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

Aluno(a): Milson António

Orientador: Mauro Marcelo Mattos

FURB
UNIVERSIDADE DE BLUMENAU

Roteiro

- Introdução
- Objetivos
- Fundamentação Teórica
- Trabalhos Correlatos
- Requisitos
- Especificação
- Implementação
- Operacionalidade da Implementação
- Resultados e discussões
- Conclusão e sugestões
- Demonstração

Introdução

- O que é o **DC Alerta?**
- Surgimento
- Importância

Objetivos

- Desenvolver um sistema para o monitoramento da temperatura e umidade de data centers.

- a) disponibilizar uma interface para o monitoramento;
- b) implementar mecanismos de alerta;
- c) disponibilizar histórico de leitura do ambiente;
- d) validar as leituras através de um conjunto teste nos dados lidos;
- e) disponibilizar um App para o monitoramento via celular

Fundamentação Teórica

- Data Center
- Internet das Coisas (IoT)
- Arduino Mega
- Sensor de temperatura e umidade DHT11
- Modulo Ethernet ENC28J60
- ThingSpeak

- Data Center

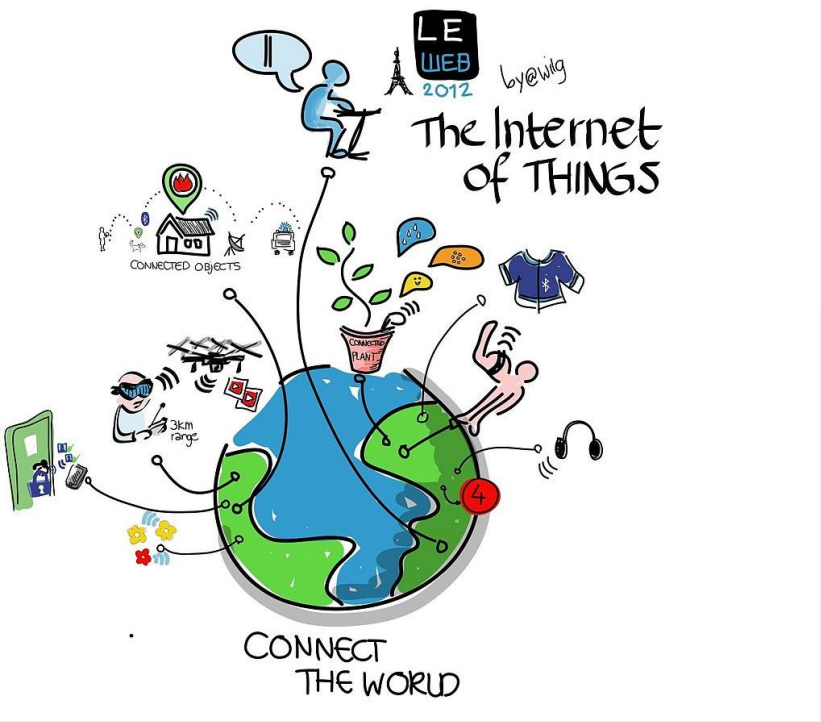
- Equipamentos eletrônicos utilizados para processamento de dados...
- Temperatura e umidade adequadas para os equipamentos.
- O cérebro de muitas companhias

- Internet das Coisa (IoT)

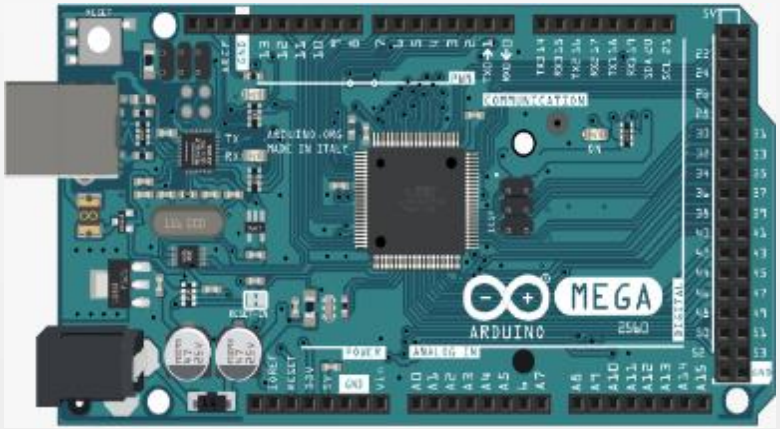
- Popularização da Internet e da possibilidade de interconectar objetos diversos.

- Categorias: “pessoas conectadas a pessoas, pessoas conectadas a coisas e coisas conectadas a outras coisas”.

- Coletar e enviar dados.

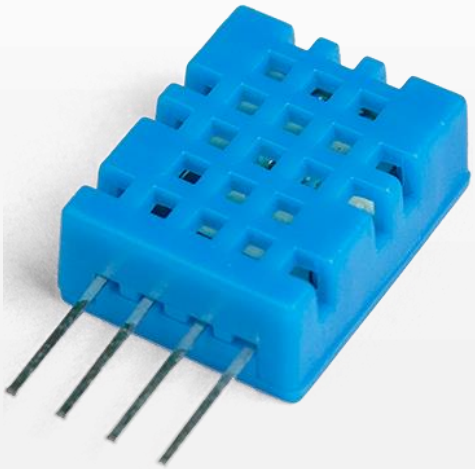


- Arduino
- Projetada para fornecer uma maneira barata e fácil para hobby, estudantes e profissionais.
- Interface amigável
- Linguagem Processing, baseada na linguagem C/C++
- Para projetos que exigem mais E/S.
- Arduino Mega1280 com 128 KB
- Arduino Mega2560 com 256 KB



Sensor de temperatura e umidade DHT11

- Alimentação 3 – 5.5 V
- Faixa de leitura da umidade - 20 – 80%
- Precisão da umidade - 5 %
- Faixa de leitura da temperatura - 0 – 50ºC
- Precisão da temperatura - +/- 2º
- Intervalo de medições – 1 segundo.

