CENTRO UNIVERSITÁRIO DE BRASÍLIA -UniCEUB
CURSO DE ENGENHARIA DE COMPUTAÇÃO

JEAN RIBEIRO LUSTOSA

ACIONAMENTO DE TELEVISOR E ILUMINAÇÃO COM CONTROLE UNIFICADO NO CELULAR

Orientador: Prof. MsC Francisco Javier De Obaldía Díaz

Brasília

Dezembro, 2013

JEAN RIBEIRO LUSTOSA

ACIONAMENTO DE TELEVISOR E ILUMINAÇÃO COM CONTROLE UNIFICADO NO CELULAR

Trabalho apresentado ao Centro Universitário de Brasília (UniCEUB) como pré-requisito para a obtenção de Certificado de Conclusão de Curso de Engenharia de Computação.

Orientador: Prof. MsC Francisco Javier De Obaldía Díaz

Brasília

Dezembro, 2013

JEAN RIBEIRO LUSTOSA

ACIONAMENTO DE TELEVISOR E ILUMINAÇÃO COM CONTROLE UNIFICADO NO CELULAR

Trabalho apresentado ao Centro Universitário de Brasília (UniCEUB) como pré-requisito para a obtenção de Certificado de Conclusão de Curso de Engenharia de Computação.
Orientador: Prof. MsC Francisco Javier De Obaldía Díaz

Este Trabalho foi julgado adequado para a obtenção do Título de Engenheiro de Computação, e aprovado em sua forma final pela Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS

_____

Prof. Abiezer Amarilia Fernandes
Coordenador do Curso

Banca Examinadora:

_____

Prof. Francisco Javier De Obaldía Diaz, Mestre.
Orientador

_____

Prof. Layany Zambrano, Mestre.
UniCEUB

_____

Prof. Leonardo Pol, Mestre.
UniCEUB

_____

Prof. Luciano Henrique Duque, Mestre.
UniCEUB

DEDICATÓRIA

Dedico esse projeto final primeiramente aos meus pais, Dr. Walter Lustosa Júnior e Dra. Sirlhey S. R. Lustosa, que sempre fizeram o melhor para que eu pudesse ter uma educação de primeira e qualidade de vida superior. Nunca deixaram faltar nenhum recurso para que eu pudesse chegar onde estou e adquirir uma base forte para enfrentar não só essa jornada acadêmica, mas a vida.

Agradeço também ao meu irmão, MSc. Leandro Ribeiro Lustosa, que me ajudou bastante no início do projeto com seu vasto conhecimento na área. Sua metodologia de ensino com palavras não filtradas que vêm à mente sem perder seu diferenciado senso de humor foram cruciais para o desenvolvimento do projeto e meu descanso mental.

Esse projeto também é dedicado a minha namorada, Carolinne V. Felix, que soube me apoiar tanto nas noites que pareciam não terminar como nas que pareciam não ser suficientes. Quem me aturou nos momentos de impaciência e estresse e comemorou comigo cada passo dado para o fim dessa trajetória.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Esse projeto possui o objetivo geral de unificar o controle do sistema de eletroeletrônicos, iluminação e televisor, através de uma aplicação Web disponível em uma intranet local. Dessa forma o projeto é dividido em duas bases: uma central de controle para acionamento de componentes e um servidor de aplicação. A central de controle é capaz de se comunicar com o televisor através de uma entrada de serviço deste e via infra vermelho, substituindo o controle remoto convencional. Para o acionamento das lâmpadas, a central estará conectada a um relé que fechará ou abrirá o circuito da fonte de alimentação da lâmpada, conforme solicitação do usuário. O servidor de aplicação possuirá um sistema web que se comunicará com a central para executar as operações solicitadas. Sua comunicação é feita através de um dispositivo sem fio comum a essas duas bases. O usuário será capaz de cadastrar funcionalidades customizadas em apenas um comando, o que facilitará a execução de tarefas rotineiras.

Palavras-chaves: Automação residencial; controle de iluminação; controle de televisor.

ABSTRACT

This project overall objective is to unify the electronics control system, lighting and television, via an local intranet network Web application. Thus, the project is divided in two bases: a control central to operate components and an application server. The control center is able to communicate with the TV via its service entrance and via infrared, replacing the conventional remote control. To operate the lamps, the central connects to a relay that closes or opens the power supply circuit of the lamp, according to the user's request. The application server will run a web system that will communicate with the control central to perform the requested operations. This communication is done through a common wireless device on both bases. The user will be able to register custom functionality in a single command, which will facilitate the execution of routine tasks.

Keywords: Home automation; lighting control; TV control.

CAPÍTULO 1 - INTRODUÇÃO

1.1 - APRESENTAÇÃO DO PROBLEMA

Com o constante lançamento de aparelhos digitais, sua ampla variedade de funcionalidades somada a interoperabilidade destes, sua operação conjunta pode se tornar confusa ou mesmo dificultada pela combinação de comandos necessários para o correto funcionamento do conjunto. Podemos citar como exemplo um sistema de televisão conectado a um *home theater* e um aparelho reprodutor de *blu-ray*: para se assistir a um vídeo em *blu-ray* é necessário selecionar o correto canal da televisão e a correta entrada do *home theater*. Caso essa combinação de seleções não esteja correta, não será possível assistir ao vídeo. Mas seria isso uma dificuldade necessária? Como centralizar ou ao menos facilitar esse controle?

1.2 - OBJETIVOS DO TRABALHO

1.2.1 - Objetivos Gerais

Esse trabalho tem como objetivo desenvolver uma aplicação intranet que possibilite ao usuário controlar uma TV de forma mais simples e centralizada assim como acionar componentes da iluminação. O usuário deve ser capaz de acionar as lâmpadas e enviar comandos a TV ao mesmo tempo.

Constitui como finalidade do projeto construir uma aplicação acessível por *smartphones*, *tablets* e computadores domésticos. Deve-se desenvolver uma central de controle que irá receber solicitações e transmitir sinais aos aparelhos eletrônicos. Deve-se centralizar o controle de televisores e iluminação residencial de forma a evitar vários controles similares para diferentes dispositivos. Seria como simular um controle remoto universal para a residência.

1.2.2 - Objetivos Específicos

A fim de atingir os objetivos gerais do projeto, este será divido em dois polos: desenvolvimento da aplicação intranet e desenvolvimento da central que executará os comandos. Os dois polos serão capazes de comunicar entre si e para concluir as solicitações efetuadas pelo usuário.

O primeiro polo consiste em uma aplicação acessível através de um *browser*, que permitirá ao usuário selecionar dentre os comandos disponíveis quais ele deseja acionar. Essa aplicação permite também que o usuário agrupe vários comandos em um botão só, como uma funcionalidade favorita, para que ele possa executá-la mais facilmente.

O segundo polo é composto de um *hardware* capaz de receber dados por rádio frequência. Esse *hardware* estará conectado as lâmpadas e televisor via cabo, por onde ele transmitirá o resultado da solicitação pedida pela aplicação.

1.3 - JUSTIFICATIVA E IMPORTÂNCIA DO TRABALHO

Atualmente diversos aparelhos necessitam de um mínimo de instrução para serem operados. Porém, com sua variedade, essa operação pode se ser dificultada pois a conexão entre aparelhos necessita de uma configuração ou combinação para funcionar corretamente. Tornar essa configuração automática e centralizada em uma aplicação resultará em uma operação mais prática e fácil para o usuário.

1.4 - ESCOPO DO TRABALHO

O trabalho deve ser constituído de uma aplicação Web capaz de enviar comandos via infravermelho ou entrada de serviço (caso esta exista) para uma TV/monitor. A aplicação deverá ser capaz de ligar, desligar, aumentar volume, diminuir volume e alterar canais da televisão. Sua aplicabilidade se estende a ativação de lâmpadas composta da iluminação de um ambiente. O usuário poderá criar, editar e excluir opções em que a TV e iluminação serão ativados simultaneamente. Inclui-se no escopo do trabalho o desenvolvimento de dispositivo capaz de receber e enviar sinais via infravermelho ou RS-232/P2 (sendo o último conexão de

Serviço da TV). O escopo do projeto também inclui a criação de uma aplicação Web para que o usuário possa disparar esses comandos. Não faz parte do trabalho a criação de dispositivos de transmissão de dados, que serão adquiridos e configurados conforme necessidade.

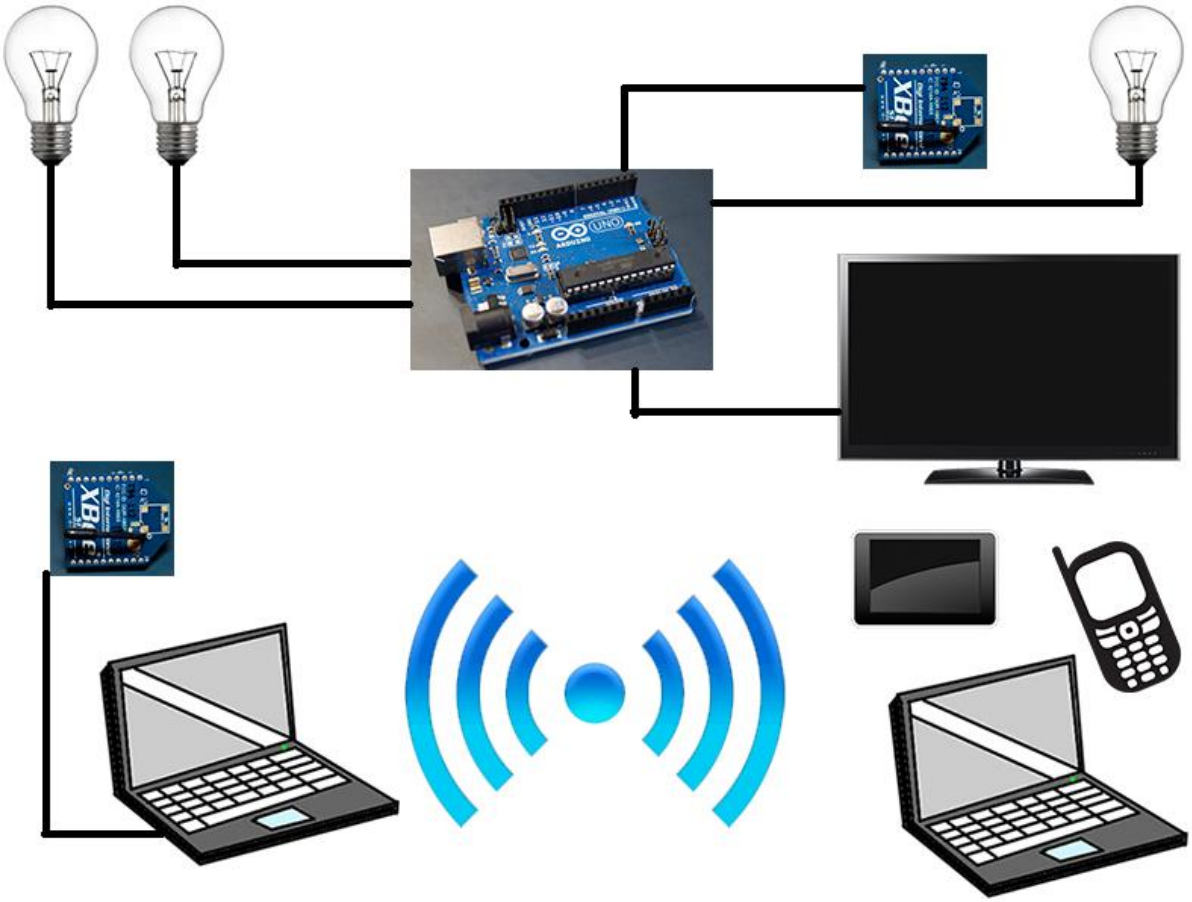

Figura 1.1 - Visão Geral do Projeto

Na figura 1.1 está sendo demonstrado a visão geral do projeto. Na parte superior, ao centro, pode-se verificar a central de controle que receberá os comandos solicitados pelo usuário. Esta está conectada a televisão, três lâmpadas e um transmissor por rádio frequência. Já na parte inferior da figura, a esquerda, encontra-se um *notebook* que representa o servidor intranet que também está conectado a um transmissor por rádio frequência. Por último, no canto inferior direito, observa-se dispositivos que poderão acessar a aplicação intranet via rede *wifi* comum.

1.5 - RESULTADOS ESPERADOS

Espera-se ao final do projeto que o usuário possa controlar o volume, canal e ativação de um televisor. Esse controle deverá ser feito através de sinais enviados por uma entrada de serviço da televisão ou via infra vermelho.

Também espera-se controlar a iluminação de um ambiente. A ativação de dispositivos de iluminação, no caso lâmpadas, será realizada através do fechamento do circuito de alimentação de energia de uma ou mais lâmpadas.

Cadastro de opções (inclusão, alteração e exclusão) customizadas pelo usuário, também serão possíveis. Essas opções farão com que o usuário possa agrupar várias solicitações e uma opção apenas.

O produto gerado deve ser um dispositivo capaz de receber solicitações por rádio frequência, identifica-las e transmiti-las aos aparelhos de destino. Essas solicitações terão origem em um software para configuração e execução dos serviços solicitados pelo usuário.

1.6 - ESTRUTURA DO TRABALHO

Este trabalho está organizado em seis capítulos. O Capítulo 1 possui o objetivo do trabalho, tal como sua justificativa, importância, resultados esperados e sua estrutura.

O Capítulo 2 apresenta uma breve explicação sobre a automação residencial, focado no que será desenvolvido no projeto e o que deve-se esperar deste.

O Capítulo 3 possui o referencial teórico e bases metodológicas para a resolução do problema. Neste capítulo será descrito um resumo dos conceitos e conhecimentos necessários para a conclusão do projeto. Também tratará dos equipamentos utilizados.

O Capítulo 4 irá descrever o modelo proposto do projeto. Neste capítulo também será mostrado os passos para o desenvolvimento da solução proposta.

O Capítulo 5 fará uma avaliação global do modelo e de sua área de aplicação.

O Capítulo 6 irá explicar o que foi possível concluir com o projeto. Sugestões para continuidade do projeto em trabalhos futuros também serão citadas.

CAPÍTULO 2 - AUTOMAÇÃO RESIDENCIAL

A automação residencial se faz cada vez mais presente nas residências facilitando algumas funções do cotidiano. Por ser uma tecnologia que exige um investimento considerável, seu desenvolvimento ainda não é tão explorado o que limita a criação de opções mais acessíveis para o mercado.

