INTERFACE TECNOLÓGICA

SISTEMA DE MONITORAMENTO DE TEMPERATURA PARA DATACENTER BASEADO EM INTERNET DAS COISAS

DATACENTER TEMPERATURE MONITORING SYSTEM BASED ON INTERNET OF THINGS

João Paulo Lemos Escola – jpescola@ifsp.edu.br

Jovander da Silva Freitas – jovander@ifsp.edu.br

Instituto Federal de São Paulo (IFSP) – Barretos – São Paulo – Brasil

DOI: 10.31510/infa.v17i1.726

RESUMO

Os dados armazenados no *datacenter* da empresa são cruciais e fazem parte de seu ativo. Constantemente se buscam formas de garantir a segurança dos dados e equipamentos instalados no ambiente do *datacenter*. Um eficiente controle de temperatura com uso de aparelhos condicionadores de ar também é de indubitável importância para o gestor. Este trabalho apresenta uma solução para controle de temperatura do *datacenter*, permitindo ao gestor um recurso adicional que possibilita diminuição em sua carga de trabalho, sendo notificado em caso de falha no resfriamento do ambiente, garantindo o funcionamento e a vida útil dos equipamentos presentes no *datacenter*. Utilizando o microcontrolador *Orange Pi,* módulo de captura da temperatura, um algoritmo eficiente e alertas por e-mail, seu custo é dezoito vezes menor do que os concorrentes do mercado.

Palavras-chave: Monitoramento, IOT, Baixo Custo, Temperatura, Umidade, *Orange Pi*.

ABSTRACT

The data stored in the company's datacenter is crucial and is part of its assets. The forms of camera bus constantly guarantee the security of data and equipment installed in the datacenter environment. An efficient temperature control with the use of air conditioners is also important for the manager. This work presents a solution for temperature control of the datacenter, allowing the manager an additional resource that allows to decrease the workload, being notified in case of failure in the cooling of the environment, allowing the use and the useful life of the equipment present in the datacenter. Using the Orange Pi microcontroller, temperature capture module, an efficient algorithm and email alerts, its use is ten times less than the market applications.

Keywords: Monitoring, IOT, Low Cost, Temperature, Humidity, Orange Pi.

1 INTRODUÇÃO

A Tecnologia da Informação (TI), na figura do administrador ou gerente de TI, tem a tarefa de prover a estabilidade dos equipamentos e sistemas utilizados na empresa. Sob sua responsabilidade estão os servidores de rede, *switchs*, roteadores, cabeamentos e demais componentes de hardware que fazem parte dos ativos da companhia. Também os *softwares* e sistemas estão sob sua tutela, necessitando que os mantenha em perfeito funcionamento durante, na maioria dos casos, vinte e quatro horas por dia, nos sete dias da semana. Um dos requisitos mais importantes para a manutenibilidade do *datacenter* é a garantia de adequada refrigeração. Para isso são instalados aparelhos de ar condicionado responsáveis por manter a temperatura e umidade da sala em níveis apropriados (SCOFIELD e WEAVER, 2008) (WOODS, 2010).

Diversas são as formas que os administradores tem à disposição para fazer o monitoramento dos equipamentos de rede e parque de computadores de uma empresa. O protocolo SNMP (*Simple Network Management Protocol*), descrito em STALLINGS (1998) é um exemplo de recurso que permite o monitoramento em tempo real dos equipamentos por meio da captura constante de dados via *ethernet*, não sendo possível, entretanto para monitora mento de equipamentos condicionadores de ar, por não serem considerados como parte do conjunto de equipamentos gerenciados pelo administrador de rede. Dessa forma, o monitoramento pode ser realizado por meio da medição real da temperatura da sala, utilizando um termômetro ou dispositivo similar. A maioria desses, apresenta o resultado em um visor (*display*) mas não se trata de um dispositivo ativo, ou seja, capaz de interagir com outro equipamento ou enviar dados de aviso e alerta. Assim, um dispositivo que garanta o monitoramento dos níveis ideais de temperatura e umidade pode ser uma ferramenta importante para o administrador do *datacenter*, evitando problemas com manutenção e diminuindo riscos de pane ou interrupção dos serviços.

