

UNIVERSIDADE REGIONAL INTEGRADA DO ALTO URUGUAI E DAS
MISSÕES CAMPUS DE ERECHIM
DEPARTAMENTO DE ENGENHARIAS E CIÊNCIA DA COMPUTAÇÃO

SAULO MATTÉ MADALOZZO

REDE DE ESTAÇÕES METEOROLÓGICAS AUTOMÁTICAS USANDO
ARDUINO

ERECHIM
2013

SAULO MATTÉ MADALOZZO

REDE DE ESTAÇÕES METEOROLÓGICAS AUTOMÁTICAS USANDO ARDUINO

Trabalho de conclusão de curso, apresentado ao curso de Ciência da Computação, Departamento de Engenharias e Ciência da Computação da Universidade Regional Integrada do Alto Uruguai e das Missões – Campus de Erechim.

Orientador:
Prof. Alexandre Magno dos Santos Adário

ERECHIM

2013

AGRADECIMENTOS

Agradeço primeiramente à todos que insistiram para que eu mudasse o tema deste trabalho. Às vezes não conseguimos ver algo que está em nossa frente e tomamos o caminho errado. De tanto vocês insistirem com isso, aqui está, um trabalho de conclusão sobre o que eu tinha concluído à tempo.

Quero agradecer à todos que me apoiaram e entenderam meus motivos quando não consegui entregar isto da primeira vez, agradeço também aos colegas de trabalho que não entenderam e me xingaram com razão.

Muito obrigado também ao meu orientador, Alexandre Magno dos Santos Adário, por me ajudar em dois trabalhos de conclusão, este que eu fiz e o outro que eu não fiz.

Agradeço também à minha família, ao meu pai e minha mãe pelo incentivo em todas as horas, aos meus irmãos por realmente me ajudarem e a reclamarem muito, mas muito mesmo, quando algo que fiz não está bom como deveria. Agradeço especialmente ao meu irmão Samuel, por ter sido solidário comigo, também não entregando o projeto final dele ano passado. Deve ser de família.

Agradeço aos meus amigos e a todos que de alguma forma me ajudaram e incentivaram este trabalho, do mesmo modo, agradeço à quem me ajudou no outro projeto que não deu certo, mesmo assim, serve de experiência e vale uma nova tentativa futura.

Quero agradecer também ao Aeroclube de Erechim, lá foi onde aprendi mais sobre meteorologia, foi onde também tive a oportunidade de aplicá-la.

E também à pessoa que sem saber incentivou uma ideia com uma simples frase: “De que me adianta uma informação muito precisa e cientificamente comprovada se a estação está à 70km daqui? Pode ser uma informação menos precisa mas daqui!”

EPÍGRAFE

*“There is no such thing as bad weather,
only different kinds of good weather.”*

John Ruskin, escritor Inglês

RESUMO

O clima exerce papel importante em diversas áreas da economia. Em um primeiro momento podemos pensar na sua importância para a agricultura, porém essa influência é muito presente em várias outras atividades, como por exemplo, no transporte aéreo. A captação de dados climáticos exige sensores diversos espalhados pelo território, formando uma rede de estações meteorológicas automáticas, centralizando em um servidor as informações sobre o tempo. As estações meteorológicas governamentais são calibradas e muito precisas, justamente por este motivo elas tem custo alto de implantação e manutenção, por outro lado isso garante dados confiáveis. Porém o custo alto é um impedimento para instalação de novas unidades, restringindo a coleta de dados às cidades maiores. Uma quantidade maior de estações, e consequentemente dados, cria uma granularidade menor, permitindo chegar a conclusões mais exatas na previsão do tempo e informações precisas do microclima a nível regional. Por isso, este trabalho propõe uma rede de estações meteorológicas automáticas, combinando um baixo custo com uma precisão razoável, com o intuito de preencher áreas entre as estações oficiais. Obtendo assim, uma granularidade menor com estações meteorológicas mais baratas nos intervalos geográficos das estações mais caras, eliminando as suposições por interpolação e gerando dados reais.

Palavras-chave: Estação meteorológica; Arduino, Microcontrolador, Internet of Things;

ABSTRACT

The weather plays an important role in many areas of the economy. At first we may think of its importance for agriculture, but this influence is very present in various other activities, such as aviation. Capturing weather data requires many sensors throughout the territory, forming a network of automatic weather stations, centralizing weather information on a server. The government weather stations are calibrated and very accurate, for this reason they have high cost of deployment and maintenance, on the other hand it ensures reliable data. But the high cost is an impediment to the installation of new units, restricting the data collection to larger cities. A larger number of stations, and consequently data, creates a smaller granularity, allowing to reach more accurate conclusions on the weather forecast and accurate microclimate regionally. This paper proposes a network of automatic weather stations, combining low cost with reasonable accuracy, in order to fill areas between official stations. Obtaining a lower granularity with cheaper weather stations in geographic ranges of the expensive stations, eliminating interpolation and generating real data.

Palavras-chave: Weather Station; Arduino, Microcontroller, Internet of Things;

LISTA DE FIGURAS

Figura 1: Dados do INMET da estação meteorológica de Erechim.....	18
Figura 2: Dados da Estação meteorológica da RBS Erechim.....	19
Figura 3: Weather Underground.....	21
Figura 4: IDE do Arduino.....	23
Figura 5: Arduino Uno.....	24
Figura 6: Alguns modelos de Arduino.....	25
Figura 7: Placas Arduino Ethernet.....	27
Figura 8: Sensor de temperatura TMP102.....	29
Figura 9: Sensor de umidade HIH-4030.....	30
Figura 10: Sensor de umidade e temperatura HIH-6130.....	31
Figura 11: Barômetro BMP085.....	32
Figura 12: Pluviômetro desmontado.....	33
Figura 13: Anemoscópio.....	36
Figura 14: Anemômetro.....	38
Figura 15: Gráfico de comparação entre pulsos por 10s e velocidade.....	39
Figura 16: Teste do anemômetro em túnel de vento.....	40
Figura 17: Sensor de Luminosidade TEMT600.....	41
Figura 18: Detalhe dos jumper wires e da protoboard.....	43
Figura 19: Montagem dos sensores.....	44
Figura 20: Peças antes da montagem final.....	45
Figura 21: Prototype Shield com todos os componentes soldados.....	46
Figura 22: Montagem dos componentes no lugar definitivo.....	47
Figura 23: Segundo protótipo finalizado e funcionando após 2 anos.....	48
Figura 24: Diagrama das conexões da Estação Meteorológica.....	49
Figura 25: Diagrama de funcionamento do firmware.....	50
Figura 26: Esquema de ligação do shield.....	51
Figura 27: Desenho do shield finalizado com todos os layers.....	52
Figura 28: Esquema da placa dos sensores.....	52
Figura 29: Desenho da placa dos sensores com todos os layers.....	53
Figura 30: Placa de circuito impresso pronta para a corrosão.....	53
Figura 31: Componentes e placas corroídas, furadas e cortadas.....	54
Figura 32: Shield durante a soldagem dos componentes.....	55

Figura 33: Sensor de umidade, temperatura e pressão com os conectores.....	55
Figura 34: Placa de apoio com o sensor de temperatura, umidade e pressão encaixados.....	56
Figura 35: Shield encaixado sobre o Arduino Ethernet.....	57
Figura 36: Teste do sensor HIH-6130 e BMP085.....	59
Figura 37: Diagrama de tráfego de dados.....	61
Figura 38: Configuração da aplicação no Jelastic.....	62
Figura 39: Diagrama de fluxo do back-end.....	64
Figura 40: Modelo ER.....	65
Figura 41: Interface com dados atuais e gráficos.....	67

LISTA DE QUADROS

Quadro 1: Código para leitura do sensor TMP102.....	29
Quadro 2: Funções usadas para leitura do pluviômetro.....	35
Quadro 3: Trecho de código da leitura do anemoscópio.....	37
Quadro 4: Código de exemplo para leitura do anemômetro.....	38

LISTA DE SIGLAS E ABREVIATURAS

PCB – Placa de circuito impresso, do inglês *Printed Circuit Board*

CI – Circuito integrado

DC – Corrente contínua, do inglês *Direct Current*

Pa - Pascais

hPa – Hectopascais

ft – pés, do inglês *feet*

SUMÁRIO

1 INTRODUÇÃO.....	14
2 ESTAÇÕES METEOROLÓGICAS.....	16
2.1 APLICAÇÕES.....	16
2.2 SISTEMAS EXISTENTES.....	17
2.2.1 INMET.....	17
2.2.2 Estações Meteorológicas Do Grupo RBS.....	18
2.2.3 Weather Underground.....	20
3 ARDUINO.....	22
3.1 PRINCIPAIS PLACAS.....	24
3.2 SHIELDS.....	26
3.3 ARDUINO ETHERNET.....	27
4 SENsoRES.....	28
4.1 TEMPERATURA.....	28
4.2 UMIDADE RELATIVA.....	30
4.3 PRESSÃO ATMOSFÉRICA.....	31
4.4 CHUVA.....	33
4.5 VENTO.....	36
4.6 RADIAÇÃO SOLAR.....	40
5 MONTAGEM.....	42
5.1 PROTOBOARD.....	42
5.2 PROTOTYPE SHIELD.....	45
5.3 SHIELD.....	49
6 SERVIDOR.....	60
6.1 ESTRUTURA DO SERVIDOR.....	61
6.1.1 Back-end e recebimento de dados.....	63
6.1.2 Banco de dados.....	65
6.1.3 Front-end.....	66

7 CONCLUSÃO.....	68
-------------------------	-----------

1 INTRODUÇÃO

Incrivelmente a meteorologia parece ser a única ciência que todas as pessoas tem interesse. Não é difícil ouvir um “será que chove?”. Isso talvez tenha uma explicação, todos somos afetados pelas mudanças do tempo, todos podemos ter problemas com uma tempestade forte ou um grande período de seca.

E há várias áreas onde o uso de estações meteorológicas é facilitador, podemos citar primeiramente a agricultura, onde podemos escolher os melhores dias para plantio, colheita e outros, com finalidade de aumentar a produtividade, aeródromos usam estações para coletar dados para fins de navegação aérea, sem contar a utilização dos dados para previsão do tempo e pesquisas do clima.

O objetivo deste projeto é criar uma base para uma rede de estações meteorológicas automáticas com Arduino. Utilizando hardware simples e sensores largamente disponíveis no mercado, buscando uma boa relação entre custo, precisão e confiabilidade.

O objetivo primário é desenvolver o hardware e o firmware, e disponibilizá-los sobre licenças que permitam qualquer tipo de uso, tanto sem fins lucrativos quanto comercial. Porém, em contrapartida, os dados gerados devem ser publicados e devem estar disponíveis também para qualquer tipo de uso.

Um segundo objetivo, é permitir que estas estações sejam instaladas principalmente onde não existem as oficiais, preenchendo ainda mais o território e gerando informações onde antes não havia nenhum dado. Assim, gerando dados reais e não médias por interpolação.

Para atingir este objetivo, um site receberá todos os dados das estações e permitirá a consulta e visualização dos mesmos com enfoque na exibição dos dados de uma maneira inovadora e interessante para os usuários.

Este trabalho foi dividido em sete capítulos, o primeiro é esta introdução.

O segundo capítulo contém detalhes sobre as aplicações das estações meteorológicas e uma breve explanação dos sistemas de redes de estações existentes.

O terceiro capítulo é sobre Arduino, que é a placa responsável por tratar os dados dos sensores e enviá-los a um servidor. Também é explicado um pouco da história, características das principais placas e um detalhamento maior do Arduino Ethernet.

O quarto capítulo tem o detalhamento de cada sensor utilizado, suas características, interfaces, exemplos de código e comparações com outros sensores do mesmo tipo.

No quinto capítulo é explicada a montagem do hardware com fotos e detalhes dos passos de cada protótipo construído. Também é explicado um pouco do firmware e da estrutura de funcionamento dele.

O sexto capítulo explica como foi feito o lado do servidor, a parte do recebimento de dados, do banco de dados e a exibição ao usuário final. É detalhado também, as tecnologias usadas para a construção da página e detalhes da plataforma de hospedagem.

O sétimo e último capítulo é a conclusão, também neste mesmo capítulo são apresentadas algumas ideias para implementações futuras.

2 ESTAÇÕES METEOROLÓGICAS

Estações meteorológicas são um conjunto de sensores analógicos ou digitais, que coletam dados sobre o tempo, automaticamente ou não, os mais importantes são: temperatura, umidade, pressão atmosférica, quantidade de chuva, direção do vento e velocidade do vento. Outros dados podem ser coletados, como por exemplo, qualidade do ar e salinidade da água, porém tudo depende da finalidade do estudo e da utilização deles.

2.1 APLICAÇÕES

As estações meteorológicas geram várias informações importantes em algumas áreas, as principais são a previsão do tempo e análise do clima. Para explicar a diferença entre ambos, segundo o Centro de Previsão de Tempo e Estudos Climáticos (CPTEC, 2013), tempo é o conjunto das condições atmosféricas e fenômenos meteorológicos que afetam um determinado local em um dado momento, já o clima é o comportamento estatístico das variáveis do tempo em um período longo, normalmente mais que 30 anos. O clima é estudado pela climatologia e o tempo pela meteorologia. (GERTZ, 2012 p. 55)

Seu uso na agricultura colabora com informações sobre a quantidade de água no solo, permitindo que se possa decidir os melhores momentos para plantio, aplicação de defensivos, necessidade de irrigação, colheita entre outros. A meteorologia aplicada à agricultura se chama agrometeorologia, e o CPTEC mantém um site somente para agregar dados aplicados à isso (METEOROLOGIA..., 2013).

Na aviação, os dados das estações são usadas para gerar informações para auxílio ao voo, entre elas estão o METAR (*METeorological Aerodrome Report*; Informe Meteorológico de Aeródromo) e o TAF (*Terminal Aerodrome Forecast*; Previsão Terminal de Aeródromo). Ambos são um conjunto de códigos alfanuméricos que podem ser interpretados por pilotos. O METAR é a situação atual do tempo e TAF é a previsão para 12 ou 24 horas (REDEMET, 2013) (SONNEMAKER, 2011 p. 126 e 136).

2.2 SISTEMAS EXISTENTES

Existem muitas estações meteorológicas espalhadas no território brasileiro, algumas mantidas por entidades que precisam destes dados, como aeroportos e fazendas por exemplo, outras pertencem à pessoas comuns com interesse nesta área, ainda outras pertencem à grupos, organizações ou governos. Excluindo-se as estações do INMET, Weather Underground e outras redes, poucas tem seus dados disponibilizados na internet.

2.2.1 INMET

Atualmente o governo Brasileiro possui uma rede de estações meteorológicas mantidas pelo INMET – Instituto Nacional de Meteorologia. Possui aproximadamente 400 estações automáticas e outras 300 comuns (com observadores), as automáticas coletam os dados e enviam à central a cada 1h, a conexão normalmente é por rede celular ou por satélite nos locais mais remotos (INMET, 2011).

Conforme o próprio INMET somente esta conexão custa pelo menos R\$250,00 cada ponto, se for satélite o valor passa a ser de R\$700,00. Soma-se a isto os custos de manter os equipamentos, as equipes de manutenção e a calibração dos sensores. (INMET, 2011)

Mas não podemos esquecer que o sistema como um todo é muito preciso e confiável, e é isso que garante a qualidade das informações. Os dados coletados por estas estações são utilizados para previsão do tempo no Brasil e também por outros órgãos no exterior, como a NOAA (*National Oceanic and Atmospheric Administration*), nos Estados Unidos. Estas informações também estão disponíveis na internet, no formato que pode ser visualizado na Figura 1 e também em forma de gráficos, porém neste caso somente de umidade e temperatura e outro gráfico com a precipitação mensal.

Consulta Dados da Estação Automática: ERECHIM (RS) [Fechar](#)

Observação: Estes são dados brutos e sem consistência com o único objetivo de deixá-los disponíveis de forma imediata.
Uma nova versão apresentará os dados depois de verificação de consistência.

