UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

TECNOLOGIA DE ANALISE E DESENVOLVIMENTO DE SISTEMAS

AISLAN LUIZ WENDLING

DIOGO BACH PRIETO

MARLOS LUIZ GEBELUCA

DESENVOLVIMENTO DE UMA APLICAÇÃO AUTÔNOMA PARA GERÊNCIA DE DISPOSITIVOS ELETRÔNICOS COM ÊNFASE NO BAIXO CUSTO E ECONOMIA DE ENERGIA ELÉTRICA

TRABALHO DE DIPLOMAÇÃO DE CURSO

PONTA GROSSA

2014

AISLAN LUIZ WENDLING

DIOGO BACH PRIETO

MARLOS LUIZ GEBELUCA

DESENVOLVIMENTO DE UMA APLICAÇÃO AUTÔNOMA PARA GERÊNCIA DE DISPOSITIVOS ELETRÔNICOS COM ÊNFASE NO BAIXO CUSTO E ECONOMIA DE ENERGIA ELÉTRICA

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Daniel Costa de Paiva

PONTA GROSSA

2014

AGRADECIMENTOS

Em circunstâncias como o presente, o velho clichê é mais do que nunca, verdadeiro: os agradecimentos são sempre inferiores ao número e ao empenho das pessoas que, de um modo ou de outro, permitiram que esse trabalho fosse possível.

Agradecemos primeiramente a Deus, pela sabedoria que nos proporciona sem a qual não teríamos conseguido chegar nessa etapa da nossa vida.

Aos nossos pais por todo apoio que nos deram nos últimos anos e nos ajudaram nos momentos de maior dificuldade.

Às nossas companheiras, namoradas e esposa, que sempre estiveram ao nosso lado, nos aguentando mesmo nas horas mais difíceis e que compreenderam quando o meu tempo precisou ser dedicado ao trabalho e não a elas.

Ao professor Daniel Costa de Paiva que nos orientou com tanta dedicação, sempre transformando uma dificuldade em um aprendizado.

A Universidade Tecnológica Federal do Paraná, Campus Ponta Grossa, representado por sua diretoria, por nos proporcionar este curso de tão alto nível.

A todos os demais, familiares, amigos, professores que sempre nos ajudaram e deram força quando foi preciso.

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Ponta Grossa

Diretoria de Graduação e Educação Profissional

TERMO DE APROVAÇÃO

DESENVOLVIMENTO DE UMA APLICAÇÃO AUTÔNOMA PARA GERÊNCIA DE DISPOSITIVOS ELETRÔNICOS COM ÊNFASE NO BAIXO CUSTO E ECONOMIA DE ENERGIA ELÉTRICA

por

AISLAN LUIZ WENDLING
DIOGO BACH PRIETO
MARLOS LUIZ GEBELUCA

Este Trabalho de Conclusão de Curso (TCC) foi apresentado(a) em 16 de julho de 2014 como requisito parcial para a obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas. O(a) candidato(a) foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Daniel Costa de Paiva
Prof. Orientador

Richard Duarte Ribeiro
Membro titular

Vinicius Camargo Andrade
Membro titular

Tânia Lúcia Monteiro
Responsável pelos Trabalhos
de Conclusão de Curso

Simone de Almeida
Coordenadora do Curso
UTFPR - Campus Ponta Grossa

RESUMO

WENDLING, Aislan L.; PRIETO, Diogo B.; GEBELUCA, Marlos L.; **Desenvolvimento de uma aplicação autônoma para gerência de dispositivos eletrônicos com ênfase no baixo custo e economia de energia**. 2014. Número total de folhas. Trabalho de Conclusão de Curso Tecnologia em Analise e Desenvolvimento de Sistemas - Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2014.

Proposta de uma aplicação que visa diminuir o consumo de energia através do controle autônomo e gerenciamento de dispositivos eletrônicos. O desenvolvimento dessa aplicação tem o intuito de reduzir consideravelmente o valor utilizado para o desenvolvimento do projeto como um todo e seu preço para o consumidor final. Para atingir esses objetivos serão utilizadas tecnologias *open source, ou de código livre,* e desenvolvidos três módulos, um de hardware e dois de software, controle autônomo e interface web. Como hardware foi utilizado o Arduíno, que permite gerenciar qualquer dispositivo eletrônico conectado a ele e também facilita a integração com a aplicação autônoma. O algoritmo desenvolvido busca se adequar à rotina do usuário e este deve escolher e configurar uma das técnicas que estão apresentadas neste trabalho. Utilizando a interface web é feito o gerenciamento da aplicação e a calibragem. Com os módulos citados são realizados experimentos iniciais de uma rede domótica autônoma, capaz de se adequar minimamente à rotina do usuário visando reduzir o consumo de energia. Devido ao fato de utilizar tecnologias livres, foi possível reduzir significativamente o custo de produção/desenvolvimento de um projeto desse porte e facilitar o acesso, devido ao baixo custo, levando conforto e economia para um grupo maior de pessoas.

Palavras-chave: Controle, Autônomo, Adaptável, Domótica, Economia, Acessível.

ABSTRACT

WENDLING, Aislan L.; PRIETO, Diogo B.; GEBELUCA, Marlos L. **Development of a autonomous application for management of electronic devices with emphasis on low cost and energy savings**. 2014. Number of pages. TD – Systems Analysis and Development Technology, Federal Technological University of Paraná. Ponta Grossa, 2014.

Proposal of an software to reduce energy consumption through the autonomic control and management of electronic devices. The development of this application seeks to significantly reduce the value used for the development of the project as a whole and their price for the end consumer. To achieve these goals, open source technologies will be used and developed three modules, one hardware and two software, autonomous control and web interface. Arduino hardware is used to integration with standalone application. The search algorithm developed to suit the user and this routine should choose and configure one of the techniques that are presented in this paper. Using the web interface can be the management of application and calibration. With the modules mentioned initial experiments a standalone home automation network, capable to adapt to the routine minimally user to reduce energy consumption were performed. Because of using free technologies, it was possible to significantly reduce the cost of production / development of a project of this size and make that accessible to a larger group of people, bringing technology, comfort and economy.

Keywords: Control, Autonomous, Adaptable, Domotics, Economic, Accessible.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

CPU - Central Processing Unit
GND - Ground
HTML - HyperText Markup Language
ICMS - Imposto Sobre Circulação de Mercadorias e Serviços
ID - Identificador
IDE - Interactive Development Environment
IO - Input/Output
KB - Kilobyte
KW/H - Kilowatt Hour
MHZ - Megahertz
PHP - PHP: Hypertext Preprocessor
PL/SQL - Procedural Language/Structured Query Language
PROCEL - Programa Nacional de Conservação de Energia Elétrica
RX - Reception
SGBD - Sistema Gerenciador de Banco de Dados
SQL - Search and Query Language
TX - Transmission
USB - Universal Serial Bus
V - Volts
W - Watts
WEB - World Wide Web

SUMÁRIO

1. INTRODUÇÃO

Atualmente, as pessoas passam muito tempo fora de casa, principalmente a trabalho ou estudo e esquecem aparelhos ou luzes ligados, consumindo energia desnecessariamente.

Uma pessoa que esquece a luz ligada ao sair de casa para o trabalho e só vai apagar essa lâmpada ao voltar para casa, vai arcar com este custo sem necessidade, alem de desperdiçar recursos que são utilizados para a produção dessa energia.

Para evitar o desperdício de tempo e dinheiro, o controle de equipamentos eletrônicos é uma tendência para imóveis residenciais, influenciado pela automação industrial e comercial. A automação residencial tem um enfoque direcionado à economia de energia, de tempo e recursos humanos. O controle dos dispositivos e a automação de uma casa é denominado de Domótica (TONIDANDEL, 2004).

Tecnologias de redes domóticas existem e são abordadas neste trabalho, porém em sua grande maioria são voltadas para a área empresarial e as poucas voltadas para o âmbito residencial possuem alto custo, devido à falta de mão de obra especializada e o domínio da tecnologia por poucas empresas, dificultando sua popularização (SANTOS, 2012).

Nos casos dos sensores de movimento e presença, podem ocorrer dois problemas. O primeiro é que o sensor de movimento trabalha com um temporizador que após o término da sua contagem precisa identificar um movimento para atuar novamente. Este comportamento pode ocasionar o desligamento da lâmpada mesmo que alguma pessoa esteja no ambiente, mas sem estar se movimentando. Atividades como leitura, assistir a televisão ou estudar são difíceis em seus ambientes por este motivo. O outro problema que pode atingir os dois tipos de sensores é o fato de atuarem com a presença de animais e em momentos onde a luz natural já satisfaz os requisitos de luminosidade. Outro agravante na utilização desse tipo de solução é o funcionamento em relação a cada tipo de lâmpada devido ao excesso de eventos de ligar e desligar (GREGGIANIN, et al. 2013).

Para contornar estes problemas de custo e de funcionamento, a alternativa é a utilização de tecnologias livres e elaboração de uma solução para o controle autônomo de equipamentos eletrônicos com ênfase na economia de energia.

A proposta neste trabalho é, então, utilizar tecnologias livres e desenvolver uma aplicação que realize controle autônomo, buscando se adequar à rotina do usuário. Para isto, o controle de equipamentos eletrônicos consiste em ligar ou desligar um equipamento utilizando o Arduíno (ARDUÍNO, 2014). Além da placa controladora de *hardware* livre, dois módulos de *software*, um para Inteligência e controle autônomo e outro para interface *web* fazem parte dos componentes do trabalho. A utilização de sensores deve ser avaliada para melhorar a qualidade do sistema, porém não deve ser uma necessidade por sua aquisição estar relacionada ao custo e ao poder aquisitivo do cliente.

Como aplicação, foi escolhido o controle de lâmpadas devido ao fato de ser o segundo maior gasto de energia elétrica segundo relatório da PROCEL (SALVADOR, SANTOS E SANTOS, 2007).

A visão geral do trabalho é apresentada na Figura 1.

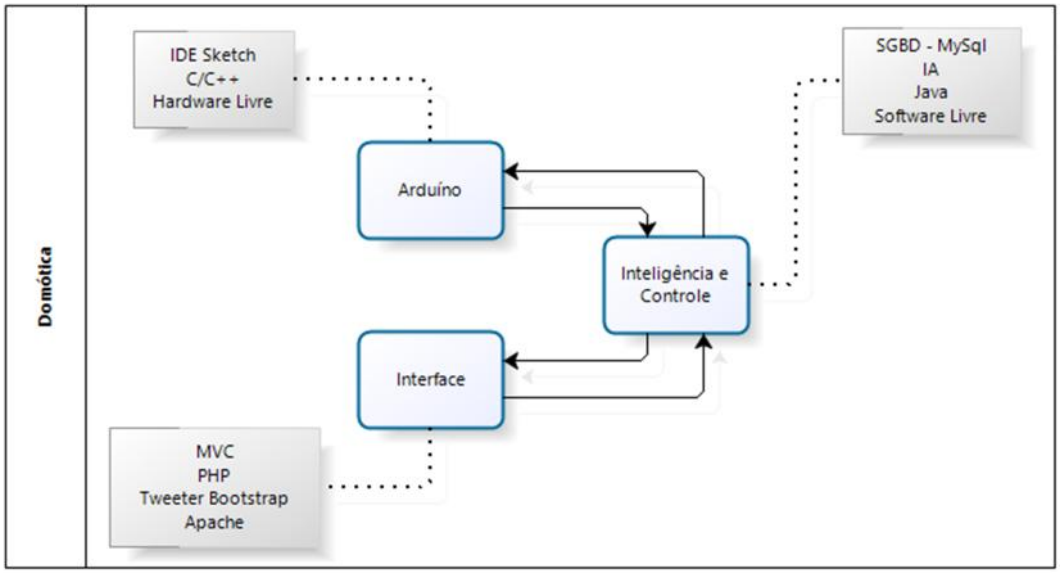

Figura 1 – Visão geral do trabalho
Fonte: Autoria própria

1.1. OBJETIVOS

O objetivo geral do trabalho é desenvolver uma aplicação que, de forma mais inteligente possível, controle dispositivos eletrônicos de um ambiente proporcionando economia, comodidade e se adequando à rotina dos usuários. Essa aplicação deve contemplar os módulos de *hardware* e *software*. O Arduíno será responsável por fazer a comunicação entre a aplicação e os dispositivos eletrônicos, além da coleta de informações utilizando sensores e atuadores eletrônicos.

1.1.1. OBJETIVOS ESPECÍFICOS

- Compreender o *hardware* Arduíno.
- Compreender a programação e comunicação da aplicação com o Arduíno.
- Levantar os requisitos da aplicação.
- Desenvolver a modelagem da aplicação.
- Desenvolver a programação do Arduíno.
- Desenvolver interface *web* para o usuário.
- Elaborar técnicas e algoritmos para a tomada de decisão da aplicação.
- Realizar testes e demonstração do protótipo.

1.2. ORGANIZAÇÃO DO TRABALHO

Este trabalho está dividido em cinco capítulos. No próximo é apresentada a revisão bibliográfica do trabalho, os estudos sobre os recursos utilizados, como o Arduíno, a linguagem PHP e o banco de dados MySQL. No capitulo três é apresentada a Contribuição do trabalho, com o desenvolvimento da aplicação. No capítulo quatro são demonstrados os resultados dos testes realizados e o cinco apresenta os resultados obtidos. No capítulo seis estão as considerações finais a respeito do trabalho.

2. REVISÃO BIBLIOGRÁFICA

Este capítulo aborda os tópicos referentes à revisão bibliográfica necessária para o embasamento dos conteúdos abordados neste trabalho. A seção 2.1 discorre sobre domótica, a seção 2.2 aborda o *hardware* Arduíno, a seção 2.3 descreve a linguagem de programação PHP, na seção 2.4 está apresentado o servidor de conteúdo *Web* Apache, na seção 2.5 o padrão de *layout* Twitter Bootstrap, na seção 2.6 é apresentado o sistema de gerenciamento de banco de dados MySQL e, por último, na seção 2.7 é apresentada a metodologia de aprendizado.

2.1. DOMÓTICA

Domótica é uma área muito interessante que atraiu um grande número de desenvolvedores e pessoas interessadas nos últimos anos (NUNES, 2009). Ela está ligada ao desenvolvimento de ambientes inteligentes, com os objetivos principais de trazer maior conforto e maior segurança ao usuário.

O usuário pode optar por ter uma rede domótica na parte de segurança da casa que pode ser uma rede que alerte um princípio de incêndio em determinado ambiente ou até mesmo a invasão por um ladrão no ambiente da residência.

Na parte de conforto as possibilidades são inúmeras como, por exemplo, controlar a iluminação da residência, o aquecimento, a intensidade da iluminação ou até mesmo ligar e desligar outros componentes com horários agendados. Exemplos em desenvolvimento são de uma cafeteira ligar pela manhã para dar início a suas atividades ou uma televisão ligando e desligando nos horários pré-determinados (WORTMEYER, 2005). Resumindo, pode se ter controle de qualquer dispositivo eletrônico na residência. E essas duas vertentes de rede domótica podem ser incluídas no mesmo pacote, tudo depende apenas do que o usuário escolher.

Embora os benefícios associados à domótica sejam vários e genericamente reconhecidos, a penetração da domótica, em termos de mercado, tem sido pequena. As razões para este fato são múltiplas, podendo

ser apontadas razões de ordem tecnológica e facilidade de utilização e percepção do grau de utilidade. Isso ocorre pelo fato de não existir uma padronização definida para a implementação de uma rede domótica, podendo utilizar diferentes aparelhos e diferentes métodos para atingir o mesmo objetivo.

Esta ausência de padrões está mudando nos últimos anos. Dados de uma pesquisa (SANTOS, 2012) feita pela AURESIDE (Associação Brasileira de Automação Residencial) demonstram que o mercado da domótica teve um bom desempenho nos últimos anos com um crescimento de 40% no período de 2007 até 2009. Apesar disto, é pouco dominada por profissionais e ainda tem custo elevado para o desenvolvimento.

Algumas iniciativas de aplicação de domótica com baixo custo são identificadas a seguir:

O trabalho de José, Giacomelli e Luis Fontes (2012), visa levar a domótica à classe média da população utilizando o *hardware open source* “Arduíno”, porém diferente do projeto apresentado neste trabalho, que terá um controle autônomo, onde o usuário terá a menor interferência possível no sistema. O deles segue um controle do usuário em uma plataforma *Web*, em que o usuário controla as ações dos componentes presentes no interior dos cômodos, sendo aconselhado às pessoas idosas e portadores de algum tipo de deficiência, devido às questões de dificuldades locomotivas e de segurança.

O trabalho desenvolvido por Meira, Delfino e Olandini (2012), é totalmente realizado com plataformas *open source*, tanto *software* quanto *hardware,* e também visa o controle de lâmpadas residenciais via “Arduíno”. O que o diferencia deste trabalho (que adapta sua agenda de acordo com a rotina do usuário) é que no projeto deles, o usuário cadastra os componentes e passa ao sistema as tarefas de acender e apagar tais componentes nos horários definidos. Cabe ao usuário se adaptar a rotina do *software*, forçando-o a atualizar o sistema sempre que sua rotina mudar.

DroidLar (EUZEBIO, RIBEIRO, 2011) é um sistema para automação residencial em telefone celular com sistema operacional Android, e possibilita que o usuário envie comandos via *smartphone* de qualquer lugar do mundo. Trata-se de um projeto com custo alto.

2.2. ARDUÍNO

O Arduíno faz parte do conceito de *hardware* e *software* livre e está aberto para uso e contribuição de toda a sociedade. O conceito Arduíno surgiu na Itália em 2005, com o objetivo de criar um dispositivo que fosse utilizado em projetos/protótipos construídos de forma menos dispendiosa do que outros sistemas disponíveis no mercado (RENNA, 2013).

A ideia por trás do Arduíno é tornar fácil seu entendimento, programação e aplicação. Trata-se de um microcontrolador chamado Atmega, contendo microprocessadores, memória e periféricos de entrada e saída. Além disso, ele conta com uma biblioteca de funções com o intuito de simplificar a programação, com sintaxe muito similar as das linguagens C e C++.

