DC ALERTA: PROTÓTIPO DE UM SISTEMA DE MONITORAMENTO DA TEMPERATURA E UMIDADE PARA DATA CENTERS

Milson António, Mauro Marcelo Mattos – Orientador

Resumo: *O termo Internet das Coisas (Internet of Things – IoT) vem sendo adotado como a convergência de várias tecnologias de automação tais como sistemas embarcados, sistemas de comunicação e hardware e que possibilita a integração de dispositivos, sensores e atuadores à rede internet. O DC Alerta é um sistema que visa permitir o monitoramento da temperatura e da umidade de Data Centers e alertar aos usuários quando uma destas grandezas encontrar-se fora dos limites estabelecidos, garantindo que o ambiente monitorado se mantenha em condições adequadas. Foi desenvolvido utilizando o microcontrolador Arduino e os sensores de temperatura e umidade DHT11. Além disso, uma aplicação de monitoramento foi desenvolvida em versão Android. Como resultados dos testes, verificou-se a viabilidade na adoção da tecnologia para monitoramento de ambientes confinados.*

Palavras-chave*: IoT. Monitoramento de Temperatura e Umidade. Data Center.*

1 INTRODUÇÃO

A chamada Internet das Coisas é um fenômeno que vem sendo capitaneado pela rápida popularização da Internet e da possibilidade de interconectar objetos diversos os quais possuem capacidade de disponibilizarem informações a respeito do seu funcionamento (SATO, 2015). Conforme Patel, et., al. (2016), IoT pode ser classificado em três categorias as quais interagem através da internet: "pessoas conectadas a pessoas, pessoas conectadas a máquinas (ou coisas) e coisas (ou máquinas) conectadas a coisas (ou máquinas)" (tradução nossa).

> Neste ambiente, objetos tornam-se identificáveis e inteligentes na medida em que podem tomar decisões baseadas no contexto graças ao fato de que podem trocar informações entre eles. Eles podem acessar informações que podem estar agregadas a outras coisas, ou podem ser componentes de serviços mais complexos. (PATEL, 2016, p. 1).

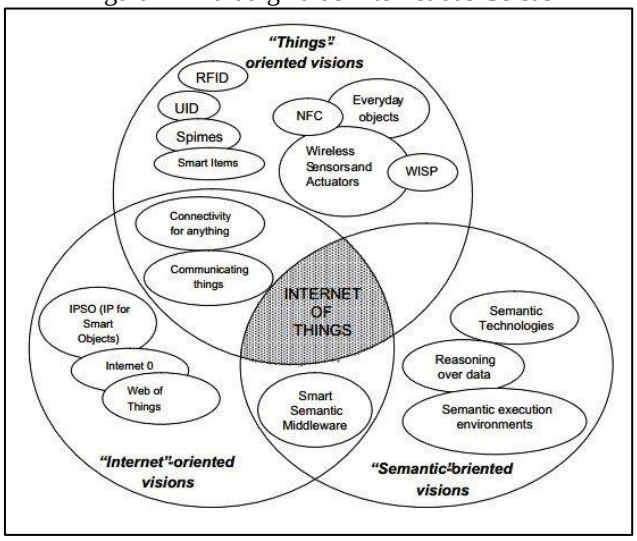
Existem diversas situações onde é estritamente necessário o monitoramento do ambiente, como por exemplo em data centers. A proliferação de soluções de tecnologias comerciais permite utilizar um único dispositivo para controlar uma série de dispositivos eletrônicos e *appliances*. Estas soluções concentram-se primariamente em monitoramento ambiental e gerenciamento de consumo de energia elétrica (VERMESAN; FRIESS; 2014). Uma das ferramentas comerciais utilizadas para monitoramento de um conjunto de sensores é o PRTG (XANDÓ; 2018).

A partir deste contexto, o artigo descreve o desenvolvimento de um protótipo de um sistema de monitoramento de variáveis ambientais voltados para *data centers* denominado de DC Alerta. Este projeto foi motivado pela provocação de uma grande empresa da região de Blumenau a qual resultou numa primeira fase em um pequeno protótipo de prova de conceito apresentado nas disciplinas de Projeto de Software II e posteriormente em TCC I do curso de Ciência da Computação da FURB. Agora expandido, o conjunto de funcionalidades elencadas naquele protótipo foi aprimorado de forma a disponibilizar-se uma solução baseada em Internet das Coisas para o monitoramento ambiental de *data centers* utilizando Arduino como plataforma de coleta de dados.

2 FUNDAMENTAÇÃO TEÓRICA

A abrangência do termo IoT envolve não só a área de sensores e atuadores, mas toda a gama de equipamentos que utilizam o protocolo TCP/IP (Transmission Control Protocol/Internet Protocol) e as tecnologias associadas a três eixos principais de análise: visão orientada a coisas, visão orientada a internet e visão orientada a semântica dos dados (SÔNEGO; 2016). A Figura 1 apresenta este modelo agrupando as tecnologias em cada um dos grupos.

Figura 1 – Paradigma de Internet das Coisas



Fonte: Sonego (2014).

Uma das áreas que se relaciona à aplicação da Internet das Coisas é a área de monitoramento energético e ambiental, o que sugere um uso adequado e consciente dos recursos naturais (SÔNEGO; MARCELINO; GRUBER; 2016). E uma das áreas que requer constante monitoramento é a área de *data centers* que nada mais são do que "equipamentos eletrônicos utilizados para processamento de dados (servidores), armazenamento de dados (equipamentos de armazenamento) e comunicações (equipamentos de rede) (GENG, 2015).

Coletivamente, estes equipamentos processam, armazenam e transmitem dados digitais e utilizam para isto equipamentos especializados de conversão e backup de energia para manter uma potência confiável e de alta qualidade, bem como equipamentos de controle ambiental para manter a temperatura e umidade adequadas para os equipamentos de tecnologia da informação e comunicação. (GENG, 2015, p. 4, tradução nossa).

2.1.1 MICROCONTROLADOR ARDUÍNO

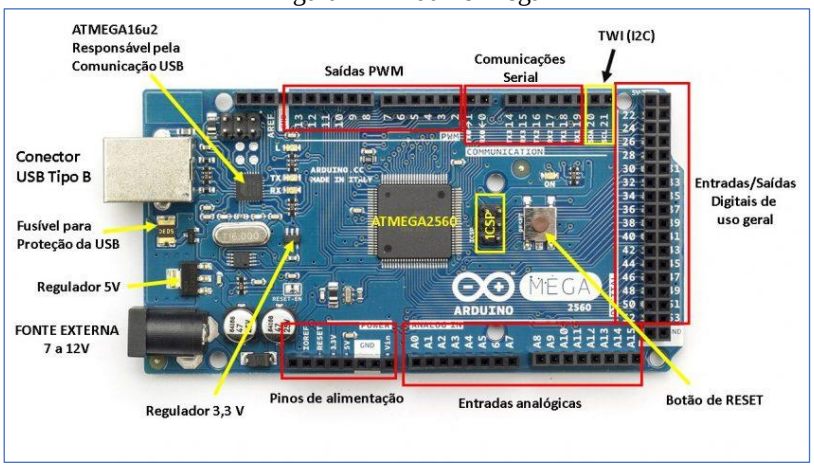
O Arduino é um microcontrolador de código aberto que pode ser facilmente programado, apagado e reprogramado a qualquer instante. Introduzido em 2005, a plataforma Arduino foi projetada para fornecer uma maneira barata e fácil para *hobby*, estudantes e profissionais criarem dispositivos que interagem com o ambiente usando sensores e atuadores. (LOUIS, 2016, pag. 21, tradução nossa). McRoberts (2011) afirma que o Arduino pode ser usado para desenvolver objetos interativos independentes ou ser conectado a um computador, a uma rede ou até mesmo a internet para recuperar e enviar dados do Arduino e trabalhar com eles, por exemplo enviar um conjunto de dados recebidos de sensores para um site ou aplicação, para serem exibidos no formato de um gráfico.

