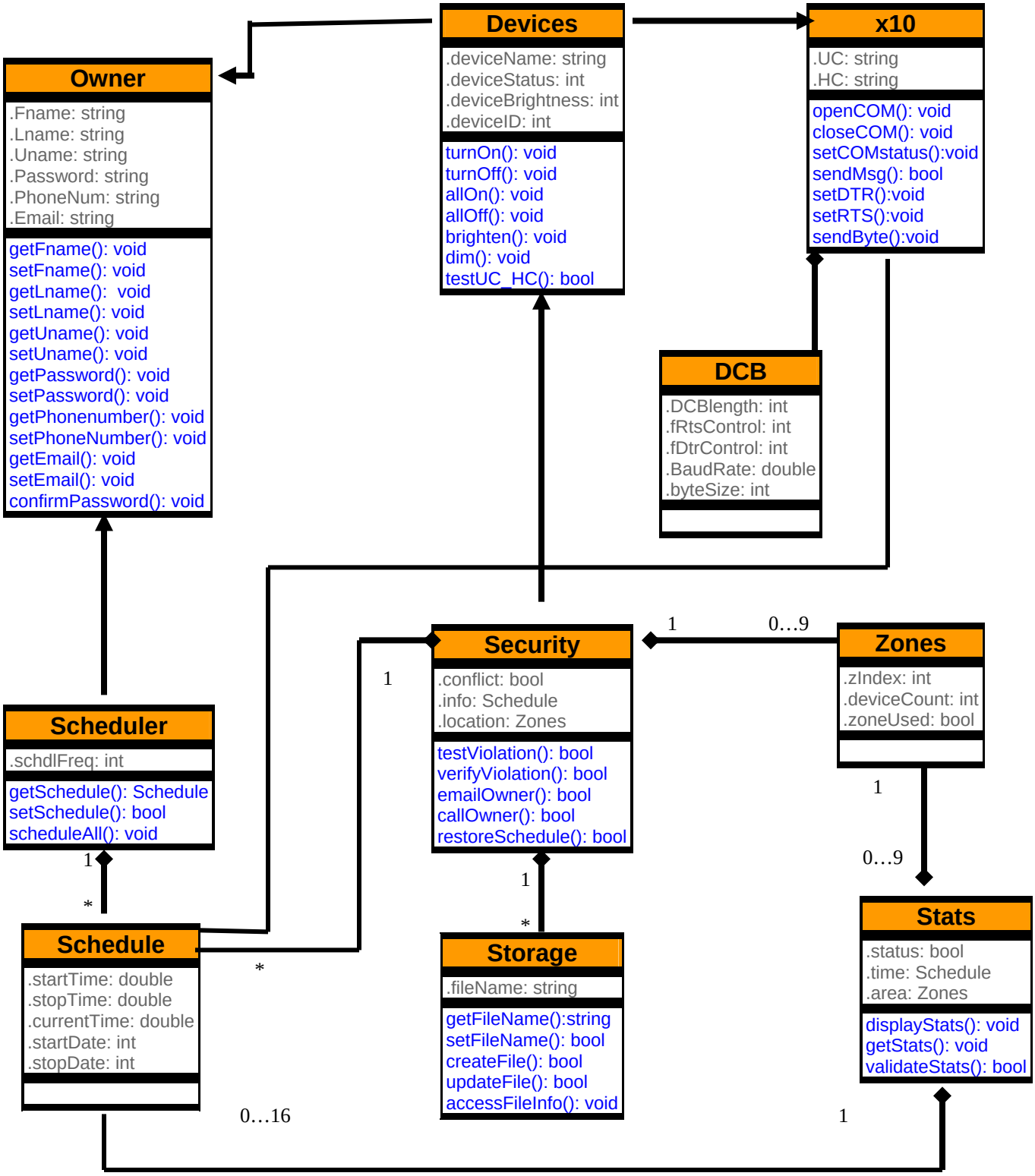


Figure 1. The context of goSmart system

Figure 1 is an architectural model that illustrates the structure of the goSmart system. In this model, you can see that goSmart consists of multiple subsystems, in which the desired functionalities and tasks are embedded. The control devices subsystem provides the main functionality which includes the ultimate operations of turning on/off or brightening/dimming specified devices or lamps. Moreover, the schedule subsystem acts in parallel to the control devices in terms of functionality, however with the scheduling attribute for the duration of specified operations. Both the control devices and scheduling subsystems are capable of enabling or disabling all devices at once for specified zones. Zones are manipulated and formulated within the select zones subsystems, which is nothing but a host area for different sections of the house. Zones are statically allocated and predefined in the goSmart system. Due to the fact that goSmart emphasizes the sense of security in automating a home, the owner's information and settings are necessary for this task and they are implemented and handled by the owner's settings subsystem. The owner's setting subsystem is in charge as well to the login and logout processes in the goSmart software. In addition to the former subsystem, there exists a completion security subsystem which acts as an alert to the owner in cases where security is breached. The security subsystem operation relies on communication with the home owner via email or phone call. For monitoring and observing the status of each device in terms of being on/off, scheduled or not, and its brightness/dimness level, goSmart uses a statistical subsystem that can reveal the indicating information upon request. Given that goSmart uses a system timer and scheduling procedures, saving information is critical and necessary for the success of the system. This is why a files database is proposed to take care of such tasks.