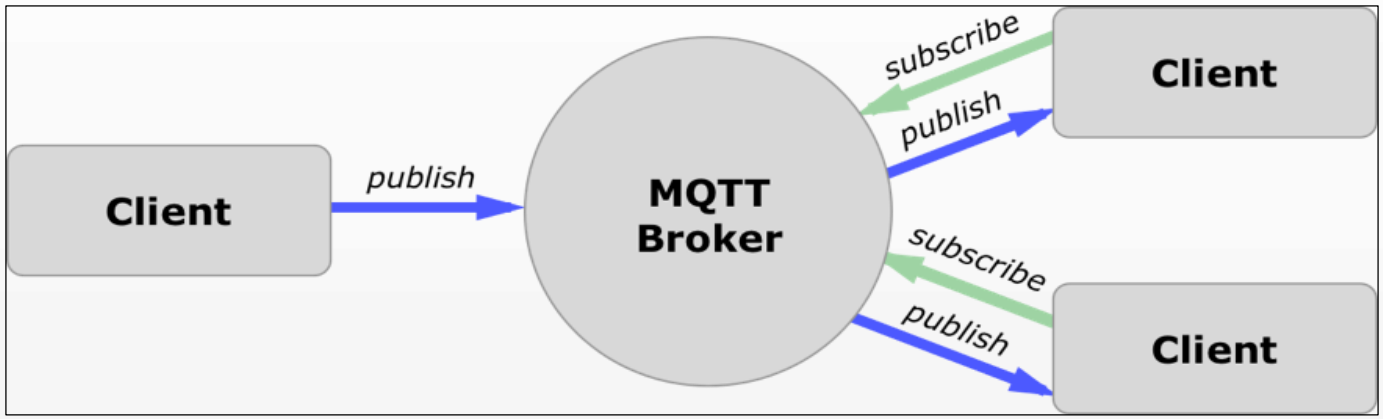


- Modulo Ethernet ENC28J60

- Projetado para servir como interface de uma rede Ethernet

- ThingSpeak

- Protocolos Http
- Protocolo MQTT
  - *Publisher, Broker, Subscriber*



- Comunicação é feita por intermédio de canais
- Chave de Leitura
- Chave de Escrita

Trabalhos Correlatos

- Monitoramento Ambiental Open Source para Data Center – Camargo (2015)
- Controle de Temperatura para Automação residencial utilizando Modelos Ocultos de Markov – Lazo (2014)
- Estudo de caso: Sistema para Monitoramento de Temperatura e Umidade em Farmácias e Almoxarifados – Alves (2014)

Monitoramento Ambiental Open Source para Data Center – Camargo (2015)

- Manter a sala de servidores dentro das normas técnicas aceitáveis para data center
- Monitora temperatura, umidade, pressão atmosférica e ponto de orvalho.
- Envio de comandos para aparelhos de ar-condicionado
- Banana Pi

Controle de Temperatura para Automação residencial utilizando Modelos Ocultos de Markov – Lazo (2014)

- Monitoramento da temperatura e automação residencial
- Implementa algoritmo de inteligência artificial, o HMM (Hidden Markov Models)
- Raspeberry Pi

Estudo de caso: Sistema para Monitoramento de Temperatura e Umidade em Farmácias e Almoxarifados – Alves (2014)

- Monitora farmácias e almoxarifados
- Monitora apenas a temperatura
- Termômetros, conversores e microcomputador (não especificado).

Requisitos funcionais

- Disponibilizar interface para monitoramento em tempo real
- Possibilitar a consulta de histórico
- Permitir que se faça ajuste das variáveis coletadas entre máxima e mínima
- Permitir gerência de usuários (nome, e-mail) para receberem alertas ou notificações
- Enviar alertas em caso de falha nos sensores
- Validar os dados dos sensores

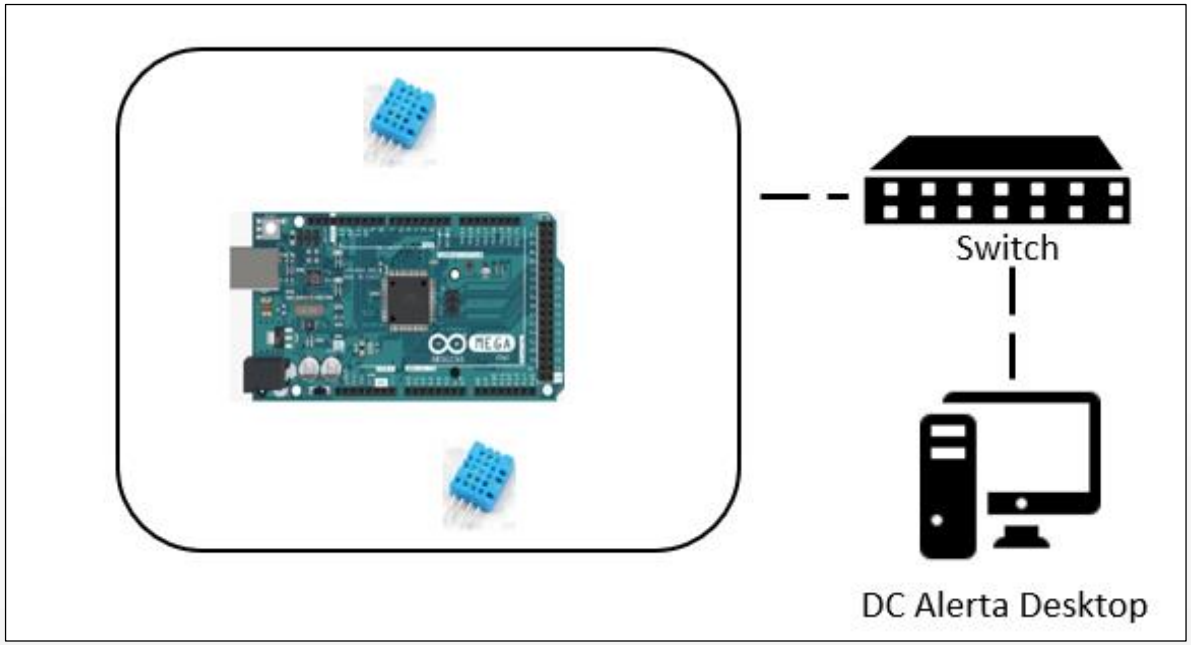
Requisitos não funcionais

- Hardware:
  - Arduino Mega
  - DHT11
  - Modulo Ethernet ENC28J60
- Software Desktop:
  - Sistema operacional Windows 10
  - Linguagem de programação C#
  - Banco de dados SQLite
- Software Mobile:
  - Sistema operacional Android
  - Linguagem de programação Java
  - Banco de dados SQLite