Está cada vez mais comum o uso de equipamentos interligados e cada vez mais os usuários querem fazer as funções rotineiras mais rápido e mais facilmente, ou até automaticamente. Seria como exigir ou esperar que um sistema saiba os costumes do seu comprador e que esse seja capaz de aprender novos costumes adquiridos.

A automação residencial aparece como um serviço para tornar as ações rotineiras mais simples (ELITE CASAS INTELIGENTE, 2013), logo, este serviço deve ser de fácil implantação. Seria errado pensar nesse serviço apenas para residências que estejam em fase de construção por ser mais fácil de se passar afiação e planejar onde os dispositivos envolvidos estarão. A implantação em residências já construídas deve ser possível também. Dessa forma, o mais prático seria a utilização de dispositivos que possam sem comunicar por uma tecnologia sem fio, para que se evite reformas e intervenções mais severas na estrutura da residência onde será instalado.

Constantemente as pessoas necessitam se preocupar mais com seus trabalhos e compromissos, logo o projeto visa auxiliar algumas tarefas simples do cotidiano. Desligar várias lâmpadas e televisores com apenas um botão torna mais prática a organização da residência em um fim de festa ou sair para um jantar fora, por exemplo.

2.1 - A ILUMINAÇÃO

“Ambientes de estar pedem bastante aconchego, relaxamento, aproximação entre as pessoas. Nesses ambientes a luz abrange diversas funções além de apenas iluminar. [...] Quando se faz um bom uso de lâmpadas, de distribuição de pontos, de luminárias em geral, de acordo com o projeto, é possível valorizar e adequar qualquer ambiente ao seu uso.” – (Danielle Holanda, em seu blog, novembro de 2013).

O controle de iluminação deve ser feito de forma que a função atual não seja prejudicada, ou seja, o usuário deve ser capaz de mesmo com o sistema instalado e funcionando

poder acionar as lâmpadas da forma convencional, por interruptores. Seria como instalar o sistema em um *three-way*, e esse seria mais um interruptor, porém acionado automaticamente ou eletronicamente.

As luminárias devem ser acionadas em conjunto ou separadamente, conforme instalação e desejo do usuário. Logo, sua instalação pode ser um fator restritivo para o funcionamento do conjunto, o que deve ser considerado no projeto.

O sistema deve ser capaz de identificar se a luminária está atualmente ligada ou desligada. Existirão casos em que o usuário apenas vai querer alterar o estado atual da lâmpada, ou seja, caso ela esteja ligada ele a desligue e vice-versa, simulando um interruptor comum. Mas existirão também casos em que o comando enviado será para definir que a lâmpada esteja acesa, independente se ela já se encontra nesse estado ou não.

2.2 - TELEVISORES

Uma das funcionalidades do sistema, que seria a mais importante por sua praticidade, é a programação de funções pelo usuário. Com essa customização, o usuário poderá criar funções nomeadas por ele que irão acionar quantos dispositivos ele quiser, selecionando a função desejada. Dessa forma fica prático para o usuário ativar ações rotineiras como acionamento de lâmpadas e ligar a televisão no canal de jornal ao chegar em casa, desligar todos os equipamentos no momento em que for se deitar ou até mesmo acionar as luzes no amanhecer para acordar.

Atualmente as televisões possuem em sua grande maioria controle remoto. Logo, uma maneira prática e abranger a maioria dos dispositivos já instalados na residência é simular esse controle remoto, fazendo com que o sistema possa executar as mesmas funções que este. Funções essas como trocar de canal, alterar a origem do sinal, modificar o volume, ligar e desligar, porém o projeto não cobre todas essas funcionalidades.

O projeto consiste em desenvolver uma comunicação que não tenha custo alto, ou seja, que possa ser inferior ou equivalente ao que é praticado hoje no mercado, disponibilizando funcionalidades equivalentes. Sua comunicação deve ser estável, com o mínimo de perdas possível e sem grandes atrasos. Nesse caso a transmissão que substitui o controle remoto poderia ser falha, pelos seguintes motivos: deverá ser colocado um transmissor apontado para a televisão, mas se por algum motivo ele ser deslocado ou conter algum obstáculo no caminho

da TV sua transmissão será interrompida. Para solucionar esse problema pode ser utilizado uma entrada de serviço disponível em alguns modelos para executar as operações necessárias. Essa entrada de serviço é conectada via cabo.

2.3 - TRANSMISSÕES SEM FIO

Outro ponto que deve ser verificado é como o computador (servidor) irá se comunicar com os demais dispositivos. Deve-se considerar conversões de sinais para que os dispositivos de *hardware* possam entender a mensagem vinda do servidor e vice-versa, tornando assim possível a comunicação. Para tal, deve-se verificar se, seja usando um arduíno ou uma placa conversora de sinais, qual o mais adequado e satisfatório para o projeto sem elevar demais o seu custo, o que é um dos objetivos.

Ainda tratando-se de conversões, é necessário fazer a conversão dos *bytes* a serem enviados para o televisor via entrada de serviço contidos no *datasheet* deste. Deve-se considerar qual a melhor maneira de se manter a base de dados com esses código, nunca deixando de lado a performance almejada e tentando, na medida do possível, deixar o sistema o mais leve.

Uma das formas de se baratear o custo do projeto está na criação de centrais que receberiam o sinal do servidor local, ou seja, não se teria um receptor para a televisão e um para cada interruptor das lâmpadas sendo que eles se encontram próximos fisicamente. A ideia seria criar uma central que receberia o sinal do servidor, identificaria para qual dispositivo esse sinal seria transmitido e o enviaria imediatamente através da porta em que o dispositivo estivesse ligado na central.

2.4 - A APLICAÇÃO

A aplicação deve ser lida por qualquer navegador. Seus estilos devem ser dinâmicos de forma que se acessados por um *notebook, smartphone* ou *tablet*, o *design* da página seja reajustado, atendendo a simplicidade que o sistema deve ter. Se for acessado por um computador de mesa ou mesmo um computador portátil, o navegador deve ser capaz de reconhecer e dispor itens na página para que essa fique de fácil navegação. No caso de um dispositivo um pouco menor como um *tablet*, a quantidade de itens na tela deve ser reduzida ou reorganizada e seus links de navegação assim como botões devem ser maiores. Os botões

devem ser maiores pois encontra-se uma dificuldade maior para selecionar itens pequenos em dispositivos sensíveis ao toque. Caso seja acessado por um *smartphone*, a quantidade de itens na tela deve ser ainda menor e o tamanhos de links e botões deve ser mantida, visto que o funcionamento de telas sensíveis ao toque são semelhantes entre *smartphones* e *tablets*.

O cadastramento de novas customizações deve ser clara e objetiva. O usuário deve ser capaz de selecionar seu dispositivo e a funcionalidade a ser ativada no momento da requisição da nova customização de forma intuitiva. Sua edição deve seguir o mesmo padrão, se possível a tela deve ser parecida, mas com um identificador, de forma que facilite o reconhecimento e aprendizado do cadastro porém não o confunda com a inclusão.

Espera-se, então, uma aplicação que possua formatação e navegabilidade compatível com vários dispositivos, sendo assim bastante flexível. Nos capítulos que se seguem abordaremos melhor como esses ajustes serão feitos e explicaremos suas bases para a conclusão deste projeto.

CAPÍTULO 3 - BASES METODOLÓGICAS PARA RESOLUÇÃO DO PROBLEMA

Esse capítulo descreve as bases de conhecimento utilizadas para o desenvolvimento do projeto. Os componentes utilizados também são tratados nesse capítulo.

3.1 - LINGUAGEM DE PROGRAMAÇÃO

Será desenvolvida uma aplicação Web. A linguagem de programação utilizada é a C# (lê-se C-Sharp) Asp.Net. Tal linguagem foi escolhida por ser orientada a objeto, que é um conceito amplamente trabalhado atualmente em faculdades ou mesmo no mercado de trabalho e que será muito útil na resolução do problema abordado. Por ser uma linguagem de programação muito trabalhada, o conteúdo encontrado a respeito dessa linguagem na internet é muito vasto, além de diversas obras publicadas por variados autores.

Para hospedar a aplicação web será utilizado um *notebook* como servidor. O sistema operacional será o Windows 8 que contará com o servidor web *Internet Information Services* versão 6.2. Esse servidor web foi desenvolvido pela Microsoft especificamente para seu sistema operacional.

A aplicação estará desenvolvida em cima do conceito de três camadas: camada de apresentação, camada de negócio e camada de dados. Essas camadas poderão ser denominadas como UI (*user-interface*), BLL (*business logic layer*) e DAL (*data access layer*), respectivamente, ao longo do projeto.

A camada de apresentação é responsável pela apresentação do sistema. O conteúdo apresentado por essa camada pode ser dinâmico ou estático, dependendo do objetivo do sistema. Essa camada oferece a interface entre sistema e usuário, ou seja, em que o usuário poderá fazer o cadastro de componentes e suas funcionalidades. Com uso de controles oferecidos pela plataforma Asp.Net, será possível desenvolver uma camada que pode ser entendida por sistemas desktop ou mobile, de forma que seus estilos serão definidos conforme a plataforma que se está usando. É de responsabilidade do programador desenvolver um sistema que seja formatado em tempo de execução, ou seja, no momento em que ela é requisitada e possua uma apresentação amigável e de fácil operação. Essa camada é muito importante para que o usuário tenha uma experiência satisfatória e simples com o sistema, podendo assim substituir os controles remotos

da residência e interruptores, caso contrário este não iria satisfazer a necessidade de se criar algo que simplifique a rotina do usuário (DAVID HILL, 2004).

A camada de negócio é responsável pelos serviços prestados pela aplicação. Ela é consumida, ou seja, invocada pela camada de apresentação e deve executar funções ou retornar informações requisitadas pelo usuário. Ela é de extrema importância para a aplicação, visto que é nela que ficará contido a forma como as instruções devem se comportar, a ordem em que devem ser realizadas e verificar condições esperadas pelos usuários. Ela é chamada também de camada de persistência, onde as regras de verificação de dados são aplicadas (CLÁUDIO CHIBA, ALEXANDRE NARDI, 2007).

A camada de dados é responsável por guardar, consultar, atualizar ou deletar dados necessários para o funcionamento da aplicação. Assim como a camada de apresentação, a camada de negócio não pode realizar uma conexão direta com o banco de dados. Faz-se essa separação para que, caso a estrutura de dados seja alterada, não seja necessário fazer alterações na camada de apresentação ou de negócio, apenas na de dados. A camada de dados deve conseguir abrir e fechar uma conexão com o banco de dados, executar instruções de inclusão, alteração e exclusão e informar se essas instruções foram bem sucedidas ou, caso ocorram exceções (erros), devolvê-los para que a camada de negócio possa trata-los e tomar as decisões cabíveis (DENNES TORRES, 2013).

Para a comunicação entre o servidor e seus dispositivos sem fio, será utilizado o X-Bee, um transmissor *wifi*. O X-Bee deve ser configurado previamente ao envio dos dados, podendo-se escolher entre broadcast ou transmissão ponto-a-ponto, ou seja, para um receptor específico. No projeto estaremos utilizando a transmissão ponto-a-ponto. Por mais que o objetivo seja controlar diversos dispositivos ao mesmo tempo, esse projeto visa uma solução com apenas uma central controladora, dessa forma não faz sentido sempre configurar o transmissor sem fio para o mesmo endereço repetidas vezes. A expressão “ao mesmo tempo” é relativa. Teoricamente, os comandos para vários dispositivos não são executados ao mesmo tempo, ou seja, em um instante de tempo apenas uma instrução está sendo enviada. Porém, a velocidade do dispositivo é grande o suficiente para que a designação de qual aparelho foi ativado primeiro não seja possível por aparentar serem ao mesmo tempo.

No momento em que o servidor web for requisitado, ou seja, em que algum dispositivo acesse o sistema e solicite alguma função, a aplicação será capaz de resolver qual dispositivo deverá ser acionado e enviar o comando para o receptor sem fio certo. Mas como dito anteriormente, esse projeto possui apenas uma central, logo o receptor será sempre o mesmo.

3.2 - COMPONENTES UTILIZADOS

3.2.1 - Arduino UNO R3

"Arduíno é uma plataforma de computação open-source baseado em uma simples placa com entradas e saídas tanto digitais como analógicas. Possui um próprio ambiente de desenvolvimento que implementa a Linguagem C. O Arduino pode ser usado para desenvolver objetos interativos autônomos ou pode ser conectado a um software em seu computador (ex. Flash, Pocessing, MaxMSP). O ambiente de desenvolvimento (IDE) open-source pode ser obtido gratuitamente". (Fonte: http://www.robocore.net).