Data Inicial: 24/09/2013				Data Final: 24/09/2013				Nova Pesquisa			Download de Dados							
Data	Hora	Temperatura (°C)			Umidade (%)			Pto. Orvalho (°C)			Pressão (hPa)			Vento (m/s)		Radiação	Chuva	
	UTC	Inst.	Máx.	Mín.	Inst.	Máx.	Mín.	Inst.	Máx.	Mín.	Inst.	Máx.	Mín.	Vel.	Dir.	(kJ/m²)	(mm)	
24/09/2013	00	10.9	11.7	10.9	93	93	90	9.7	10.1	9.7	931.5	931.7	931.0	1.9	223º	5.7	-3.28	0.0
24/09/2013	01	10.5	10.9	10.4	93	94	93	9.4	9.7	9.4	931.8	931.8	931.5	2.3	245º	5.7	-3.21	0.0
24/09/2013	02	10.1	10.5	10.1	91	93	91	8.8	9.4	8.8	931.9	931.9	931.7	2.6	232º	6.3	-2.12	0.0
24/09/2013	03	9.6	10.1	9.5	95	95	91	8.8	8.8	8.6	931.5	931.9	931.5	2.7	248º	5.5	-3.20	0.0
24/09/2013	04	8.3	9.7	8.2	96	96	95	7.6	8.9	7.6	931.5	931.7	931.5	3.8	239º	8.6	-1.71	0.0
24/09/2013	05	7.1	8.3	7.1	96	97	96	6.5	7.6	6.5	931.7	931.8	931.5	4.2	226º	9.1	-0.98	0.2
24/09/2013	06	6.5	7.1	6.5	97	97	96	6.0	6.6	6.0	931.5	931.9	931.5	3.3	220º	9.1	-1.63	0.2
24/09/2013	07	6.2	6.5	6.0	97	97	97	5.8	6.1	5.7	930.9	931.5	930.9	3.4	225º	7.7	-0.77	0.2
24/09/2013	08	5.4	6.2	5.4	96	97	96	4.9	5.8	4.9	931.6	931.6	930.9	3.7	247º	9.6	-1.10	0.2
24/09/2013	09	5.1	5.5	5.1	97	97	96	4.7	4.9	4.7	932.1	932.1	931.5	3.4	245º	8.2	-1.32	0.0
24/09/2013	10	4.9	5.1	4.9	97	97	97	4.5	4.7	4.4	932.6	932.6	932.1	2.6	259º	8.2	37.15	0.4
24/09/2013	11	5.3	5.7	4.9	96	97	95	4.7	5.2	4.5	933.0	933.1	932.6	2.4	240º	6.6	489.2	0.2
24/09/2013	12	5.3	5.5	4.9	92	96	92	4.1	4.8	4.1	933.1	933.2	933.0	3.6	252º	6.3	592.1	0.2
24/09/2013	13	6.0	6.3	5.3	81	93	78	3.0	4.9	2.5	933.4	933.4	933.1	4.5	261º	8.7	1279.	0.0
24/09/2013	14	7.7	7.7	5.9	68	82	65	2.2	3.9	1.1	933.2	933.4	933.2	4.5	254º	9.0	1983.	0.0
24/09/2013	15	7.6	8.9	6.8	67	76	63	1.8	4.2	1.3	933.2	933.3	933.1	4.6	244º	8.8	2004.	0.0
24/09/2013	16	8.2	9.2	7.4	64	70	61	1.8	3.5	1.2	932.7	933.3	932.7	5.1	256º	10.7	2159.	0.0
24/09/2013	17	7.3	9.1	7.3	76	76	60	3.4	4.0	1.4	932.7	932.7	932.4	3.4	228º	10.9	1474.	0.0
24/09/2013	18	8.5	9.8	6.7	72	83	64	3.7	5.3	2.1	932.4	932.8	932.4	2.5	233º	11.4	1986.	0.2
24/09/2013	19	9.1	10.2	8.3	62	72	58	2.3	4.4	1.4	932.3	932.4	932.2	3.8	238º	10.2	1422.	0.0
24/09/2013	20	6.7	9.3	6.6	83	83	62	4.1	4.1	2.3	933.0	933.0	932.3	2.8	222º	10.2	498.8	0.0
24/09/2013	21	6.0	6.8	6.0	87	87	81	3.9	4.1	3.6	933.4	933.4	933.0	3.0	230º	8.7	172.0	0.0
24/09/2013	22	5.6	6.0	5.4	92	92	87	4.4	4.4	3.8	933.7	933.7	933.4	2.3	234º	8.1	9.885	0.0
24/09/2013	23	5.3	5.6	5.3	95	95	92	4.6	4.7	4.3	933.9	933.9	933.7	2.5	222º	5.9	-2.26	0.0

Figura 1: Dados do INMET da estação meteorológica de Erechim

Fonte: INMET, 2013

2.2.2 Estações Meteorológicas Do Grupo RBS

No Rio Grande do Sul e em Santa Catarina também existe uma rede de estações mantidas pelo grupo RBS. São estações meteorológicas da marca Davis, usadas largamente pelo mundo todo. A RBS mantém as estações com o objetivo de informar dados em tempo real no seu site e também como referência para noticiar eventos climáticos (RBS, 2013).

Erechim

Estação Meteorológica atualizada em 26/09/2013 23:35. Dados acumulados a partir das oh.



Figura 2: Dados da Estação meteorológica da RBS Erechim

Fonte: RBS, 2013

Seus dados podem ser consultados no site da própria emissora, os dados são apresentados em um formato mais intuitivo que o INMET, contando com gráficos diários de temperatura e velocidade do vento, máximas e mínimas de umidade, pressão e temperatura, como pode ser visto na Figura 2.

2.2.3 Weather Underground

Outra rede de estações meteorológicas, em nível mundial, é a WeatherUndergound, a qual centraliza informações de estações mantidas por entusiastas e também dados disponibilizados por entidades governamentais. A grande maioria delas são de estações disponíveis comercialmente, pessoas às compram, instalam, conectam a um computador e passam a alimentar o site com as informações voluntariamente. (WEATHER..., 2013)

No site, os dados são disponibilizados de várias formas, a mais interessante delas é o *Wundermap*, que pode ser visto na Figura 3, que reúne os dados dispostos em um mapa interativo, com as temperaturas (bolinhas), direção do vento (linha reta saindo da bolinha) e velocidade do vento (linha transversal). Ao clicar em cada estação, um quadro com as informações detalhadas é exibido, na Figura 3 vemos os dados de uma estação na cidade de Uberlândia.

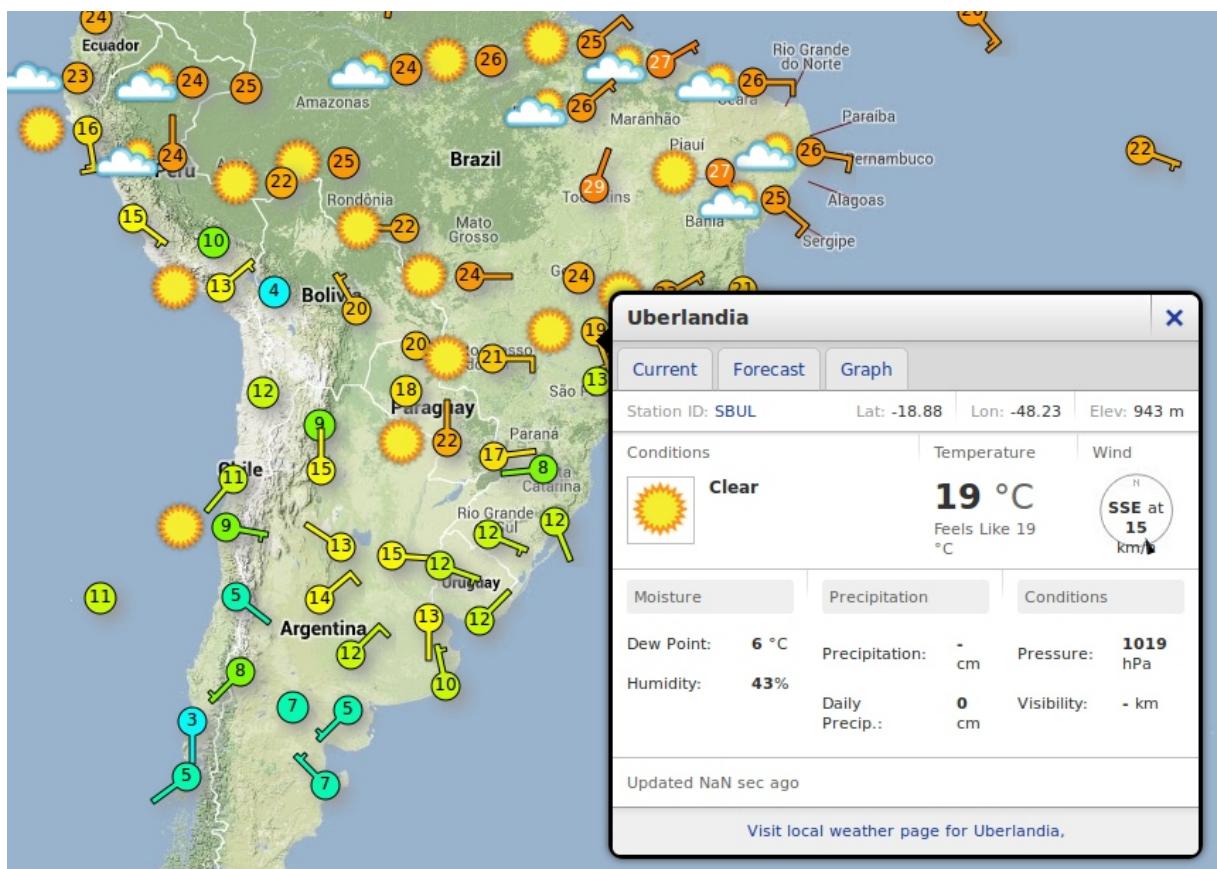


Figura 3: Weather Underground

Fonte: WEATHER..., 2013

3 ARDUINO

Arduino é uma plataforma de computação física, é basicamente um microcontrolador que pode ser facilmente programado por uma interface USB. Ele foi criado em 2005 por Massimo Banzi, um italiano de Milão, com a ajuda de David Cuartielles, David Mellis, Gianluca Martino, e Tom Igoe.

O Arduino é uma plataforma de prototipagem eletrônica *open-source* baseada em hardware e software flexíveis, que pode perceber o ambiente através de sensores e pode modificá-lo através do acionamento de luzes, motores ou outros atuadores (BANZI 2011, pg 9) (ARDUINO, 2013).

O sistema é composto por duas partes, o hardware, que pode ser qualquer uma das placas Arduino ou compatíveis, e o software ou IDE, onde você programa os códigos que rodarão no hardware, normalmente o código é chamado de *sketch*. É o *sketch* que diz ao hardware o que fazer (BANZI 2011, pg19).

Há pouco tempo atrás, montar uma placa que fizesse a mesma coisa que um Arduino e algumas poucas linhas de código simples, era uma tarefa muito complicada, que demandava tempo, muita soldagem, componentes e protótipos. Com a chegada dos microprocessadores todo este trabalho foi substituído por software. Modificar o programa é muito mais fácil do que modificar a parte física de uma placa.

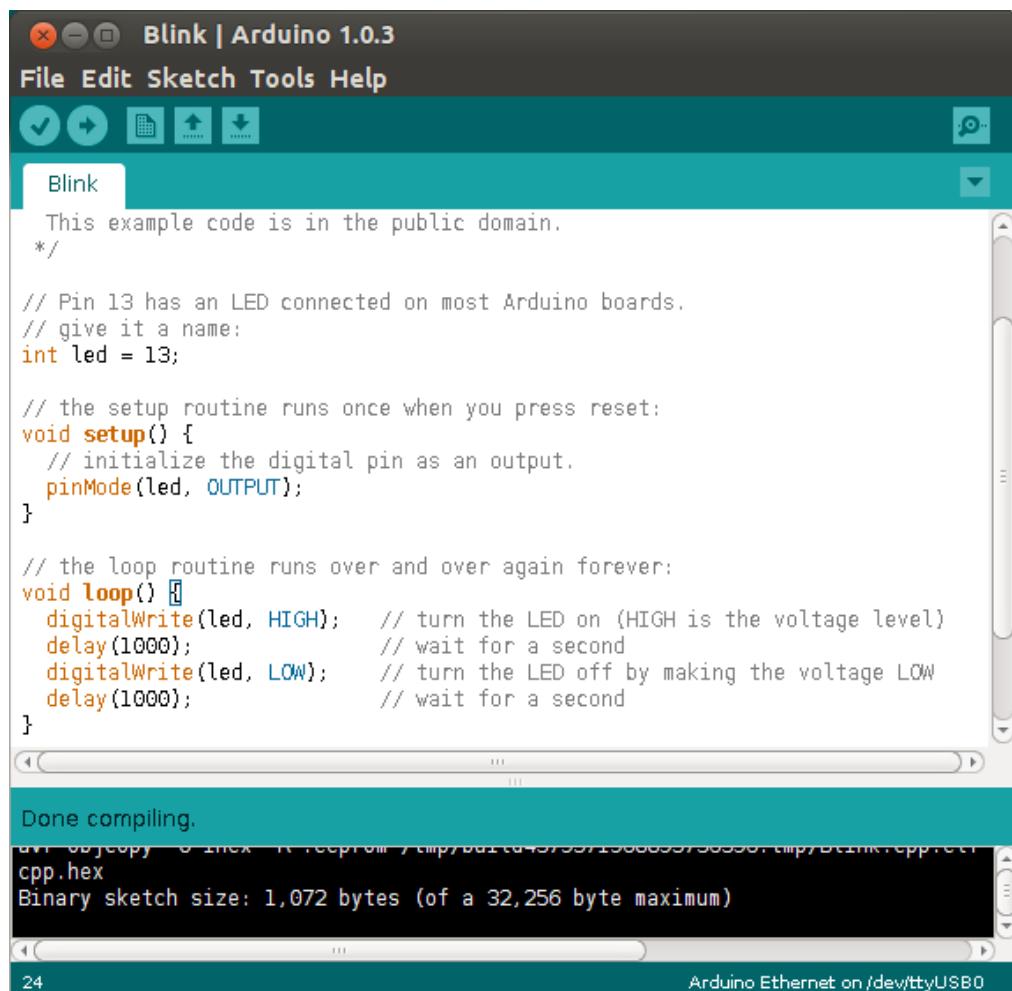


Figura 4: IDE do Arduino

O Arduino tem uma IDE simples, com poucos botões e com uma interface clara, a linguagem usada tem a sintaxe do C, que foi herdado do Processing. Um exemplo de código “hello word” pode ser visto na Figura 4, onde ele aparece junto com a própria IDE. Quando você clica no botão para gravar o código no Arduino, a IDE converte tudo em C puro, envia ao compilador avr-gcc que gera o código a ser gravado no microprocessador.

3.1 PRINCIPAIS PLACAS

Existe uma quantidade razoável de placas que levam o nome Arduino e são consideradas oficiais, a mais comum é a Arduino Uno, podemos vê-la na Figura 5, ela é recomendada para quem está começando. Nem por isso ela faz menos do que as outras, o processamento é de 16mhz e dependendo do CI utilizado, pode ter 16k ou 32k de memória Flash. Conta também com 1k de memória EEPROM e 2k de SRAM.

Possui 14 entradas ou saídas digitais, 6 delas podem ser saídas PWM (*Power With Modulation*) e 2 podem ser entradas de interrupção, 6 entradas analógicas com 10 bits de resolução (1024 níveis), 1 serial do tipo TTL (*Transistor-transistor Logic*), uma conexão SPI (*Serial Peripheral Interface*) e uma TWI (*Two Wire Interface*).



Figura 5: Arduino Uno

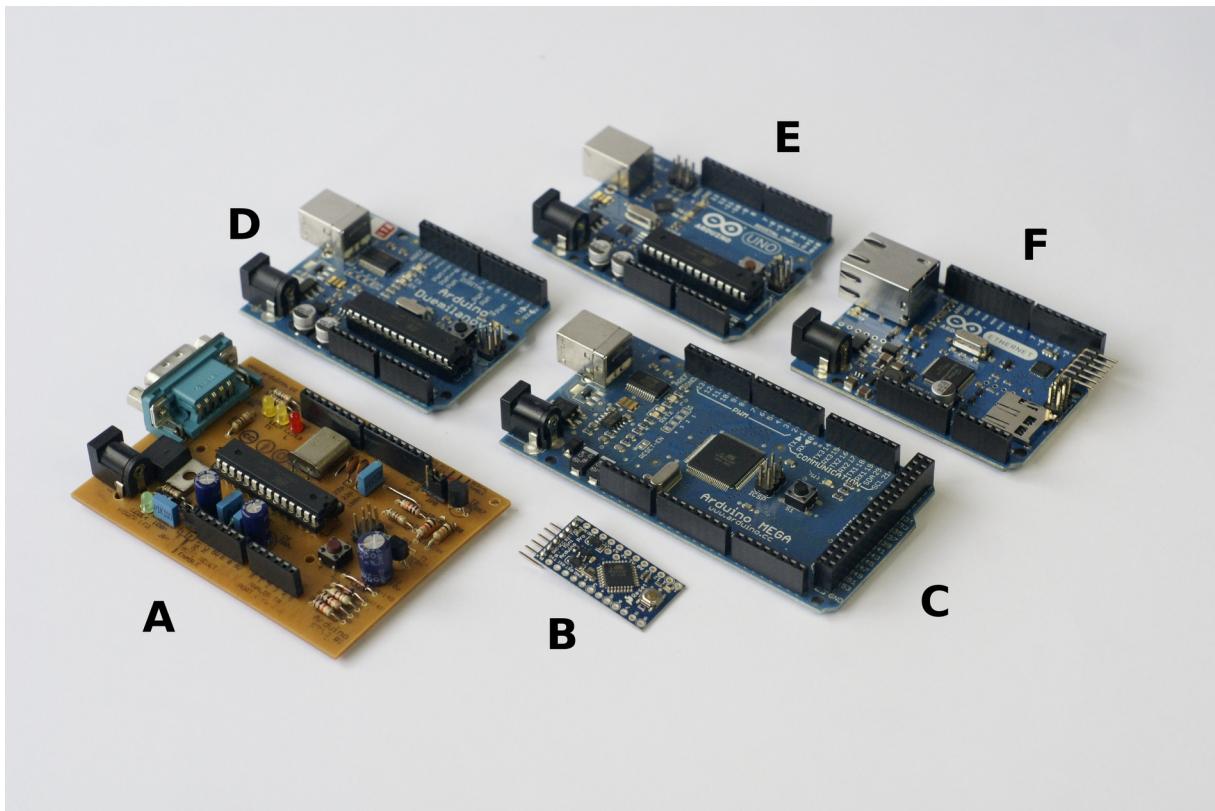


Figura 6: Alguns modelos de Arduino

Na Figura 6, podemos visualizar alguns modelos de Arduino. A placa 'A' é o Arduino Severino, ou *Single-Sided Serial Board*, é uma versão do Arduino que pode ser montada em casa, foi desenvolvida pelo brasileiro Adilson Akashi. No site oficial é disponibilizado um arquivo com o diagrama pronto para ser transferido à uma PCB, contém instruções de montagem e lista de peças necessárias (ARDUINO, 2013). O custo não passa de R\$25,00, porém ele precisa ser gravado por uma interface serial ou com o uso de um adaptador FTDI (interface USB para serial).

Ao lado, a placa 'B', é o Arduino Mini, ele não tem uma conexão USB nem conectores para fios, é usado mais para projetos permanentes, pois exige a soldagem dos cabos e também de sua gravação através do adaptador FTDI.