Class Diagram



Class: x10	The role of the x10 class is to communicate with the firecracker and generate all needed wireless communication in order to manipulate the turn on/off and scheduling operations on the specified devices with respect to the matching pair of unit code and house code (predefined by user input).
openCOM()	This function accepts a COM port, passes it to createFile to assign a port handler to it for control. It will also set the port state to TRUE. Function then calls setCOMstatus to attempt to open the COM port. If there is a failure then closeCOM is called and an error is thrown
closeCOM()	Function removes the current port handler and sets the port state to FALSE.
setCOMstatus()	This function is in charge of initializing the COM port. The following functions of DCB (The DCB structure defines the control setting for a serial communication device) are : dcb.DCBlength = specifies the size of the structure dcb.fRtsControl = this enables the RTS line and leaves it open dcb.fDtrControl = this enables the DTR line and leaves it open dcb.BaudRate = specifies the baud rate dcb.ByteSize = specifies the byte size
sendMsg()	This function accepts a house code, a unit code, and a command. This function will first force the house code to lowercase and then convert it to use as index into the houseCodes array. Next it will toggle the DTR and RTS lines to put the firecracker unit into ready mode. It then uses the sendByte function to send the first and second byte of the header, as well as, the house code. Next a case statement is used to determine which command byte to send. Finally, the footer is sent using the same function.
setDTR()	This function accepts the port state that directs a specified communications device to perform an extended function) to set the DTR line to either high or low. If it fails then an error is thrown.
setRTS()	This function accepts the port state and directs a specified communications device to perform an extended function) to set the RTS line to either high or low. If it fails then an error is thrown.
sendByte()	This function accepts a Byte value and uses it to send data bit by bit to the Firecracker unit by toggling the DTR and RTS lines.

Class: Devices	The role of the Devices class is to execute the desired operation of turn on/off, brighten/dim, and turn all lights on/off for specified device (or all) on selected zone. Also, it has the role of testing for possible error where house codes mistakenly was assigned to more than one unit code.
turnOn()	This function accepts a house code and a unit code and passes them to the sendMessage function. It also these values to the previousHC and previousUC variables.
turnoff()	This function accepts a house code and a unit code and passes them to the sendMessage function. It also these values to the previousHC and previousUC variables.
allOn()	This function passes the current house code, and the all_on command to the sendMessage function.
allOff()	This function passes the current house code and the all_off command to the sendMessage function.
brighten()	This function passes the current house code, unit code, and the brighten command to the sendMessage function.
dim()	This function passes the current house code, unit code, and the dim command to the sendMessage function.
testUC_HC()	This function tests if the user input for the house code has not been defined already to multiple unit codes.

Class: Security	The role of the Security class services the core concept of the goSmart software, where security is principle and a main goal. This class is in charge of detecting any failure or an unexpected status change of devices. It works as an interrupt and acts as an alarm for the owner of the system.
testViolation()	This function will be called each time a device is scheduled for operations done in both manually or by the system timer to ensure there is no conflict between the intended operations and the already scheduled ones.
verifyViolation	This function will be invoked automatically upon failure of scheduled devices (i.e. an unexpected change of status). The verification process relies on the information residing in the Storage class.
emailOwner()	This function will also be conducted automatically if violation was found to be true in the previous function. The concept of emailing owner can be concluded as an alert of security violation message sent by attachment to predefined email address (i.e. previously inputted owner email address).
callOwner()	This function is similar to the previous function, however security breach can lead to static phone call notification to the owner's phone number.
restoreSchedule()	This function is activated only if violation was detected. Its role is to restore the schedule of the devices exactly as it was before the violation occurred.

Class: Stats	The role of the Stats class is to provide the overall state of the devices in the house within specified zone. This class is very critical to the owner and plays as a tool to view stored information such as scheduling times, and device ID numbers, and device settings.
displayStats()	This function is to display and reveal a device's information to the owner of the system. Such information consists of status of a device, scheduled or not, also if it was scheduled, the start and end time and date for that scheduling.
getStats()	This function gets the information needed for the previous function. Its information is pulled out of the Storage class and its data member values.
validateStats()	This function services the idea of security as well by validating all device information before displaying it to the user. This function exists to ensure the pace of time in which a device's status was modified within a short time more than once.