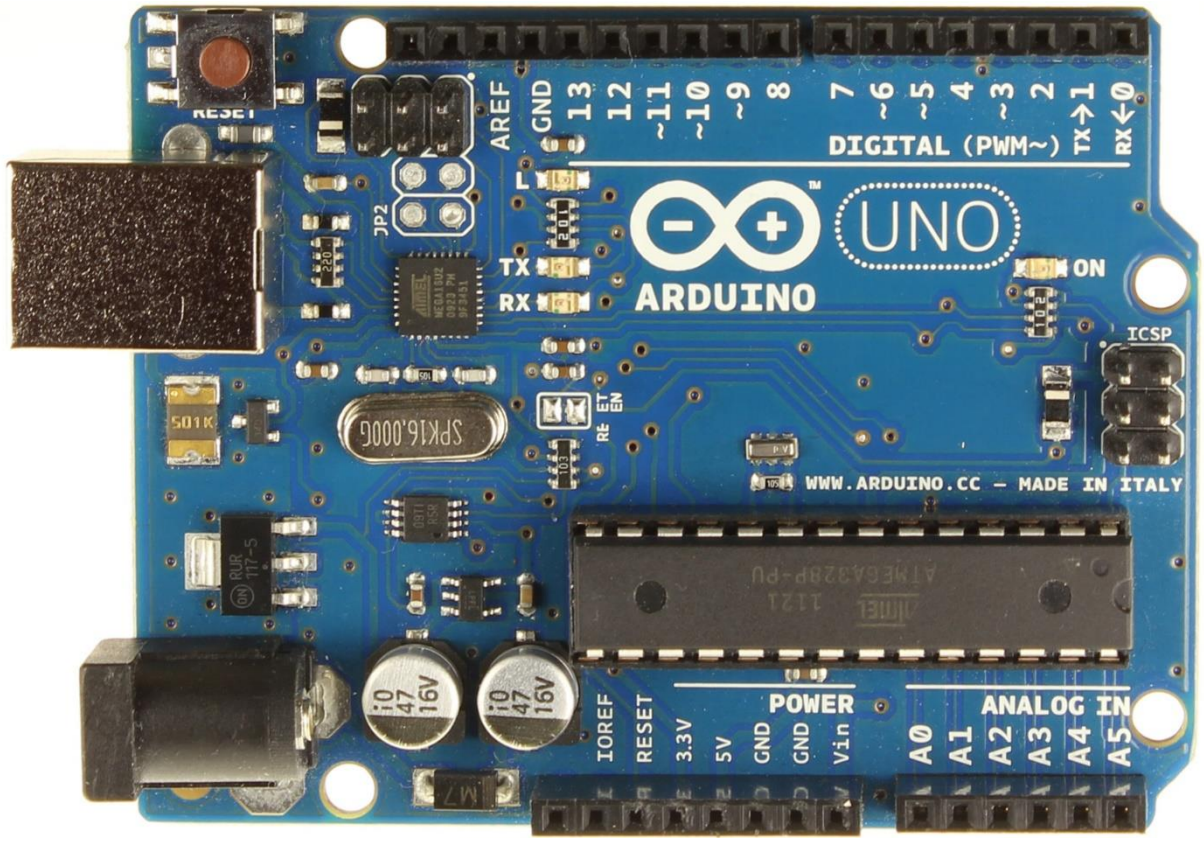

Figura 3.1 – Arduino
(http://arduino.cc/en/uploads/Main/ArduinoUno_R3_Front.jpg, setembro de 2013)

Na Figura 3.1 é possível ver o arduíno e suas portas. Na trilha superior encontram-se as portas digitais e, ainda na parte superior da imagem, o botão de reset do programa do arduino ao lado esquerdo. Já na parte inferior existem as trilhas relacionadas a alimentação e portas analógicas. No lado esquerdo da imagem estão a porta para conexão USB e entrada para alimentação.

O Arduíno é uma excelente plataforma para testes e desenvolvimento de protótipos, sejam esses de cunho acadêmico ou profissional. Seu ótimo custo/benefício o torna uma das plataformas mais utilizadas por estudantes da área de tecnologia da informação.

O Arduíno precisa de uma alimentação com tensão de entrada recomendada entre 7V e 12V e sua tensão de operação é de 5V. Essa alimentação pode ser realizada de três maneiras: via USB, fonte externa de corrente contínua ou via bateria, através das entradas USB B, conector de 2,1mm e pinos Gnd e Vin, respectivamente. Qualquer que seja a maneira escolhida o arduíno a reconhece automaticamente, sem necessidade de definição por chave ou outro método de câmbio.

Essa plataforma dispõe de um *software*, denominado Arduino IDE (*Integrated Development Environment*), para a programação. Sua linguagem de programação é o C/C++ e a IDE foi escrita em *Java*. Nessa IDE é possível escrever o código, compilar, detectar os erros e enviar o programa para o próprio arduino. O envio para o arduino é feito via cabo USB A/B. Também é possível monitorar a entra/saída de dados através de uma porta de comunicação do computador. Esse monitoramento se torna bastante útil na identificação de erros.

3.2.2 - X-Bee

O X-Bee é um módulo de transmissão via rádio frequência que segue o padrão IEEE 802.15.4. Seu alcance é de aproximadamente 30 metros em ambientes fechados e pode ultrapassar 90 metros em ambientes abertos. Ele utiliza a comunicação UART (*Universal Asynchronous Receiver/Transmiter*).

O módulo em questão foi escolhido para ser uma forma alternativa de comunicação. Dessa maneira ele não irá interferir tanto no tráfego sem fio já instalado na casa, essa sendo utilizada apenas para as solicitações via HTTP, apesar de a quantidade de tráfego resultante do sistema todo ser extremamente baixa, podendo-se dizer insignificante em uma rede doméstica comum.

Esse dispositivo também foi escolhido por ser bastante confiável. Ele já possui uma checagem de dados interna que, ao receber um determinado dado, ele verifica internamente se ocorreu alguma falha ou perda na transmissão. Com sua utilização fica a critério do desenvolvedor do sistema criar mais algum mecanismo de checagem da informação, tornando-se uma redundância talvez desnecessária para a finalidade em que será aplicado.

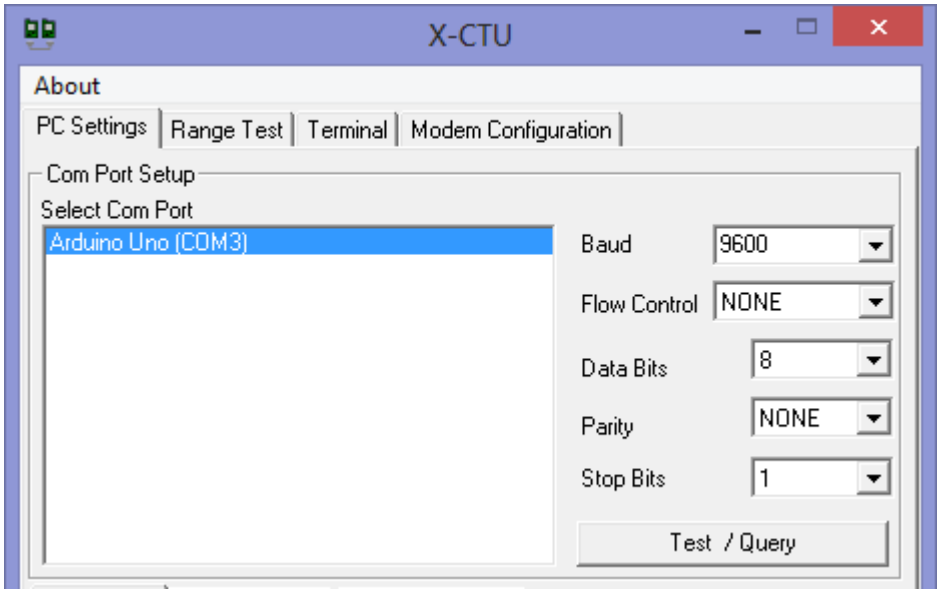

Figura 3.2 - Tela principal do software X-CTU
(Fonte: Autor, novembro de 2013)

Para a configuração do X-Bee utilizou-se o *software* X-CTU, desenvolvido pelo próprio fabricante. Como pode-se analisar na figura 3.2, esse *software* pode ser resumido em quatro abas: *PC Settings*, onde serão definidas as configurações do computador para comunicação com o X-Bee; *Range Test*, aba destinada ao teste de alcance e conexão entre um X-Bee e outro; *Terminal*, utilizado para monitoramento das portas de comunicação do computador, verificando os dados que entram e saem através dela; e *Modem Configuration*, onde será realizada a configuração dos parâmetros de envio e recebimento do dispositivo.

Segue abaixo tabela com descrição dos pinos do X-Bee.

Pino	Nome	Direção	Descrição
1	VCC	-	Alimentação
2	DOUT	Saída	Saída de dados UART
3	DIN / CONFIG	Entrada	Entrada de dados UART
4	DO8*	Saída	Saída digital B
5	RESET	Entrada	Resetar módulo (pulso deve ter ao menos 200ns)
6	PWM0 / RSSI	Saída	Saída PWM 0 / Indicador de força do sinal RX
7	PWM1	Saída	Saída PWM 1
8	[reserved]	-	Não conectar
9	DTR / SLEEP_RQ / DI8	Entrada	Pino de controle de hibernação ou entrada Digital 8
10	GND	-	Terra
11	AD4 / DIO4	Ambas	Entrada analógica 4 ou E/S digital 7
12	CTS / DIO7	Ambas	Clear-to-Send Flow Control or Digital I/O 7
13	ON / SLEEP	Saída	Indicador de status do módulo
14	VREF	Entrada	Referência de voltage para entradas digitais/analógicas
15	Associate / AD5 / DIO5	Ambas	Associated Indicator, Analog Input 5 or Digital I/O 5
16	RTS / AD6 / DIO6	Ambas	Request-to-Send Flow Control, Analog Input 6 or Digital I/O 6
17	AD3 / DIO3	Ambas	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Ambas	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Ambas	Analog Input 1 or Digital I/O 1

20	AD0 / DIO0	Ambas	Analog Input 0 or Digital I/O 0

Tabela 3-1 - Descrição dos Pinos do X-Bee

(Fonte: <https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf>, setembro de 2013)

No projeto foram utilizados apenas os pinos um, dois, três e dez. O pino um conectado a uma fonte de alimentação de 3.3V, os pinos dois e três conectados ao circuito max232 e o pino dez conectado ao terra do circuito.

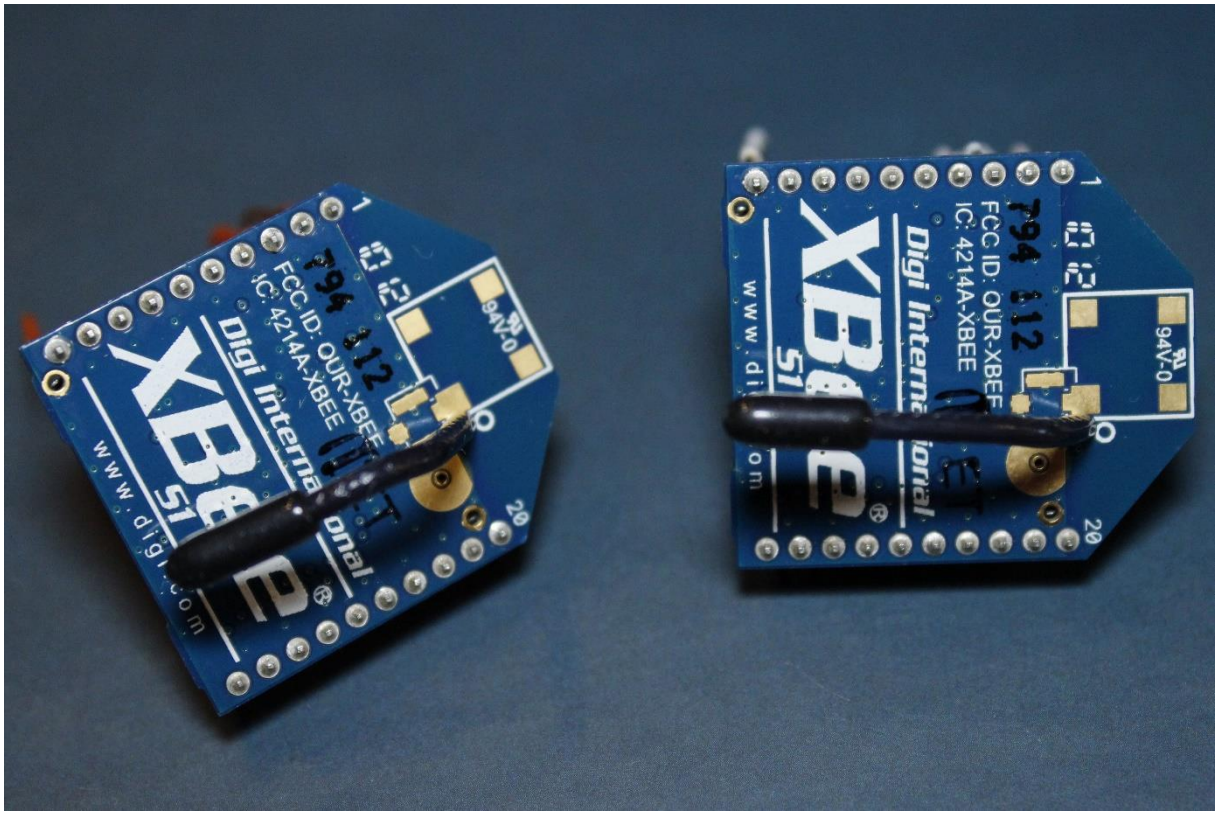

Figura 3.3 - X-Bee's utilizados no projeto
(Fonte: Autor, setembro de 2013)

A figura 3.3 mostra os dois transmissores utilizados no projeto. No lado superior do dispositivo é possível verificar os pinos de 1 a 10 e, no lado inferior, os pinos de 11 a 20.

3.3 - COMUNICAÇÃO

A transmissão UART funciona com a transmissão de *bytes* em sequência para formar a mensagem inteira. No final da mensagem o dispositivo envia um *byte* de paridade que é utilizado para a checagem da mensagem, ou seja, para conferir se a mensagem está correta ou se houve alguma falha ou *byte* não transmitido. Existindo a falha ou inconsistência da mensagem, dependendo da gravidade do erro o sistema pode se utilizar de lógica para corrigir automaticamente ou requisitar o reenvio desta.

No projeto apresentado foi utilizada a transmissão UART assíncrona. Nessa transmissão, tanto receptor quanto transmissor devem estar transmitindo na mesma frequência, configurados antes do início da transmissão.

O Xbee utiliza a lógica serial TTL. A lógica TTL (Transistor-Transistor-Logic) funciona da seguinte forma: um bit ‘0’ é representado por 0V ou próximo a isso enquanto o bit ‘1’ é representado pelo Vcc que costuma ir de 3.3V a 5V.

A porta serial do computador (RS-232) é uma interface muito simples e bastante utilizada, apesar de estar se tornando cada vez menos comum nas máquinas atuais. Sua comunicação é bem similar a TTL, porém seus níveis de operação são diferentes. A porta serial costuma variar entre -25V e 25V, sendo que o menor valor (ou suas proximidades) representam um bit ‘0’ e seu maior valor representa o bit ‘1’.

Tanto o Xbee quanto a porta serial se comunicam utilizando a transmissão UART. Nos dois casos é definido uma taxa de transmissão (*baud* rate) comum, para que os dispositivos possam identificar a informação enviada. Porém seus níveis de operação diferem, o que tornaria impossível sua conexão direta. Para essa conversão, deve-se utilizar o circuito integrado Max232, que fará essa conversão de níveis.