A plataforma utiliza-se de uma camada simples de software implementada na placa, que é um *bootloader*, e uma interface amigável no computador que utiliza a linguagem Processing, baseada na linguagem C/C++, a qual é também *open source*. Neste ambiente de desenvolvimento, são disponibilizadas bibliotecas que permitem o interfaceamento com outros dispositivos de hardware, permitindo o completo desenvolvimento de aplicações simples ou complexas em qualquer área. (SOUZA et. al., 2011, p.2).

A plataforma Arduino possui uma Interface de Desenvolvimento - IDE multiplataforma, isto é, há a possibilidade de utilizar a IDE com sistemas operacionais distintos, como, Windows, Linux, Mac OS. Esta característica facilita o desenvolvimento dos algoritmos, podendo ser escritos, alterados e enviados ao Arduino de qualquer Sistema Operacional suportado (MARCHESAN, 2012 p. 31).

Segundo Evans et al. (2013), existem várias versões do Arduino, porém todas baseadas na tecnologia Atmel AVR. Ele é um microcontrolador RISC de chip único com uma arquitetura Harvard modificada de 8-bits, desenvolvido em 1996 pela Atmel (SOUZA; 2014).A primeira placa foi baseada no ATmega8 rodando a uma velocidade de clock de 16 MHz com 8 KB de memória flash; mais tarde placas como o Arduino NG plus e o Diecimila foram equipadas com o ATmega168 e 16 KB de memória flash. As versões mais recentes do Arduino, Duemilanove e Uno, usam o ATmega328 têm 32 KB de memória flash. Para projetos que exigem mais E/S e memória, há o Arduino Mega1280 com 128 KB de memória e o Arduino Mega2560 com 256 KB de memória. O Mega oferece uma funcionalidade de entrada-saída significativamente maior em comparação com o Arduino padrão, então com o aumento da memória, é ideal para projetos maiores, que controlam muitos LEDs, possuem um grande número de entradas e saídas, ou precisam mais de uma porta serial de hardware. As placas têm 54 pinos de entrada-saída digitais, 14 dos quais podem fornecer saída analógica PWM, e 16 analógicos pinos de entrada (Figura 2).

Figura 2 – Arduino Mega



Fonte: Souza (2014).

2.1.2 SENSOR DE TEMPERATURA E UMIDADE DHT11

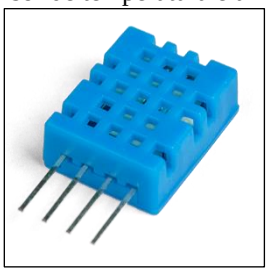
Conforme Patsko (2006),

> Em eletrônica, sensor é definido como aquilo que sente. Um sensor é conhecido como qualquer componente ou circuito eletrônico que permita a análise de uma determinada condição do ambiente, podendo ser algo como temperatura ou luminosidade; uma medida um pouco mais complexa como a rotação de um motor ou a distância de um carro até algum obstáculo próximo ou até mesmo eventos distantes do nosso cotidiano, como a detecção de partículas subatômicas e radiações cósmicas.(PATSKO, 2006, p. 4).

Os autores e especialistas em sensores classificam os em sensores passivos e ativos, pelo meio de detenção (meio biológico, químico, elétrico ou radioativo), pelo fenômeno de conversão, ou seja, entrada e saída (fotoelétricos, termoelétricos, eletroquímicos, eletromagnéticos, termo-ópticos) e finalmente entre analógicos e digitais, sendo essa última mais complexa (RAVI, 2017). Patsko (2006), afirma que esta divisão entre analógico e digital é feita de acordo com a forma a qual o componente responde à variação da condição, os analógicos são mais comuns e têm essa designação pois baseiam-se em sinais analógicos, que mesmo limitados entre dois valores de tensão, podem assumir infinitos valores intermediários. Já os sensores digitais baseiam-se em níveis de tensão bem definidos, tais níveis de tensão podem ser descritos como alto ou baixo, ou simplesmente “1” e “0”. Ou seja, esses sensores utilizam lógica binária, que é a base do funcionamento dos sistemas digitais.

O DHT11 (Figura 3), é um sensor de temperatura e umidade composto, que possui uma saída de sinal digital calibrada e faz uso de um componente de medição de umidade do tipo 32 resistivo e um componente de medição de temperatura NTC (Negative Temperature Coeficient), o qual se conecta a um microcontrolador de 8-bits de alto desempenho para verificar o ar ambiente, oferecendo uma resposta rápida e capacidade de anti-interferência. (CÂMARA, 2016, p. 31-32).

Figura 3 – Sensor de temperatura e umidade DHT11



Fonte: RoboCore (2018).

O sensor DHT11 é de tamanho pequeno, e pode ser aplicado em diversas áreas tais como, em sistemas HVAC (*heating, ventilation, and air-conditioning*), coletores de dados, desumidificadores, eletrodomésticos, estações meteorológicas, reguladores de umidade, na área médica, ou outro aplicativo que realize o controle ou medição de umidade e temperatura. A Figura 4 mostra algumas de suas características. (CÂMARA, 2016 p. 32).

Figura 4 - Caraterísticas técnicas do DHT11

Alimentação	3 - 5.5V
Faixa de leitura - Umidade	20 - 80 %
Precisão - Umidade	5%
Faixa de leitura - Temperatura	0 - 50 °C
Precisão - Temperatura	+/- 2 °C
Intervalo entre medições	1s

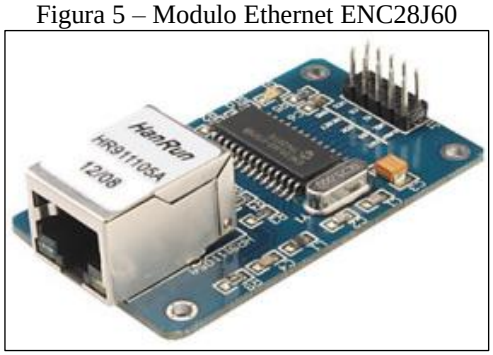
Fonte: Cia (2015).

2.1.3 MODULO ETHERNET ENC28J60

O ENC28J60 é um controlador rede independente com uma Interface Periférica Serial padrão, projetado para servir como interface de uma rede Ethernet para qualquer controlador equipado com a mesma Interface Periférica Serial – SPI. Atende a todas as especificações IEEE 802.3 e incorpora vários esquemas de filtragem de pacotes para limitar os pacotes de entrada. Ele também fornece um módulo DMA interno para processamento rápido de dados e cálculo de soma de verificação assistida por hardware, que é usado em vários protocolos de rede. (INC, 2006).

A comunicação com o controlador host é implementada através de um pino de interrupção e do SPI, com taxas de clock de até 20 MHz. Dois pinos dedicados são usados para link de LED e indicação de atividade de rede. (INC, 2006). O Modulo Ethernet ENC28J60 pode ser visto na Figura 5.

