ASHESI UNIVERSITY COLLEGE

A SOFTWARE CONFIGURABLE AUTOMATION SYSTEM (SCAS)

BY

JOSEPH OFORI AMEGATCHER

Dissertation submitted to the Department of Computer Science,
Ashesi University College. In partial fulfilment of
Bachelor's of Science degree in Computer Science

April 2010

Declaration

I hereby declare that the dissertation is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Name:……………………………………………………………………………………

Candidate's Signature:……………………………………………………………………………

Date:…………………………………………………………………………………………………

I hereby declare that the preparation and the presentation of the dissertation were supervised in accordance with the guidelines on supervision of dissertation laid down by Ashesi University

Supervisor's Name:…………………………………………………………………………………..

Supervisor's Signature:………………………………………………………………………….

Date:………………………………………………………………………………………………..

Acknowledgements

I am first of all grateful to my Heavenly Father, without whom none of this would have been possible and my parents and siblings who have always believed in me.

To everyone who has encouraged and critiqued this work I would like take this opportunity to say a big thank you for supporting me. To my all my friends whose high expectations pushed me to work harder I am most grateful.

My deepest gratitude also goes to all the faculty of Ashesi's Computer Science department whom I have badgered with many questions, thank you for being the most supportive and helpful faculty alive.

Special thanks got o Dr. Nathan Amanquah who supervised this project.

Abstract

Along with a proliferation of electronic gadgets and devices for the home and office, the past few years has seen a steady increase in the degree of automation of devices in the home or office. A number of opportunities have been provided by commercial products as well as self developed systems that incorporate a means of controlling and/or monitoring devices. However many of the solutions provided are either expensive or limited in their application. This project is concerned with the design of a flexible and low cost automation system that provides a platform upon which various automation systems can be built. It consists of hardware to interface devices to a home server which a PC is running the Windows Operating System. A desktop application which provides a Graphical User Interface to set up and configure the connected devices is developed, as well as software for a mobile device that can be used as a remote controller for the system.

Table of Contents

Table of Figures

Chapter 1

1.1 Introduction

Over the past half century Automation has developed from an obscure term coined to describe the substitution of human effort with mechanical power into a whole field of study on in its own right. In the general sense it is a means of “performing a process by means of programmed commands combined with automatic feedback control.”[1] Although most people think about assembly lines and huge robotic arms when automation is mentioned it deals which much more. Especially now with the exponential growth in processing power, microcontroller technology and miniaturization advancements ,automation has found its way into many other non industrial areas such as the home and office.

A typical home or office usually contains several devices ranging from basic lamps all the way to sophisticated ventilation systems. However, for the most part, the proliferation of devices (even the more sophisticated ones) has not always led to increased automation of devices. This scope of this project is limited to the kind of automation that occurs in the home and office, the primary definition of home automation which is used in this paper is, the introduction of technology into the home for the control and monitoring of various home appliances by a single system in order enhance the quality of life [2, 3]. More and more the distinction between automation and what is termed ubiquitous computing is becoming more blurred however it must be noted that while both have the aim of enhancing quality of life through the use of technology in the home and office, at the heart of Ubiquitous Computing concept of many low cost computing elements working in concert this is not necessarily the case for home automation [4].

1.2 Relevance of Project

Most people have had the experience of having to rush back to their homes because they forgot to turn off a device, or perhaps have returned home late to meet the their home in total darkness and have wished to find a way to turn on their lights before actually getting home. Ultimately with more technological advancement the possibility of instructing devices to execute different behaviours in the home or office has become more appealing and relevant in several ways.

1.2.1 Convenience

Having the ability to control devices without necessarily being next to the device physically is a huge convenience with many implications. The ubiquitous television remote control is a good example of the convenience remote controls afford us. Many people cannot imagine using a television without a remote controller. In a home or office with many devices a system that provides a means to remotely control devices would serves as a way to save time since it would allow for multiple devices possibly located at different places to be turned on or off from a distance. Further gains in convenience can be made by allowing devices states to be pre programmed such that a device will come on or go off at the stipulated time without requiring human intervention.

1.2.2. Energy-use /monitoring or conservation

The automation features of such a system also provide economic benefits like energy conservation. By automating devices it is possible to accurately know and determine their times of operation and length of operation. These two factors translate directly into energy conservation. The automaton system can be used for instance to set quotas that minimize energy consumption and it can also monitor devices and prevent thus prevent careless use of device such as turning off devices at times when everyone in the home or office is known to be out.

1.2.3. Security

In terms of security it is possible to use the automation system as burglar deterrent system that simulates human presence by turning lights on and off at particular intervals, switching on the television sets and other appliances. Here there is also the possibility of configuring the system to work intelligently by giving video feedback or communicating with home owners and security officials when an intrusion is detected.

1.2.4. Other Applications

Since the proposed system is not a particular automation system but a platform to rapidly develop varying automation systems its relevance goes beyond the home and office(though these are the main target areas).This feature of being able to deploy multiple automation systems merely by configuring the system is a major. It provides a means of reuse and scalability since the most of the configuration needed would be done in the software and not the hardware. So without adding any new equipment or making changes to the physical system a new automation system can be set up.

The first three points have been centred on the home or office and are typical of most other systems. But there is relevance even in agriculture. The system could be used to configure a set of equipment for example sprinklers, humidity detectors to form an intelligent irrigation system. A simple scenario being that the sprinklers would be configured to go off in the morning and the evening with the humidity sensor constantly monitoring and setting off the sprinklers when necessary to supplement the regular watering cycle.

1.3. Objectives and Project Overview

The objectives of this project are:

1. To utilize a low cost means of interfacing electronic devices and appliances to a personal computer in order to create a centralized generic platform for controlling connected devices.
2. To create supporting applications to provide easy to use Graphical User Interfaces (GUI) for the configuration and control of connected devices to produce desired automated behaviour.
3. To allow for the remote control of devices in the automation system.

This is not a novel thing as will be seen from the review of the literature and similar projects however, a key factor that must be kept in mind is this project's constraints in keeping the costs of the solution minimal by using cheap, readily available technology and utilizing redundant components.

The proposed solution consists of a home server which is a PC. This home server runs a Java desktop application and hosts a database which primarily stores device information. The application provides a user-friendly GUI for managing and configuring the behaviour of the devices that have been connected via the parallel port. It also gives access to a subset of its functions via a Java Micro Edition (J2ME) Application; Internet access is made possible by a dynamic DNS service that provides a domain name tied to the homes server.

The rest of this document is organized as follows; Chapter 2 deals with previous work done. Chapter 3 presents the design for the proposed solution incorporating justifications design decisions Chapter 4 focuses on the actual implementation of the solution, while Chapter 5 contains conclusions as well as limitations and further work.

Chapter 2

2. Previous work

Many of the concepts of home automation and remote control involved in this project (as has already been acknowledged) are not novel hence, there is a substantial amount of literature available especially from academic sources [2].This review however will focus on the work that is closest in nature. The two predominant design issues include centralization through a computer versus some other microcontroller board as well as remote control via wireless technologies or the Internet.

Werf Gui and Xu in [3] tackle the issue of remote control by proposing a mobile-based home automation system that consists of a mobile phone with Java capabilities, a cellular modem, and a home server. The user could issue commands from the phone using a Java application running on it. The commands were forwarded via Short Messaging System (SMS) to the modem which in turn forwarded the commands to a microcontroller board acting as the home server. The home server was responsible for operating and monitoring the connected appliances. The emphasis on that particular project was the elimination of the need for a PC as part of the architecture. The project was also a build up on earlier work which had relied on telephone as the interface for control; this explains the introduction of Java capable phones for the interaction and the use of a microcontroller board as the home server. Another critical design choice from that project was the decision to avoid using the Internet as a means of connection and instead to use SMS commands to the modem; this was done to avoid the need for the home server to be an Internet enabled device or PC.

Gill Lang and Lu [2] present an architecture that takes a slightly different approach based on the ZigBee standard. Their solution combined the Wi-Fi network with the ZigBee automation system which works via power lines and radio frequency. It tried to the address the issues presented by integrating the Internet at the core of the system as well the need to eliminate installation of intrusive wiring. Bluetooth could have been used but in terms of range and cost with increases in devices, the ZigBee system was a better option. Remote control was possible either over the Internet or the Wi-Fi network. A home gateway was developed to link the Wi-Fi network with the ZigBee system and to provide a common interface to all devices. This particular design made provision for more ways to access the system this included the ZigBee remote control.To provide an extra layer of security a virtual home which pre-processed all communications before they were realized on the actual system was implemented.

The first two approaches avoided the use of a PC, however Al-Ali and AL-Rousan[5] developed a Java based solution that could monitor and control home appliances connected to a PC based server with an integrated embedded system. In this solution the devices were connected via the system board which itself was connected to the PC through the parallel port. Also, unlike [2, 3] Al-Ali and Al-Rousan embrace the Internet as a means to provide remote control. In this system Java Server Pages and JavaBeans were used to develop the server and control was possible via a web browser from any location or the system board itself.

Zhang, Sun and Zhou [6] also propose a Java based solution, this system also involved the use of an embedded controller (this time connected via the serial port). The significant difference between [5] and [6] was the use of a dynamic DNS. Al-Ali and Al-Rousan's approach required the Home server to have a static public address by which the system could be reached which is quite expensive. In [6] instead of requiring the user to obtain a static IP address to allow remote users to access the home server, a dynamic DNS service was used to link a fixed domain to the homer server's dynamically generated IP address.

Ali, Nawaz and Jawad [4] implemented a home automation system with remote control features implemented with a PC as part of the architecture but completely without the Internet. The focal point of [4] was to research into the development and construction of a wireless real-time control system. As at the time it was published this project was the only one to have worked solely with General Packet Radio Service (GPRS) networks. Their system also included Bluetooth technology. As before there was a PC to which devices were connected using device controllers of their own making. Two phones connected via GPRS using the IP protocol acted as client and server, the server being connected to the PC via Bluetooth. Using the GPRS network was also a way of overcoming the need for static IP address as required by [6].