A placa 'C', é o Arduino MEGA. Esse modelo é o que mais tem entradas e saídas, tanto analógicas como digitais e também conta com a maior quantidade de memória Flash e RAM. Atualmente essa placa já tem uma sucessora chamada Arduino Due, com 84mhz e 512kb de memória Flash (ARDUINO, 2013).

A placa 'D' e 'E' são o Arduino Duemilanove e o novo Arduino Uno, sobre o qual falamos no começo deste capítulo. Ambas são iguais, mas o Duemilanove é mais antigo e conta com menos pinos para *shields*. A placa 'F' é o Arduino Ethernet, que tem o capítulo 3.3 com explicações detalhadas sobre ele.

Além de todas estas placas oficiais, também temos as placas compatíveis, por ser um projeto de software e hardware *open-source* (*Creative Commons CC-SA-BY License*), muitas placas são criadas paralelamente. Do ponto de vista do grupo Arduino, isso não é um problema desde que a marca Arduino não seja utilizada. Isso é muito importante pois possibilita a criação de “Arduinos” derivados para os mais variados fins, entre eles podemos citar o brasileiro Garagino.

3.2 SHIELDS

Todos os conectores na parte de cima das placas, servem à dois propósitos, facilitam a conexão de fios do tipo *wire jumper* e também o encaixe de placas chamadas de *shields*, que executam funções específicas, como um *display*, um *joystick* ou um *driver* para motores.

A plataforma conta com vários *shields*, entre eles está o Shield Ethernet. Recentemente foi lançado o Arduino Ethernet (Capítulo 3.3) que é o Arduino normal mais o Shield Ethernet em conjunto na mesma placa, o preço é menor que as duas peças em separado. A liberdade da plataforma também incentiva que qualquer um crie um *shield* novo.

3.3 ARDUINO ETHERNET

Antes da Arduino lançar esta placa, existia uma similar, que era fabricada pela Sparkfun (SPARKFUN, 2013) chamada de Ethernet Pro, ela tinha as mesmas funcionalidades desta nova. No momento que a Arduino Ethernet começou a ser fabricada, a Ethernet Pro foi tirada do mercado, ambas custavam o mesmo preço, tinham as mesmas características e faziam o mesmo trabalho.

Com estas placas é possível rodar qualquer código de Arduino e ainda contar com a possibilidade de usar a biblioteca Ethernet para comunicação em rede. Pelo motivo do Atmega ser um microprocessador bastante limitado em termos de memória e processamento, essa placa permite os usos mais básicos da rede.

Com relação ao código, é preciso montar todas as requisições (do inglês *request*) à rede de maneira muito primitiva, do mesmo modo com o tratamento das respostas (do inglês *response*). Uma requisição GET a um servidor HTTP deve levar em consideração inclusive o caractere “\n” no final da linha. Algumas funcionalidades de rede foram facilitadas e abstraídas nas últimas versões da biblioteca, como por exemplo, o cliente DHCP, NTP e DNS.

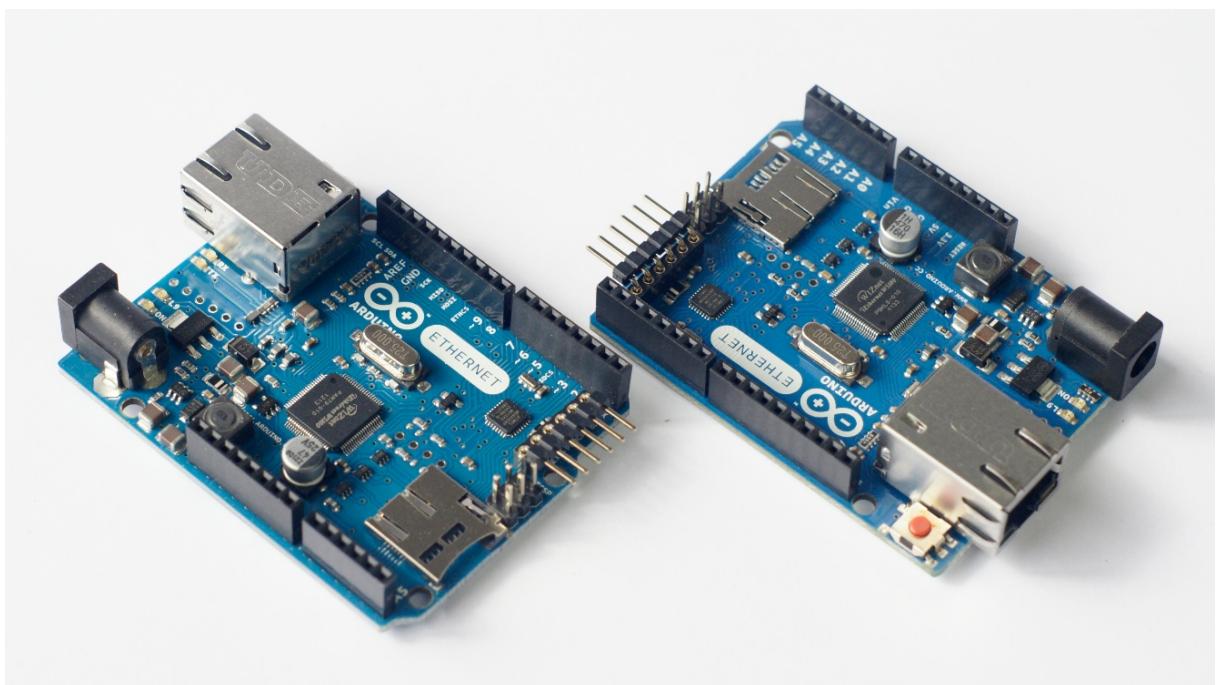


Figura 7: Placas Arduino Ethernet

4 SENsoRES

Vários tipos de sensores foram testados até chegar aos ideais para o projeto, ao final foi percebido que os mais confiáveis são do tipo que tem interface digital e não sofrem com a baixa resolução da conversão A/D das portas do Arduino, que tem 10 *bits* de resolução.

Sensores com interface digital são mais caros que os analógicos, mas isso é compensado pela certeza da informação mais precisa e pela facilidade de conexão ao Arduino, normalmente estes sensores podem ser comprados já soldados à uma *breakboard*, que também facilita a sua montagem.

Outros sensores foram acertados já no primeiro teste, ou por não existir outros disponíveis ou porque um sensor melhor custava extremamente caro, fugindo da ideia inicial de manter uma boa razão entre precisão e preço. Os sensores de vento e chuva são um exemplo disso, o kit completo custa US\$69,00 e vem com pluviômetro, anemômetro e anemoscópio, ao passo que somente um pluviômetro melhor não custa menos que US\$90,00.

4.1 TEMPERATURA

A temperatura é o dado mais simples e o mais importante a ser medido por uma estação meteorológica. O sensor é também o mais simples e barato de todos, uma corrente elétrica passa por uma junção de dois metais diferentes que variam a sua resistência de acordo com a temperatura, também conhecido como termopar.

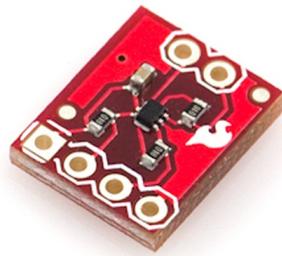


Figura 8: Sensor de temperatura TMP102

Fonte: SPARKFUN, 2013

Para o primeiro protótipo foi usado o sensor TMP102, da Figura 8, ele tem interface TWI (*Two Wire Interface*), que pode ser usada com a biblioteca WIRE do Arduino, precisão de 0.5° e resolução de 0.06° (TEXAS INSTRUMENTS, 2013). Para as versões mais atuais da estação, foi adotado um sensor de temperatura em conjunto com o de umidade, mas sobre ele no capítulo 4.2. No Quadro 1 podemos ver a função utilizada para leitura dos dados deste sensor.

```

int tmp102Address = 0x48;
float getTemperatura(){
    Wire.requestFrom(tmp102Address,2);
    byte MSB = Wire.receive();
    byte LSB = Wire.receive();
    int TemperatureSum = ((MSB << 8) | LSB) >> 4; //it's a 12bit
    int, using two's compliment for negative
    float celsius = TemperatureSum*0.0625;
    //float fahrenheit = (TemperatureSum*0.1125) + 32;
    return celsius;
    // return fahrenheit;
}

```

Quadro 1: Código para leitura do sensor TMP102

4.2 UMIDADE RELATIVA

O ar pode ser formado por no máximo 4% de água, o que equivale a 100% de umidade relativa. Pressão e temperatura podem fazer com que o ar sature ou fique mais seco, logo a umidade relativa depende destes fatores também. Os sensores de umidade usam um tipo de polímero que muda sua resistência elétrica de acordo com a quantidade de água que eles absorvem, e quantidade de água absorvida é proporcional à umidade relativa do ar. (SONNEMAKER, 2011 p. 37)

Três sensores foram testados, o HIH-4030 (HONEYWELL, 2013) vem sendo usado na estação montada, ele é analógico e pode ser lido facilmente pelo Arduino, na Figura 9 podemos ver uma imagem dele. Na época não havia sensores digitais que pudessem ser usados com facilidade e custassem pouco, logo esse sensor analógico, com precisão de 3.5% e resolução de 0.1% foi o ideal.



Figura 9: Sensor de umidade HIH-4030

Fonte: SPARKFUN, 2013

Outro sensor, o HH10D (Figura 33 pg. 55) (HOPE RF, 2013) foi testado em bancada e apresentou bons resultados, com precisão de 3% e resolução de 0.08%, mas após instalado na placa definitiva ele começou apresentar problemas de instabilidade e seu uso foi abandonado em prol do HIH-6130 (Figura 10) (HONEYWELL, 2013), que é um sensor de umidade com temperatura, tem interface digital e é mais atual. O código de leitura deste sensor é extenso e pode ser visto nos anexos.

Com o HIH-6130 é possível conseguir valores mais exatos. A umidade em conjunto com a temperatura também é utilizada para o cálculo da temperatura do ponto de orvalho, é a qual a umidade passa a condensar. Um exemplo disto é quando ocorre nevoeiro, a temperatura do ar e a do ponto de orvalho são muito próximas ou iguais. (GERTZ, 2012 p. 54)



Figura 10: Sensor de umidade e temperatura HIH-6130

Fonte: SPARKFUN, 2013

Um cálculo interessante também pode ser feito com o ponto de orvalho, que é a altura da base da nuvem, basta usar a diferença entre a temperatura e a temperatura do ponto de orvalho e multiplicar o valor por 125, o resultado será a altura da base das nuvens em metros (SONNEMAKER, 2012).

4.3 PRESSÃO ATMOSFÉRICA

O atmosfera é um fluído, e sua densidade e pressão são maiores ao nível do mar, é lendo esta informação que obtemos também a altitude. A pressão média padrão ISA (*International Standard Atmosphere*) ao nível médio do mar (NMM) é de 1013,25hPa, conforme a altitude aumenta, a pressão diminui na proporção média de 1hPa para cada 30ft. (SONNEMAKER, 2011 p. 26 e 29)

Além da diferença de pressão pela altitude, temos variações de pressão conforme a hora do dia, a chamada maré barométrica, onde se percebe um pico de pressão perto das 4h e 16h e uma queda de pressão as 10h e 22h. Também podemos perceber mudanças de pressão de acordo com as condições climáticas e avanços de frentes (SONNEMAKER, 2011 p. 33).

Para efeitos de comparação entre diversas estações, é calculado a diferença de pressão da altitude onde ela está instalada com a pressão padrão ISA ao nível do mar, este resultado é somado ao valor lido no barômetro, assim temos a pressão reduzida ao nível médio do mar. Desta maneira torna-se fácil dizer se a pressão está baixa ou alta somente olhando se ela está acima ou abaixo do padrão ISA.

O barômetro BMP085 (Figura 11) mede a pressão atmosférica em pascals. Na normas internacionais de medidas, a pressão atmosférica é medida em hPa, ou seja, hectopascals, para chegar a esse valor basta dividir a unidade em Pa por 100 e obter hPa. O altímetro, um instrumento comumente usado na aviação usa o mesmo princípio do barômetro para indicar a altitude (PRESSURE... 2013) (BOSCH, 2013).

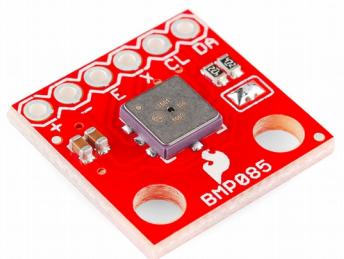


Figura 11: Barômetro BMP085

Fonte: SPARKFUN, 2013

4.4 CHUVA

Os dados sobre a chuva são muito importantes para determinar o clima de uma região, essa informação é coletada por um instrumento chamado pluviômetro. Em uma estação meteorológica convencional, a chuva enche um recipiente graduado onde pode-se ler a quantidade de chuva caída.



Figura 12: Pluviômetro desmontado

Em uma estação automática, o sensor da Figura 12 acumula uma pequena quantidade de água que pinga em uma gangorra, e gera um pulso assim que o peso for suficiente para fazê-lo mover-se e eliminar a água acumulada. O sistema mecânico é muito simples e o pulso pode ser registrado por uma IRQ no Arduino.

O código de leitura do pluviômetro (Quadro 2) é como qualquer outra interrupção, e neste caso também é preciso validar se não ocorreu um efeito repique, que é quando ocorre pulsos falsos subsequentes a um pulso verdadeiro. Segundo o *datasheet* da ARGENT (2013), cada pulso é equivalente a 0.28mm de chuva.

Porém comentários de outros usuários do mesmo conjunto de sensores da ARGENT, afirmam que os valores do pluviômetro e do anemômetro não estão corretos, como não existia um teste comprovando os mesmos, decidiu-se fazer um. Foi deixado pingar no pluviômetro 1 litro de água enquanto um Arduino contou a quantidade de pulsos.

Como 1 litro de água está contido em uma forma de cubo de 100mm de lado, basta calcular a altura deste volume de água, levando-se em conta a área da base do topo do pluviômetro, que mede 50mm por 110mm. Logo, sabemos que 1 litro de água é igual a 1000000mm^3 , então a fórmula do volume ficaria desta forma $1000000 = 110 * 50 * x \rightarrow 1000000 = 5500 * x \rightarrow x = 181,81$. Então a altura deste volume de água teria 181,81mm, e como o Arduino contou 530 pulsos, dividimos a altura pela quantidade de pulsos para obter o valor em mm equivalente à cada pulso registrado. Deste modo chegamos ao valor de 0.343mm que foi adotado como correto durante o desenvolvimento.

```
volatile unsigned int chuva = 0;
volatile unsigned long last_interrupt_timeChuva = 0;
volatile unsigned long interrupt_timeChuva = 0;
attachInterrupt(1, interrupcaoChuva, FALLING); //pino 3

void interrupcaoChuva()
{
    interrupt_timeChuva = millis();
    //less than 100ms, assume its a bounce and ignore
    if (interrupt_timeChuva - last_interrupt_timeChuva > 100) {
        chuva++;
    }
    last_interrupt_timeChuva = interrupt_timeChuva;
}

unsigned int getChuva() {
    unsigned int retorno = chuva;
    chuva = 0;
    return retorno;
}
```

Quadro 2: Funções usadas para leitura do pluviômetro.

4.5 VENTO

Para obter informações sobre o vento, precisamos de 2 sensores, um deles é o anemoscópio (Figura 13) ou comumente conhecido como biruta, ele tem uma série de resistências que indicam a origem do vento. Este modelo não é linear, e sua resistência é alterada por chaves acionadas por imã. Sua leitura é feita analisando o valor da entrada analógica onde ele está conectado, o exemplo do código está no Quadro 3. (SONNEMAKER, 2011 p. 73)



Figura 13: Anemoscópio

```

int funcaoDirVento() {
    int aux = analogRead(1);
    if (aux < 99)
        return 2; //O
    else if (100 < aux && aux < 150)
        return 4; //NO
    else if (160 < aux && aux < 219)
        return 2; //O
    else if (220 < aux && aux < 279)
        return 0; //N
    else if (280 < aux && aux < 349)
        return 4; //NO
    else if (350 < aux && aux < 450)
        return 6; //SO
    else if (535 < aux && aux < 589)
        return 5; //NE
    else if (590 < aux && aux < 650)
        return 0; //N
    else if (700 < aux && aux < 799)
        return 1; //S
    else if (800 < aux && aux < 915)
        return 7; //SE
    else if (916 < aux)
        return 3; //E
    else
        return 8; //algum erro
}

```

Quadro 3: Trecho de código da leitura do anemoscópio

O outro sensor é o anemômetro da Figura 14, este mede a velocidade do vento (SONNEMAKER, 2011 p. 73). Segundo o *datasheet* da ARGENT (2013), ele gera um pulso por segundo sob um vento de 2.4km/h, a partir deste dado conseguimos facilmente calcular qualquer velocidade usando uma regra de três. Para fazer a leitura, conectamos este sensor em uma porta que pode servir de IRQ, sendo assim, atrelamos uma interrupção e a qualquer momento que houver um pulso ele será gravado para posterior processamento, vemos um exemplo da declaração da interrupção e da função que faz isso no Quadro 4.



Figura 14: Anemômetro

```
attachInterrupt(0, interrupcaoVento, FALLING); //pino 2
void interrupcaoVento() {
    vento++;
}
```

Quadro 4: Código de exemplo para leitura do anemômetro

Mas como os dados do *datasheet* deste componente não são confiáveis, foi feito um teste em um túnel de vento para aferir os valores corretos (Figura 16). Assim, procedeu-se leituras de velocidade do vento no túnel com anemômetro de fio quente e também com cálculo de pressão através de um Tubo de Pitot, obtendo-se uma média entre ambas. Paralelamente registrou-se a quantidade de pulsos por 10 segundos naquela velocidade específica, a partir de onde gerou-se um gráfico e uma equação específica para este modelo (Figura 15).

