

Arthur Pinheiro Côrtes

Controle e Supervisão da Umidade em Estufa de Plantas
Baseado em *Arduino*



Universidade Federal de Uberlândia
Faculdade de Engenharia Mecânica

2017

Arthur Pinheiro Côrtes

CONTROLE E SUPERVISÃO DA UMIDADE EM ESTUFA DE PLANTAS BASEADO EM *ARDUINO*

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia Mecatrônica da Faculdade Federal de Uberlândia, como parte dos requisitos para conclusão do curso e obtenção do título de **bacharel em Engenharia Mecatrônica**.

Área de Concentração: Automação, Controle, Sistemas embarcados.

Orientador: José Jean-Paul Z. S. Tavares

Uberlândia – MG

2017

Agradecimentos

À Universidade Federal de Uberlândia e à Faculdade de Engenharia Mecânica pela oportunidade de realizar esse curso.

Ao meu orientador e colegas do MAPL (Laboratório de Planejamento Automático de Manufatura) pelo apoio às questões técnicas do trabalho.

À minha família pela paciência, carinho e apoio emocional.

Resumo

O presente trabalho apresenta um projeto de controle e supervisão da umidade aplicado a estufa de plantas, utilizando-se da plataforma *Arduino*. Os objetivos do trabalho são projetar um sistema capaz de fazer o controle de umidade relativa do ar na estufa, armazenar um histórico das medições de umidade e temperatura do sensor e disponibilizar uma interface que pode ser acessada remotamente para supervisionar o sistema. Um protótipo foi construído baseado no estudo de caso de uma estufa real de 48 m². O protótipo se tornou um sistema de automação com controle da umidade relativa do ar e com uma interface web para acesso local. Para se chegar ao resultado final foi usado programação em C, programação de páginas web, adição de diversos *hardwares* e aplicação de conhecimentos de eletrônica. O funcionamento do protótipo foi testado e obteve resultados relevantes para um melhor aproveitamento dos recursos da estufa.

Palavras-chave: Arduino, controle, controle *on/off*, interface web, estufa de plantas, interface homem-máquina, sistema supervisor.

Abstract

The present paper presents a Project of humidity supervision and control applied to a greenhouse, using the Arduino platform. The objectives of this paper is to project a system capable of controlling the relative air humidity in the greenhouse, track the humidity and temperature measurements from the sensor e make an interface that is remotely available to manage the system. A prototype has been built based on the case study of a real 48m² greenhouse. The prototype became an automation system with relative air humidity control and with web interface locally connected. To achieve the final results the c language was used, page web programming was used, various hardware were added and electronics knowledge was applied. The operation of the prototype was tested and achieved meaningful results for a better employment of the greenhouse resources.

Keywords: Arduino, control, on/off control, web interface, greenhouse, user interface, supervisory system.

Sumário

Capítulo I - Introdução	1
1.1. Objetivo geral:	2
1.2. Justificativa	2
1.3. Apresentação	2
Capítulo II - Fundamentação Teórica	3
2.1. Automação Industrial	3
2.2. Interface Homem-Máquina (IHM) e Sistemas Supervisório	5
2.3. Automação de estufas de plantas	8
2.4. <i>Arduino</i>	16
2.5. Interface web	19
Capítulo III - Metodologia	24
3.1. Requisitos do Projeto	24
3.2. Especificação da Solução	24
3.3. Implementação e testes	24
Capítulo IV - Estudo de Caso	25
Capítulo V - Projeto	27
5.1. Sistema físico	28
5.2. Processamento	31
5.3. Interface do Usuário	36
Capítulo VI - Resultados	41
6.1. Protótipo	41
6.2. Página web	42
6.3. Desempenho do <i>Arduino</i>	45
6.4. Ethernet Shield	47
6.5. Controle da umidade relativa do ar	49
Capítulo VII - Discussão e Conclusão	53
7.1. Trabalhos futuros:	54

8. Referências Bibliográficas	55
9. Apêndice.....	58
9.1. Controle da umidade relativa do ar, gráficos extras	58
9.2. Código do Arduino (Servidor).....	59
9.3. Código da página WEB (Cliente)	59

Capítulo I

1. Introdução

Para o cultivo das plantas, é muito importante ter as condições certas para seu crescimento, isto é, temperatura, umidade, espaço e iluminação adequados. Tendo como exemplo as orquídeas, a umidade relativa do ar é o fator mais decisivo, e também outra variável muito próxima dela, a temperatura.

A automação em estufa de plantas é muito usada comercialmente. Dependendo da estrutura e cultura da estufa, diferentes tecnologias de automação industrial são empregadas. Equipamentos como termostatos, janelas automatizadas, climatizadores e desumidificadores então aparecem, além de CLPs e IHMs para o controle deles.

Alternativas para esses equipamentos industriais robustos e caros são simples regadores de plantas, que podem, por exemplo, fazer o papel de resfriar e umidificar o ambiente. E para o controle temos como alternativa o Arduino, uma plataforma de programação acessível em preço e comodidade.

O desafio desse trabalho é adaptar as tecnologias de automação mais atuais para o controle e supervisão da umidade de orquídeas em uma estufa doméstica.

1.1. Objetivo geral:

- Projetar e implementar um sistema de controle e supervisão de umidade para estufa utilizando *Arduino*

1.1.1. Objetivos específicos:

1. Controlar a umidade relativa do ar
2. Criar uma interface para acionar e programar a irrigação
3. Monitorar remotamente umidade relativa do ar e temperatura
4. Construir um protótipo para testar o sistema
5. Obter gráfico de funcionamento para avaliar o sistema

1.2. Justificativa

O curso de engenharia mecatrônica da UFU tem como componentes curriculares matérias da engenharia mecânica, elétrica e de computação. Tal configuração permite que o engenheiro mecatrônico tenha uma visão diferente de um problema, uma visão mais ampla que trabalha com eletrônica e programação, sem perder de vista as variáveis do sistema físico.

O trabalho com estufa de plantas coloca em observação a temperatura e umidade relativa do ar, variáveis da termodinâmica. Para a atuação sobre essas variáveis são necessários componentes elétricos e eletrônicos, como uma unidade de bombeamento e relés. Além desses aspectos da engenharia mecânica e elétrica, temos a engenharia de computação ao utilizar um micro controlador programável para supervisionar o sistema.

Em suma, a automação de uma estufa de plantas permite trabalhar quase todas as qualificações de um engenheiro mecatrônico.

1.3. Apresentação

O capítulo 2 desse trabalho trata da fundamentação teórica apresentando os conceitos da automação industrial, estufa de plantas e interface *web*. O capítulo 3 apresenta a metodologia utilizada nesse trabalho. O capítulo 4 contém o estudo de caso da automação de uma estufa de plantas em que o trabalho é baseado. O projeto desenvolvido está descrito no capítulo 5. A implementação do protótipo, seus testes e resultados são apresentados no capítulo 6. Por fim, no capítulo 7 temos a discussão e conclusão.

Capítulo II

2. Fundamentação Teórica

2.1. Automação Industrial

“Uma boa definição para automação é um conjunto de técnicas destinadas a tornar automáticas a realização de tarefas, substituindo o gasto de bioenergia humana, com esforço muscular e mental, por elementos eletromecânicos computáveis. Percebe-se, portanto, que este amplo conceito se estende a diversos cenários, como, por exemplo, o robô para o operário industrial.” (Silevira, *et al.*, 2003)

Conforme visto em (Silevira, *et al.*, 2003), no contexto da automação, vemos o aparecimento recorrente de equipamentos como os CLPs (Controlador Lógico Programável), computadores industriais, relés e diversos atuadores e sensores. Parte dos avanços na automação industrial se deve à integração de todos esses componentes. Essa integração pode ser entendida pelo estudo da pirâmide da automação e das redes industriais.

2.1.1. Pirâmide da automação

A pirâmide da automação (Figura 1) tenta organizar os diferentes níveis de controle através da divisão em níveis hierárquicos. Os níveis mais baixos da pirâmide representam os equipamentos mais próximos do campo de trabalho. Os níveis mais altos representam a gerência dos processos e a gestão da automação.



Figura 1: Pirâmide da automação, retirado de: (Santos, 2012)

O texto de (Santos, 2012), nos descreve os níveis da pirâmide da seguinte forma:

Nível 1 – Aquisição de Dados e Controle Manual: O primeiro nível é majoritariamente composto por dispositivos de campo. Atuadores, sensores, transmissores e outros componentes presentes na planta compõem este nível.

Nível 2 – Controle Individual: O segundo nível compreende equipamentos que realizam o controle automatizado das atividades da planta. Aqui se encontram CLP's (Controlador Lógico Programável), SDCD's (Sistema Digital de Controle Distribuído) e relés.

Nível 3 – Controle de Célula, Supervisão e Otimização do Processo: O terceiro nível destina-se a supervisão dos processos executados por uma determinada célula de trabalho em uma planta. Na maioria dos casos, também obtém suporte de um banco de dados com todas as informações relativas ao processo.

Nível 4 – Controle Fabril Total, Produção e Programação: O quarto nível é responsável pela parte de programação e também do planejamento da produção. Auxilia tanto no controle de processos industriais quanto também na logística de suprimentos. Podemos encontrar o termo Gerenciamento da Planta para este nível.

Nível 5 – Planejamento Estratégico e Gerenciamento Corporativo: O quinto e último nível da pirâmide da automação industrial se encarrega da administração dos recursos da empresa. Neste nível encontram-se *softwares* para gestão de venda, gestão financeira e BI

(Business Intelligence) para ajudar na tomada de decisões que afetam a empresa como um todo.

É possível observar em (Silveira, 2017), A automação traz diversos benefícios:

- Produtividade – As máquinas conseguem manter um ciclo de trabalho firme e constante, e muitas vezes mais rápido do que o trabalho humano
- Qualidade – A precisão e a exatidão das máquinas mais uma vez vencem o trabalho humano
- Redução de custos – Os vários benefícios da automação normalmente pagam rapidamente o seu custo de implantação.
- Segurança – Tarefas difíceis, perigosas, insalubres e não-ergonômicas podem ser executadas por robôs.
- Monitoramento remoto e interconexão – A interconexão entre as máquinas e o seu monitoramento remoto permite melhor controle e planejamento.

Essa capacidade das máquinas de serem monitoradas remotamente e de serem interconectadas a outros dispositivos é uma das áreas que mais vem ganhando inovações. Atualmente é possível que as leituras de todos os sensores de uma máquina sejam lidas em uma única tela, é possível que o controle de várias máquinas seja feito a partir de um único computador, e é possível que o planejamento da produção seja alterado mesmo fora da planta industrial.

2.2. Interface Homem-Máquina (IHM) e Sistemas Supervisório

É comum haver alguma confusão entre o que é uma IHM e o que é um sistema supervisório. As diferenças entre os dois não são apenas conceituais. Os termos servem para identificar dispositivos parecidos, mas com funções majoritariamente distintas.

Observando a pirâmide da automação é possível encontrar (dependendo da bibliografia) a IHM no segundo nível da pirâmide, o nível operacional, já que a IHM é um dispositivo dedicado à operação de uma máquina específica. Ela também pode ser considerada como terceiro nível.

A função da IHM é permitir que a máquina seja operada por uma pessoa, dessa forma, a maioria das IHMs pode ser um painel com botões e ajustes destinados a regular o funcionamento da máquina (Figura 2). É cada vez mais comum que a IHM tenha uma tela com imagens e gráficos, destinados a auxiliar a tomada de decisão, essa seria uma das semelhanças com um sistema supervisório.

Outra característica importante das IHMs é que seus gráficos, alarmes e banco de dados ficam centralizados na tela da IHM. Além disso, a IHM fica próxima ao equipamento monitorado.

Mais propriedades foram listadas por (Rocha, 2015), e também algumas comparações com os sistemas supervisórios.

Do outro lado, os sistemas supervisórios não têm sua base de informações centralizadas em uma única tela. A base de informações pode estar em um computador *desktop* ou em um servidor. Isso permite que um supervisório seja acessado por diferentes computadores, e tenha uma gama muito maior de ferramentas ou de equipamentos supervisionados.



Figura 2: IHM comum no mercado, fabricada pela Siemens. Fonte: <http://www.maxtronick.com.br>

Além disso, os supervisórios estão no nível 3 da pirâmide da automação, e pode ser usado mais acessivelmente por gerentes ou outros usuários com menor conhecimento de determinados equipamentos.

Resumindo, normalmente um sistema supervisório pode fazer o que uma IHM faz, mas a IHM não pode fazer o que o sistema supervisório faz.

2.2.1. Sistemas Supervisórios (SCADA)

É importante notar a complexidade e interdependência dos níveis. Os sistemas industriais tendem a crescer em número, tamanho e complexidade, a supervisão e controle dos diversos sistemas e processos se torna mais difícil e são necessários componentes dos níveis superiores para a administração desses recursos.

Assim como na computação e programação, os níveis superiores abstraem os detalhes da camada inferior, e reúnem as principais variáveis e comandos em um, ou poucos, programas. Os sistemas supervisórios, localizados no terceiro nível da pirâmide, tem esse mesmo objetivo.

Os sistemas supervisórios (também chamados sistemas SCADA - *Supervisory Control and Data Acquisition*) reúnem em uma, ou em poucas telas de computador, os principais dados de todos equipamentos da célula produtiva (Figura 3). A leitura dos sensores fica disponível a qualquer momento, o comando de atuadores pode ser feito ou apenas o monitoramento. É possível também criar um banco de dados com dados do sistema, e obter estatísticas de funcionamento.



Figura 3: Exemplo de Sistema supervisório. Fonte: <http://www.professoremersonmartins.com.br>

Conforme (Silva, *et al.*, 2004), que descreve com muita exatidão os sistemas supervisórios, temos:

“Os primeiros sistemas SCADA, basicamente telemétricos, permitiam informar periodicamente o estado corrente do processo industrial, monitorando sinais representativos de medidas e estados de dispositivos, através de um painel de lâmpadas e indicadores, sem que houvesse qualquer interface aplicacional com o operador”.

“Atualmente, os sistemas de automação industrial utilizam tecnologias de computação e comunicação para automatizar a monitoração e controle dos processos industriais, efetuando coleta de dados em ambientes complexos, eventualmente dispersos geograficamente, e a respectiva apresentação de modo amigável para o operador, com recursos gráficos elaborados (IHM) e conteúdo multimídia”.

“O processo de controle e aquisição de dados se inicia nas estações remotas, PLCs e RTUs (*Remote Terminal Units*), com a leitura dos valores atuais dos dispositivos que a ele estão associados e seu respectivo controle. ”

“As estações de monitoração central são as unidades principais dos sistemas SCADA, sendo responsáveis por recolher a informação gerada pelas estações remotas e agir em conformidade com os eventos detectados, podendo ser centralizadas num único computador ou distribuídas por uma rede de computadores, de modo a permitir o compartilhamento das informações coletadas. ”

2.3. Automação de estufas de plantas

De acordo com (Perdigones, *et al.*, 2015), existem tecnologias adequadas para cada tipo de estufa. Elas podem variar, por exemplo, em seu tamanho, cultura, material e clima. Porém, todas elas têm em comum dois processos automatizáveis: controle climático e fertirrigação.

Por controle climático entende-se a busca pelas condições ideais de temperatura e umidade para o crescimento saudável das plantas. Já fertirrigação se refere à manutenção da vida das plantas, da água e dos nutrientes necessários à sua sobrevivência.

No âmbito do controle climático, a estufa pode dispor de diferentes mecanismos que atuam na temperatura e umidade: estufas semiabertas, janelas, exaustores, aquecedores, etc. Nesse sentido, a automação pode ser feita usando desde termostatos até controladores de clima. Os sensores usados nesse tipo de automação são principalmente os termômetros e os higrômetros.

No âmbito da fertirrigação, temos os sistemas de irrigação, que são compostos normalmente de: tanque de regulação (ou um poço), um sistema de distribuição, aspersores, unidades de bombeamento e unidades de fertirrigação. Nos tanques das unidades de fertirrigação são colocados nutrientes.