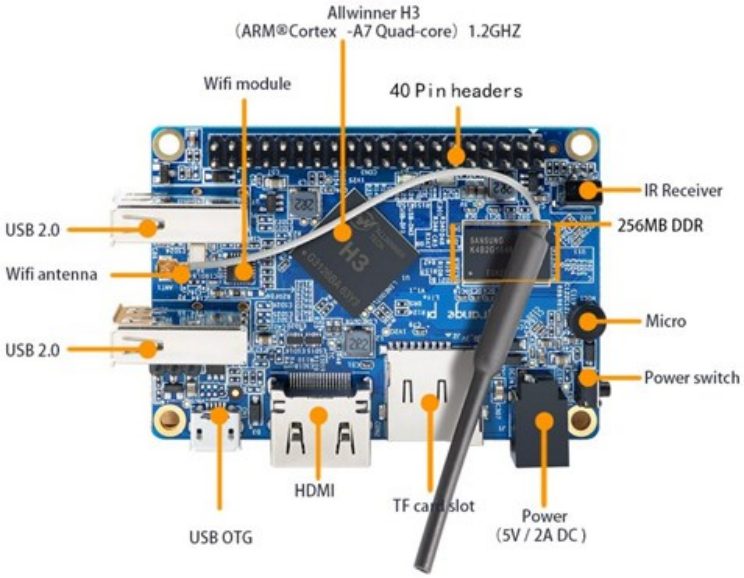
O presente trabalho evidencia a necessidade de controle de temperatura e umidade no ambiente interno do *datacenter* e propõe uma solução prática. Concluindo as informações introdutórias para melhor entendimento do presente trabalho, nos capítulos a seguir serão explanados os tópicos relacionados ao seu desenvolvimento.

2 FUNDAMENTAÇÃO TEÓRICA

De acordo com Atzoei et al. (2010) a internet das coisas (IOT) é um paradigma que vêm crescendo no cenário das telecomunicações e, graças à evolução da indústria dos semicondutores, os componentes necessários para a construção de um computador agora cabem em um *chip* de tamanho reduzido e baixo custo.

O *Orange Pi*, apresentado na Figura 1, é uma plataforma *OpenSource* de baixo custo para desenvolvimento de projetos de eletrônica, permitindo a integração de sistemas com dispositivos eletrônicos diversos. Possui características favoráveis para IOT, tais como: tamanho reduzido de 69mm por 48mm, baixo consumo de energia, características essas que permitem flexibilidade nos projetos, além da possibilidade de inclusão dos módulos compatíveis com outras tecnologias IOT como a plataforma Arduino (ORANGE_PI, 2016).

Figura 1. Orange Pi Lite



Fonte: ORANGE_PI (2016)

Graças à possibilidade de inclusão de um sistema operacional de rede, conforme indicada por WETHERALL e TANENBAUM (2011), os microcontroladores como o *Raspberry Pi* e *Orange Pi*, permitem que aplicações de baixo custo possam ser desenvolvidas para solucionar problemas que vão desde automação residencial até pesquisas científicas aplicadas.

3 PROCEDIMENTOS METODOLÓGICOS

A presente pesquisa se classifica como uma pesquisa exploratória de caráter experimental onde são apresentados os procedimentos para desenvolvimento de um protótipo tecnológico de dispositivo de processamento de sinais de temperatura e umidade do ambiente do *datacenter*.

O protótipo foi desenvolvido em quatro etapas. Na primeira etapa foi desenvolvida uma pesquisa bibliográfica e busca de soluções similares disponíveis no mercado. Na segunda etapa foi realizada definição e a aquisição dos componentes necessários para a construção do protótipo. A terceira etapa foi responsável pelo desenvolvimento dos algoritmos de captura e gerenciamento dos dados, bem como de apresentação, de forma amigável ao usuário final. A quarta e última etapa consistiu na implantação do referido protótipo em um *datacenter* de uma unidade de ensino. Essas etapas serão descritas em maiores detalhes a seguir.

Durante o processo de análise de viabilidade do projeto, primeira etapa, os equipamentos comumente empregados para projetos do tipo Internet das Coisas, conforme (IOT – Internet Of Things), melhor detalhado em GUBBI et al. (2013), foram cogitados, prioritariamente a plataforma Arduino, de acordo com BLUM (2013). Isso devido ao suporte bibliográfico disponível na literatura e fundamentalmente na Internet. Entretanto, essa opção, no projeto atual, foi substitutiva pela plataforma *Orange Pi,* de acordo com ORANGE_PI (2016), devido a possibilidade de inclusão de um sistema operacional de rede, o que permitiu uma solução mais estável e segura em relação ao acesso remoto ao dispositivo e também a disponibilidade online da listagem de registros capturados em tempo real. Apesar de ser possível a incorporação de um módulo de rede no Arduino, essa solução não se mostrou atrativa principalmente pela questão da segurança,uma vez que os sistemas operacionais e seus aplicativos provém atualizações constantes por suas comunidades, mostrando-se assim, mais apropriada a adoção da plataforma *Orange Pi*.