Figura 5 – Modulo Ethernet ENC28J60

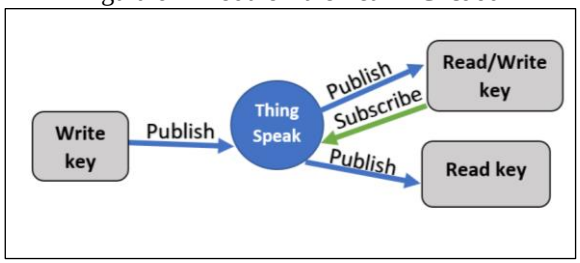


Fonte: Reis (2014).

2.1.4 THINGSPEAK

A plataforma ThingSpeak é um serviço aberto que oferece uma infraestrutura web e protocolo baseado em *Http* para o envio e recebimento de dados gerados pela plataforma Arduino ou qualquer outro dispositivo com recursos para comunicação em rede. Os dados são armazenados no site web e podem ser acessados publicamente ou privadamente, dependendo da configuração do usuário. Este serviço permite a criação de um canal de dados que contém oito campos capazes de comportarem qualquer tipo de dados, além de três campos para dados de localização e um para dados de status. (AZEVEDO JUNIOR, 2016, pag. 40). A Figura 6 ilustra uma das formas de funcionamento do ThingSpeak.

Figura 6 – Modulo Ethernet ENC28J60



Fonte: elaborado pelo autor.

Para publicar os dados na plataforma é preciso utilizar uma chave *'write key'* a qual é disponibilizada após o canal ser criado. De forma análoga, existe também a chave *'read key'*, usada para acessar os dados do canal (caso ele esteja configurado como privado) (MAUREIRA, et. al, 2016, pag. 2, tradução nossa).

O ThingSpeak oferece uma série de aplicações para serem usadas com os dados obtidos. Elas podem ser processadas e exibidas na forma de lista, indicadores visuais ou gráficos que podem ser personalizados usando modelos pré-definidos e até mesmo integrada ao software Matlab (MATHWORKS, 2016).

2.2 TRABALHOS CORRELATOS

Foram localizados os seguintes trabalhos correlatos: Camargo et al. (2015), Lazo (2014) e Alves et al. (2014).

Camargo et. al. (2015) (Quadro 1) desenvolveram o GreenHop, uma solução cuja finalidade é manter a sala de servidores dentro das normas técnicas aceitáveis para data center, tendo em vista aumentar a durabilidade dos equipamentos e oferecer uma gerência personalizada e eficiente das variáveis ambientais no local. Para tal, os autores utilizaram um grupo de sensores que efetuam o monitoramento do ambiente, sendo subdivido em dois grupos chamados de nodo final e nodo coordenador. O nodo coordenador é um computador single board Banana Pi que recebe dados dos nodos finais e armazena-os num software denominado Zabbix, este por sua vez envia comandos para os aparelhos de ar-condicionado.

Quadro 1 – Trabalho Correlato 1

Referência	Camargo, et., al. (2015) - MONITORAMENTO AMBIENTAL OPEN SOURCE PARA DATA CENTER
Objetivos	Manter a sala de servidores dentro das normas técnicas aceitáveis para data center. Eficiência energética.
Principais funcionalidades	Monitora temperatura, umidade, pressão atmosférica e ponto de orvalho. Envio de comandos para aparelhos de ar-condicionado.
Ferramentas de desenvolvimento	Banana Pi Zabbix
Resultados e conclusões	Conforme Camargo (2015), foi elaborado um plano de testes para comparar o ambiente antes e depois da implantação do sistema, tendo como redução de 43,7% do consumo de energia pelos equipamentos de refrigeração.

Fonte: elaborado pelo autor.

Lazo (2014) (Quadro 2) descreve um sistema para controle de temperatura e automação residencial baseado no método de aprendizagem de máquina que utiliza a técnica de Modelos Ocultos de Markov (Hidden Markov Models – HMM) para aplicação em automação residencial tendo como foco minimizar a intervenção humana no controle dos equipamentos.

Quadro 2 - Trabalho Correlato 2

Referência	Lazo (2014) - CONTROLE DE TEMPERATURA PARA AUTOMAÇÃO RESIDENCIAL UTILIZANDO MODELOS OCULTOS DE MARKOV
Objetivos	Monitoramento da temperatura e automação residencial
Principais funcionalidades	Implementa um algoritmo de aprendizagem, o HMM (Hidden Markov Models).
Ferramentas de desenvolvimento	Rasperry Pi
Resultados e conclusões	O autor propôs um caso de estudo usando o HMM para identificar e aprender o perfil considerado confortável para o usuário, tendo concluído que o tempo considerado ideal para o algoritmo aprender e consequentemente criar o perfil perfeito para oferecer o conforto térmico desejado pelos usuários são de três semanas.

Fonte: elaborado pelo autor.

O trabalho de Alves (2014) (Quadro 3) descreve um estudo de caso de um sistema de monitoramento de temperatura e umidade em farmácias e almoxarifados e que foi instalado no Hospital Badim localizado no Rio de Janeiro. A implatanção do mesmo teve como objetivo a redução de custos, eficiência, precisão e um melhor controle no monitoramento dos medicamentos. O sistema é composto pela parte de hardware e software, as medições são realizadas num intervalo de 30 minutos e enviadas para o software na web via *Wi-Fi,* e um mecanismo de alertas por e-mail e SMS impede a perca dos medicamentos caso a temperatura fuja dos padrões estabelecidos.

Quadro 3 - Trabalho Correlato 3

Referência	Alves (2014) - ESTUDO DE CASO: SISTEMA PARA MONITORAMENTO DE TEMPERATURA E UMIDADE EM FARMÁCIAS E ALMOXARIFADOS.
Objetivos	Monitoramento de temperatura e umidade em farmácias e almoxarifados para redução de custos e um melhor controle no monitoramento dos medicamentos.

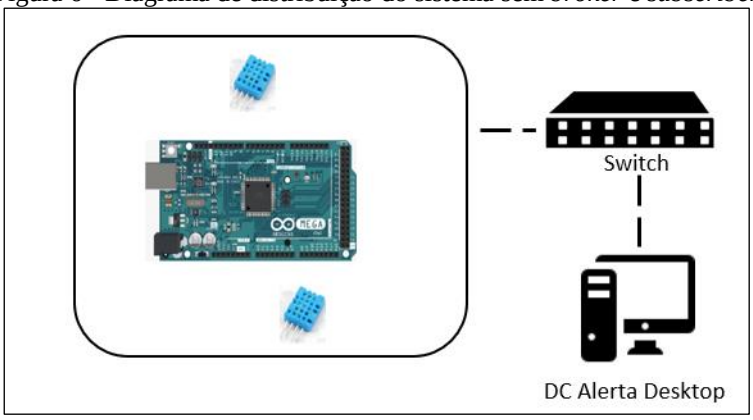
Principais funcionalidades	Monitoramento da temperatura
Ferramentas de desenvolvimento	Termômetros, conversores e microcomputador (não especificado).
Resultados e conclusões	Com a adoção do sistema a instituição passou a economizar 88% de R$12.000,00 destinados a três funcionários que faziam esse controle. Além de benefícios do monitoramento constante da temperatura (ALVES, 2014).