Nos sistemas de bombeamento podem conter válvulas de dosagem, sensores, eletroválvulas, e outros equipamentos de controle, representando a automação desse processo.

2.3.1. Controle climático

Ainda para (Perdigones, *et al.*, 2015), em estufas onde o único mecanismo são janelas automáticas (Figura 4), o controle climático se dá apenas com o uso de termostatos. A estratégia é usar um controle *on/off* com histerese ou sobreposição. Isto é, a temperatura do sistema vai oscilar em torno de dois valores de temperatura, com instruções opostas.



Figura 4: Estufa que utiliza janelas para o controle climático. Fonte: <http://www.vdh.com.br>

Por exemplo: a 25°C a instrução é abrir as janelas da estufa e deixar ar frio entrar, e a 20°C, a instrução é fechar as janelas novamente. A diferença de temperatura de 5°C entre as duas instruções impede que a janela seja aberta várias vezes num curto espaço de tempo e sobrecarregue o motor.

O sensor de temperatura é colocado no centro da estufa, à altura do crescimento da cultura.

Essa estratégia faz controle apenas da temperatura, e é comum em regiões mais frias.

Os controladores climáticos integram uma maior quantidade de sensores em relação ao controle com termostatos. Dentre esses sensores temos os de temperatura, umidade do ar, vento e chuva. Sensores de vento e chuva são usados em casos mais específicos de estufas semiabertas, ou de estufas em locais cujos ventos podem alcançar velocidade capaz de danificar a estrutura. Os sensores de temperatura e umidade relativa do ar são os mais comuns.

Sobre a umidade relativa do ar em estufas de plantas, segundo (Perdigones, *et al.*, 2015): “A umidade relativa do ar é outro fator importante na produção em estufa, por isso deve ser controlada, tanto quanto possível. Em geral, as plantas desenvolvem-se corretamente com níveis de umidade entre 60% e 80%, se a umidade relativa for superior a 90% é reduzida a transpiração e o desenvolvimento da planta é prejudicado. Com a umidade relativa perto de 100%, pode gerar-se condensação no telhado, nas plantas ou em ambos que gera pingos de água que caem sobre as folhas podendo causar doenças fúngicas. Por sua vez, um baixo nível de umidade relativa abaixo de 30%, provoca o fecho dos estomas, o pode

causar problemas e agravar o stress por calor. Por estas razões, o controlo da humidade é importante do ponto de vista da saúde da cultura. ”

Em certas regiões mais frias, como Portugal, uma estufa com sistema de aquecimento consegue controlar bem a temperatura e humidade relativa do ar. Para tanto, usa-se o aquecimento em combinação com abertura de janelas.

Os controladores climáticos são controladores comerciais preparados para o sinal dessa variedade de sensores e atuadores presentes em estufas. Normalmente não necessitam de um computador.

Uma característica importante que os controladores climáticos podem ter sobre os termostatos é o chamado “tempo astronômico”. O controlador climático consegue determinar o nascer e o pôr do Sol, e assim trabalhar com diferentes parâmetros e ações de controle durante o dia e a noite.

2.3.2. Umidade relativa do Ar

Como já foi apresentado, numa estufa, o controle climático se baseia no controle de temperatura e umidade.

Usaremos como exemplo uma estufa onde as plantas são orquídeas, e, portanto, a umidade é a variável de maior interesse (ver seção “estudo de caso”). As orquídeas possuem raízes aéreas que absorvem água contida no ar. Por esse motivo, seu crescimento e saúde está diretamente ligado à umidade do ar.

Há duas medidas para a umidade do ar: umidade relativa, e umidade absoluta.

Segundo (ETEC, 2017): “Se entende por umidade o conteúdo de água em uma substância ou material. No caso da umidade do ar, a água está misturada com o mesmo de forma homogênea no estado gasoso.

Como qualquer outra substância o ar tem um limite de absorção, este limite se denomina saturação. Abaixo do ponto de saturação (ponto de orvalho) o ar úmido não se distingue do ar seco ao simples olhar sendo absolutamente incolor e transparente. Acima do limite de saturação a quantidade de água em excesso se precipita em forma de neblina ou pequenas gotas de água (chuva). A quantidade de água que o ar absorve antes de atingir a saturação depende da temperatura e aumenta progressivamente com ela.”

A 0°C o ponto de saturação do ar é de 4,9 g/m³ e a 20°C alcança 17,3 g/m³. A umidade absoluta é a relação entre o peso da água dissolvida no ar e o peso do ar seco. Sua unidade de medida é grama por Quilograma (g/Kg).

A umidade relativa é a relação existente entre a umidade absoluta do ar e a umidade absoluta do mesmo ar no ponto de saturação a mesma temperatura (vide equações Eq. (1) e Eq.2, baseadas em (ETEC, 2017)). Se indica normalmente em %UR (umidade relativa) ou %RH (em inglês, *Relative humidity*).

$$Umidade\ relativa[\% RH] = \frac{vapor\ de\ água[g/m^3]}{vapor\ de\ água\ no\ ponto\ de\ saturação[g/m^3]} \quad (2.1)$$

$$Umidade\ absoluta\left[g/kg\right] = \frac{Peso\ da\ água\ dissolvida\ no\ ar[g]}{Peso\ do\ ar\ seco[kg]} \quad (2.2)$$

Observação: A pressão do ambiente também influencia no ponto de saturação.

Voltando às orquídeas, elas absorvem a água necessária à sua sobrevivência mais facilmente quando a umidade relativa do ar está alta. A exposição à umidade relativa baixa, pode matar essas plantas, ou fazer com que elas não cresçam e floresçam. A exposição à uma umidade relativa muito alta também é prejudicial, e pode fazer com que as raízes apodreçam. Uma estufa com umidade relativa muito alta também pode apresentar problemas de condensação, a água se condensa nas superfícies e pode gerar problemas com fungos ou respingos. Essas informações podem ser encontradas em (American Orchid Society, 2017) e (Venamy Orchids, 2002).

2.3.3. Sensores de Umidade

Na verdade, os sensores utilizados para medir a umidade possuem vários nomes, normalmente os sensores eletrônicos é que são chamados “sensores de umidade”. Os outros instrumentos utilizados para medição da umidade relativa são simplesmente chamados de higrômetros, e são classificados segundo seu princípio de funcionamento, que pode ser por saís, fios de cabelo, ou por psicrometria. Informações sobre como medir a umidade são encontradas em (Jain, 2017) e (Oblack, 2016).

Psicrômetros

Os psicrômetros de bulbo seco e de bulbo úmido (Figura 5) são usados em conjunto para medir a umidade relativa do ar. Esse tipo de higrômetro usa dois termômetros de mercúrio, um com bulbo seco, e outro com bulbo úmido. A evaporação da água no bulbo úmido faz sua temperatura cair, causando uma diferença de temperatura entre os dois termômetros.

A umidade relativa é então determinada usando uma tabela que relaciona a temperatura ambiente (bulbo seco) com a diferença de temperatura entre os dois termômetros.



Figura 5: Exemplos de psicrômetro. Fonte: <http://parumao.blogspot.com.br/2014/08/higrometro-e-psicrometro.html>

Higrômetros mecânicos

Esse sistema utiliza um material orgânico (normalmente cabelo humano) que se expande e contrai como resultado da umidade do ambiente (Figura 6). O material orgânico é mantido sobre tensão por uma pequena mola, que por sua vez está ligada à agulha que indica o nível de umidade numa escala.



Figura 6: Higrômetro mecânico. Fonte: <https://pt.aliexpress.com/item/hair-hygrometer-polymeter-hair-hygrograph-Saussure-s-hygrometer-mechanical-hygrometer/32279701023.html>

Sensores eletrônicos de umidade

Os higrômetros de sais de lítio são os principais “sensores de umidade” eletrônicos (Figura 7). Alguns polímeros e/ou cerâmicas também são utilizados como elemento sensível. Esses elementos sensíveis podem variar sua capacitância, resistência ou condutividade. Em todo o caso, essa variação é medida por um instrumento elétrico, e então convertida em sinal elétrico.

A partir do sinal elétrico do sensor, é possível associar o sensor a vários outros dispositivos elétricos e eletrônicos, ou tratar o sinal de maneira digital.



Figura 7: Sensor eletrônico de umidade, fonte: (Jain, 2017)

2.3.4. Anteparo de Stevenson (*Stevenson Screen*):

Segundo (WeatherOnline, 2017) e (Australian Bureau of Statistics, 2006), o anteparo de Stevenson (tradução livre), também conhecido por abrigo de termômetros ou abrigo de sensores, é usado principalmente em estações meteorológicas. Esse tipo de proteção é usado para evitar alguns erros de medição aos quais os sensores de temperatura e de umidade estão sujeitos ao ficarem ao ar livre.

O anteparo de Stevenson (Figura 8) protege o termômetro e o sensor de umidade de receber vento, água e radiação solar diretamente. Para tanto, esse abrigo é uma caixa de madeira pintada de branco, com persianas duplas nas laterais. O anteparo também pode ser

feito de outros materiais e ter outros formatos, mas sempre é pintado de branco para refletir a radiação solar direta.



Figura 8: *Stevenson Screen*, ou anteparo de Stevenson (abrigo para sensores). Fonte: <http://aaybee.com.au/images/StevensonScreen.jpg>

2.3.5. Controle da Umidade Relativa do Ar em estufas

Um sistema de controle é um dispositivo que controla outros, ou um dispositivo que controla uma determinada variável do sistema. Ao buscar o controle da umidade relativa do ar, estamos criando um sistema que é capaz de aumentar e diminuir a umidade de forma a manter ela em valores pré-determinados.

Como visto na seção (2.3.2. Umidade do ar), a umidade relativa do ar está relacionada com o ponto de saturação da água no ar. Então é possível alterar a umidade relativa ao se modificar a temperatura e a pressão do ambiente, além de apenas modificar a quantidade de água no ar.

Conforme (Netafim, 2017), uma das maneiras de aumentar a umidade relativa, é fornecendo água para a estufa, normalmente utilizando uma bomba hidráulica e aspersores ou nebulizadores. Os Aspersores ou nebulizadores expõem a água na forma de mini partículas, causando uma evaporação mais rápida. Com a evaporação dessas partículas, temos não apenas o aumento da quantidade absoluta de água no ar, como também tempo o seu resfriamento, por um processo chamado “resfriamento adiabático”.

Para diminuir a umidade podem ser usados exaustores. Um exaustor colocado no alto da estufa, expõe o ar quente que sobe e causa a entrada de ar externo pelas janelas da estufa. Se o ar externo tiver umidade relativa menor q o ar interno, essa estratégia funciona. Ao

favorecer a troca de ar com o ambiente externo, o exaustor, para uma estufa de plantas, também exerce a função de renovar o oxigênio para as plantas.

Também existem equipamentos capazes de remover a umidade do ar, os chamados desumidificadores, como mostra a Figura 9. Esses são equipamentos que possuem uma bomba de calor para resfriar o ar, até que a água contida nele se condense, e então aquecer o ar de volta à temperatura do ambiente.



Figura 9: Como funciona um desumidificador. Fonte: <http://www.theairgeeks.com/dehumidifier/>

2.3.6. Controle *on/off*

Esse tipo de controle possui também alguns outros nomes: bang-bang, controlador com histerese ou controlador de dois passos. Todos esses nomes designam um controlador de malha fechada que muda abruptamente entre dois estados.

Esse controlador é comumente utilizado para plantas que possuem uma entrada binária, como um forno, que ou está completamente ligado ou completamente desligado (Figura 10).

Normalmente, esse tipo de controle é guiado por um “*Setpoint*”, e por um valor de tolerância, ou histerese. O “*Setpoint*” é o valor ótimo para a variável controlada.

É comum que o “*Setpoint*” exija que o atuador não trabalhe em sua potência máxima, e mantenha o sistema com o valor desejado, sem exceder ou faltar. Mas, isso não é possível para os atuadores de um controle *on/off*, como o forno do exemplo anterior. Dessa forma é adicionado o valor de tolerância, para que o atuador não fique preso num loop “liga – desliga”.

Um exemplo de uso de controle on/off foi dado na seção 2.3.1 - controle climático:

“A 25°C a instrução é abrir as janelas da estufa e deixar ar frio entrar, e a 20°C , a instrução é fechar as janelas novamente. A diferença de temperatura de 5°C entre as duas instruções impede que a janela seja aberta várias vezes num curto espaço de tempo e sobrecarregue o motor.”

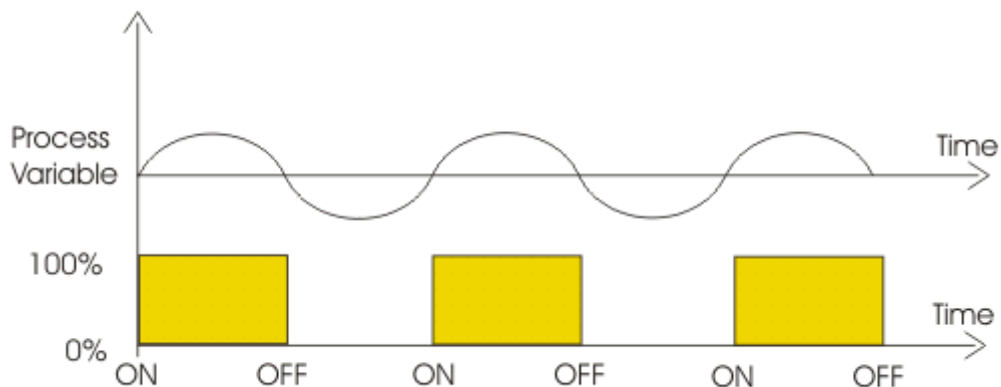


Figura 10: Comportamento de um sistema com controlador on/off, Fonte: <https://www.electrical4u.com/on-off-control-theory-controller/>

2.4. *Arduino*

Avanços nas áreas da computação e da eletrônica possibilitaram a criação de plataformas de desenvolvimento baratas e versáteis que podem atender os mais diversos públicos. Plataformas como o *Arduino* e o *RaspberryPI* permitem a criação de projetos de eletrônica e automação em relativamente pouco tempo, com custo reduzido e pouca programação.

Embora essas plataformas tenham sido produzidas com finalidades didáticas, engenheiros, técnicos e hobbystas passaram a usá-las para os mais diversos fins e ambientes. Foi gerada uma demanda por melhorias, e hoje temos muito material sobre o assunto e novas tecnologias desenvolvidas para projetos mais ambiciosos. Uma placa *Arduino* pode ser vista na Figura 11.

A plataforma *Arduino* foi uma das plataformas mais inovadoras e de mais sucesso. Hoje possui dezenas de modelos disponíveis, e uma grande quantidade de *software* e *hardware* já pode ser usada em conjunto.

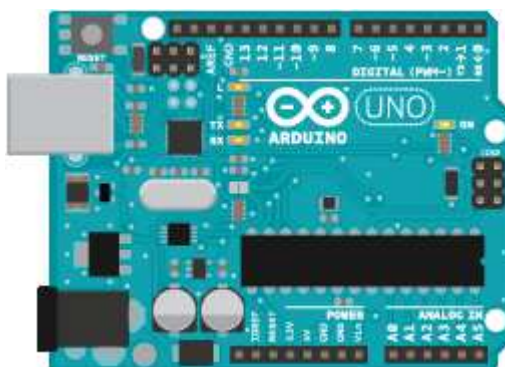


Figura 11: Imagem ilustrativa de uma placa Arduino. Fonte: arduino.cc

O *Arduino*, que a princípio se conectava a sensores e atuadores via circuitos eletrônicos e programação, agora tem uma gama de funcionalidades, como conexão à internet. Além disso, pode funcionar em conjunto com *softwares* de interface gráfica, e criar sistemas supervisórios para suas aplicações.

2.4.1.Características

Na página oficial (Arduino Official Site, 2017) podemos encontrar todas as informações sobre a plataforma. A placa possui um processador capaz de muitas funções diferentes, como: pinos com múltiplas funções, saídas e entradas digitais, PWM, conversor Analógico-Digital, comunicação *Serial*, comunicação I²C, comunicação SPI, *timers*, interrupções, etc.