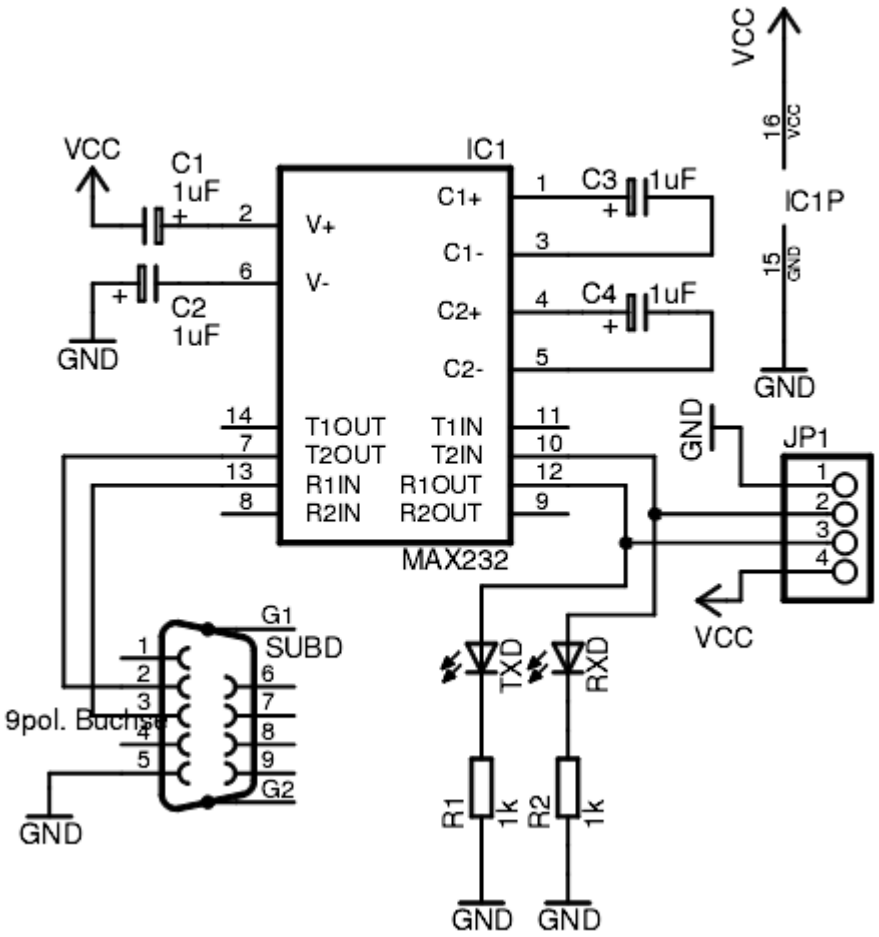

Figura 3.4 - Circuito Max232
(Fonte: http://5vcc.blogspot.com.br/2010/05/trocando-o-o-bom-e-velho-max232-por.html)

Na figura 3.4 pode-se visualizar o circuito max232. No canto esquerdo inferior encontra-se a interface RS-232, que é conectada ao Max232 através dos pinos 2 e 3. Os pinos 8 e 13, R2IN e R1IN respectivamente, são os pinos que devem receber sinais de uma interface RS-232. Já os pinos 9 e 12, R2OUT e R1OUT respectivamente, são os pinos que transmitem o resultado da conversão de nível das entradas de dados da interface serial. O mesmo acontece para o nível TTL, em que os pinos 10 e 11, T2IN e T1IN respectivamente, recebem dados no nível TTL e os disponibilizam convertidos para o nível RS-232 nos pinos 7 e 14, T2OUT e T1OUT respectivamente. Já nos pares de pinos 1 e 3 e 4 e 5 encontram-se capacitores. Os pinos 2 e 6 conectam-se a fonte de energia 5V e *ground*, respectivamente, com um capacitor entre cada conexão. No lado direito da figura podemos observar uma interface com pinos de 1 a 4, que podem ser comparados aos pinos do transmissor de dados, em que o pino 1 conecta-se ao terra do sistema, o 2 a saída de dados RS-232 convertidos para TTL, o pino 3 a entrada de dados do circuito max232 para serem convertidos e o 4 a fonte de energia.

A vantagem dos níveis muitos altos para o bit ‘1’ e muito baixos para o bit ‘0’ é que ele é mais tolerante a ruídos do meio. Podemos verificar isso no exemplo de comunicação serial RS-232: o bit ‘1’, representado por 25V, pode sofrer uma variação de até 20V para baixo, aproximadamente, que o hardware continuará entendendo como um bit ‘1’. Isso não ocorre na comunicação TTL, por exemplo, em que o ‘0’ e o ‘1’ estão muito mais próximos, logo podemos concluir que essa comunicação não tolera 5V de diferença causada por ruídos pois já seria a representação de outro bit.

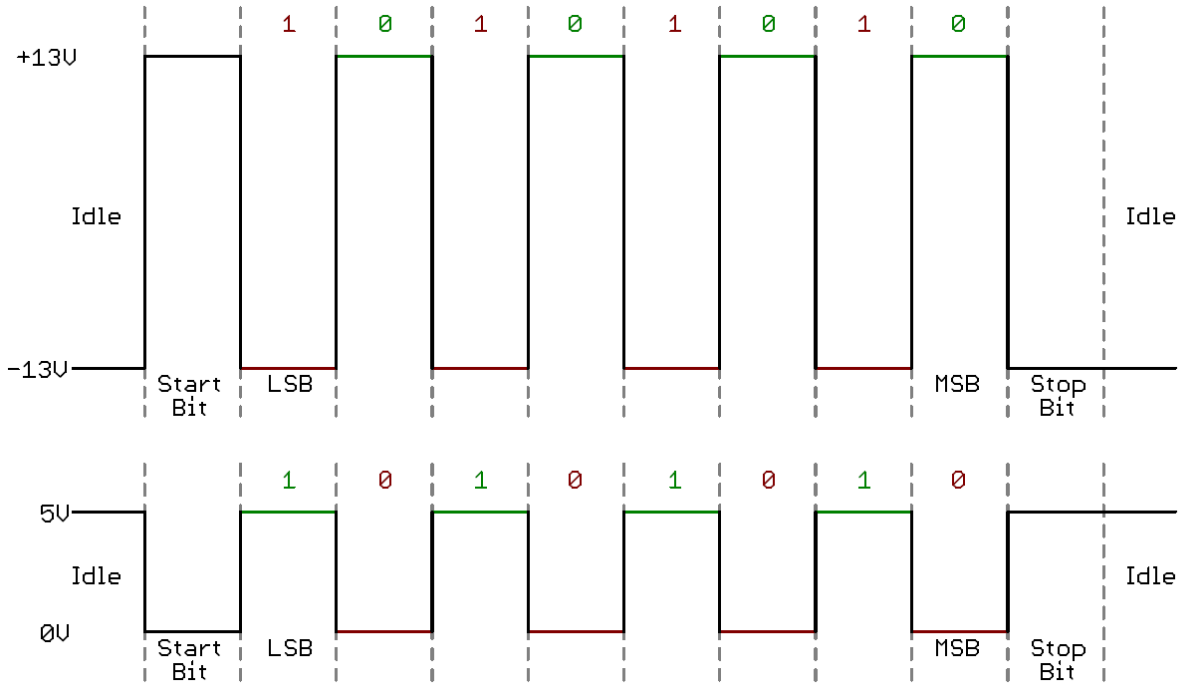

Figura 3.5 - Comparação de níveis RS-232 e TTL
(Fonte: https://www.sparkfun.com/tutorial/RS232vsTTL-BiteSize/ttl-timing.PNG, outubro de 2013)

Na figura 3.3 é possível visualizar a comparação da diferença de tensões entre os níveis citados no parágrafo anterior. Na parte superior da imagem, o nível aplicado na conexão RS-232, na inferior, o nível TTL.

CAPÍTULO 4 - PROJETO PROPOSTO

Nesse capítulo vamos verificar como será realizado o desenvolvimento do sistema de automação residencial. Discutiremos a implementação e etapas necessárias para a obtenção do produto desejado, tal qual as tecnologias e conceitos utilizados.

4.1 - APRESENTAÇÃO GERAL DO PROJETO PROPOSTO

O projeto proposto é composto por duas partes principais, julgadas como essenciais para o funcionamento correto do sistema. São essas a aplicação web, que possui toda a lógica para disponibilizar quais funções estão à disposição do usuário, e o arduíno, que desempenha a função de uma estação para execução dessas tarefas.

Figura 4.1 - Topologia
(Fonte: Autor, novembro de 2013)

Como descrito no parágrafo anterior, é possível verificar na Figura 4.1 a divisão das duas partes do projeto. Ao lado esquerdo desta figura, existe a aplicação ASP.NET/C# acessível por computadores, *smartphones* e *tablets*. Esse acesso se dá através de um roteador sem fio compatível com o padrão 802.11b.

Já no lado direito, ainda da Figura 4, podemos verificar a central desempenhada pelo arduino e os dispositivos de iluminação e televisão. A conexão entre a central e os dispositivos é realizada via cabo ou infra vermelho.

4.2 - DESCRIÇÃO DAS ETAPAS DO PROJETO

A seguir trataremos das etapas necessárias para a execução do modelo proposto. Falaremos sobre as dificuldades, suas soluções e uma visão mais detalhada de como o sistema foi desenvolvido.

4.2.1 - Simulação No Proteus

O Proteus é uma aplicação desenvolvida pela *Labcenter Eletronics*, fundada em 1988 por John Jameson. Esse software é bastante utilizado no meio acadêmico e para desenvolvimento de produtos de grandes renomadas empresas como: Volvo, Ferrari, Sony, Xerox, Hewlett Packard (HP), entre outras.

Figura 4.2- Software Proteus
(Fonte: Programa Proteus 7 Professional)

Uma das dificuldades encontradas foi para simular o arduíno. No Proteus não existe componente que simule esta placa, logo foi necessária uma pesquisa para solucionar esse problema. Uma solução encontrada foram as bibliotecas do Simulino, que além de trazerem design bastante similar ao do arduíno, funcionaram satisfatoriamente.

Foi possível simular no Proteus o arduíno e seu programa desenvolvido pois o *software* permite que se carregue o arquivo *.hex gerado pelo arduino. O esquema de transmissão sem fio não foi representado, já que não encontrou-se o XBee ou equivalente para ser simulado nesse *software*.

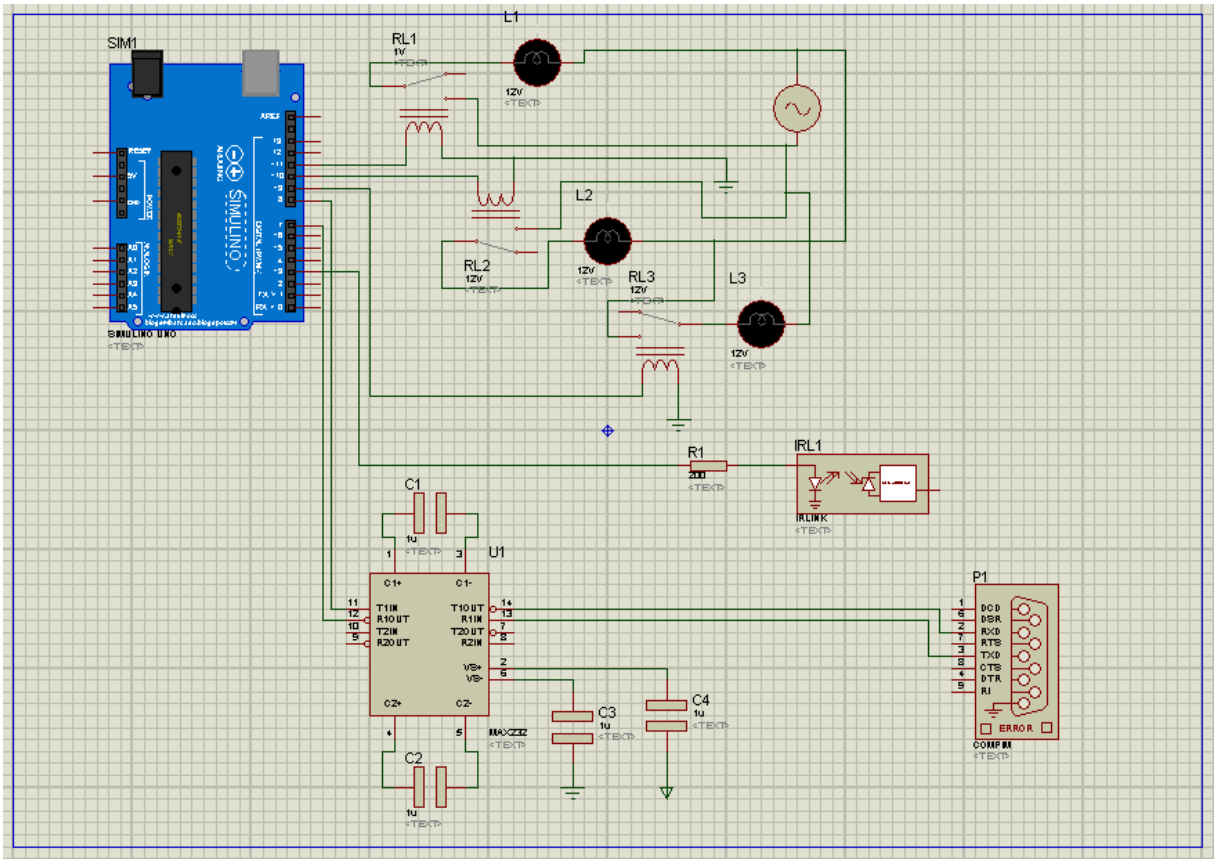

Figura 4.3 - Protótipo no Proteus de central controladora
(Fonte: Autor, novembro de 2013)

Na figura 4.3 está representado o circuito de uma central controladora. No canto superior esquerdo, em azul, está o arduino que recebe e identifica os comandos recebidos via comunicação sem fio. No centro da figura até o topo estão os três relés, cada um conectado a uma lâmpada. Já na parte inferior verificamos um transmissor infra vermelho e a saída para RS-232, essa última passando antes por uma conversão de sinal TTL através do circuito integrado Max232.

4.2.2 - Desenvolvimento De Programa No Arduino

O programa do arduino deve ser desenvolvido em torno da recepção do sinal transmitido pelo computador servidor, o qual discutiremos nos capítulos que se seguem. Ele será responsável pelo recebimento e transmissão do código destinado ao equipamento.

Figura 4.4 - Software Arduino
(Fonte: Programa Arduino)

Conforme explicado anteriormente o arduino terá três formas de comunicação com aparelhos: infravermelho, serviço e liga/desliga (para acionamento das lâmpadas). A distinção dessa comunicação será feita na mensagem recebida pelo arduino, ou seja, o conteúdo enviado pelo servidor já vai conter qual a porta para acionamento, o tipo do sinal (serial, infra vermelho ou comando lógico) e o código de identificação para distinguir o sinal a ser transmitido.