Em geral, o Arduíno é um *kit* de desenvolvimento que pode ser visto como uma unidade de processamento capaz de mensurar variáveis do ambiente externo, transformadas em sinais elétricos correspondentes, por meio de sensores ligados aos seus terminais de entrada (RENNA, 2013). O Arduíno pode processar computacionalmente a informação contida e pode atuar no controle ou acionamento de outros elementos eletrônicos conectados a ele.

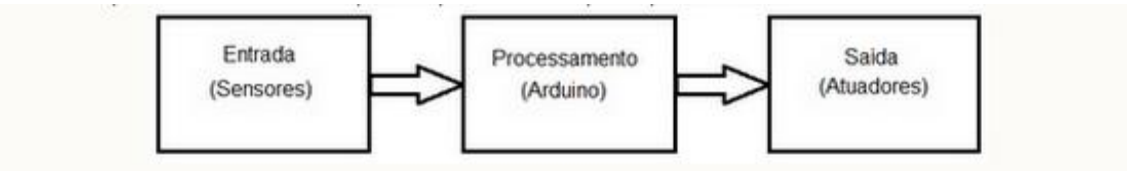

Figura 2 - Diagrama de blocos de uma cadeia de funcionamento do Arduíno
Fonte: Renna, 2013

A Figura 2 representa o processamento utilizando o Arduíno. No primeiro bloco ocorrem as entradas de valores captadas pela placa, por exemplo, um toque na chave táctil. Logo após ela faz um processamento dos dados assumindo que ao tocar na chave ele envie o sinal 1, com esse dado no segundo bloco é realizado o processamento do que esse dado representa, retornando para o terceiro bloco uma resposta que poderá ligar um componente/executar uma ação pré-determinada.

2.2.1. Hardware

Existem vários tipos de Arduíno e cada um possui uma configuração diferente, com inúmeras portas, entradas, memória e outras características especificas. A Tabela 1 apresenta as configurações e informações relevantes do Arduíno UNO, escolhido para desenvolvimento neste trabalho.

Tabela 1 - Especificações Arduíno Uno

ESPECIFICAÇÕES - ARDUÍNO UNO	
Microcontrolador	ATmega328
Tensão de operação	5V
Tensão de entrada (recomendada)	7-12V
Tensão de entrada (limites)	6-20V
Pinos de I/O digitais	14 (seis deles com saída PWM)
Pinos analógicos	6,00
Corrente CC por I/O Pino	40mA
Corrente do Pino 3.3V	50mA
Memória Flash	32KB (ATmega328) e 0,5KB usado pelo bootloader
SRAM	2KB (ATmega328)
EEPROM	1KB (ATmega328)
Velocidade do Clock	16Mhz

Este Arduíno é ideal para quem está começando a desenvolver aplicações, sendo apenas limitado no número de portas de entrada e saída que possui.

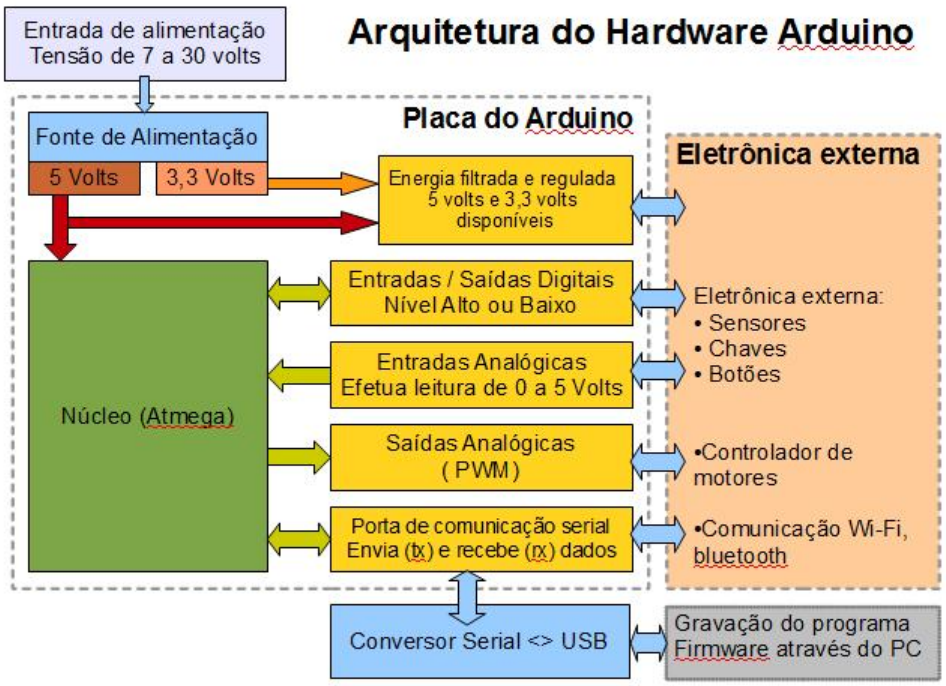

Figura 3 - Arquitetura do Arduíno
Fonte: Robotizando, 2012

A arquitetura do Arduíno está representada na Figura 3, e na Figura 4 é possível identificar os seus componentes eletrônicos.

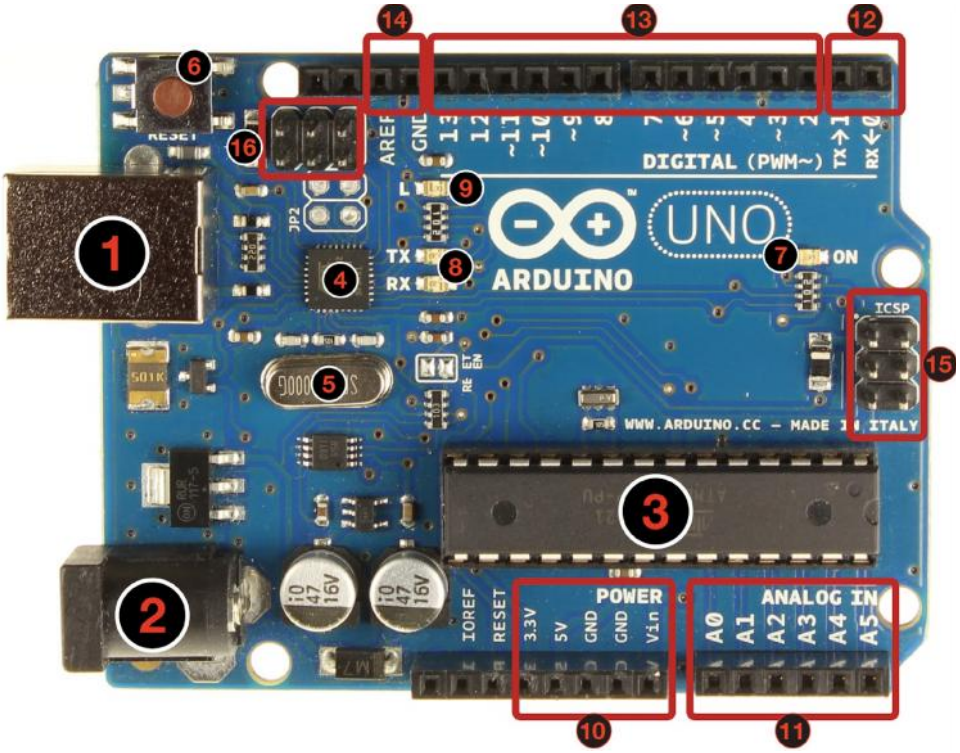

Figura 4 - Blocos de uma placa Arduíno
Fonte: CreativeCoder, 2012

1. Porta USB;
2. Entrada da fonte de alimentação;
3. Processador ATmega8;
4. Chip de comunicação;
5. Cristal de 16 MHz;
6. Botão de reset;
7. Led que indica se o Arduíno está ligado;
8. Leds TX/RX para indicar trafego de dados;
9. Led de uso geral;
10. Pinos de energia para alimentar componentes;
11. Seis entradas analógicas;
12. Pinos TX/RX;
13. Doze entradas e saídas digitais, sendo que seis delas podem emular sinais analógicos;
14. Pinos terra;
15. Pinos de programação do processador;
16. Pinos de programação da porta USB.

2.2.2. Software

O ambiente para desenvolvimento de *software* para Arduíno é um compilador C e C++ que usa uma interface gráfica construída em JAVA. É uma IDE (*Integrated Development Environment* ou Ambiente Integrado de Desenvolvimento), de nome Sketch, representado na Figura 5.

Depois da realização da programação, o *software* deve ser enviado para a placa para que sejam realizados os testes.

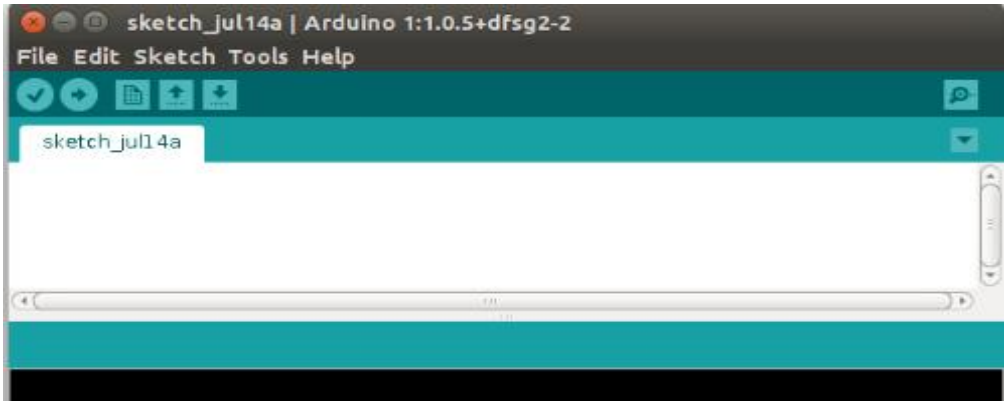

Figura 5 - Interface do ambiente de desenvolvimento
Fonte: Autoria própria

2.3. PHP

Além da linguagem especial do Arduíno, neste trabalho será utilizada a linguagem de programação PHP (PHP *Hypertext Preprocessor*), para fazer a interface a ser utilizada pelo usuário. A mesma é uma linguagem de script, com código livre e muito utilizada para o desenvolvimento *Web*. O que difere o PHP do Java no lado do cliente é que o programa é executado no servidor, gerando um código HTML que é então enviado para o cliente, que recebe os resultados da execução desse script, porém não tem conhecimento do código fonte (PHP, 2014).

O PHP é uma linguagem simples, para iniciantes e, que possui ao mesmo tempo, inúmeros recursos para programadores profissionais. Vasto material é encontrado para a integração PHP/Arduíno (SARIK E KYMISSIS 2010), fato esse que motivou a escolha do PHP. Com a linguagem é possível gerar páginas com conteúdo dinâmico para internet, coletar dados de formulário e enviar ou receber *cookies*. Outro fator que interferiu na escolha é o fato desta linguagem ser compatível com qualquer sistema operacional moderno e a maioria dos servidores *Web* atuais, permitindo que o programador escolha entre programação estruturada, orientada a objetos, ou uma mistura desses dois gêneros, de acordo com a facilidade do mesmo. Por fim, PHP possui suporte a banco de dados, o que se mostrou útil para o presente projeto.

2.4. APACHE

O Apache é um servidor de páginas *Web* e, igualmente a todo *software/hardware* que é utilizado nesse projeto, é *de código livre*, mantido por uma comunidade de desenvolvedores (APACHE, 2012).

Algumas características que tornam esse servidor atrativo são suporte à outras plataformas de código livre e também a proprietárias. Sua estruturação é feita em módulos e o suporte à várias linguagens de programação, assim como o controle de acesso e a possibilidade de utilização de criptografia por meio de certificados digitais são outras vantagens do Apache.

2.5. TWITTER BOOTSTRAP

Quando vamos desenvolver um projeto *web*, como em um sistema, por exemplo, é importante pensar em navegação, desenvolver padrões e estar preparado para novas telas e mudanças. A solução é criar um "tema" para a interface, ou seja, criar todos os elementos (botões, formulários e etc.) e chama-los quando necessário. Isso permite que novas telas sejam criadas com facilidade e também torna o trabalho mais produtivo. (MARQUES, 2014)

O Twitter Bootstrap é um *framework* que auxilia a produtividade dos desenvolvedores, com um visual bonito e agradável de forma simples e flexível, permitindo manter mais foco no que está se desenvolvendo. Ele foi projetado para auxiliar pessoas de todos os níveis e foi desenvolvido para navegadores modernos, mas até mesmo os navegadores mais antigos suportam sua funcionalidade.

Ele utiliza programação responsiva, com dezenas de componentes funcionais totalmente prontos para uso, além dos *plug-ins* em jQuery.

Uma de suas características, que torna vantajoso seu uso, é a sua licença de código livre, que significa que, além de grátis, ele possui uma comunidade de colaboradores que o torna melhor e aumenta ainda mais as chances de se encontrar uma boa documentação.

2.6. SISTEMA DE GERENCIAMENTO DE BANCO DE DADOS - SGBD

O sistema de banco de dados está presente no cotidiano da sociedade. Estes são utilizados de formas simples e complexas, desde uma agenda que armazena nomes, endereços e telefones de diversas pessoas até sistemas bancários ou de caixas eletrônicos. (ALECRIM, 2005).

"framework": Uma estrutura básica de um Sistema, conceito, ou texto.Oxford Dictionaries. Oxford University Press. http://www.oxforddictionaries.com/us/definition/american_english/framework (Acessado em: 29 Junho 2014).

Um sistema de gerenciamento de banco de dados (SGBD) pode ser definido como:

> *“Uma coleção de programas que permite aos usuários criar e manter um banco de dados. O SGBD é, portanto, um sistema de software de propósito geral que facilita os processos de definição, construção, manipulação e compartilhamento de bancos de dados entre vários usuários e aplicações. A definição de um banco de dados implica especificar os tipos de dados, as estruturas e as restrições para os dados a serem armazenados em um banco de dados" (ELMASRI, 2005, p. 3).*

Pela sua funcionalidade e praticidade foi escolhido o MySQL para utilização neste projeto.

O MySQL é um dos mais populares sistemas de gerenciamento de banco de dados que existem, devido ao fato de ser otimizado para aplicações *web* e amplamente utilizado na internet. O MySQL é muito utilizado com a linguagem de programação PHP.

A aplicação é suportada por diferentes sistemas operacionais, dentre eles, Windows e Linux e, assim como as demais aplicações a serem utilizadas neste projeto é um *software* livre e pode ser utilizado sem ônus por qualquer um que queira.

2.7. APRENDIZAGEM

Aprendizado é a capacidade de se adaptar, de modificar e melhorar um comportamento e suas respostas, o que é uma das propriedades mais importantes dos seres ditos inteligentes (OSÓRIO, 1999). A capacidade de aprender está ligada diretamente a alguns itens: adaptação e mudança de comportamento de maneira a evoluir; correção dos erros cometidos no passado, de modo a não repeti-los no futuro; melhoria do desempenho do sistema como um todo; interação com o meio, pois é através deste contato com o mundo que nos cerca que se podem trocar experiências e assim adquirir novos conhecimentos; representação do conhecimento adquirido - guardar uma massa muito grande de conhecimentos requer uma forma de representar

estes conhecimentos que permita ao sistema explorá-los de maneira conveniente (OSÓRIO, 1999).

O Homem nasce apto a iniciar seu aprendizado, ampliando seu conhecimento através de reordenações sucessivas, entretanto as máquinas não possuem programa inicial para aprender, e precisam de um para fazê-lo.

2.7.1. Sistema ABC

O processo normal de criação de regras é aquele onde o habitante é quem cria as regras, inserindo-as em um sistema. O sistema ABC (Automação Baseada em Comportamento) (TONIDANDEL, TAKIUCHI e MELO, 2004), o qual aprende regras em função do comportamento do habitante, foi testado e demonstrou através de simulações que é possível reverter o processo normal de criação de regras.

A arquitetura do sistema ABC, que será resumidamente descrita a seguir, define a existência de sensores (detectores de presença, medidores de temperatura, medidores de luminosidade, etc.), atuadores (interruptores de luz, ar-condicionado, etc.), bancos de dados e demais elementos necessários para criação e controle das regras.

Para cada atuador usado no sistema de controle da casa existe um banco de dados de aquisição de comportamento, o qual é alimentado com eventos e os respectivos dados dos sensores vinculados ao atuador. Como exemplo, pode-se ter o atuador “Ar-condicionado” e os sensores “Temperatura”, “Luminosidade”, “Horário” e “Presença” (o nome do sensor é utilizado como nome do atributo, portanto o sensor “Presença” que identifica se existe a presença ou não do habitante, terá seu valor refletido no atributo “Presença”, neste caso com valores “Sim” ou “Não”). Quando o estado do Ar-condicionado muda, por ação do habitante, os dados do próprio atuador acrescidos dos dados dos sensores são armazenados em uma linha do banco de dados de aquisição de comportamento.

Quando o banco de dados atinge o valor configurado de eventos armazenados, o mesmo é inserido no algoritmo de aprendizado com árvores de indução ID3 (QUINLAN, 1990), o qual generaliza os dados e cria as regras armazenadas no banco de dados de regras ativas. Uma regra aprendida

poderia ser se temperatura = alta, e horário = noite e luminosidade = alta e presença = sim, então, ar condicionado ligado.

Assim, quando os sensores indicam os valores presentes na regra o ar-condicionado é automaticamente ligado pelo Sistema ABC. Ou seja, a partir do momento em que novos eventos acontecem, de acordo com as ações do habitante, é feita uma varredura no banco de dados de regras.

Existe outro banco de dados onde estão as regras de segurança. Nele podem existir regras do tipo: Fogo = Sim ENTÃO Energia = Desligada.

A manutenção das regras é simples, quando uma regra fica sem ser utilizada por um valor de tempo pré-configurado, ela é removida. Também é possível remover regras manualmente pelo habitante.

Apesar de muito interessante, o sistema possui limitações. Ele não detecta sequências causais de eventos no tempo. Se um evento de atuador acontece depois de pouco ou muito tempo de um evento de sensor isto não é considerado. Neste sistema, com o uso do ID3, é possível trabalhar somente com variáveis lógicas, não sendo possível trabalhar com valores contínuos. Outra restrição é o fato de regras criadas pelo ID3 se tornarem diretamente ativas, isto pode desagradar o habitante da casa.