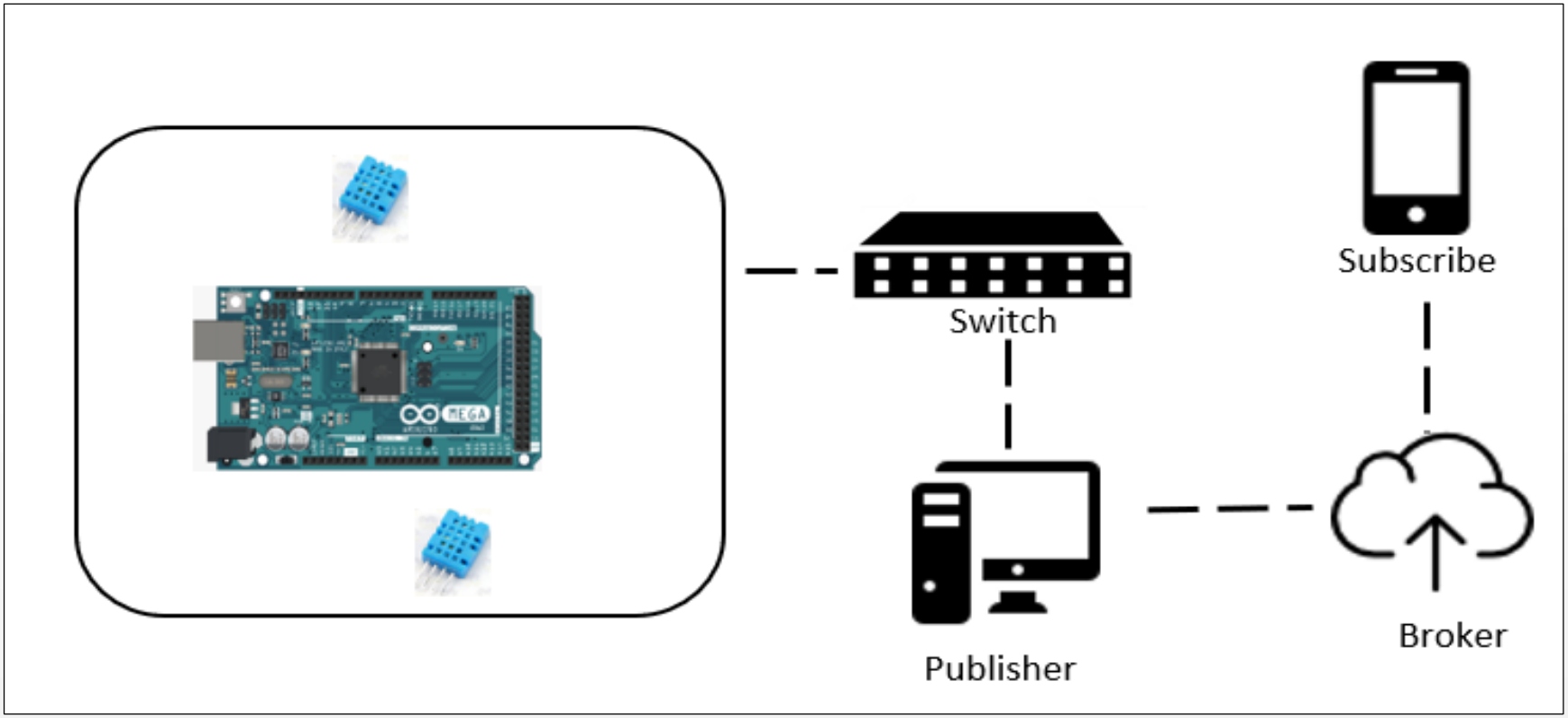
Especificação

- Diagrama de distribuição I
- Diagrama de distribuição II
- Diagrama de casos de uso
- Diagrama de atividade