Finally,[7] presents a design closely related to this project, an office and home remote control system based on client/server architecture that interfaces with devices through the parallel port. It described a public server and a home server which link to a J2ME application for manipulating device states. However [7] focuses more on the J2ME application being a remote controller it also does not discuss in detail the manner by which the various servers communicate.

2.1. Differences in Approach /Implementation

In the reviewed projects and papers, emphasis was placed more on the way the devices were connected to each other and how to communicate with them. [8] goes as far as to define home automation in terms of networking. Also, most of the papers present ways or actually implemented systems that were concerned merely with turning devices of and on, but not much was spoken about scheduling the devices to produce specific behaviour. Therefore the major difference between this work and the previous work done is the focus on making it possible to not only change states but schedule the devices to work in concert, this serves as the foundation for the concept of developing a platform for deploying different automated applications rather than a fixed automation system.

To achieve the objective of low cost the solution avoided the use of high-end microcontroller boards or a Field Programmable array (FFPGA) and rather relies on readily available components, in fact for the most part any PC with Windows 98 could be used to implement the system.

Also, another focus of this work is derived from the objective of making it configurable though an easy to use software. There is therefore a heavy bias on the software aspects of the system rather than the hardware aspects. Another issue that differentiates it is the emphasis placed on usability of the software and the quality of user interaction it provides. These issues affected the implementation of the project and are discussed further in the next chapter.

Chapter 3

3. System Design Issues and Decisions

The two predominant design issues that came up from the previous work was (1) the use of a microcontroller board as against a PC as the home server as well as (2) the means of providing remote access to the system. Each approach has its merits and demerits. This chapter focuses on the choices that were made for this project and supplies reasons why the choices were made.

3.1 General Conceptual View of the proposed System

Home Server

For the proposed system a PC was chosen as the home server. Using the PC as the home server has a number of advantages; firstly that decision is in line with the objective of using low cost and readily available components. Most of the target users of this system are likely to already own a PC so there is there is no need to buy any additional microcontroller chipsets in order to implement the system. Also the PC allows more flexibility. Especially since this project requires configuration, the PC provides a platform to deploy the configuration system complete with Graphical User Interfaces easily. Thirdly the PC's modular architecture allows for more possibilities of extending the system either by software or hardware and this makes it an ideal choice [8].

Remote Access

For the proposed system the means of providing remote access to the system is via a combination of the Internet as a 'network' and an Internet ready mobile device. By using the Internet, there once again is a reduction in deployment costs because there would be no need to acquire wireless devices such as Bluetooth controllers or ZigBee remotes. By using the Internet it also means that automatically the range of the system is almost global, as long as the home server is online and the user has access to the Internet, the system can be used. The challenge of requiring a static public address for the home server was addressed by subscribing to a dynamic DNS service similar to the one in [6].This design choice does mean that the users' home server must always have a connection to the Internet.

The figure below shows the conceptual view of the proposed system as discussed above. This conceptual design can be broken down further to show more details concerning the various blocks. The rest of the chapter deals with details of the various blocks.

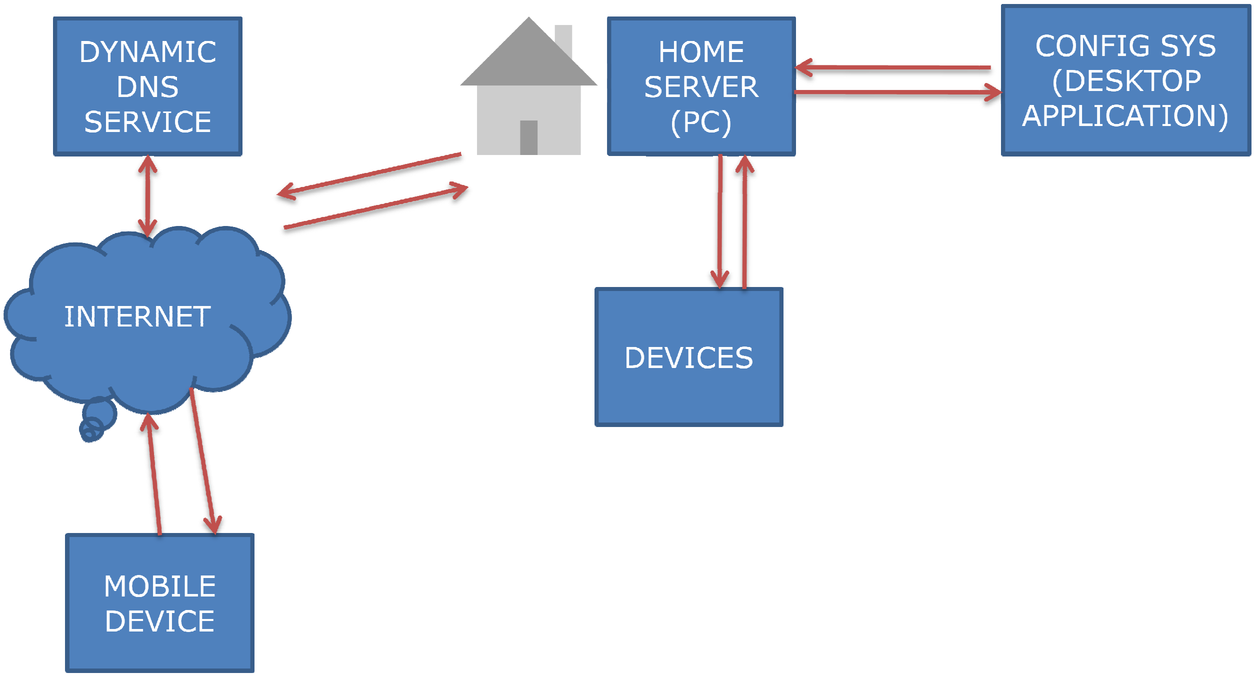


Figure 1 Conceptual view of Proposed System

3.2 Detailed description of sub-components

3.2.1. Home Server

The home server was implemented using a PC with Windows operating System (a Linux distribution such as Ubuntu could also have been used however Windows Operating System has a larger market share and hence wider user base).The server consisted of a desktop application that provided a user interface for configuring the entire system and managing devices as well as a database to store the device information such as the state (whether on or off) at any point in time.

The desktop application was developed using Java because

1. It is easy to rapidly develop beautiful GUI using Java especially in conjunction with the NetBeans IDE. This was one of the emphases of the project
2. Java is lightweight and platform independent making it possible when necessary to easily deploy the application on various operating systems where the Java platform is available.
3. Considering that a mobile application would be developed it would be possible to easily integrate the desktop application as well as the mobile application. This is more of an advantage from the perspective of software development.
4. Java is designed from ground up for network applications.

However, it must be noted that not all the components of the desktop application was developed in Java. Firstly a security feature of later Windows versions(those based on the NT Kernel) prevents direct access to the ports, and hence it was necessary to find a means to circumvent it in order to communicate directly with port. This is where the choice of Linux as the operating system would have been beneficial because Linux allows direct access to the ports. This notwithstanding, a free dynamically linked library (DLL) downloadable from the Internet was used as a solution. The DLL known as inpout circumvents the windows restrictions and provides C functions that can be used to write to the port. The C code was called by the Java application by utilizing Java Native Interfaces (JNI) which provides way of calling source code written in other languages other than Java.

The database was implemented with MySQL, MySQL was chosen because:

1. MySQL provides the basic features required to implement the system. It allows remote connections and has a decent response time.
2. Additionally, no advanced features where used in the design of the database and therefore only a basic DBMS was needed. The nature of the application (a home application with fixed users) also means that there will not be many records or huge amounts of traffic to the database server
3. Finally, keeping in mind development costs MySQL has a free version which is the one being used.

3.2.2. Interfacing devices to the Home Server

The parallel port, a 25 pin D-shaped female connector found at the back of the computer, was chosen as the means to interface the devices to the server. The original purpose of the parallel port was to connect printers to the PC however with the introduction of higher speed and more sophisticated ports. The parallel ports main function has become redundant and in fact the port is largely fading out. However the design of the parallel port makes it an ideal means of interfacing for the purposes of this project. The port was used as an entry point for devices being connected to the system. Newer ports such as the fire wire and USB have physical port heads which cannot be plugged into directly, however this is not a critical factor since a male connector can be modified to connect devices. The main issue is the complexity of the protocols used in such high speed ports.

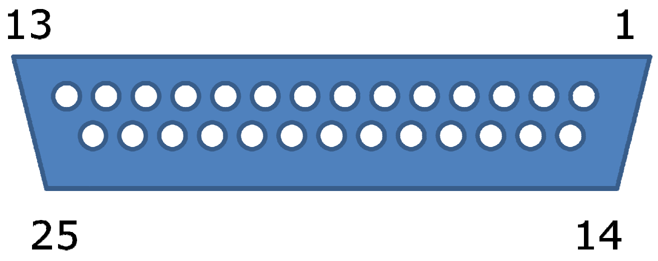


Figure 2 Parallel Port

The 25 pins are grouped into 4 categories data, control, status and ground pins From a programming viewpoint the control status and data ports are mapped to registers as which can be written to, setting the voltages high or low. For this project the data pins (pin 2-9) which are mapped to the data register at address Ox378 are the ones we are concerned with.

3.2.3. Data and Addressing Scheme

Another design issue to consider was the means of representing data being sent between the hardware components and software. Considering only the 8 data pins means that the data sent between the PC and the devices is at most 8-bits long. The 8 bits were broken were used for different purposes [9, 10]. A possible scheme was to use one bit to represent the device state whereas the remaining bits would represent addresses to select a device or any other information that would be required.

Theoretically, such an arrangement makes it possible to control 2^7 (128) devices (i.e. if all 7 pins left are used for addressing) this arrangement however requires the inclusion of some other components.

First of all a de-multiplexer is necessary. A de-multiplexer is a device that accepts a single line as input and via selectors chooses one of its output lines [11]. So here a de-multiplexer with 7 selectors is required[1].