3. DESENVOLVIMENTO

Para melhor entendimento do presente trabalho, dividiu-se o desenvolvimento do projeto em Seções. Na Figura 6 está uma ilustração das relações entre as ações de usuário, linguagens e banco de dados.

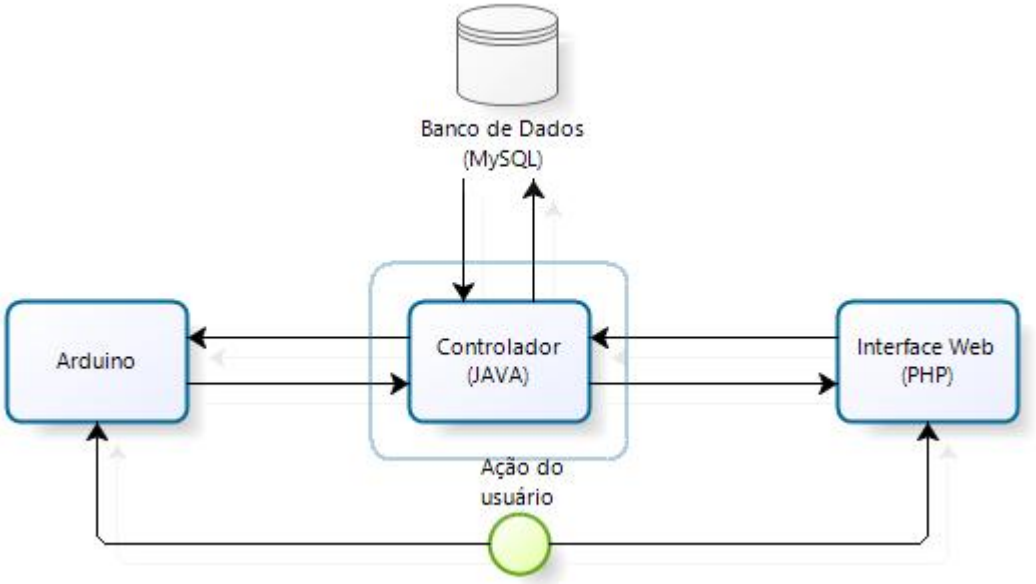

Figura 6 - Fluxograma das relações entre todos os estudos
Fonte: Autoria própria

A parte que fala sobre o Arduíno está na seção 3.1, e ilustra a modelagem e programação do *Hardware* Arduíno. A seção 3.2 exibe a modelagem e implementação da Inteligência, já voltado para o conteúdo do Controlador (JAVA) e do banco de dados. E a seção 3.3 apresenta o desenvolvimento e a interface da aplicação web.

3.1. *HARDWARE*

O *Hardware* Arduíno utilizado no desenvolvimento desse projeto foi o UNO. Comparado aos outros modelos de Arduíno ele é um hardware simples, barato, podendo ser comprado por pessoa física ou jurídica a valores entre R$ 69,00 e R$ 119,00 em sites nacionais e internacionais. Ele atende as necessidades iniciais para o desenvolvimento proposto, além de ser um kit de

desenvolvimento de fácil entendimento. Com esse *hardware* podemos controlar qualquer dispositivo eletrônico.

Neste projeto o Arduíno ficará encarregado de receber instruções e executá-las para o controle das lâmpadas de uma residência.

Inicialmente o Arduíno deve ficar ligado a um computador/servidor para os experimentos, no entanto é possível adquirir um novo componente, o *shield* de *ethernet*, para que se utilize uma bateria e a comunicação através de uma rede local.

A seguir estão apresentados os materiais utilizados e como é feita a montagem dos componentes no Arduíno.

3.1.1. Materiais utilizados.

- Arduíno (*Hardware*);
- Cabo Serial para USB;
- *Protoboard* (Matriz de contatos para suporte e montagem de protótipos, apresentada na Figura 7);

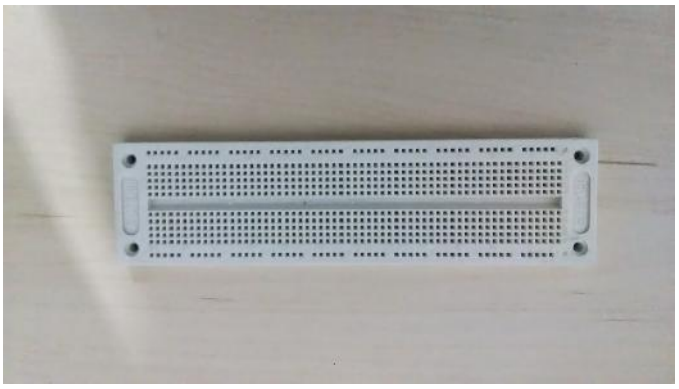

Figura 7 – Protoboard
Fonte: Autoria própria

- Cabos de conexão (Para fazer conexão dos componentes com a placa);
- Três LEDs (Usados para representar as lâmpadas no estudo de caso/protótipo);
- Três Chaves Tácteis (Ou Botões, representando o interruptor no estudo de caso/protótipo, como mostra a Figura 8).

Figura 8 - Chaves Tácteis
Fonte: Autoria própria

3.1.2. Montagem do Protótipo

Para conectar um led, ao *protoboard*, lembrando-se de que um led tem um lado positivo e o outro negativo, são utilizados os fios conectores para ligar o positivo do led na porta 13 e o negativo no terra ou *Ground*. Feito isso o circuito de um LED está feito, sendo necessário replicar isso para outros componentes conectando respectivamente na porta 12 e 11.

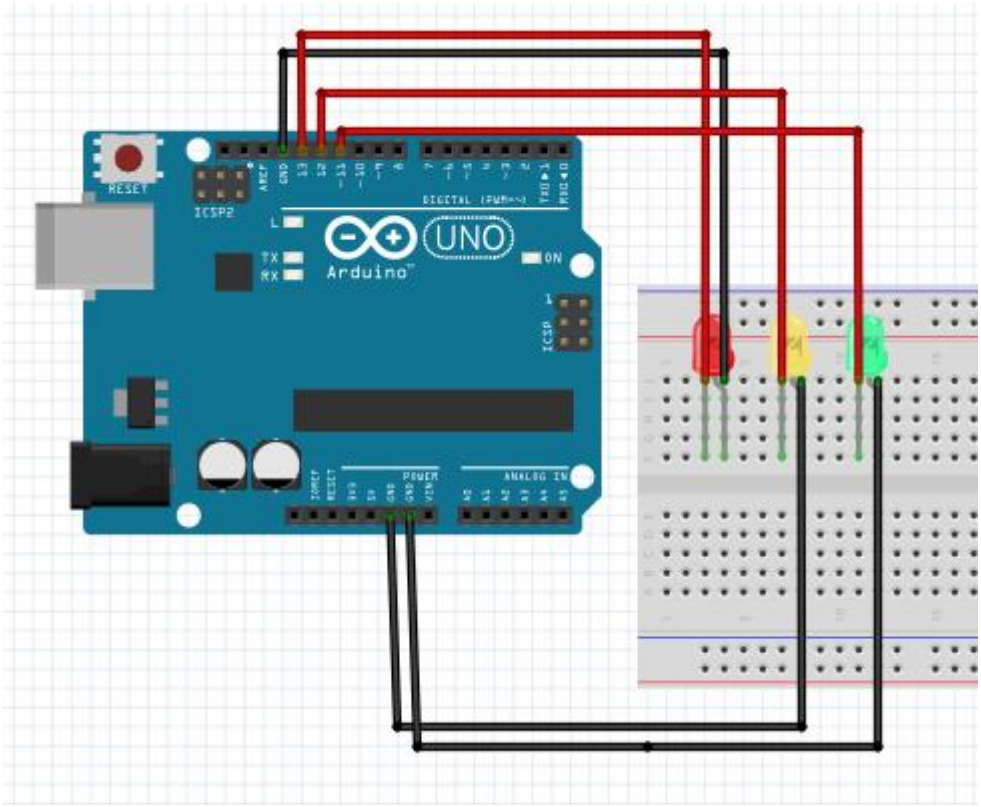

Figura 9 - Montagem do Protótipo
Fonte: Autoria Própria

Na representação da Figura 9, os fios conectores vermelhos representam a conexão com as portas digitais e os fios pretos representam a conexão GND.

Nota-se que o Arduíno UNO possui apenas três entradas GND, para ampliar é utilizado o *protoboard.* Ao conectar apenas um GND a uma linha do *protoboard* estende-se sua porta e se pode utilizar mais componentes ligados ao terra, como demonstrado na Figura 10.

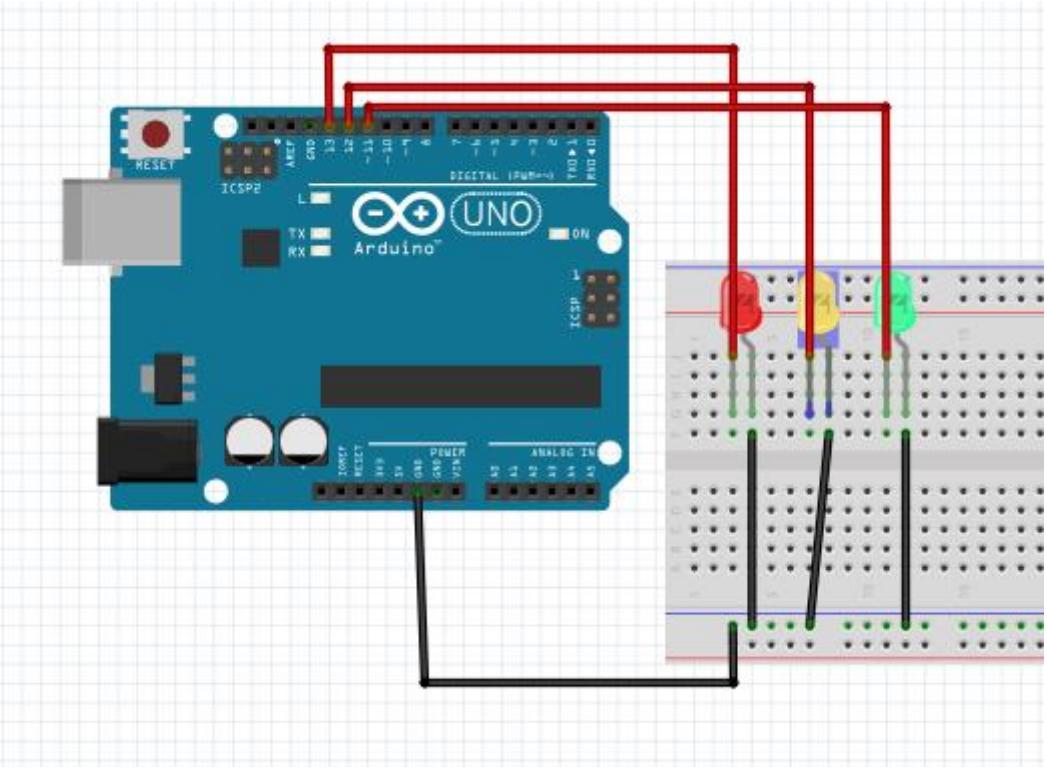

Figura 10 - Circuito Otimizado GND
Fonte: Autoria Própria

Após ampliar o circuito podemos conectar os botões. Estes não tem lado positivo ou negativo, por isso indifere qual dos dois lados será utilizado para porta ou para o GND.

Depois de conectar o botão e ligar um de seus lados ao GND e o outro na porta 3, o circuito de um botão está pronto. Após isto replica-se para os outros e liga-os respectivamente nas portas 4 e 5.

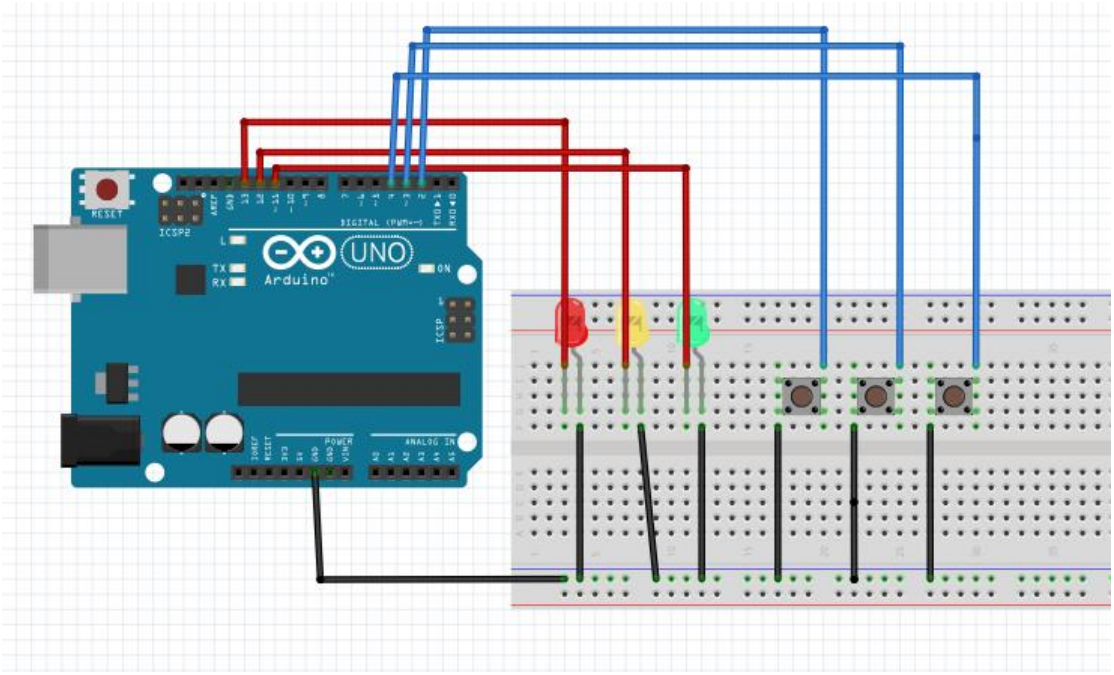

Figura 11 - Circuito Completo
Fonte: Autoria Própria

Na representação da Figura 11, os fios azuis representam as conexões com as portas digitais. Os botões e os leds estão ligados a portas digitais, as quais vão do 1 ao 13. Os componentes não precisam ser necessariamente ligados às portas descritas aqui, a escolha de números distantes se deu para diferenciar os botões e os leds.

As portas 0 e 1 são utilizadas como canais de recepção (RX) e transmissão (TX) de dados, desta forma, se houvesse uma necessidade de implementar uma solução com o uso de *Bluetooth*, como por exemplo, comando de voz, essas portas poderiam ser utilizadas.

As portas analógicas que se encontram do outro lado da placa do Arduíno podem também ser utilizadas como digitais também.

Foram realizados testes com sensores de movimento e luminosidade, individualmente e simultaneamente para verificar o funcionamento das portas analógicas em modo digital e melhorar o desempenho do sistema, porém, como o foco do trabalho é o baixo custo, os sensores foram considerados como opcionais.

3.1.3. Programação inicial

Depois de finalizada a parte de hardware, são abordadas as duas etapas de desenvolvimento de software. Com o protótipo montado na *protoboard,* a IDE do Arduíno é utilizada para a programação de cada componente.

Para simular a utilização dos interruptores de uma casa, foram conectadas ao *protoboard* chaves tácteis analógicas, ou botões. Estas chaves são utilizadas para simular uma interrupção que muda o estado do led para ligado ou desligado. Os leds serão a representação das lâmpadas de uma residência dentro do protótipo proposto.

Para programar no Arduíno é preciso ter conhecimentos de lógica de programação e linguagem C, porém esta, possui algumas diferenças da linguagem do Arduíno. Um programa no Arduíno deve possuir duas funções principais, as funções *void setup()* e *void loop().* Sem elas a IDE não é capaz de identificar a linguagem e não realiza a compilação do código. Na Figura 12 é exibido o método *void setup().*

```
void setup() {

  inputString.reserve(200);
  pinMode(quarto1,OUTPUT);
  pinMode(sala,OUTPUT);
  pinMode(cozinha,OUTPUT);
  pinMode(banheiro,OUTPUT);

  pinMode(btn13,INPUT);
  digitalWrite(btn13,HIGH);
  pinMode(btn12,INPUT);
  digitalWrite(btn12,HIGH);
  pinMode(btn11,INPUT);
  digitalWrite(btn11,HIGH);
  pinMode(btn10,INPUT);
  digitalWrite(btn10,HIGH);

  Serial.begin(9600);
}
```

Figura 12 - Trecho programação Arduíno – void setup()
Fonte: Autoria Própria

A função *void setup()* é como uma função de inicialização, o que está contido nela executará apenas uma vez no programa, no momento da primeira execução, por este motivo, tudo que for necessário para a execução do mesmo

deve ser declarado nessa função. Já a função *void loop()* deverá conter todas as instruções que o programa deverá executar, conforme a Figura 13.

```
void loop() {
  if (stringComplete) {
    inputString = "";
    stringComplete = false;
    comando= "";
    stringPorta = "";
    porta = 0;
  }

  int ler13 = digitalRead(btn13);
  int ler12 = digitalRead(btn12);
  int ler11 = digitalRead(btn11);
  int ler10 = digitalRead(btn10);
```

Figura 13 - Trecho programação Arduíno – void loop()
Fonte: Autoria Própria

Para o desenvolvimento deste trabalho também é utilizada a função void *serialEvent(),* apresentada na Figura 14, que é chamada após o *loop*. A *serialEvent* monitora a comunicação na porta serial e possibilita recebimentos e envios de dados.

```
void serialEvent() {
  while (Serial.available()) {
    char inChar = (char)Serial.read();
    inputString += inChar;

    delay(150);
    comando = inputString.substring(0, inputString.indexOf(';'));
    stringPorta = inputString.substring(inputString.indexOf(';')+1, inputString.length());

    if(comando == "LIGAR"){
      digitalWrite(stringPorta.toInt(),HIGH);

      if(stringPorta.toInt() == 13){
        btn13Clique = true;
      }

      if(stringPorta.toInt() == 12){
        btn12Clique = true;
      }

      if(stringPorta.toInt() == 11){
        btn11Clique = true;
      }

      if(stringPorta.toInt() == 10){
        btn10Clique = true;
      }
    }
```

Figura 14 - Trecho programação Arduíno – serialEvent()
Fonte: Autoria Própria

3.2. INTELIGÊNCIA E CONTROLE

Depois de realizada a parte de *hardware* e estruturação básica do código, a tarefa necessária é a programação da inteligência e do controle dos componentes do Arduíno.