Além disso, a placa já possui entradas para a alimentação desse processador via *jack* ou USB. Um outro chip presente na placa permite a comunicação do processador do *Arduino* com computadores via USB. Outras partes importantes da placa são mostradas na Figura 12.

A comunicação USB com o computador e um *bootloader* vindo de fábrica permitem que programas sejam gravados no processador sem *hardwares* adicionais.

A plataforma também disponibiliza uma IDE (Ambiente de desenvolvimento integrado) gratuita na internet para a criação dos programas. A IDE conta com editor de linhas de código, administrador de bibliotecas e um Monitor *Serial*, entre outros.

A programação do *Arduino* é feita em uma linguagem própria, baseada na linguagem C. A linguagem é de certa forma simples, possui várias funções próprias e bibliotecas padrão que facilitam a programação de iniciantes ou que agilizam o teste do *software*. Suas bibliotecas podem ser escritas na linguagem própria, em C, ou C++.

A plataforma *Arduino* é *open source*, o que significa que várias pessoas contribuíram gratuitamente para o seu desenvolvimento, e vários projetos e ferramentas adicionais podem ser encontradas na internet.

Principais especificações:

- Tensão de operação: 5 V
- Pinos E/S (digital): 14 (6 deles com PWM)
- Pinos E/S Analógicos: 6
- Corrente DC nos pinos E/S: 20 mA
- *Clock*: 16 MHz

Arduino Uno Rev 3

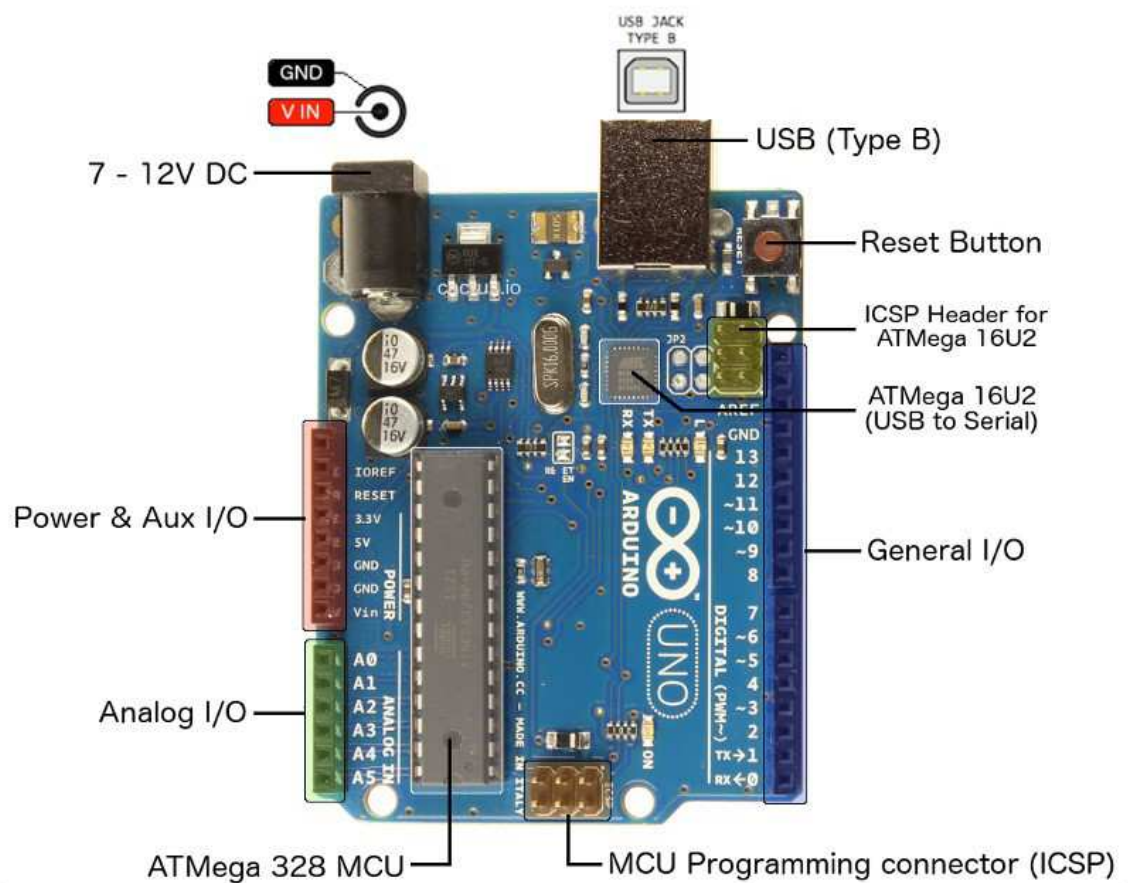


Figura 12: Partes de um Arduino UNO. Fonte: <http://cactus.io/platform/arduino/arduino-uno>

A tensão de operação de 5V é facilmente encontrada em dispositivos que utilizam conectores padrão USB.

A quantidade de pinos digitais é suficiente para a conexão de diversos periféricos, que normalmente não utilizam mais de 4 pinos. Os pinos analógicos e o PWM permitem trabalhar com variáveis analógicas.

O *Clock* de 16 MHz, apesar de não ser um dos melhores encontrados no mercado, na mesma faixa de preço, supre a maioria das aplicações.

2.4.2. Módulos e Shields do Arduino (Ethernet Shield)

Outro recurso que ajudou a popularizar o *Arduino* foi a montagem da sua placa, com bornes e pinos em posições bem definidas. Esses bornes e pinos são usados para encaixe direto de *hardware* adicional à placa principal (Figura 13).

Esses *hardwares* adicionais são comumente chamados de “módulos para *Arduino*” ou *Shields*. Como exemplo podemos citar o *Ethernet Shield*. Com esse *Shield*, o *Arduino* é capaz de se conectar a redes domésticas ou a internet, utilizando um cabo *Ethernet* e o protocolo TCP/IP. Além disso, ele possui um slot para inserção de cartão SD.

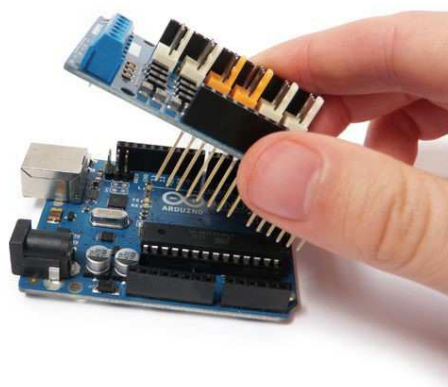


Figura 13: O acoplamento de outra placa aumenta as funções do Arduino UNO. Fonte: <http://www.instructables.com/id/Elevetor-Model-Design-and-Control-Using-Arduino/>

Com esse *Shield*, o *Arduino* torna-se capaz de se conectar à internet, ser utilizado como servidor interno ou web, pode armazenar dados e abrir ou editar arquivos de texto.

2.5. Interface web

A interface é formada por três componentes principais: sua estrutura, seu visual e seu comportamento. Esses três componentes são desenvolvidos com as respectivas ferramentas: HTML, CSS e Javascript. (Figura 14)

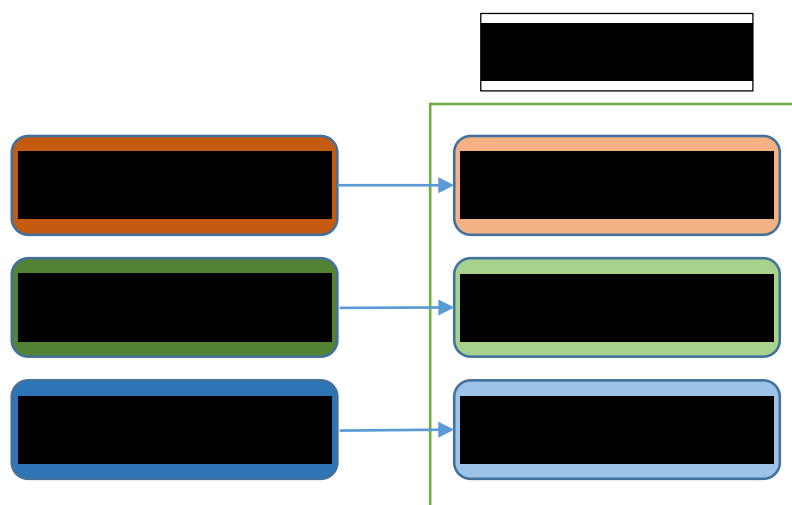


Figura 14: Componentes de uma página web. Fonte: <http://bulbulcse.com/2016/06/02/html-vs-css-2/>

Existem muitas outras ferramentas no mundo do desenvolvimento web, mas essas três são as mais comuns.

2.5.1.HTML

HTML significa “*Hypertext Mark-up Language*”. Embora seja considerada uma linguagem de programação, na verdade ela é apenas uma linguagem interpretada, ou seja, você escreve seu texto de forma que o computador vá conseguir reproduzi-lo da maneira desejada. O *JavaScript* sim é uma linguagem de programação, e é usado em conjunto com o HTML.

O HTML é a linguagem dos navegadores de internet. Um texto escrito em HTML são apenas caracteres ordenados, o navegador é capaz de interpretar esse texto escrito em HTML e escolher um tamanho para a letra, uma fonte, adicionar cores, imagens, formatação, etc. Uma página simples em HTML pode ser vista na Figura 15.

A internet funciona à base de HTML. As páginas web criadas são decompostas em caracteres, os caracteres são enviados pela internet, e os navegadores são capazes de reconstruir as páginas para uso.

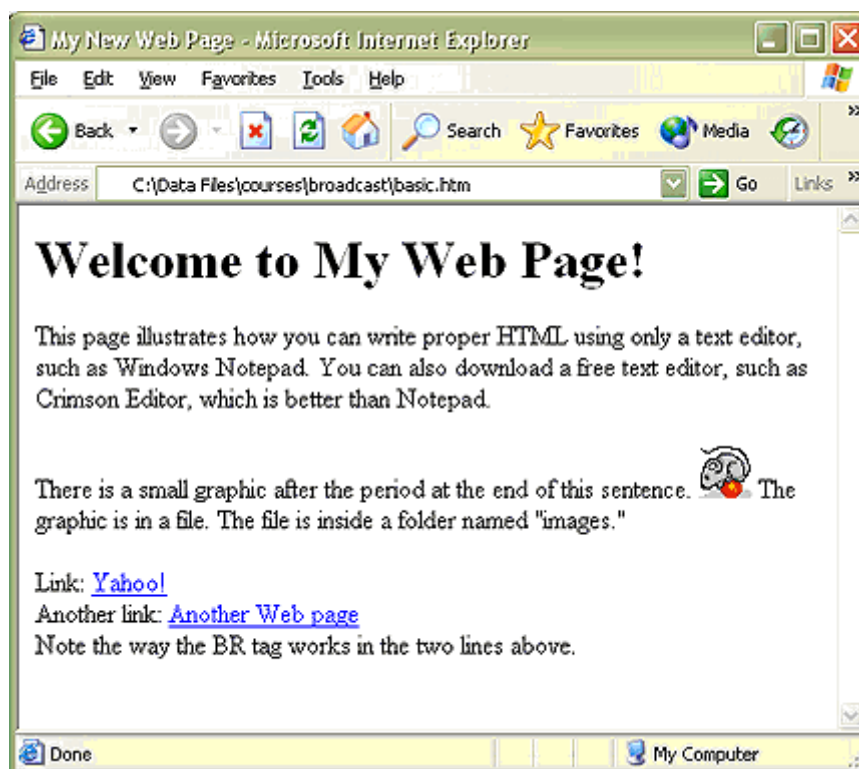


Figura 15: Exemplo de página feita com HTML. Fonte: <http://www.macloo.com/examples/html/basic.htm>

2.5.2. JavaScript

O *JavaScript* é uma linguagem de programação que, segundo (Chapman, 2017), serve para tornar as páginas web interativas. *JavaScript* é uma linguagem de programação baseada em texto, desenvolvida para aplicações web. Seus códigos não precisam ser compilados, são apenas interpretados pelo browser.

JavaScript e HTML são coisas diferentes, mas são complementares. O HTML é uma linguagem de marcação e dá à página web sua estrutura básica, e é estático. O *JavaScript* foi feito para rodar dentro do HTML, e é responsável pelo conteúdo dinâmico da página, como caixas de pesquisa ou animações.

Resumindo: O HTML define como a página deve ser exibida, mas qualquer mudança de comportamento que deva acontecer na página acontece com o auxílio do *JavaScript*. Por exemplo, menus expansíveis e atualização de setores da página ao invés de reenviar a página inteira.

2.5.3. AJAX

Existe um conjunto de funções do JavaScript que, usadas em conjunto, são chamadas de AJAX. (Smith, 2013) ensina o que é e como utilizar o AJAX.

AJAX significa, em inglês, “Asynchronous JavaScript and XML”. Esse conjunto de funções é utilizado para facilitar a troca de informações entre cliente e servidor, mais diretamente, ele permite que as informações da página web sejam atualizadas sem que a página inteira tenha que ser atualizada automaticamente.



Figura 16: Diagrama de funcionamento do AJAX. Fonte: https://www.123rf.com/photo_39827419_stock-vector-ajax-asynchronous-javascript-and-xml-programming-xml-server.html

2.5.4.XML

Segundo (Pereira, 2009), XML significa “Extensible Markup Language”, e é uma linguagem e de marcação para criação de documentos com dados organizados.

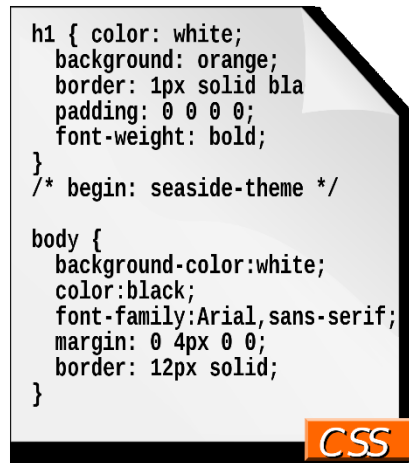
(Smith, 2013) afirma ser mais simples o recebimento de dados do servidor via XML, pois os dados são mais fáceis de serem extraídos com o JavaScript.

2.5.5.CSS

Para (Smith, 2013), CSS, ou *Cascading Style Sheets*, controla a aparência do conteúdo da web. O CSS age sobre as *tags* do HTML, alterando os atributos do texto. Ele pode alterar principalmente a fonte do texto, as cores e a posição dos objetos na página.

Em vez de colocar a formatação dentro do documento, o programador pode criar um documento contendo CSS, e criar um link na página web para esse documento. Assim, o programador pode alterar a aparência de várias páginas web apenas modificando esse arquivo.

No exemplo da Figura 17 temos um documento CSS que especifica como devem ser estilizadas as *tags* “h1” e “body” de um documento HTML. Ou seja, ao escrever o HTML e se usar a *tag* “h1”, o texto terá cor branca e fundo laranja, por exemplo.



```
h1 { color: white;
background: orange;
border: 1px solid bla
padding: 0 0 0 0;
font-weight: bold;
}
/* begin: seaside-theme */

body {
background-color:white;
color:black;
font-family:Arial,sans-serif;
margin: 0 4px 0 0;
border: 12px solid;
}
```

Figura 17: Exemplo de código CSS. Fonte: https://en.wikipedia.org/wiki/Cascading_Style_Sheets

Capítulo III

3. Metodologia

A metodologia adotada foi a do projeto clássico, o qual separa as atividades do projeto em levantamento de requisitos, especificação da solução, implementação e testes.

3.1. Requisitos do Projeto

Os principais requisitos a serem atendidos pelo projeto de controle da estufa de plantas são:

- Baixo custo (não utiliza soluções prontas de mercado ou equipamentos industriais);
- Controle da umidade;
- Supervisão remota da umidade (com IHM);
- Armazenagem do histórico dos dados do processo de controle da umidade.

Maiores detalhes do estudo de caso podem ser vistos no capítulo 4.