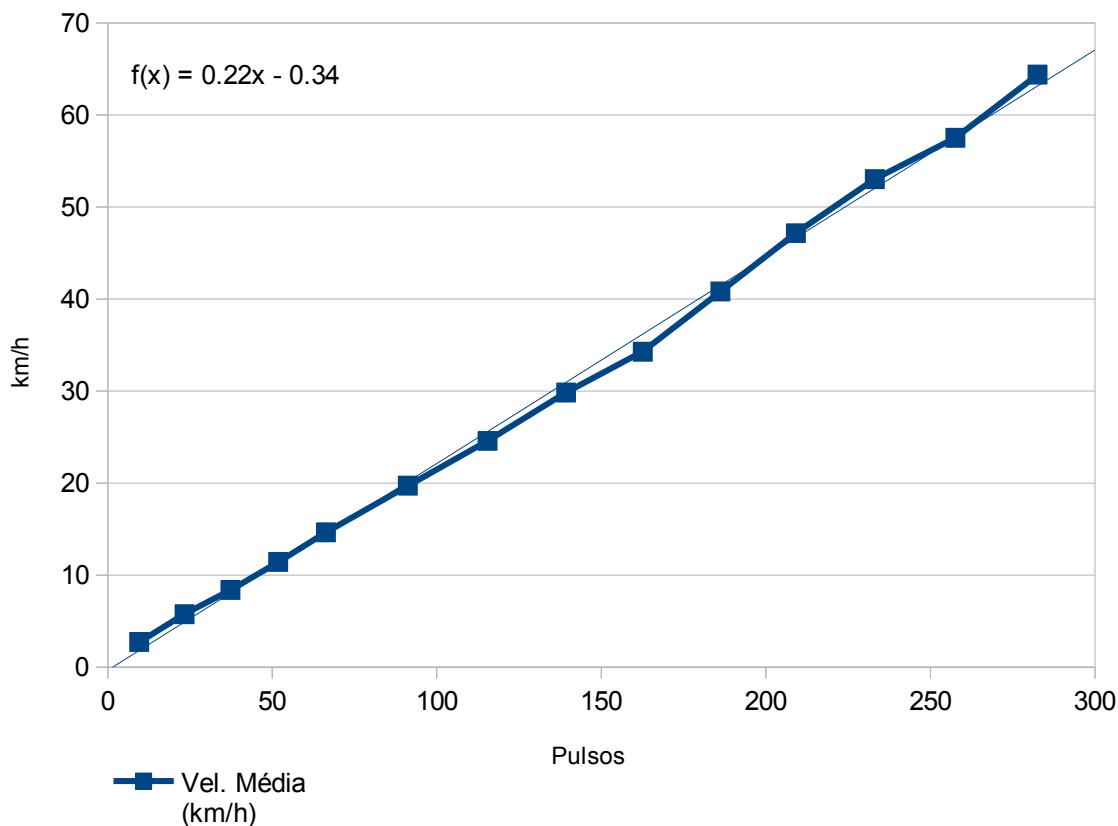


Figura 15: Gráfico de comparação entre pulsos por 10s e velocidade

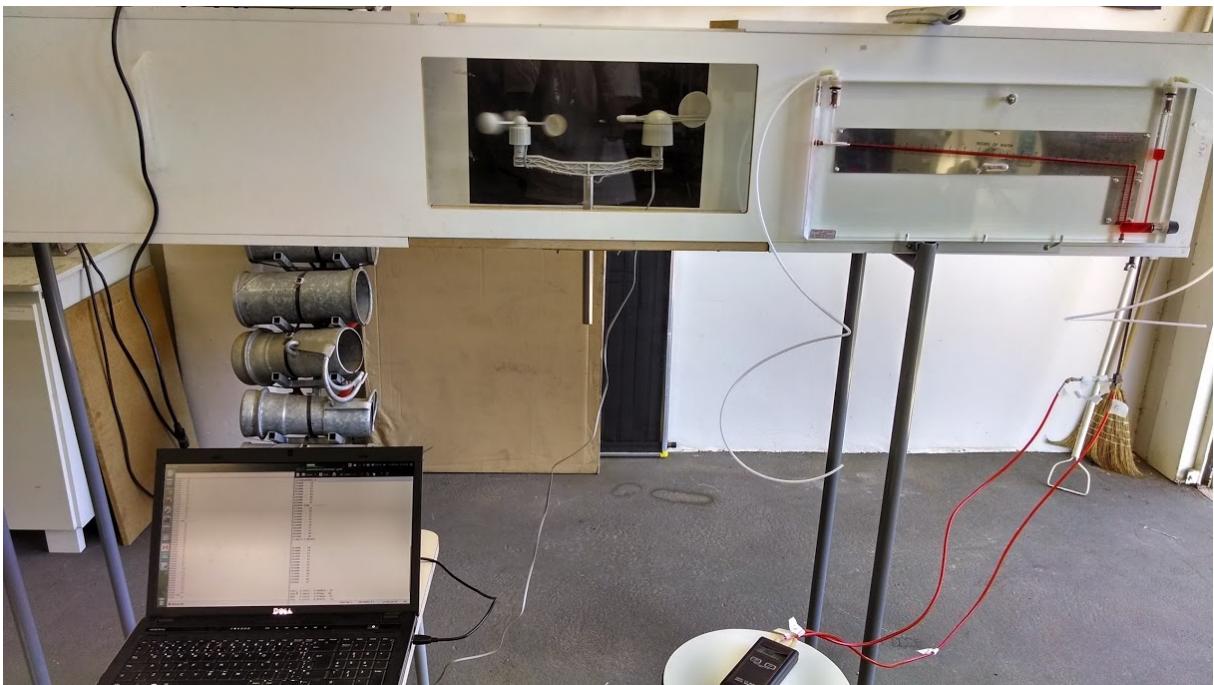


Figura 16: Teste do anemômetro em túnel de vento

No ensaio em túnel de vento, obtemos uma velocidade de 2.73km/h por pulso por segundo, onde o *datasheet* informava 2.4km/h. Mesmo assim, mais testes são necessários, visto que a velocidade medida no Tubo de Pitot foi diferente da medida no anemômetro. Por enquanto, será utilizado o valor de 2.4km/h.

4.6 RADIAÇÃO SOLAR

Este sensor é responsável por captar a radiação solar no espectro da luz visível, de 400nm até 700nm, primeiramente tentou-se utilizar uma fotocélula mas isso mostrou-se inadequado para a aplicação porque tinha pouca sensibilidade, não era linear e saturava facilmente com a luz mais forte do meio dia.

Então o sensor analógico TEMT600 (VISHAY, 2013), da Figura 17 foi testado e apresentou uma funcionalidade satisfatória, ele é analógico e tem fácil leitura, porém ainda tem o problema da calibração dos dados. É muito difícil obter um parâmetro de comparação entre os dados do sensor e valores padrão de W/m^2 ou kJ/m^2 . Por esse motivo os dados de radiação estão com a observação de que são 'aproximados'.

O termo radiação é usado para fenômenos similares porém não idênticos. Radiação é o nome dado a tudo que irradia de uma fonte, no caso do sol, a maior parte da radiação que chega até nós é em forma de luz visível, já no caso do urânio e plutônio por exemplo, o que é irradiado são partículas subatômicas. Os dois tipos de radiação são diferentes e são medidas por sensores diferentes também. (GERTZ, 2012 p. 28)



Figura 17: Sensor de Luminosidade TEMT600

Fonte: SPARKFUN, 2013

5 MONTAGEM

A versão atual é o terceiro protótipo construído, antes dele foi montado uma prova de funcionamento em uma *protoboard*, após duas semanas de funcionamento sem apresentar problemas, as peças foram soldadas em um Prototype Shield, e assim funciona até hoje, por quase dois anos. Após isto o último protótipo apresentado aqui foi terminado.

5.1 PROTOBOARD

A primeira estação foi montada sobre uma *protoboard*, com os componentes encaixados. Permaneceu funcionando desta maneira por aproximadamente duas semanas, como pode ser visto na Figura 18, a placa é uma “Arduino *compatible*”, a Pro Ethernet da Sparkfun. Foi utilizada por que na época ainda não existia o Arduino Ethernet, mas como citado anteriormente, ambas tem as mesmas características.

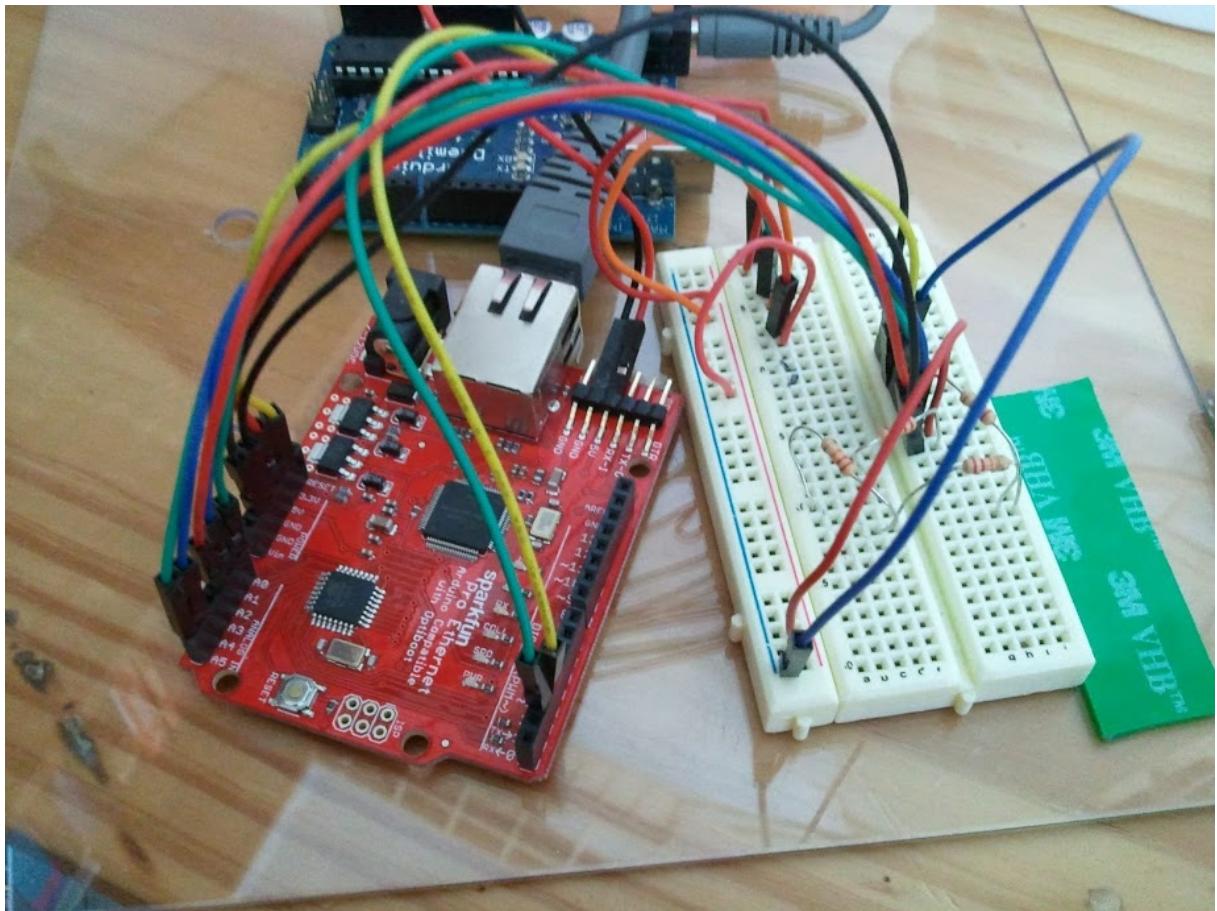


Figura 18: Detalhe dos jumper wires e da protoboard

A Figura 19 mostra os detalhes da montagem dos sensores, eles foram presos à um pedaço de alumínio, com uma isolação de borracha. De cima para baixo, o sensor barométrico BMP085, o sensor de umidade analógico HIH-4030, o sensor de temperatura TMP102 e abaixo dele o sensor de luminosidade TEMT600 (não visível na foto).

O abrigo foi feito de modo artesanal, com 6 potes de sobremesa de formato levemente triangular, e mostrou-se eficiente em manter os sensores longe da luz solar direta e da água. Apesar da simplicidade o interior do abrigo permite a passagem do ar, não forma uma estufa quando está sob luz solar direta e protege os sensores da chuva.

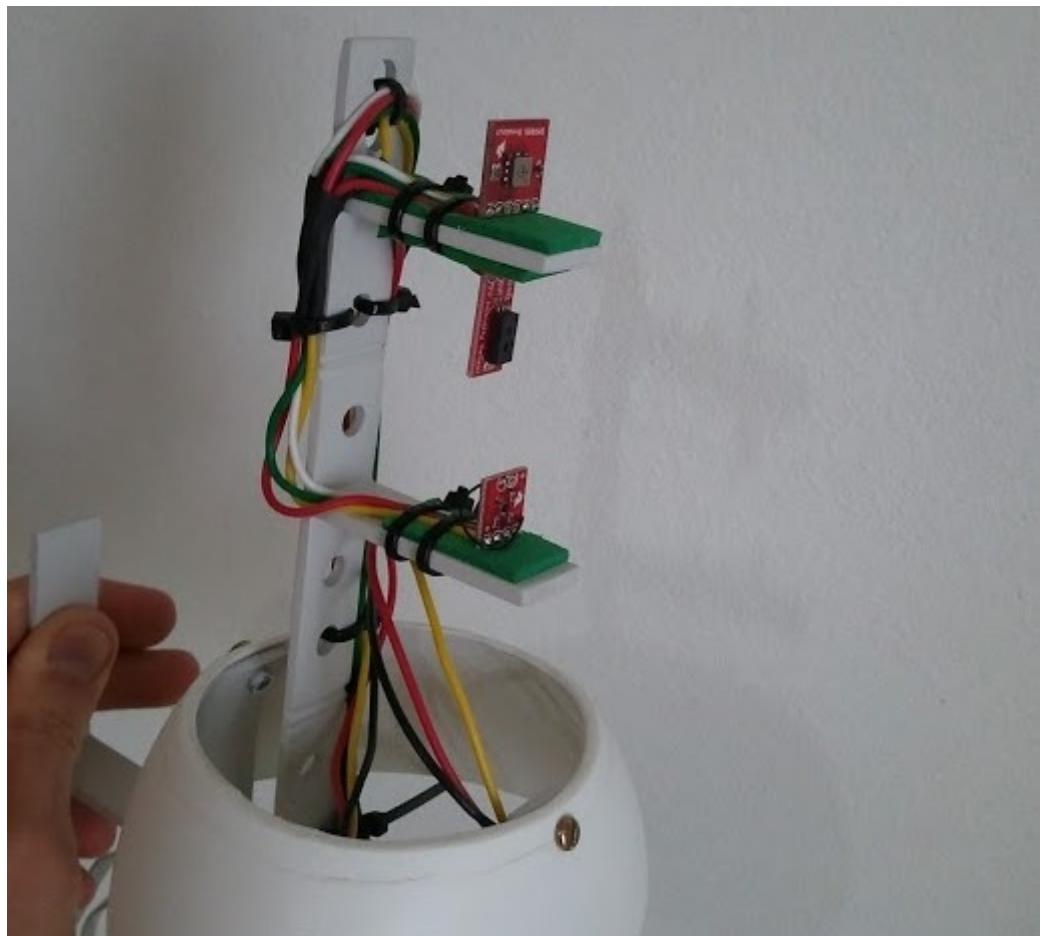


Figura 19: Montagem dos sensores

A Figura 20 mostra todas as peças antes da montagem final, algumas modificações foram necessárias, como a adição de um dissipador sobre o CI responsável pela comunicação de rede, sem ele ocorriam travamentos após 2 ou 3 horas de uso. Também foi adicionado um Arduino somente para transformar os 9v da fonte em 5v diretamente para o Ethernet Pro. O motivo do segundo Arduino é também o superaquecimento do CI 7805 da Ethernet Pro, que esquentava toda a placa e também fazia a rede travar. A solução era temporária mas ainda continua da mesma maneira após 2 anos.



Figura 20: Peças antes da montagem final

5.2 PROTOTYPE SHIELD

O segundo passo foi a troca da *protoboard* por uma placa mais definitiva, por isso foi optado por uma Prototype Shield, que é composta de furos e trilhas que podem ser soldadas para criarem o circuito necessário.