Another component that is necessary for this approach is a flip-flop. A flip-flop in the most basic sense is a logic device capable of acting as a 1-bit memory. The flip-flopss purpose in this arrangement is to persist the data value sent to any particular line such that if another line is selected the value that was last set is not lost [11].There are a number of different kinds of flip-flops, namely Set-Reset (SR), Toggle (T), Delay (D) and Jump-Key (JK).The JK which can be considered a universal flip-flop was chosen for the design. It has two data inputs, J and K, a clock or chip select input and an output Q. The JK functions such that when both inputs ore 0 there is no change in the output when only K is 1 the output is reset to 0 when only J is 1 the output is set to 1 as well and when both are set to one the output is set to the logical complement of the last output.

As applied in the design both J and K inputs are assigned to separate bits this means that the 2 bits are used to represent data to making it possible to fine tune the commands to send to the devices at hardware level to an extent and providing more flexibility for development at software level. For instance

[1] De multiplexers with 7 selectors are not commercially available this means that a multiplexer with eight selectors or a combination of smaller multiplexers would be used.

with this arrangement it easy to implement commands to toggle the state of devices without much intervention from the software.

The final component needed is a relay, the relay is a device that is used for switching (it is needed irrespective of the addressing scheme).The voltage provided by the parallel port is not large enough to drive a mains level appliance hence the relay acts as a bridge between the parallel port voltage to switch on a bigger AC voltage however, the relay itself must be turned on by a transistor [12]. The relay could also offer a means of providing feed back to the circuit in order to know whether commands issued have actually been successful.

Figure 3 shows a generalised conceptualization of the PC-device interface circuit just described see while Figure 4 gives a more detailed circuit diagram.

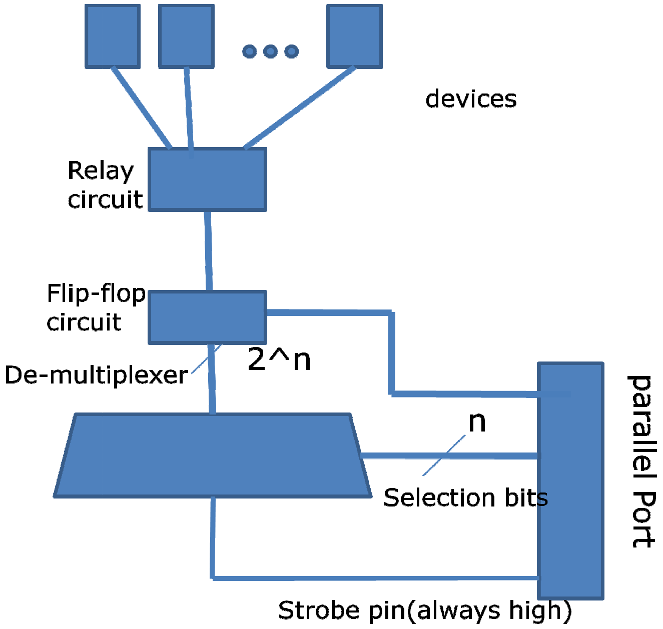


Figure 3 Conceptualization of PC-Device interface

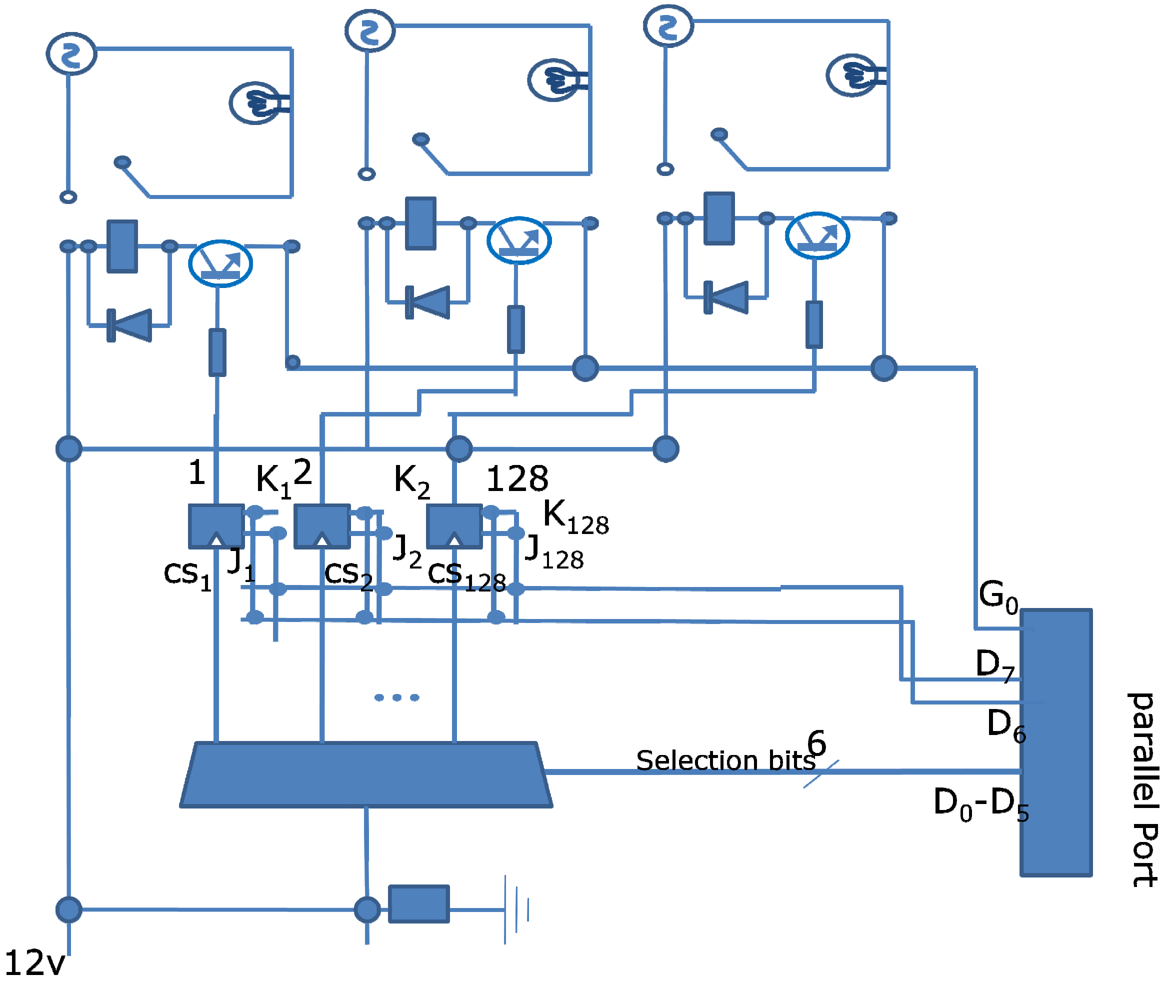


Figure 4 Detailed Circuit Diagram

3.3.4. Description of design of the PC-Interface Circuit

As shown above, the chip selects of the flip-flops are connected to the multiplexer's outputs whiles the de-multiplexer's selection inputs are connected to the parallel port pins D0 to D6.The input of the de-multiplexer is connected such that it remains high always. To turn on a device the appropriate address combined with the command 01(i.e. the command to set the J input high) is sent to the parallel port. The flip-flop connected to the device to be turned on would have its chips select set to 1 and its j input also set to 1 the output would then become 1 this signal would then be used by the relay to switch the AC circuit and turn on the attached device. To turn off the device the addressed combined with 01 would be sent instead.

3.2.4. Mobile Device

The mobile device is any Java enabled device, this could be a mobile phone or PDA, any handheld device that is capable of running Java applications would be able to give remote access to the system.

A J2ME device was used because of:

1. The decision to allow remote access via the Internet. The J2ME platform in conjunction with the NetBeans IDE allows for rapid development of Java phone applications which can easily connect to a web server using GPRS capabilities of the phone.
2. Although other mobile programming platforms and languages such as QT and Python currently the most common platform is Java this essential in making the application available to the widest audience.
3. Also the J2ME application would work seamlessly with the Java Desktop application.

Chapter 4

4. System Development and Implementation

System Development and Implementation can be organized into two sections, the hardware and software sections each made up of smaller subsystems. This chapter focuses on the things that were actually implemented and how they were implemented.

4.1. Hardware

4.1.1. Device-PC interface circuit

For this projects' prototyping purposes each of the 8 data pins were used to represent the state of a single device meaning that only 8 devices could be controlled. The De-multiplexer and flip-flops were not implemented in the prototype resulting in a much simpler circuit. Figure 5 shows the circuit that was prototyped, it was implemented using a reusable breadboard.

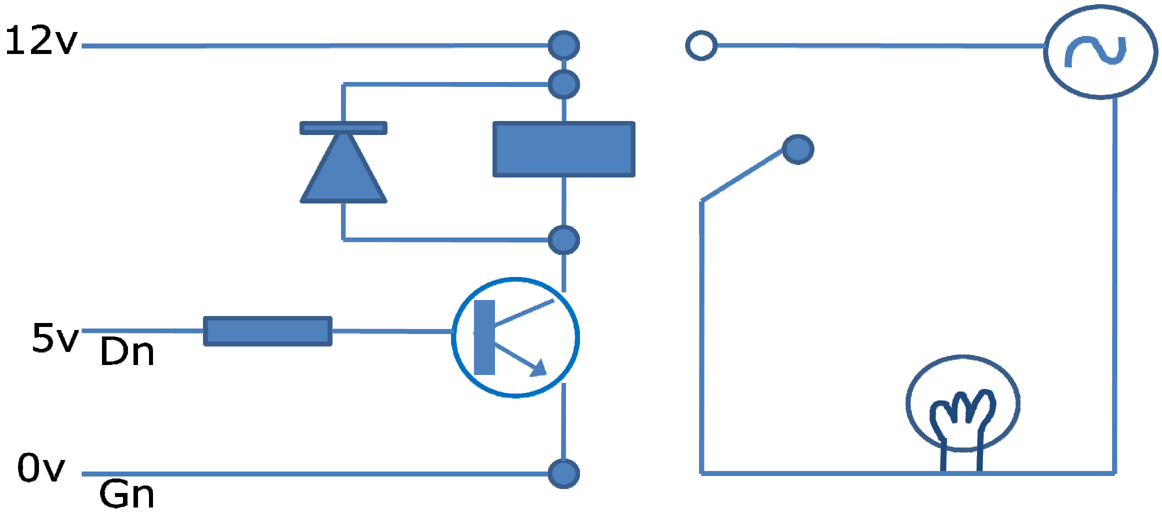


Figure 5 PC-Device Interface Circuit

As shown in the diagram, the PC-Device Interface circuit consists of the input from the parallel port Dn. It is a Transistor-Transistor Logic(TTL) level voltage theoretically ranging between 0 and 5v [12]. The current from the port turns on the transistor which switches on a relay that can drive mains level appliances. A diode was connected across the relay as a protective measure to safe guard the coil from spikes in voltage that could happen during discharge. The relay itself was powered by a twelve volt battery. When Dn, the parallel port pin goes high the transistor would turn on the relay which in turn would close the contacts and switch on the lamp on the mains circuit. This simple circuit formed the basis for all the device-pc connections.