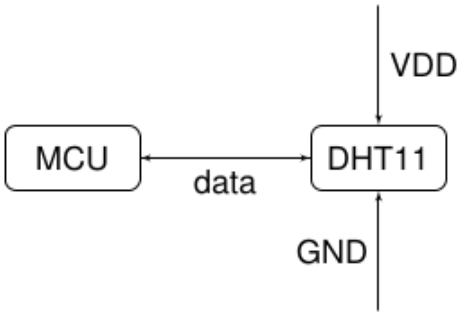
O modelo *Lite,* do *Orange Pi,* foi escolhido por possuir configuração compatível com os requisitos mínimos para executar os serviços levantados como soluções para o presente projeto. Esse modelo possui memória RAM (512 MB), processador Quad-core ARM 1.2GHz e interface de rede WiFi. Este último facilitou a implantação, pois permitiu a integração com a rede local sem a necessidade de instalação de um cabo de rede próximo ao protótipo.

O sistema operacional escolhido para ser instalado no dispositivo é o Armbian, melhor detalhado em ARMBIAN (2019), que trata-se de uma versão baseada na distribuição Ubuntu

Server do sistema operacional Linux. Dessa forma, todos os recursos que podem ser disponibilizados em um servidor de redes de hardware tradicional com Sistema Operacional Linux,s emelhantemente podem ser implantados em um dispositivo de baixo custo rodando o Armbian. Essa característica permite desenvolver projetos extremamente complexos em um dispositivo de tamanho reduzido e com baixo consumo de energia.

Quando o assunto é IOT, conforme GUBBI et al. (2013), existem diversos recursos que podem ser incorporados por meio de módulos, conhecidos como *shields* (módulos) compatíveis com a maioria das plataformas IOT. Dessa forma, foi incorporado um módulo de temperatura e umidade, modelo DHT11, conforme diagramado em TIANLONG (2010), detalhado em MOUSER_ELECTRONICS (2019) e em UUGear (2018). Trata-se de um sensor de temperatura e umidade de baixo custo para aquisição digital, composto por um componente resistivo de medição de umidade e um termistor para temperatura. Um termistor é um semicondutor sensível à temperatura, ou seja, sua resistência diminui com o aumento da temperatura, confirmado em MOUSER_ELECTRONICS (2019). Também é composto por um microcontrolador de 8 bits embutido, responsável por fornecer a resposta (data), conforme o esquema da Figura 2. Opera a uma tensão de entrada 3V a 5.5V de corrente continua.

Figura 2. Funcionamento do módulo DHT11



Fonte: Os autores (2020)

A integração dos módulos com o MCU (*Microcontroller Unit*) é feita via pinos GPIO (*General Purpose Input Output*), que é a interface, ou seja, o conjunto de pinos de entrada e saída que podem ser gerenciados via softwares que podem ser desenvolvidos pelo usuário dentro do Sistema Operacional implantado no dispositivo. No caso do *Orange Pi Lite* existem 40 pinos, incluindo pinos de alimentação de entrada de 3.3V, 5V e GND, assim sendo, nem todos os pinos são utilizados para entrada/saída de dados (GPIO), de acordo com o descrito

em WIRING_PI (2018). A importação dos dados do DHT para o dispositivo, no caso do diagrama o MCU, é feito por meio de um protocolo próprio de transmissão, que consiste em um conjunto de 40 bits encapsulados por meio de pulsos entre 26 e 28 microssegundos (bit 0) e pulsos de 70 micro segundos (bit 1). O formato do pacote de transmissão é ilustrado na Tabela 1.

Tabela 1. Formato do pacote de transmissão do DHT11.

Bits	Descrição
0-7	Umidade (parte inteira)
8-15	Umidade (parte decimal)
16-23	Temperatura (parte inteira)
24-31	Temperatura (parte decimal)
32-39	Controle (checksum)

Fonte: Os autores (2020)

Na segunda etapa do projeto, um ponto importante de estudo foi a busca por dispositivos considerados de baixo custo. Como um dos requisitos do presente projeto foi o foco no baixo custo, algumas tecnologias se diferenciam, como o caso da plataforma *Raspberry Pi* que provê basicamente os mesmos recursos da plataforma *Orange Pi*, mas,e m geral, com maior custo em relação a soluções de tamanho reduzido. A maioria dos componentes pode ser encontrada em lojas de eletrônica ou em sites de lojas virtuais na Internet. A disponibilidade cada vez maior de lojas virtuais disponíveis com produtos do ramo de eletrônica e soluções de automação comercial, mostra-se uma vantagem para o engenheiro iniciante e estudos voltados a IOT. Terminada a segunda etapa do projeto, seguiu-se para a próxima etapa, que consistiu na implementação dos algoritmos para acesso às informações do módulo. O primeiro algoritmo, responsável por coletar os dados do módulo DHT11, detalhado em UUGear (2018) foi desenvolvido na linguagem de programação C (Apêndice A). A coleta dos dados é feita por meio do pino ”data”. O algoritmo envia um sinal de requisição e recebe os dados no formato de pacotes de 40 bits, conforme descrito anteriormente na Tabela 1. A biblioteca ”wiringPi.h” permite a integração dos módulos com o MCU (WIRING_PI, 2019). Por meio dela, é possível utilizar as funções de envio (output) e recebimento (input) de dados. A função *pinMode* permite receber ou enviar dados para qualquer módulo, utilizando como parâmetros o endereço do pino e o modo *OUTPUT* para