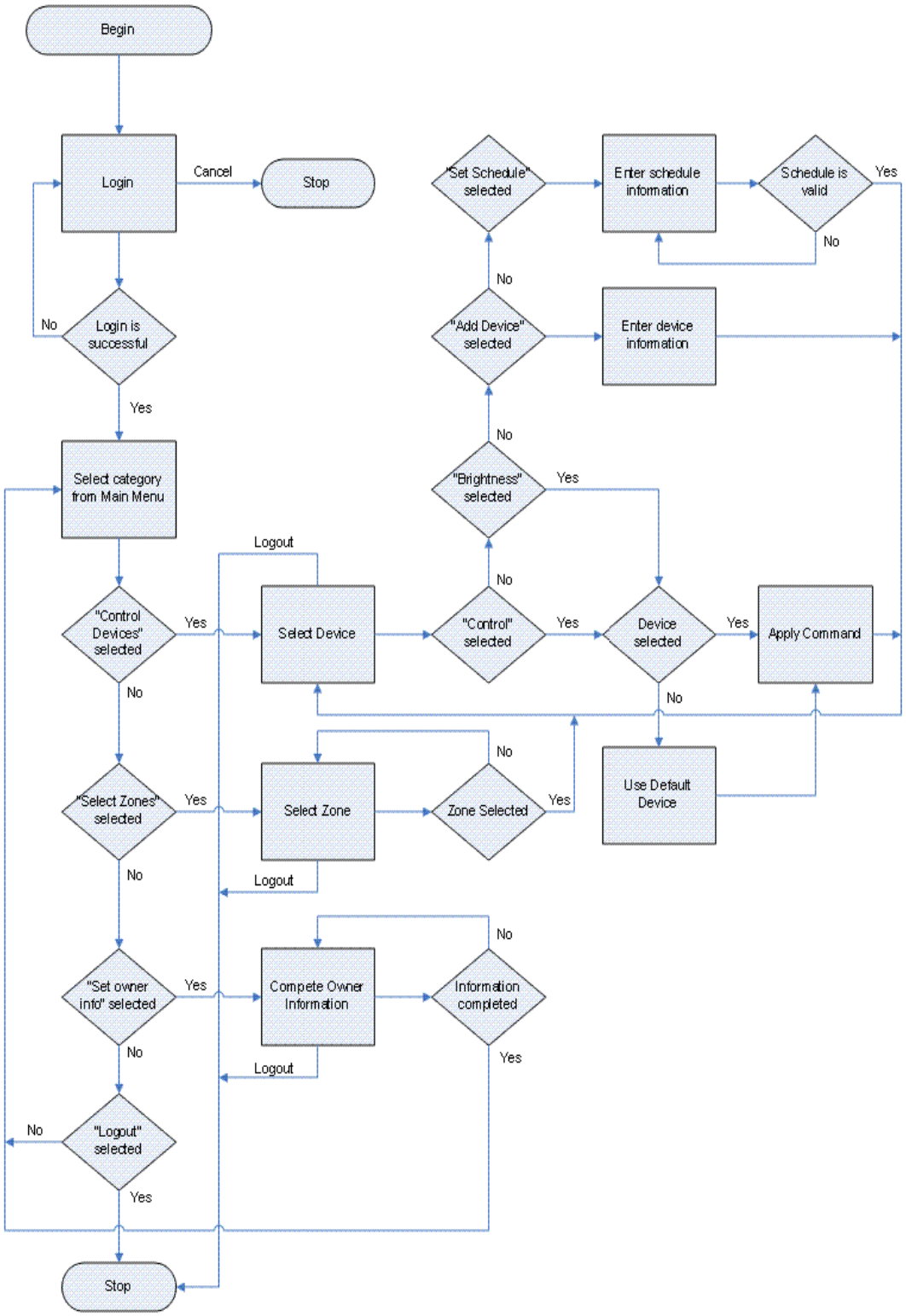
Detailed Design

Pseudocode

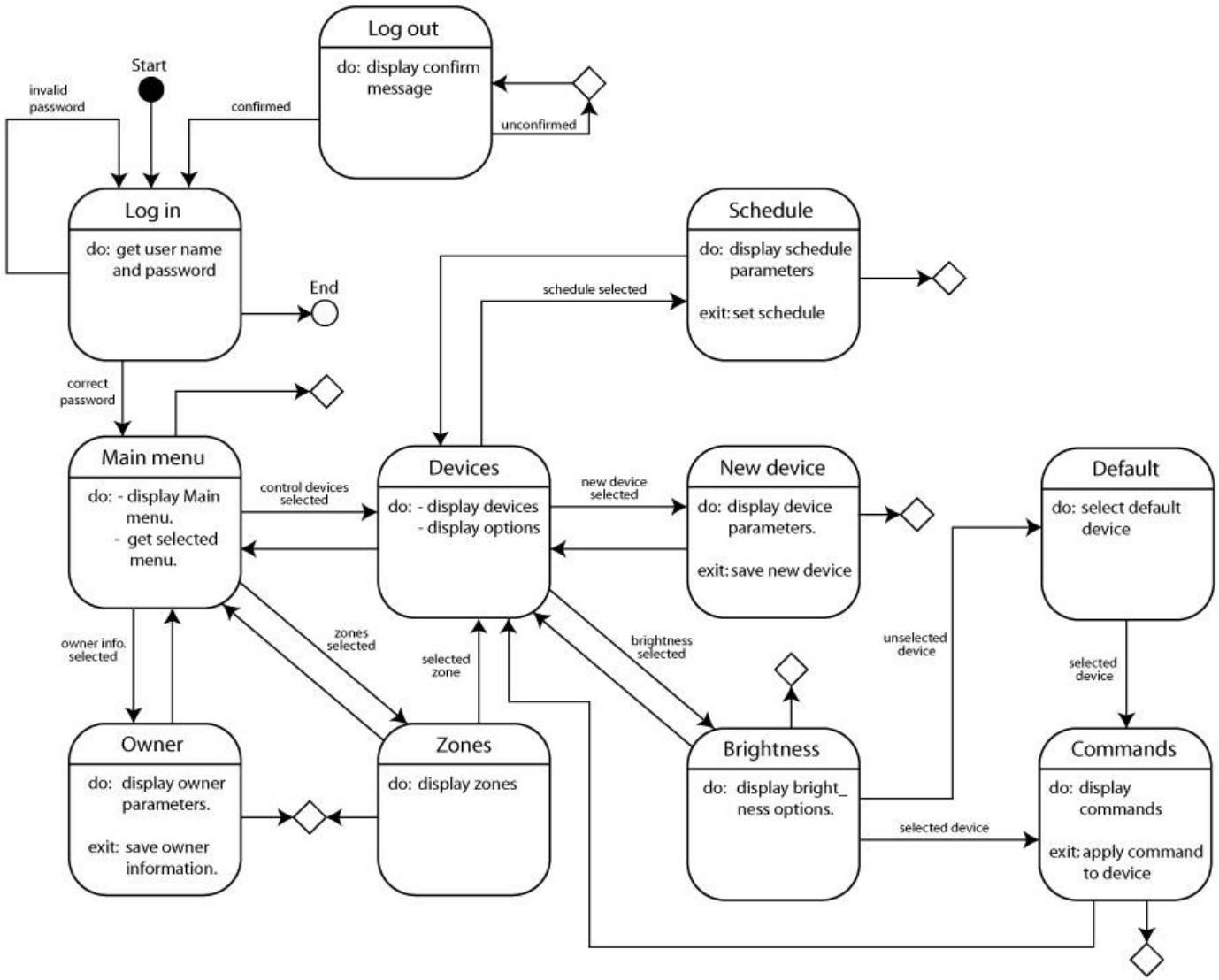
```
verify login information
if correct
        proceed to Main Menu
else prompt for user input
        if user wants to use generic login
                proceed to Main Menu
        else return to login screen
wait for user input from Main Menu
Case 1: "Control Devices" button selected
        open new form for controlling devices
        if previous devices were saved
                populate device box
        wait for control selection from user
        verify device selection
        if no device selected
                use default device ID (A1)
        else pass device ID to control function
        if Add device function selected
                open new form for add device
                wait for user input
                if 'OK' button selected
                        verify information has been entered
                        if information is good
                                save device information to file
                                return to control devices form
                                populate device box
                        else prompt user and return to add device form
                if 'Cancel' button selected
                        return to control device form
        if Brightness function selected
                apply brightness command (dim/brighten) to device ID
                return to control devices menu
        if Add device function selected
                open new form
                enter device information
                save information to file
                return to control devices menu
        if set schedule function selected
                enter schedule information
                if information is valid
                        return to control devices menu
                else prompt user and return to schedule input
        if logout selected
                exit program
Case 2:  "Select Zones" button selected
        open new form for select zones
        wait for selection of zone from user
        if zone is selected
                open form for control devices
                query file for zone information
                if zone information is present in file
                        populate device column
```

```
        else wait for input
Case 3: “Set Owner Information” button selected
        open new form for owner information
        complete information
        if ‘OK’ button selected
                if owner information completed
                        save information to file
                        return to Main Menu
                else prompt user and return to owner information form
        if ‘Cancel’ button selected
                return to Main Menu form
Case 4: “Logout” button selected
        exit program
```

