PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ
ESCOLA POLITÉCNICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

DANIEL RISTOFF PAZ
EDSON GEOVANI DOS SANTOS

RELATÓRIO FINAL DE PROJETO INTEGRADOR
PROJETO COFF

CURITIBA
2015

DANIEL RISTOFF PAZ

EDSON GEOVANI DOS SANTOS

RELATÓRIO FINAL DE PROJETO INTEGRADOR

PROJETO COFF

Relatório de Projeto apresentado ao Curso de Engenharia de Computação da Pontifícia Universidade Católica do Paraná, como requisito parcial para a disciplina de Resolução de Problemas em Engenharia1.

Orientador: Prof. MSc Afonso Ferreira Miguel

CURITIBA

2015

AGRADECIMENTOS

Agradecemos o professor Afonso Ferreira Miguel, que teve a paciência para nos orientar com grande maestria e dedicação.

Ainda cabe um agradecimento ao Fabiano Reino Beraldo que teve uma participação crucial na concepção a ideia inicial.

Agradecemos ao Instituto Lactec e ao SENAI, que disponibilizou os profissionais para a orientação de duvidas.

Um agradecimento especial para os familiares dos integrantes, que contribuíram com todo o apoio e opiniões de melhoria.

RESUMO

Este trabalho apresenta uma solução prática para o controle de fluxos de fluidos, sendo que as possibilidades são de uma vazão continua, ou uma quantidade específica. O controle da quantidade de fluidos é feito para satisfazer a necessidade do consumidor, uma vez que se faz necessário para o uso em projetos específicos relacionado a controle de medidas. O sensor responsável por medir à vazão é disposto antes da válvula solenoide, esta é encarregada de abrir e fechar o fluxo. Os dados gerados são tratados pelo firmware desenvolvido para o arduino, este por sua vez é encarregado de acionar os led(s) que informam a situação do equipamento. Já o envio da informação pelo usuário é feito por meio de um programa desenvolvido em Java que se conecta via Wireless com o modulo Wi-Fi ESP8266 disposto junto ao arduino. O projeto modelo foi desenvolvido com intuito de satisfazer a necessidade do uso de um controlador de fluidos em um ambiente específico que necessite de flexibilidade no controle da quantidade.

Palavras-chave: Controle. Fluidos. Desenvolvido. Necessidade.

ABSTRACT

This paper presents a practical solution for controlling fluid flows, and the chances are of a flow continues or a specific amount. The control of the quantity of fluid is done to satisfy the need of the consumer since it is necessary to use specific designs related to control measures. The accounts for the average flow sensor is disposed before the solenoid valve, the same is in charge of opening and closing the flow. The data generated are handled by firmware developed for Arduino, this in turn is in charge of driving the LED(s) to inform the status of the equipment. Since the user information sending is done through a program developed in Java that connects via wireless with the ESP8266 Wi-Fi module prepared by the Arduino. The model project was developed in order to meet the need of using a fluid controller in a specific environment that requires flexibility in controlling the abundance.

Key-words: Control. Fluids. Developed. Need.

LISTA DE ILUSTRAÇÕES

Figura 1: Projeto Monitoramento e Controle de Nível de Líquido 14
Figura 2: Projeto ECO- Hidráulica 15
Figura 3: Aviso de fluxo de água 15
Figura 4: Estrutura do modelo OSI 17
Figura 5: Esquemático do funcionamento da rede Wireless 18
Figura 6: Exemplo da conectividade da Internet das coisas 19
Figura 7: Dispositivos com Sistemas embarcados 21
Figura 8: Mangueira e Conector 24
Figura 9: Sensor de fluxo e a Válvula solenoide 25
Figura 10: Diagrama Estrutural 25
Figura 11: Diagrama Funcional 26
Figura 12: Diagrama de conexão 28
Figura 13: Esquemático das ligações 29
Figura 14: Esquemático da placa 29
Figura 15: Shield PCI pronta 30
Figura 16: SGF Botão Abrir 32
Figura 17: SGF Botão Fechar 32
Figura 18: SGF Botão Iniciar 32
Figura 19: SGF Botão Conectar 33
Figura 20: SGF Botão Sair 33

LISTA DE MATERIAIS

Lista 1. Materiais Mecânicos. 27
Lista 2. Materiais Eletro Eletrônicos 30
Lista 3. Lista de softwares 31