Como descrito nas seções anteriores, o controle de equipamentos eletrônicos se dá pelo simples fornecimento, ou não, de energia elétrica aos componentes. O importante a partir daqui é elaborar um programa que contemple algumas técnicas para controle de forma que os leds sejam acessos ou apagados conforme configuração ou de modo autônomo. Após o ajuste e aceitação, modificações de robustez e escala se farão necessárias.

Uma solução autônoma deve ser transparente o suficiente para que não seja tentador ao usuário uma intervenção desnecessária no funcionamento do sistema, contudo, também não pode ser engessada de forma que os desejos do usuário tenham que se adaptar ao sistema e não o contrário.

Neste caminho, optou-se por definir uma agenda que deve representar o perfil do usuário, com os horários que as lâmpadas devem ser ligadas ou desligadas.

Estes dados foram armazenados em um banco de dados e, para contornar as limitações do Arduíno, devido a não utilização do módulo *shield* de *ethernet*, foi implementada uma aplicação JAVA que envia as instruções no horário que o estado de cada lâmpada deve ser alterado e recebe instruções quando o usuário intervém ligando uma lâmpada usando o respectivo botão. Certamente há alternativas para contornar a não presença do módulo *shield,* como a não utilização de banco de dados, por exemplo. Na opção adotada a aplicação em JAVA faz o papel de *middleware* entre o banco de dados e o Arduíno. Na Figura 15 estão representados os métodos utilizados na aplicação.

> "middleware": Um *software* que atua como ponte entre um sistema operacional ou entre bancos de dados e aplicações. Oxford Dictionaries. Oxford University Press. http://www.oxforddictionaries.com/us/definition/american_english/middleware (Acessado em: 27 Junho 2014).

void	atualizaAgenda(int diaSemana) Atualiza o Array agenda.
void	close() Fecha a comunicação entre a porta.
void	enviaDados(java.lang.String dados)
void	escreverArquivoWeb(java.lang.String txt, java.lang.String arquivo)
void	inicializaar()
void	inicializaar(int paramX, int perfil1, int perfil2, int perfil3)
void	initialize() Metodo que inicia as variaveis da porta Serial.
void	run()
void	serialEvent(gnu.io.SerialPortEvent oEvent) Lê a porta serial, e insere na tabela historico.
void	verificaAgenda(java.lang.String horario, int diaSemana) Verifica a Agenda para ligar/desligar algum componente.
void	verificarHorario() Verifica o horario do sistema para atualização da agenda e verificar se está na hora de ligar algum componente.
void	veririficaArquivoWeb()

Figura 15 - Métodos em linguagem Java
Fonte: Autoria Própria

O fluxo é executado na seguinte ordem:

1 - Inicializar() - Carrega o vetor com os horários da tabela agenda;

2 - initialize() - Carrega todas as informações sobre portas do Arduíno;

3 - run() – *Thread* (processo) que verificará o horário do sistema;

4 - verificarHorario() - se o horário for igual 00:00:00 o método atualizaAgenda() será executado;

5 - verificaAgenda() - executa o veririficaArquivoWeb() que verificará se houve alguma alteração por meio da aplicação web;

5.1 - compara o horário do sistema com o horário que foi carregado no vetor;

5.2 - se o horário do sistema for igual a um horário contido no vetor, o método enviaDados() é executado enviando os dados para a porta serial contendo o componente e qual status assumirá.

Quando o usuário intervém, ligando ou desligando um led durante o dia, a aplicação recebe uma mensagem através da porta serial com os dados de qual componente está sendo alterado e para qual estado ele foi mudado (ligado/desligado). Para que esse processo ocorra sem que trave a aplicação, foi utilizada uma *Thread*, de forma que as comparações de horários, e cadastro de um novo horário no histórico podem ser executados simultaneamente.

O horário pré-definido para a atualização uma agenda foi 00h00min (meia noite) quando então é chamada uma *stored procedure* no banco de dados MySQL. Na figura 16 está o código que calcula o horário para ligar a luz.

```
SELECT SEC_TO_TIME(AVG(TIME_TO_SEC(tb1.datahora)))
    INTO horaLigarCalculada
    FROM (SELECT h.datahora
        FROM tcc.historico h
            WHERE h.idcomponente = componente
            AND h.estado = 1
            AND TIME(h.datahora) BETWEEN
                (SELECT addtime(a.hora,-(SEC_TO_TIME(tolerancia)))
                    FROM tcc.agenda a
                        WHERE a.idagenda = impar
                        AND a.idcomponente = componente) AND
                (SELECT addtime(a.hora,(SEC_TO_TIME(tolerancia)))
                    FROM tcc.agenda a
                        WHERE a.idagenda = impar
                        AND a.idcomponente = componente)
            ORDER BY h.datahora DESC
            LIMIT 5) AS tb1;

IF (horaLigarCalculada is NULL) THEN
    SET horaLigarCalculada = horaLigarAgenda;
END IF;
```

Figura 16 - Stored Procedure de cálculo da Inteligência – Ligar
Fonte: Autoria Própria

O procedimento para desligar tem apenas a modificação do valor do estado (de 1 para 0) e a utilização de variáveis com nomes sugestivos à ação de desligar, como é ilustrado na Figura 17.

```
SELECT SEC_TO_TIME(AVG(TIME_TO_SEC(tb2.datahora)))
    INTO horaDesligarCalculada
    FROM (SELECT h.datahora
        FROM tcc.historico h
            WHERE h.idcomponente = componente
            AND h.estado = 0
            AND TIME(h.datahora) BETWEEN
                (SELECT addtime(a.hora,-(SEC_TO_TIME(tolerancia)))
                    FROM tcc.agenda a
                        WHERE a.idagenda = par
                        AND a.idcomponente = componente) AND
                (SELECT addtime(a.hora,(SEC_TO_TIME(tolerancia)))
                    FROM tcc.agenda a
                        WHERE a.idagenda = par
                        AND a.idcomponente = componente)
            ORDER BY h.datahora DESC
            LIMIT 5) AS tb2;

IF (horaDesligarCalculada is NULL)  THEN
    SET horaDesligarCalculada = horaDesligarAgenda;
END IF;
```

Figura 17 - Stored Procedure de cálculo da Inteligência – Desligar
Fonte: Autoria própria

Ao decorrer desta seção será explicada com detalhes a inteligência proposta para controlar o aprendizado da rotina de três perfis pré-estabelecidos e os demais recursos necessários para seu pleno funcionamento.

Para desenvolver o algoritmo de aprendizado foi preciso definir anteriormente o que deveria ser aprendido e o que deveria ser descartado.

Para uma melhor demonstração da adaptabilidade do sistema, foram assumidos três perfis de usuários. Cada um com rotinas diferentes que variavam de acordo com uma tolerância máxima proporcionalmente definida pelo tempo que a lâmpada permanecia acesa.

Os perfis foram identificados como:

- Pessoa que trabalha;
- Pessoa que estuda de manhã;
- Pessoa que trabalha e estuda.

Foram assumidos os seguintes intervalos de luz acesa, também chamados de ciclos, para cada perfil:

- Pessoa que trabalha:
    - 07:00:00 – 08:00:00;
    - 18:30:00 – 21:00:00.

- Pessoa que estuda de manhã:
    - 06:30:00 – 07:30:00;
    - 18:00:00 – 23:30:00;
- Pessoa que trabalha e estuda:
    - 07:10:00 – 07:45:00;
    - 18:15:00 – 18:30:00;
    - 23:00:00 – 23:50:00;

Para a definição das rotinas foi utilizada uma tabela com os horários dos eventos que foi chamada de Agenda. Para cada perfil há um código identificador da Agenda, um código identificador do perfil, um valor em formato de horas, minutos e segundos e um identificador do tipo de evento. A tabela 2 exibe os valores inseridos na Agenda.

Tabela 2 - Exemplo de valores nos perfis criados
Fonte: Autoria própria

Agenda	Perfil	Hora	Comando
1	1	18:00:00	LIGAR
2	1	19:00:00	DESLIGAR
1	2	07:30:00	LIGAR

Foi definido que cada perfil atuaria em uma lâmpada apenas e que a agenda controlaria os eventos de ligá-la e desligá-la.

O controlador do Arduíno foi desenvolvido na linguagem de programação Java pela facilidade de interagir com dispositivos eletrônicos do computador utilizando os recursos do sistema operacional com bibliotecas nativas.

Inicialmente as tentativas de comunicação com a porta serial do Arduíno foram feitas em PHP, a mesma linguagem utilizada para interagir com o usuário e para parametrizar inicialmente o sistema, entretanto o PHP não possui tantas facilidades quanto o Java no monitoramento de dispositivos físicos, por isso da escolha do Java para controlar o Arduíno.

O controlador foi programado para respeitar a interação do usuário nos horários não cadastrados na agenda e controlar o Arduíno conforme a agenda vai se modificando pela adaptação aos hábitos do usuário.

O monitoramento é feito escutando a porta ligada ao Arduíno e também por um laço controlado por tempo que verifica se o horário atual é igual ao horário gravado na agenda, executando o comando que estiver gravado no horário monitorado.

Outra tabela, chamada de Histórico, foi criada para registrar cada evento ocorrido. Os eventos gerados pelo usuário ou pelo controlador do Arduíno são identificados por um identificador único, um código identificador do perfil, um valor de data, hora, minuto e segundo e um identificador do estado em que se encontra o componente, no caso, a lâmpada.

A tabela 3 mostra um exemplo dos registros da tabela Histórico.

Tabela 3 - Exemplo da tabela Histórico
Fonte: Autoria própria

Histórico	Perfil	Data Hora	Estado
1	1	2014-05-01 17:59:42	1
2	1	2014-05-01 18:55:01	0
3	3	2014-05-04 07:30:00	1

A tabela Histórico é utilizada pelo algoritmo de aprendizado para adaptar a tabela Agenda aos horários mais próximos das interações do usuário. O registro é feito por data e hora para facilitar futuras funcionalidades ao algoritmo, como tipo de dia de semana, contudo, nesta etapa foi assumido que todos os dias são considerados como dia útil e que todo o histórico influencia no resultado do algoritmo.

Conforme parâmetros informados pela interface do usuário, o histórico é utilizado para gerar um novo horário na agenda. Esses parâmetros controlam a tolerância máxima entre o horário gravado na agenda e o horário da interação do usuário com o sistema. A tolerância pode variar de 0 segundo a até no máximo o mesmo tempo que a lâmpada fica acesa. Caso o parâmetro seja definido para 0 segundo e não exista nenhuma interação do usuário que coincida com a agenda, o valor da agenda é adotado.

Para aproveitar os recursos consumidos pelo banco de dados, evitar troca de informações entre canais de comunicação e pelo maior desempenho, foi implementado o algoritmo na linguagem PL/SQL.

O algoritmo de aprendizado varre os registros da tabela Histórico e cria uma média entre os horários onde houveram mudança de estado em cada componente. O processo é feito por ciclo, utilizando um laço que identifica quantos ciclos cada perfil possui.

O ciclo é definido por um comando de ligar e um comando de desligar, nesta sequência. A tolerância nunca será maior que o intervalo do ciclo e é aplicada tanto no evento de ligar quanto no de desligar. Uma tolerância de 100% em um intervalo de uma hora quer dizer que o algoritmo vai considerar valores até meia hora antes e até meia hora depois do evento.

No código apresentado na Figura 18 pode ser melhor verificado o processo de ciclos e a utilização do parâmetro de tolerância.

```
SET regraX = x/100;

SELECT (max(a.idagenda)) INTO ciclos FROM tcc.agenda a WHERE a.idcomponente=componente;

SET impar = 1;

WHILE impar <= ciclos DO

    SELECT a.hora
        INTO horaLigarAgenda
        FROM tcc.agenda a
            WHERE a.idagenda = impar
            AND a.idcomponente = componente;

    SELECT (TIME_TO_SEC(a.hora) - (TIME_TO_SEC(horaLigarAgenda)))
        INTO tolerancia
        FROM tcc.agenda a
            WHERE a.idagenda = impar+1
            AND a.idcomponente = componente;

    SET tolerancia = (tolerancia/2)*regraX;
```

Figura 18 - Código referente aos ciclos da agenda e tolerância
Fonte: Autoria própria

Para o funcionamento do sistema existem várias outras tabelas que podem ser verificadas na Figura 19.

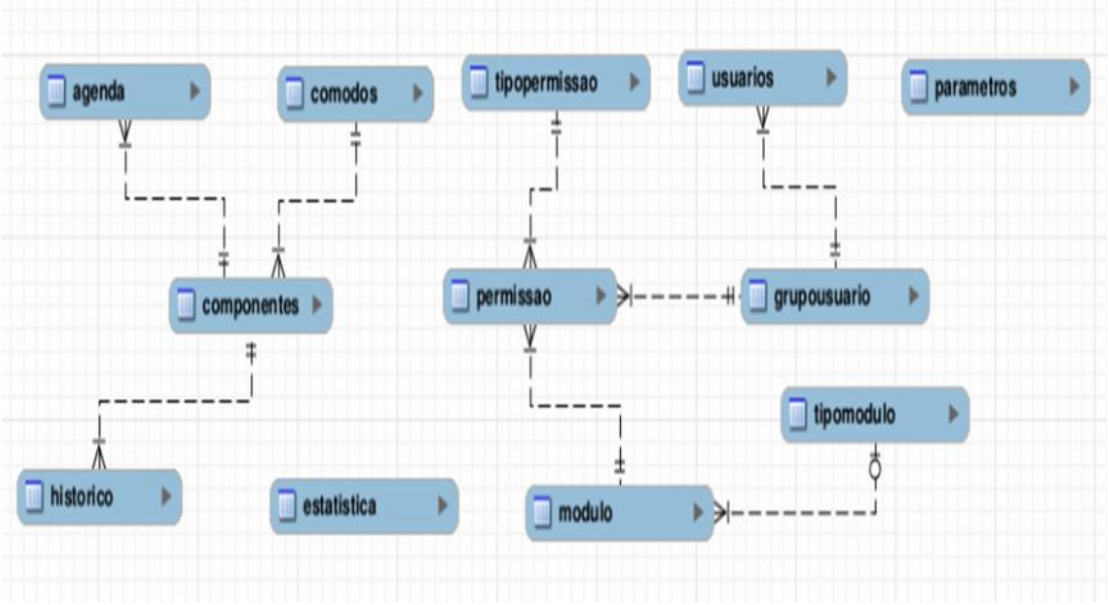

Figura 19 - Tabelas do banco de dados
Fonte: Autoria própria

As tabelas usuários, grupousuario, permissao, modulo, tipomodulo e tipopermissao controlam o acesso da interface com o usuário, permitindo a interação com as tabelas comodos, componentes, agenda e parâmetro. A tabela estatística grava o histórico da alteração da agenda.

A alguns grupos de usuários, como o dos usuários, não são permitidas alterações em parâmetros ou quaisquer tipo de cadastro, somente a opção de ativar ou desativar componentes e ligar ou desligar os mesmos.

3.2.1. Java

Nesta subseção serão detalhados o controlador e o simulador desenvolvidos em linguagem Java.

3.2.2. Controlador

Ao decorrer do projeto notou-se a necessidade de desenvolver um controlador, cuja funcionalidade é fazer uma ligação entre os dados enviados pelo Arduíno e os dados cadastrados no banco de dados. O mesmo foi desenvolvido na linguagem Java. Na Figura 20 um diagrama demonstra essa ligação.

O controlador ficará lendo a porta serial e enviando dados para ela, ao iniciar o controlador uma busca no banco de dados é realizada para trazer todos os horários da agenda para um vetor, esses horários são apenas do dia da semana no qual o sistema está funcionando, por exemplo, se for uma segunda-feira os horários carregados no vetor serão apenas os cadastrados para esse dia. Isso deixa o vetor com menos dados e torna a comparação mais rápida.

Figura 20 - Diagrama de fluxo do Controlador
Fonte: Autoria Própria

Logo depois uma *Thread* é iniciada, que irá verificar o horário do sistema e irá comparar se o horário existe no vetor e, no momento em que o horário no vetor for igual o do sistema o controlador envia uma ação ser realizada pelo Arduíno, essa ação será de ligar ou desligar o LED, isso dependerá dos dados cadastrados na agenda.

As interferências causadas pelo usuário são cadastradas na tabela Histórico, a partir dos dados cadastrados, é que será definida a agenda.

É utilizado 00h00min (meia noite) como o horário da atualização da agenda. Quando esse horário for reconhecido pelo sistema ele irá atualizar a agenda para o próximo dia, por exemplo, se for segunda-feira o controlador carregará os dados de terça-feira, além de atualizar os novos horários do sistema, chamando a *stored procedure* que faz o cálculo a partir dos dados históricos, de acordo com as interferências do usuário.

3.2.3. Simulador

O Simulador foi desenvolvido para facilitar os testes e verificar se os leds acendem de acordo com os horários cadastrados na agenda, como o controlador é na linguagem JAVA o desenvolvimento da interface gráfica do simulador também foi adotado para essa linguagem. Com uma interface amigável e de fácil compreensão esse simulador proporciona algumas

funcionalidades que ajudam nos testes, como por exemplo, permitir que perfis sejam testados separadamente, escolher o nível das tolerâncias usadas para a interferência do usuário, permite também que o programa utilize o horário do sistema para realizar teste ou um horário fictício que passa mais rápido do que o tempo normal. As Figuras 21 e 22 ilustram a interface do simulador e uma captura de tela do funcionamento.

Figura 21 - Simulador Controlador Tela Inicial
Fonte: Autoria Própria

Figura 22 - Simulador Controlador Iniciado
Fonte: Autoria Própria

3.3. INTERFACE

Devido à necessidade de obter uma agenda inicial, com os horários pré-definidos pelo próprio usuário do sistema, foi criado uma interface WEB, onde o usuário terá acesso junto ao servidor e controlará todos os recursos do sistema. Além de criar essa agenda inicial, o usuário poderá adicionar, editar e excluir cômodos e componentes, quando necessário e acrescentar horários novos na agenda e também visualizar todos os componentes e verificar se estão ligados ou desligados e seus dias e horários de interações. Foi utilizado a linguagem de programação PHP, banco de dados MySQL e o Twitter Bootstrap como *framework*, devido a todos serem tecnologias *open source* e de fácil manipulação.