Fonte: elaborado pelo autor.

3 DESCRIÇÃO DO DC ALERTA

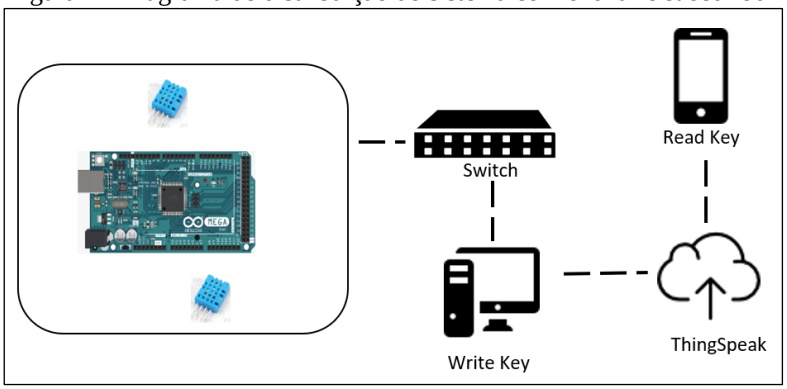
O DC alerta é uma solução composta de 4 módulos e pode operar em duas modalidades: sem *broker* e *subscriber* (Figura 7) ou com *broker* e *subscriber* (Figura 8). Na primeira forma de operação, os dados dos sensores são enviados para um servidor local o qual monitora os parâmetros estabelecidos na configuração inicial. Na segunda forma de operação, o servidor desktop é acoplado a um servidor ThingSpeak que atua como *middleware* para acesso via aplicativo Android desenvolvido.

Figura 6 - Diagrama de distribuição do sistema sem *broker* e *subscriber*



Fonte: elaborado pelo autor.

Figura 7 - Diagrama de distribuição do sistema com *broker* e *subscriber*



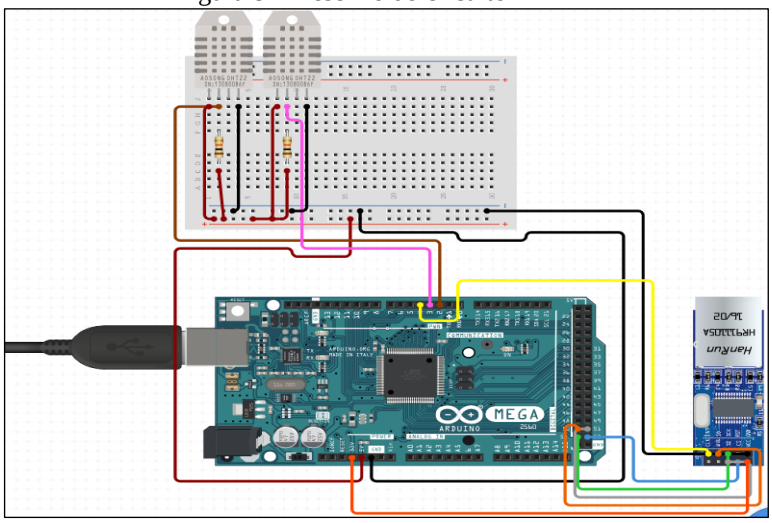
Fonte: elaborado autor.

A seguir é apresentada a descrição do hardware de monitoramento e posteriormente o software servidor desktop e o software Android. Os diagramas de casos de uso e de atividades são apresentados, respectivamente nas Figuras 14 e 15.

3.1 COMPONENTE DE HARDWARE

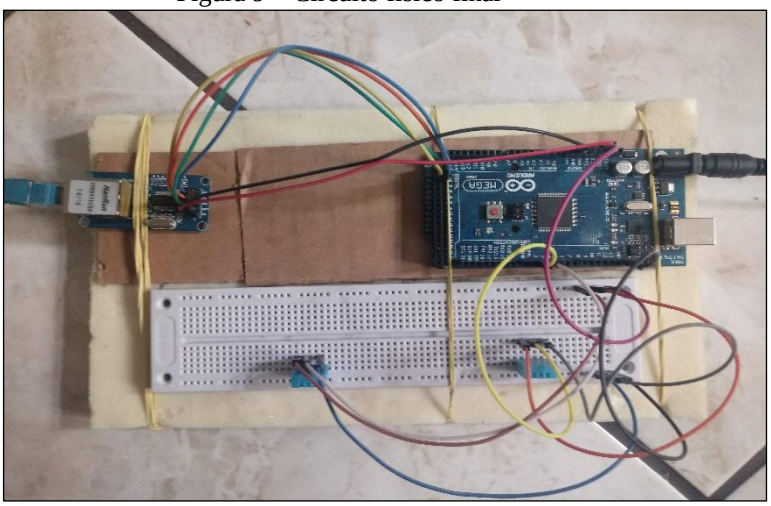
O módulo de hardware foi desenhado com a ferramenta *circuit.io* a qual serviu como um guia para a montagem do circuito físico e seleção dos componentes adequados. Foram utilizados o Arduino mega, dois sensores de temperatura e umidade DHTT11 e um módulo Ethernet ENC28J60. A Figura 9 ilustra o desenho no circuit.io e a Figura 10 o circuito físico final.

Figura 8 – Desenho do circuito



Fonte: elaborado pelo autor.

Figura 9 – Circuito físico final



Fonte: elaborado pelo autor.

A programação do microcontrolador, sensores e o modulo Ethernet foi feita através da IDE do Arduino versão 1.8.3. Uma das vantagens do Arduino é a facilidade de importação de bibliotecas prontas com uma única linha de código, facilitando uma boa parte do trabalho. A biblioteca "DHT.h" é responsável por estabelecer a comunicação digital entre os sensores e as portas do Arduino. A biblioteca "UIPEthernet.h" foi utilizada para viabilizar o funcionamento do modulo Ethernet permitindo a atribuição de um endereço MAC “virtual” bem como um endereço IP para que o Arduino possa se conectar a uma rede local.

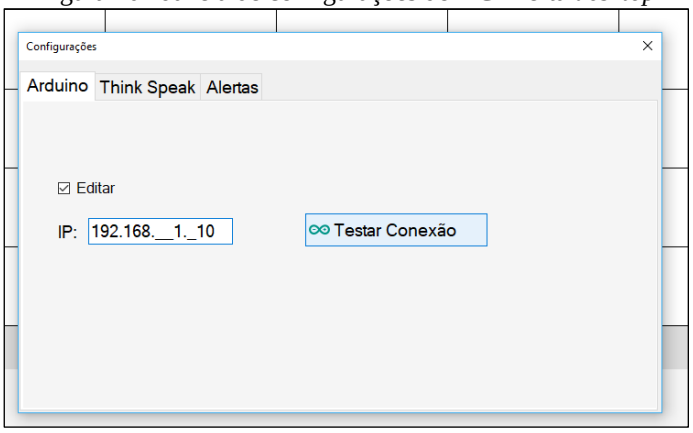
A estrutura de um programa para Arduino é geralmente resumida em dois métodos, *setup* onde é feita a inicialização dos componentes e o método *loop* que é executado infinitamente. Neste último é feita a leitura dos sensores num intervalo de 2000 milissegundos, e esses dados são disponibilizados em formato JSON para que possam ser acessados por qualquer *host* dentro da mesma rede através de algum programa específico.