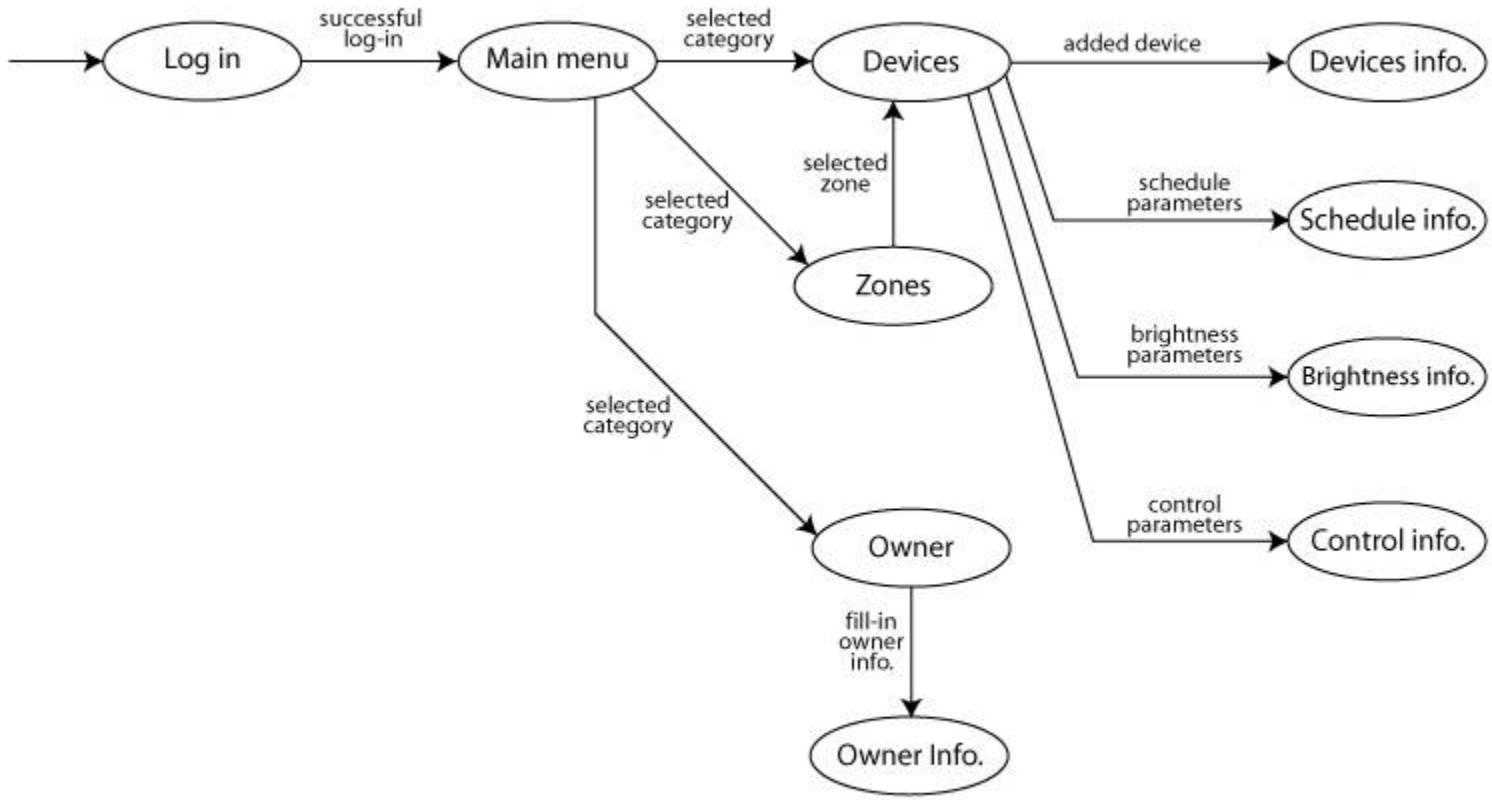
Flowchart


Begin
Login
Cancel
Stop
Login is successful
No
Yes
Select category from Main Menu
"Control Devices" selected
Yes
No
Select Device
Logout
"Select Zones" selected
Yes
No
Select Zone
Zone Selected
No
Yes
Logout
"Set owner info" selected
Yes
No
Compete Owner Information
Information completed
No
Yes
Logout
"Logout" selected
No
Yes
Stop
"Set Schedule" selected
Enter schedule information
Schedule is valid
Yes
No
"Add Device" selected
No
Enter device information
"Brightness" selected
No
Yes
"Control" selected
No
Yes
Device selected
Yes
No
Apply Command
Use Default Device