No desenvolvimento do circuito para transmissão do sinal infravermelho para a televisão foi necessário o uso de uma biblioteca que não é nativa do Arduino. Essa biblioteca é a IRemote, que possui três arquivos principais: IRemote.cpp, IRemote.h e IRemoteInt.h. Esses arquivos são o código C++, e dois *headers* para o código, respectivamente.

Essa biblioteca possui a função sendRaw(), que é a utilizada para o envio do código via infravermelho. Seus parâmetros de entrada são: "*unsigned int*", que é o código Raw a ser transmitido, *int*, um inteiro definindo o tamanho desse código e mais um *int*, para especificar a frequência que esse código deve ser transmitido.

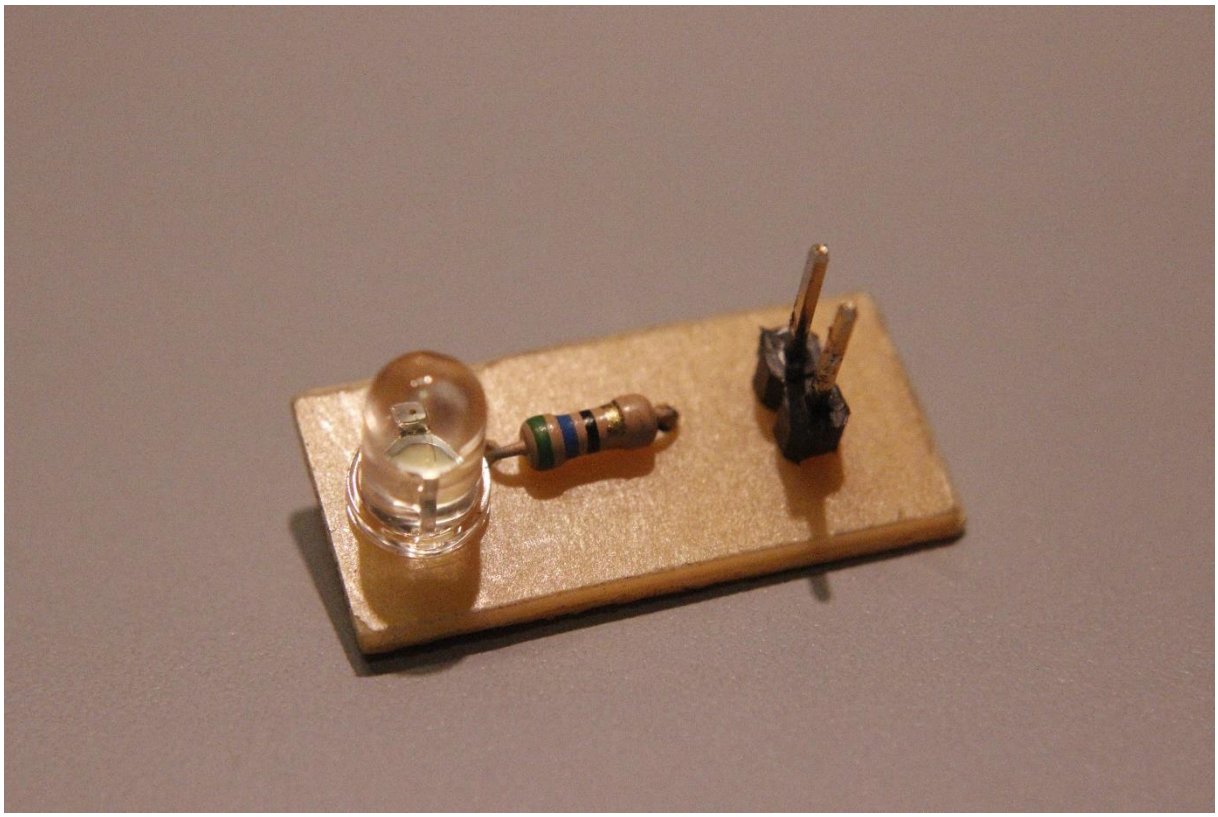

Figura 4.5 - Dispositivo de transmissão por infra vermelho
(Fonte: Autor, novembro de 2013)

Como podemos observar na figura 4.5, o circuito infravermelho é bem simples. Composto apenas por um LED infravermelho e um resistor de 200 ohms. Para isso foi criado esse pequeno dispositivo que possui dois pinos: um para entrada do sinal, que deve ser ligado na porta de saída do arduino, e outro que deve ser ligado ao terra do circuito.

O programa que está rodando no arduino já possui os códigos para transmissão infra vermelho gravados para a TV utilizada no projeto. Esse código é constituído de um *array* de inteiros para cada comando. No apêndice é possível conferir tabela com os códigos utilizados no projeto.

Para a comunicação com o televisor via porta serial foi necessário efetuar a conversão de níveis para compatibilidade entre TTL para RS-232. Criou-se então uma placa para essa

conversão, contendo o circuito Max232, pinos para alimentação, terra e entrada/saída em nível TTL.

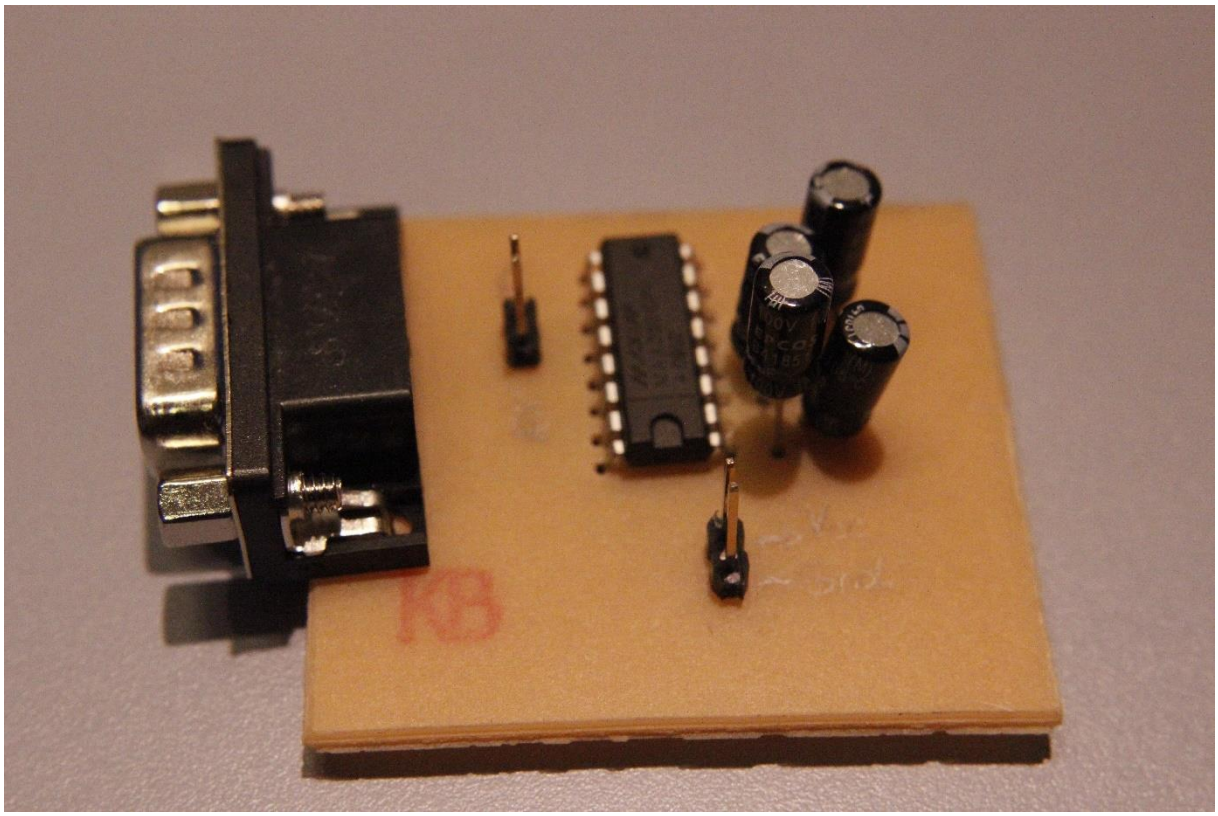

Figura 4.6 - Circuito Max232 para conversão RS-232/TTL.
(Fonte: Autor, novembro de 2013)

Na figura 4.6 pode-se visualizar o dispositivo para conversão RS-232/TTL. No lado esquerdo da imagem encontra-se interface de conexão RS-232. Ao centro verifica-se o circuito integrado Max232, os pinos para transmissão e recepção de dados via TTL e a alimentação do circuito.

Foi necessário também a confecção de um cabo para a conexão entre o arduino e o televisor. Como a entrada de serviço da TV é do tipo P2, aquela utilizada na maioria dos fones de ouvido, e a saída do arduino é do tipo RS-232, o cabo deveria possuir essas duas pontas.

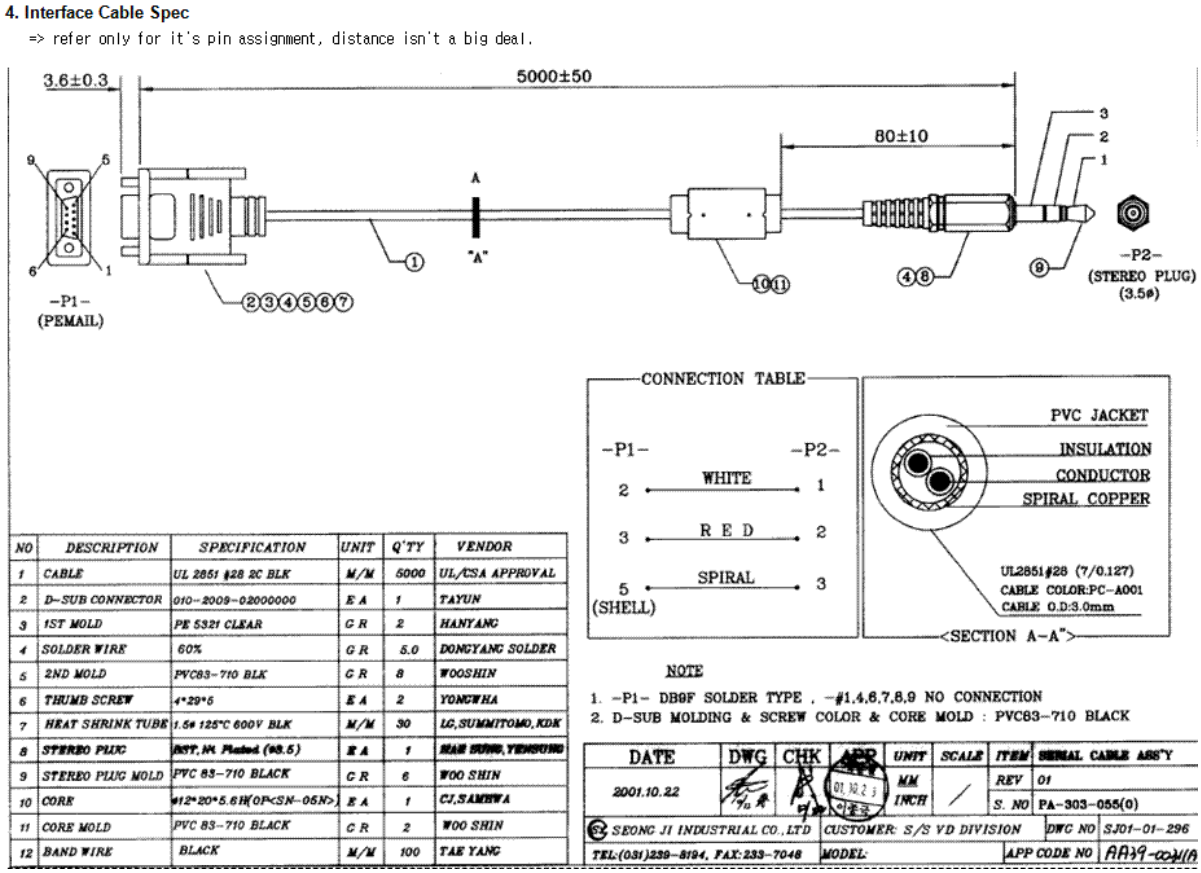

Figura 4.7 - Esquemático Cabo RS-232/P2
(Fonte: http://cdn.avsforum.com/3/3b/3b0f08f2_vbattach116765.gif, novembro de 2013)

Na figura 4.7 pode-se visualizar o esquemático de como o cabo RS-232/P2 deve ser montado. Na interface RS-232 apenas os pinos 2, 3 e 5 serão utilizadas. Esses se conectarão aos pinos 1, 2 e 3, respectivamente, do *plug* P2. Na figura 4.8, a seguir, está o cabo após sua confecção.

Figura 4.8 - Cabo RS232/P2 Confeccionado
(Fonte: Autor, novembro de 2013)

4.2.3 - Desenvolvimento de Aplicação WEB

Para o desenvolvimento da aplicação Web utilizamos o Microsoft Visual Studio 2012, disponibilizado pelo UniCEUB através de sua licença para estudantes. Como a linguagem de programação escolhida foi a C# / Asp.Net, a IDE mais recomendada seria a do próprio desenvolvedor da linguagem: a Microsoft.

".NET é a estratégia da Microsoft para fornecer software como um serviço. Ela foi criada para resolver uma série de problemas que surgem quando se desenvolve software para a Internet. Basicamente o que se pretende com a plataforma .NET é uma maior integração entre os diferentes tipos de dispositivos que existem hoje em dia, como é o caso dos PDAs, telefones inteligentes, pagers, laptops e qualquer outro tipo de dispositivo que tenha condições de acessar a internet." – Alfredo Lotar, em seu livro "ASP.NET com C#".

Figura 4.9 - Visual Studio Professional 2012
(Fonte: Programa Microsoft Visual Studio Professional 2012)

Não entraremos muito em detalhes sobre o ambiente de desenvolvimento visto que é um *software* que dispõe de inúmeras funcionalidades e ferramentas para execução de muitas tarefas que não são abrangidas pelo escopo do projeto. Sendo assim, teremos apenas uma visão superficial e objetiva de como o Visual Studio funciona e onde ele se fez útil.