leitura ou *INPUT* para envio de dados. As funções *digitalRead* e *digitalWrite* permitem leitura e escrita no pino especificado, respectivamente (Apêndice A). O dispositivo também inclui um *display* de cristal liquido (LCD), modelo LCD1602 que apresenta os dados de temperatura, umidade, data e hora (LI et al., 2016). O LCD1602 tem 16 pinos que podem ser ligados em paralelo com o MCU via GPIO, entretanto, com o módulo I2C LCD2004 é possível reduzir o número de ligações com o GPIO para apenas quatro: VCC, GND, SDA (Serial data) e SCL (Serial clock) (MANTECH_ELECTRONICS, 2017) (SUNFOUNDER, 2018). O algoritmo do Apêndice B é responsável por apresentar os dados no *display* LCD, por meio de dois argumentos de linha de comandos, sendo o primeiro argumento apresentado na linha superior e o segundo argumento na linha inferior do *display*. A biblioteca *liblcm1602*, quando compilada, cria a interface "i2c-0" que é mapeada por meio do diretório "/dev" do Linux. Com isso, por meio do algoritmo, podemos interagir com omódulo LCD através do módulo LCD2004 utilizando a referida biblioteca (GRAZIOLI, 2018). Uma vez que os dados de monitoramento estão disponíveis, partiu-se para o desenvolvimento de um algoritmo, utilizando a linguagem *shell script*, em conformidade com NEVES (2008), para executar o procedimento de coleta dos referidos dados e armazenamento em um arquivo de texto, juntamente com a data e hora da coleta, conforme o diagrama da Figura 4. O registro periódico das coletas permite ao administrador rastrear possíveis oscilações de temperatura e umidade que possam ocorrer no *datacenter* mesmo em períodos anteriores, podendo, até mesmo, detectar diminuição do desempenho do aparelho de ar condicionado causados por possíveis problemas internos ou sujeira. O algoritmo principal (Apêndice C), executa o comando de captura dos dados do sensor e armazena os dados em disco. Esse armazenamento é feito no formato de texto puro (padrão ASCII) de uma linha por registro com dados separados por vírgula. Em seguida, o algoritmo atualiza os dados no *display* e verifica se a temperatura/umidade estão dentro do limiar de aceitação, caso contrário um e-mail é disparado para o administrador. O algoritmo é executado a cada 5 minutos por meio de uma configuração no serviço de agendamentos cron (NEVES, 2008).

A função que define se o alerta deve ser disparado a cada coleta de temperatura t e umidade h dentro do algoritmo é definida pela equação da Figura 3:

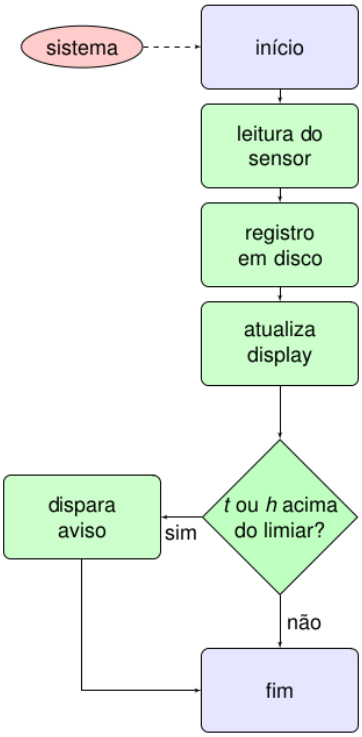
Figura 3. Equação Temperatura/Humidade

$$f(t,h) = \begin{cases} true & \text{if } t > 30 \text{ or } h > 90 \\ false & \text{otherwise} \end{cases}$$

Fonte: Os autores (2020)

Os valores dos limiares foram definidos empiricamente e podem ser ajustados de acordo com o desejo do administrador.