3.2. Especificação da Solução

O capítulo 5 – Projeto, especifica a solução para o problema de automação apresentado, baseado nos requisitos.

3.3. Implementação e testes

A construção do protótipo e seus resultados são apresentados no capítulo 6 – Resultados.

Capítulo IV

4. Estudo de Caso

A estufa de plantas em que se baseia o trabalho está na cidade de Patrocínio (MG). Ela possui as plantas dispostas sobre mesas ou penduradas, e estão fixas em vasos ou amarradas à madeira. A maioria das plantas são orquídeas.

Na Figura 18 temos uma foto da estufa, e na Figura 19 temos um esboço dos seus principais elementos. Ela possui uma área de 48m² (4x12m), e é coberta por plástico e sombrite. O sombrite reduz parte da iluminação, e o plástico isola as plantas do ambiente externo, conservando calor e umidade. O topo da estufa possui uma pequena fresta que permite a passagem de ar. Essa passagem de ar serve para proteger a estufa de ventos, e também funciona como exaustor. O ar que sai pelo exaustor permite a entrada de novos ares pelas aberturas laterais da estufa.

Do lado esquerdo do esboço, separadas da estufa por uma parede, encontram-se as ligações elétricas da iluminação e da irrigação. A bomba da estufa é ligada através de um timer digital, que aciona periodicamente um contator, e que, por fim, aciona a bomba.

O sistema de irrigação começa com o barril de água, que se enche como uma caixa de água, ou seja, até o limite da boia. A bomba possui 0,5 CV de potência e bombeia essa água para os irrigadores. Os irrigadores se encontram conectados a uma mangueira, que percorre o topo da estufa. A princípio, a função dos irrigadores é apenas regar as plantas.

A maior parte das plantas da estufa são orquídeas, o que significa que a umidade relativa do ar deve ser alta e a iluminação deve ser amena (simulando uma floresta tropical, ambiente nativo da maioria das orquídeas). É por essa necessidade das orquídeas que a estufa será beneficiada pelo sistema de controle de umidade.



Figura 18: Foto da estufa de plantas

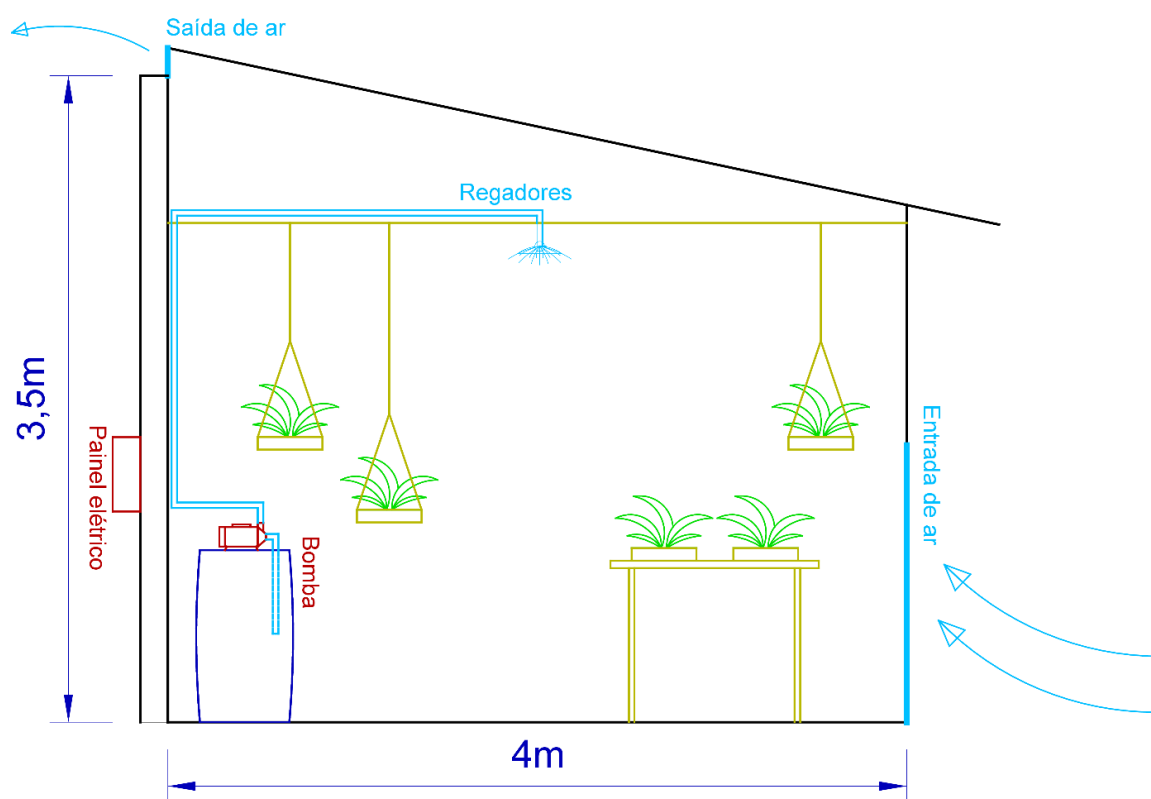


Figura 19: Esboço da estufa. Além das dimensões mostradas no desenho, a estufa possui 12 metros de profundidade

Capítulo V

5. Projeto

O projeto do protótipo de controle e supervisão da umidade será explicado em três partes para melhor entendimento: sistema físico (a estufa, sensores e atuadores), parte de processamento (*Arduino* e outros *Hardwares*), e parte do usuário (página web ou interface).

O diagrama de blocos da Figura 20 resume o programa a ser implementado, seus componentes, e suas interações.

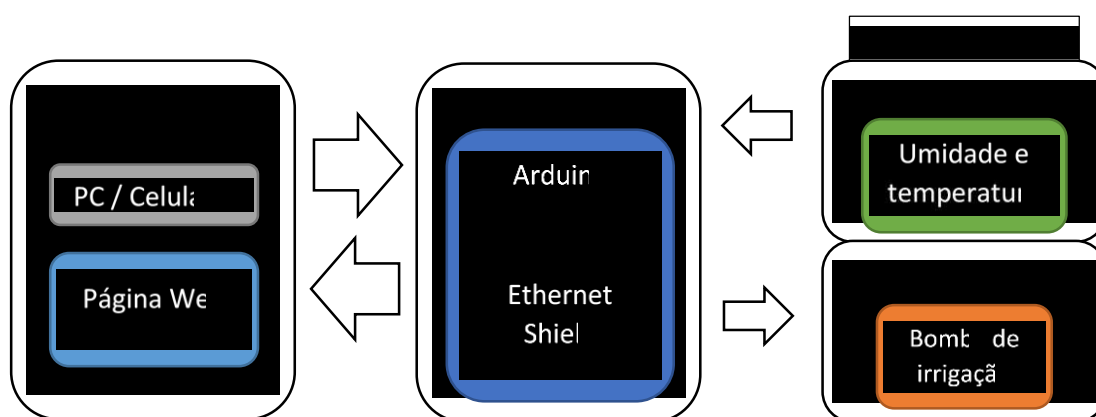


Figura 20: Visão geral do programa a ser implementado.

No diagrama é possível notar que o sistema é alimentado pela leitura de umidade e temperatura dos sensores. O *Arduino* é o computador, e é encarregado de trabalhar a informação e decidir quando a bomba deve ser ligada. O *Arduino* também deve dispor as informações do sistema para o usuário através da página web, que pode ser acessada via rede graças ao *Ethernet Shield* do *Arduino*. Também é possível armazenar as leituras do sensor em um cartão SD. O usuário pode acompanhar o sistema (monitoramento), e também pode operar de forma manual.

5.1. Sistema físico

Seguindo o objetivo principal do projeto, a estufa (ver seção: Estudo de caso) foi analisada à procura de alterações para a implementação de um sistema de controle e supervisão de umidade.

Para a construção do protótipo, o projeto buscou apenas um número mínimo de sensores e atuadores. Foram descartados os sensores e atuadores que não se alinhavam ao objetivo, ou que custariam muito dinheiro ou tempo.

5.1.1. Atuadores:

O único atuador usado no projeto será o sistema de irrigação, composto pela bomba de 0,5 CV e a mangueira com irrigadores.

O ideal para o controle da umidade seria ter toda a estrutura da estufa coberta, e adicionar janelas automatizadas e/ou exaustores. A troca de ar com o ambiente externo é fundamental para a qualidade do ar para as plantas, mas também serve para regular a temperatura e a umidade, dependendo das condições do ambiente.

Aquecedores e desumidificadores permitem um grande controle climático da estufa, mas ultrapassam o porte do projeto.

Para uma melhor leitura dos sensores, seria útil que o ar dentro da estufa fosse uniforme, ou seja, todos os pontos teriam mesma temperatura e umidade relativa. Para isso, o ambiente totalmente fechado e ventiladores contribuiriam.

5.1.2. Sensores:

Para esse projeto, todos os objetivos dependem da umidade relativa do ar. Então é indispensável a presença de um sensor de umidade. Como a medição da umidade relativa está diretamente relacionada à temperatura do ambiente, temos que a grande maioria dos sensores de umidade digitais possuem também medição de temperatura. Dessa forma, um único sensor é capaz de suprir as necessidades desse projeto.

5.1.3. Escolha do Sensor

Os sensores de umidade (higrômetros) pesquisados encontram-se na Tabela 1. A maioria deles possui medição de temperatura em conjunto, já que a umidade relativa está diretamente relacionada a essa variável.

O sensor SHT10 na verdade é um microchip, e é a base de outros sensores. Sua utilização necessitaria da montagem de uma placa para soldá-lo, e ele é um componente bem pequeno. Utilizá-lo significaria mais tempo gasto para confecção do sensor, e aumentariam as

chances de erros de medição, decorridos de uma montagem malfeita. Além disso, não é fácil encontra-lo no Brasil. Provavelmente teria que ser importado.

O PCmini70 já vem em um encapsulamento bem protegido, pronto para ser colocado em campo. É vendido em lojas de equipamento industrial, e o preço é conseguido somente pedindo-se orçamento. A saída dele é em nível de tensão, o que significa que um tratamento de sinal adicional pode ser necessário.

O HMP155 é um sensor para estação meteorológica. Ele é mais robusto e muito maior que os outros. Sua saída é por comunicação RS-485 ou também tensão. Além desses inconvenientes, sua tensão de operação é maior que a do *Arduino*.

Tabela 1: Comparativo dos Sensores de Umidade

	DHT11	DHT22	SHT10	PCmini70	HMP155
Intervalo de medição	20 a 90% RH	0 a 100% RH	0 a 100% RH	0 a 100 %RH	0 a 100 %RH
Resolução	1 %RH	± 0.1 %RH	0.05 %RH	1 %RH	0.1 %RH
Precisão	± 1 %RH	± 1 %RH	± 0.1 %RH	-	-
Exatidão	± 5 %RH	± 2 %RH	± 4.5 %RH	± 2 %RH	± 1.0 %RH
Tensão de operação	3.5 a 5.5V	3.3 a 6V	2.4 a 5.5V	4.75 a 5.25V	7 a 28V
Interface	<i>Serial</i> (1-wire)	<i>Serial</i> (1-wire)	<i>Serial</i> (2-wire)	Tensão ou frequência	RS485, ou tensão

Para o uso em conjunto com o *Arduino*, os outros dois sensores da tabela são os melhores: DHT11 e DHT22. Essa classe de sensores também pode medir umidade relativa do ar e temperatura como os outros. Além disso, eles já contêm um microprocessador que permite uma saída digital de nível 5V, tensão que o *Arduino* também opera.

Por questão de custo benefício, o sensor escolhido foi o DHT22. A seguir, suas especificações em detalhes, retiradas de (Liu):

- Medição de umidade relativa do ar:
 - Resolução: 0.1% RH
 - Precisão: ± 1% RH
 - Exatidão: ±2%, a 25°C
 - Intervalo de medição: de 0% RH até 100% RH, a 25°C

- Tempo de resposta: 2 segundos
- Medição de temperatura:
 - Resolução: 0.1°C
 - Precisão: $\pm 0.2^\circ\text{C}$
 - Exatidão: $\pm 0.5^\circ\text{C}$
 - Intervalo de medição: de -40 a 80°C
 - Tempo de resposta: 2 segundos

Comparando-se as especificações do DHT22 com o DHT11, é possível perceber uma grande vantagem de um sobre o outro. A resolução de umidade relativa do ar do DHT11 não possui casas decimais e a exatidão é baixa. O intervalo de medição também é menor.

Na verdade, para uma estufa doméstica, as especificações do DHT11 são suficientes. Mas os dois podem ser obtidos facilmente no Brasil pela internet, e a diferença de preço é justificável pelas especificações. (DHT11: R\$ 10,90, e DHT22: R\$ 37,90. Fonte: (usinainfo.com.br)

A temperatura é uma variável de menor interesse, e ambos os sensores tem especificações adequadas para serem utilizados em uma estufa doméstica.

A obtenção das medidas do sensor é feita por uma interface de comunicação *Serial*, de protocolo próprio. Os detalhes dessa comunicação são explicados em detalhe no *datasheet*. No presente trabalho, a comunicação foi intermediada por uma biblioteca própria para *Arduino*. (Ver apêndice)

5.1.4. Circuito elétrico

Para o acionamento do motor que faz a irrigação é usado um contator. O contator funciona como interruptor para uma carga de potência (motor), e permite acionamento por comandos elétricos. Os contadores possuem um circuito de comando e um circuito de carga. O contator utilizado na estufa (Figura 21) possui ambos os circuitos trabalhando a 127V. Por isso, para que o comando, vindo do Arduino, possa acionar o contator, será utilizado um relé. O acionamento manual (presencial) da bomba é feito por um interruptor comum, instalado antes do contator.

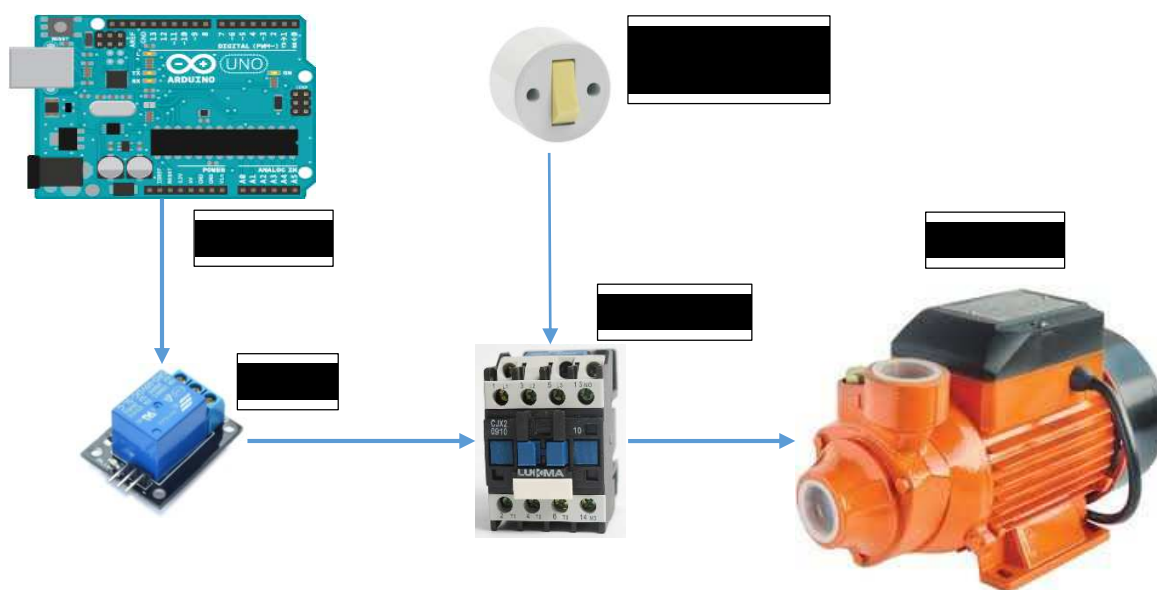


Figura 21: Circuito de acionamento da bomba.