3.3.1. Acesso

A tela inicial do projeto é o Acesso de usuários, onde o usuário já cadastrado coloca seu nome de usuário e senha para ter acesso ao sistema. Como pode ser visto na Figura 23.

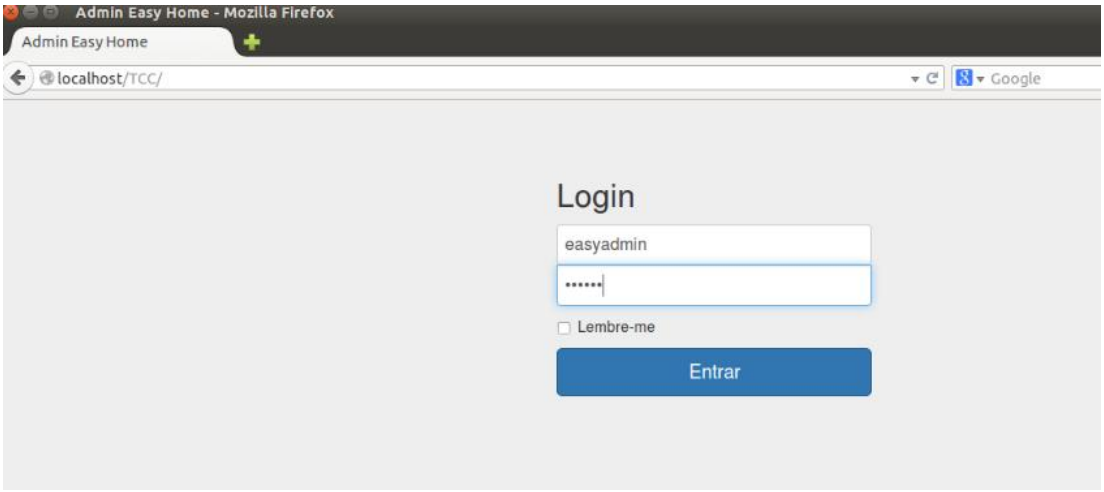

Figura 23 - Tela de Acesso
Fonte: Autoria própria

3.3.2. Opções do Menu:

As opções do Menu são:

- Home;
- Cadastros: onde tem as opções Acessos e Cômodos;
- Visualizar cômodos;

Ao clicar na opção Acessos, o usuário terá as opções: Usuário, onde ele poderá cadastrar novos usuários e editar ou excluir usuários já cadastrados, a opção Permissões, dará permissões aos grupos de usuário, em que cada grupo de usuários terá acessos diferentes, alguns com mais privilégios e outros com menos, na opção Grupo de usuários, o usuário poderá criar grupos de usuários e colocar os usuários cadastrados em cada grupo de acordo com os privilégios que eles tenham, conforme a Figura 24. Estas opções não estão implementadas, visto que estão previstas para trabalhos futuros onde o projeto não seja com intuito simples e de fácil manipulação.

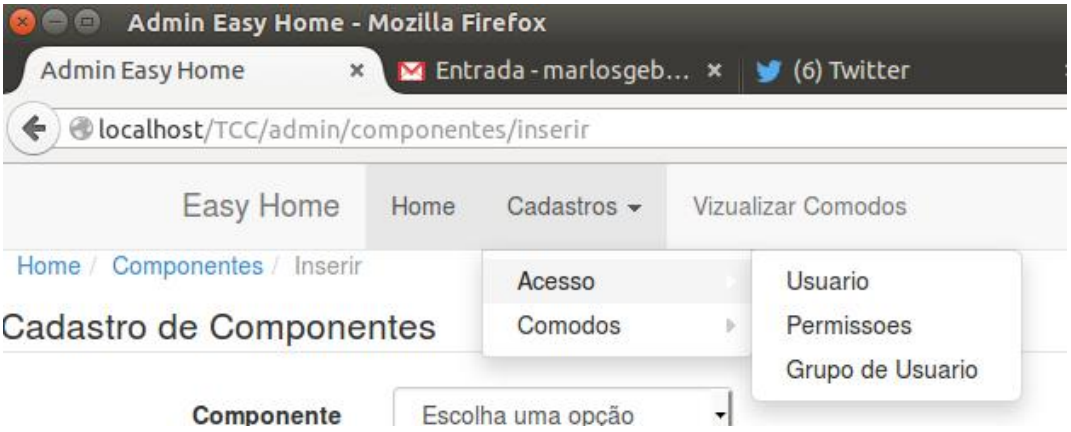

Figura 24 - Tela Acessos
Fonte: Autoria própria

Ao clicar em Cômodos, abrirá uma lista de submenu, conforme Figura 25, que trará Cômodos, Componentes e Agenda.

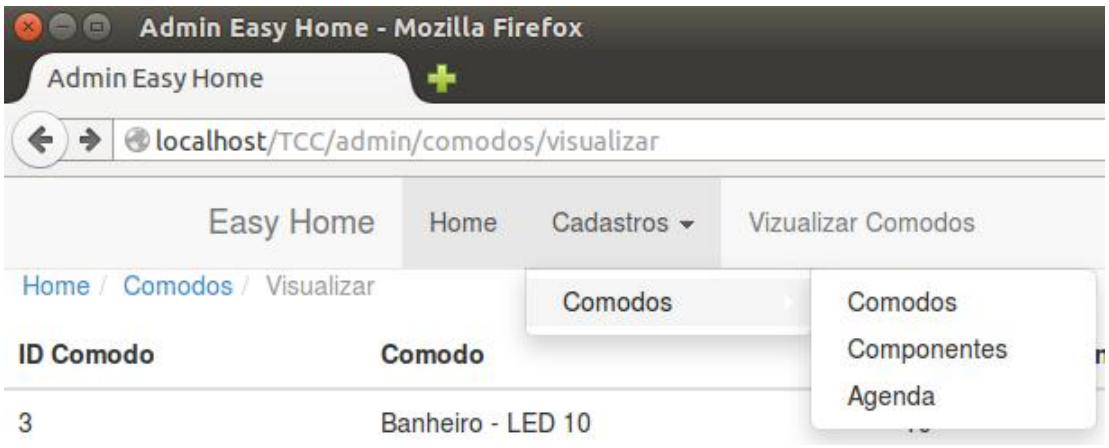

Figura 25 - Menu Cadastros
Fonte: Autoria própria

Ao selecionar a opção Cômodos obtêm-se a tela Listar Cômodos, como mostra a Figura 26, que lista todos os cômodos já cadastrados e traz as opções de editar ou excluir algum cômodo existente.

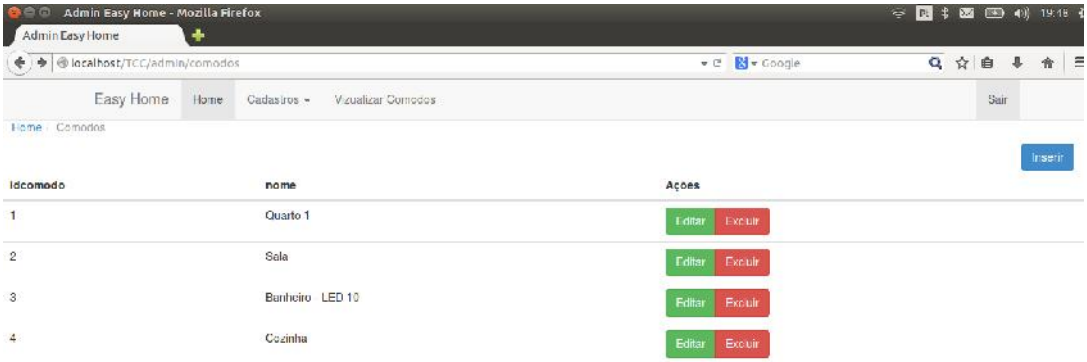

Figura 26 - Listar Cômodos
Fonte: Autoria própria

Nesta tela também existe a opção de inserir um novo cômodo que levará até a tela Cadastro de Cômodos, conforme a Figura 27, na qual o usuário irá apenas dar um nome ao componente e salvar, de uma maneira simples o usuário pode inserir ou editar um cômodo nesta mesma tela.

Figura 27 - Cadastro de Cômodos
Fonte: Autoria própria

Na opção Componentes, da mesma maneira que funciona a opção cômodos, teremos a tela Listar Componentes, conforme Figura 28, a qual irá mostrar uma lista com todos os componentes cadastrados, com os atributos respectivos, como ID do componente, que é a porta que o Arduíno vai utilizar para aquele componente, ID do cômodo onde o componente vai estar, nome e

status, além de trazer as opções de editar ou excluir algum componente já existente

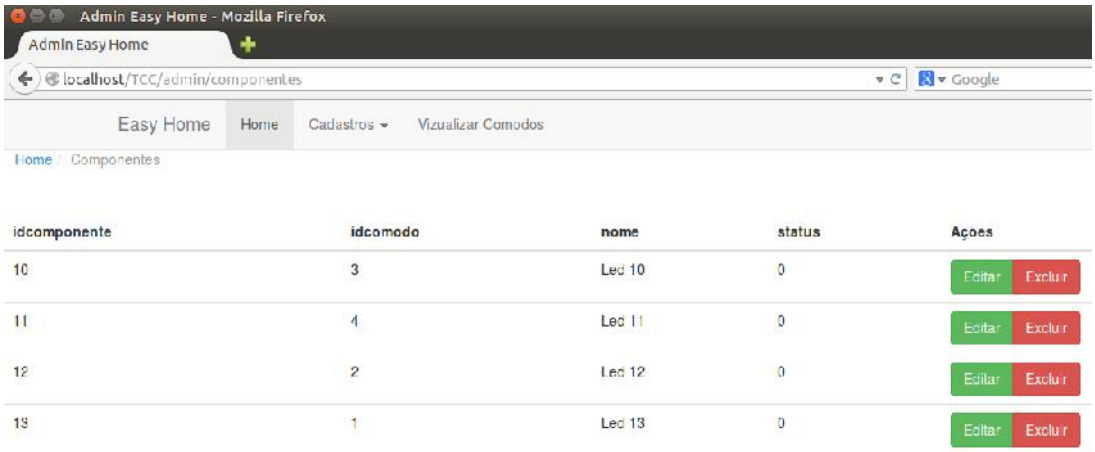

Figura 28 - Listar Componentes
Fonte: Autoria Própria

Há a opção de inserir, onde o usuário pode criar um novo componente no sistema, utilizando a tela Cadastro de componentes, que é ilustrado na Figura 29. Esta tela também serve para edição, onde o usuário irá colocar o nome do componente, escolher um cômodo já cadastrado para colocar onde o componente está instalado e alterar o Status.

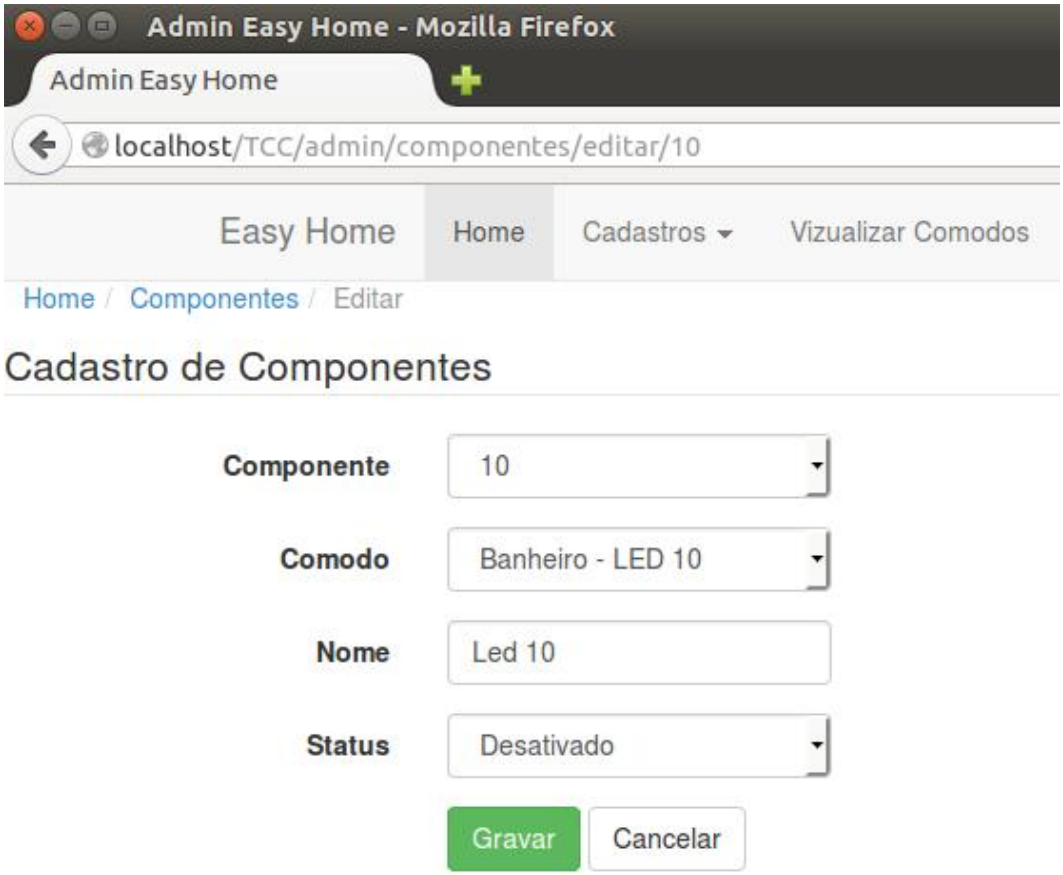

Figura 29 - Cadastro de Componentes
Fonte: Autoria Própria

Na opção Agenda, o sistema trará para o usuário uma lista com todas as entradas de todos os cômodos, bem como seus atributos, ID de agenda, ID do componente, dia da semana, hora e comando além das opções editar ou excluir, como mostra a Figura 30.

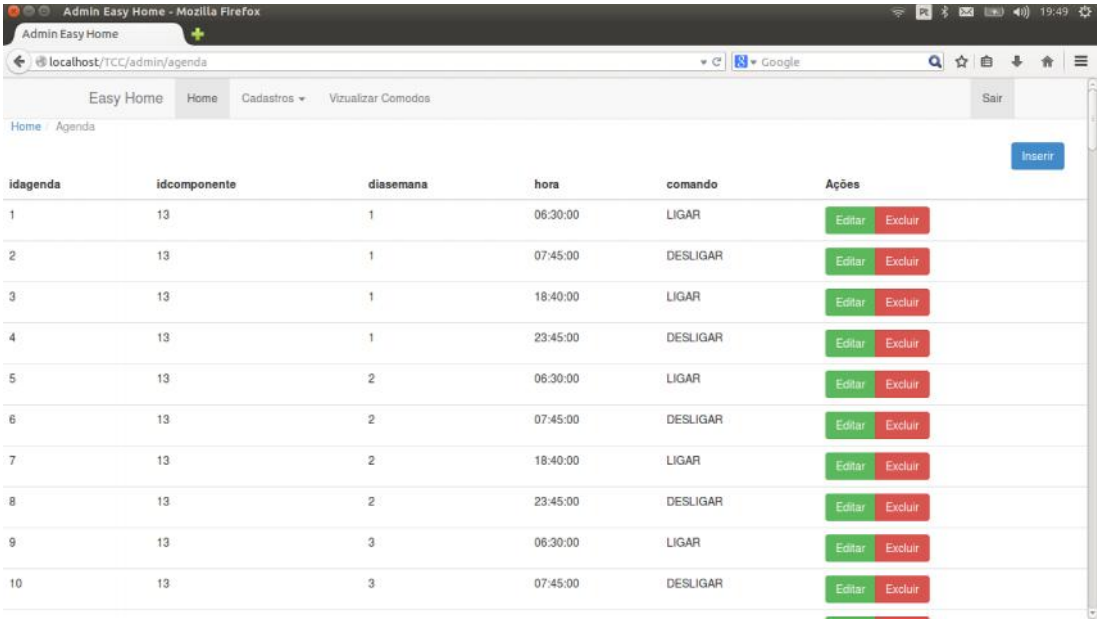

Figura 30 – Agenda
Fonte: Autoria Própria

Ilustrado na Figura 31, na mesma tela ainda tem-se a opção inserir, que possibilitará, por meio da tela Cadastro de agenda, ao usuário criar um novo horário na agenda, colocando o componente que irá receber o comando, o dia da semana, o horário e o comando (ligar ou desligar o led).

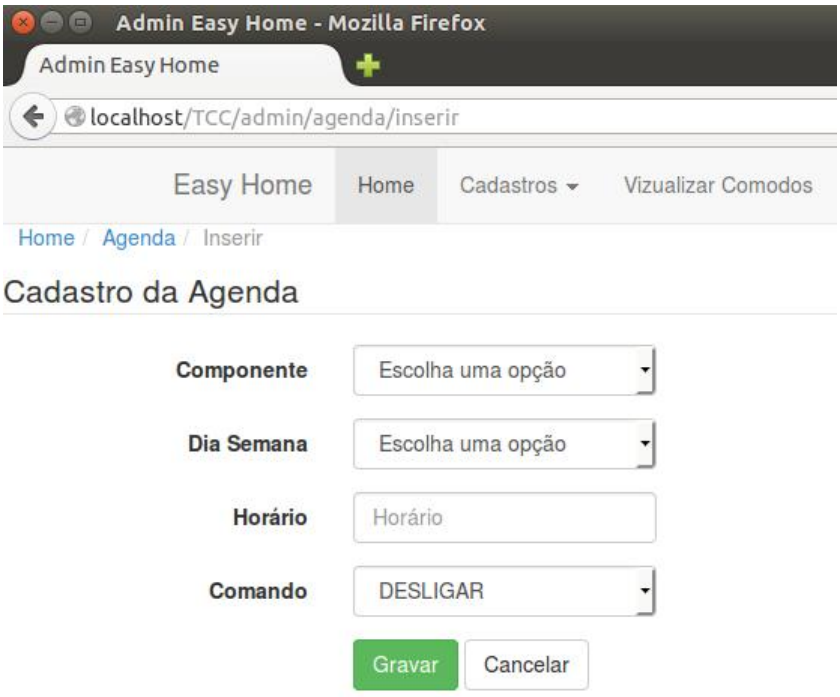

Figura 31 - Cadastro de Agenda
Fonte: Autoria Própria

O Menu Principal, existe a opção Visualizar cômodos, que contém todas as informações do sistema, nela estão as informações de cômodos, portas do componente e seu status, dando ao usuário uma visão geral do funcionamento de todos os cômodos e seus componentes. A tela possui o botão ligar/desligar, que permite o usuário alterar o status do componente, ligando ou desligando o led escolhido, como ilustrado na Figura 32.