Figura 4. Algoritmo Coleta



Fonte: Os autores (2020)

A interface de comunicação de rede do *Orange Pi* acontece via rede sem fio e o acesso remoto do administrador acontece via protocolo SSH (*Secure Shell*), melhor descrito em YLONEN et al. (2006), na rede local ou por rede pública via VPN (*Virtual Private Network*) (WETHERALL and TANENBAUM, 2011) (WILSON, 1999). O acesso público dos dados de monitoramento pode ser feito via Internet de qualquer navegador, para isso, foi necessário o desenvolvimento de uma aplicação em PHP, em consonância com PHP.NET (2018) e

anexado no Apêndice D, responsável por carregar o arquivo de registro e apresentar uma interface amigável ao usuário. Outro algoritmo implementado foi o responsável pelo envio de e-mails de avisos disparado em caso de ocorrência. Esse algoritmo utiliza a biblioteca PHPMailer (PHPMAILER, 2018). A instalação do servidor Apache, detalhado em APACHE (2019) e dos recursos PHP possibilitou a interpretação do algoritmo no sistema operacional do dispositivo. O algoritmo consiste na leitura do arquivo de registro (*log*) por meio de uma rotina que extrai os dados de cada linha do arquivo, que estão armazenados em um formato separado por vírgulas, carregando-os em uma variável para posteriormente apresentar em uma tabela formatada na página Web. A quarta etapa do projeto consistiu na implantação do protótipo em um *datacenter* real. O *datacenter* da unidade de ensino foi escolhido por questões de facilidade de disponibilidade e por ser, seu controle de temperatura, uma das preocupações dos responsáveis.

4 RESULTADOS E DISCUSSÃO

Foram encontradas diversas soluções no mercado, durante a primeira etapa do projeto,que consistiu no levantamento de soluções comerciais ou gratuitas que tenham objetivos similares aos do presente trabalho. A tabela 2 apresenta um comparativo em relação aos custos dos dispositivos encontrados comercialmente disponíveis.

Tabela 2. Modelos disponíveis no mercado.

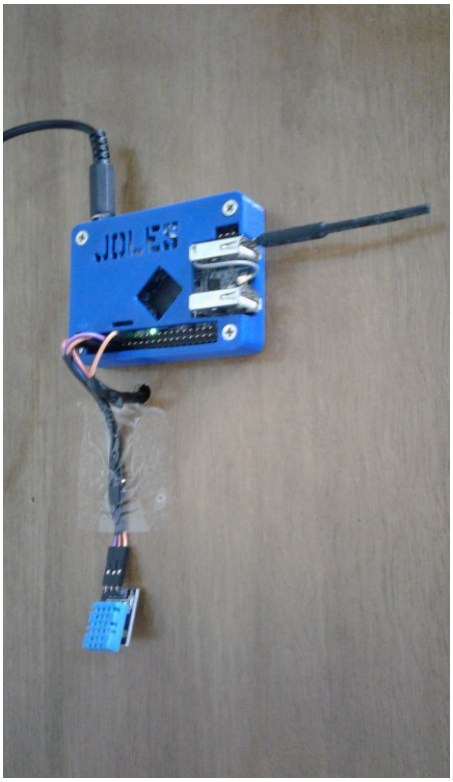
Dispositivo	Preço US$
Term-2S	876,39
SE-10	480,10
i-DL Ethernet WEB	N/A

Fonte: Os autores (2020)

O dispositivo proposto foi acondicionado em uma caixa plástica personalizada, que tem função de proteção permitindo acesso às interfaces da placa. Com uma dimensão final de 120mm por 100mm, na disposição apresentada na Figura 5, ao qual foi implantado em um *datacenter* de uma unidade de ensino, sendo instalado na porta de entrada e o *display* afixado no lado de fora da porta, conforme apresentado na Figura 6. Conforme citado anteriormente, a função do *display* é apresentar a temperatura e umidade atual da sala ao usuário que esteja

passando pela porta, sem que haja a necessidade de adentrar a sala, o que poderia resultar em oscilação da temperatura interna da sala. O fato de o dispositivo ser afixado por dentro da sala permite maior segurança, visto que, somente o administrador com a chave da sala pode ter acesso ao dispositivo, impedindo o acesso por leigos ou profissionais mal intencionados. Para fins de implantação, que se mostrou razoavelmente simples, somente dois procedimentos foram necessários: furação da porta e instalação de tomada de energia elétrica. No caso da furação da porta, bastou um pequeno orifício para passagem de cabos *flat* para possibilitar a interligação do dispositivo com o *display*. Para fins de alimentação, este pode ser desnecessário caso exista uma tomada de fácil acesso próxima à porta.

Figura 5. Protótipo do dispositivo



Fonte: Os autores (2020)

Figura 6. *Display* do dispositivo

Fonte: Os autores (2020)

5 CONCLUSÃO

A implantação do referido protótipo em uma unidade de ensino possibilitou aumento considerável da segurança do parque de equipamentos alocados no *datacenter*, visto que, apesar de haver redundância de equipamentos condicionadores de ar no estabelecimento, não havia como conhecer incidentes de falha de refrigeração antes que problemas mais graves acontecessem. Durante o processo de testes, após o período de implantação, houve ocorrência de falha de refrigeração onde o dispositivo disparou aviso por e-mail ao administrador que pode se deslocar ao *datacenter* e realizar os procedimentos de desligamento manual dos servidores antes que os efeitos colaterais fossem agravados. Anteriormente à implantação houve um incidente que ocasionou prejuízo financeiro a referida unidade escolar.