A aplicação Web foi separada em três camadas: a camada de apresentação, camada de negócio e camada de dados. Cada camada está separada em um projeto (extensão *.csproj), sendo assim ela pode ser substituída conforme a necessidade sem grande dificuldade e

reutilizada para outros projetos. O conjunto de projetos será organizado em uma solução (extensão *.sln), arquivo que reúne logicamente os projetos necessários e em desenvolvimento.

A camada de apresentação é a interface pela qual o usuário fará suas requisições e terá contato com o sistema. Ela é o próprio site e só pode ser utilizada na WWW (*World Wide Web*), logo terá uma interface amigável e intuitiva. O usuário ao acessar o site faz com que o *browser* carregue um arquivo HTML e CSS. O arquivo HTML possui as *tags* que irão definir os componentes que devem estar no site e a estrutura da página. O arquivo CSS é responsável pelos estilos aplicados aos componentes e controles do site tais como fonte e seu tamanho, cor de fundo, imagem de fundo, margens entre vários outros.

Figura 4.10 - Interface Web
(Fonte: Autor, novembro de 2013)

Como pode-se visualizar na Figura 4.10, essa primeira camada é responsável pela interação do usuário com o sistema e será operada por qualquer pessoa capaz de utilizar um computador pessoal ou *smartphone*. Essas páginas conterão botões, menus e *links* que geram eventos a serem tratados de acordo com a solicitação.

Se tratando de uma interface para o usuário, a primeira camada deve ser capaz de identificar erros e trata-los conforme necessário. Existem exceções que são esperadas e erros que não são esperados, e a aplicação deve verificar a melhor maneira de contorna-los e, caso impossível resolvê-los via código, deve-se informar ao usuário que ocorreu um problema

através de uma mensagem amigável, sem entrar em detalhes no erro (caso este seja inesperado) e alimentar um histórico de erros para auxiliar o mantenedor do sistema a corrigi-los numa versão futura.

A segunda camada, a camada de negócio, é a que possui as regras necessárias para execução das funções invocadas pela camada de apresentação. Ela é a camada que faz verificações de lógicas necessárias para o funcionamento do sistema.

Podemos citar, por exemplo, o cadastro de um novo componente para o sistema. Ao incluir esse novo componente iremos especificar em qual porta do arduino este estará conectado e nesse momento o sistema deve verificar se já existe algum dispositivo conectado aquela porta pois, caso tenha, o sistema entraria em conflito já que não se pode ter dois dispositivos conectados fisicamente a mesma porta. Dessa forma a camada de negócio deve gerar uma exceção para a primeira camada afim de informar que não foi possível concluir a inclusão desse componente. No momento da gravação no banco de dados o sistema deve verificar automaticamente qual o próximo ID de comandos favoritos a ser criado, já que o sistema deve assimilar um identificador a cada opção cadastrada.

A terceira camada, a camada de dados, é a responsável pelo comunicação entre a camada de negócio e o banco de dados. Ela é composta de comandos que são executados no banco de dados e podem retornar valores requisitados pela camada de negócio.

Essa camada deve respeitar a estrutura do banco de dados e segui-la fielmente, já que caso ocorra alguma diferença isso pode comprometer o funcionamento de uma funcionalidade inteira. Ela é responsável também pelo tratamento do tipo de dado que o banco está retornando. O banco de dados é composto por tabelas e essas, por sua vez, compostas por colunas. Cada coluna possui um tipo de dado, dentre eles podemos citar: *varchar*, *byte*, *decimal*, *int*, entre outros. Logo, se uma coluna possui o tipo de dado *int*, será impossível incluir a letra “a” nessa coluna. A camada de dados deve obter os registros das colunas da tabela e convertê-los para o tipo equivalente na linguagem utilizada (C#). Um exemplo dessa equivalência é o tipo de banco de dados *varchar* que pode ser tratado pelo sistema em uma variável *string*.

O banco de dados utilizado foi o Microsoft SQL 2012, do mesmo desenvolvedor da IDE utilizada no desenvolvimento da aplicação. O modelo de banco de dados foi desenvolvido seguindo a orientação a objeto, conceito bastante recente e prático de se trabalhar. Para a modelagem de dados foi usado o software *CASE Studio* da fabricante *CharonWare*.

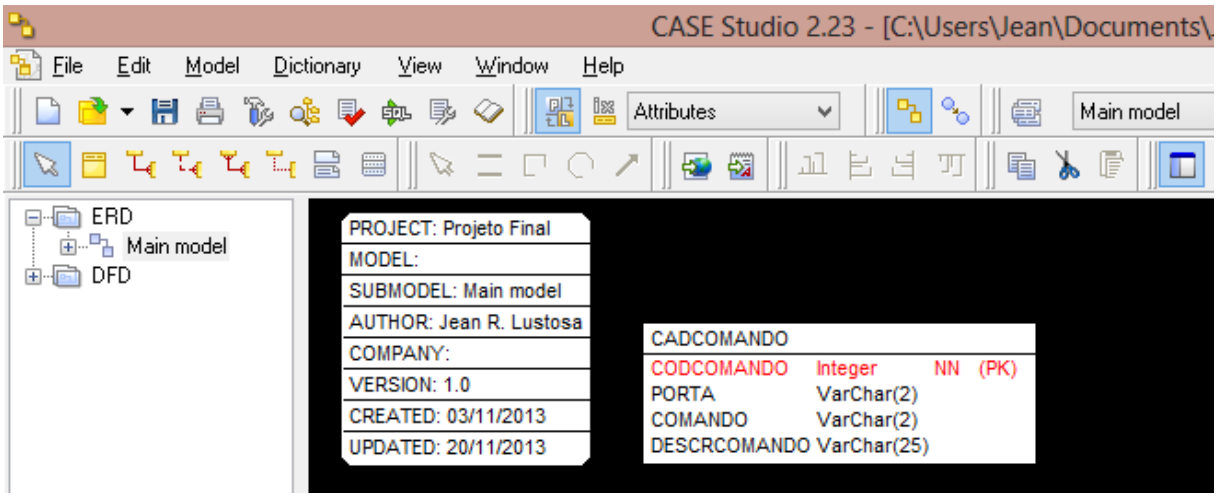

Figura 4.11 - Modelo de Dados no Case Studio
(Fonte: Autor, novembro de 2013)

A figura 4.11 mostra parte da interface do *software* Case Studio. Nele é possível visualizar uma das tabelas utilizadas no sistema assim como suas colunas com seus tipos.

CAPÍTULO 5 - APLICAÇÃO DO PROJETO PROPOSTO

Este capítulo trata da área de aplicação do modelo proposto, sua compatibilidade com dispositivos e descreve como foi feita a avaliação do modelo.

5.1 - APRESENTAÇÃO DA ÁREA DE APLICAÇÃO DO MODELO

O modelo deve ser aplicado em residências de pequeno porte. Levando em consideração que o X-Bee não possui um raio de alcance muito grande, como será explicitado no capítulo 5.3, pode-se experimentar perdas de sinal e falhas ao aplicar em residências maiores ou com muitos obstáculos (paredes, divisórias, portas).

Como sua implantação é simples e não requer intervenções severas na estrutura da casa, o modelo pode ser aplicado em qualquer residência que disponha de um computador ligado durante todo o dia e todos os dias. Caso o computador venha a ser desligado, deve-se lembrar que o modelo é dependente da aplicação *web*, ou seja, esta deve estar funcionando no servidor para que se possa utilizar o sistema.

5.2 - DESCRIÇÃO DA APLICAÇÃO DO MODELO

O modelo desenvolvido é capaz de disponibilizar ao usuário o acionamento de eletrodomésticos via celular ou qualquer navegador de internet. Para que o sistema funcione, basta que o equipamento que vá se conectar ao sistema esteja na mesma rede (intranet) que a aplicação.

O usuário pode criar funções customizadas para acionar vários dispositivos ao mesmo tempo, de acordo com as funcionalidades selecionadas. Essa customização gera um conforto e comodidade para o consumidor, tornando tarefas do cotidiano mais simples e práticas.

Para o funcionamento do modelo, a central de controle (arduino) deve estar devidamente ligada a uma fonte de 9V, para que não se experimente falhas na comunicação com o servidor. A configuração do transmissor deve ser realizada para estabelecer a conexão ponto-a-ponto com o servidor, assim a central pode receber as solicitações e tomar as decisões necessárias.

O servidor necessita estar visível na rede e operando 24 horas por dia. O *Internet Information Service* não pode estar bloqueado pelo *firewall*, caso contrário seu acesso por outro dispositivo seria negado e o usuário ficaria impossibilitado de acessar a aplicação.

5.3 - AVALIAÇÃO GLOBAL DO MODELO

Para a avaliação do modelo proposto foram criados alguns quesitos que deveriam ser supridos e ter sua funcionalidade estável. Esses quesitos foram criados com a finalidade de se focar nos testes e definir metas e assim avaliar o sistema como um todo, classificando-o em satisfatório ou insatisfatório. Esses testes serão descritos nos parágrafos que se seguem.

Primeiramente testou-se a confecção do cabo RS-232/P2. Como sua confecção foi encomendada por terceiros, foi preciso testar sua funcionalidade isoladamente de todo o sistema. Para isso, utilizou-se um multímetro para verificar se os pontos que estavam em curto estavam de acordo com a especificação do cabo.

Após a verificação do cabo, foi necessário verificar se os códigos obtidos para controle do televisor via entrada de serviço estavam funcionando. Foi feita uma aplicação de testes apenas com botões fixos (entenda-se por botões fixos estes onde os comandos não eram obtidos do banco de dados) que enviavam o comando do servidor direto para a TV.

Com o sucesso de testes de comandos, precisou-se testar a conversão de níveis de transmissão de RS-232 para TTL. Dessa forma, o teste consistiu em enviar o comando da saída RS-232 para o circuito Max232 que converte esse sinal para o nível TTL e após essa conversão, conectar a outro circuito Max232 que transforma esse sinal novamente para o nível RS-232, conectado a TV.

O teste que se segue é o da transmissão sem fio, via X-Bee. Após a configuração dos dois transmissores através do programa X-CTU, conectou-se um transmissor ao computador pelo circuito Max232 e o outro ao arduino. O arduino possuía um programa que caso recebesse um determinado dado através do X-Bee acendia um *LED* e, ao acionar um botão, enviava pelo X-Bee um dado para o computador. A monitoração de recebimento do dado pelo servidor foi feita através do programa X-CTU, que possui uma funcionalidade de monitoramento de porta.

Próximo passo foi testar o cadastro e obtenção de dados do banco para envio de comandos. O cadastro foi verificado inserindo-se dados pela aplicação e conferindo diretamente

no banco de dados, através do *SQL Server Management Studio*. Sua obtenção foi conferida depurando a aplicação pelo *Visual Studio* e conferindo as variáveis obtidas.

Após esses testes, a aplicação do servidor estava pronta e funcional. Restou então testar apenas os dados recebidos pelo arduino. Esse teste pode ser realizado imprimindo os dados recebidos em um *LCD* conectado ao arduino. Após o correto tratamento dos dados e suas correspondentes tomadas de decisões o sistema estava pronto para funcionar.

5.3.1 - Compatibilidade Com *Smartphones*/*Tablets*

A compatibilidade com *smartphones* e *tablets* foi satisfatória. O objetivo de se criar uma página com layouts diferentes dependendo do tamanho do browser foi bem sucedido e atendeu as expectativas.

O teste de *layout* e *design* do sistema foi feito acessando o sistema através de um *smartphone*. Realizou-se testes para cada página criada e alteração dos estilos aplicados, corrigindo-os conforme necessário.

O usuário foi capaz de ter acesso prático e fácil aos botões de comandos personalizados, que se encontram na primeira página do site. O clique dos botões também não foi prejudicado, já que o tamanho desses botões foi pensado para acesso em *smartphones* e dispositivos de manipulação por toque.

5.3.2 - Avaliação Do Sinal E Área de Cobertura

Para teste do sinal, montou-se o seguinte cenário: arduino conectado a um computador fixo e um notebook para se deslocar em diferentes cômodos de uma casa. Tanto o arduino quanto o notebook estavam conectados ao X-Bee e ambos utilizando o programa X-CTU para monitoramento das portas e envio de sinais.

Foi realizado um teste com ambos transmissores no mesmo cômodo, onde obteve-se total confiabilidade nos dados transmitidos. Em um segundo momento, testou-se a transmissão com uma distância aproximada de 5 a 6 metros, com uma parede de concreto separando os transmissores e também se obteve resultado satisfatório.

A qualidade caiu quando se aumentou a distância para aproximadamente 10 metros com obstáculos. Em um terceiro momento, testou-se o envio de dados com os transmissores a uma distância de aproximadamente 10 metros com algumas paredes de gesso e concreto, experimentando então a perda de alguns dados onde a mensagem chegou incompleta.

Já em um quarto momento, testou-se a transmissão em andares diferentes, com uma distância também aproximada de 10 metros, onde não se obteve conexão alguma, ou seja, nenhum dado foi recebido pelo notebook.

Podemos concluir então que o alcance dos transmissores foi muito inferior ao esperado. No *datasheet* do dispositivo informava um alcance entre 30 a 90 metros, conforme quantidade de paredes e obstáculos. Porém o alcance que se experimentou foi quase 10 vezes menor que o valor teórico.

CAPÍTULO 6 - CONCLUSÕES

Neste capítulo discutiremos do que foi concluído com a realização desse projeto. Também serão feitas sugestões de melhorias e novas implementações para trabalhos futuros no caso de se dar continuidade a essa projeto.

6.1 - CONCLUSÕES GERAIS

O modelo proposto funcionou satisfatoriamente. Como esperado, o envio de sinais para os equipamentos em questão foram feitos com rapidez e confiança, apesar de sua limitação quanto a distância entre a central de controle e o servidor. Foi observado tal eficiência comparando-se o tempo de resposta ao acionar o televisor através do projeto proposto e através de seu controle remoto, onde a diferença ficou imperceptível.

A aplicação web também funcionou com eficiência e rapidez. Por se tratar de páginas sem muitos recursos a serem carregados como imagens, vídeos e outros arquivos que necessitam de uma quantidade de banda considerável para funcionar, seu carregamento é feito de forma rápida e agradável ao usuário. Pois uma das preocupações do projeto era que a aplicação fosse ao menos tão eficiente quanto pegar um controle remoto ou levantar-se para alcançar um interruptor, visando ser ainda mais eficiente do que isso.