3.2 COMPONENTES DE SOFTWARE: DC ALERTA *desktop*

Este módulo é o aplicativo responsável por ler os dados dos sensores disponibilizados pelo Arduino, processá-los, armazená-los e disponibilizá-los quando necessário. Foi desenvolvido utilizando-se: a linguagem de programação *C Sharp* C#, .Net 4.5.2 e o Banco de Dados *SQL*-Lite.

Ao ser executado, o sistema sempre verifica a existência do arquivo de Banco de Dados SQLite no diretório definido. Se este for encontrado, o sistema busca o IP do Arduino e tenta uma conexão. Caso a conexão seja bem-sucedida inicia-se o monitoramento da temperatura e umidade; caso haja insucesso na conexão, antes ou durante o monitoramento, o usuário é avisado, pois os motivos podem ser variados como IP incorreto ou outro problema na rede. Nas ocasiões em que o sistema é executado pela primeira vez em um determinado computador, o arquivo do Banco de Dados é criado e uma janela com três abas é mostrada ao usuário, sendo a primeira aba referente a configuração do IP do Arduino e é de preenchimento obrigatório já as abas ThingSpeak e Alertas são opcionais (Figura 11).

Figura 10 - Janela de configurações do DC Alerta *desktop*



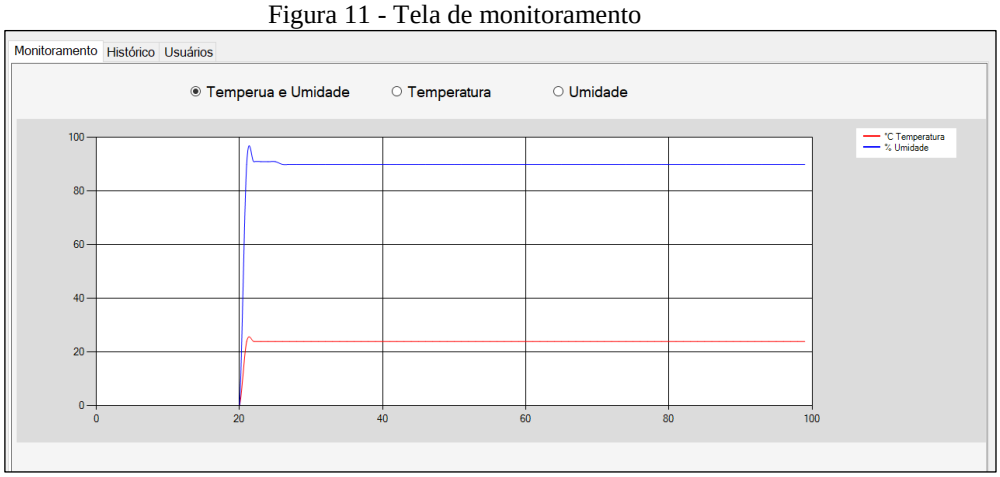
Fonte: elaborado pelo autor.

Na janela de configurações, as informações solicitadas nas abas ThingSpeak só são de caráter obrigatório para permitir que o sistema se conecte com o servidor ThingSpeak disponibilizando assim os dados de monitoramento para consumo do aplicativo DC Alerta mobile.

3.2.1 MONITORAMENTO

Uma vez que a conexão com Arduino foi bem-sucedida, o aplicativo inicia o monitoramento e as informações processadas de temperatura e umidade são apresentadas ao o usuário através de um gráfico em tempo real (Figura 12). Existe também a possibilidade de mostrar apenas a umidade ou temperatura.

Figura 11 - Tela de monitoramento



Fonte: elaborado pelo autor.

A estrutura do DC Alerta Desktop utiliza duas *threads:* uma para aquisição e desenho do gráfico e outra para envio dos dados para o servidor ThingSpeak. Se o aplicativo estiver rodando e por exemplo é desligado um cabo de rede, um novo processo é iniciado para tentar reestabelecer a conexão com o Arduino. O DC Alerta mantem o usuário informado sobre o seu estado de funcionamento, isso é, se a conexão com Arduino foi bem sucedida e se mantém, se existe conexão à internet, e se existe comunicação com ThingSpeak.

3.2.2 ALERTAS

Uma das propostas deste trabalho é alertar, notificar os usuários quando uma das variáveis monitoradas (temperatura ou umidade) estiverem fora dos parâmetros estabelecidos, enviando uma mensagem de e-mail pra os usuários cadastrados no Banco de Dados. Para cadastrar um usuário, é necessário informar o nome e o e-mail. Os usuários podem ser editados ou apagados a qualquer momento. Se não houver nenhum usuário cadastro, o sistema limita-se a exibir as informações no gráfico. Se durante o envio de uma mensagem, houver algum erro, o usuário também é informado.

O trecho de código exibido no Quadro 4 é executado na mesma *thread* em que ocorre o monitoramento. A cada leitura dos sensores, é feita uma verificação no banco, se este possui as informações de temperatura e umidade mínima e máxima para compará-las com as lidas. Se essa comparação retornar um valor verdadeiro, uma nova comparação para verificar se alguma variável se encontra fora do intervalo estabelecido situação em que mais uma comparação é feita, esta terceira é baseada no tempo decorrido em que foi enviado um último possível e-mail. Esta validação serve para evitar

algum tipo de bloqueio ou restrição dos serviços de e-mails, uma vez que a maioria desses tendem a classificar remetentes que enviem demasiadas mensagens num curto intervalo de tempo como suspeitos.

Quadro 4 - Funções que verificam a necessidade de envio de alertas

```
if (banco.LimiteAlertaInformado()) {
  if (temperatura < banco.TemperaturaMinima()
  || temperatura > banco.TemperaturaMaxima()
  || umidade < banco.UmidadeMinima()
  || umidade > banco.UmidadeMaxima()
    {
      if (enviouEmial && ((cronometro.Elapsed.Minutes
          * 60) + cronometro.Elapsed.Seconds) >= 100) {
              EnviarEmail(temperature, umidade);
          } else if (enviouEmial == false) {
              EnviarEmail(temperature, umidade);
          }
    }
 }
```

Fonte: elaborado pelo autor.

O Quadro 5 mostra o método responsável pelo envio dos e-mails de alerta. O método `EnviarEmail` recebe dois parâmetros, faz uma consulta no Banco de Dados, a fim de verificar a existência dos possíveis destinatários. O corpo da mensagem é constituído pela data e hora atualizada, bem como os valores da temperatura e umidade (ambos recebidos por parâmetro). Apesar da validação feita no cadastro de usuários, eventualmente podem ser cadastrados e-mails inválidos o que pode originar erros durante o envio de e-mails. Para contorno desta situação dentro da captura destes erros é invocado um *Delegate* que exibe temporariamente um ícone, com o significado de que algum problema surgiu durante o envio do e-mail. Assim o funcionamento da aplicação não é comprometido.

O DC Alerta implementa a seguinte regra: quando executado, a primeira leitura da temperatura e umidade sempre é salva no Banco de Dados e as próximas serão salvas apenas se houver alguma diferença entre os valores lido anteriormente e o mais recente, tanto para a temperatura como para a umidade. Esta solução objetivou evitar o armazenamento de dados desnecessários. Outro momento que o DC alerta salva os dados lidos é quando uma das variáveis estiver fora do intervalo estabelecido.