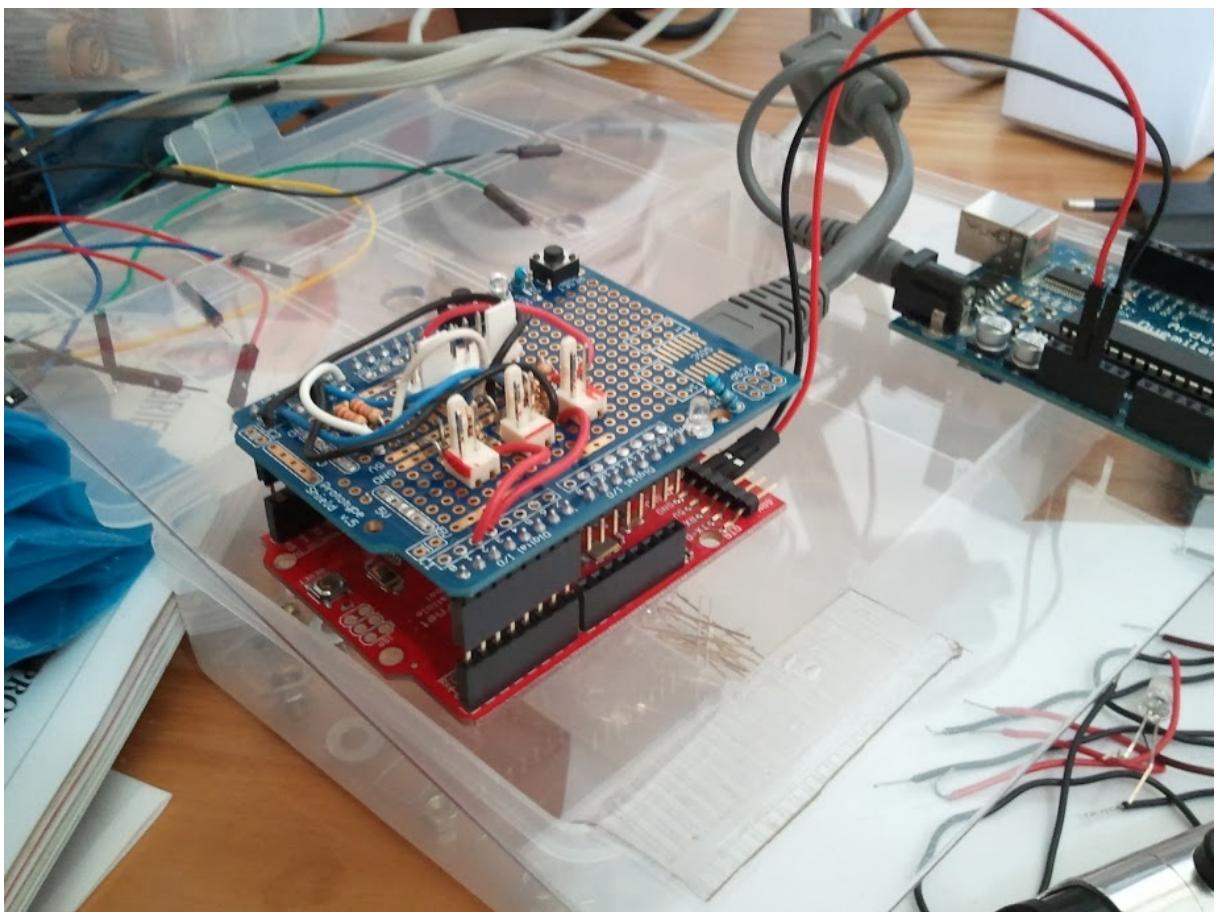


Figura 21: Prototype Shield com todos os componentes soldados

A Figura 21 apresenta uma imagem da Prototype Shield encaixada sobre a placa Ethernet Pro, já com os componentes soldados e todas as conexões feitas. Um conector a mais foi emendado nos pinos do *shield* para afastar as duas placas de modo que permita a passagem de ar para diminuir o calor na Ethernet Pro.



Figura 22: Montagem dos componentes no lugar definitivo

A Figura 22 é uma imagem de como o Ethernet Pro e o shield estão instalados, a energia vem por POE (*Power Over Ethernet*) passivo e vai para um Arduino que tem a única finalidade de fornecer 5v para a outra placa, como comentado anteriormente, tem a finalidade de evitar o superaquecimento.

Na foto da Figura 23, podemos observar o conjunto completo, já funcionando à 2 anos, ainda com o mesmo abrigo artesanal de potes de sobremesa. Após as modificações para evitar o aquecimento, o Ethernet Pro mostrou-se estável e não apresentou mais travamentos.



Figura 23: Segundo protótipo finalizado e funcionando após 2 anos

5.3 SHIELD

Quando este *shield* foi desenvolvido, o firmware foi reescrito novamente, algumas abordagens novas foram adicionadas, como os *timers* para controlar exatamente o tempo entre cada captura de dados e envio, redução do tamanho da função de leitura do barômetro e também validação de senha e token na hora do envio para o servidor.

A Figura 24 apresenta um diagrama da conexão dos componentes nesta ultima versão da estação meteorológica. O tipo de comunicação também está especificado nas ligações.

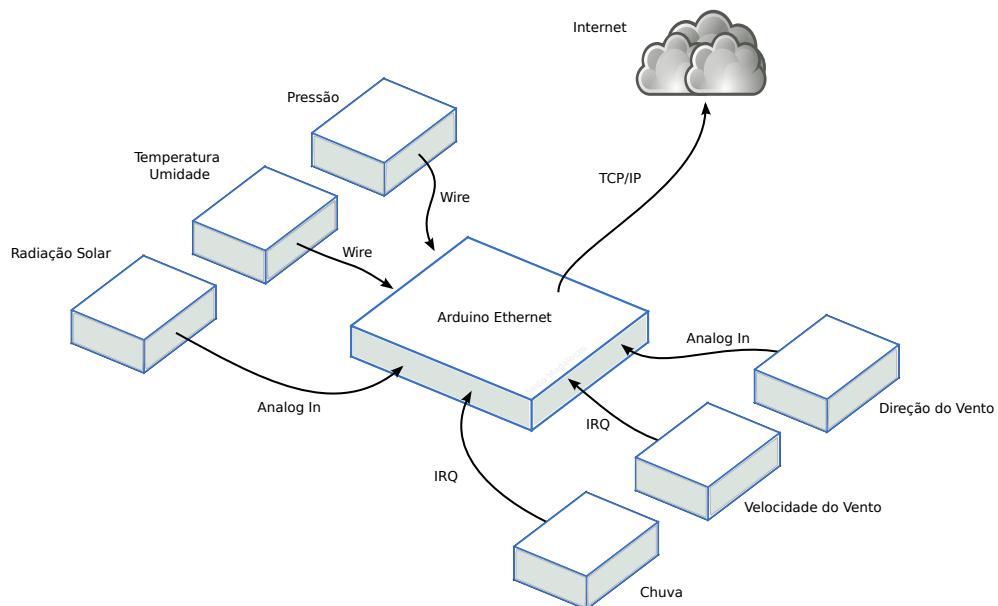


Figura 24: Diagrama das conexões da Estação Meteorológica

Na Figura 25 podemos ver um diagrama que explica de uma forma simples o funcionamento do firmware. A princípio, o `Setup()` é executado, inicializando todos os sensores e declarando as IRQ's do anemômetro e pluviômetro, após isso, um loop verifica a cada milissegundo a próxima tarefa a fazer, ou capturar os dados dos sensores, ou enviá-los ao servidor.

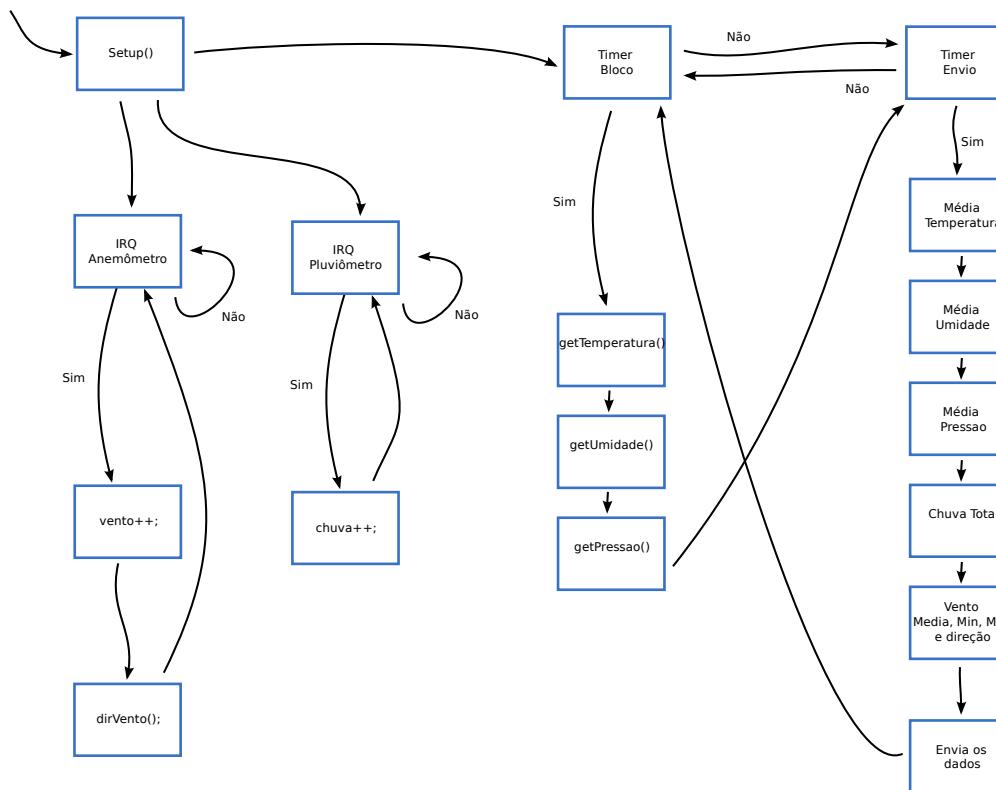


Figura 25: Diagrama de funcionamento do firmware

O protótipo atual foi desenvolvido para ser um *shield* que encaixa diretamente sobre o Arduino Ethernet, para isso uma placa foi desenhada no software Eagle 6.4.0 Light Edition para Linux, usando uma licença de estudante. Uma placa para os sensores também foi criada com o mesmo software.

Este aplicativo permite a montagem de qualquer tipo de circuito, ele possui uma biblioteca de componentes muito vasta e também conta com a possibilidade de adicionar outras bibliotecas de terceiros. Este mesmo arquivo pode ser enviado à uma empresa especializada em confecção de circuitos impressos e assim obtemos uma placa com acabamento profissional.

O desenho das placas é separado em duas etapas, primeiro desenha-se o esquema elétrico da ligação dos componentes, como pode ser visto na Figura 26, cada objeto vermelho é um componente e as linhas verdes são a conexão elétrica entre eles. O objeto “ARDUINO_SHIELDLONGPADS” é um componente de uma biblioteca montada por terceiros, que representa a forma e as conexões de um *shield* Arduino. Todos os outros componentes são próprios do Eagle.

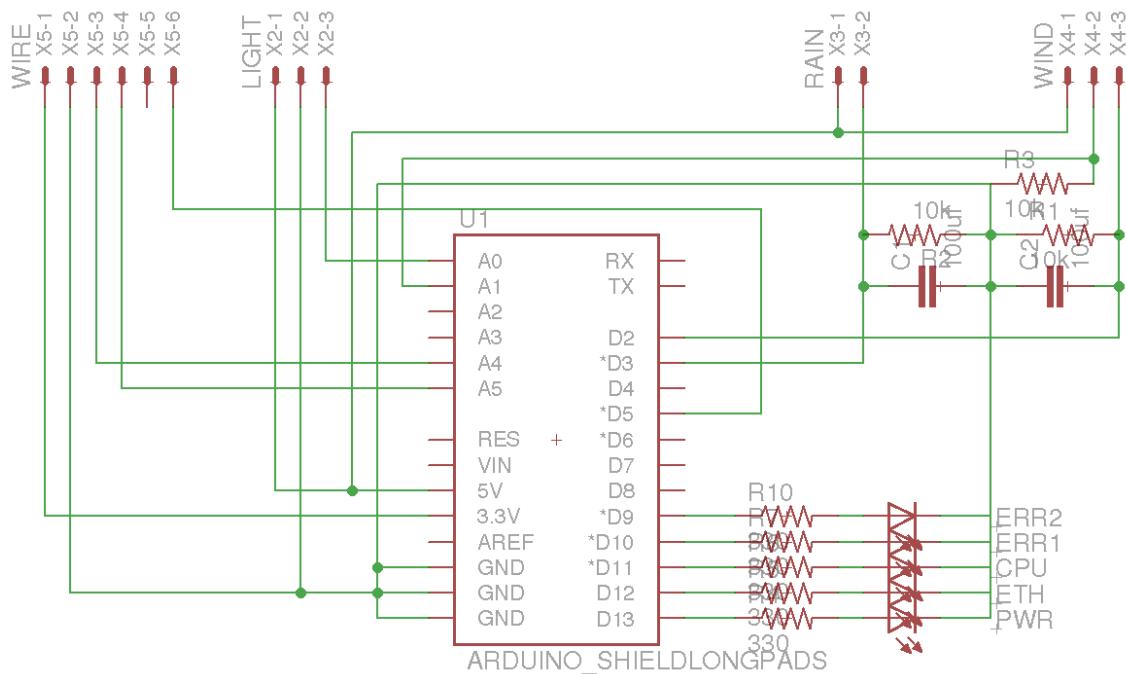


Figura 26: Esquema de ligação do shield

Já na Figura 27, podemos ver a segunda etapa da montagem da placa, que consiste em desenhar as trilhas que fazem a conexões elétrica propriamente dita. Todos os componentes são dispostos na placa e as trilhas são desenhadas respeitando o esquema elétrico feito anteriormente. O Eagle pode tentar fazer isto de modo automático, mas o resultado nunca é satisfatório e sempre é necessário ajuste.

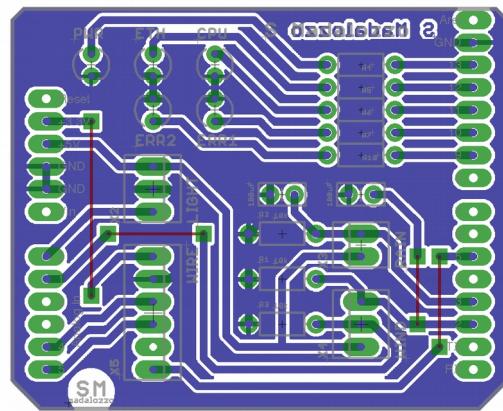


Figura 27: Desenho do shield finalizado com todos os layers

É nesta etapa também que são acertados todos os itens a serem serigrafados e também algumas informações diretamente no cobre. As marcas verdes são pontos onde os componentes serão soldados, é onde não será passado tinta, permitindo que a solda fixe. A parte azul é o cobre que será protegido por esta tinta, que tem função principal de evitar a oxidação. Os textos e formas em cinza são a serigrafia que a placa irá receber, ela contém informações sobre os componentes e conexões.

A Figura 28 e a Figura 29 mostram o esquema da placa dos sensores e o *layout* da placa já pronta para ser gravada na PCB.

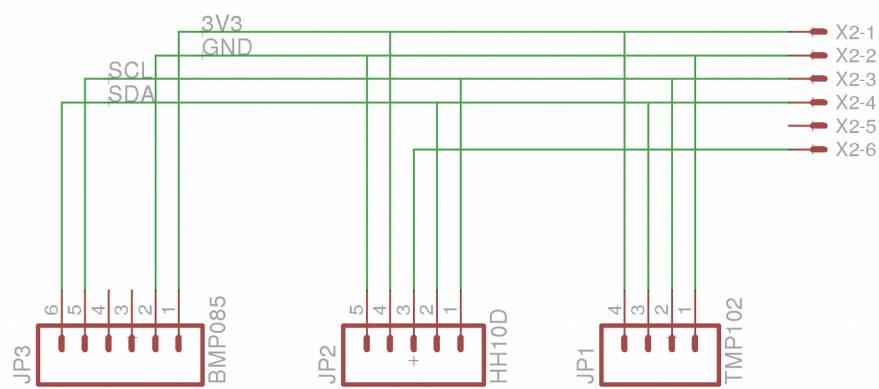


Figura 28: Esquema da placa dos sensores

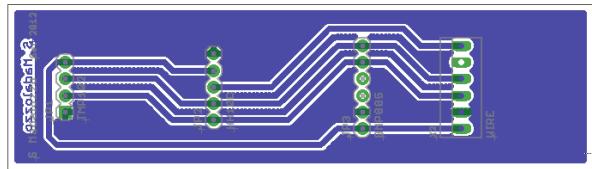


Figura 29: Desenho da placa dos sensores com todos os layers

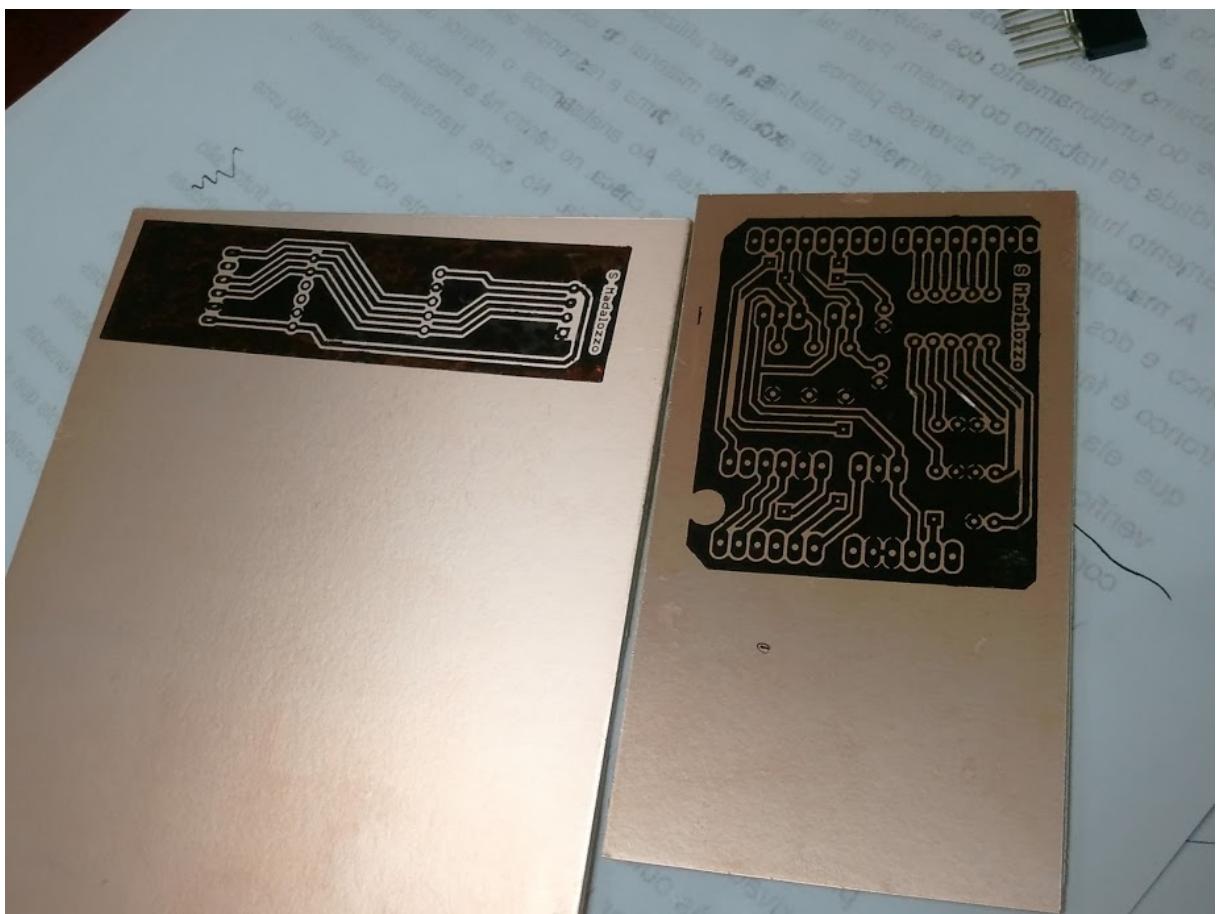


Figura 30: Placa de circuito impresso pronta para a corrosão

Na Figura 30 podemos ver as placas antes do processo de corrosão com percloroeto de ferro, o desenho do circuito foi gravado nelas usando um processo caseiro chamado de “*toner transfer method*”, que consiste em imprimir em uma impressora à laser o circuito espelhado e em seguida, usando um ferro de passar roupa na temperatura mais alta, esquentar o papel com o desenho virado para a face cobreada da placa.