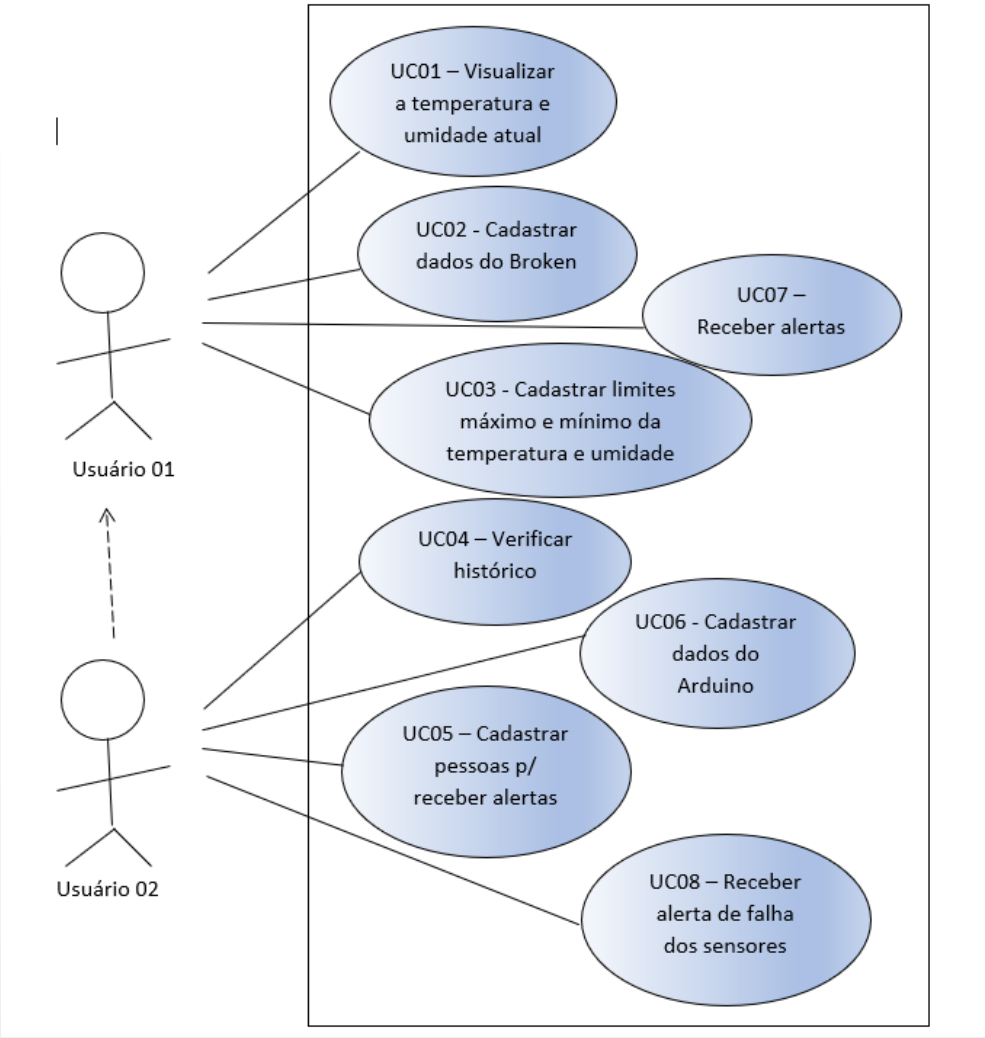
Descrição - Diagrama de distribuição I



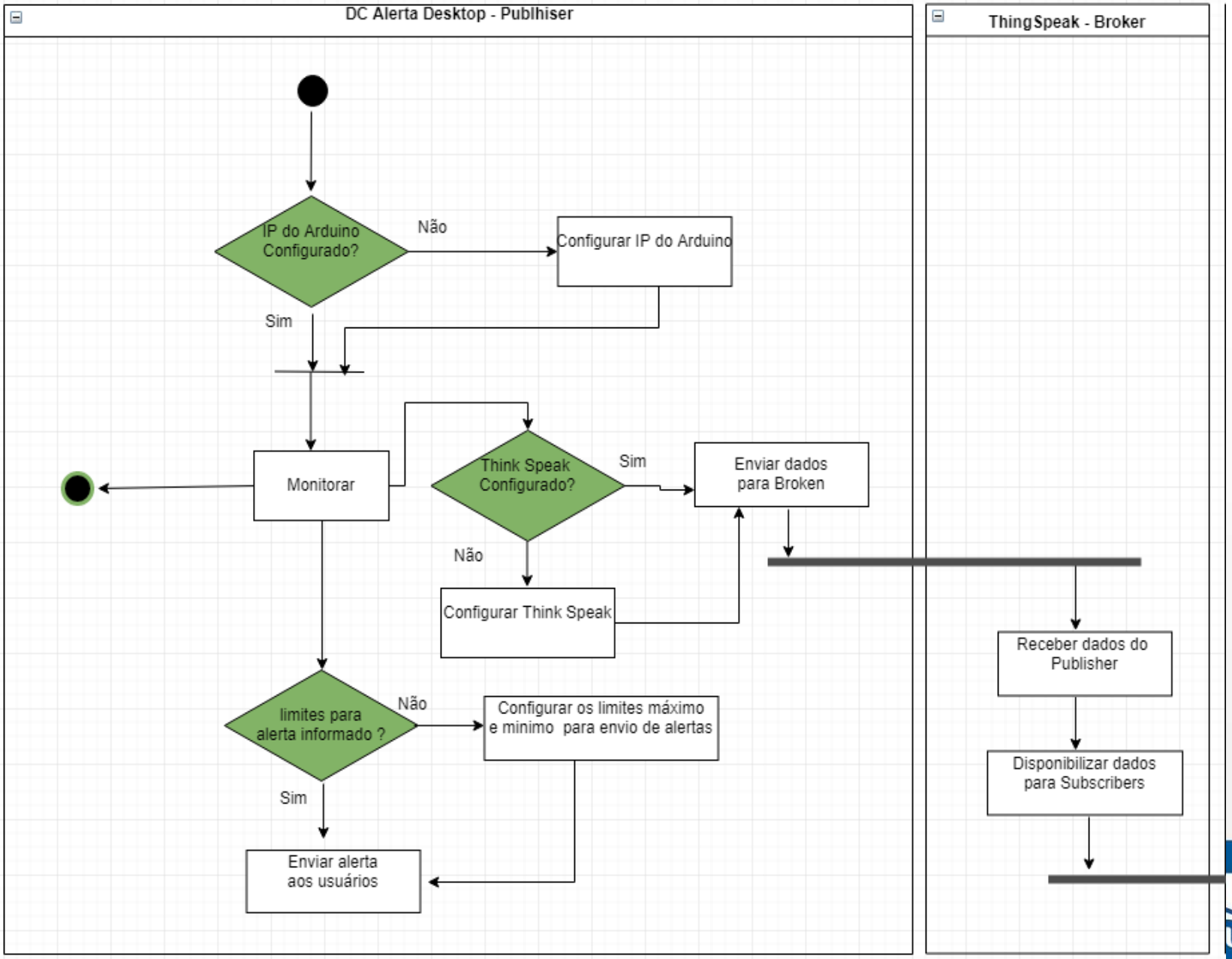
Descrição – Diagrama de distribuição II



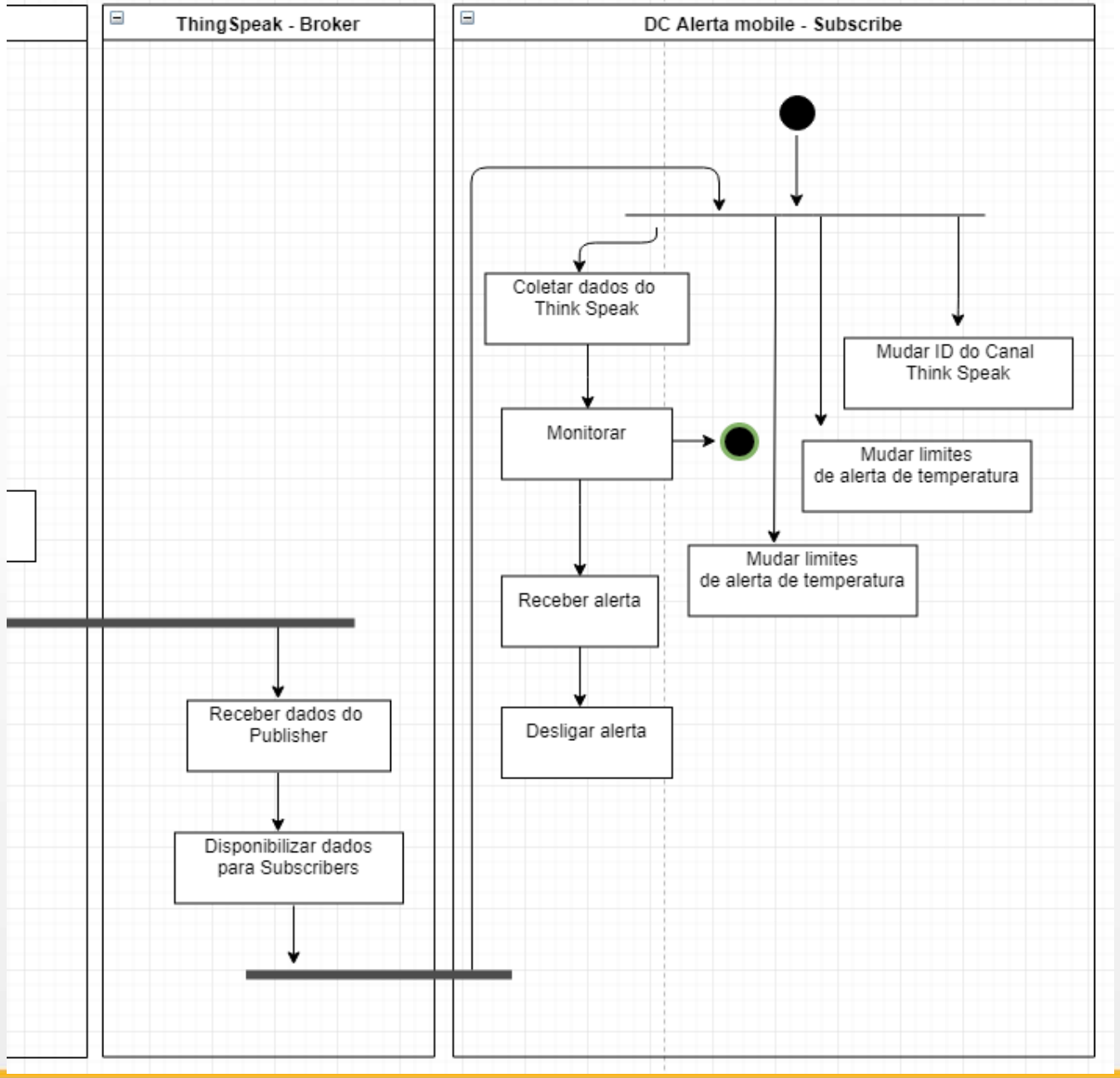
Descrição - Diagrama de casos de uso



Descrição - Diagrama de atividades



Descrição - Diagrama de atividades parte 2.