4.2. Java Desktop Application

Requirements

Systems requirements are generated to focus the scope of a project and ensure certain minimum aspects for the software .The system requirements for the desktop component of the system are the following.

4.2.1 Functional requirements

1. The system shall be able to read and write to a port.
2. The system shall provide an interface to turn connected devices on and off.

3. The system shall provide a means of scheduling device activity.
4. The system shall store the state of devices in a manner which is accessible to other applications.

4.2.2. Non-Functional requirements

1. The system shall employ the parallel port.
2. The system shall store device state and information in a database.
3. The system shall clearly display the state of connected devices at all times.
4. The systems shall have an intuitive Graphical interface.
5. The system shall be developed in Java.

4.2.3. Developing the Java Desktop Application

In developing this component of the system an evolutionary model was used. This model was the most appropriate considering the size of the project and the fact that multiple components which interacted where being developed concurrently. Developing with this model as a basis meant that the application was went through several incremental steps as well as iterations. In terms of the development paradigm, Object Oriented paradigm was adopted. This paradigm represents systems in terms of objects and the association or relationships between them. Objects in this sense usually represent real-world objects but can sometimes represent abstract concepts. This design paradigm is very much in line with development using the Java language since object oriented language.

Firstly, a thorough analysis of the problem scenario was carried out to identify the objects involved. Three objects were identified Device, Event and Profile (see appendix for complete description in UML)

Device represents the devices or appliances that will be controlled by the system. A device has properties such as name and type and takes part in an event. A device object in the system would correspond to a real-world object such as a lamp.

Event represents a particular occurrence in the system it involves a device(s) in the system. An event has properties such as trigger which is what causes an event to take place. An event also has a type property which describes the nature of an event. An event is either a fixed event that occurs at a particular time or a relative event that occurs when another event has occurred. An event also has an action property; this is what is actually interpreted as the event whether to turn a device on or off or to reduce brightness.

A key issue that came up in the defining devices and events was about association with devices. In the initial modelling a device was the object that an event was associated with directly as such a device had an associated events property. However in re-analysing that relationship it was decided that an event rather should have the property associated devices. From this perspective a more natural view emerges. An event such as “turn off all lamps” would have all connected lamps as its associated devices and the occurrence of this event would affect all necessary devices at once. This is contrast with attaching the turn of Lamp event multiple times to the devices which would be the way to do it from the initial perspective. This change in model had direct consequences on the final object profile and also the design of the Graphical User Interface which is discussed in the next section.

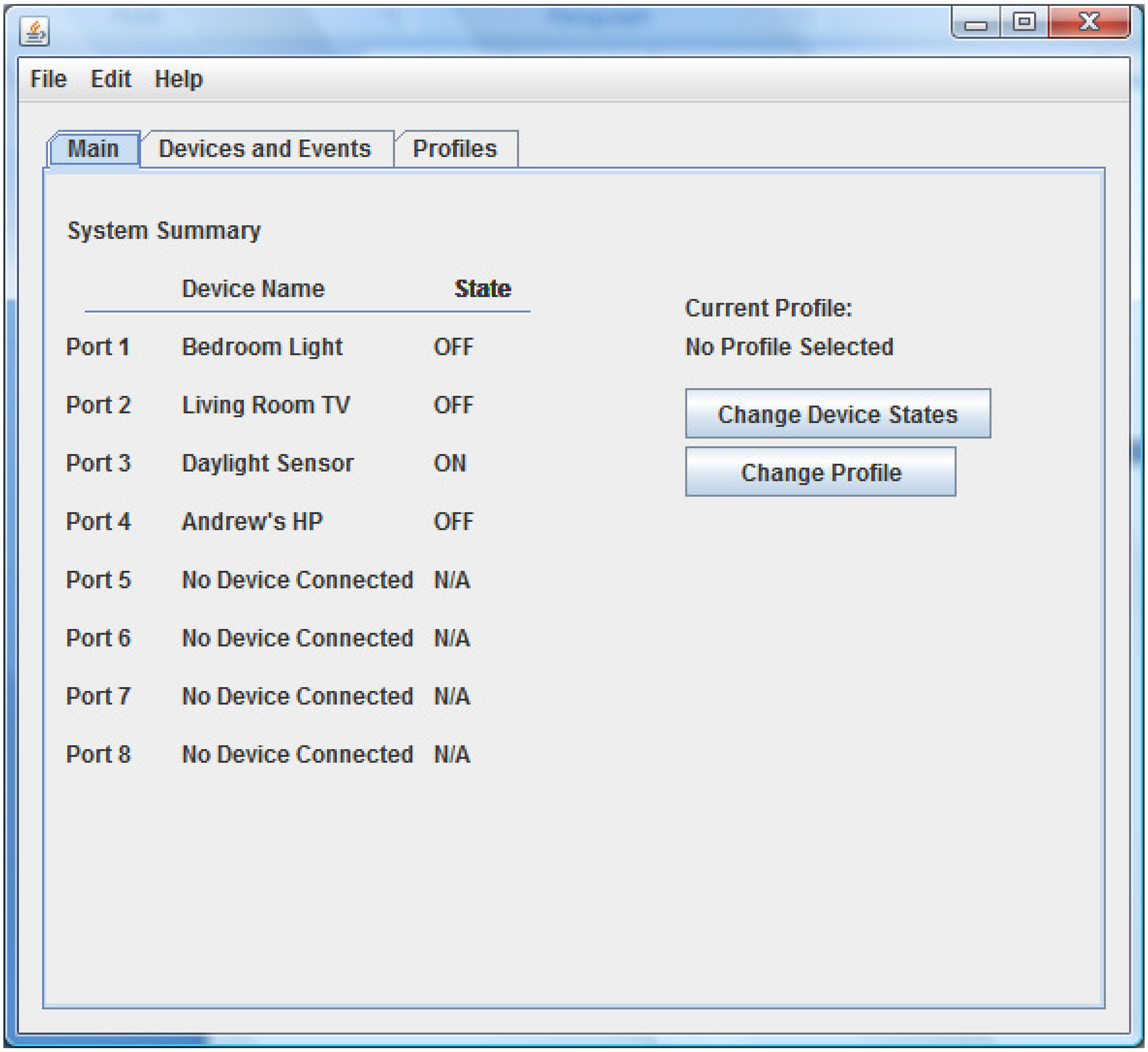
A profile is a collection of events(rather than a collection of devices) this forms the basis of different automation systems so by loading different profiles different sets of behaviours can be achieved from the system. For instance by loading a burglar deterrent profile the associated devices and events that have already been pre-programmed are loaded and the desired behaviour is implemented by the system.

4.2.4. Description of the Desktop Application

One of the objectives of this project was to provide software for the configuration that was easy to use as well as intuitive. This design objective was a crucial factor in designing the interactions. The application was implemented based on tabs which allow the user to work on logically grouped tasks. There was the Main tab, a tab for Devices and Events and a Tab for Profiles.

Main

This tab had a summary of the systems state. This is a snap shot of what is happening at any point in time. The tab displays all the devices that have been connected and their current state whether on or off .It also allows the user to know if a profile has been chosen to run. The main tab additionally had a "Change device states" to manually change the states of connected devices and a 'Switch Profile" button to change the current profile(see Figure 6).



System Summary

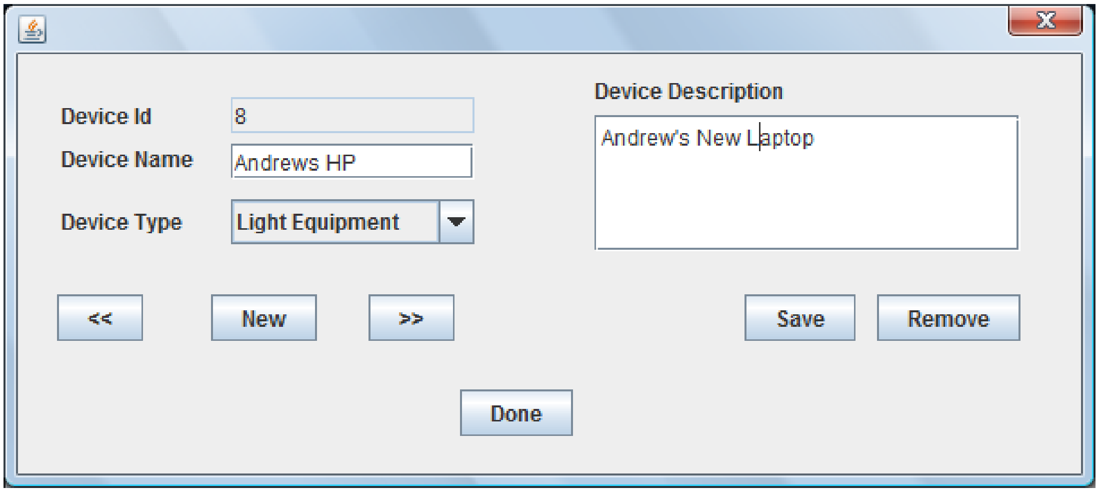
	Device Name	State
Port 1	Bedroom Light	OFF
Port 2	Living Room TV	OFF
Port 3	Daylight Sensor	ON
Port 4	Andrew's HP	OFF
Port 5	No Device Connected	N/A
Port 6	No Device Connected	N/A
Port 7	No Device Connected	N/A
Port 8	No Device Connected	N/A

Current Profile:
No Profile Selected

Figure 6 Main Tab (Desktop Application)

Devices and Events

This tab is where all issues concerned with devices and events are handled. It is basically made up of 3 buttons .The “Manage” button for Devices mange button for Events respectively and the “Map devices” button. When a user clicks the manage button for Devices a dialog screen is presented that allows the user to enter the name of the device, select a type and add a short description. The navigation buttons at the bottom of the dialog screen allows the user to browse through existing devices and to edit them. The “Save” button allows changes made to be saved whiles the “Done” button closes the popup. Any changes made without saving will be lost if the “Done” button is pressed (see Figure 7).