Quadro 5 - Método que envia e-mail

```
public void EnviarEmail(int temperatura, int umidade) {
   if (banco.selectUsuario().Count > 0) {
                for (int i = 0; i < banco.selectUsuario().Count; i++) {
                    MailMessage mail = new MailMessage();
                    mail.From = new MailAddress("dcalertafurb@gmail.com");
                    mail.To.Add(banco.selectUsuario()[i].Email);
                    mail.Subject = "Variaveis fora do intervalo";
                    mail.Body = "Data: " + retornaData_ou_Hora("data") +
                    "\n Hora: " + retornaData_ou_Hora("hora") +
                        "\n Temperatura: " + temperatura.ToString() +
                        "\n Umidade: " + umidade.ToString();
                        Try {
                            using (var smtp = new SmtpClient("smtp.gmail.com")){
                                smtp.EnableSsl = true;
                                smtp.Port = 587;
                                smtp.DeliveryMethod = SmtpDeliveryMethod.Network;
                                smtp.UseDefaultCredentials = false;
                                smtp.Credentials = new
                           NetworkCredential("dcalertafurb@gmail.com", "bola10xut");
                                smtp.Send(mail);
                                boolIconeEnvioEmaiSucesso = true;
                                boolMostraIcone = true;
                                cronometro.Restart();
                             if (this.pictureBoxEvniaEmail.InvokeRequired) {
                                DelegateEnvioEmail d = new
                           DelegateEnvioEmail(GerenciaIconeEnvioEmail);
                                this.Invoke(d, new object[] { });
                             }
                        }
                        } catch {
                            boolIconeEnvioEmaiSucesso = false;
                            boolMostraIcone = true;
                            cronometro.Start();
                        if (this.pictureBoxEvniaEmail.InvokeRequired) {
                                DelegateEnvioEmail d = new
                            DelegateEnvioEmail(GerenciaIconeEnvioEmail);
                                this.Invoke(d, new object[] { });
                            }
                        }
```

```
                    }
                }
            }
```

Fonte: elaborado pelo autor.

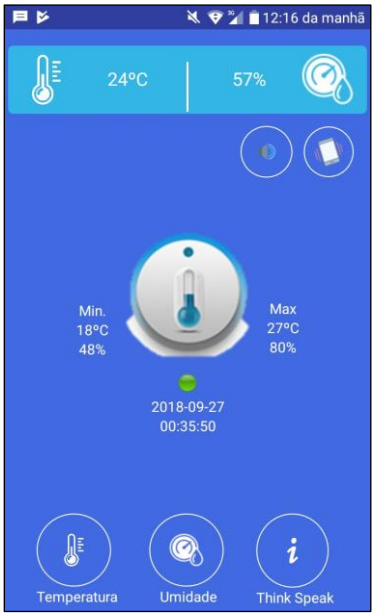
Para que o DC Alerta possa comunicar-se com o *broker* ThingSpeak foi necessário criar uma conta no site. O único requisito é o fornecimento de um e-mail válido. Após a criação da conta, foi preciso criar e configurar um canal, e configurar a estrutura dos campos a serem armazenados e seus respectivos tipos. Quando esta operação foi concluída o ThingSpeak forneceu um *ID Channel* e uma chave *write key*. Estes dados foram armazenados no banco de dados para acesso ao ThingSpeak quando necessário.

3.3 COMPONENTES DE SOFTWARE: DC ALERTA *mobile*

O módulo DC Alerta *mobile* (Figura 13) é um aplicativo desenvolvido para a plataforma Android e assume o papel de um *Subscriber* do ThingSpeak. Assim o aplicativo pode realizar o monitoramento da temperatura e umidade remotamente. Ele foi desenvolvido usando o Ambiente de Desenvolvimento Android Studio e a versão 8 da linguagem de programação Java.

O DC alerta *mobile* possui uma chave *read key* que foi gerada pelo ThingSpeak após a criação do canal e a partir dela é possível realizar requisições HTTP a cada 30 segundos e obter os dados deste canal. Após a aquisição, os dados são apresentados na tela do aplicativo. Se os valores recebidos estiverem fora dos parâmetros estabelecidos o dispositivo dispara uma campainha e começa a vibrar alertando o usuário.

Figura 12 - Tela de monitoramento do DC Alerta *mobile*



Fonte: elaborado autor.

4 RESULTADOS

O protótipo DC Alerta vem sendo desenvolvido desde a disciplina de Projeto de Software II. A partir dos primeiros resultados foi realizada uma revisão do projeto culminando com a solução descrita neste trabalho. Durante a fase de desenvolvimento e validação algumas dificuldades foram encontradas. Uma delas, relacionada com a comunicação serial com a porta COM no Windows. Ao ser desconectada a conexão com o Arduino, o Windows não permitia a reconexão. Este problema foi resolvido substituindo-se a conexão serial pela conexão ethernet. Outra dificuldade foi relacionada ao sensor de temperatura utilizado. O modelo DHT11 utilizado em dias com muita humidade gerou leituras fora das especificações que são valores compreendidos entre 20% a 80%. Pesquisando-se o problema constatou-se que a substituição do DHT11 por um modelo DHT22 resolveria o problema (apesar deste modelo ser mais caro). O tempo médio de testes da solução foi de 6 horas por dia durante dois meses.

O Quadro 6 apresenta um comparativo dos os trabalhos correlatos. O primeiro item a ser analisado é o hardware utilizado em cada trabalho desenvolvido. O Banana Pi utilizado no trabalho de Camargo, et., al. (2015), é um microcontrolador fortemente influenciado pelo Raspeberry Pi logo pode ser considerado uma extensão deste, sendo, portanto, similar ao que foi adotado neste projeto. Alves (2014) não mencionou o hardware utilizado. Os microcontroladores da família Raspeberry (Raspeberry Pi e Banana Pi) apresentam um custo de 60% a mais em relação ao Arduino mega porém possuem versões com muito mais recursos como Bluetooth, Wireless, mais *clock* de CPU e mais memória RAM.

Pela natureza do trabalho, Camargo (2016) utilizou vários sensores, mas não os especificou. Em outros trabalhos foram utilizados sensores distintos desde o LM35 que possui um valor de mercado baixo, Lazo (2014), afirma que este sensor foi escolhido porque apresenta boas características para monitoramento de temperatura em uma residência. Os termômetros STA e STB1 foram considerados ideias para trabalho de Alves (2014) pelo fato de registrar temperaturas muito baixas como por exemplo, dentro de *freezers* e geladeiras. O presente trabalho utilizou o DHT11 por ser um sensor de custo razoável e possuir maior precisão em relação ao LM35, é aconselhado seu uso em ambientes ao que o trabalho propôs.

No que tange a grandeza física monitorada pôde-se verificar que todos os trabalhos apresentam uma característica em comum que é o monitoramento da temperatura, mas Camargo (2016), além da temperatura e umidade também monitora ponto de orvalho e pressão atmosférica.

Quanto ao envio de alertas, apenas dois dos trabalhos têm está características: Alves (2014) e o presente trabalho, sendo considerado de extrema importância. Outra característica importante e considerado um diferencial para o trabalho desenvolvido é a validação dos dados, que consistiu na instalação de dois sensores, comparação dos valores lidos por ambos, significando algum possível problema quando esta comparação resultar em um valor alto.

Dos trabalhos apresentados apenas Lazo (2014), implementa um algoritmo de inteligência artificial que visa ajudar o sistema a aprender perfis adequados para seus usuários. A última comparação é sobre um suporte de monitoramento remoto ou por intermédio de um celular, sendo esta característica atribuída apenas para o presente trabalho.