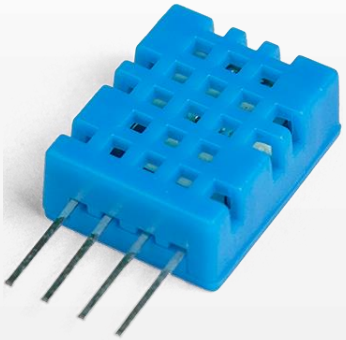


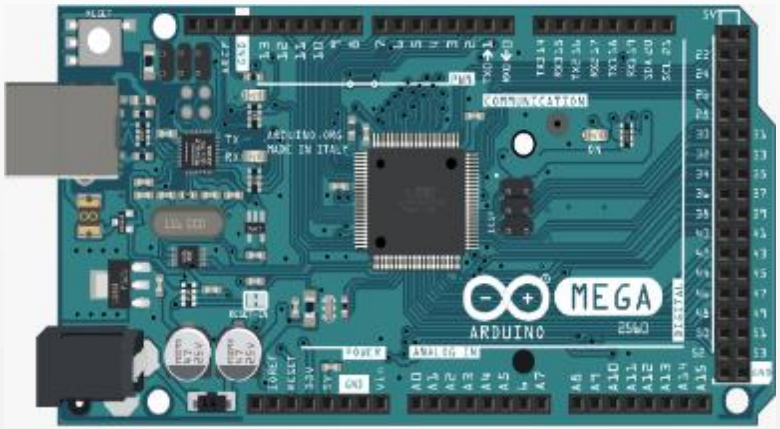
Implementação

- Componente de Hardware
- Componente de Software
  - DC ALERTA *desktop*
  - DC ALERTA *mobile*
- Verificação inicial
- Envio de dados para ThingSpeak
- Alertas (envio de e-mail)
- Validação de leitura dos sensores