Device Id 8

Device Name Andrews HP

Device Type Light Equipment

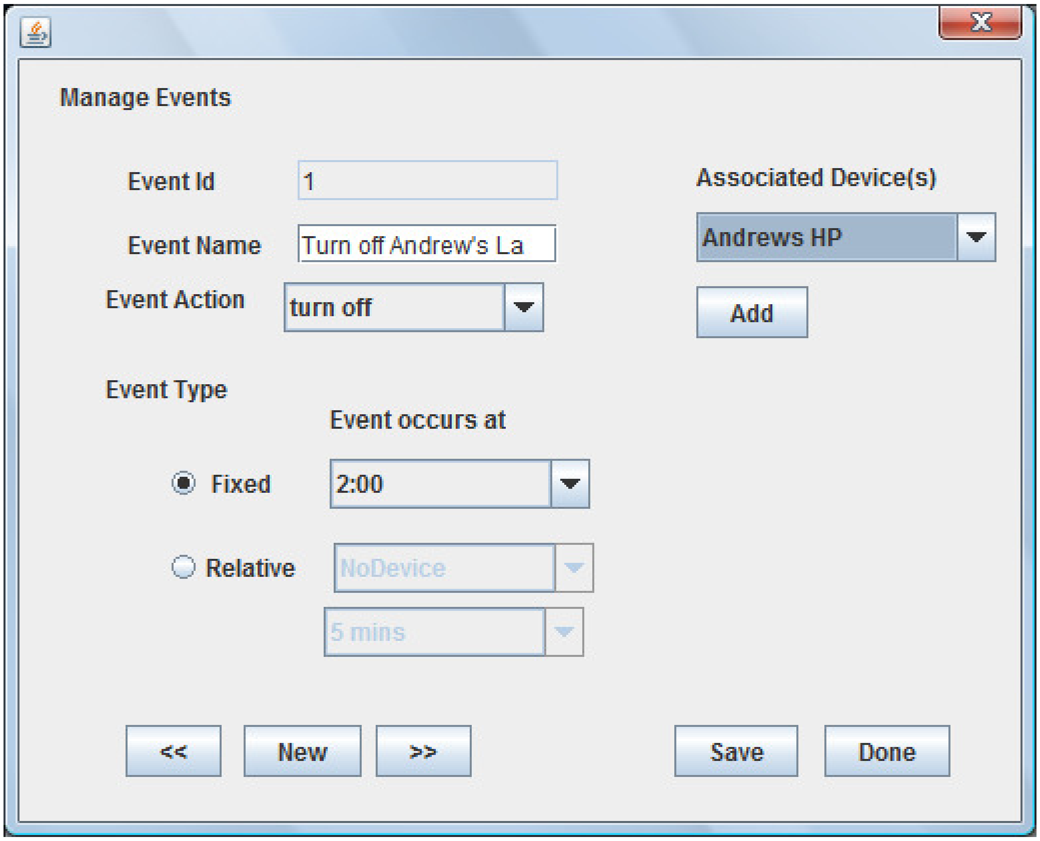
Device Description

Andrew's New Laptop

Figure 7 Device Creation

Likewise, when a user clicks the “manage” button for Events another dialog screen appears which similarly facilitates the creation and editing of events. In line with principles of consistency the “New”, “Save” and “Done ” buttons are also available in this dialog screen in similar positions as in the manage device dialog screen.

The “Map Devices” button also brings up a dialog screen where devices can be mapped to specific pins of the parallel port. This is more for the management purposes of the configuration software.



Manage Events

Event Id: 1

Event Name: Turn off Andrew's La

Event Action: turn off

Associated Device(s): Andrews HP

Event Type

Event occurs at

● Fixed: 2:00

○ Relative: NoDevice, 5 mins

Figure 8 Event Creation

Profiles

This tab, like the previous tabs is where all the functions of managing profiles are carried out. Creation and editing of profiles once again is consistent with creation and editing of devices and events. A variation was needed to represent the associated events. This was done by a table component which displays all events associated with a particular profile. ”Add” and “Remove” buttons are made available to edit the devices associated with a profile. The profiles tab like the main tab gives users and opportunity to select the profile that should be operating at a particular time.

4.2.5. Interaction Design Discussion

The Tab System was designed to correspond with objects and hence each object has a corresponding tab where instances of that object can be manipulated. Also in line with the objective to increase usability, descriptive labels are used wherever possible such that even a first timer to the application would be able to use it with ease. As much as possible the application tries to give the user feedback concerning actions taken such as creation and editing of devices, events or profiles. The Main Tab also keeps the user abreast with the devices and their states. To make sure it is up to date a thread was implemented that would check the database at regular intervals and update the interface with the most current data. To also enhance the user experience the tool tip functionality was utilised, this allowed hints for various buttons to pop up whenever the mouse is placed over them providing in-context help and reducing reliance on separate manuals.

4.3. The J2ME Mobile Application

4.3.1. Functional Requirements

1. The system shall allow users to change device states.
2. The system shall display current states of connected devices.
3. The system shall provide a means of changing profiles that are being run.

4.3.2. Non Functional Requirements

1. The system shall require user authentication.
2. The system shall employ the phones native User Interface.

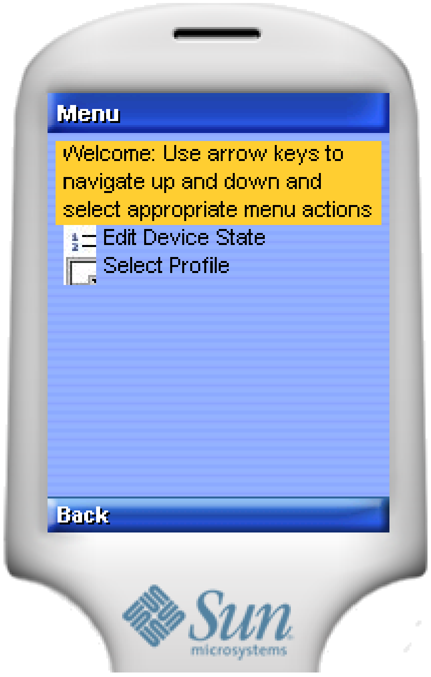
4.4. Developing the J2ME Mobile application

The mobile application presents a subset of the Java Desktop Applications functions. So it is inadvertently based on the assumptions and model created for the desktop application. The application was developed such that the J2ME client contained part of the logic but mostly the presentation of data where as a PHP script on the home server had the functions which were being called form the J2ME client via httpConnection. Here too PHP on its own cannot access the parallel port it is therefore necessary to call functions from the inpout DLL from the PHP script.

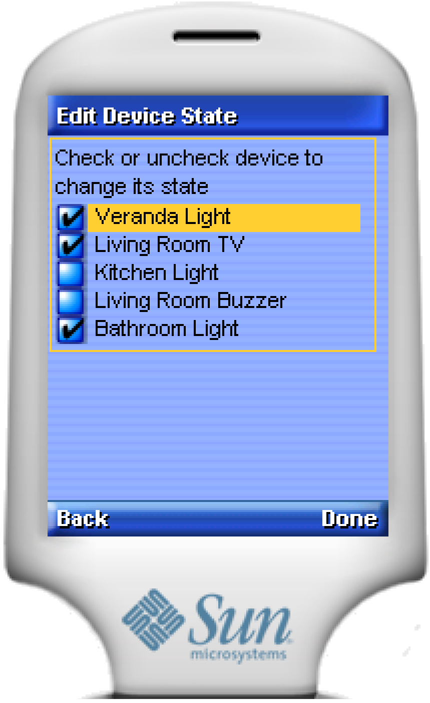
4.4.1. Description of the Mobile Application

On the first screen, a user is presented with is a login screen which is used for authentication purposes. This is important since remote access is being giving to the system it is necessary to authenticate all users. An unsuccessful login will keep the user on the first screen and give an opportunity to retry whereas a successful login will move the user to a menu implemented with the J2ME list objects. This menu consists of two options "Edit Device States" and "Select profile"(See Figure 9).

Figure 9 J2ME application Login screen



Welcome: Use arrow keys to navigate up and down and select appropriate menu actions



Check or uncheck device to change its state

☑ Veranda Light
☑ Living Room TV
☐ Kitchen Light
☐ Living Room Buzzer
☑ Bathroom Light

Figure 10 Editing Device States

When the user clicks on "Edit Device States" a new screen is presented which displays all connected devices alongside check boxes .Devices with unchecked boxes represent devices that have been turned off whiles the checked devices represent devices that have been turned on. The user has the option of changing states by checking or un-checking the boxes (see Figure 10).

The "Select Profile" option takes a user to a screen where the available profiles are displayed and any of them can be selected. The user is given the option to confirm the change.

The mobile application's implementation is purposely made very simple. It is not expected to do much more than just changing states of devices and switching profiles. Creation of devices and events has been limited to the desktop application because of the limitations of screen and inputs methods on most mobile devices. Additionally the simple nature of the mobile application means that it is very small in size and quite responsive.

4.5. Development of the Database

The database was developed using the entity-relational model which is similar in many ways to the object model. The entities created are similar to the objects mentioned earlier. Here too there was a Device entity an Event Entity a profile

Entity and a Port entity. In linking the desktop application to the database NetBeans automatically generated Object-Relational Classes these are not part of the discussion. In converting the entities to tables some of the entities became redundant or where changed to properties of other entities (See Appendix).The database is a crucial component since it serves as the means to persist all information concerning devices and makes it possible for the desktop application and the J2ME application to be in synchronization.