Figura 32 - Visualizar Cômodos
Fonte: Autoria Própria

4. TESTES

Para a realização dos testes de forma mais veloz, entretanto sem a possibilidade de resultados diferentes da realidade, foi adicionada a opção de passar o tempo mais rápido no simulador. Desta forma podem-se simular dias em poucos minutos, preenchendo a tabela Estatística, criada especificamente para os testes, e verificando em tempo real a adaptação da agenda à rotina do perfil que estava configurado.

No simulador também é possível escolher qual LED especifico utilizar, o que ajudou a testar as diversas portas do hardware.

Na tabela Estatística são gravados os registros já calculados com o parâmetro selecionado. Além dessa tabela, uma versão adaptada da *store procedure* foi criada para facilitar a mudança dos parâmetros. A tabela estatística é alimentada conforme o código mostrado na Figura 33. Também pode ser verificada a atualização da agenda.

```
INSERT INTO `tcc`.`estatistica` (`idagenda`,`idcomponente`,`hora`,`estado`,`parametro_x`,`parametro_y`)
    VALUES (impar,componente,horaLigarCalculada,1,regraX,regraX);

UPDATE `tcc`.`agenda`
    SET `hora` = horaLigarCalculada
    WHERE `idagenda` = impar AND `idcomponente` = componente;
```

Figura 33 - Código de inserção e atualização das tabelas Estatística e Agenda
Fonte: Autoria própria

A *store procedure* com o algoritmo de aprendizagem foi chamada com parâmetros diferentes com o objetivo de verificar qual seria o parâmetro mais adequado à proposta do trabalho, que é a economia de energia.

Para simular o comportamento do algoritmo, foram realizados diversos testes. Foram assumidos que todos os dias seriam dia de semana e que somente um perfil seria utilizado para todos os testes, o Perfil 1, pois seus horários permitem uma melhor leitura dos gráficos.

Para os testes foram utilizados os aplicativos Notepad++ v6.5.5, Microsoft Office Excel 2007 e MySql Workbench 6.0.

O software Excel foi utilizado para criar os comandos SQL de inserção na tabela Histórico, pela facilidade de trabalhar com tipos de dados de data e hora. Após a conversão de uma coluna com valores no formato AM/PM para o

formato 24 horas é utilizada a função concatenar no Excel para gerar várias linhas de comandos SQL. Esses comandos são copiados para o aplicativo Notepad++ pois um dos caracteres da sintaxe do SQL não gera texto no Excel. Uma vez copiado ao Notepad++, utiliza-se o recurso de copiar em colunas para incluir o caractere “‘” antes e depois dos valores no formato data e hora.

Os comandos são copiados do Notepad++ para o MySql Workbench, já conectado ao banco de dados, e então são executados em lote. Cada dia de histórico será precedido pelo cálculo do dia anterior, com exceção do primeiro dia, utilizando o procedimento criado.

Para efeitos de comparação, foram assumidas constantes cujos valores não serão modificados entre os cenários, conforme a Tabela 4.

Tabela 4 - Constantes utilizadas nos testes
Fonte: Autoria Própria

Constantes			
Lâmpada		Valores Copel 01/01/2014	
Tipo	Incandescente	Unitário kWh	R$ 0,210194
Potência em W	60	Alíquota ICMS	29%

Foi escolhida a lâmpada do tipo incandescente conforme relatório da PROCEL (Salvador, Santos e Santos , 2007), por ser o tipo de lâmpada com maior posse para uso residencial. Os valores unitários do kilowatt e da alíquota do ICMS são de janeiro de 2014, da Companhia paranaense de energia elétrica, COPEL. Outras taxas envolvidas na conta de energia elétrica não foram consideradas, pois não há influência da quantidade de energia consumida. A potência de 60 watts (W) foi escolhida por ser um meio termo entre a baixa luminosidade da lâmpada de 40 W e a alta luminosidade da lâmpada de 100 W.

Foram criados vários cenários diferentes, começando com o cenário ideal, onde os hábitos do usuário não geram nenhum desperdício e o sistema não interfere, até o cenário onde o sistema está totalmente adaptado e com a ajuda de sensores o desperdício é inexistente.

Os dados calculados nos testes visam mostrar o consumo e o valor em reais de desperdício que cada cenário pode gerar. Esse valor é calculado pela

média diária em que a luz fica acesa, utilizando as constantes apresentadas na tabela 4.

O objetivo dos testes é identificar qual é o melhor cenário em termos de desperdício e custo de aquisição. Os testes sem sensores somente levaram em consideração os valores de desperdício devido ao fato de custarem a mesma quantia nos diversos cenários.

Foram feitos também testes com mudanças de horário no perfil durante a fase de aprendizado, simulando uma mudança de comportamento do usuário, para testar a adaptabilidade dos algoritmos. Para tanto, a tolerância nos testes foi definida como 100%, visando não ignorar uma eventual mudança de comportamento nos horários, mas ainda assim não consideraria interrupções em horários que ultrapassassem o intervalo de cada ciclo.

4.1. CENÁRIOS SEM CONTROLE AUTÔNOMO

Nos cenários sem controle autônomo foram separados os casos onde o comportamento do usuário é simulado sem interferência de nenhum tipo de sistema.

4.1.1. Ideal

Neste cenário é assumido que o usuário acende e apaga as luzes exatamente no mesmo horário e não há desperdício. É apena um cenário para comparações, pois é praticamente impossível para o usuário realizar as ações de acender e apagar as luzes exatamente no mesmo horário todos os dias. A Tabela 5 apresenta os resultados deste cenário.

Tabela 5 - Resultados do cenário Ideal
Fonte: Autoria Própria

Cenário Ideal			
Média diária	03:30:00	Dias	30
Consumo Mensal		**Desperdício Mensal**	
kWh	6,30	kWh	0,00
R$	1,71	R$	0,00
Consumo Diário		**Desperdício Diário**	
kWh	0,21	kWh	0,00
R$	0,06	R$	0,00
Sistema			
Técnica	Não se aplica	Sensor geral	0
Interferência	100%	Sensores cômodos	0
Controle Autônomo	0%	Custo de Aquisição	R$ -

A Figura 34 representa em forma de gráfico o horário da agenda ao decorrer de 30 dias.

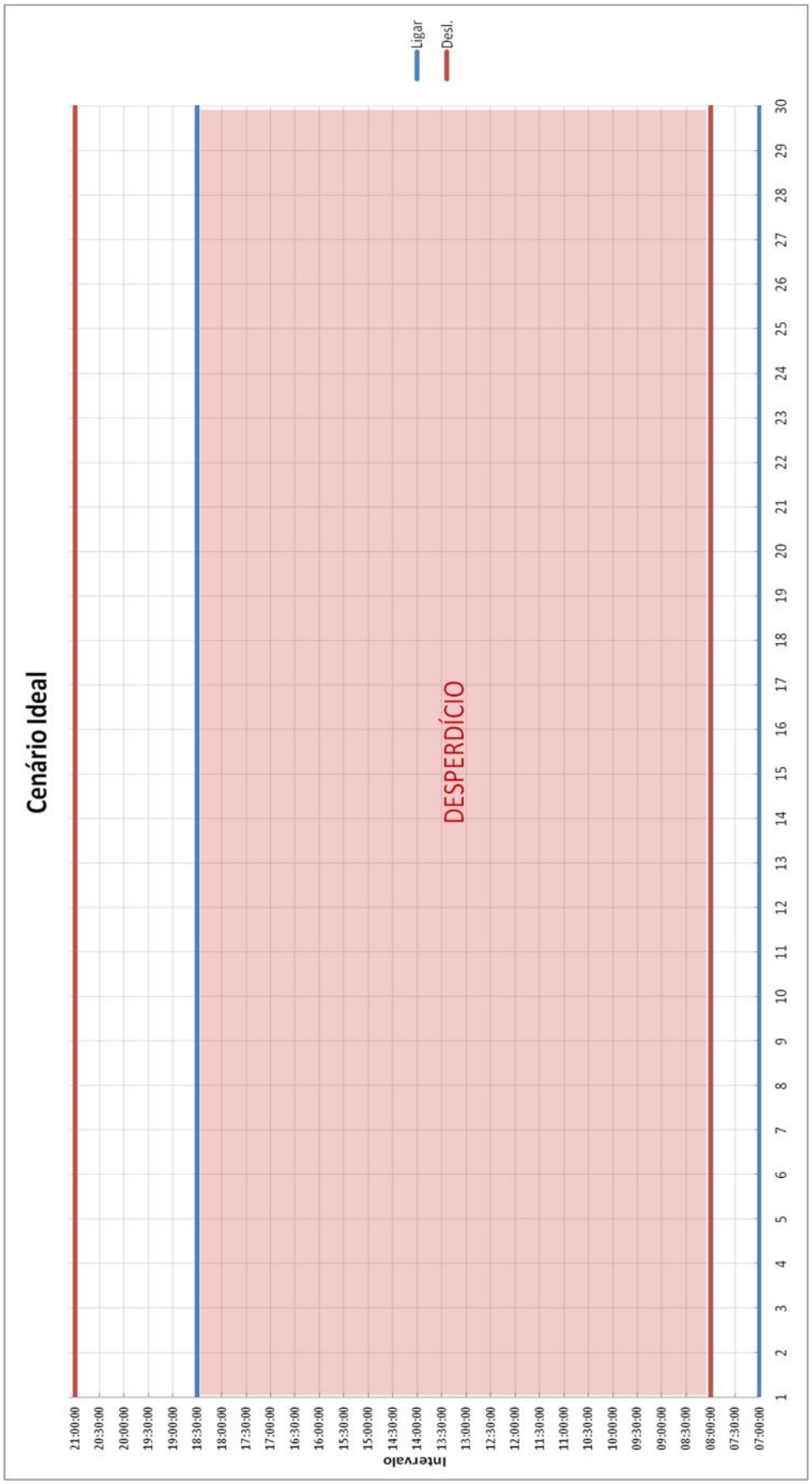

Figura 34 - Gráfico do cenário Ideal
Fonte: Autoria própria

4.1.2. Desperdício máximo

Neste cenário o usuário nunca interfere, permanecendo a luz acesa 24 horas por dia. É outro exemplo de cenário apenas a título de comparação para verificar para qual resultado as técnicas autônomas estão se aproximando. A Tabela 6 apresenta os resultados deste cenário.

Tabela 6 - Resultados do cenário Desperdício Máximo
Fonte: Autoria Própria

Desperdício Máximo			
Média diária	23:59:59	Dias	30
Consumo Mensal		**Desperdício Mensal**	
kWh	43,20	kWh	36,90
R$	11,71	R$	10,01
Consumo Diário		**Desperdício Diário**	
kWh	1,44	kWh	1,23
R$	0,39	R$	0,33
Sistema			
Técnica	Não se aplica	Sensor geral	0
Interferência	0%	Sensores cômodos	0
Controle Autônomo	0%	Custo de Aquisição	R$ -

4.1.3. Desperdício constante

O usuário neste cenário somente acende a luz quando acorda e apaga a luz quando vai dormir, deixando a luz acesa durante todo o dia, desperdiçando dez horas e trinta minutos por dia. A Tabela 7 apresenta os resultados deste cenário.

Tabela 7 - Resultados do cenário Desperdício Constante
Fonte: Autoria Própria

Desperdício Constante			
Média diária	14:00:00	Dias	30
Consumo Mensal		**Desperdício Mensal**	
kWh	25,20	kWh	18,90
R$	6,83	R$	5,12
Consumo Diário		**Desperdício Diário**	
kWh	0,84	kWh	0,63
R$	0,23	R$	0,17
Sistema			
Técnica	Não se aplica	Sensor geral	0
Interferência	100%	Sensores cômodos	0
Controle Autônomo	0%	Custo de Aquisição	R$ -

4.2. CENÁRIOS COM CONTROLE AUTÔNOMO

Os cenários com controle autônomo simulam um cômodo onde o sistema está instalado. Há cenários com e sem interferência do usuário, bem como técnicas diferentes de aprendizado.

4.2.1. Sem sensores

Nos cenários desta seção não há a presença de sensores de nenhum tipo para auxiliar no aprendizado e na economia de energia.

4.2.1.1. Configuração da agenda

O usuário configura uma agenda no sistema, porém não pode interferir. Neste caso não há aprendizado e o sistema também não interfere. Não há desperdício, uma vez que a configuração da agenda passa a ser o cenário ideal. Para efeito de comparação, a agenda será configurada conforme o cenário ideal em todos os casos. A Tabela 8 apresenta os resultados deste cenário.

Tabela 8 - Resultados do cenário Configuração da Agenda
Fonte: Autoria Própria

Configuração da agenda			
Média diária	03:30:00	Dias	30
Consumo Mensal		**Desperdício Mensal**	
kWh	6,30	kWh	0,00
R$	1,71	R$	0,00
Consumo Diário		**Desperdício Diário**	
kWh	0,21	kWh	0,00
R$	0,06	R$	0,00
Sistema			
Técnica	Não se aplica	Sensor geral	0
Interferência	0%	Sensores cômodos	0
Controle Autônomo	0%	Custo de Aquisição	R$ 96,28

4.2.1.2. Espelhamento

Na técnica de espelhamento o algoritmo sempre repetirá a última interrupção do usuário. Considerando 30 dias de treinamento, supõe-se que os próximos dias tenham comportamento parecido e sendo assim o desperdício seria sempre o mesmo se comparado com o cenário ideal.

Também conhecido como Método de Previsão de Média Móvel, esta técnica utiliza os $\boldsymbol{n}$ últimos valores da série temporal $\boldsymbol{x_t}$, como a previsão para o tempo $\boldsymbol{t+1}$, sendo representada pela equação abaixo.

$$F_{t+1} = \sum_{i=t-n+1}^{t} \frac{x_i}{n}$$

Com essa técnica a previsão começa a partir do 31º dia, porém ela é alterada conforme o valor de $\boldsymbol{n}$ muda. Quanto maior o valor de $\boldsymbol{n}$, menor é a série para aprendizado. A desvantagem deste método é que valores muito discrepantes influenciam bastante na previsão e que o peso dos valores antigos é igual ao dos novos valores, diminuindo a precisão da previsão caso o usuário mude seu comportamento. A Tabela 9 apresenta os resultados deste cenário. Neste teste foi utilizado o valor de $\boldsymbol{n = 1}$.

Tabela 9 - Resultados do cenário Espelhamento
Fonte: Autoria Própria

Espelhamento			
Média diária	03:37:46	Dias	30
Consumo Mensal		**Desperdício Mensal**	
kWh	6,53	kWh	0,23
R$	1,77	R$	0,06
Consumo Diário		**Desperdício Diário**	
kWh	0,22	kWh	0,01
R$	0,06	R$	0,00
Sistema			
Técnica	Espelhamento	Sensor geral	0
Interferência	100%	Sensores cômodos	0
Controle Autônomo	100%	Custo de Aquisição	R$ 96,28

A Figura 35 mostra os horários que o usuário acendeu e apagou a luz e o valor que o algoritmo de espelhamento utilizou, repetindo o horário do usuário no dia anterior.

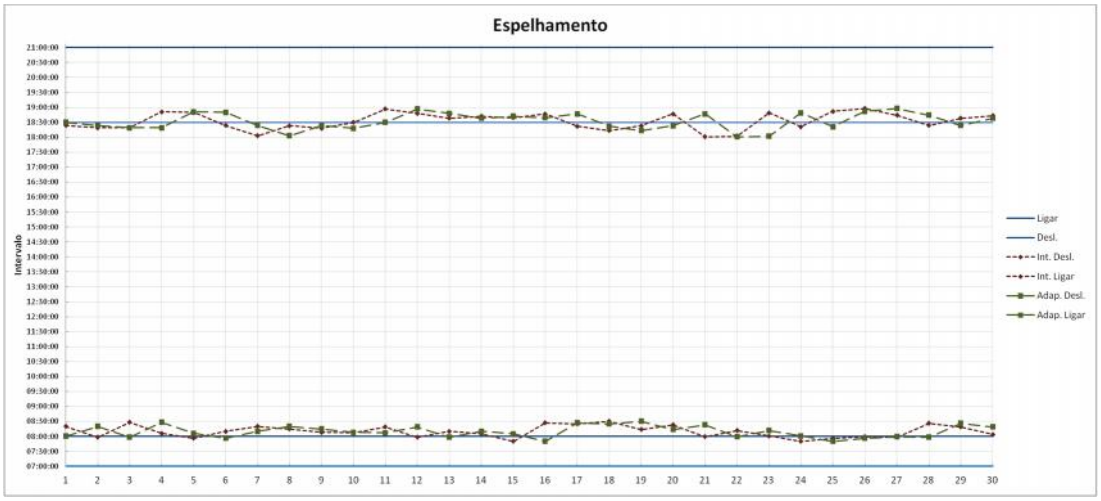

Figura 35 - Gráfico do cenário Espelhamento
Fonte: Autoria Própria

Esse padrão será repetido de trinta em trinta dias, não sendo a melhor solução para um sistema com aprendizado e adaptabilidade. Uma mudança de comportamento na fase de aprendizado traria um resultado insatisfatório e possivelmente a fase de aprendizado teria de ser refeita.

4.2.1.3. Média simples com séries de 30 dias

Na média simples o algoritmo atualizará a agenda diariamente e a tolerância acompanhará essa atualização. A tolerância máxima é de cinquenta por cento do tempo em que a luz ficaria acesa na agenda, para mais ou para menos. Ou seja, com tolerância parametrizada para cem por cento no sistema, o algoritmo aceitaria valores de 07h30min até 08h30min para a agenda configurada às 08h00min. Como a agenda é adaptada diariamente, a faixa de tolerância acompanha a atualização da agenda.

Nesta técnica o aprendizado acompanha a rotina do usuário no início, entretanto, quanto mais perto de completar a série, menor a adaptação, tendo um comportamento parecido com a técnica de espelhamento. A Tabela 10 apresenta os resultados deste cenário.