State chart


Log out
do: display confirm message
Start
invalid password
confirmed
unconfirmed
Log in
do: get user name and password
End
Schedule
do: display schedule parameters
exit: set schedule
schedule selected
correct password
Main menu
do: - display Main menu.
- get selected menu.
control devices selected
Devices
do: - display devices
- display options
new device selected
New device
do: display device parameters.
exit: save new device
Default
do: select default device
owner info. selected
zones selected
selected zone
brightness selected
unselected device
selected device
Owner
do: display owner parameters.
exit: save owner information.
Zones
do: display zones
Brightness
do: display bright_ ness options.
selected device
Commands
do: display commands
exit: apply command to device

Data-flow diagram



User Interface Design

Figure 1 shows the initial login box the user will be presented with upon execution of the software. Logins will not be mandatory; however, will be needed to use advanced features in future revisions of the software.

Figure 1: Login Box

Figure 2 shows the main menu of the software program. Once past the login screen the user will be presented with interactive dialog. Three main portals are contained within the main menu to help make a more intuitive interface.

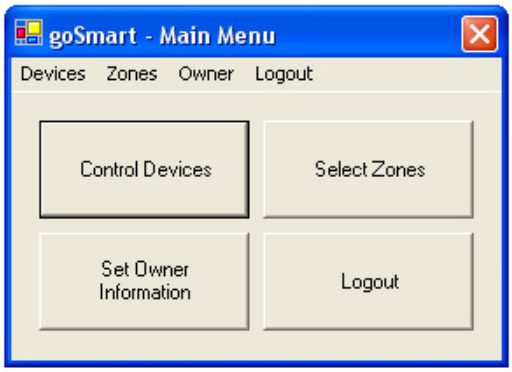


Figure 2: Main Menu

Figure 3 details the Select Zones section of the software. If the user selects this option from the Main Menu they will be taken here to select zones for controlling. Initially, pre-defined zones will be provided with the idea that future revisions will provide more interactive features for zone creation and control.

Figure 3: Select Zones

Figure 4 outlines the Control Devices section of the software. If the user selects this option from the Main Menu they will be presented with this interface. This interface provides a quick and easy interface for controlling various devices.

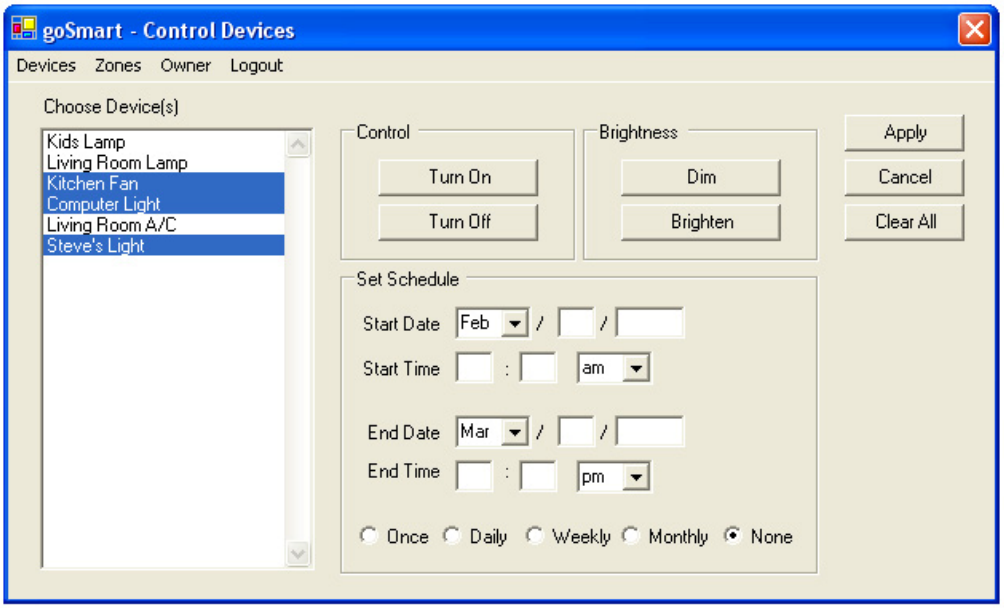


Figure 4: Control Devices

Figure 5 is a representation of the interface that will be used for adding a device to a selected zone. The interface will address such fields as Device ID (Unit/House code

assigned to device) [see Glossary], Name, Location, and Description of the device. This interface will also be used as the editing interface for devices.

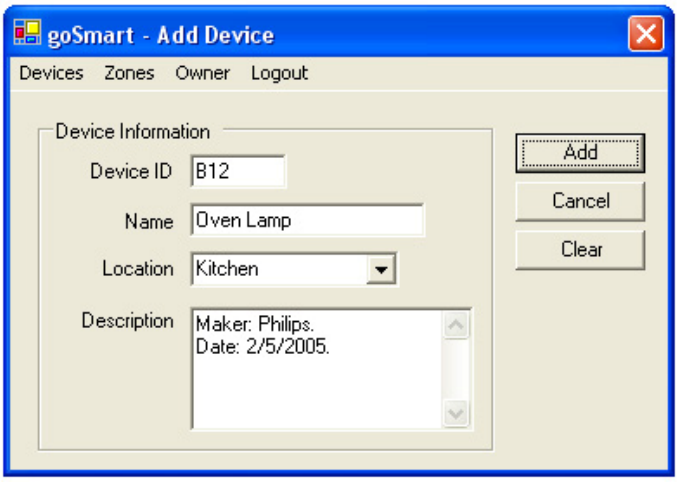


Figure 5: Add Device

Figure 6 is the default owner (user) profile editing interface. Here the home owner can save some information related to authentication, like username, password, email address and cell phone number.