4.5.1. Tools Used

The NetBeans IDE was used for majority of the development both Desktop and Mobile applications were developed using the IDE which provided drag and drop capabilities for Interface design as well as code completion and syntax highlighting both of which significantly aid development

PhpMyAdmin and MySQL Browser were the tools used to create and manage the database; both have their strengths in making certain routine tasks such as editing of tables easy.

Chapter 5

5.1. Results and Limitations

The Desktop application was tested and it was possible to carry out the administrative tasks such as creation and editing of devices and events via the GUI. The application was able to identify and send commands to specific devices that had been created by modifying the database. The Mobile application was also able to access the databases and modify the states of devices as well as to select profiles. Sending commands to the parallel port was tested in Windows XP and Windows 98 however, from the test it appeared that he application was not able to get full control of the port. The PC-Device interface was also not implemented to work with a mains voltage, only Light Emitting Diodes (LEDs) where tested.

5.2. Further work

5.2.1. Enhancement of the PC-Device Interface Circuit

The current implementation of the data and addressing scheme is such that only 8 devices can be controlled simultaneously, this is one major limitation since the usefulness of the system and the ability to create complex automation systems rests on the number of devices that can be controlled. In the next implementation the de-multiplexer described could be implemented in order to allow the use of more than eight devices and to ensure that the states displayed by the software is always accurate. Still concerning the data and addressing scheme, this current design does not cater for setting devices in intermediate states as such a lamp cannot be dimmed or a fan's speed controlled. This can be included in the further work by the addition of an analogue-to-digital converter and the utilization of other free I/O pins in the parallel port to represent the data required.

Instead of using a mechanical relay a solid stare relay or opto-coupler would be a better choice to improve response time for switching and insure better isolation of the mains powered circuit. All these will go a long way to make the hardware more efficient especially as its complexity increases.

5.2.2. Extension of Remote Access

Currently remote access is possible only from the mobile application, and the mobile application has very limited functions. The Integration of a full blown web application as a companion to the desktop application making would make it possible to access the system from a mobile device or any other Internet enabled device. Although the mobile application would remain restricted for earlier mentioned reason the web application could make it possible to create new events and profiles remotely[2].

[2] Creation of devices is deliberately not allowed from outside the house since the devices.

5.3. Conclusion

Although the entire design was not prototyped and tested the partial results suggest that the design would be effective in achieving the project's objectives. The GUI was simple and easy to use without the need for much documentation on how to use it. This was one of the major objectives of the project also all the components used where available from local electronics vendors at a total cost not exceeding 100GHC which is very reasonable making it an affordable solution.

References

[1] Britannica Encyclopaedia, "Automation", [Online] Available:http://www.britannica.com/EBchecked/topic/44912/automation.

[2] K. Gill,S. Yang, F. Yao, and Xin Lu, "A ZigBee-Based Home Automation System", *IEEE Transactions on Consumer Electronics*, Vol. 55, No. 2, MAY 2009 422.

[3] M. Van der werff, X. Gui, and W.L. Xu, "A mobile-based home automation system", [Online] Available: IEEE Xplore [Accessed February 19,2010].

[4] U. Ali, S. J. Nawaz and Nazish Jawad, "A Real-time Control System for Home/Office appliances automation, from mobile device through GPRS network", [Online] Available: IEEE Xplore [Accessed February 19,2010].

[5] A. R. Al-Ali and M. AL-Rousan, "Java-Based Home Automation System", *IEEE Transactions on Consumer Electronics*, Vol. 50, No. 2, May 2004, pp.498-504.

[6] X. Zhang, J. Sun, L. Zhou , "Development of an Internet Home Automation System using Java and Dynamic DNS Service", *Proceedings of the Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'05)*.

[7] Z. Abubakr, "Home and Office Remote Control System: *Using distributed systems to control domestic and office devices remotely*", B.Sc, Ashesi University College,Accra,2008.

[8] I. Han, H. Park, Y. Jeong, and K. Park, "An Integrated Home Server for Communication, Broadcast Reception, and Home Automation", *IEEE Transactions on Consumer Electronics*, Vol. 52, No. 1, February 2006.

[9] M. J. Johnson and G. Subramanyani, "A parallel port interface circuit for computer control applications involving multiple stepper motors", University of Northern Iowa Cedar Falls.

[10] Z. Alkar and U. Buhur, "An Internet Based Wireless Home Automation System for Multifunctional Devices", [Online] Available: IEEE Xplore [Accessed April 8,2010].

[11] P. Horowitz and W. Hill, *The art of electronics,* 2nd ed. Cambridge: Cambridge University Press, 1995, 507-509.

[12] T. Engdahl. "Parallel port interfacing made easy: Simple circuits and programs to show how to use PC parallel port output capabilities", [Online] Available: http://www.epanorama.net/circuits/parallel_output.html [Accessed: February 19, 2010].