Sua equação pode ser representada abaixo, com $\boldsymbol{i} <= \mathbf{3}$.

$$F_t = \sum_{i}^{t} \frac{x_i}{i}$$

Tabela 10 - Resultado do cenário Média Simples - Série de 30 dias - Tolerância 100%
Fonte: Autoria Própria

Média Simples - Série 30 dias - Tolerância 100%			
Média diária	03:41:29	Dias	30
Consumo Mensal		**Desperdício Mensal**	
kWh	6,64	kWh	0,34
R$	1,80	R$	0,09
Consumo Diário		**Desperdício Diário**	
kWh	0,22	kWh	0,01
R$	0,06	R$	0,00
Sistema			
Técnica	Média Simples	Sensor geral	0
Interferência	100%	Sensores cômodos	0
Controle Autônomo	100%	Custo de Aquisição	R$ 96,28

A Figura 36 mostra os horários que o usuário acendeu e apagou a luz e o valor que o algoritmo de aprendizado calculou como sendo ideal segundo os últimos 30 valores anteriores.

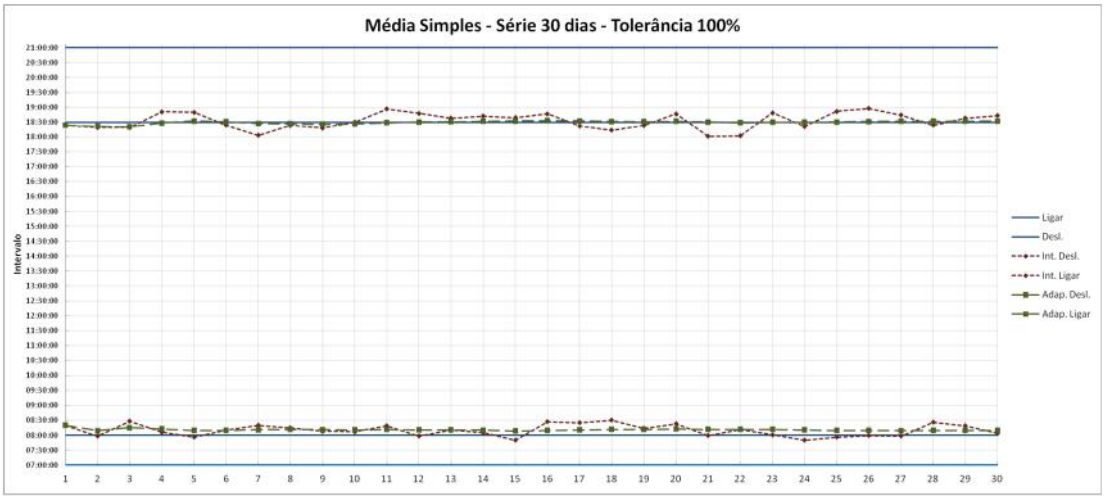

Figura 36 - Resultado do cenário Média Simples - Série de 30 dias - Tolerância 100%
Fonte: Autoria própria

Foi feito mais um teste com uma mudança no comportamento do usuário entre o dia 10 e o dia 20, para verificar o acompanhamento do aprendizado. Neste caso será ignorado o desperdício, pois o foco é a curva de aprendizado do algoritmo.

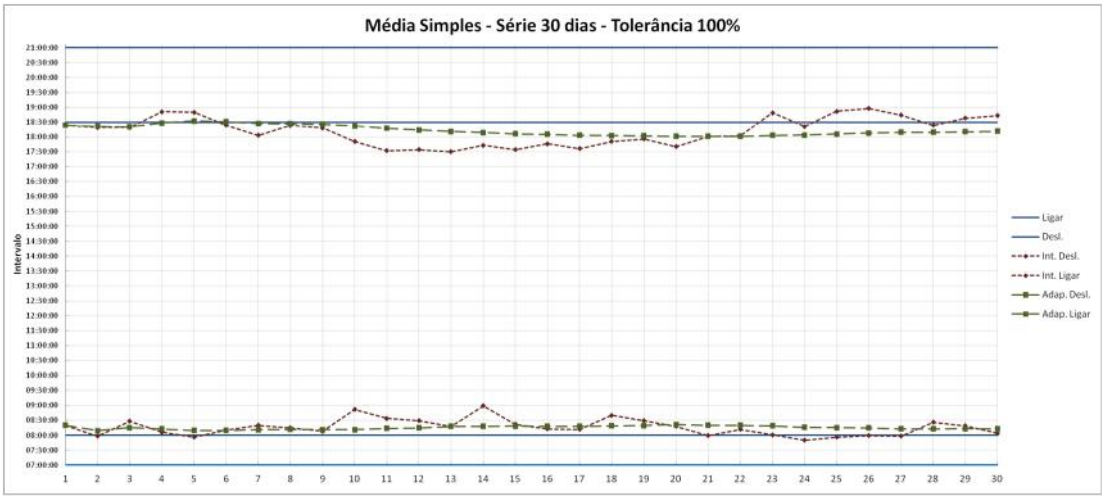

Figura 37 - Resultado do cenário Média Simples - Série de 30 dias - Tolerância 100% - Mudança de comportamento 1
Fonte: Autoria própria

Como pode ser verificado na Figura 37, a curva de aprendizado está atrasada. Não chega a acompanhar a mudança de rotina do usuário antes dele voltar à sua rotina normal.

4.2.1.4. Média simples com séries de 15 dias

Esta técnica é muito semelhante à anterior, porém a série que acompanha o aprendizado é de 15 dias. A vantagem de trabalhar com séries limitadas é que a adaptação à rotina do usuário é mais fiel, uma vez que, diferente do espelhamento, o aprendizado não depende de um período de treinamento.

Quanto menor a série, mais fiel à rotina do usuário, porém um comportamento caótico diminui as chances de economia de energia. A Tabela 11 apresenta os resultados deste cenário.

Sua equação pode ser representada abaixo, com $\boldsymbol{i} <= 15$.

$$F_t = \sum_{i}^{t} \frac{x_i}{i}$$

Tabela 11 - Resultado do cenário Média Simples - Série de 15 dias - Tolerância 100%
Fonte: Autoria Própria

Média Simples - Série 15 dias - Tolerância 100%			
Média diária	03:40:51	Dias	30
Consumo Mensal		**Desperdício Mensal**	
kWh	6,63	kWh	0,33
R$	1,80	R$	0,09
Consumo Diário		**Desperdício Diário**	
kWh	0,22	kWh	0,01
R$	0,06	R$	0,00
Sistema			
Técnica	Média Simples	Sensor geral	0
Interferência	100%	Sensores cômodos	0
Controle Autônomo	100%	Custo de Aquisição	R$ 96,28

A Figura 38 mostra os horários que o usuário acendeu e apagou a luz e o valor que o algoritmo de aprendizado calculou como sendo ideal segundo os últimos 15 valores anteriores.

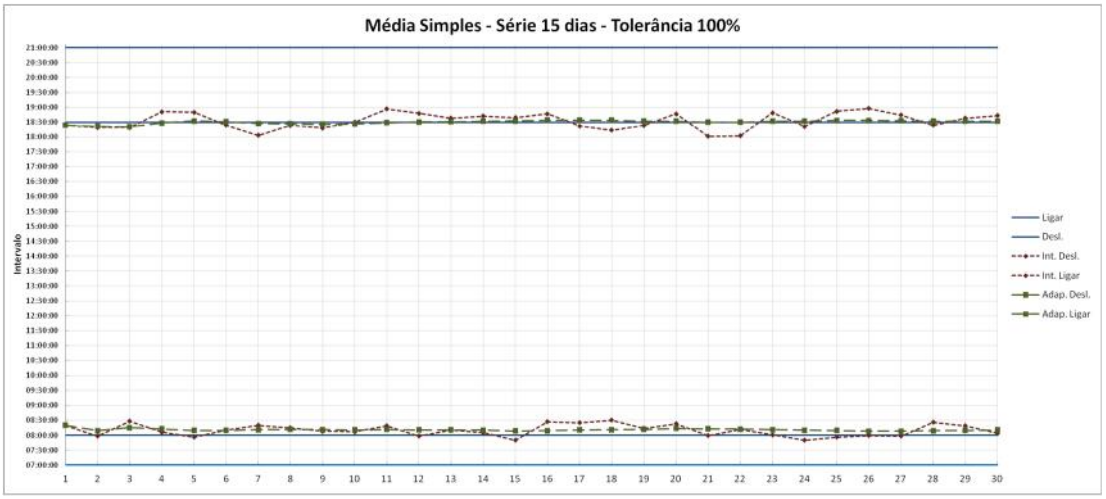

Figura 38 - Resultado do cenário Média Simples - Série de 15 dias - Tolerância 100%
Fonte: Autoria própria

Também foi feito mais um teste com a mesma mudança no comportamento do usuário apresentada no teste anterior.

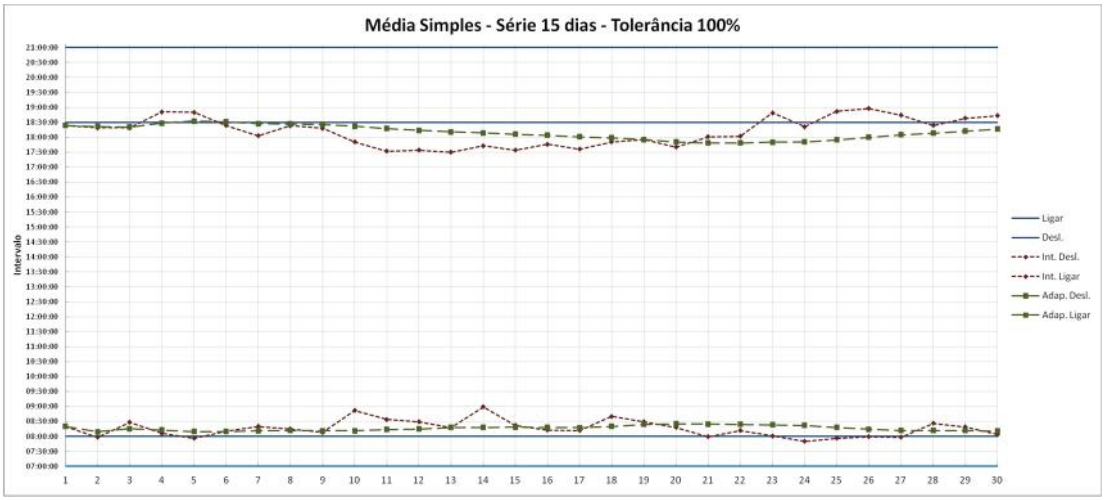

Figura 39 - Resultado do cenário Média Simples - Série de 15 dias - Tolerância 100% - Mudança de comportamento 1
Fonte: Autoria própria

Resultado semelhante à série de 30 dias, com uma leve diferença nos picos maiores, conforme pode ser visto na Figura 39.

4.2.1.5. Média simples com séries de cinco dias

Mesmos valores testados utilizando uma série móvel menor. Com apenas cinco dias de memória a adaptabilidade é mais fiel ao comportamento do usuário. A Tabela 12 apresenta os resultados deste cenário.

Tabela 12 - Resultado do cenário Média Simples - Série de cinco dias - Tolerância 100%
Fonte: Autoria Própria

Média Simples - Série 5 dias - Tolerância 100%			
Média diária	03:38:14	Dias	30
Consumo Mensal		**Desperdício Mensal**	
kWh	6,55	kWh	0,25
R$	1,78	R$	0,07
Consumo Diário		**Desperdício Diário**	
kWh	0,22	kWh	0,01
R$	0,06	R$	0,00
Sistema			
Técnica	Média Simples	Sensor geral	0
Interferência	100%	Sensores cômodos	0
Controle Autônomo	100%	Custo de Aquisição	R$ 96,28

A Figura 40 mostra os horários que o usuário acendeu e apagou a luz e o valor que o algoritmo de aprendizado calculou como sendo ideal segundo os últimos 5 valores anteriores.

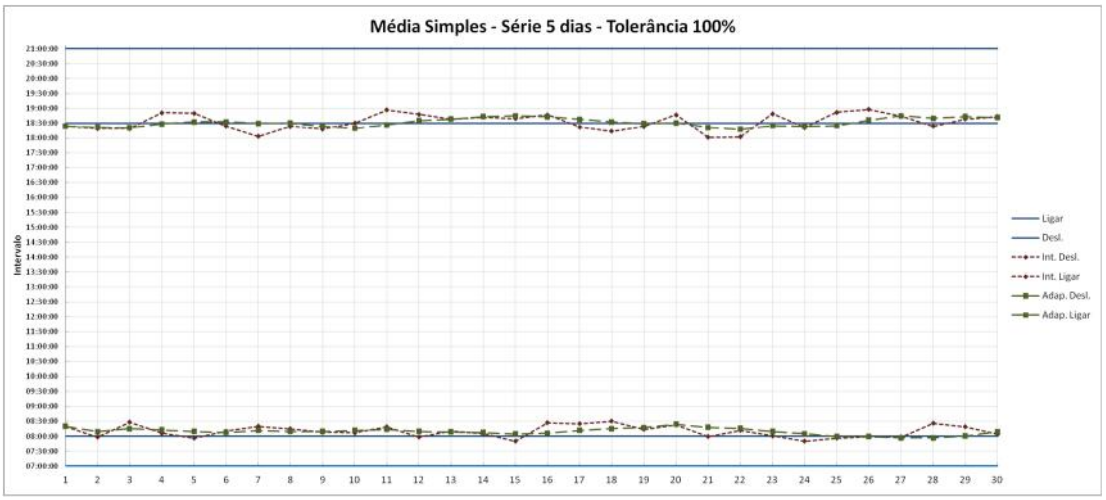

Figura 40 - Resultado do cenário Média Simples - Série de cinco dias - Tolerância 100%
Fonte: Autoria própria

Na Figura 41 é representado mais um teste com a mudança de comportamento dentro do período analisado.

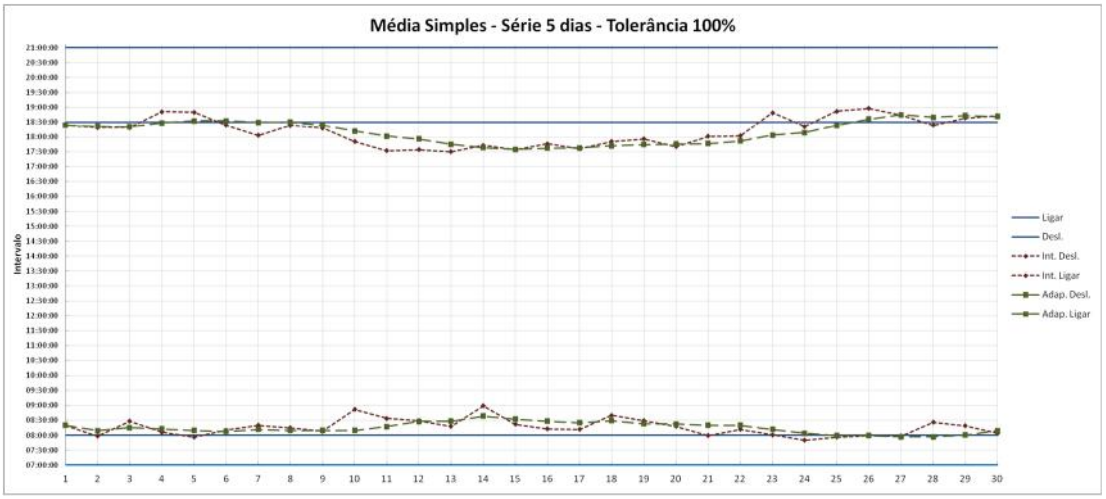

Figura 41 - Resultado do cenário Média Simples - Série de cinco dias - Tolerância 100% - Mudança de comportamento 1
Fonte: Autoria própria

A curva de aprendizado no algoritmo aplicando adaptação nos últimos cinco dias é mais responsiva se comparada aos outros testes. Não restam dúvidas que para as duas rotinas testadas, na média simples quanto menor a série, maior a adaptação. Todavia, um comportamento muito variado ainda pode levar o algoritmo a dar resultados indesejáveis, sendo imprescindível uma análise junto ao usuário do melhor valor de tolerância a ser utilizado.

4.2.2. Com sensores

Os testes com sensores não puderam ser realizados com simulação devido à falta dos mesmos, porém o comportamento dos sensores é de permitir que o sistema atue, sendo de fácil entendimento o efeito causado na adição desses elementos.

No aprendizado os sensores seriam interpretados como interruptores, não exercendo algum papel especial no algoritmo de aprendizado.

Para uma comparação coerente, foi assumido que no período de 30 dias, em cinco dias os sensores não acusaram movimento nenhum, ou seja, o usuário não estava presente, portanto não realizou nenhuma interrupção e a agenda adaptada também não foi utilizada.

4.2.2.1. Com sensor de presença geral

O sensor de presença geral tem o comportamento de não deixar o sistema atuar caso não detecte movimento dentro do período agendado para a luz ficar acesa. É uma maneira de garantir que não haja desperdício, contudo, o valor inicial do sistema aumenta. A Figura 42 ilustra a atuação do sensor.

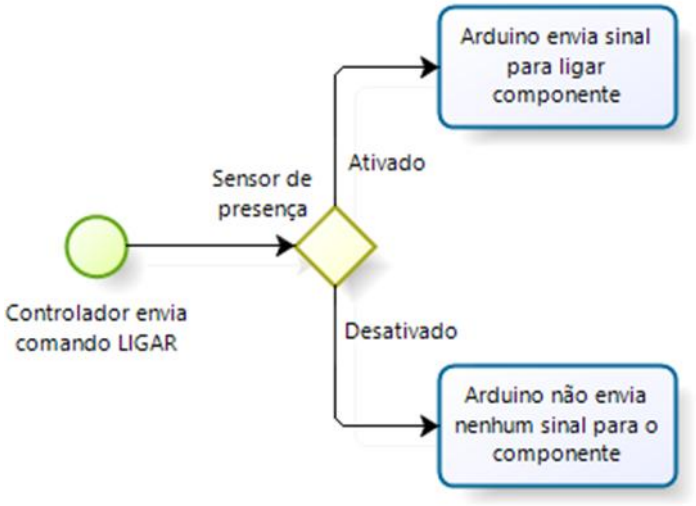

Figura 42 - Atuação do sensor de presença
Fonte: Autoria Própria

Utilizou-se a média simples com série de cinco dias para aplicar a atuação dos sensores, pois foi o melhor resultado nestes testes. A Tabela 13 apresenta os resultados deste cenário.