O custo total do protótipo ficou abaixo de US$ 27,00, sendo este valor dezoito vezes menor do que o dispositivo mais barato encontrado no levantamento de mercado e milhares de vezes menor do que o custo nominal dos equipamentos alocados no *datacenter*. Além do baixo custo, outra vantagem do protótipo está na possibilidade de personalização, visto que, por utilizar dispositivos de plataformas de código aberto, pode ser aprimorado por qualquer usuário qualificado, podendo incorporar recursos e refinar códigos- fonte. Os módulos, também devido ao custo bastante reduzido, podem ser facilmente encontrados em lojas especializadas e na Internet.

Para trabalhos futuros, existe o intuito de desenvolver aprimoramentos como a possibilidade de gerenciamento via interface Web, possibilitar análise de registro de múltiplos dispositivos incluindo análise comparativa dos dados entre os mesmos. A substituição do módulo de envio de e-mails por um módulo de envio de mensagem SMS pode ser uma

alternativa para prover acesso rápido ao administrador. É possível utilizar ambos em conjunto e este também é um dos objetivos de estudos futuros do presente trabalho.

Outra possibilidade tangível para trabalho futuro, como aprimoramento do pre- sente projeto, seria a incorporação de um sensor de abertura da porta, a fim de registrar em disco sua periodicidade. Um *beep* sonoro poderia ser acionado caso a porta ficasse aberta mais tempo que o tempo t limite definido. Uma interessante possibilidade seria a integração com os servidores e demais equipamentos de rede do *datacenter*, onde, durante o procedimento de disparo do alerta, também fosse enviado um sinal de rede do tipo ”shutdown”, tornando-se, assim, um dispositivo ativo na ação de contingência dentro do *datacenter* em lugar do próprio administrador, que estaria restrito a apenas um aviso da ocorrência. Da mesma forma, atuando como sujeito passivo no processo de alerta, poderia esse acionar uma variável V que seria consultada pelos equipamentos de rede a cada tempo T e iniciariam de forma autônoma o processo de desligamento de acordo com o valor de V.

REFERÊNCIAS

APACHE. **Apache HTTP Server Project**, 2018. Disponível em: <https://httpd.apache.org/>. Acesso em 19 de Janeiro de 2018.

ARMBIAN. **Linux for ARM development boards**, 2018. Disponível em: <https://www.armbian.com>. Acesso em 19 de Janeiro de 2018.

BLUM, J. **Exploring Arduino: tools and techniques for engineering wizardry**. [S.l.]: John Wiley & Sons, 2013.

GRAZIOLI, G. **I2C LCD2004**, 2018. Disponível em: <https://github.com/wargio/liblcm1602>. Acesso em 25 de Janeiro de 2018.

GUBBI, J. et al. **Internet of things (iot): A vision, architectural elements, and future directions. Future generation computer systems**, Elsevier, v. 29, n. 7, p. 1645–1660, 2013.

LI, B. et al. **Design of electronic compass**. In: ATLANTIS PRESS. 2016 6th International Conference on Machinery, Materials, Environment, Biotechnology and Computer. [S.l.], 2016. Disponível em <https://download.atlantis-press.com/article/25858816.pdf>. Acesso em 20 de Janeiro de 2018.

MANTECH_ELECTRONICS. **I2C interface for LCD**, 2017. Disponível em: <http://www.mantech.co.za/datasheets/products/lcd2004-i2c.pdf>. Acesso em 25 de Janeiro de 2018.

MOUSER_ELECTRONICS. **DHT11 Humidity & Temperature Sensor**, 2018. Disponível em: <https://www.mouser.com/ds/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf>. Acesso em 23 de Janeiro de 2018.

NEVES, J. C. **Programação Shell Linux**, 7a edição. [S.l.]: Brasport, 2008.

ORANGE_PI. **New generation mini pc**, 2016. Disponível em: <http://www.orangepi.org/>. Acesso em 19 de Janeiro de 2018.

PHPMAILER. **The classic email sending library for PHP**, 2018. Disponível em: <https://github.com/PHPMailer>. Acesso em 19 de Janeiro de 2018.

PHP.NET. **PHP: Hypertext Preprocessor**, 2018. Disponível em: <http://php.net/>. Acesso em 19 de Janeiro de 2018.

SCOFIELD, Mike C.; WEAVER, Thomas S. **Data center cooling: using wet-bulb economizers**. ASHRAE Journal, v. 50, n. 8, p. 52-58, 2008.

STALLINGS, W. **SNMP, SNMPv2, SNMPv3, and RMON 1 and 2**. [S.l.]: Addison-Wesley Long-man Publishing Co., Inc., 1998.