Quadro 6 – Comparação com trabalhos correlatos

Trabalhos / Características	Camargo (2016)	Lazo (2014)	Alves (2014)	Aplicação Desenvolvida
Hardware principal	Banana Pi e Arduino	Raspberry Pi	Não informado	Arduino mega
Sensor	Vários (não especificados).	LM35	Termômetros STA e STB1	DHT11
Grandezas monitoradas	Temperatura, umidade, ponto de orvalho e pressão atmosférica.	Temperatura	Temperatura e umidade	Temperatura e umidade
Envio de alertas	Não	Não	Sim	Sim
Validação dos dados	Não	Não	Não	Sim
Algoritmo de inteligência artificial	Não	Sim	Não	Não
Monitoramento por celular	Não	Não	Não	Sim

Fonte: elaborado pelo autor.

5 CONCLUSÕES

O artigo descreveu o desenvolvimento do DC Alerta, um sistema de monitoramento da temperatura e umidade para *data centers* através de uma rede de objetos que constitui a Internet das Coisas. Para se chegar ao objetivo, foi construído um circuito eletrônico e duas aplicações, o DC alerta versão *desktop*, responsável por recolher os dados dos sensores e realizar o monitoramento local e o DC alerta versão *mobile* que recebe os dados de temperatura e umidade através de um serviço intermediário que usa o protocolo MQTT, possibilitando monitoramento remoto.

O circuito eletrônico foi idealizado e desenhado com ajuda da ferramenta circuit.io, que permitiu efetuar testes e selecionar os componentes adequados para o protótipo final. A plataforma Arduino possui uma IDE simples, uma linguagem facilmente programável e uma série de bibliotecas que facilitam o uso dos *Shields* e/ou módulos. O sensor de temperatura e umidade DHT11 atendeu as expetativas com a ressalva do range de operação relatado anteriormente. A solução para isto é a substituição do modelo utilizado no protótipo pelo modelo DHT22 que faz leituras mais precisas.

Cabe destacar que durante os testes realizados em um ambiente simulado, o DC Alerta fez o monitoramento da temperatura e umidade e enviou as alertas sempre que se fez necessário.

O desenvolvimento do protótipo serviu como prova de conceito da adoção de uma tecnologia de monitoramento ambiental que pode ser empregada para monitoramento de *data centers*. A solução desenvolvida permite que o monitoramento possa ser realizado a distância permitindo a equipe de suporte monitorar as condições ambientais dos servidores em horários de plantão, por exemplo.

A partir da experiência no desenvolvimento do protótipo, sugere-se como extensões ao presente trabalho as seguintes possibilidades:

a) adicionar um modulo *wi-fi* ao circuito para que seja possível acessar os dados do sensor numa rede sem fio;

b) substituir os sensores DHT11 pelos DHT22, a fim de aumentar a capacidade, precisão e eficiência na leitura das variáveis ambientais;

c) adicionar ao circuito eletrônico módulos de infravermelhos ou outros para envio de comandos para os parelhos de ar condicionado;

d) implementar algoritmos de inteligência artificial para detectar falha nos sensores;

e) implementar o protocolo MQTT diretamente no Arduino.

REFERÊNCIAS

ALVES, E. T. A.; MORIM, D. S.; SOUZA, M. N.; ROSÁRIO, F. F. Estudo de caso: Sistema para Monitoramento de Temperatura e Umidade em Farmácias e Almoxarifados. In: XXIV CONGRESSO BRASILEIRO DE ENGENHARIA BIOMÉDICA – CBEB 4, 2014. **Anais...** Rio de Janeiro, 2014. p. 1208-1211.

AZEVEDO JUNIOR, Amber Leite de. **Sistema de monitoramento e climatização de estufa de pequeno porte em um contexto doméstico**. 2016. 84 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia da Computação) - Faculdade de Tecnologia e Ciências Sociais Aplicadas – Centro Universitário de Brasília- UniCEUB, Brasília.

CÂMARA, Matheus Luiz N. **Monitoramento de temperatura e umidade de um ambiente utilizando o protocolo zigbee**. 2016. 64 f. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Redes de Computadores) - Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, Natal.

CAMARGO, Daniel S. et. MIERS, Charles C. Sensoriamento em sala de servidores baseado em software e hardware livres. In: SEMINÁRIO DE INICIAÇÃO CIENTIFICA UNIVERSIDADE DO ESTADO DE SANTA CATARINA, 25, 2015. **Anais...** Florianopolis: SIC UDESC, 2015. p. 1-2.

CIA, e Arduino. **Sensor de temperatura e umidade DHT22 (AM2302)**. [2015?]. Disponível em: <https://www.arduinoecia.com.br/2015/02/sensor-de-temperatura-e-umidade-dht22.html/>. Acesso em: 5 out. 2018.

EVANS, Martin; NOBLE, Joshua; HOCHENBAUM, Jordan. **Arduino in Action**. New York: Manning Publications, 2013.

GENG, Hwaiyu. **Data center handbook**, Palo Alto, CA: John Wiley & Sons, 2015.

INC, Microchip Technology. **ENC28J60 Data Sheet Stand-Alone Ethernet Controller with SPI Interface.** 2006. Relatório de pesquisa n. 240 arquivado na Microchip Technology Inc.

LAZO, Enzo Mendes M. **Controle de Temperatura para Automação Residencial Utilizando Modelos Ocultos de Markov.** 2014. 110 f. Trabalho de Conclusão do Curso (Bacharel em Engenharia Eletrônica) - Departamento de Engenharia Eletrônica da Universidade de Brasília, Brasília.

LOUIS, Leo. Working Principle of Arduino and Using it as a Tool for Study and Research. In: INTERNATIONAL JOURNAL OF CONTROL, AUTOMATION, COMMUNICATION AND SYSTEMS (IJCACS), 9, 2016. **Proceedings…** Ahmedabad: Ahmedabad University 2016. p. 1-9.

MARCHESAN, Marcelo. **Sistema de Monitoramento Residencial Utilizando a Plataforma Arduino**. 2012. 62 f. Trabalho de Conclusão do Curso (Tecnologo em Redes de Computadores) - Curso Superior de Tecnologia em Redes de Computadores, Universidade Federal de Santa Maria, Santa Maria.

MAUREIRA Marcello A. G.; OLDENHOF, Daan; TEERNSTRA, Livia. **ThingSpeak – an API and Web Service for the Internet of Things**. Leiden, [2011]. Disponível em: < http://mediatechnology.leiden.edu/images/uploads/docs/wt2014_thingspeak.pdf />. Acesso em: 5 out. 2013.

McROBERTS, Michael. **Arduino Básico.** São Paulo: Novatec, 2011.

PATEL, Keyur K.; PATEL, Sunil M. Internet of Things - IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future Challenges. **International Journal of Engineering Science and Computing**, Volume 6 Issue No. 5 May 2016. pp-6122-6131. DOI 10.4010/2016.1482.

PATSKO, Fernando Luís. **Tutorial Aplicações, Funcionamento e Utilização de Sensores**. Porto Alegre, [2006]. Disponível em: < http://www.maxwellbohr.com.br/downloads/robotica/mec1000_kdr5000/tutorial_eletronica_aplicacoes_e_funcionamento_de_sensores.pdf/>. Acesso em: 5 out. 2018.