5.2. Processamento

A arquitetura eletrônica preparada para o projeto possui vários componentes, dentre eles: O *Arduino*, um *Ethernet Shield*, um relé, um relógio RTC e um sensor de umidade. O contator e a bomba também fazem parte, mas são componentes elétricos, ligados em corrente alternada.

A Figura 22 mostra a montagem desses componentes.

5.2.1. Circuito da rede elétrica

A rede elétrica da casa é o que dá energia para todo o protótipo. A tensão de 127 V é utilizada para a irrigação das plantas com a bomba. Uma fonte chaveada de 12V também é alimentada pela rede elétrica, e liga o *Arduino*. O *Arduino*, por sua vez, possui um regulador de tensão de 5V e alimenta o restante dos componentes.

O acionamento do restante dos componentes está explicado na seção (5.1.4 Circuito elétrico).

5.2.2. Relógio RTC

A sigla RTC significa *Real Time Clock*. Esse relógio se diferencia do *clock* do processador porque serve realmente para marcar as horas do dia.

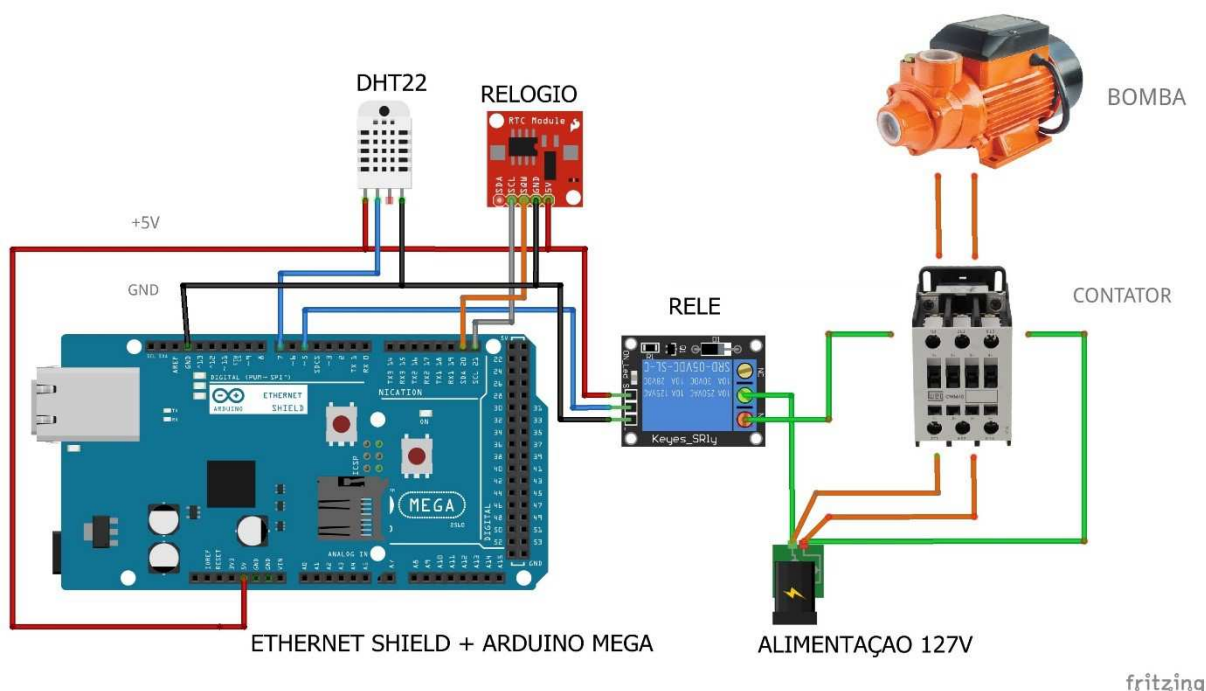


Figura 22: Diagrama esquemático do protótipo

A utilidade dele no protótipo é a de contar tempos muito maiores do que os micro controladores normalmente trabalham, como horas e dias. Com esse relógio é possível programar o *Arduino* para ligar a bomba todo dia no mesmo horário.

O modelo de relógio utilizado é o DS1307 (Figura 23). Este modelo é capaz de marcar horas, minutos e segundos. Ele também armazena a data e o dia da semana. Possui 56kB de memória FLASH não volátil para uso, se necessário.

Ele possui uma bateria de Lítio de 3,7V que consegue preservar os dados mesmo sem alimentação externa.

A comunicação do relógio com o *Arduino* se dá via interface I²C, utilizando dois pinos apenas.

5.2.3. Ethernet Shield

O *Ethernet Shield* (Figura 24) está presente no projeto por três motivos:

- Conectar o *Arduino* à rede local ou à Internet
- Permitir que a IHM do protótipo seja uma página web armazenada em cartão SD
- Permitir que os dados coletados pelo sensor sejam armazenados no cartão SD

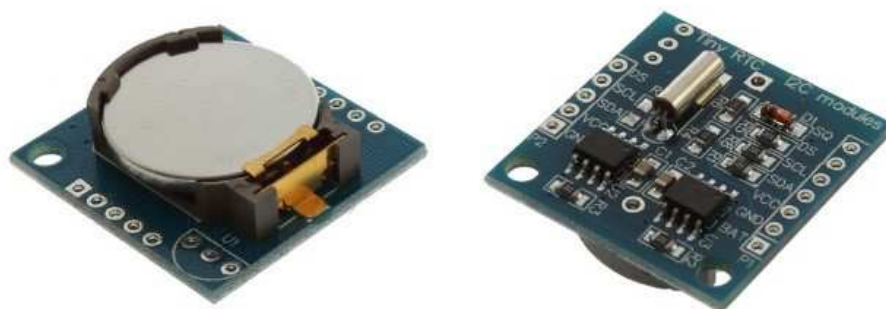


Figura 23: Relógio digital RTC DS1307. Fonte: <https://www.filipeflop.com/blog/relogio-rtc-ds1307-arduino/>

O *Ethernet Shield* é encaixado sobre o *Arduino*, e utiliza a comunicação SPI para trabalhar tanto com a conexão *Ethernet*, quanto com a comunicação do cartão SD. Além disso, ele é compatível com *Arduino* UNO e MEGA.

O *Ethernet shield*, permite o *Arduino* se conectar a uma rede local via *http*. Usando a *http*, ele pode receber e enviar mensagens de texto para um computador ou celular, utilizando um browser de internet. Dessa maneira, o *Ethernet Shield* permite o monitoramento remoto do sistema.

A resposta de texto do *Arduino* para o navegador, pode ser um texto HTML, que será interpretado pelo navegador como um site qualquer. Para o usuário, acessar o sistema de supervisão e monitoramento da estufa será como acessar qualquer site. Com essa página web, teremos a interface do usuário.



Figura 24: Ethernet Shield para Arduino. Fonte: <https://www.banggood.com/Ethernet-Shield-Module-W5100-Micro-SD-Card-Slot-For-Arduino-UNO-MEGA-p-908461.html>

O cartão SD será usado não apenas para armazenar a página web em HTML, mas também será responsável por guardar as informações coletadas pelo sensor. Ao inserir o cartão SD em um computador, o usuário terá acesso aos dados coletados, e será possível acompanhar a umidade da estufa ao longo do dia, ou de um período bem maior de tempo.

Com acesso remoto, interface para usuário e armazenamento dos dados, o sistema já será capaz de supervisionar a estufa como um sistema supervisorio.

5.2.4. Arduino

O *Arduino* é o coração do protótipo. É nele que está o micro controlador, ou seja, é lá que será guardada a lógica da automação. O modelo usado no projeto foi o *Arduino* MEGA. O *Arduino* é alimentado por uma fonte 12V, já que possui regulador de tensão para 5V. Essa alimentação também é usada para os outros componentes que trabalham com 5V.

Os objetivos de automação do projeto podem ser resolvidos utilizando-se um micro controlador. A escolha da plataforma *Arduino* se deu pela sua grande variedade de funções e velocidade de programação para protótipo.

As funções presentes no *Arduino* que foram utilizadas foram: comunicação I²C (para comunicação com o relógio RTC), saída digital (para acionamento da bomba), uma entrada digital (para comunicação *Serial* com o sensor) e comunicação SPI (para utilizar cartão SD e se conectar à uma rede).

Para o protótipo desse trabalho, O *Arduino* UNO e o *Arduino* MEGA seriam adequados, ambos possuem as funções descritas. Os dois modelos são os mais utilizados e foram testados. Podemos comparar suas especificações na Tabela 2.

Tabela 2: Comparativo *Arduino* UNO e MEGA

	<i>Arduino</i> UNO	<i>Arduino</i> MEGA
Tensão de operação	5V (fonte 7 a 12V)	5V (fonte 7 a 12V)
Portas Digitais	14	54
Portas PWM	6	15
Portas analógicas	6	16
Clock	16MHz	16MHz
Memória FLASH	32kB	256kB
Memória SRAM	2kB	8kB
Memória EEPROM	1kB	4kB
Corrente máxima portas E/S	40mA	40mA

Arduino UNO. Ele tem um processador relativamente lento, mas com muitas funções. Então o *Arduino* UNO, a princípio, pode fazer quase tudo que um microprocessador pode fazer.

O *Arduino* MEGA, possui todas essas mesmas funções, porém em maior número. Ele possui mais pinos, mais entradas e saídas, possui mais de um barramento para comunicação *Serial*, etc.

5.3. Estimativa de custo

A Tabela 3 apresenta uma estimativa de custo do projeto, baseado nos componentes listados e seus preços no ano de 2017. Os componentes foram escolhidos levando em consideração o requisito do baixo custo.

Tabela 3: Estimativa de custo do projeto.

Produto	Preço	Loja
Arduino MEGA com cabo USB	R\$ 58,49	Mercado Livre
Sensor de temperatura e umidade DHT22	R\$ 15,00	Mercado Livre
Relógio RTC para Arduino DS1307	R\$ 6,00	Mercado Livre
Shield Ethernet W5100 Arduino	R\$ 32,49	Mercado Livre
Cartão de memória 8GB micro SD	R\$ 21,90	Mercado Livre
Módulo Relé 5V 2 Canais	R\$ 12,90	FilipeFlop
Fonte DC Chaveada 12V 2A Plug P4	R\$ 24,90	FilipeFlop
Caixa para Montagem Plástica 300x200x130	R\$ 82,42	Loja elétrica
Conector Multiplo Borne (sindal) Steck 6mm	R\$ 4,90	Mercado Livre
TOTAL	R\$ 259,00	

Para fins de comparação, a Tabela 4 apresenta itens de alto custo de poderiam ser incluídos no projeto, caso não houvesse o requisito de baixo custo.

Tabela 4: Itens de alto custo, para comparação com os itens escolhidos para o projeto.

Produto	Preço	Loja
Desumidificador Desidrat Plus I 150m³ Thermomatic	R\$ 1.940,00	Submarino
Kit Logo Siemens, Cabo Logo, Cpu Oba6, Software (6 entradas e 4 saídas)	R\$ 720,00	Mercado livre
IHM Siemens Simatic Panel	R\$ 1.000,00	Mercado livre
Termostato Triac + Umidostato + Temporizador Coel Y39uhqrr	R\$ 329,00	Mercado livre

5.4. Interface do Usuário

Como explicado na seção *Ethernet Shield*, a interface do usuário é uma página web escrita em HTML, armazenada num cartão SD, enviada ao navegador do usuário pela conexão do *Ethernet Shield* e gerenciada pelo *Arduino*. Esse modelo cumpre com os objetivos do projeto de monitorar remotamente a estufa, permitir ao usuário manipular o sistema e armazenar leitura do sensor e outros.

Essa comunicação entre usuário e sistema será baseada no modelo cliente-servidor, onde o *Arduino* com *Ethernet Shield* será o servidor, e o usuário, acessando pelo computador ou celular, será o cliente.

5.4.1. Servidor

O servidor do protótipo será o *Arduino*. Ele será responsável por, além de controlar os sensores e atuadores da estufa, disponibilizar o acesso do usuário ao funcionamento do sistema.

O *software* armazenado na memória do *Arduino* tem duas funções importantes: primeiro, manipular os sensores, atuadores e módulos conectados ao *Arduino* de acordo com a lógica da automação; e segundo, responder às requisições de possíveis clientes, disponibilizando a interface e os dados do sistema. As funções do cliente estarão no cartão SD, escritas num documento HTML.

Resumindo, o *software* do *Arduino* é o “lado servidor”, e a página web em HTML é o “lado cliente”. Na Figura 26, temos um fluxograma mostrando como trabalha o “lado servidor”. Durante toda execução do *software*, o servidor estará cuidando de suas duas tarefas simultaneamente, a automação e a comunicação com o cliente.

Após iniciar a automação, o servidor fica no aguardo de uma conexão *http* vinda de algum cliente. O protocolo *http* é baseado em mensagens de texto, ou em *Strings*, como o *Arduino* interpretará. Ao receber a requisição de um cliente, o *Arduino* responderá com o texto em HTML contido no cartão SD.

No próximo momento, o *Arduino* entrará num ciclo, onde a cada segundo ele atualizará a página web do usuário com as leituras mais recentes do sensor, além de confirmar alguma alteração feita pelo usuário na programação.

O artifício usado pelo cliente para pedir a atualização da leitura do sensor é o AJAX, e o artifício usado pelo *Arduino* para responder é o XML (Figura 25). O XML nada mais é que texto estruturado, bem semelhante ao HTML. Para o *Arduino*, essas operações de envio de texto são como “*prints*” para o *Serial Monitor*, mas na verdade estão sendo envidadas ao cliente via *Ethernet Shield*.

```
<?xml version = "1.0" ?>
<inputs>
  <relogio>8</relogio>
  <relogio>12</relogio>
  <prog1>0</prog1>
  <prog1>21</prog1>
  <prog1>8</prog1>
  <prog1>1</prog1>
  <prog2>0</prog2>
  <prog2>21</prog2>
  <prog2>8</prog2>
  <prog2>1</prog2>
  <prog3>0</prog3>
  <prog3>21</prog3>
  <prog3>8</prog3>
  <prog3>1</prog3>
  <bomba>0</bomba>
  <automatico>0</automatico>
  <sensor>31</sensor>
  <sensor>28</sensor>
</inputs>
```

Figura 25: Como o Servidor responde ao cliente (XML).

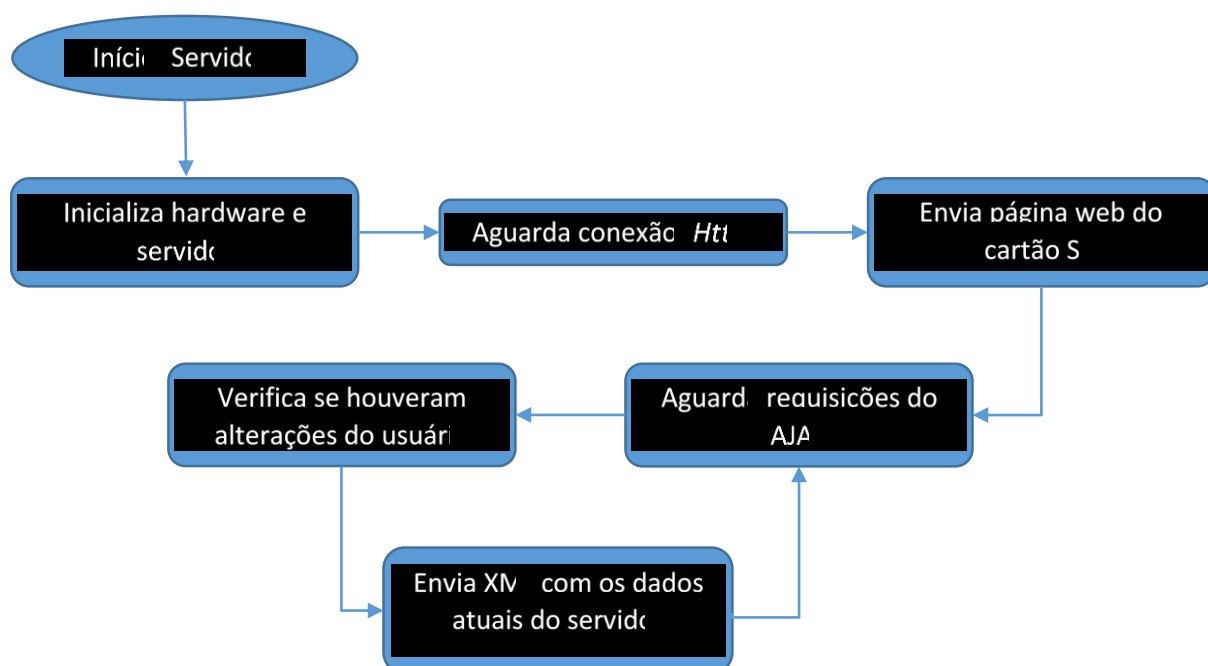


Figura 26: Fluxograma com o funcionamento do *software* do Servidor *Arduino*

Caso o cliente tenha feito alterações na página web, o servidor ficará sabendo ao receber a requisição do AJAX. O XML já será enviado contendo as alterações do usuário.