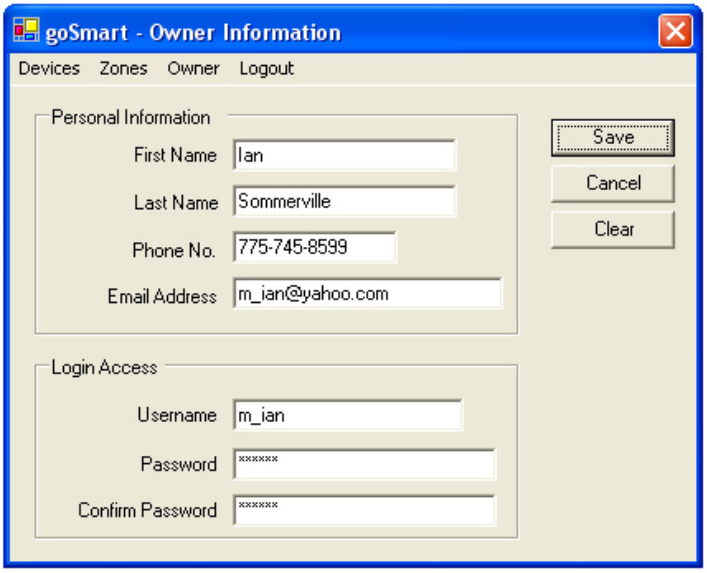


Figure 6: Add User

Figure 7 is a drop down menu of goSmart main menu interface. It allows the user to select the main features of goSmart. Notice that this drop down menu appears in all sub windows except the login and error windows.

Figure 7: Drop Down Menu

Figure 8 is a typical warning message that goSmart generates upon useless manual operations by the user. The warning in this case reminds the user that turn on operations have no effect if a device is already on.

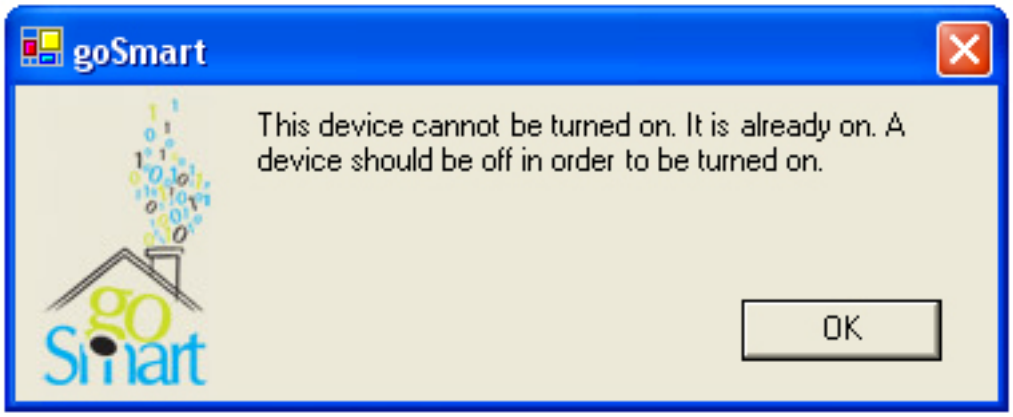


Figure 8: Warning Message

Figure 9 is a typical error message that goSmart generates upon invalid input by the user. In this case the user is trying to access the system with an invalid password. This is an error message indicating the error and how to recover (i.e. a follow up login dialog box will be opened automatically if the user clicks “OK” on this error message.

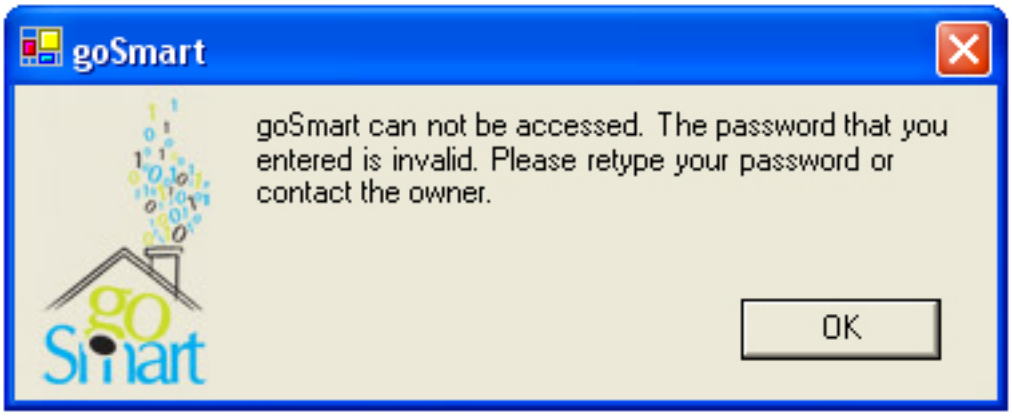


Figure 9: Error Message

Glossary

Automation **-** The act of using computers to control machinery and processes.

COM Port **–** The DOS name of the serial ports on PC's.

Device **-** A device generally corresponds to the model of a physical device, for example, a receiver. A device can also be a peripheral that can transfer data to and from a computer.

Firecracker **-** Firecracker is really just a wireless transmitter.