6.2 - SUGESTÕES PARA TRABALHOS FUTUROS

Um leitor de controle remoto poderia ser incorporado ao projeto. Seria uma tentativa de eliminar os controles da residência sem precisar recorrer a *datasheets* e manuais dos fabricantes. Essa tentativa é útil pois alguns manuais não possuem os códigos para comunicação ou podem ser de difícil acesso, não sendo disponibilizados facilmente. Bastaria incluir ao sistema um receptor de infravermelho e desenvolver uma interface em que o usuário poderia selecionar para qual dispositivo ele estaria gravando os sinais, qual o sinal a ser inserido (aumentar volume, canais, liga/desliga) e pressionar o botão apontando o controle para esse leitor. Dessa forma essa nova funcionalidade ficaria disponível para o usuário no sistema.

Com o desenvolvimento desse leitor de controle remoto poder-se-ia estender a automação a condicionadores de ar, *home-theaters* e qualquer outro dispositivo que possua controle remoto capaz de receber dados via infra vermelho.

Disponibilização de controle além da intranet: na internet. Disponibilizando o sistema na internet o usuário seria capaz de definir funcionalidades críticas que deveriam ser acessadas pela internet, para garantir e suprir possíveis esquecimentos ao sair de casa.

Hospedagem da aplicação Web em um servidor menor. Verificar a possibilidade de se hospedar a aplicação em um computador como o *Raspberry Pi* ou *Cubieboard*, o que reduziria significativamente tanto o consumo de energia do sistema e o volume físico ocupado na residência para operação.

REFERÊNCIAS

Arduino – ArduinoBoardUno. Disponível em: http://arduino.cc/en/Main/ArduinoBoardUno. Acessado em Setembro de 2013.

Arduino stuff: Samsung Remote IR-codes. Disponível em: http://arduinostuff.blogspot.com.br/2011/06/samsung-remote-ir-codes.html. Acessado em Outubro de 2013.

Arduino UNO R3. Disponível em: http://www.portalrobotica.com.br/site/index.php?option=com_content&view=article&id=81:arduino-uno-r3&catid=50:arduino&Itemid=26. Acessado em Outubro de 2013.

Arduino, IR sender with Samsung TV. Disponível em: http://www.powenko.com/en/?p=6247. Acessado em Outubro de 2013.

Blog Embarcado: [Índice]. Disponível em: http://blogembarcado.blogspot.com.br/. Acessado em Outubro de 2013.

Criando Uma Camada de Dados em .NET. Disponível em: http://msdn.microsoft.com/pt-br/library/cc517995.aspx. Acessado em Dezembro de 2013.

CSS Tutorial. Disponível em: http://www.w3schools.com/css/. Acessado em Outubro de 2013.

Danielle Holanda: MUITA LUZ!!!!! O ESPETÁCULO DOS AMBIENTES ILUMINADOS – Disponível em: http://danielleholanda.blogspot.com.br/2012/02/muita-luz-o-espetaculo-dos-ambientes.html. Acessado em Novembro de 2013.

Desenvolvimento em Camadas. Disponível em: http://www.microsoft.com/brasil/msdn/tecnologias/arquitetura/Layers_Developing.mspx. Acessado em Dezembro de 2013.

Elite Casas Inteligente. Disponível em: http://www.redeelite.com.br/index.php. Acessado em Dezembro de 2013.

Escolhendo a arquitetura da camada de apresentação correta. Disponível em: http://msdn.microsoft.com/pt-br/library/aa480039.aspx. Acessado em Dezembro de 2013.

HowStuffWorks – Introdução: Como funciona o relé. Disponível em: http://eletronicos.hsw.uol.com.br/rele.htm. Acessado em Setembro de 2013.

Ken Shirriff's blog: A Multi-Protocol Infrared Remote Library for the Arduino. Disponível em: http://www.righto.com/2009/08/multi-protocol-infrared-remote-library.html. Acessado em Outubro de 2013.

Labcenter Eletronics – Professional PCB Design and Simulation Software. Disponível em: http://www.labcenter.com/index.cfm. Acessado em Outubro de 2013.

LOTAR, Alfredo. **ASP.NET com C# Curso Prático**. Novatec Editora Ltda.

PRO RF Modules – 802.15.4 – v1.xEx [2009.09.23]. Disponível em: https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf. Acessado em Setembro de 2013.

RC: 2012 + Samsung TV RS-232 Commands. Disponível em: http://www.remotecentral.com/cgi-bin/mboard/rc-custom/thread.cgi?30761,1. Acessado em Novembro de 2013.

RICHTER, Jeffrey. **Programação Aplicada com Microsoft .NET Framework.** Editora Bookman.

Sandeep Shanbhag, Sonal Mukhi, Vijay Mukhi. **C# Fundamentos**. MAKRON Books.

Serial and UART Tutorial – Disponível em: http://www.freebsd.org/doc/en/articles/serial-uart/. Acessado em Setembro de 2013.

Tutorial Emissor e Receptor Infra-vermelho com Arduino – Laboratorio de Garagem (arduino, eletrônica, robótica, hacking). Disponível em: http://labdegaragem.com/profiles/blogs/tutorial-emissor-e-receptor-infra-vermelho-com-arduino. Acessado em Outubro de 2013.XBee/XBee-

5vcc: Trocando o Bom e Velho Max232 por Circuitos Alternativos. PEREIRA, Alexandre. Disponível em http://5vcc.blogspot.com.br/2010/05/trocando-o-o-bom-e-velho-max232-por.html. Acessado em Novembro de 2013.

APÊNDICE A – Códigos de Sinais Infra Vermelho e RS-232

Abaixo segue tabela com sinais infra vermelho. Esses sinais são enviados pelo arduino como um *array* de inteiros.

ID	Função	Comando
01	Power	4600, 4350, 700, 1550, 650, 1550, 650, 1600, 650, 450, 650, 450, 650, 450, 650, 450, 700, 400, 700, 1550, 650, 1550, 650, 1600, 650, 450, 650, 450, 650, 450, 700, 450, 650, 450, 650, 450, 650, 1550, 700, 450, 650, 450, 650, 450, 650, 450, 650, 450, 700, 400, 650, 1600, 650, 450, 650, 1550, 650, 1600, 650, 1550, 650, 1550, 700, 1550, 650, 1550, 650
02	Número 01	4650, 4300, 700, 1550, 700, 1550, 650, 1550, 700, 400, 700, 400, 700, 400, 700, 450, 700, 400, 700, 1500, 700, 1500, 700, 1550, 700, 450, 650, 400, 700, 450, 650, 450, 700, 400, 700, 400, 700, 450, 650, 1550, 700, 400, 700, 400, 700, 400, 700, 450, 650, 450, 650, 1550, 700, 1500, 700, 450, 650, 1550, 700, 1550, 650, 1550, 700, 1500, 700, 1550, 650
03	Número 02	4600, 4350, 650, 1550, 700, 1500, 700, 1550, 700, 400, 700, 400, 700, 450, 650, 450, 700, 400, 700, 1500, 700, 1500, 700, 1550, 700, 400, 700, 450, 650, 450, 700, 400, 700, 400, 700, 1500, 700, 400, 700, 1550, 700, 400, 700, 400, 700, 450, 650, 450, 700, 400, 700, 400, 700, 1550, 650, 450, 700, 1500, 700, 1550, 650, 1550, 700, 1500, 700, 1550, 650
04	Número 03	4600, 4350, 700, 1500, 700, 1550, 650, 1600, 650, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 1550, 650, 1550, 700, 1500, 700, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 1500, 700, 450, 650, 450, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 400, 700, 400, 700, 1550, 650, 1550, 700, 1500, 700, 1550, 700, 1500, 700
05	Número 04	4600, 4350, 650, 1550, 700, 1500, 700, 1550, 700, 400, 700, 400, 700, 450, 650, 450, 700, 400, 700, 1500, 700, 1550, 650, 1550, 700, 400, 700, 450, 650, 450, 700, 400, 700, 400, 700, 400, 700, 400, 700, 450, 650,

		1550, 700, 400, 700, 400, 700, 450, 700, 400, 700, 1500, 700, 1550, 650, 1550, 700, 400, 700, 1550, 650, 1550, 700, 1500, 700, 1550, 650
06	Número 05	4650, 4350, 700, 1500, 700, 1550, 650, 1550, 700, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 1500, 700, 1550, 700, 1500, 700, 450, 650, 450, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 400, 700, 400, 650, 1550, 700, 450, 650, 450, 700, 400, 700, 450, 650, 450, 650, 1550, 650, 1550, 700, 400, 700, 1550, 700, 1500, 700, 1500, 700, 1550, 700
07	Número 06	4600, 4350, 650, 1550, 700, 1500, 700, 1550, 700, 400, 700, 400, 700, 450, 650, 450, 700, 400, 700, 1500, 700, 1550, 650, 1550, 700, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 400, 700, 1500, 700, 450, 650, 450, 700, 400, 700, 400, 700, 1550, 650, 450, 650, 1550, 700, 400, 700, 1550, 650, 1550, 700, 1500, 700, 1550, 650
08	Número 07	4600, 4350, 700, 1500, 700, 1550, 650, 1550, 700, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 1550, 650, 1550, 700, 1500, 700, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 450, 650, 450, 650, 1550, 700, 1500, 700, 450, 700, 400, 700, 400, 700, 450, 650, 1550, 650, 1550, 700, 450, 650, 400, 700, 1550, 700, 1500, 700, 1550, 650, 1550, 700
09	Número 08	4600, 4350, 650, 1600, 650, 1500, 700, 1550, 700, 400, 700, 400, 700, 400, 700, 450, 700, 400, 700, 1500, 700, 1550, 650, 1550, 700, 400, 700, 450, 650, 450, 700, 400, 700, 400, 700, 1550, 650, 450, 650, 1550, 700, 1500, 700, 450, 700, 400, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 400, 700, 450, 650, 1550, 650, 1550, 700, 1500, 700, 1550, 650
10	Número 09	4600, 4350, 700, 1500, 700, 1550, 650, 1550, 700, 400, 700, 450, 650, 450, 650, 450, 700, 400, 700, 1500, 700, 1550, 700, 1550, 650, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 450, 650, 1550, 650, 1600, 650, 1550, 650, 450, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 1500, 700, 1500, 700, 1550, 700
11	Número 0	4650, 4300, 700, 1550, 700, 1500, 700, 1550, 700, 400, 700, 400, 700, 400, 700, 450, 650, 450, 650, 1550, 700, 1550, 650, 1550, 700, 400, 700, 400, 700, 400, 700, 450, 700, 400, 700, 1550, 650, 400, 700, 450, 700,

		400, 650, 1550, 700, 400, 700, 450, 700, 400, 700, 400, 700, 1500, 700, 1550, 700, 1500, 700, 400, 700, 1550, 650, 1550, 700, 1500, 700
12	Source	4600, 4350, 700, 1550, 650, 1550, 700, 1500, 700, 450, 650, 450, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 1500, 700, 1550, 700, 400, 700, 400, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 400, 700, 450, 650, 450, 650, 450, 700, 400, 700, 400, 700, 400, 700, 450, 650, 1550, 700, 1500, 700, 1550, 650, 1550, 700, 1500, 700, 1550, 700, 1500, 700
13	Power UP	4600, 4350, 700, 1500, 700, 1500, 700, 1550, 700, 450, 650, 400, 700, 450, 650, 450, 700, 400, 700, 1500, 700, 1550, 650, 1550, 700, 450, 650, 450, 700, 400, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 400, 700, 400, 700, 1550, 650, 450, 700, 400, 700, 400, 700, 1550, 650, 450, 650, 1600, 650, 1550, 650, 450, 700, 1500, 700, 1500, 700, 1550, 650
14	Power Down	4650, 4300, 700, 1550, 700, 1500, 700, 1550, 700, 400, 700, 400, 700, 400, 700, 450, 650, 450, 650, 1550, 700, 1500, 700, 1550, 700, 400, 700, 400, 700, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 450, 650, 450, 650, 1550, 700, 400, 700, 450, 650, 400, 700, 1550, 700, 1500, 700, 1550, 700, 1500, 700, 400, 700, 1550, 650, 1550, 700, 1500, 700
15	Aumentar Volume	4600, 4350, 650, 1550, 700, 1500, 700, 1550, 700, 400, 700, 400, 700, 450, 650, 450, 700, 400, 700, 1500, 700, 1550, 650, 1550, 700, 400, 700, 400, 700, 450, 650, 450, 700, 400, 700, 1500, 700, 1550, 650, 1550, 700, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 450, 650, 450, 650, 450, 650, 1550, 700, 1500, 700, 1550, 700, 1500, 700, 1550, 650
16	Diminuir Volume	4600, 4350, 700, 1550, 650, 1550, 700, 1500, 700, 450, 650, 450, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 1500, 700, 1550, 700, 400, 700, 400, 700, 400, 700, 450, 650, 450, 650, 1550, 700, 1500, 700, 450, 650, 1550, 700, 400, 700, 400, 700, 450, 700, 400, 700, 400, 700, 400, 700, 1550, 700, 400, 700, 1500, 700, 1500, 700, 1550, 700, 1500, 700
17	Mudo	4650, 4350, 650, 1550, 650, 1550, 700, 1550, 700, 400, 700, 400, 700, 400, 700, 450, 650, 450, 650, 1550, 700, 1500, 700, 1550, 700, 400, 700, 450, 650, 400, 700, 450, 700, 400, 700, 1500, 700, 1550, 650, 1550, 700,

		1500, 700, 450, 700, 400, 700, 400, 700, 400, 700, 400, 700, 450, 650, 450, 700, 400, 700, 1500, 700, 1550, 650, 1550, 700, 1500, 700

Tabela 6-1 - Códigos de sinais infra vermelho para televisor.

Abaixo segue a lista de comandos enviados pela porta RS-232. O sinal é enviado pelo arduino como um *array* de *byte*.

ID	Função	Comando
18	Power	0x08, 0x22, 0x00, 0x00, 0x00, 0x00, 0xD6
19	Desligar	0x08, 0x22, 0x00, 0x00, 0x00, 0x01, 0xd5
20	Mudo	0x08, 0x22, 0x02, 0x00, 0x00, 0x00, 0xd4
21	Diminuir Volume	0x08, 0x22, 0x01, 0x00, 0x02, 0x00, 0xd3
22	Aumentar Volume	0x08, 0x22, 0x01, 0x00, 0x01, 0x00, 0xd4

Tabela 6-2 - Código de sinais via comando Serial.
(Fonte: http://forum.team-mediaportal.com/threads/samsung-led-tv-serial-control-help.91088/, novembro de 2013)

APÊNDICE B – Aplicação Web (Sem Design)

A aplicação Web possui duas interfaces principais para operação do usuário: a Default.aspx e Config.aspx. A primeira é a página onde o usuário poderá visualizar os comandos e acioná-los. Já a segunda é a página onde o usuário efetua o cadastro desses comandos.