5.4.2. Cliente

Observando agora a programação do “lado cliente”, devemos acompanhar o fluxograma da Figura 28.

O cliente pode ser uma pessoa qualquer que digita o endereço *http* do servidor em um browser de internet. Essa pessoa pode estar usando um computador, celular ou outro dispositivo que tenha acesso à internet através de um *browser* que interprete HTML como página web.

Ao receber o texto HTML do servidor, o *browser* o exibirá como uma página web qualquer. No presente caso, a página web do sistema de supervisão e controle de umidade da estufa.

Nessa página web o usuário terá acesso à determinadas variáveis do sistema. Ele poderá monitorar os valores de temperatura e umidade lidos pelo sensor, poderá verificar se a bomba se encontra ligada ou não e poderá mudar os parâmetros do controle de umidade. Apenas esses menus e informações estarão disponíveis ao usuário.

Com a página web aberta no dispositivo do usuário, o sistema entrará no mesmo ciclo que o servidor, de atualizar as informações que o usuário enxerga na página web. Como citado anteriormente, o cliente usará AJAX para pedir informações ao servidor, e receberá sua resposta em XML.

O AJAX é um tipo de abordagem do *JavaScript*, ou seja, são scripts escritos no HTML, feitos para serem executados no dispositivo do cliente ao invés de executados pelo servidor. O AJAX se utiliza de requisições http comuns para pedir e para enviar informações ao servidor. O AJAX permite que as informações na tela do cliente sejam atualizadas sem que a página tenha que ser inteiramente recarregada.

Na Figura 27 é possível ver alguns exemplos de requisições http utilizadas pelo cliente para enviar e pedir informações ao servidor. Além do endereço da página (<http://192.168.100.12>), há informações que o AJAX coletou na página (interações do usuário) que serão enviadas ao servidor.

```
http://192.168.100.12/uP=&nocache=146278.90531722532
```

```
http://192.168.100.12/sT=&hr=17&mn=16&nocache=56989.4388528488
```

```
http://192.168.100.12/aT=&stA=0&nocache=363001.471053246
```

Figura 27: Como o cliente faz requisições ao Servidor (requisição http AJAX).

Além disso, o XML recebido do servidor é interpretado na execução de outros scripts contidos no HTML da página.

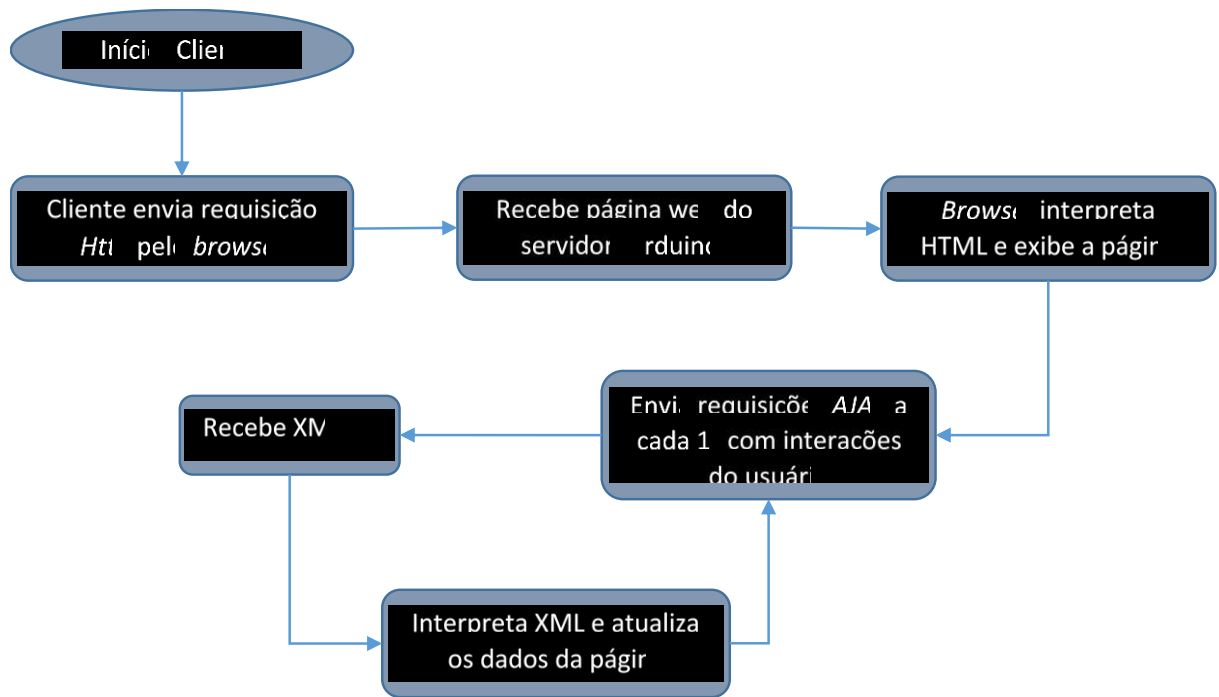


Figura 28: Fluxograma com o funcionamento do *Software* do Cliente.

Capítulo VI

6. Resultados

6.1. Protótipo

6.1.1. Painel Elétrico

Como o contator já estava instalado em conjunto com a bomba, ele não precisou ser colocado junto com o Arduino e demais componentes.

Houveram muitos cabos saindo da caixa, cabo ethernet, alimentação do Arduino, sinal do sensor e acionamento do contator pelo relé. As ligações do Arduino para os módulos foram feitas com um conector múltiplo (Sindal), e não ficou organizado. O ideal seria a fixação de uma mini protoboard junto com esses componentes, para facilitar modificações. (Figura 29)

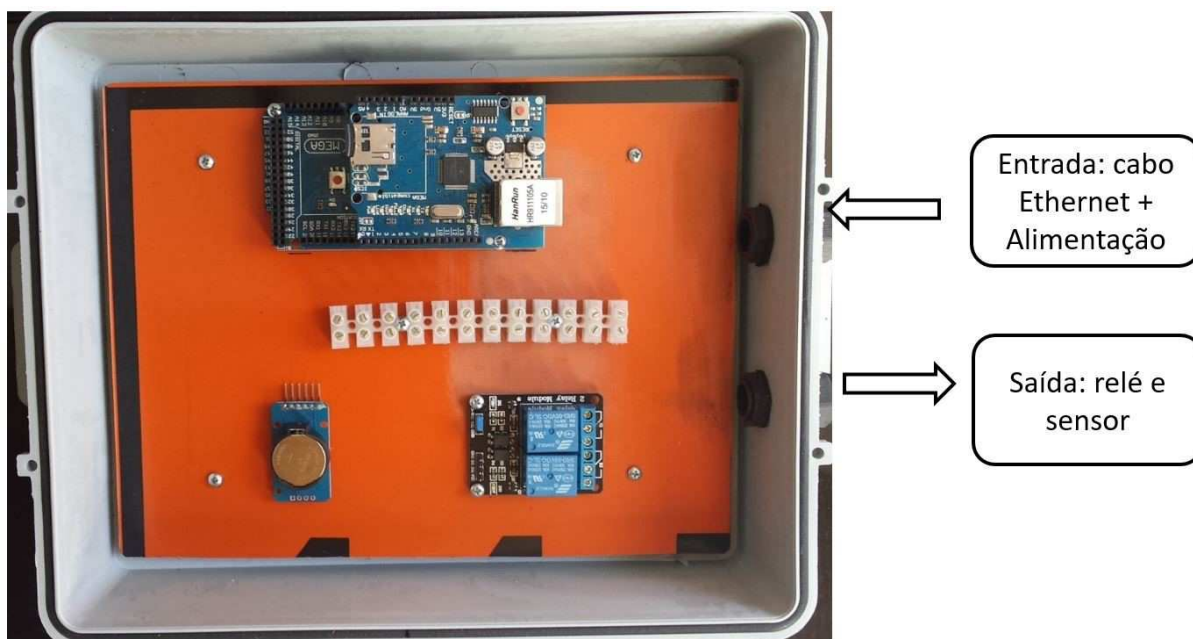


Figura 29: Painel do protótipo

6.1.2. Proteção do Sensor

Baseado no anteparo de Stevenson, foi adaptada uma proteção capaz de evitar radiação solar e contato direto com a água para o sensor sem prejudicar a circulação de ar. (Figura 30)

O recomendado é instalar o sensor no meio das plantas, já que o ambiente não é fechado e então não é possível saber se a umidade está distribuída uniformemente pela estufa. O sensor foi então instalado no centro da estufa, à altura das plantas que estão penduradas.



Figura 30: Proteção para o sensor

6.2. Página web

A interface do protótipo, a página web, ficou bem versátil. Foram usados recursos primários do HTML como divisórias, botões de seleção, cores, campos de preenchimento, etc. O HTML ainda possui muitas possibilidades de design, como adição de imagens ou menus mais interativos, porém, com a interface como ficou já é possível cumprir os objetivos do projeto.

A Figura 31 mostra como ficou a página.

Controle da estufa

Sensor Umidade: 31 %RH Temperatura: 28 °C Bomba: Bomba desligada	Relógio 8:29 Ajustar relógio alterar	Modo de operação ☒ Automático ☐ Manual
---	---	--

Funções automáticas:			
☒ Controle de Umidade Umidade mínima: 55 % tolerância(+/-): 5 % alterar: Umidade maior que: tolerancia (+/-): alterar	☒ Programação 1 Marcado para: 16:30 duração: 5 marcar hora: : duração(em minutos): marcar	☒ Programação 2 Marcado para: 9:30 duração: 10 marcar hora: : duração(em minutos): marcar	☐ Programação 3

Figura 31: Interface do supervisor

6.2.1. Acesso à página

Para acessar a página o usuário deve abrir seu navegador, no celular ou computador, e digitar o endereço da página. O objetivo do projeto é fazer o monitoramento da estufa de maneira remoto, porém o protótipo só foi testado de maneira local.

Para ser realmente um monitoramento remoto, apenas algumas adições deveriam ter sido feitas ao protótipo, para lhe garantir acesso à internet e ter um endereço de IP que pudesse ser acessado de qualquer lugar.

Acessar a internet também traria alguns inconvenientes, como a necessidade uma chave de segurança ou outros mecanismos de controle de acesso.

O protótipo foi ligado à rede local, ou seja, um cabo *Ethernet* ligava o *Arduino* ao roteador da casa. Dessa forma, qualquer pessoa conectada à rede sem fio da casa teria acesso à interface ao chamar pelo endereço 192.168.1.12 no navegador.

6.2.2. Funcionamento

Sua utilização pode ser tanto apenas para supervisão, ou para fazer alguma alteração no funcionamento da automação.

É possível deixar a automação operando no modo automático, e é possível acionar a bomba quando quiser no modo manual. A cor verde diferencia as funções automáticas das funções nativas. As cores azuis, designam as funções que podemos chamar de nativas, pois estarão sempre visíveis quando a página for acessada. Já as funções automáticas, é possível desmarcar a caixa de seleção de cada uma delas, e assim ela será ocultada e desativada (repare na “Programação 3” da Figura 31, que está oculta).

A primeira aba azul da página mostra a leitura de umidade e temperatura do sensor e se a bomba está ou não desligada (Figura 32). O botão “bomba desligada” ou “bomba ligada” tem duas funções diferentes. Quando no modo automático, ele apenas indica o funcionamento ou não da bomba. Quando no modo manual, se pressionado, ele liga ou desliga a bomba.

A segunda aba azul serve para garantir que o relógio do protótipo está correto, e é possível ajustá-lo pelo campo de preenchimento. A terceira aba alterna entre modo automático e modo manual.



Figura 32: A página exibe se a bomba se encontra ligada ou desligada.

As funções automáticas (verdes) são de dois tipos, controle e programação. O controle é o controle da umidade, e é possível escolher o valor mínimo de umidade e a tolerância para esse valor (histerese).

As funções de programação podem ser utilizadas para que a bomba seja sempre ligada no horário escolhido, e que fique ligada por um determinado tempo. O protótipo pode ser programado para que a bomba ligue até três vezes por dia automaticamente (usando as três programações presentes).

6.2.3. Desempenho

A página funcionou corretamente, mas apresentou problemas de desempenho.

O primeiro problema notado foi que a página, durante o carregamento, pode reiniciar a conexão com o servidor e travar. Com poucas tentativas a página sempre acaba carregando por completo, e depois disso esse erro não interfere mais.

O segundo problema notado, é que depois de alguns minutos que a página fica aberta no navegador, ela começa a ocupar muita memória RAM do computador do usuário. Isso provavelmente acontece porque o cliente está enviando requisições a todo tempo para o servidor, para atualizar os campos da página com as informações da automação, como a leitura do sensor. Essas requisições mesmo depois de respondidas pelo servidor, parecem permanecer na memória do navegador, e por isso ele começa a apresentar lentidão depois de um tempo.

Esse problema foi amenizado, mudou-se o intervalo de tempo para cada requisição ao servidor para acontecer de dois em dois segundos. Assim, outro problema surgiu. Agora, ao clicar em algum botão da página, o usuário terá a impressão de que não houve alteração, porque pode demorar até dois segundos para que a interface confirme a mudança feita.

A página também não tem rotinas para evitar que o preenchimento errado dos campos. Então se o usuário digitar, por exemplo, letras no campo que muda as horas do relógio, a reação do servidor pode ser algo inesperado.

Apesar disso, a interface exerce todas as funções desejadas. Com o uso do AJAX a navegação se torna bem confortável, porque a página não precisa recarregar por completo a cada alteração feita, assim como a maior parte dos sites atuais.

A interface foi testada em computadores e celulares, e mais de um dispositivo pode acessar a interface ao mesmo tempo. Quando mais de um dispositivo está conectado, a atualização da página pode ter um pequeno atraso. A navegação pelo celular é menos confortável, mas ainda assim é possível utilizar todas as funções.

As mudanças feitas pelo usuário são confirmadas pelo servidor, e ao se desconectar da interface, o protótipo continuará funcionando segundo a programação do usuário.

6.3. Desempenho do *Arduino*

Os primeiros testes do protótipo foram feitos com o *Arduino* UNO, que como apresentado anteriormente, possui todas as funcionalidades necessárias para construção do protótipo.

6.3.1.Problemas

Reset das configurações

O primeiro problema não é nada que comprometa o funcionamento do protótipo. Mas ao desligar o equipamento, ele perde as configurações feitas pelo usuário. Portanto, no caso de uma queda de energia, o protótipo voltaria a utilizar sua configuração padrão até que o usuário se atente para o ocorrido.

Falta de memória

O funcionamento do protótipo foi satisfatório até certo ponto. Quando a página web começou a ficar maior, e as respostas do servidor em XML também foram aumentando, o Servidor começou a apresentar erros inesperados, como perda de conexão, perda da formatação da página web, envio incompleto ou errado de dados, etc.