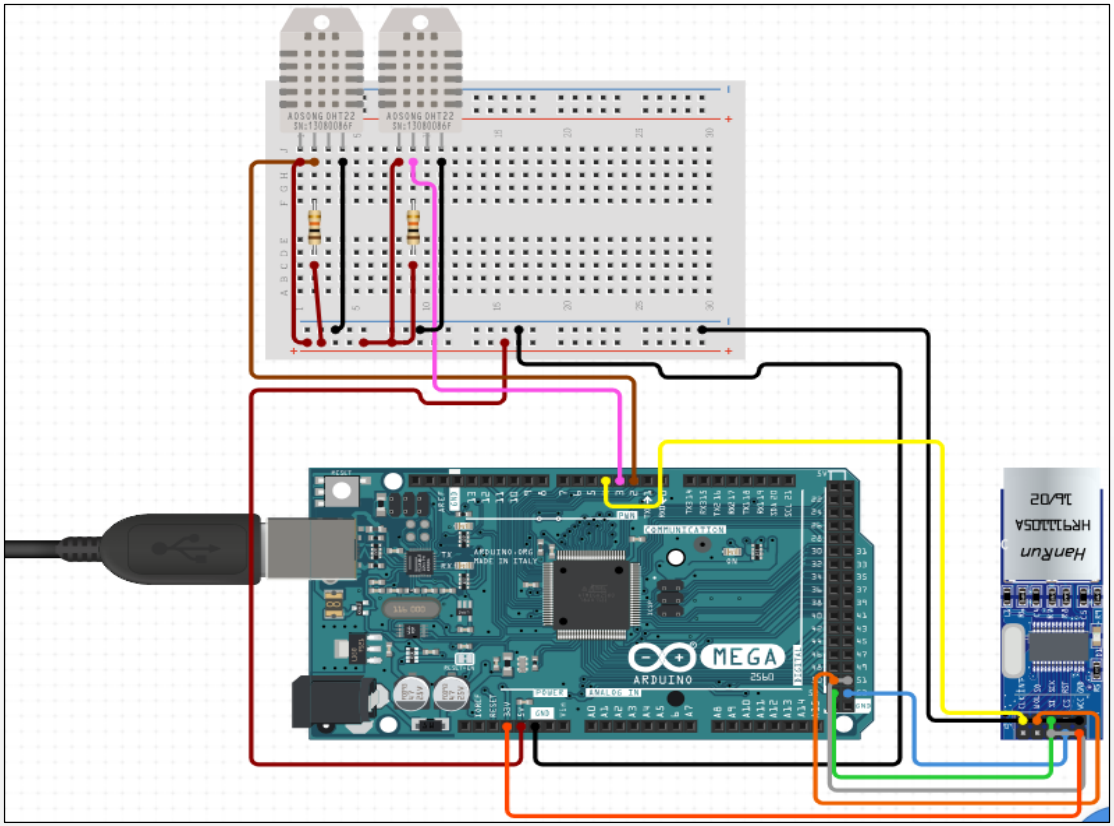
Componentes de Hardware

- Arduino Mega
- Sensor de temperatura e umidade DHT11
- Modulo Ethernet ENC28J60