RAVI. **Different Types of Sensors.** [2017]. Disponível em: <https://www.electronicshub.org/different-types-sensors/>. Acesso em: 13 out. 2018.

REIS, Valdinei Rodrigues dos. **Colocando o Arduino na rede com o ENC28J60 Ethernet Shield**. [2014]. Disponível em: < http://www.arduinobr.com/arduino/arduino_shield/colocando-o-arduino-na-rede-com-o-enc28j60-ethernet-shield/>. Acesso em: 10 out. 2018.

ROBOCORE. **Sensor DHT11 de Temperatura e Umidade**. [S1], [2018?]. Disponível em: <https://www.robocore.net/loja/produtos/sensor-de-temperatura-dht11.html?gclid=Cj0KCQiA2o_fBRC8ARIsAIOyQ-kdR0RBpWSNORbzID_bBvX0M8rhjqE15WEv9m5WM7iXXwpY4WtlQJEaAt1EEALwwcB/>. Acesso em: 10 out. 2018.

SATO, Silvio K. **Mobilidade, comunicação e consumo: expressões da telefonia celular em Angola, Brasil e Portugal.** 2015. 366 f. Tese (Doutorado em Ciências da Comunicação) - Curso de Pós-Graduação em Ciências da Comunicação, Universidade de São Paulo, São Paulo.

SÔNEGO, António S.; MARCELINO, Roderval; GRUBER, Vilson. A Internet das Coisas aplicada ao conceito de eficiência energética: uma análise quantitativo-qualitativa do estado da arte da literatura. **AtoZ: novas práticas em informação e conhecimento.** Florianópolis, v.5, n.2, P. 82-844, jul/set. 2016.

SOUZA, Anderson R.; PAIXÃO, Alexsander C.; UZÊDA, Diego D.; DIAS, Marco A.; DUARTE, Sergio; AMORIM, de Helio S. A placa Arduino: uma opção de baixo custo para experiências de física assistidas pelo PC. **Revista Brasileira de Ensino de Física**, Rio de Janeiro, v. 33, n. 1, p. 1-5, jan./mar. 2011.

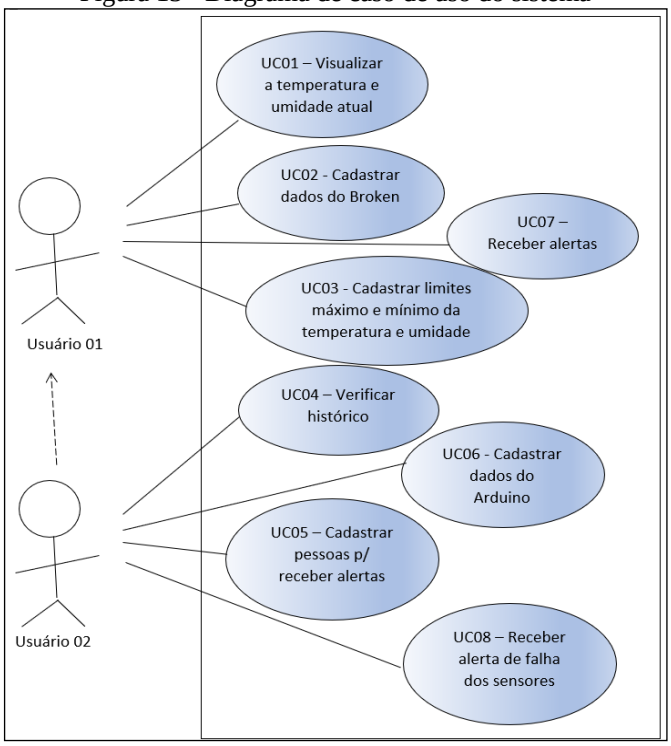
SOUZA, Fábio. **Arduino Mega 25600**. [2014]. Disponível em <https://www.embarcados.com.br/arduino-mega-2560/>. Acesso em: 13 out. 2018.

VERMESAN, Ovidiu; FRIESS, Peter. **Internet of Things - From Research and Innovation to Market Deployment**. Aalborg: River Publishers, 2014.

XANDÓ, Flavio. **PRTG – monitorando totalmente sua rede e infraestrutura**. São Paulo, [2012]. Disponível em <http://www.fxreview.com.br/2012/11/prtg-monitorando-totalmente-sua-rede-e.html/>. Acesso em: 30 set. 2018.

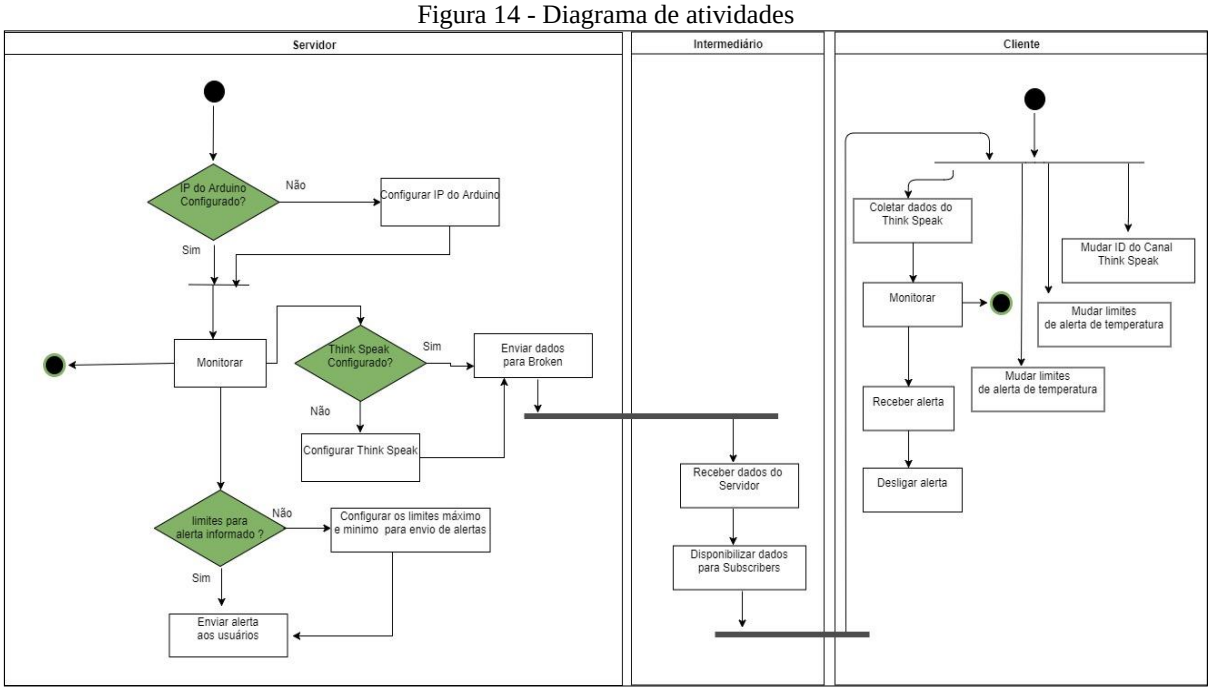
APÊNDICE A – DIAGRAMAS DE ESPECIFICAÇÃO

Figura 13 - Diagrama de caso de uso do sistema



Fonte: elaborado pelo autor.

Figura 14 - Diagrama de atividades



Fonte: elaborado pelo autor.