Foi constatado que os erros decorriam de falta de memória SRAM no *Arduino* UNO. Isso acontece quando a memória está próxima de 100% de uso, e é recorrente acontecer que o *Arduino* utilize um bloco de memória já ocupado para alguma informação nova. As consequências disso são inesperadas, e normalmente o programa para de funcionar até que o *Arduino* seja resetado.

A memória SRAM do *Arduino* UNO não é pequena para um micro controlador da mesma faixa de preço ou de desempenho (Memória SRAM do *Arduino* UNO: 2kB). Porém, a utilização do *Arduino* como servidor exigiu o tratamento de mensagens de texto, ou seja, tratamento de *Strings*.

Cada caractere de uma *String* ocupa 1 byte da memória. As requisições *http* presentes na comunicação cliente – servidor poderiam ser de até 200 caracteres (contendo as informações necessárias, e também cabeçalhos enviados automaticamente pelo browser). Isso corresponde a 20% da memória. Além disso, por padrão, o *Arduino* copia todas as *Strings* presentes no seu código para a memória SRAM ao iniciar seu funcionamento. Então todas as mensagens enviadas do servidor para o cliente também ocupavam a memória SRAM.

6.3.2.Solução

Para o problema da perda de configuração a solução é fazer o armazenamento das variáveis do programa na memória EEPROM do *Arduino*. Essa solução não foi implementada, mas foi possível continuar com os testes normalmente.

O *Arduino* MEGA foi a solução. Com quatro vezes mais memória SRAM os travamentos repentinos desapareceram. A partir daí o projeto seguiu utilizando o *Arduino* MEGA, pois ele era compatível com o *Ethernet Shield* e com os outros componentes assim como o UNO. (Ver tabela 2)

Apesar disso, haviam outras soluções para otimizar o *Software*, como a biblioteca PROGMEM, ou a macro F() no *Serial.print*.

A biblioteca PROGMEM (Campos, 2015), tem, entre outras funções, a função de forçar o *Arduino* a armazenar determinadas variáveis na memória FLASH ao invés da memória SRAM. Na prática, ela evitaria que as *Strings* fossem para a SRAM.

Sua utilização não é tão prática. É necessário utilizar uma rotina para colocar a variável na memória FLASH, e quando for utilizá-la, é necessária outra rotina para tirar ela de lá.

Já a macro F(), embora seja muito prática, só funciona para o *Serial.print*. Apesar disso é muito útil, pois o excesso de *Serial.print* com muito texto estático pode ser a causa do enchimento da memória SRAM.

Para utilizar a macro F(), apenas coloque entre os parênteses o texto do *Serial.print* que será constante durante todo o programa. Ele será armazenado na FLASH, e o *Arduino* será capaz de utilizá-lo quando necessário.

6.4. Ethernet Shield

6.4.1. Vantagens do uso do cartão SD

O combo Ethernet Shield com cartão SD foi muito útil ao projeto, resolveu diversos problemas como conectividade e armazenamento. Além disso o cartão SD também se mostrou útil na hora de programar.

O tamanho final da página web desse projeto foi de 15kB. Isso significa que nem a memória do Arduino MEGA seria suficiente para armazenar a página. Isso também quer dizer, que ao utilizar um cartão SD você tem mais liberdade para estilizar sua página, sem se preocupar com o limite de memória do seu microcontrolador.

Também foi notado que escrever a página HTML na interface do Arduino não é confortável. É necessário sempre utilizar a função *client.print* para escrever uma linha de texto em HTML. O uso de várias linhas de texto em HTML logo se torna desconfortável para ser editado. Ao utilizar o cartão SD, é possível utilizar um editor de texto como o notepad++, que inclusive destaca as tags e funções do javascript com outras cores.

Para os iniciantes pode ser útil a utilização do cartão SD para separar o cliente do servidor. Na teoria, o Arduino é apenas o servidor. Mas, na prática, é ele que possui o programa do cliente. Então, quando se usa a interface do Arduino para escrever o código HTML (cliente), logo se torna confuso saber se o trecho do código escrito está, por exemplo, utilizando variáveis do Servidor ou do cliente.

Não se deve esquecer da capacidade do cartão SD de armazenar os dados de funcionamento do programa, ou datalogging. Com poucas linhas de programação é possível salvar um arquivo de texto com as variáveis do Arduino Figura 33 e depois utilizar essa informação para plotar gráficos ou procurar erros.

```

333 void data_log() {
334   if (millis() - timer_log > timeout_log) {
335     File dataFile = SD.open("datalog.txt", FILE_WRITE);
336     if (dataFile) {
337       dataFile.print(t.year); dataFile.print("/");
338       dataFile.print(t.mon); dataFile.print("/");
339       dataFile.print(t.date); dataFile.print(",");
340       dataFile.print(t.hour); dataFile.print(":");
341       dataFile.print(t.min); dataFile.print(":");
342       dataFile.print(t.sec); dataFile.print(",");
343       dataFile.print(STATE_bomba); dataFile.print(",");
344       dataFile.print(temperatura); dataFile.print(",");
345       dataFile.print(umidade);
346       dataFile.println();
347       dataFile.close();
348     }
349     else {
350       Serial.println("error opening datalog.txt");
351     }
352     timer_log = millis();
353   }
354 }

```

Figura 33: Trecho do código usado para fazer o *datalogging* no cartão SD.

6.4.2. Alternativas ao *Ethernet Shield* e página Web.

Apesar do ótimo desempenho da interface como página web, da conexão usando cabo *ethernet* e do cartão SD incluso, surgiu a dúvida se a utilização do *Ethernet Shield* foi a melhor escolha.

Devido ao grande trabalho decorrido da programação em HTML, *JavaScript*, XML e outros conceitos antes desconhecidos, talvez houvesse alguma possibilidade de utilizar algo mais prático.

- ESP8266(Módulo Wi-Fi):

O ESP8266 está praticamente substituindo o *Ethernet Shield* no mercado. Utilizar a rede Wi-Fi tem sido mais comum e confortável do que usar conexões com cabos *Ethernet*.

A possibilidade de se utilizar um módulo Wi-Fi foi descartada pois sua programação seria parecida, utilizando HTML, e também seria perdido o advento do cartão SD. Porém, a possibilidade de utilizar esse módulo em otimizações futuras ainda é uma opção interessante.

- Elipse Mobile:

A plataforma Elipse é utilizada para construção de sistemas supervisório para CLP. Existe a versão Elipse Mobile, que permite que um supervisório seja desenvolvido para uma automação com *Arduino*, e que possui interface acessível via celular para o usuário.

A programação da interface não seria em HTML, seria utilizando a linguagem gráfica do *software* Elipse. Ou seja, a programação seria mais fácil, e o *software* final ainda teria um visual melhor.

O problema do Elipse Mobile, é que ele não utiliza o *Arduino* como servidor. Na verdade, um computador será ligado ao *Arduino*, que controlará apenas os atuadores e sensores como um micro controlador qualquer. O computador será o servidor, e será responsável por intermediar a comunicação entre os usuários (clientes) e o *Arduino* (sistema).

O Elipse mobile não vai contra os objetivos desse projeto, mas para fins de simplificação do protótipo, usar o *Arduino* com *Ethernet Shield* como servidor foi a melhor opção.

6.5. Controle da umidade relativa do ar

O controle *on/off* sofreu algumas mudanças para melhor se adaptar aos testes.

Além das variáveis “*Setpoint*” e “*tolerância*”, que estão presentes na interface do usuário, foram adicionadas as variáveis “*duração*” e “*descanso*” no algoritmo presente no *Arduino*.

Essas variáveis foram adicionadas para que a bomba não funcionasse ininterruptamente caso a umidade não aumentasse. Dessa forma foi evitado o desperdício de água e energia.

A variável “*duração*” define quanto tempo a bomba pode funcionar para tentar atingir o valor do “*Setpoint*”. Já a variável “*descanso*”, define quanto tempo a bomba deve descansar a ação de controle, ou seja, o tempo que deve passar para que a variável “*duração*” seja renovada.

6.5.1. Coleta dos dados

Depois da *montagem* do protótipo, o sistema foi testado com apenas a função automática de controle ativada. Enquanto isso, o sistema armazenou dados de funcionamento no cartão SD, num arquivo de texto.

Os dados armazenados foram: data, hora, minutos, segundos, temperatura, umidade e bomba ligada ou desligada. Os dados foram posteriormente tratados e colocados numa planilha do excel.

6.5.2. Condições dos testes

Tomaremos para análise os dados do dia 19 de novembro, que foi um dia nublado, temperatura média, e que teve um pouco de chuva. Os dados estão presentes nos gráficos das figuras Figura 34 e Figura 35.

Na realidade, os testes começaram no dia 18, e terminaram no dia 20 novembro. Dois gráficos complementares desses dias podem ser encontrados nos Apêndices desse trabalho.

O *Setpoint* de controle foi colocado em 70 %RH, e a tolerância era de 5 %RH. Isso significa que a bomba era ligada em 65 %RH, e apenas desligada em 75 %RH. A outra condição de desligamento da bomba era se ela ficasse ligada por mais 10 minutos e não atingisse 75 %RH. Nesse caso, ela só tornaria a ser ligada em uma hora.

Variáveis do controle:

- *Setpoint* = 70%
- Tolerância = 5%
- Duração = 10min
- Descanso = 1 hora

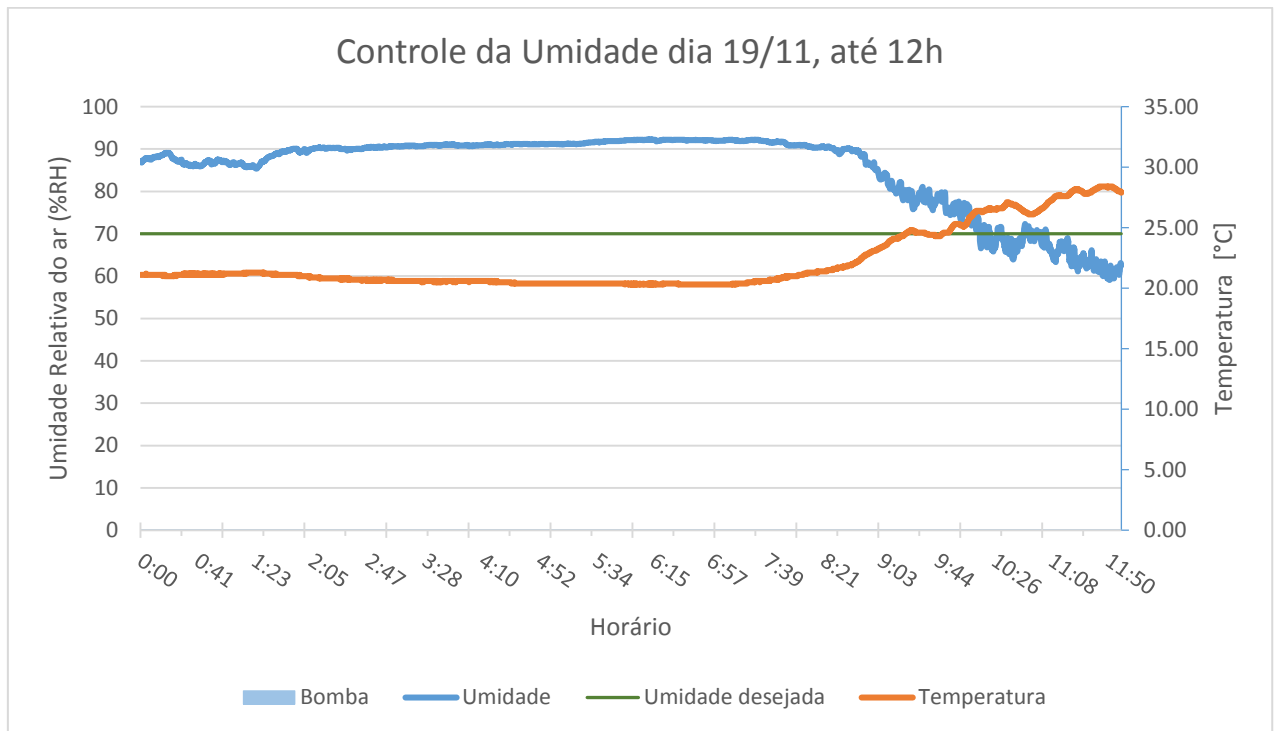


Figura 34: Histórico da umidade e temperatura de 0:00 até 12:00 do dia 19/11.

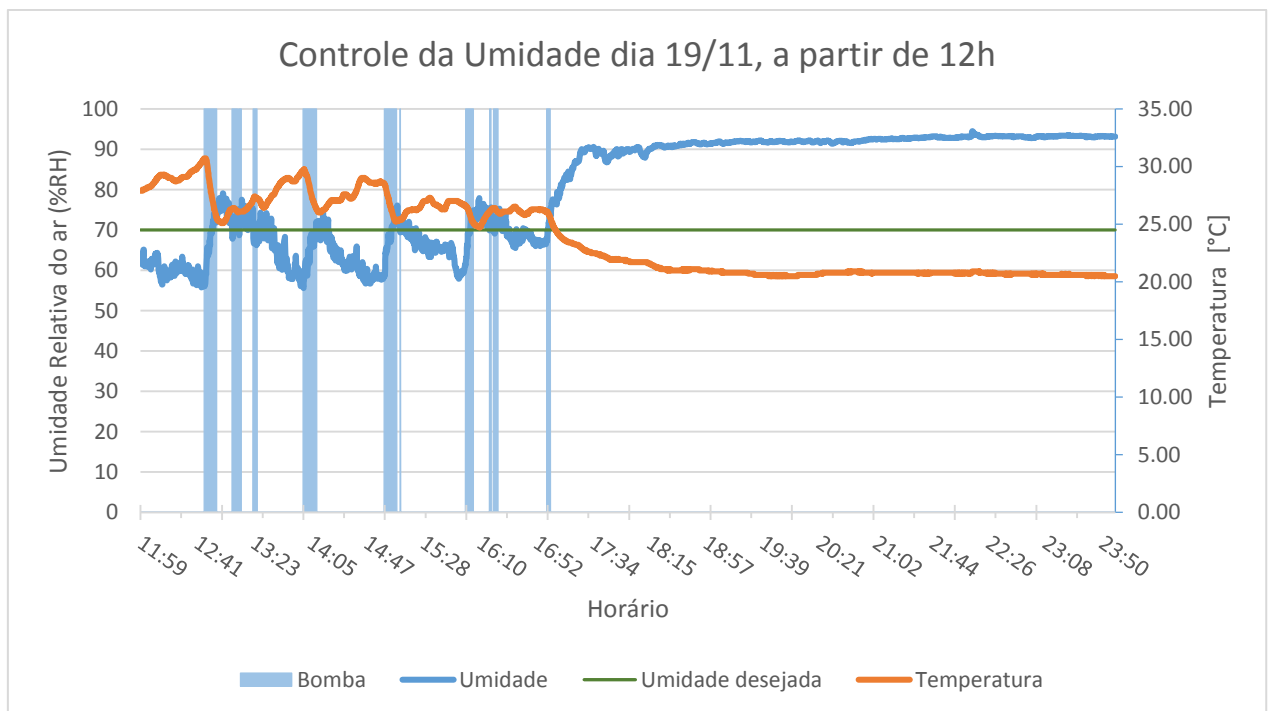


Figura 35: Histórico da umidade e temperatura de 12:00 até 0:00 do dia 19/11.

6.5.3. Análise dos gráficos

Nos gráficos, a variável umidade relativa é representada pela linha azul escuro, a temperatura pela linha laranja, o *Setpoint* de controle pela linha verde, e atuação da bomba pela área sombreada em azul.

Alguns problemas técnicos podem ser notados, mas não influenciam nos resultados finais. Por exemplo, às 10:00 a umidade foi abaixo de 70% e a bomba não foi ligada. Isso aconteceu porque o equipamento foi reiniciado manualmente e perdeu a configuração ajustada anteriormente.

Também é importante reparar, que além da irrigação da bomba, a umidade e temperatura podem ter sido influenciadas pelo amanhecer e consequente aparecimento do Sol, ou por chuva. A presença ou não de irrigação pôde ser incluída no gráfico (área sombreada de azul), mas chuva e Sol não.