Default.aspx.cs

```
using ProjFinal.Negocio;
using System;
using System.Collections.Generic;
using System.IO.Ports;
using System.Linq;
using System.Threading;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace ProjFinal.Web
{
    public partial class Default : BasePage
    {
        private ComandoNegocio cmdSvc { get; set; }

        protected void Page_Load(object sender, EventArgs e)
        {
            try
            {
                if (!IsPostBack)
                {
                    cmdSvc = new ComandoNegocio();
                    rptFavoritos.DataSource = cmdSvc.ListarFavoritos();
                    rptFavoritos.DataBind();
                }
            }
            catch (Exception ex)
            {
                TrataErro(ex);
            }
        }
    }
}
```

Config.aspx.cs

```
using ProjFinal.Entidades;
using ProjFinal.Negocio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
```

```
using System.Web.UI.WebControls;

namespace ProjFinal.Web
{
    public partial class Config : BasePage
    {
        private ComandoNegocio cmdSvc;

        protected void Page_Load(object sender, EventArgs e)
        {
            try
            {
                if (!IsPostBack)
                {
                    CarregaPagina();
                }
            }
            catch (Exception ex)
            {
                TrataErro(ex);
            }
        }

        private void CarregaPagina()
        {
            cmdSvc = new ComandoNegocio();
            IEnumerable<ComandoEnt> lstCmd = cmdSvc.Listar();

            cblFunc.DataSource = lstCmd;
            cblFunc.DataBind();
        }

        protected void btGrava_Click(object sender, EventArgs e)
        {
            try
            {
                List<string> lstComandos = new List<string>();

                foreach (ListItem func in cblFunc.Items)
                {
                    if (func.Selected)
                    {
                        string codComando = func.Value;
                        lstComandos.Add(codComando);
                    }
                }

                cmdSvc.IncluirFavorito(lstComandos);
            }
            catch (Exception ex)
            {
                TrataErro(ex);
            }
        }
    }
}
```

APÊNDICE C – Aplicação Arduino

Abaixo segue código da aplicação que está rodando no arduino. Ela é composta por um arquivo (ProjFinal_Arduino.ino), que reconhece as instruções recebidas e toma as decisões necessárias para concluir a tarefa requisitada pelo usuário.

ProjFinal_Arduino.ino

```
#include <IRremote.h>
#include <IRremoteInt.h>
#include <SoftwareSerial.h>

unsigned int S_pwr[68]={4600,4350,700,1550,650,1550,650,1600,650,450,650,450,650,450,650,450,700,400,700,1550,650,1550,650,1600,650,450,650,450,650,450,700,450,650,450,650,450,650,1550,700,450,650,450,650,450,650,450,650,450,700,400,650,1600,650,450,650,1550,650,1600,650,1550,650,1550,700,1550,650,1550,650};
// channel 1
unsigned int S_1[68]={4650,4300,700,1550,700,1550,650,1550,700,400,700,400,700,400,700,450,700,400,700,1500,700,1500,700,1550,700,450,650,400,700,450,650,450,700,400,700,400,700,450,650,1550,700,400,700,400,700,400,700,450,650,450,650,1550,700,1500,700,450,650,1550,700,1550,650,1550,700,1500,700,1550,650};
// channel 2
unsigned int S_2[68]={4600,4350,650,1550,700,1500,700,1550,700,400,700,400,700,450,650,450,700,400,700,1500,700,1500,700,1550,700,400,700,450,650,450,700,400,700,400,700,1500,700,400,700,1550,700,400,700,400,700,450,650,450,700,400,700,400,700,1550,650,450,700,1500,700,1550,650,1550,700,1500,700,1550,650};
// channel 3
unsigned int S_3[68]={4600,4350,700,1500,700,1550,650,1600,650,400,700,450,700,400,700,400,700,400,700,1550,650,1550,700,1500,700,400,700,450,700,400,700,400,700,400,700,400,700,400,700,1550,700,1500,700,450,650,450,700,400,700,400,700,400,700,1550,700,400,700,400,700,1550,650,1550,700,1500,700,1550,700,1500,700};
// channel 4
unsigned int S_4[68]={4600,4350,650,1550,700,1500,700,1550,700,400,700,400,700,450,650,450,700,400,700,1500,700,1550,650,1550,700,400,700,450,650,450,700,400,700,400,700,400,700,400,700,400,700,450,650,1550,700,400,700,400,700,450,700,400,700,1500,700,1550,650,1550,700,400,700,1550,650,1550,700,1500,700,1550,650};
// channel 5
unsigned int S_5[68]={4650,4350,700,1500,700,1550,650,1550,700,400,700,450,700,400,700,400,700,400,700,1500,700,1550,700,1500,700,450,650,450,700,400,700,400,700,400,700,1550,700,400,700,400,650,1550,700,450,650,450,700,400,700,450,650,450,650,1550,650,1550,700,400,700,1550,700,1500,700,1500,700,1550,700};
```

```
// channel 6
unsigned int S_6[68]={4600,4350,650,1550,700,1500,700,1550,700,400,700,400,700,450,650,450,700,400,700,1500,700,1550,650,1550,700,400,700,450,700,400,700,400,700,400,700,400,700,1550,700,400,700,1500,700,450,650,450,700,400,700,400,700,1550,650,450,650,1550,700,400,700,1550,650,1550,700,1500,700,1550,650};
// channel 7
unsigned int S_7[68]={4600,4350,700,1500,700,1550,650,1550,700,400,700,450,700,400,700,400,700,400,700,1550,650,1550,700,1500,700,400,700,450,700,400,700,400,700,400,700,400,700,450,650,450,650,1550,700,1500,700,450,700,400,700,400,700,450,650,1550,650,1550,700,450,650,400,700,1550,700,1500,700,1550,650,1550,700};
// channel 8
unsigned int S_8[68]={4600,4350,650,1600,650,1500,700,1550,700,400,700,400,700,400,700,450,700,400,700,1500,700,1550,650,1550,700,400,700,450,650,450,700,400,700,400,700,1550,650,450,650,1550,700,1500,700,450,700,400,700,400,700,400,700,400,700,1550,700,400,700,450,650,1550,650,1550,700,1500,700,1550,650};
// channel 9
unsigned int S_9[68]={4600,4350,700,1500,700,1550,650,1550,700,400,700,450,650,450,650,450,700,400,700,1500,700,1550,700,1550,650,400,700,450,700,400,700,400,700,400,700,450,650,1550,650,1600,650,1550,650,450,700,400,700,400,700,400,700,1550,700,400,700,400,700,400,700,1550,700,1500,700,1500,700,1550,700};
// channel 0
unsigned int S_0[68]={4650,4300,700,1550,700,1500,700,1550,700,400,700,400,700,400,700,450,650,450,650,1550,700,1550,650,1550,700,400,700,400,700,400,700,450,700,400,700,1550,650,400,700,450,700,400,650,1550,700,400,700,450,700,400,700,400,700,1500,700,1550,700,1500,700,400,700,1550,650,1550,700,1500,700};
// source
unsigned int S_scr[68]={4600,4350,700,1550,650,1550,700,1500,700,450,650,450,700,400,700,400,700,400,700,1550,700,1500,700,1550,700,400,700,400,700,400,700,400,700,400,700,400,700,1550,700,400,700,450,650,450,650,450,700,400,700,400,700,400,700,450,650,1550,700,1500,700,1550,650,1550,700,1500,700,1550,700,1500,700};
// channel up
unsigned int S_pup[68]={4600,4350,700,1500,700,1500,700,1550,700,450,650,400,700,450,650,450,700,400,700,1500,700,1550,650,1550,700,450,650,450,700,400,700,400,700,400,700,400,700,400,700,1550,700,400,700,400,700,1550,650,450,700,400,700,400,700,1550,650,450,650,1600,650,1550,650,450,700,1500,700,1500,700,1550,650};
// channel down
unsigned int S_pdown[68]={4650,4300,700,1550,700,1500,700,1550,700,400,700,400,700,400,700,450,650,450,650,1550,700,1500,700,1550,700,400,700,400,700,400,700,450,700,400,700,400,700,400,700,450,650,450,650,1550,700,400,700,450,650,400,700,1550,700,1500,700,1550,700,1500,700,400,700,1550,650,1550,700,1500,700};
// volume up
unsigned int S_vup[68]={4600,4350,650,1550,700,1500,700,1550,700,400,700,400,700,450,650,450,700,
```

```
400,700,1500,700,1550,650,1550,700,400,700,400,700,450,650,450,700,400,700,1500,700,1
550,650,1550,700,400,700,450,700,400,700,400,700,400,700,450,650,450,650,450,650,1550,
700,1500,700,1550,700,1500,700,1550,650};
// volume down
unsigned int
S_vdown[68]={4600,4350,700,1550,650,1550,700,1500,700,450,650,450,700,400,700,400,7
00,400,700,1550,700,1500,700,1550,700,400,700,400,700,400,700,450,650,450,650,1550,70
0,1500,700,450,650,1550,700,400,700,400,700,450,700,400,700,400,700,400,700,1550,700,4
00,700,1500,700,1500,700,1550,700,1500,700};
// TV/DTV
unsigned int
S_tv[68]={4600,4350,650,1550,700,1500,700,1550,700,400,700,400,700,400,700,450,700,40
0,700,1500,700,1500,700,1550,700,400,700,400,700,450,650,450,700,400,700,1500,700,155
0,700,400,700,400,700,400,700,400,700,1550,700,400,700,400,700,400,700,1550,700,1500,7
00,1550,650,1550,700,400,700,1500,700};
// guide
unsigned int
S_guide[68]={4600,4350,700,1500,700,1550,700,1500,700,450,650,450,700,400,700,400,70
0,400,700,1550,650,1550,700,1500,700,450,650,450,700,400,700,400,700,400,700,1550,700,
1500,700,1550,650,1550,700,400,700,400,700,1550,700,400,700,400,700,400,700,450,700,4
00,650,1550,700,1550,650,450,700,1500,700};
// exit
unsigned int
S_exit[68]={4650,4300,700,1550,650,1550,700,1550,700,400,700,400,700,450,650,450,650,
450,650,1550,700,1500,700,1550,700,450,650,400,700,450,650,450,700,400,700,1500,700,4
00,700,1550,700,1500,700,400,700,1550,700,450,650,400,700,450,650,1550,700,400,700,40
0,700,1550,650,450,650,1550,700,1500,700};
// mute
unsigned int
S_mute[68]={4650,4350,650,1550,650,1550,700,1550,700,400,700,400,700,400,700,450,650
,450,650,1550,700,1500,700,1550,700,400,700,450,650,400,700,450,700,400,700,1500,700,1
550,650,1550,700,1500,700,450,700,400,700,400,700,400,700,400,700,450,650,450,700,400,
700,1500,700,1550,650,1550,700,1500,700};

SoftwareSerial serialTV(7,8);
IRsend irsend;

void setup() {
  Serial.begin(9600);
  serialTV.begin(9600);
}

void EnviaIR(unsigned int* comando)
{
  irsend.sendRaw(comando, 68, 38);
}

void loop() {

}
```

```
void LigaLampada(int pinLampada){
  digitalWrite(pinLampada, HIGH);
}

void DesligaLampada(int pinLampada){
  digitalWrite(pinLampada, LOW);
}

void AlteraEstado(int pin){
  digitalWrite(pin, !digitalRead(pin));
}

void serialEvent() {
  char character;
  int passo = 1;

  String TpComando = "";
  String Valor = "";
  String Porta = "";

  while (Serial.available()) {
    character = Serial.read();
    if (character != '\n')
    {
      if (passo == 1)
      {
        if (character != ';')
        {
          Porta.concat(character);
        }
        else
        {
          passo++;
        }
      }
      if (passo == 2)
      {
        if (character != ';')
        {
          TpComando.concat(character);
        }
        else
        {
          passo++;
        }
      }
      if (passo == 3)
      {
        Valor.concat(character);
      }
    }
```

```
}

if (TpComando == "IR")
{
 if (Valor == "01")
 {
  EnviaIR(S_pwr);
 }
 if (Valor =="02")
 {
  EnviaIR(S_pwr);
 }
 if (Valor =="03")
 {
  EnviaIR(S_vup);
 }
 if (Valor =="04")
 {
  EnviaIR(S_vdown);
 }
 if (Valor =="05")
 {
  EnviaIR(S_pup);
 }
 if (Valor =="06")
 {
  EnviaIR(S_pdown);
 }
}
if (TpComando == "CL")
{
 if (Valor == "00")
 {
  DesligaLampada(stringToInt(Porta));
 }
 if (Valor == "01")
 {
  LigaLampada(stringToInt(Porta));
 }
 if (Valor == "99")
 {
  AlteraEstado(stringToInt(Porta));
 }
}
if (TpComando == "SE")
{
 byte SE_VolUp[] = { 0x08, 0x22, 0x01, 0x00, 0x01, 0x00, 0xd4 };
 byte SE_VolDown[] = { 0x08, 0x22, 0x01, 0x00, 0x02, 0x00, 0xd3 };
 byte SE_Mudo[] = { 0x08, 0x22, 0x02, 0x00, 0x00, 0x00, 0xd4 };
 byte SE_PowerOFF[] = { 0x08, 0x22, 0x00, 0x00, 0x00, 0x01, 0xd5 };
```

```
    if (Valor == "07")
    {
      serialTV.write(SE_VolUp, sizeof(SE_VolUp));
    }
    if (Valor == "08")
    {
      serialTV.write(SE_VolDown, sizeof(SE_VolDown));
    }
  }

  TpComando = "";
  Valor = "";
  Porta = "";
}
int stringToInt(String minhaString) {
  int i, x;
  int tam = minhaString.length() - 1;
  int numero = 0;

  for(i = tam; i >= 0; i--) {
    x = minhaString.charAt(i) - 48;
    numero += x * potencia(10, tam - i);
  }

  return numero;
}

int potencia(int base, int expoente)
{
  if(expoente == 0)
    return 1;
  else
    return base * potencia(base, expoente - 1);
}
```