Appendix A

Complete System Architecture

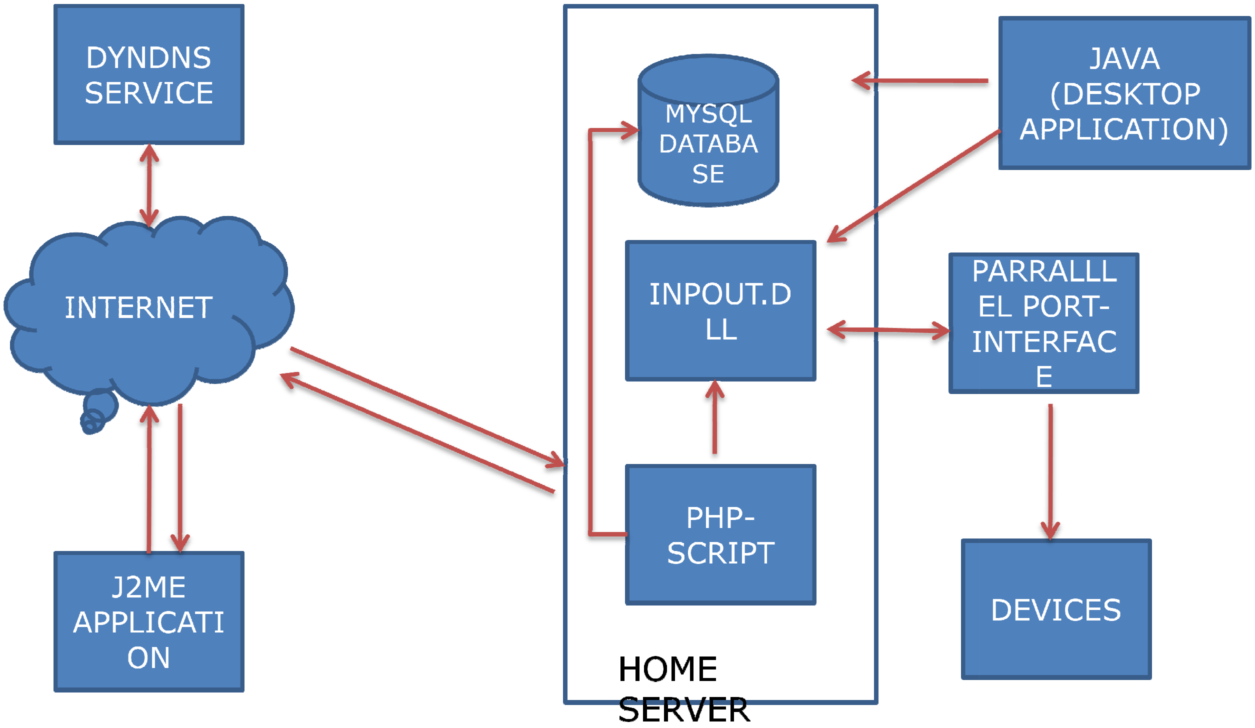


Figure 11 System Architecture

Appendix B

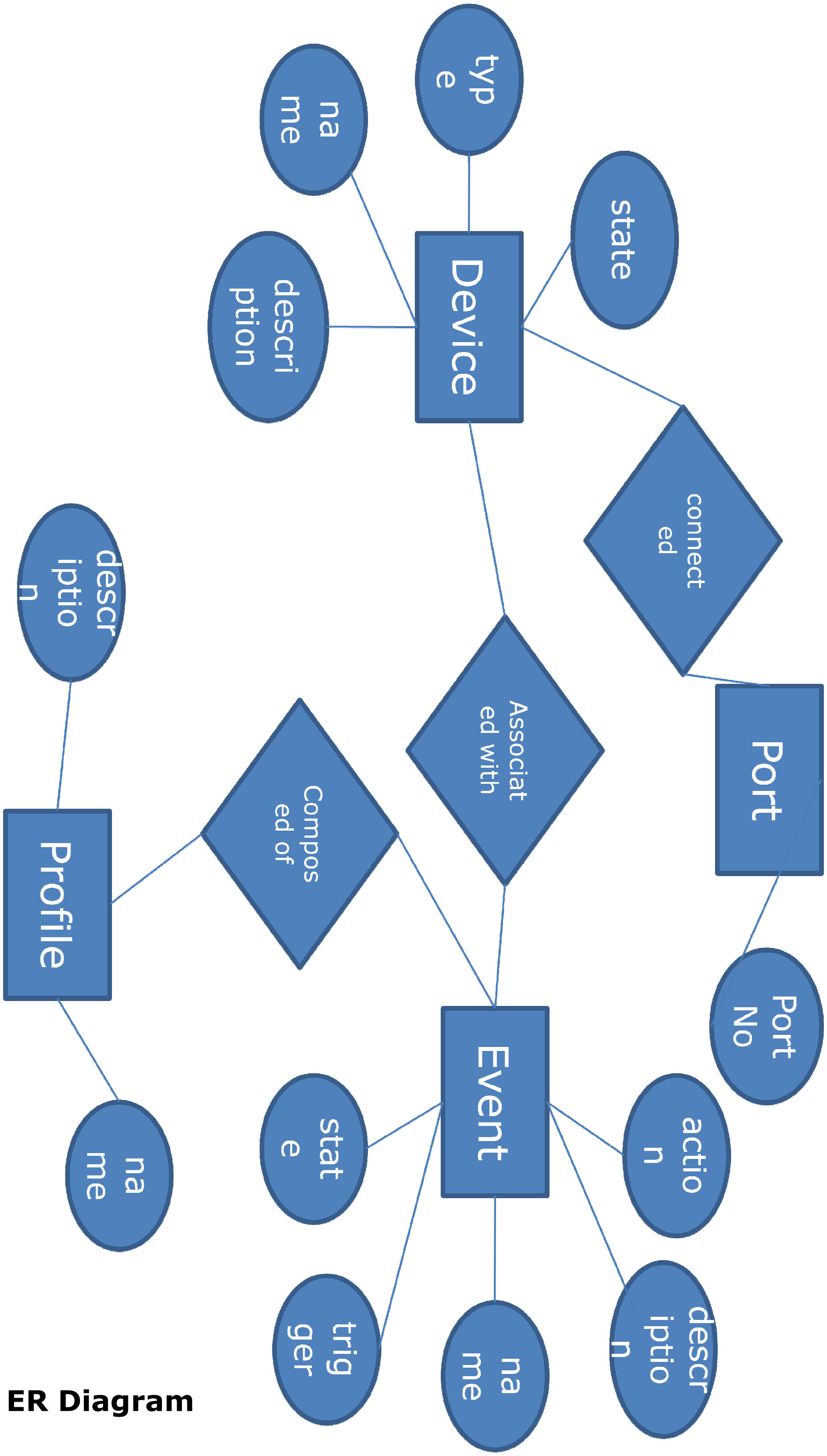
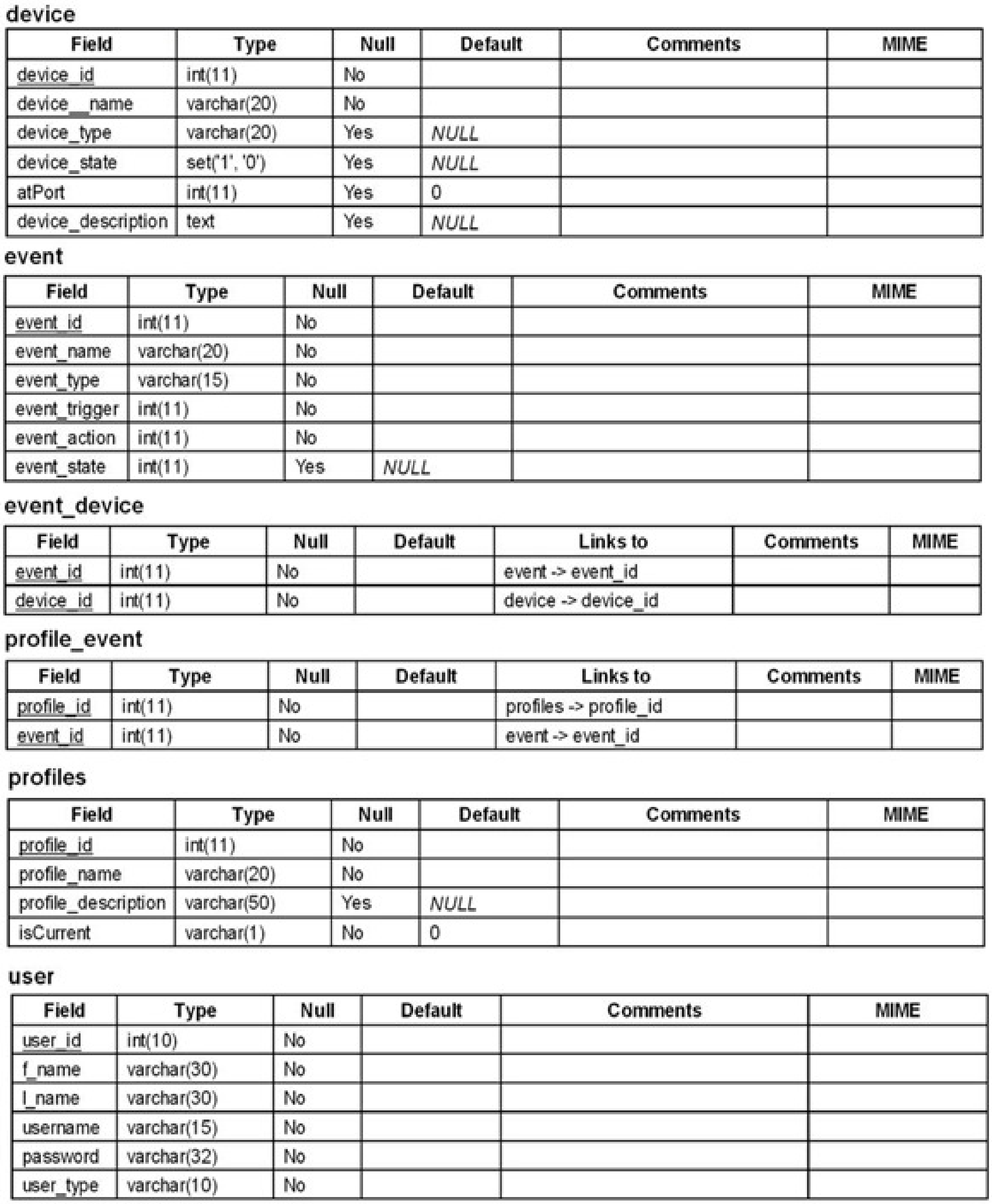


Figure 12 ER Diagram

Appendix C



device

Field	Type	Null	Default	Comments	MIME
device_id	int(11)	No			
device__name	varchar(20)	No			
device_type	varchar(20)	Yes	*NULL*		
device_state	set('1', '0')	Yes	*NULL*		
atPort	int(11)	Yes	0		
device_description	text	Yes	*NULL*		

event

Field	Type	Null	Default	Comments	MIME
event_id	int(11)	No			
event_name	varchar(20)	No			
event_type	varchar(15)	No			
event_trigger	int(11)	No			
event_action	int(11)	No			
event_state	int(11)	Yes	*NULL*		

event_device

Field	Type	Null	Default	Links to	Comments	MIME
event_id	int(11)	No		event -> event_id		
device_id	int(11)	No		device -> device_id		

profile_event

Field	Type	Null	Default	Links to	Comments	MIME
profile_id	int(11)	No		profiles -> profile_id		
event_id	int(11)	No		event -> event_id		

profiles

Field	Type	Null	Default	Comments	MIME
profile_id	int(11)	No			
profile_name	varchar(20)	No			
profile_description	varchar(50)	Yes	*NULL*		
isCurrent	varchar(1)	No	0		

user

Field	Type	Null	Default	Comments	MIME
user_id	int(10)	No			
f_name	varchar(30)	No			
l_name	varchar(30)	No			
username	varchar(15)	No			
password	varchar(32)	No			
user_type	varchar(10)	No			

Figure 13 Data Dictionary

Appendix D

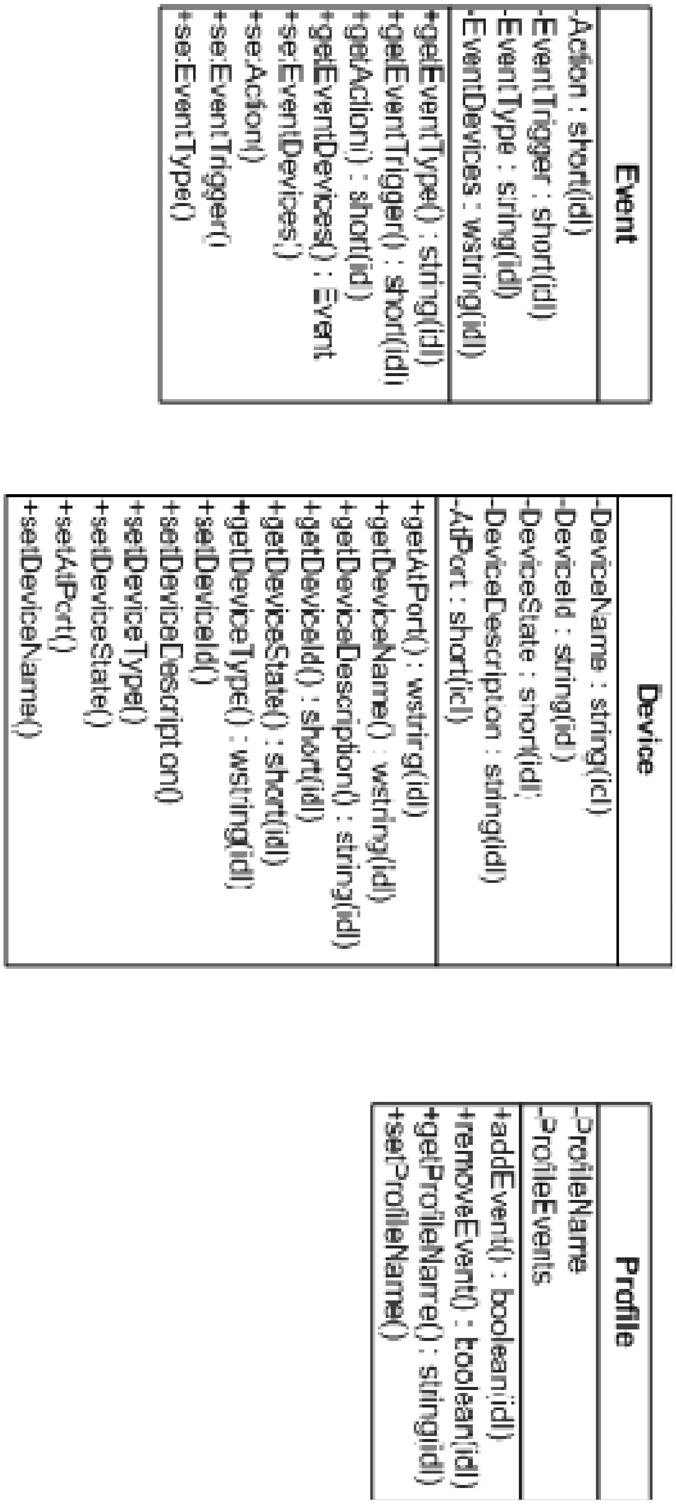
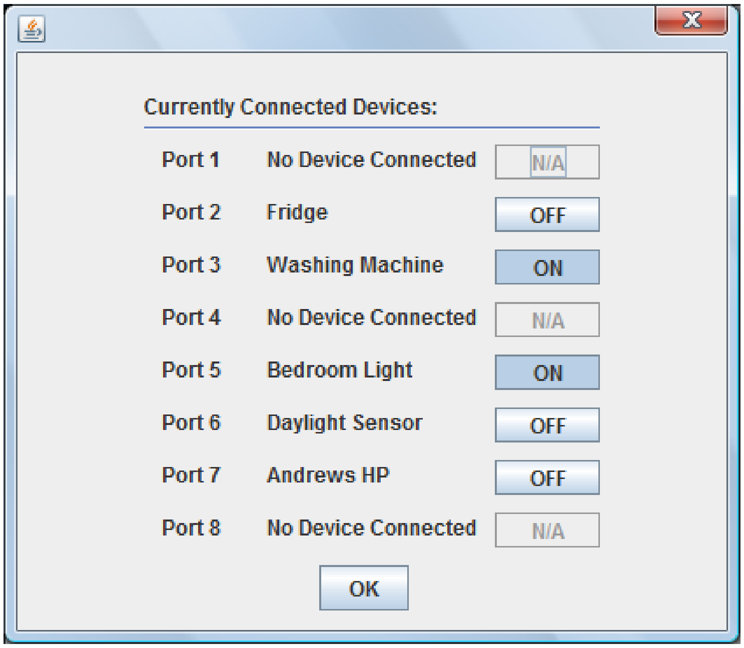