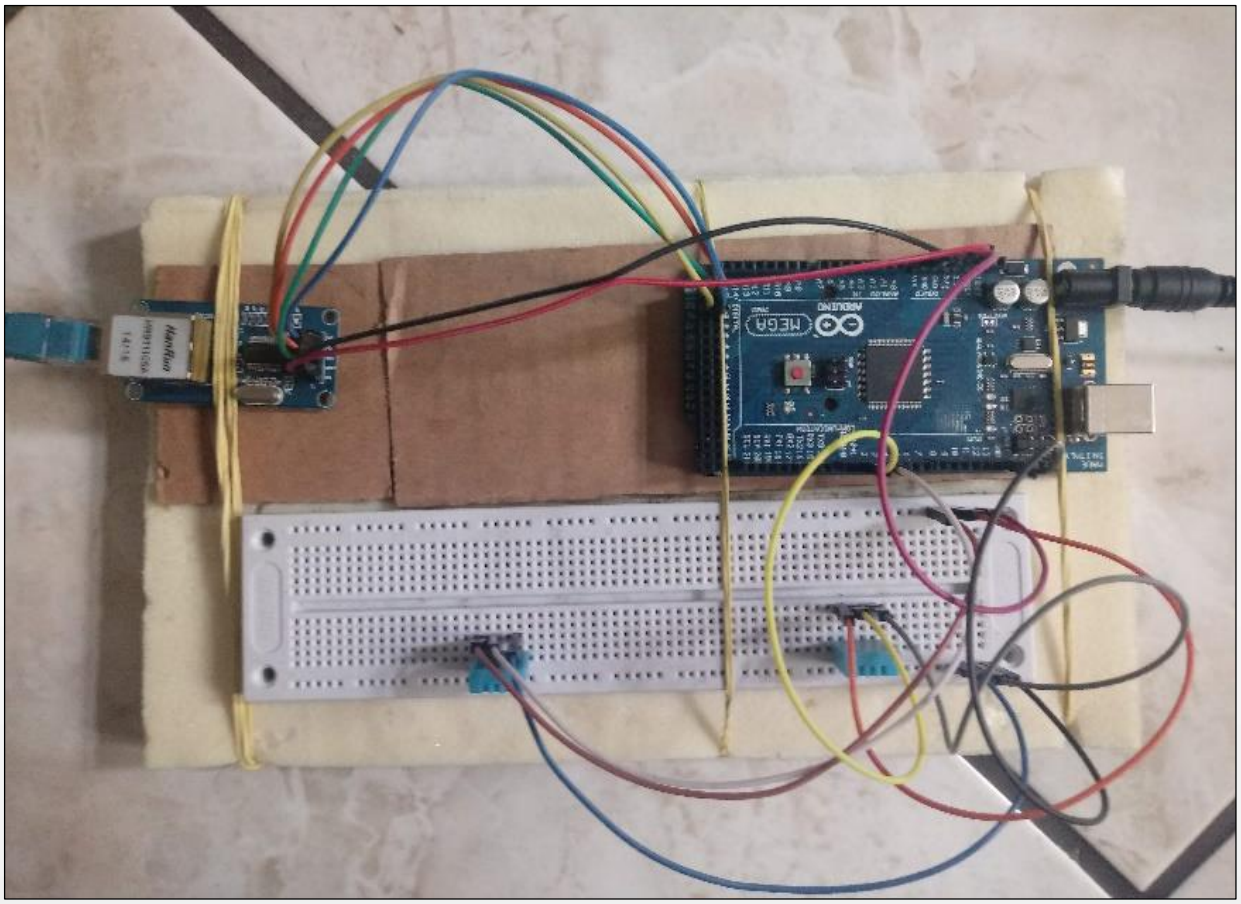




Componentes de Hardware

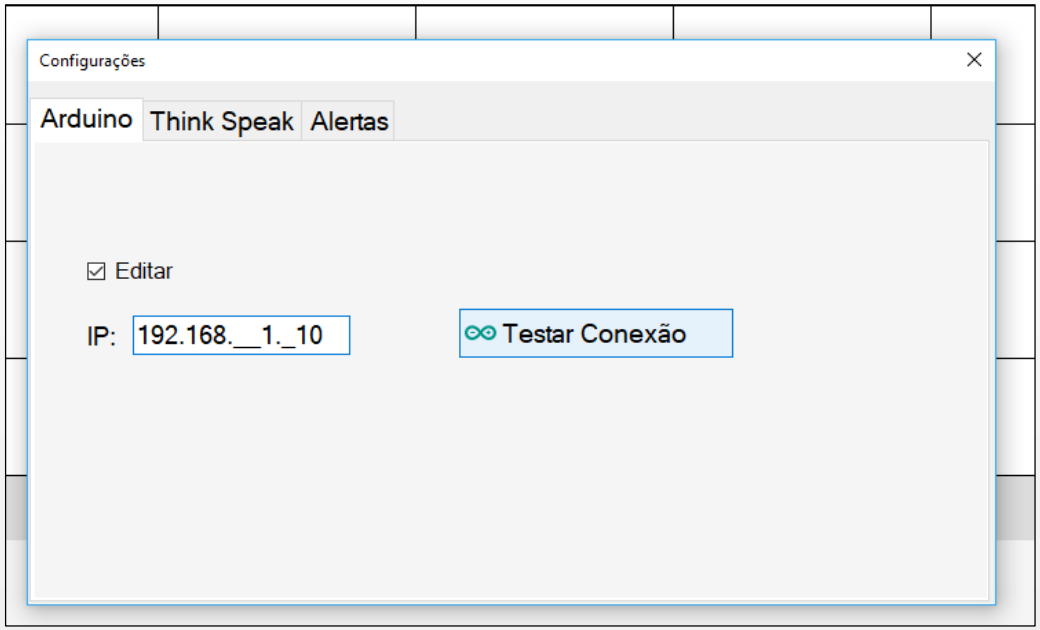


Componentes de Hardware



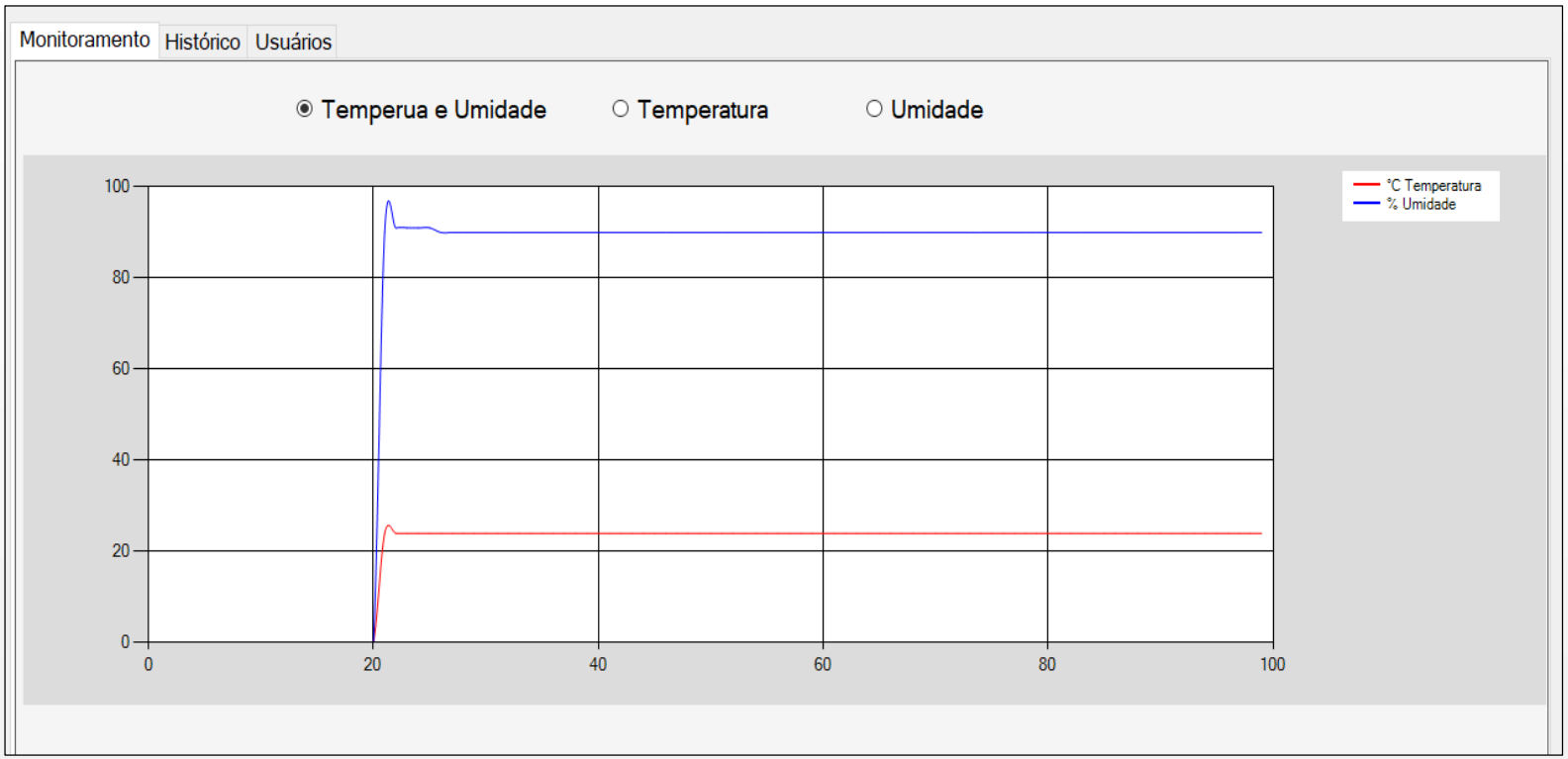
Componentes de Software - DC Alerta: *desktop*

- Conecta ao Arduino
- Recebe os dados dos sensores...
- Envia dados dos sensores para o ThingSpeak