SUNFOUNDER. **I2C LCD2004**, 2018. Disponível em: <http://wiki.sunfounder.cc/index.php?title=I2C_LCD2004>. Acesso em 25 de Janeiro de 2018.

TIANLONG, N. **Application of single bus sensor dht11 in temperature humidity measure andcontrol system**. Microcontrollers & Embedded Systems, v. 6, p. 026, 2010.

UUGEAR. **DHT11 Humidity & Temperature Sensor Module**, 2018. Disponível em: <http://www.uugear.com/portfolio/dht11-humidity-temperature-sensor-module/>. Acesso em 28 de Janeiro de 2018.

WETHERALL, J.; TANENBAUM, A. **Redes de computadores**. 5ª edição. Rio de Janeiro: Editora Campus, 2011.

WILSON, Matthew D. **VPN HOWTO**, 1999.

WIRING_PI. **Wiring Pi**, 2018. Disponível em: <http://www.orangepi.org/Docs/WiringPi.html>. Acesso em 25 de Janeiro de 2018.

WOODS, A. **Cooling the data center**. Communications of the ACM, ACM, v. 53, n. 4, p. 36–42, 2010.

YLONEN, Tatu et al. **The secure shell (SSH) protocol architecture**, 2006. Disponível em: <https://tools.ietf.org/html/rfc4251>. Acesso em 25 de Janeiro de 2018.

Apêndice A: Código-fonte sensor DHT11 (UUGEAR, 2018)

```
#include <wiringPi.h>
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>

#define MAX_TIMINGS 85
#define DHT_PIN 7
int data[5] = { 0, 0, 0, 0, 0 };

void read_dht_data()
{
   uint8_t laststate    = HIGH;
   uint8_t counter      = 0;
   uint8_t j            = 0, i;

   data[0] = data[1] = data[2] = data[3] = data[4] = 0;

   /* pull pin down for 18 milliseconds */
   pinMode( DHT_PIN, OUTPUT );
   digitalWrite( DHT_PIN, LOW );
   delay( 18 );

   /* prepare to read the pin */
   pinMode( DHT_PIN, INPUT );

   /* detect change and read data */
   for ( i = 0; i < MAX_TIMINGS; i++ ){
      counter = 0;
      while ( digitalRead( DHT_PIN ) == laststate ){
         counter++;
         delayMicroseconds( 1 );
         if ( counter == 255 ){
            break;
         }
      }
      laststate = digitalRead( DHT_PIN );

      if ( counter == 255 )
         break;

      /* ignore first 3 transitions */
      if ( (i >= 4) && (i % 2 == 0) ){
         /* shove each bit into the storage bytes */
         data[j / 8] <<= 1;
         if ( counter > 16 )
            data[j / 8] |= 1;
         j++;
      }
   }

   /*
    * check we read 40 bits (8bit x 5 ) + verify checksum in the last byte
    * print it out if data is good
    */

   if ( (j >= 40) &&
        (data[4] == ( (data[0] + data[1] + data[2] + data[3]) & 0xFF) ) ){
      float h = (float)((data[0] << 8) + data[1]) / 10;
      if ( h > 100 ){
         h = data[0];   // for DHT11
      }
      float c = (float)(((data[2] & 0x7F) << 8) + data[3]) / 10;
      if ( c > 125 ){
         c = data[2];   // for DHT11
      }
      if ( data[2] & 0x80 ){
         c = -c;
      }
      float f = c * 1.8f + 32;
      printf( "Humidity = %.1f %% Temperature = %.1f *C (%.1f *F)\n", h, c, f );
   }
   else {
     printf( "Data not good, skip\n" );
   }
}

int main( void ){
   printf( "DHT11 temperature/humidity\n" );
```

Apêndice B: Código-fonte Display LCD (GRAZIOLI, 2018)

```
#include <stdio.h>
#include <string.h>
#include "i2c.h"
#include "lcd.h"

#define I2C_FILE_NAME "/dev/i2c-0"
int main(int argc, char **argv){
  int i2c_dev, i;
  lcd lcd0;
  i2c_dev = open_i2c(I2C_FILE_NAME, 0x3f);

  if(i2c_dev <0){
    printf("Error: %d\n", i2c_dev);
    return 1;
  }

  lcd_init(&lcd0, i2c_dev);
  lcd_clear(&lcd0);

  for(i=1; i<3 && i<argc; ++i)
     lcd_print(&lcd0, argv[i], strlen(argv[i]), i-1);

  close_i2c(i2c_dev);
  return 0;
}
```