Este processo nem sempre fica correto na primeira tentativa, mas as eventuais falhas podem ser corrigidas com uma caneta antes da corrosão. A mesma técnica pode ser usada para imprimir a localização e nomes dos componentes na parte de cima da placa.

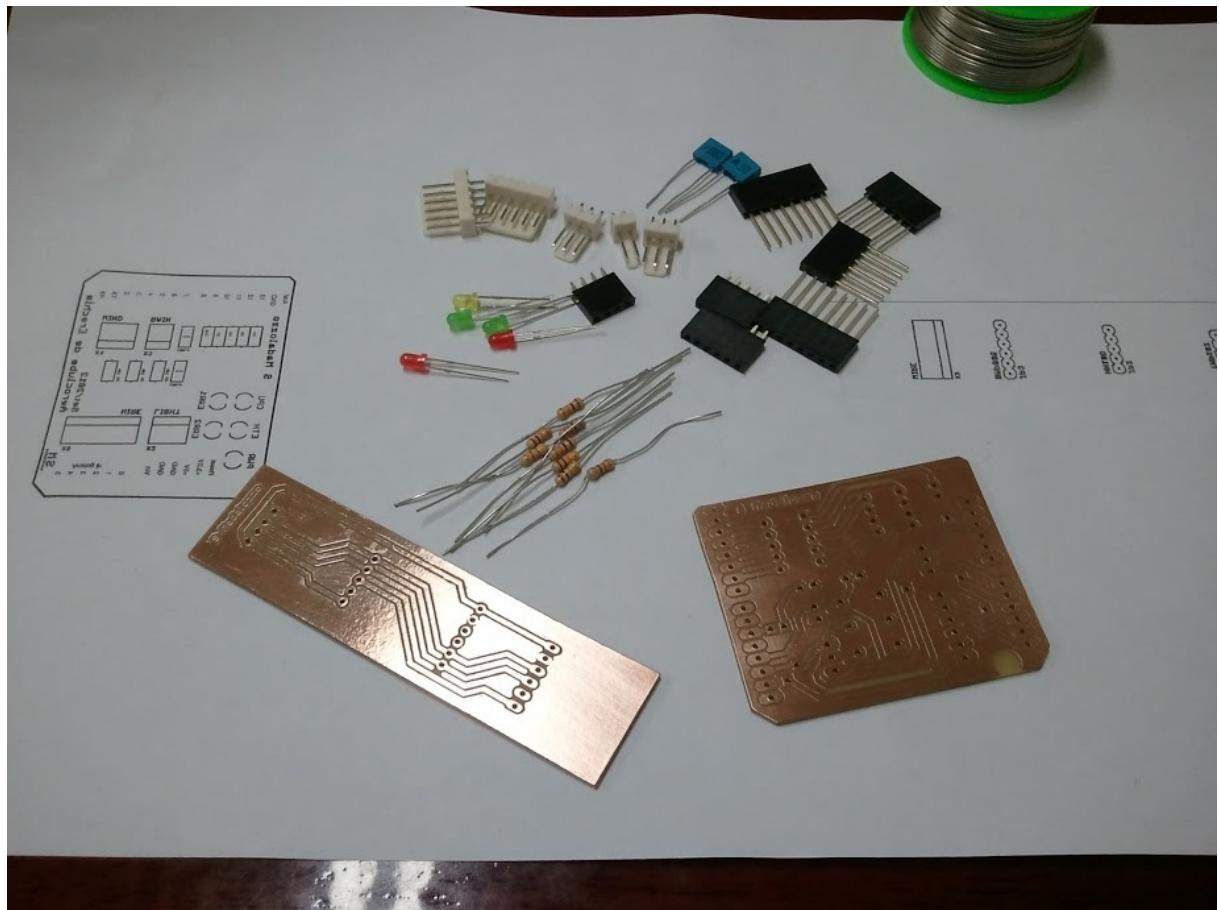


Figura 31: Componentes e placas corroídas, furadas e cortadas

Na Figura 31 temos uma imagem de todas as peças antes do início da soldagem dos componentes, e na Figura 32 podemos ver o processo de soldagem em andamento.

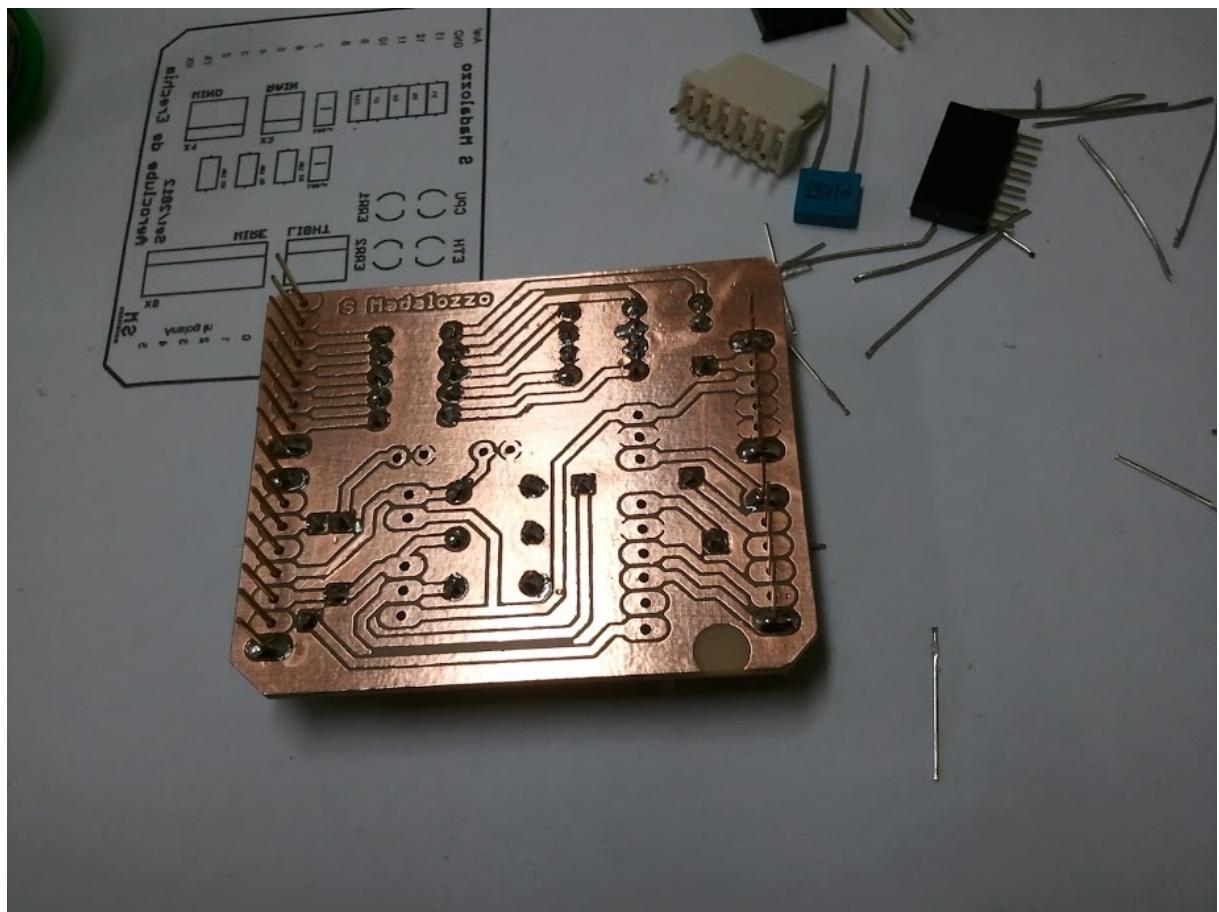


Figura 32: Shield durante a soldagem dos componentes

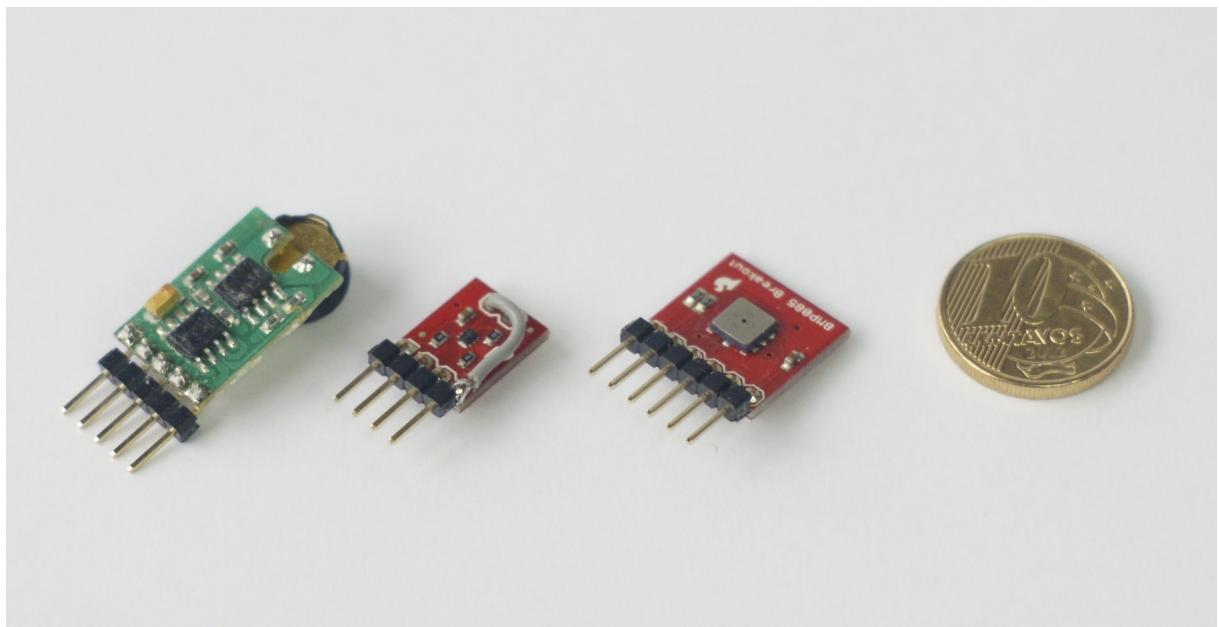


Figura 33: Sensor de umidade, temperatura e pressão com os conectores

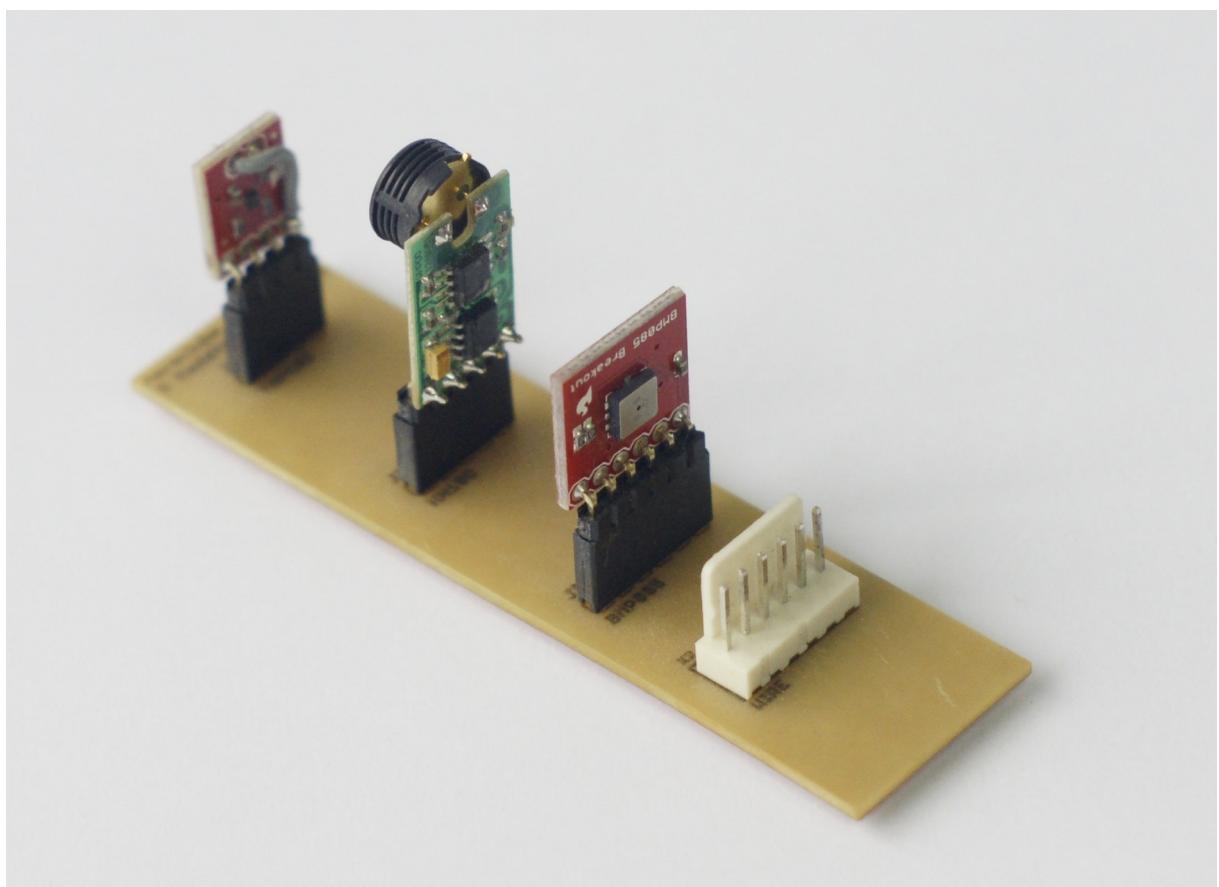


Figura 34: Placa de apoio com o sensor de temperatura, umidade e pressão encaixados.

A Figura 33 exibe os três sensores com os conectores já soldados, e a Figura 34 mostra os mesmos conectados à placa pronta.



Figura 35: Shield encaixado sobre o Arduino Ethernet

A Figura 35 mostra o *shield* pronto e já encaixado sobre o Arduino Ethernet, o conector branco maior, é a ligação entre o shield e a placa dos sensores mostrada na Figura 34.

Mesmo esta placa, feita com atenção, possui um erro que foi percebido somente quando começou-se os testes, como o Arduino Ethernet é um conjunto de um Arduino com um Ethernet Shield, os pinos 10, 11, 12 e 13 são reservados para a comunicação entre as duas placas e não podem ser usadas para outro fim.

O que ocorreu é que esses pinos foram conectados aos leds de *status* do Shield montado, ou seja, 4 dos leds não podem ser utilizados. Isso somente prova que o hardware precisa ser muito bem projetado antes de ser construído, e que não podemos fazer como no firmware, trocando algumas linhas e corrigindo o problema com uma nova compilação.

Outro ponto que não ocorreu como o planejado foi a placa dos sensores, uma trilha invertida também foi percebida antes de ligar o circuito todo. Esse problema já foi corrigido e na Figura 28 e Figura 29 eles estão corretos, porém na Figura 30 a falha pode ser percebida no sensor do meio.

Após o início dos testes com o sensor de umidade HH10D percebeu-se que o desempenho não era satisfatório. Este sensor usa um EEPROM onde são gravados dois valores, um de sensibilidade e outro de *offset* ou desvio , ambos precisam ser lidos para o calculo dos valores, a umidade é lida através da contagem da frequência e depois calculada com os outros valores lidos.

Nos testes com uma *protoboard* este sensor mostrou-se funcional, mas quando o sinal passa por um pedaço de fio mais longo os valores começam a flutuar de maneira irreal. Soma-se também o fato de que a leitura de frequência à nível de processamento é muito cara, por estes motivos, abandonou-se o uso do mesmo.

Para substituir o HH10D foi adquirido o HIH6130, que é um sensor de temperatura e umidade digital (Figura 36). Apesar de custar 3 vezes mais que o TMP102 e o HH10D juntos, a troca permite uma facilidade um pouco maior na programação, uma peça a menos no conjunto inteiro e uma precisão maior nas duas leituras. Deste modo, também já não tem muitos motivos para construir uma placa somente para os sensores, já que os dois que sobraram (BMP085 e HIH6130) são digitais e precisam dos mesmos 4 fios. Ela pode ser montada com facilidade usando uma placa de prototipagem comum.