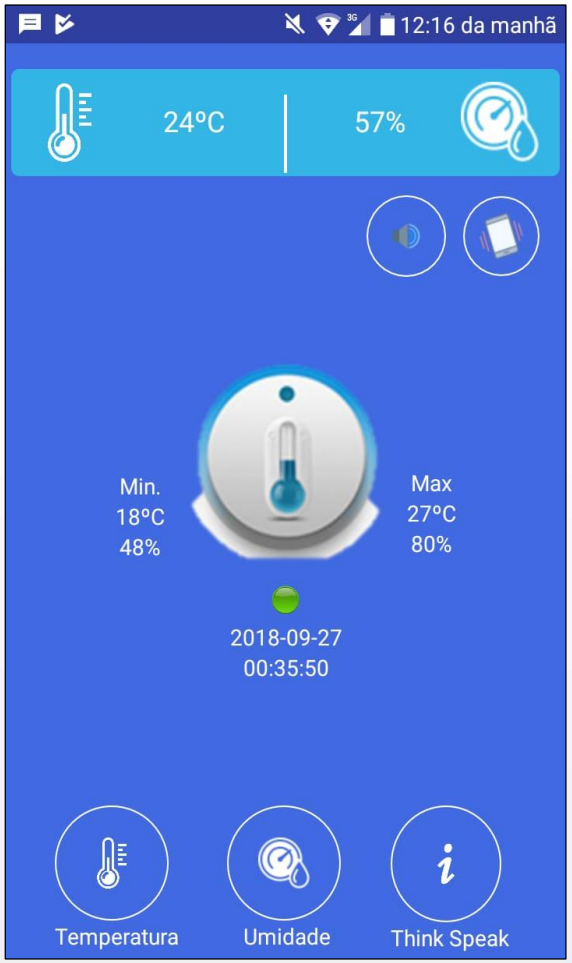
Componentes de Software - DC Alerta: *desktop*

- Gráfico de monitoramento



Componentes de Software - DC Alerta: *mobile*

- Consome os dados a partir do Broker ThingSpeak
- Configuração dos parâmetros do canal ThingSpeak
- Alerta: Vibração e campainha



Componentes de Software - DC Alerta: *mobile*

Verificação inicial

```
if (!banco.verificaSeBancoExiste())
{
    banco.CriarBancoSQLite();
}
tabelaHistoricoMonitoramento();
tabelaUsuario();

if (!banco.jaTemIp())
{
    Configuracoes conf = new Configuracoes(true, 0);
    conf.ShowDialog();
}

if (banco.SelecThinkSpeak().Count <= 0)
{
    Configuracoes conf = new Configuracoes(true, 1);
    conf.ShowDialog();
}
```

```
ipArduino = banco.SelecIp();
ArduinoEstaConetado = banco.pingar(ipArduino);
if (ArduinoEstaConetado)
{
    //Thinks Speak
    ativadoThinkSpeak = banco.testeThinkSpeak();
    if (ativadoThinkSpeak)
    {
        string WRITEKEY = banco.SelecThinkSpeak()[1];
        string strUpdateBase = "http://api.thingspeak.com/update";
        updateURI = strUpdateBase + "?key=" + WRITEKEY;
    }
    else
    {
        pictureBoxThinkSpeak.Image = Image.FromFile(@"C:\Temp\OneDri
    }
    MetodoThreadMonitoramento();
} else
{
```

Envio de dados para ThingSpeak

```
1 referencia
public void ThinkSpeak()  {
    try
    {
        string hora = retornaData_ou_Hora("hora").Replace(":", "");
        string data = retornaData_ou_Hora("data").Replace("/", "");
        updateURI += "&field1=" + temperatura.ToString();
        updateURI += "&field2=" + umidade.ToString();
        updateURI += "&field3=" + data;
        updateURI += "&field4=" + hora;
        HttpWebRequest ThingsSpeakReq;
        HttpWebResponse ThingsSpeakResp;
        ThingsSpeakReq = (HttpWebRequest)WebRequest.Create(updateURI);
        ThingsSpeakResp = (HttpWebResponse)ThingsSpeakReq.GetResponse();
        pictureBoxThinkSpeak.Image = Image.FromFile(@"C:\Temp\OneDrive - FURB\FURBR_2018_II\TCC\DCAlerta\DCAlerta\icones\nuvem.png");
        if (!(string.Equals(ThingsSpeakResp.StatusDescription, "OK")))
        {
            if (this.pictureBoxThinkSpeak.InvokeRequired)
            {
                delegateFalhaThinkSpeak d = new delegateFalhaThinkSpeak(falhaThinkSpeak);
                this.Invoke(d, new object[] { });
            }
        }
    }
    catch (Exception ex)
    {
        if (this.pictureBoxThinkSpeak.InvokeRequired)...
    }
}
```

Alertas (envio de 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)...
                }
                } catch
                {
                    boolIconeEnvioEmaiSucesso = false;
                    boolMostraIcone = true;
                    cronometro.Start();
                if (this.pictureBoxEvniaEmail.InvokeRequired)...
                }
        }
    }
}
```

Validação de leitura dos sensores

```
public void verificaSensores(int tempUm, int tempDois, int umidadeUm, int umidadeDois)
{
    bool problema = false;
    int difTemp = tempUm - tempDois;
    if (difTemp < 0)
    {
        difTemp = difTemp * -1;
    }
    if (difTemp > 5)
    {
        problema = true;
    }
    int difUmidade = umidadeUm - umidadeDois;
    if (difUmidade < 0)
    {
        difUmidade = difUmidade * -1;
    }
    if (difUmidade > 10)
    {
        problema = true;
    }
    if (problema)
    {
        EnviarEmailProblemaSensor(tempUm, tempDois, umidadeUm, umidadeDois);
        if (this.pictureBoxProblemanoSensor.InvokeRequired)
        {
            DelegateMostraIconeProblemaSensor d = new DelegateMostraIconeProblemaSensor(mostraIconeProblemaSensor);
            this.Invoke(d, new object[] { });
        }
    }
}
```

Resultados e Discussões

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

Conclusões

- O protótipo serviu como prova de conceito
- Durante os testes realizados em ambiente simulado, o DC Alerta fez o monitoramento da temperatura e umidade e enviou as alertas sempre que se fez necessário.
- 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.
- Viável e funcional

Sugestões

- Adicionar um modulo *wi-fi* ao circuito para que seja possível acessar os dados do sensor numa rede sem fio;
- Substituir os sensores DHT11 pelos DHT22, a fim de aumentar a capacidade, precisão e eficiência na leitura das variáveis ambientais;
- Adicionar ao circuito eletrônico módulos de infravermelhos ou outros para envio de comandos para os parelhos de ar condicionado;
- Implementar o protocolo MQTT diretamente no Arduino.

Demonstração