Function Buttons **–** This refers to the actual buttons the user will see on the user interface.

goSmart **-** The name of the proposed software to be developed.

goSmart Zones **–** These are pre-defined areas within a home (i.e. Bedroom, Living Room, Kitchen, Etc.). These zones will be initially pre-defined; however, will become more interactive in later revisions.

House Code **–** Part of a code used to identify the x10 device. The House Code consists of a letter between A – P. Each House Code has an associated Unit Code to accompany it. These codes are user defined and programmed on the hardware itself.

Interface **-** Method or piece of equipment for interconnecting units or systems which may not be directly compatible.

JFMS **-** The name of the team that proposed and will be developing the goSmart software. The composition of the team name is nothing but the abbreviation of the team members' names.

Packetizing Commands **–** This refers to the process of creating packets to be sent out on the wireless transmitter. These packets must be created and assembled in the correct order if they are to be affective.

Phase Bridges **–** Phase bridges are devices used to connect both phases of electricity entering into residential homes. Typically, there are two phases of electricity that enter into a residential home that can be bridged in order to give a shorter path between the two. This can also be accomplished by running appliances that operate on 220v.

Smart Homes **-** The ability of a user to remotely or automatically control lights around their house. The definition can be extended to adding security to a home by means of using the technology involved in remotely controlling devices.

SMS – Abbreviation for Short Message Service it is the transmission of short text messages. Messages must be no longer than 160 alpha-numeric characters and contain no images or graphics.

Test Bed – For this project, refers to a reliable source of electrical wiring for testing purposes. Typically, home electrical power consists of two phases which do not bridge each other. This can cause unexpected results for software testing.

Wireless **-** Radio-based systems that allow transmission of information without a physical connection, opposed to transmission systems, which require a physical connection, such as copper wire or optical fiber.

Unit Code **–** Second part of a code used to identify the x10 device. The Unit Code consists of a number from 1-16 and together with the House Code provides a unique ID for an x10 device. These codes are user defined and programmed on the hardware itself.

Web Portal **-** A web site that provides a starting point or gateway to other resources on the internet or an intranet.

x10 **–** A communications "language" that allows compatible products to talk to each other using the existing electrical wiring in the home. [www.smarthome.com/about_x10.html]

x10 Coding Scheme **–** Format used to uniquely identify an x10 device. See House Code and Unit Code for further detail.

Zones **-** The term zone is applied to a certain division within the house. Each zone consists of a number of devices configured with x10.

References

1. http://www.nomad.ee/micros/x10faq.html. Oct. 19, 2005.

   This article is a simple FAQ. There are many useful questions answered within the article that address the use and operation of x10 technology. This article is a good starting point for anyone interested in x10 technology who wants to get an idea of how the technology works and what they can do with it. However the FAQ is also a helpful reference to experienced users and developers of x10 technology. For example, too many drawbacks and unexpected results that might occur using the wireless approach are explained within some of the answers. More over, a lot of

technical issues are explained in detail in the FAQ which give a step by step approach to solve such issues in the process.

2. http://www.smarthome.com/about_x10.html. Oct.20, 2005.
   This is a general overview article regarding x10 technology. It reviews products available on the market that are utilizing x10 technology.
3. http://www.x10.com/home.html Oct. 28, 2005.
4. http://www.x10pro.com/ Oct. 28, 2005.
5. http://www.hackaday.com Oct. 28, 2005.
6. http://www.x10ideas.com/ Nov. 1, 2005
7. http://www.letsautomate.com/ Nov. 1, 2005.
8. Andy L. Jackson, **Integrating The Smart Home & Its Owner**: Books 1 And 2. Integratorpro. 2003.
   This book is a pseudo abstraction for home automation. It helps to understand how x10 programming works. It is helpful for the development of goSmart because of its simplicity and guidance. Modularity is illustrated by practical examples from real life, which also refers to some hardware parts that are needed for any home automation system. The approach taken by the author over emphasizes how a user can save time and money to accomplish an automated system in the house. This book was specific to our project since it deals with the security aspects of x10 programming which is a fundamental aspect of goSmart. A special chapter of this book is considered to be the start point of a comprehensive explanation of how x10 programming works.
9. Gordon Meyer, **Smart Home Hacks**. O'Reilly Media, Inc. First edition. 2004.
   Smart Home Hacks offers integrated solutions to enhance safety, comfort, and convenience in a house. The book is built upon tips and tools for automating your house. It illustrates the use of the x10 programming throughout numbered "hacks". Smart Home Hacks was very helpful in illustrating the idea of automation and control by clear and concise examples. It's worth to mention that the meaning of "hacks" in this book has a positive connotation. Smart Home Hacks is full of sample user interfaces that inspired us to develop a very concise and custom interface of our own for goSmart. Some of the hacks in this book get very complex,

however we found them very intuitive and in fact they helped us come up with our own original ideas.

10. James Gerhart, McGraw, **Home Automation & Wiring**. Hill-TAB Electronics. First edition. 1999.

    Home Automation and Wiring is a comprehensive guide for installation and maintenance of control (automation) systems using wiring. It contains concrete architecture for any generic automated system used. Understanding the use and implementation of x10 programming in automation via wiring, will make it easy for us to grasp the new concepts of wireless communication throughout the development of goSmart. One chapter of this book is very related in its concept to out project, in particular it examines and investigates the wireless remote control in automating homes. It talks about the general idea of transmission types and navigation systems. Reading this small chapter of this book was necessary to our project due to the fact that goSmart builds upon wireless communications.