Outra falha que ainda permanece é o problema de calibração do sensor de radiação solar, na falta de um modo correto de fazer isso, este sensor foi removido desta última versão da estação meteorológica até que obtenha-se um sensor melhor ou uma maneira de calibrá-lo.

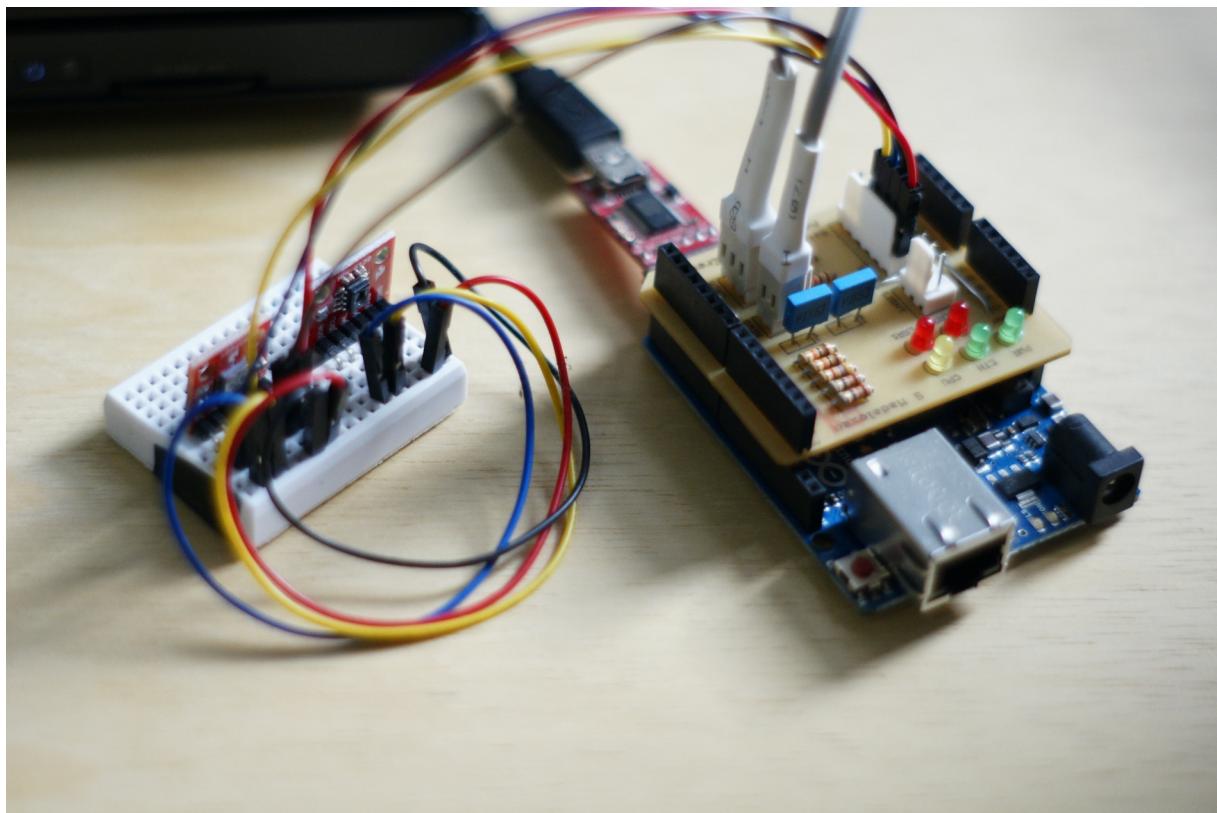


Figura 36: Teste do sensor HIH-6130 e BMP085

Todo esse conjunto, com *shield* e sensores novos, até a entrega deste trabalho, não foram colocados em campo para teste em um ambiente real. Mas o que todos os testes indicam é que funcionarão muito bem e ainda provendo dados mais precisos.

6 SERVIDOR

Além do hardware e do firmware, também temos o software que roda no servidor. Enquanto o firmware lida com poucos dados, tem pouca memória e pouco processamento disponíveis, o lado do servidor precisa trabalhar com grandes quantidades de dados ao mesmo tempo.

Após 22 meses de funcionamento, o banco de dados da primeira estação meteorológica já tem 966000 linhas e mais de 140MB. Passará de 1 milhão em menos de 30 dias. Se adicionarmos mais estações à este sistema, vamos criar uma base de dados gigantesca em pouco tempo, tornando o backup e a manutenção cada vez mais demoradas.

Por isso, a frequência do envio de dados teve que ser repensada e alterada de acordo com a razão de mudança do tempo e a quantidade de dados disponibilizados. Somente os valores acumulativos, como a chuva, continuam sendo tratados da mesma maneira.

Os intervalos foram modificados baseado no documento do INMET (2011), que faz muitas médias para gerar os valores durante 1 hora. Como é necessário manter a premissa da informação em tempo real, optou-se por ler os dados a cada 30 segundos e após 5 minutos fazer uma média e enviá-los ao servidor

Com o intervalo maior, não perdemos as mudanças mais sutis no tempo e ainda assim mantemos uma regularidade suficiente para continuar sendo em tempo real. Afinal, as variações de temperatura, umidade, pressão e todos os outros dados climáticos não mudam drasticamente no intervalo curto de 1 minuto.

6.1 ESTRUTURA DO SERVIDOR

A estrutura do servidor consiste em três partes, um banco de dados, um script de recebimento de dados e um *front-end* para a exibição. Como pode ser visto na Figura 37, o Arduino Ethernet da estação meteorológica envia os dados para o *back-end* (script de recebimento de dados), que grava as informações na base, o que possibilita posteriormente a exibição pelo *front-end*.

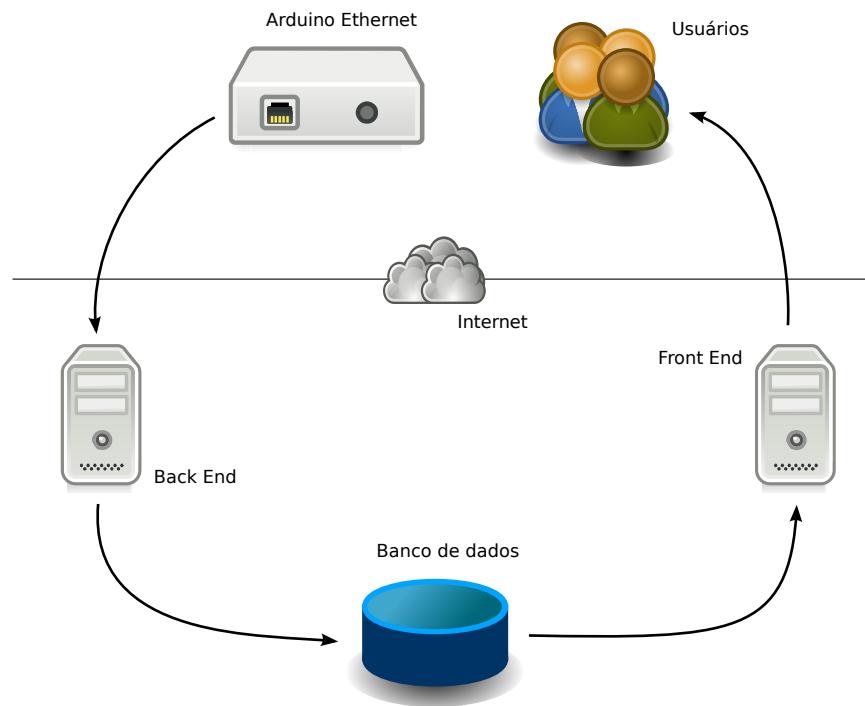


Figura 37: Diagrama de tráfego de dados

Inicialmente o serviço rodava em um site pessoal hospedado na LocaWeb, com o banco de dados, o script de recebimento dos dados e uma única página que exibia os gráficos da única estação que existia até o momento.

No projeto de expansão para suportar mais de uma estação meteorológica, percebeu-se que o plano de hospedagem atual não era suficiente e aumentá-lo geraria um custo maior e uma dificuldade na manutenção do servidor. Durante os dias de mais acesso, a página carregava lentamente e algumas vezes não exibia nenhuma informação. O problema seria muito maior com mais estações.

Por este motivo foi optado pela troca contratando a PaaS (*Platform as a Service*) Jelastic (JELASTIC, 2013), provido pela empresa brasileira Websolute. É um serviço que cobra por hora/máquina, cada servidor ou banco de dados é chamado de *cloudlet*, e é possível adicionar até 32 *cloudlets* em cada *node* e pode-se ter até 16 nodes, cada *cloudlet* tem 200 Mhz e 128 MB de memória RAM. O uso de disco é contabilizado separadamente por GB/hora e o tráfego de rede não é cobrado. Tudo isso configurado de modo transparente e funcionando sem qualquer modificação do código da aplicação, veja a Figura 38.

A parte prática disto é que enquanto o acesso é baixo, somente um *cloudlet* de cada tipo (PostgreSQL e Apache) é contabilizado por hora, assim que a demanda aumenta, mais *cloudlets* são designados para tratar as requisições, isto permite o custo baixo enquanto o acesso é pouco, e só é pago mais nas horas de pico, ao mesmo tempo que mantém o serviço sempre respondendo.

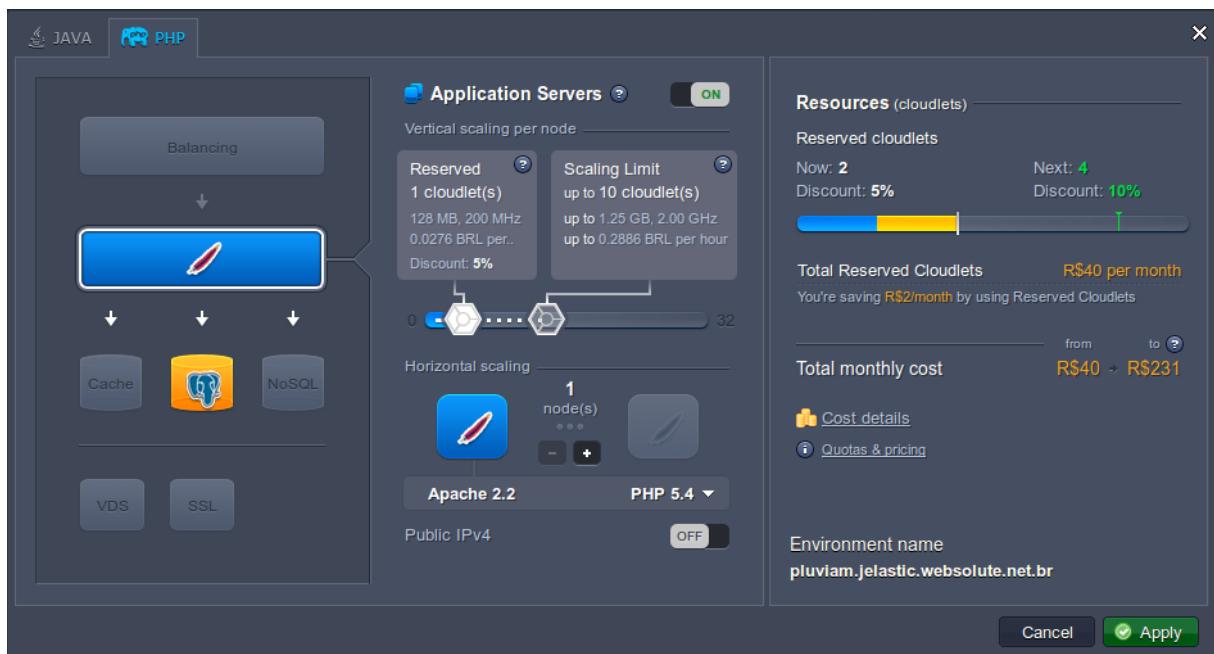


Figura 38: Configuração da aplicação no Jelastic

O Jelastic foi criado em 2010, foi uma das primeiras plataformas de computação em nuvem. A empresa mantenedora tem uma filosofia de não monopolizar as vendas do seu software, ela disponibiliza sua plataforma para parceiros no mundo todo, e foi isso um dos fatores que afirmaram a escolha deste serviço, pois a Websolute é daqui e o *datacenter* está em solo brasileiro. Seus concorrentes diretos são o Heroku, Amazon Beanstalk, Google App Engine e o OpenShift ainda em fase *beta* porém gratuito.

6.1.1 Back-end e recebimento de dados

O *back-end* é composto de um banco de dados e de um script de recebimento de dados. Tanto o script de recebimento de dados quanto o *front-end* rodam no mesmo *node*, por questões de custo. Porém a aplicação foi concebida de modo que isso possa rodar em *nodes* diferentes, um para cada módulo. O banco de dados já roda em um *node* separado.

O sistema de recebimento de dados funciona baseado em um único arquivo PHP que recebe por GET as informações à serem inseridas no banco. O Arduino Ethernet faz uma requisição GET na URL deste script, passando nos parâmetros as informações. Nos parâmetros também são passados os dados para autenticação da estação, tais como o id, *token* e senha.

O script primeiro faz a validação da origem da requisição através do *header* que contém o id da estação, se o mesmo é encontrado, a senha e o *token* também são validados, à partir deste ponto, os dados são inseridos na base e já estão disponíveis para consulta. Em qualquer ponto do script, se algo ocorrer fora do previsto, a execução é interrompida e uma resposta do tipo 404 é enviado ao cliente, caso contrário uma página com o número '1' indica que todos os dados foram gravados na base. A Figura 39 exibe todo este fluxo.

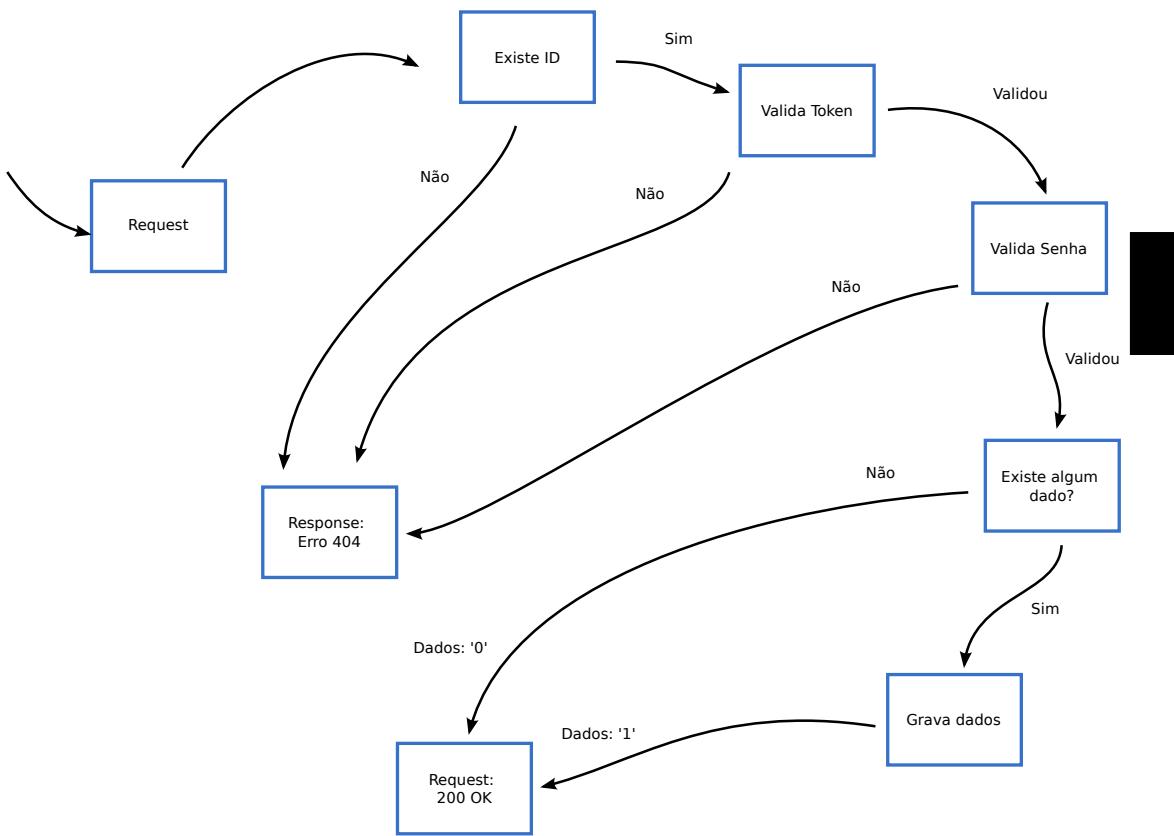


Figura 39: Diagrama de fluxo do back-end

Este mesmo script gera logs para posterior verificação de erros nos dados, erros de conexão e também qualquer tentativa de inserir dados não reais na base. Futuramente com novas versões do Arduino, com mais memória e poder de processamento, talvez seja possível criar uma conexão HTTPS capaz de mitigar esse tipo de problema.

6.1.2 Banco de dados

O banco de dados PostgreSQL foi escolhido por afinidade de uso, já que durante a graduação foram feitos trabalhos usando este SGBD. O PostgreSQL tem uma licença BSD, que permite sua utilização sem pagamento, tanto para uso comercial quanto não comercial. (POSTGRESQL... 2013). A Figura 40 apresenta o modelo ER do banco de dados, e como podemos ver ele é muito simples e tem poucas tabelas.