INTERFACE TECNOLÓGICA

Apêndice C: Código-fonte do Algoritmo Principal (Os Autores)

```
#!/bin/bash
# Joles - Sensor Inteligente

n=0
c=0;

while :
do
  # Inicializa variaveis de controle
  date=`date +"%Y/%m/%d %H:%M:%S"`;
  dia=`echo $date | cut -d' ' -f 1 | cut -d'/' -f3`;
  mes=`echo $date | cut -d' ' -f 1 | cut -d'/' -f2`;
  hora=`echo $date | cut -d' ' -f 2 | cut -d':' -f1`;
  minuto=`echo $date | cut -d' ' -f 2 | cut -d':' -f2`;
  segundo=`echo $date | cut -d' ' -f 2 | cut -d':' -f3`;

  # captura dados de temperatura e umidade
  dht=`/root/dht | tail -1`;
  if [[ $dht == *"Temp"* ]]; then
    hum=`echo $dht | awk '{print $3}' | cut -d. -f1`;
    temp=`echo $dht | awk '{print $7}' | cut -d. -f1`;
    echo $hum,$temp > /dht.tmp;
  else
    hum=`cat /dht.tmp | cut -d',' -f1`
    temp=`cat /dht.tmp | cut -d',' -f2`
  fi

  # Envia aviso em caso de alerta
  if [ $temp -gt 30 ]; then
    php /var/www/html/Joles/mail.php "Temperatura alta: ${temp}C";
  elif [ $hum -gt 90 ]; then
    php /var/www/html/Joles/mail.php "Humidade alta: ${hum}%";
  fi

  # atualiza o display
  linha1="Data Center  ${temp}C";
  linha2="$dia/$mes $hora:$minuto  ${hum}%";
  sleep 1;
  /root/lcd "${linha1}" "${linha2}"
  echo "${linha1}" "${linha2}"

  # atualiza o registro em disco
  if [ $n -gt 9 ]; then
     /root/lcd 1 1
     echo "$temp,$hum,$date" >> /dht.log
     n=0
  fi

  n=$(($n+1))
  echo $n
  sleep 60

  c=$(($c+1))
done
```

INTERFACE TECNOLÓGICA

Apêndice D: Código-fonte apresentação dos registros via URL (Os Autores)

```
<html>
  <head>
    <meta charset="UTF-8">
    <title>Joles Smart Sensor - IFSP Barretos</title>

    <link rel="stylesheet" href="css/bootstrap.min.css">
    <link rel="stylesheet" href="css/dataTables.bootstrap.min.css">
    <link rel="stylesheet" href="css/buttons.dataTables.min.css">

    <script src="js/jquery.min.js"></script>
    <script src="js/bootstrap.min.js"></script>
    <script src="js/jquery.dataTables.min.js"></script>
    <script src="js/dataTables.bootstrap.min.js"></script>
    <script src="js/dataTables.buttons.min.js"></script>
    <script src="js/buttons.print.min.js"></script>
    <script src="js/buttons.html5.min.js"></script>
    <script src="js/jszip.min.js"></script>

    <style>
      .dt-top2{margin-top: -35px;}
      .dt-bottom2{margin-top: -20px;}
      @font-face {font-family: lcd2;src: url(fonts/lcd2.woff);}
      .jumbotron {background: #0067ff;color: white;font-family: lcd2;font-weight: bold;}
      .dt-bottom2 a {color: red;}
      .panel {margin-top: -15px;}
    </style>
  </head>
  <body>
    <?php
    $file = "/dht.log";
    $dht = fopen($file, "r");
    while (!feof($dht)) {
      $array[] = fgets($dht);
    }

    $i = count($array);
    while ($i) {
      $linha = $array[--$i];
      if (!empty($linha)) {
        $dados = explode(",", $linha);
        $temp = $dados[0];
        $hum = $dados[1];
        $data = $dados[2];

        $lista .= "<tr class='item'><td>$data</td><td>" . $temp . "°C</td><td>$hum%</td></tr>";
      }
    }
    fclose($dht);
    ?>
    <div class="container" style="width: 95%">
      <div class="jumbotron" style="padding: 15px; text-align: center; margin-bottom: 20px; margin-top: 5px">
        <h1>Joles Smart Sensor</h1>
        <p>Nucleo de Tecnologia da Informacao - IFSP campus Barretos</p>
      </div>
      <div class="panel panel-primary">

        <div class="panel-body">
          <div class="table-panel">
            <table class="table table-striped table-bordered">
              <thead>
                <tr><th>Data</th><th>Temperatura</th><th>Humidade</th>
              </thead>
              <tbody>
                <?= $lista ?>
              </tbody>
            </table>
          </div></div></div></div>

    <script>
      $(document).ready(function () {
        $('.table').DataTable({
          dom: '<"dt-top1"Bl><"dt-top2"f>rt<"dt-bottom1"i><"dt-bottom2"p><"clear">',
          buttons: [
            'excelHtml5', 'print',
```