Tabela 13 - Resultado do cenário Média Simples - Série de cinco dias - Tolerância 100% - Sensor de presença geral
Fonte: Autoria Própria

Média Simples - Série 5 dias - Tolerância 100%			
Média diária	03:01:40	Dias	30
Consumo Mensal		**Desperdício Mensal**	
kWh	5,45	kWh	0,20
R$	1,48	R$	0,05
Consumo Diário		**Desperdício Diário**	
kWh	0,18	kWh	0,00
R$	0,05	R$	0,00
Sistema			
Técnica	Média Simples	Sensor geral	0
Interferência	100%	Sensores cômodos	0
Controle Autônomo	100%	Custo de Aquisição	R$ 111,28

Houve uma redução no consumo mensal de R$ 1,10 e um aumento de R$ 15,00 no custo de aquisição. Com uma média de cinco dias sem atuação do sistema esta diferença pagaria o aumento de investimento em 14 meses.

4.2.2.2. Com sensor de presença nos cômodos

O sensor de presença nos cômodos funciona praticamente da mesma forma que o sensor geral, porém, atuando especificamente em um cômodo somente. Cada sensor deste aumentaria o custo do sistema em R$ 15,00 e se a agenda deste cômodo for semelhante ao perfil testado, o resultado seria o mesmo do alcançado com o sensor geral para uma ausência de cinco dias do usuário.

4.2.2.3. Com todos os sensores

Esta configuração contempla cercar de todas as formas o desperdício de energia dispensando aprendizado e agenda.

Utilizando um conjunto de sensores, a luz só acenderia quando não houvesse iluminação natural suficiente, houvesse algum ser humano no ambiente e não apagaria por tempo ou hora determinada mesmo que o usuário ficasse estático por algum tempo.

O nível de iluminação seria informado por um sensor de luminosidade, conhecido como sensor fotossensível. Um sensor térmico pode ser ajustado na própria unidade quanto à sua sensibilidade e no Arduíno pode ser configurado um valor de temperatura como gatilho para algum comando, evitando a detecção de animais domésticos por exemplo. Os sensores de movimento e de presença, apesar de parecerem funcionar da mesma forma, são utilizados em conjunto para saber a direção do usuário e se o mesmo continua no ambiente, mas de forma imóvel, que poderia enganar o sistema caso só existisse o sensor de movimento.

Estes conjuntos de sensores teriam que satisfazer as condições programadas e parametrizadas no Arduíno para que uma lâmpada pudesse

acender. Todavia, o custo de aquisição do sistema com tantos sensores fugiria do escopo do trabalho, que preza além da economia de energia, o baixo custo do sistema.

Para efeitos de comparação, a Tabela 14 mostra os valores de cada sensor. Com exceção do *shield* sensor, os outros sensores são necessários para cada lâmpada controlada pelo sistema. O *shield* sensor é necessário devido à limitação das portas do Arduíno.

Tabela 14 - Valor dos sensores compatíveis com o Arduíno Uno e o sistema autônomo
Fonte: Autoria Própria

Sensores Compatíveis Arduino Uno			
Componente	Valor	Quantidade	Total
Shield Sensor	R$ 60,00	1	R$ 60,00
Sensor Temperatura	R$ 35,00	1	R$ 35,00
Sensor Movimento	R$ 15,00	1	R$ 15,00
Sensor Presença	R$ 15,00	1	R$ 15,00
Sensor Luminosidade	R$ 2,00	1	R$ 2,00

5. RESULTADOS

Após os testes ficou claro que quanto menor a série calculada no algoritmo, maior é a semelhança entre o real e o adaptado. Contudo, o valor de tolerância é um parâmetro mais pessoal do que técnico e tem grande influência na adaptabilidade.

Usuários com uma rotina flexível podem optar por valores maiores e/ou adquirir o sistema com um ou mais sensores, resolvendo assim boa parte dos problemas que poderiam ser gerados por uma rotina muito variada.

Usuários com rotinas com menos variações podem utilizar o parâmetro de tolerância com valores mais baixos para que, mesmo com algum desvio, a adaptação se mantivesse mais próxima aos valores mais constantes.

Comparando os resultados, foi verificado que a utilização de um sistema autônomo ajuda a reduzir o consumo de energia elétrica, chegando cada vez mais perto do ideal conforme é diminuída a série utilizada para fazer o cálculo.

No decorrer do projeto houve a necessidade de montar uma maquete para ilustrar as funcionalidades do sistema em um ambiente mais próximo ao real.

Nos cômodos foram simulados os perfis já mencionados na seção 3.2, em que cada led correspondia a um perfil, como pode ser visto na Figura 43.

Figura 43 - Maquete
Fonte: Autoria Própria

Considerando o valor de implantação do sistema conforme Tabela 15, pode ser verificado que na configuração mais simples, o investimento seria pago em seis meses (o valor do dispositivo necessário que fica conectado ao Arduíno não está incluído, podendo ser qualquer computador com entrada USB) se for assumido o cenário Desperdício Constante.

Tabela 15 – Valor mínimo para instalação do sistema
Fonte: Autoria Própria

Mínimo para funcionamento do sistema - 1 Lâmpada			
Componente	Valor	Quantidade	Total
Arduino Uno	R$ 71,85	1	R$ 71,85
Led	R$ 0,23	1	R$ 0,23
Resistor	R$ 0,20	3	R$ 0,60
Sensor Movimento	R$ 15,00	1	R$ 15,00
Botão	R$ 4,00	1	R$ 4,00
Rele	R$ 4,60	1	R$ 4,60
Total			**R$ 96,28**

Para o sistema com o sensor geral, o valor do sistema seria pago em aproximadamente sete meses, porém seria aconselhado aos casos onde a residência fica sem ninguém durante alguns dias no mês. Na Tabela 16 podemos ver o valor com o sensor geral.

Tabela 16 - Sistema com sensor geral
Fonte: Autoria Própria

Sistema com sensor geral - 1 Lâmpada			
Componente	Valor	Quantidade	Total
Arduino Uno	R$ 71,85	1	R$ 71,85
Led	R$ 0,23	1	R$ 0,23
Resistor	R$ 0,20	3	R$ 0,60
Sensor Movimento	R$ 15,00	1	R$ 15,00
Botão	R$ 4,00	1	R$ 4,00
Rele	R$ 4,60	1	R$ 4,60
Sensor Presença	R$ 15,00	1	R$ 15,00
Total			**R$ 111,28**

Se o usuário quiser o controle total, utilizando os demais sensores no cômodo para evitar qualquer tipo de desperdício, o valor do sistema aumenta conforme a Tabela 17.

Tabela 17 - Sistema com sensores
Fonte: Autoria Própria

Sistema com sensores			
Componente	Valor	Quantidade	Total
Arduino Uno	R$ 71,85	1	R$ 71,85
Led	R$ 0,23	1	R$ 0,23
Resistor	R$ 0,20	3	R$ 0,60
Sensor Movimento	R$ 15,00	1	R$ 15,00
Botão	R$ 4,00	1	R$ 4,00
Rele	R$ 4,60	1	R$ 4,60
Sensor Presença	R$ 15,00	1	R$ 15,00
Shield Sensor	R$ 60,00	1	R$ 60,00
Adicional por lâmpada controlada			
Sensor Temperatura	R$ 35,00	1	R$ 35,00
Sensor Movimento	R$ 15,00	1	R$ 15,00
Sensor Presença	R$ 15,00	1	R$ 15,00
Sensor Luminosidade	R$ 2,00	1	R$ 2,00
Total			**R$ 238,28**

É importante ressaltar que até os resultados apresentados o sistema precisou de um computador com porta USB conectado para realizar os cálculos dos algoritmos de aprendizagem, bem como armazenar os horários da agenda no banco de dados.

O Raspberry PI – Modelo B é um mini computador que pode ser utilizado para se conectar ao Arduino. O Modelo B custa em média R$ 230,00 no mercado brasileiro (RASPBERRY PI, 2014).

6. CONCLUSÃO

O desenvolvimento desse projeto possibilitou alcançar os objetivos descritos, mostrando-se eficaz ao realizar as suas funções básicas, mostrando que é possível, de uma maneira simples, automatizar uma residência com um baixo custo, deixando essa tecnologia acessível a grande parte da população de classe média baixa, além de provar que a economia de energia que o projeto disponibiliza o faz ter um enorme custo benefício junto com o conforto gerado.

Como observado nas seções anteriores, o controle de equipamentos eletrônicos se dá pelo simples fornecimento, ou não, de energia elétrica aos componentes. O importante a partir daqui é elaborar um programa que contemple algumas técnicas para controle de forma que os leds sejam acessos ou apagados conforme configuração ou de modo autônomo. Após o ajuste a aceitação, modificações de robustez e escala se farão necessárias.

6.1 TRABALHOS FUTUROS

Nada é tão perfeito que não possa ser melhorado, ainda mais quando estamos falando de tecnologia da informação. Quando ocorre uma mistura de softwares e hardwares em uma aplicação os dois estarão sujeitos a mudanças, entre melhorias futuras estão:

- Incorporar sensores a fim de ter uma aplicação mais próxima do resultado ideal quanto ao gasto de energia elétrica;
- Desenvolver um aplicativo para sistemas operacionais de telefonia móvel que permita a interação via Internet;
- Utilização do sistema para fins de segurança residencial, com alarmes e sensores;
- Simulador de presença residencial, onde o sistema pode acender e apagar as luzes dos cômodos;
- Aplicação totalmente embarcada ou em conjunto com configuração minimalista, como o Raspberry PI.

REFERÊNCIAS

Alecrim, Elmasri. Banco de dados MySql e PostgreSQL. 2008. Disponível em: <http://www.infowester.com/postgremysql.php> Acesso em: 16 Julho 2014

Alvares, Luis Otavio e Jaime Simão Sichman. “Introdução aos Sistemas Multiagentes.” *XVII Congresso da SBC.* Brasília, 1997.

Apache HTTP Server Project. Apache. [S.l.]. Disponível em: <http://httpd.apache.org/ABOUT_APACHE.html>. Acesso em: 04 Abril 2014.

Arduíno. Disponível em: <http://www.Arduino.cc> Acesso em: 10 Janeiro, 2014.

Bai, Ying-Wen, e Yi-Te Ku. “Automatic Room Light Intensity Detection and Control.” *IEEE Transactions on Consumer Electronics, Vol.54, No.4*, 2008: 1173-1176.Disponível em: <http://dx.doi.org/10.1109/TCE.2008.4637603>. Acesso em: 19 Novembro 2013.

Chen, Shizhi, YingLi Tian, Qingshan Liu, e Dimitris N. Metaxas. “Recognizing expressions from face and body gesture by temporal normalized motion and appearance featuresand appearance features.” *Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE*, 2011: 7-12. Disponível em: <http://dx.doi.org/10.1109/CVPRW.2011.5981880>. Acesso em: 29 Outubro 2013.

Cook, Diane J. “How Smart Is Your Home?” *Science, Vol.335*, 2012: 1579-1581. Disponível em: <http://www.sciencemag.org/content/335/6076/1579>. Acesso em: 29 Outubro 2013.

Di Renna e outros. *Introdução ao kit de desenvolvimento Arduíno.* Junho 2013.

Elmasri, Ramez. “Sistemas de banco de dados”. 4. ed. São Paulo: Pearson Addison Wesley, 2005.

Euzebio, Michel Vinicius de Melo e Emerson Ribeiro de Mello. “Droid-lar – Automação Residencial através de um celular Android”. Instituto Federal de Santa Catarina, 2011.

Greggianin, Calisto Antônio, et al. “Estudo comparativo entre lâmpadas: incandescentes, fluorescentes compactas e LED.” *ESPAÇO ENERGIA*, 2013: 19-27. Disponível em: <http://www.espacoenergia.com.br/edicoes/18/EE018-07-11%20Comparative%20study%20of%20bulbs%20incandescent%20bulbs%20fluorescent%20bulbs%20and%20LED%20bulbs.pdf>. Acesso em: 14 Junho 2014.

Kim, Eunju, Sumi Helal, e Diane Cook. “Human Activity Recognition and Pattern Discovery.” *Pervasive Computing, IEEE, Vol.9, No.1*, 2009: 48-53. Disponível em: <http://dx.doi.org/10.1109/MPRV.2010.7>. Acesso em: 29 Outubro 2013.

Lima, Luiz Eduardo R. H. de, et al. “Hometec: Automação Residencial utilizando Comando por Voz e Tecnologia de Rede Sem Fios.”

Matta, Sherif, e Masud Syed Mahmud. “An Intelligent Light Control Systemfor Power Saving.” *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, 2010: 3316-3321. Disponível em: <http://dx.doi.org/10.1109/IECON.2010.5675331>. Acesso em: 19 Novembro 2013.

MARQUES, E. “Twitter Bootstrap, aumentando sua produtividade”. Disponível em <http://www.devmedia.com.br/twitter-bootstrap-aumentando-sua-produtividade/24967 acessado em 07/07/2014> Acesso em: 7 Julho 2014.

Meira, Adriano Bachetta; Delfino, Sergio Roberto e Orlandini, Guilherme. “Domotica utilizando softwares e hardwares livres”. Curso de Sistemas de Informação – Faculdade de Ourinhos, 2012.

Nunes, Renato Jorge Caleira. "Analise Corporativa de tecnologias para Domotica". 2002.

Nogueira, Fernando. “Modelagem e Simulação - Modelos de Previsão.” *Notas de Aula.* Juiz de Fora, 02 de 2009. Disponível em: <www.ufjf.br/epd042/files/2009/02/previsao1.pdf>. Acesso em: 16 Junho 2014.

Osório, Fernando Santos. “Tutorial: Redes Neurais – Aprendizado Artificial”. Fórum de Inteligência Artificial. 1999.

Paradiso, Joseph, Prabal Dutta, Hans Gellersen, e Eve Schooler. “Smart Energy Systems.” *Pervasive Computing, IEEE, Vol.10, No. 1*, 2011: 11-12. Disponível em: <http://dx.doi.org/10.1109/MPRV.2011.4>. Acesso em: 29 Outubro 2013.

Php, Hypertext preprocessor. Disponível em: <https://php.net/> Acesso em: 14 Fevereiro, 2014.

Quinlan, John Ross. “Induction of Decision Trees. Machine Learning”, Capítulo 1, Morgan Kaufmann. 1990. 81-106.

Quinlan, John Ross. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers. San Mateo, California, EE.UU. 1993

RASPBERRY PI FOUNDATION. Disponível em: < http://www.raspberrypi.org/about/ > Acesso em: 04 Agosto 2014.

Salvador, Emerson, Moisés Antônio dos Santos, e Marcelo José dos Santos. *Resultados do PROCEL.* Procel Avaliação, Gráfica da Eletrobrás - DAAG, 2007.Disponível em: <http://www.procelinfo.com.br/main.asp?View=%7B5A08CAF0-06D1-4FFE-B335-95D83F8DFB98%7D&Team=&params=itemID=%7B6AF5DAFB-E415-4F43-968B-0A2FB997E54F%7D;&UIPartUID=%7B05734935-6950-4E3F-A182-629352E9EB18%7D>. Acesso em: 16 Junho 2014.

Santos, Jose Mauro, Waldiney Giacomelli, Luis Vinicius e Mariana Fontes. “Domotica via WEB ao alcance da classe media baixa”. VII CONNEPI, 2012.

Sarik, John, e Ioannis Kymissis. “Lab Kits Using the Arduíno Prototyping Platform.” *Frontiers in Education Conference, IEEE*, 2010: T3C-1 - T3C-5.Disponível em: <http://dx.doi.org/10.1109/FIE.2010.5673417>. Acesso em: 19 Novembro 2013.

Song, Eugene Y., e Kang Lee. “Understanding IEEE 1451-Networked smart transducer interface standard - What is a smart transducer?” *Instrumentation &Measurement Magazine, IEEE, Vol.11, No.2*, 2008: 11-17.Disponível em: <http://dx.doi.org/10.1109/MIM.2008.4483728>. Acesso em: 29 outubro 2013.

Sun, Maoheng, Qian Liu, e Min Jiang. “An Implementation of Remote Lighting Control System Based on Zigbee.” *International Conference on Audio, Language and Image Processing, 2008. ICALIP 2008.*, 2008: 629-632. Disponível em: <http://dx.doi.org/10.1109/ICALIP.2008.4590223>. Acesso em: 19 Novembro 2013.

Tonidandel, Flavio, Marcelo Takiuchi e Erica Melo. “Domótica Inteligente: Automacao baseada em comportamento”, Congresso Brasileiro de Automática, 2004.

Twitter Bootstrap, disponível em: <http://getbootstrap.com/> Acesso em: 16 Julho 2014.

Weiser, Mark. “Some computer science issues in ubiquitous computing.” *Communicatios of the ACM, Vol.36, No. 7*, 1993: 75-84. Disponível em: <http://dl.acm.org/citation.cfm?id=159617>. Acesso em: 29 outubro 2013. Acesso em: 19 Novembro 2013.

Wortmeyer, Charles, Fernando Freitas e Liuam Cardoso. “Busca de Tecnologias visando o Conforto, a Economia, a Pratica e a Seguranca do Usuario” - Automacao Residencial. II Simpósio de Excelência em Gestão e Tecnologia – SEGeT. 2005.

Yeh, Lun-Wu, Che-Yen Lu, Chi-Wai Kou, Yu-Chee Tseng, e Chih-Wei Yi. “Autonomous Light Control by Wireless Sensor and Actuator Networks.” *Sensors Journal, IEEE Vol.10, No. 6*, 2010: 1029-1041. Disponível em: <http://dx.doi.org/10.1109/JSEN.2010.2042442>. Acesso em: 19 Novembro 2013.

Youngblood, G. Michael, e Diane J. Cook. “Data Mining for Hierarchical Model Creation.” *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Vol.37, No. 4*, 2007: 561-572. Disponível em: <http://dx.doi.org/10.1109/TSMCC.2007.897341>. Acesso em: 29 Outubro 2013.