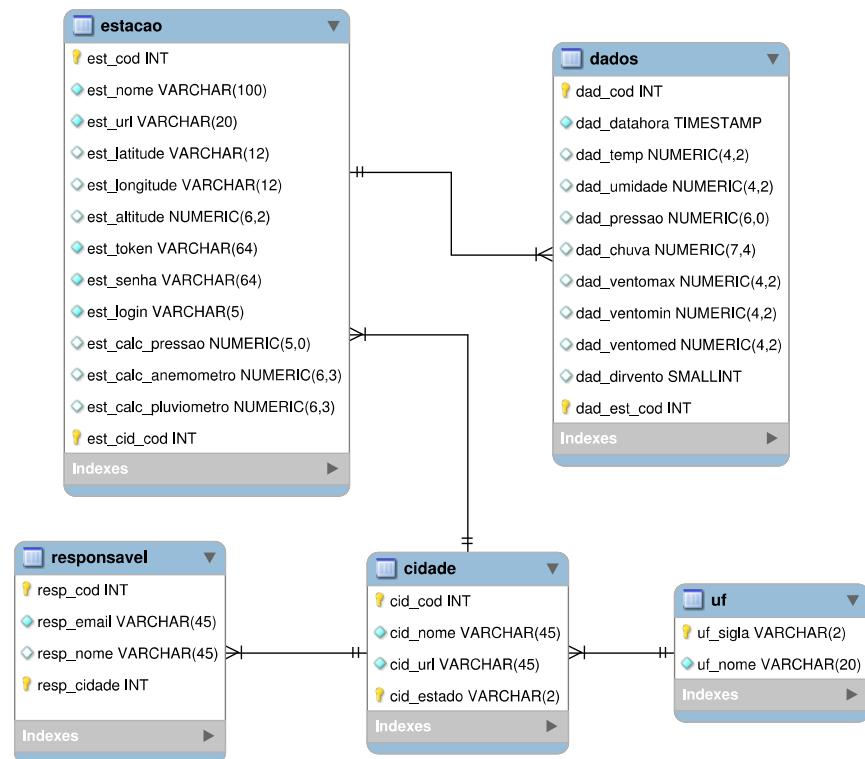


Figura 40: Modelo ER

6.1.3 Front-end

O *front-end* (Figura 41) é uma interface para visualização dos dados das estações meteorológicas, exibindo gráficos e informações. Foi desenvolvido em PHP, usando Twitter Bootstrap para gerar o layout de toda a página, automaticamente tornando-o responsivo, compatível com os navegadores de tablets e celulares com qualquer tamanho de tela. Essa compatibilidade é muito importante visto que atualmente 14,5% são acessos de celulares e 7,5% de tablets.

A aplicação utiliza também a biblioteca Google Chart para gerar e exibir os gráficos, esta biblioteca pode exibir dados em vários tipos de gráficos, estáticos ou interativos, é compatível com todos os navegadores atuais e também com mais antigos como o IE7.

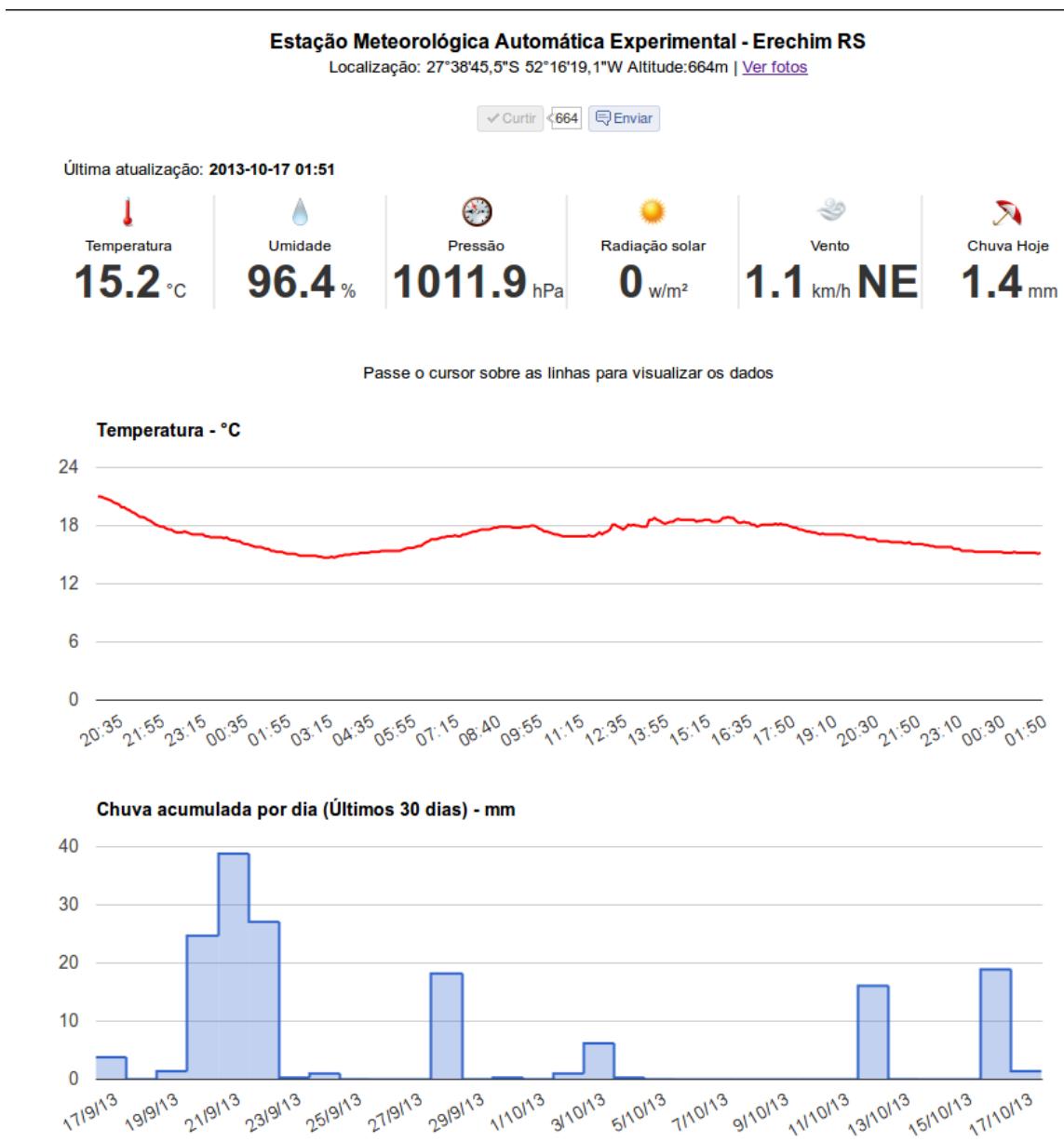


Figura 41: Interface com dados atuais e gráficos

7 CONCLUSÃO

Os dados gerados por uma estação meteorológica conectada à internet podem ser muito bem aproveitadas, como primeiro fato, podemos destacar a informação em tempo real, também é muito útil a quantidade de informações, sendo que a cada 5 minutos temos dados atualizados.

Recentemente, em um acontecimento climático na cidade de Taquarituba, interior do estado de São Paulo, onde ocorreu um tornado, a velocidade do vento não pode ser averiguada pois o INMET não tem uma estação na cidade, e na cidade vizinha a estação marcou meros 54km/h.

Segundo FOLHA (2013) levando-se em conta a destruição gerada pelo efeito, em comparação com outros locais, ele pode ser considerado um tornado de magnitude F1, que tem velocidade entre 117km/h a 138km/h. Porém, isso é somente hipótese, e nunca saberemos a velocidade que o vento atingiu na cidade pois lá não tinha uma estação meteorológica para medi-lo.

São nestes espaços geográficos entre as estações oficiais, que uma estação meteorológica mais simples se destacaria, gerando informações valiosas onde nenhuma empresa ou órgão governamental se compromete ou tem interesse.

Mas a aplicação mais inesperada para uma estação meteorológica é saciar a curiosidade das pessoas quanto aos eventos climáticos extremos, como frio e calor intenso, chuvas e ventos fortes. E percebi que elas gostam de saber estas informações da sua cidade e em tempo real.

Por mais de um ano venho analisando os acessos da primeira estação que fica em cima de minha casa e é claramente perceptível que em dias de chuva, frio, calor e vento forte que os acessos aumentam. Quando o Rio Grande do Sul, Santa Catarina e Paraná tiveram neve neste ano, segundo o Google Analytics do site, os acessos aumentaram 2440% em uma semana, alcançando 8830 acessos em um único dia e chegando a 79 simultâneos.

Para todos estes usuários, o que importa não são os dados cientificamente comprovados do INMET, com hora UTC, vento em m/s, e que na maioria das vezes estão à quilômetros de distância, e sim os dados locais, fáceis de interpretar e entender, exibidos em gráficos intuitivos onde se pode ver a evolução do tempo nas últimas horas e até arriscar hipóteses sobre o comportamento futuro do tempo, respondendo à possível pergunta do “será que chove?”.

Como trabalhos futuros, vários itens podem ser desenvolvidos. Partindo da ideia de que o sensor de umidade e temperatura estão no mesmo CI, pensou-se em uma versão mais barata da estação, voltada para o meio das cidades, é deixar somente o sensor de temperatura e umidade, com isso o preço total cai pela metade e permite que o conjunto seja instalado na sacada de uma casa ou apartamento, lugar que os sensores de vento e chuva gerariam dados com muita influência das construções que os cercam, tornando-os imprecisos e inúteis.

Outra ideia para implementações futuras é a adição de novos sensores, um dado muito interessante seria obtido com um sensor de descargas atmosféricas, ou raios. Este é um sensor que não existe para venda, e os que existem são muito específicos e caros, dificultando a sua integração com o Arduino. Mas uma versão simples pode ser construída com componentes comuns e com desempenho satisfatório (BITSON, 2006).

Outro sensor que poderia ser adicionado à uma estação seria um contador Geiger, para medir a radiação beta e gama, na maior parte dos anos ele não acusaria muito mais do que a radiação de fundo do big-bang, tempestades solares e os resquícios das bombas atômicas. Mas como já aconteceu há alguns anos, a radiação liberada no acidente de Fukushima no Japão, chegou à Califórnia em poucos dias e podê ser detectada por contadores Geiger lá instalados.

Outro projeto de sensor que pode ser adicionado à estação meteorológica é um conjunto de leds comuns e ultravioleta, funcionando como “receptores” e não emissores de luz. Com um circuito próprio e uma função a mais no firmware, podemos comparar a quantidade de energia recebida por cada conjunto de leds. Com dados observados ao longo dos meses será possível saber o estado da camada de ozônio exatamente sobre a localidade e ainda emitir avisos nos dias com maior incidência. (JUSTO, 2012 p. 64)

Com a chegada do novo Arduino Yún, com um processador mais rápido, talvez seja possível tornar a conexão com o servidor criptografada, garantido a confiabilidade dos dados e a segurança da informação. Isso também pode ser possível utilizando um Raspberry Pi, hardware de baixo custo que roda Linux, pode ser estudada as maneiras de conectar os sensores à ele ou permitir a comunicação de um Arduino através da conexão SPI, tornando o Raspberry praticamente um Ethernet Shield mais robusto.

Todo o layout do hardware e códigos do firmware da estação estão licenciados para qualquer tipo de uso, e nada impede que sejam criados e programados sensores novos para capturarem diferentes dados. O firmware também pode ser modificado para enviar os dados para um site particular.

BIBLIOGRAFIA

ARDUINO. Arduino Home <arduino.cc> Acesso em: 3 nov. 2013.

ARGENT Data Systems, Weather Sensor Assembly p/n 80422
<https://www.argentdata.com/files/80422_datasheet.pdf> Acesso em: 30 out. 2013.

BANZI, Massimo. **Getting Started with Arduino**. EUA: O'Reilly Media Inc., 2011.

BITSON, Tim. **Weather Toys: Building and Hacking Your Own 1-Wire Weather Station**. 1. ed. EUA: Wiley Publishing Inc., 2006.

BOSCH. Data Sheet BMP085 Digital Pressure Sensor
<<http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Sensors/Pressure/BST-BMP085-DS000-06.pdf>> Acesso em: 17 nov. 2013.

CPTEC. Centro de Previsão de Tempo e Estudos Climáticos
<<http://www.cptec.inpe.br/glossario.shtml>> Acesso em: 3 nov. 2013.

FOLHA, Destruição indica que ventos passaram de 117 km/h em Taquarituba (SP)
<<http://www1.folha.uol.com.br/cotidiano/2013/09/1346654-destruicao-indica-que-ventos-passaram-de-117kmh-em-taquarituba-sp.shtml>> Acesso em: 30 out. 2013.

FORREST, M. Mins III. **Science and Communication Circuits & Projects**. EUA: Master Publishing Inc. 2007.

GERTZ, Emily e JUSTO, Patrick Di. **Environmental Monitoring with Arduino**. Sebastopol, CA, EUA: O'Reilly, 2012.

HONEYWELL. Data Sheet HIH-4030/31 Humidity Sensors
<<http://www.sparkfun.com/datasheets/Sensors/Weather/SEN-09569-HIH-4030-datasheet.pdf>> Acesso em: 17 nov. 2013.

HONEYWELL. Data Sheet Honeywell HumidIcon Digital Humidity/Temperature Sensors HIH-6130/31 Series
<<http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/1443945.pdf>> Acesso em: 17 nov. 2013.

HOPE RF. Data Sheet HH10D Humidity sensor module
<<http://www.hoperf.com/upload/sensor/HH10D.pdf>> Acesso em: 17 nov. 2013.

IGOE, Tom. **Making Things Talk**. EUA: O'Reilly Media Inc., 2007.

INMET, Nota Técnica No. 001/2011 - Rede de Estações Automáticas do INMET, 2011.
<http://www.inmet.gov.br/portal/css/content/topo_iframe/pdf/Nota_Tecnica-Rede_estacoes_INMET.pdf> Acesso em: 30 out. 2013.

INMET, Consulta Dados da Estação Automática.
<http://www.inmet.gov.br/sonabra/pg_dspDadosCodigo.php?QTgyOA==> Acesso em: 30 out. 2013.

JELASTIC. Java and PHP Web Cloud Hosting <jelastic.com> Acesso em: 3 nov. 2013.

JUSTO, Patrick Di e GERTZ, Emily. **Atmospheric Monitoring with Arduino.** Sebastopol, CA, EUA: O'Reilly, 2012.

METEOROLOGIA para Agricultura. CPTEC, 2013. <<http://agricultura.cptec.inpe.br/>> Acesso em: 3 nov. 2013.

PLATT, Charles. **Make: Electronics.** 1. ed. EUA: O'Reilly Media Inc., 2009.

POSTGRESQL 9.2 Documentation <<http://www.postgresql.org/docs/9.2/static/index.html>> Acesso em: 3 nov. 2013.

PRESSURE Conversion Guide
<<http://www.sensorsone.co.uk/pressure-units-conversion.html>> Acesso em: 30 out. 2013.

RBS, Condições meteorológicas de Santa Catarina e do Rio Grande do Sul atualizadas em tempo real <<http://www.clicrbs.com.br/meteorologia/jsp/default.jsp?uf=1&local=1&action=meteorologia>> Acesso em: 30 out. 2013.

REDEMET. Rede de Meteorologia do Comando da Aeronáutica
<<http://www.redemet.aer.mil.br/#>> Acesso em: 3 nov. 2013.

SPARKFUN. Sparkfun Electronics <www.sparkfun.com> Acesso em: 3 nov. 2013.

SONNEMAKER, João Baptista. Meteorologia PP-PC-IFR-PLA. 31. ed. São Paulo, SP: ASA, 2011.

TEXAS INSTRUMENTS. Data Sheet TMP102 - Low Power Digital Temperature Sensor With SMBusTM/Two-Wire Serial Interface in SOT563
<<http://www.sparkfun.com/datasheets/Sensors/Temperature/tmp102.pdf>> Acesso em: 17 nov. 2013.

VISHAY. Data Sheet TEMT6000X01 Ambient Light Sensor
<<http://www.vishay.com/docs/81579/temt6000.pdf>> Acesso em: 17 nov. 2013.

WEATHER Forecast & Reports - Long Range & Local <<http://www.wunderground.com>> Acesso em: 3 nov. 2013.

ANEXOS

Anexo 1

Custos de peças e componentes de uma estação meteorológica na versão final (em dólares):

Argent Data System 80422 (Vento e chuva)	69,00
Arduino Ethernet	45,95
Temperatura e umidade HIH6130	29,95
Pressão BMP085	19,95
Radiação Solar (luminosidade)	5,00
Outras peças (caixas, cabos, conectores e shield)	40,00
Frete + impostos (aproximadamente)	155,00
TOTAL	US\$ 364,85

Anexo 2

Código de leitura do sensor de umidade HIH-6130:

```
// copyright, Peter H Anderson, Baltimore, MD, Nov, '11
// You may use it, but please give credit.

// modified by Saulo Matte Madalozzo (saulo.zz@gmail.com) Oct,
2013

float temperature, humidity;

float getTemperature() {
    return temperature;
}

float getHumidity() {
    return humidity;
}

void getHIH6130Data(void)
{
    unsigned int H_dat, T_dat;
    fetch_humidity_temperature(&H_dat, &T_dat);
    humidity = (float) H_dat * 6.10e-3;
    temperature = (float) T_dat * 1.007e-2 - 40.0;
}

void fetch_humidity_temperature(unsigned int *p_H_dat, unsigned
int *p_T_dat)
{
    byte address, Hum_H, Hum_L, Temp_H, Temp_L;
    unsigned int H_dat, T_dat;
    address = 0x27;;
    Wire.beginTransmission(address);
    Wire.endTransmission();
```

```
delay(100);

Wire.requestFrom((int)address, (int) 4);

Hum_H=Wire.read();
Hum_L=Wire.read();
Temp_H=Wire.read();
Temp_L=Wire.read();

Wire.endTransmission();

//_status = (Hum_H >> 6) & 0x03;

Hum_H=Hum_H & 0x3f;
H_dat=(((unsigned int)Hum_H) << 8) | Hum_L;
T_dat=(((unsigned int)Temp_H) << 8) | Temp_L;
T_dat=T_dat / 4;
*p_H_dat=H_dat;
*p_T_dat=T_dat;
//return(_status);

}
```