Event
-Action : short(idl)
-EventTrigger : short(idl)
-EventType : string(idl)
-EventDevices : wstring(idl)
+getEventType() : string(idl)
+getEventTrigger() : short(idl)
+getAction() : short(idl)
+getEventDevices() : Event
+setAction()
+setEventDevices()
+setEventTrigger()
+setEventType()
Device
-DeviceName : string(idl)
-DeviceId : string(idl)
-DeviceState : short(idl)
-DeviceDescription : string(idl)
-AtPort : short(idl)
+getAtPort() : wstring(idl)
+getDeviceName() : wstring(idl)
+getDeviceDescription() : string(idl)
+getDeviceId() : short(idl)
+getDeviceState() : short(idl)
+getDeviceType() : wstring(idl)
+setDeviceId()
+setDeviceDescription()
+setDeviceType()
+setDeviceState()
+setAtPort()
+setDeviceName()
Profile
-ProfileName
-ProfileEvents
+addEvent() : boolean(idl)
+removeEvent() : boolean(idl)
+getProfileName() : string(idl)
+setProfileName()

Figure 14 UML Class Diagrams

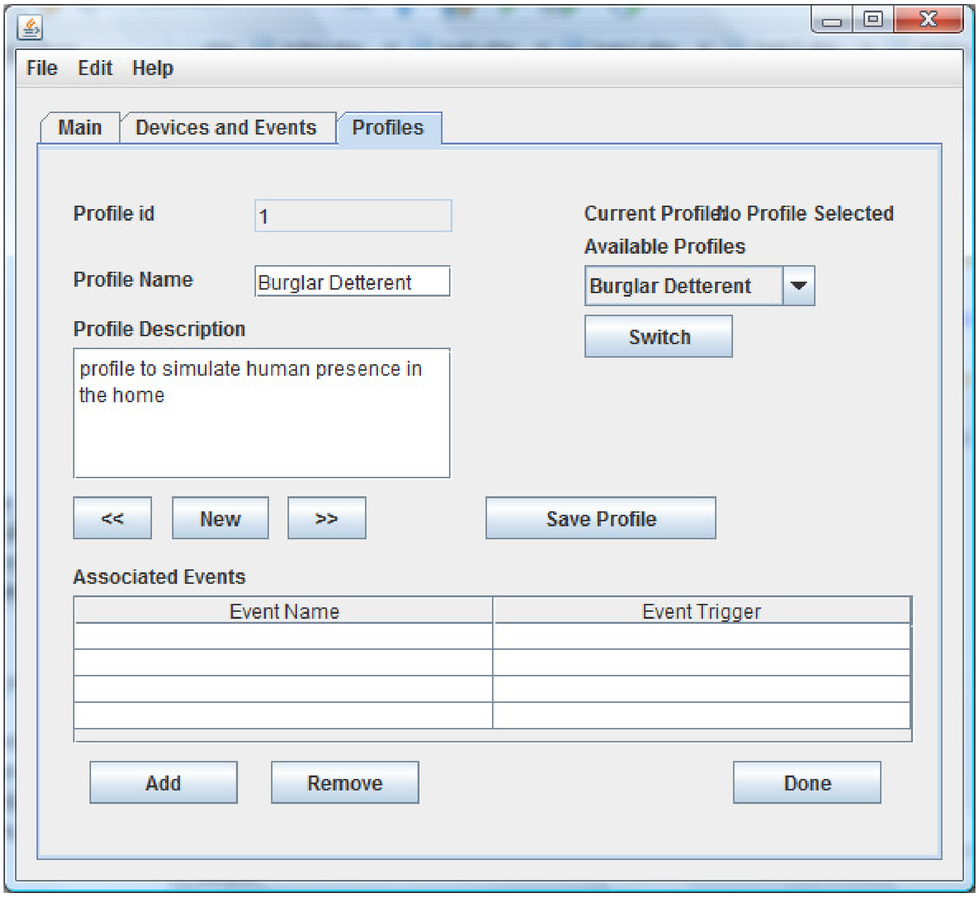
Appendix E



Currently Connected Devices:

Port 1	No Device Connected	N/A
Port 2	Fridge	OFF
Port 3	Washing Machine	ON
Port 4	No Device Connected	N/A
Port 5	Bedroom Light	ON
Port 6	Daylight Sensor	OFF
Port 7	Andrews HP	OFF
Port 8	No Device Connected	N/A

Figure 15 Profile Tab



Profile id: 1

Profile Name: Burglar Detterent

Profile Description: profile to simulate human presence in the home

Current ProfileNo Profile Selected

Available Profiles: Burglar Detterent

Associated Events

Event Name	Event Trigger

Figure 16 Map Devices Dialog

Appendix F

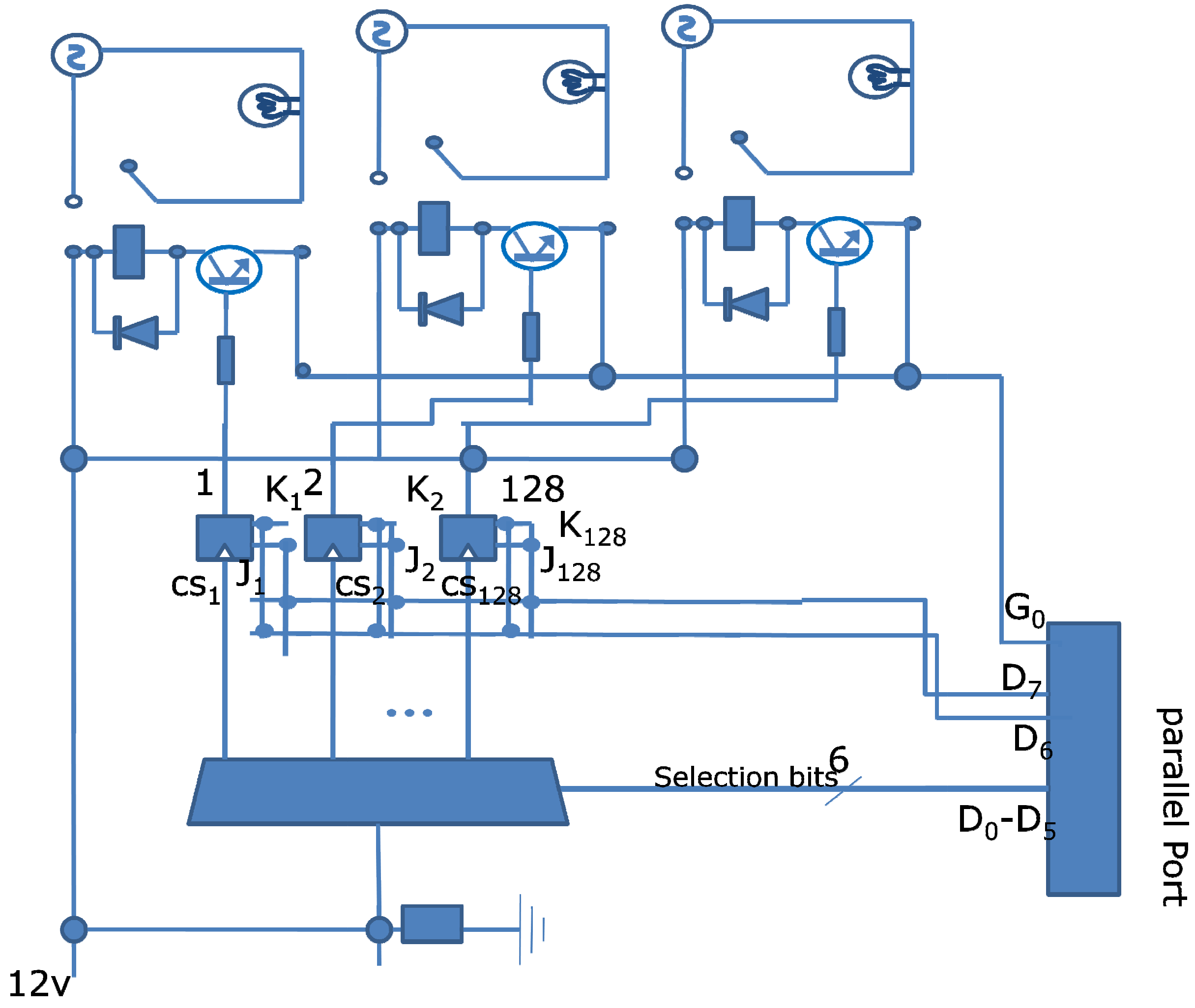


Figure 17 Pc-Device Interface Circuit Diagram