11. Technica Pacifica, **Easy X10 Projects for Creating A Smart Home**. Indy-Tech Publishing. 2005.

    Easy x10 Projects for Creating a Smart Home is a guideline that assists you with the design and installation of all related x10 devices. We found this book very useful for our project because it goes step by step with coupled diagrams of the physical installation of x10 components. More over, this book gives a thorough description of each device and how it could be expanded to reach a comprehensive automated system in the house. Similar to other resources that we have encountered in this project, this book examines the use of x10 programming with wiring to accomplish a smart home environment. Never the less, our project uses a wireless approach, the samples in the book were still very useful.

12. Thomas Leonik, **Home Automation Basics - Practical Applications Using Visual Basic 6**, Prompt DPI - 8/01. First edition. 2000.

    This book shows you how to use Visual Basic 6 in a home monitoring system. Even though we are using Visual C++ in implementing goSmart, we found this book very helpful in explaining the programming logic of automation. The natural structure of Visual Basic 6 syntax, made this book an important resource for developing

goSmart. One of the advantages of this book is that it illustrates the idea of communication with serial ports. Most if not all, of the examples in this book, have different concepts than the idea that goSmart entails. However structure of the provided pseudo code for some of application used in the book, is easy to adapt and integrate into any new idea of automation.

Contributions:

	Title Page & Logo:	(Muhanna)
0.	*Table of Contents:*	(Fares)
1.	*Introduction:*	(Fares and Jim)
2.	*High level & medium level design:*	(Fares, Siraj, and Jim)
3.	*Detailed design:*	(Muhanna and Jim)
4.	*User interface design:*	(Muhanna)
5.	*Glossary updates:*	(Jim)
6.	*List of References:*	(Fares and Siraj)
7.	*Completed Checklist:*	(Fares and Siraj)
8.	*Contributions:*	(Muhanna, Jim, Fares and Siraj)

Appendix A – Checklist

Defect Checklist Template for Technical Artifacts	
Team: JFMS	
Course: CS425	
Project Part: III	
Note: The notation <X/Y>, for example <18/24>, indicates that X applies to a team of 3 students, while Y applies to a team of 4 students.	For each check, answer YES or NO

Design	Requirements Version no.	Requirements Version no.	Internal reviews added here	Requirements Version no.	Requirements Version no.	External reviews added here
	Review date:	Review date:	[optional]	Review date:	Review date:	[optional]
	Internal Review no.	Internal Review no.		External Review no.	External Review no.	

The Design Document...						
…has a cover page…	**YES**					
…with appropriate information on university, department, course, project title, project part, team name, team members, instructor, and date.	**YES**					
…has all pages numbered.	**YES**					
…has a table of contents…	**YES**					
…that shows page numbers for all sections and subsections.	**YES**					
…has an Introduction.	**YES**					
…that consists of between 400 and 800 words.	**YES**					
…that gives a precise account of progress made since the submission of the Specification report.	**YES**					

…that indicates changes/refinements to the project.	**YES**					
…that indicates the current status of the project, as a whole.	**YES**					

High- and Medium-Level Design…						
…is represented by at least one system-level diagram that…	**YES**					
…reflects and illustrates the system's organizational structure ...	**YES**					
…is formatted as either…						
…a context model.	**YES**					
…a block diagram.	**NO**					
…a behavioral model.	**NO**					
…a data model.	**NO**					
…another system-level model.	**NO**					
…is accompanied by a description.	**YES**					
…is represented in terms of program units which…	**YES**					
…are represented graphically by either…						
…object-oriented units (e.g. classes)…	**YES**					
…that number 7 or greater.	**YES**					
…that show…	**YES**					
…attributes.	**YES**					
…operations.	**YES**					
…relationships.	**YES**					
…multiplicity constraints.	**YES**					
…that are accompanied by a description.	**YES**					
…that include brief descriptions of, in total, <15/20> operations.	**YES**					
…abstracted, non-object-oriented program units (e.g. modules, functions, procedures, subroutines, etc.)…	**NO**					
…that number at least <12/16>.	**NO**					
…that illustrate the organization of the system.	**NO**					
…each of which consists of…						

…an appropriate name.	**NO**					
…a description.	**NO**					
…an indication of the higher-level unit (parent).	**NO**					
…inputs.	**NO**					
…outputs.	**NO**					
…other units utilized.	**NO**					
…exceptions/interrupts.	**NO**					
…additional comments (optional).	**NO**					
…details the structure of database tables used (if any)...	**NO**					
…by indicating the fields.	**NO**					
…by indicating the primary key(s).	**NO**					
…graphically representing the structure of the table columns.	**NO**					

Detailed Design…						
…reflects low-level design of the system.	**YES**					
…consists of <3/4> examples which…	**YES**					
…are each accompanied by a description.	**YES**					
…must be represented in at least two of the following ways…	**YES**					
…pseudocode.	**YES**					
…flowcharts.	**YES**					
…statecharts.						
…data-flow diagrams.						
User Interface Design…						
…is represented by <6/9> snapshots which…	**YES**					
…are each accompanied by a description.	**YES**					
…illustrate report/statistics reports, if applicable.	**YES**					
…illustrate user messages.	**YES**					
…illustrate a penultimate interface design with appropriate details.	**YES**					

References…						
…number <4/6> or more.	**YES**					
…are formatted using a standardized style (i.e. MLA, APA, etc.)	**YES**					
…each include a description of 100-200 words describing how the reference relates/contributes to the project.	**YES**					

The Design Checklist (this one!) …						
…has been fully completed with "Yes" or "No" responses to all entries	**YES**					
…has undergone at least one internal review.	**YES**					
…is included with the Design document.	**YES**					

Contributions of Team Members …						
…are included in a separate section of the Design document.	**YES**					
…list every team member…	**YES**					
…with each section/subsection to which they contributed.	**YES**					
…with each responsibility they shoulder for the development of the project.	**YES**					
…with any additional contributions they have made.	**NO**					

Glossary Updates (optional)…						
…include additions to the project glossary.	**YES**					