A principal conclusão tirada da análise dos gráficos é: Durante o dia é necessária muita água para manter a umidade.

Especialmente no período de 10:00 às 17:00, houve maior atuação da automação. De hora em hora a bomba era ligada durante 10 minutos para reerguer a umidade acima de 70%RH. Ainda assim, após poucos minutos a umidade já estava abaixo do *Setpoint*.

Ao observar a estufa durante os testes foi possível perceber que ela ficava encharcada durante boa parte do dia, e mesmo assim a umidade não era mantida no valor desejado. O motivo disso provavelmente era a falta de evaporação. Isso quer dizer que parte da água que era jogada na estufa durante a atuação da bomba evaporava e aumentava a umidade, e o restante ficava em excesso no chão, evaporando bem lentamente, causando o excesso de umidade que se pode ver durante a noite.

Há dois problemas sérios nessa situação. Primeiro, o acúmulo de água na estufa, no chão, nas plantas e nos vasos é prejudicial para as plantas, que podem adquirir fungos nas folhas. Segundo, a bomba estava consumindo muita energia e água. Possíveis soluções para esses problemas estão contidas no capítulo VII – Discussão e conclusão.

Capítulo VII

7. Discussão e Conclusão

Esse trabalho apresentou resultados satisfatórios. O protótipo criado possui sensor de umidade e temperatura, possui controle de umidade, possui interface gráfica para o usuário, possui histórico de umidade e temperatura da estufa e possui conectividade com redes locais.

O sensor de umidade escolhido, com a devida proteção, consegue ficar no meio das plantas sem se molhar e obter leituras exatas. Para se obter leituras mais exatas ainda, seria recomendado utilizar um ambiente fechado e ventilado, com umidade uniforme em todos os pontos.

O protótipo com Arduino UNO não apresentou desempenho satisfatório, mas o Arduino MEGA solucionou esse problema. As funções de processamento efetuadas pelo Arduino foram muitas: comunicação serial com o sensor; comunicação I²C com o relógio; acionamento do relé da bomba no horário programado; processamento das requisições de servidor web, com tratamento de *strings*; leituras e escritas no cartão SD, via SPI.

A interface web do protótipo também apresentou resultados positivos. Ela não era obrigatória para o funcionamento do sistema, mas era usada para acessar suas informações e alterar a programação. Nela também se encontravam as funções de controle de umidade.

O controle de umidade foi testado, o histórico da umidade foi armazenado durante os testes e usado para confeccionar os gráficos da seção (6. 5 Resultados). Pela análise dos gráficos foi possível determinar que o controle de umidade é mais necessário durante o dia, onde a temperatura é alta e é necessário constante atuação.

A análise desse aspecto permitiu também tirar a conclusão de que usar a irrigação não é a melhor maneira de aumentar a umidade. O melhor é usar aspersores, aspersores ultrassônicos, ou outros que conseguem emitir água em partículas menores. Com partículas menores de água sendo jogadas, o potencial de evaporação seria maior, os pingos quase não

chegariam até o chão antes de se misturarem ao ar. Assim, a umidade da estufa seria mantida mais estável durante o dia.

Pela análise presencial dos testes, foi possível perceber um alto desperdício de água. A estufa se encharcava facilmente, com pouca evaporação a umidade não aumentava, fazendo o sistema jogar mais água ainda. Esse problema também seria solucionado ao se utilizarem aspersores.

Assim sendo, o protótipo cumpriu o propósito de controle e supervisão da umidade para a estufa.

7.1. Trabalhos futuros:

Otimizações do projeto:

- Substituição dos regadores por aspersores
- Conexão com internet
- Conexão Wi-fi com a internet
- Melhorias na confiabilidade do sistema: usar EEPROM, consertar vazamento de memória da página web, verificar preenchimento de dados na página web.
- Página web responsiva para dispositivos móveis

Novos projetos:

- Usar *RaspberryPI* como servidor
- Usar a página web para supervisão de outro sistema

8. Referências Bibliográficas

American Orchid Society. The Importance of Humidity and Air Movement in Successful Orchid Culture. *American Orchid Society*. [Online] <http://www.aos.org/orchids/additional-resources/humidity-and-air-movement.aspx>.

Arduino Official Site. 2017. Arduino Uno Rev3. *Arduino*. [Online] Arduino AG, 2017. <https://store.arduino.cc/usa/arduino-uno-rev3>.

Australian Bureau of Statistics. 2006. Temperature measurement and the Stevenson screen. *Australian Bureau of Statistics*. [Online] Australian Bureau of Statistics, Agosto de 2006. <http://www.abs.gov.au/ausstats/abs@.nsf/0/e4d19a717ba99a65ca256f7200832fd6?OpenDocument>.

Campos, Augusto. 2015. Mais memória no Arduino: indo além dos 2KB de RAM com a PROGMEM. *Br-Arduino.org*. [Online] Junho de 2015. <https://br-arduino.org/2015/06/arduino-proGMEM-sram.html>.

Chapman, Stephen. 2017. Introduction to JavaScript. *ThoughtCo*. [Online] Setembro de 2017. <https://www.thoughtco.com/what-is-javascript-2037921>.

D-Robotics. 2010. DHT11 Humidity and temperature sensor. *D-Robotics UK*. [Online] 2010. <http://www.micropik.com/PDF/dht11.pdf>.

Earl, Bill. 2013. Optimizing Program Memory. *Adafruit*. [Online] 2013. <https://learn.adafruit.com/memories-of-an-arduino/optimizing-program-memory>.

ETEC. UMIDADE RELATIVA. ETEC. [Online] <http://etec.com.br/muda3.html>.

Faria, Caroline. Umidade Relativa do Ar. *InfoEscola*. [Online] <https://www.infoescola.com/meteorologia/umidade-relativa-do-ar/>.

Jain, Preeti. Humidity Sensor. *Engineers Garage*. [Online] <https://www.engineersgarage.com/articles/humidity-sensor>.

Liu, Thomas. Digital-output relative humidity & temperature sensor/module datasheet. *Sparkfun*. [Online] <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf>.

Michell Instruments. 2011. PCMini70 Relative Humidity and Temperature Probe. *Steam equipment*. [Online] 2011.

http://www.steamequipments.com/pdf/RelativeHumidityInstruments/PCMini70_97177_UK_Datasheet.pdf.

Netafim. Tecnologia para estufas. *Netafim*. [Online] http://www.netafim.com.br/crop/greenhouse_technology/cooling.

Oblack, Rachelle. 2016. What Is a Hygrometer and How Does It Work? *ThoughtCo*. [Online] Junho de 2016. <https://www.thoughtco.com/what-is-a-hygrometer-3444417>.

Perdigones, Alicia, Benedicto, Susana e García, José Luis. 2015. Automatização de estufas: a evolução das tecnologias. *Agronegocios*. [Online] Janeiro de 2015. <http://www.agronegocios.eu/noticias/automatizacao-de-estufas-a-evolucao-das-tecnologias/>.

Pereira, Ana Paula. 2009. O que é XML? *TecMundo*. [Online] Março de 2009. <https://www.tecmundo.com.br/programacao/1762-o-que-e-xml-.htm>.

Rocha, Jordão Silva. 2015. IHM ou Supervisório? *AJ Automação*. [Online] Março de 2015. <http://www.ajautomacao.com/ihm-ou-supervisorio/>.

Santos, Guilherme. 2012. A pirâmide da Automação Industrial. *Automação Industrial*. [Online] 2012. <https://www.automacaoindustrial.info/a-piramide-da-automacao-industrial/>.

Santos, Vitor. 2012. AUTOMAÇÃO E REDES INDUSTRIAIS. *REYMASTER*. [Online] Agosto de 2012. <http://reymasterblog.blogspot.com.br/2012/08/automacao-e-redes-industriais.html>.

Sensirion. 2011. Datasheet SHT1x. *Sparkfun*. [Online] Dezembro de 2011. https://cdn.sparkfun.com/datasheets/Sensors/Pressure/Sensirion_Humidity_SHT1x_Datasheet_V5.pdf.

Silevira, Leonardo e Lima, Weldson Q. 2003. Um breve histórico conceitual da Automação Industrial e Redes para Automação Industrial. *Departamento de Engenharia de Computação e Automação UFRN*. [Online] 2003. https://www.dca.ufrn.br/~affonso/FTP/DCA447/trabalho1/trabalho1_13.pdf.

Silva, Ana Paula e Salvador, Marcelo. 2004. O que são sistemas supervisórios? *Wectrus*. [Online] Setembro de 2004. http://www.wectrus.com.br/artigos/sist_superv.pdf.

Silveira, Cristiano Bertulucci. 2017. Sete benefícios conquistados através da Automação Industrial. *Citisystems*. [Online] Setembro de 2017. <https://www.citisystems.com.br/sete-beneficios-automacao-industrial/>.

Smith, W. A. 2013. Arduino Ethernet Shield Web Server Tutorial. *Starting Electronics*. [Online] Janeiro de 2013. <https://startingelectronics.org/tutorials/arduino/ethernet-shield-web-server-tutorial/>.

Vaisala. HMP155 Humidity and temperature probe. *Vaisala*. [Online] <https://www.vaisala.com/sites/default/files/documents/HMP155-Datasheet-B210752EN-E-LoRes.pdf>.

Venamy Orchids. 2002. A Comprehensive Guide to orchid Culture. *Venamy Orchids*. [Online] 2002. <http://www.orchidsusa.com/Culture-Guide.pdf>.

WeatherOnline. Stevenson Screen . *WeatherOnline*. [Online] <https://www.weatheronline.co.uk/reports/wxfacts/The-Stevenson-Screen.htm>.

9. Apêndice

9.1. Controle da umidade relativa do ar, gráficos extras

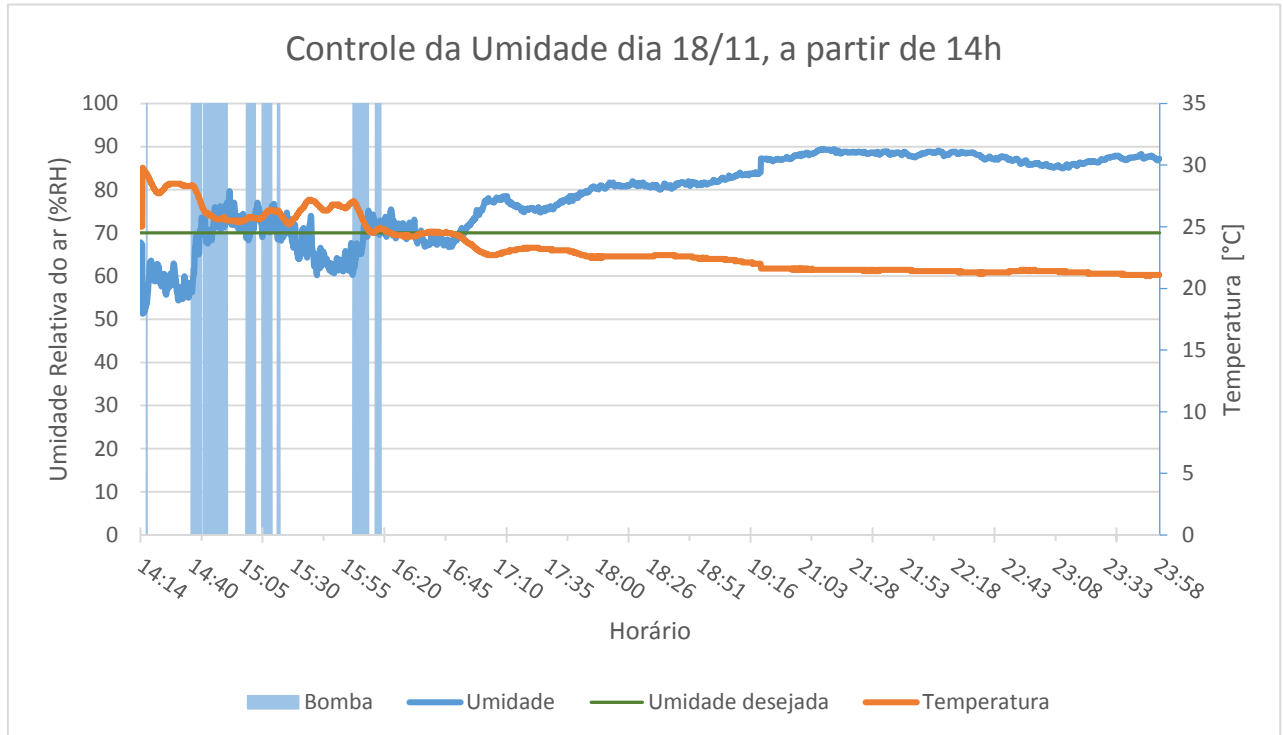


Figura 36: Histórico da umidade e temperatura de 14:00 até 00:59 do dia 18/11.

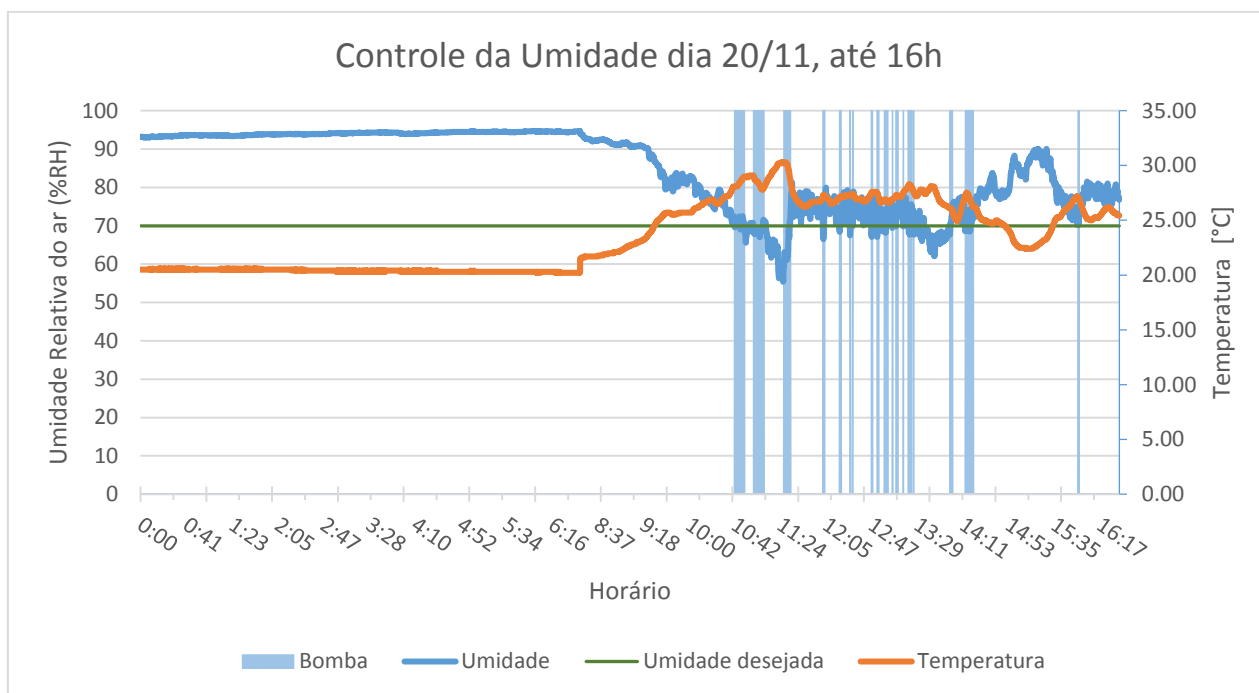


Figura 37: Histórico da umidade e temperatura de 00:00 até 16:30 do dia 20/11.

9.2. Código do Arduino (Servidor)

Acessar:

<https://1drv.ms/f/s!AhCBKPxO9fSWgZ8WpNuMLPIgm3nJFQ>

9.3. Código da página WEB (Cliente)

Acessar:

<https://1drv.ms/u/s!AhCBKPxO9fSWgZ84u26QMWrUAZYW1g>