LISTA DE ABREVIATURAS E SIGLAS

HCF	Hardware de controle de fluxo
SGF	Sistema de Gerenciamento de fluxo
Wi-Fi	Wireless Fidelity
WLANs	Wireless Local Area Networks
UCP	Unidade central de processamento
E/S	Entrada / Saída
OSI	Open Systems Interconnection - Interconexão de Sistemas Abertos
OOP	Programação Orientada a Objeto
ROM	Read Only Memory - Memória Apenas de Leitura
CONAMA	Conselho Nacional do Meio Ambiente
LED	Diodo Emissor de Luz
CC	Corrente Contínua
A	Ampere
V	Volt
mm	Milímetro

SUMÁRIO

1 INTRODUÇÃO 11
1.1 HISTÓRICO DO PROJETO 11
1.2 JUSTIFICATIVAS 12
1.3 OBJETIVOS 12
1.3.1 Objetivo Geral 13
1.3.2 Objetivos Específicos 13
2 ESTADO DA ARTE 14
3 REFERENCIAL TEÓRICO 17
3.1 REDES DE COMPUTADORES 17
3.1.1 Internet Das Coisas 18
3.2 SISTEMAS EMBARCADOS 20
3.3 JAVA 21
4 METODOLOGIA 22
5 O PROJETO 24
5.1 PROJETO MECÂNICO 24
5.1.1 Diagrama Estrutural 25
5.1.2 Diagrama Funcional 26
5.1.3 Lista De Materiais 27
5.1.4 Problemas E Soluções Encontrados 27
5.2 PROJETO ELETRO ELETRÔNICO (HARDWARE) 27
5.2.1 Diagramas Elétricos Eletrônicos 28
5.2.2 Lista De Materiais 30
5.2.3 Problemas E Soluções Adotados 31
5.3 SOFTWARE 31
5.3.1 Lista De Programas 31
5.3.2 Problemas E Soluções Adotados 34
6 RESULTADOS 35
7 IMPACTO AMBIENTAL 36
8 CONSIDERAÇÕES FINAIS 38
REFERÊNCIAS 39

ANEXO A – CÓDIGO FIRMWARE 40
ANEXO B – CÓDIGO SGF.....48

1 INTRODUÇÃO

A medição de vazão de fluidos está sempre em nosso cotidiano. Para exemplificar, uma residência que dispõe de um hidrômetro, ou mesmo uma bomba injetora de combustível, etc.

Essa medição sempre foi um fascínio para Leonardo da Vinci que buscou meios para canalizar a água que escorria de um riacho, ao analisar esse fluxo Leonardo observou que a quantidade de água em um determinado tempo era a mesma em qualquer parte, independente da profundidade a inclinação ou até mesmo da largura (Herbert, 2002). Entretanto só no inicio da era industrial que Bernoulli deu inicio a desenvolvimentos de dispositivos práticos que realizasse essa medição.

Em diversas empresas que mexem com líquidos faz-se necessário o uso de controles do volume e do fluxo, baseando-se nesse princípio e seguindo a sugestão do Fabiano Reino Beraldo, propomos o desenvolvimento de uma estrutura que ao ser acoplado no cano responsável por transportar a água, a mesma faz um controle do fluxo por meio do computador. A infraestrutura de comunicação entre o SGF (Sistema do Gerenciamento de Fluxo) e o hardware que compõem a solução HCF (Hardware de Controle do Fluxo) é feita por meio da rede sem fio (*Wireless*).

1.1 HISTÓRICO DO PROJETO

A sugestão foi levada pelo Fabiano Reino Beraldo, que relatou o problema no controle do fluxo de água usado em sua mini fábrica de cerveja. Pela dificuldade de controlar a quantidade de água, surgiu à ideia do protótipo COFF, que busca realizar essa situação de uma forma prática e de fácil manuseio.

Na busca pela solução do problema, o integrante Daniel Ristoff Paz se atentou para pesquisar qual seria os equipamentos a serem usado, e foi decido como o hardware central o arduino Mega 2650, os periféricos: sensor de fluxo YF-21, válvula solenoide, módulo Wi-Fi ESP8266, e mangueiras para a interligação da válvula com o sensor de fluxo.

O mesmo integrante provido de conhecimentos de programação, ficou também com a função de programar o firmware para o gerenciamento dos

periféricos, e a programação do SGF que é desenvolvida na linguagem de alto nível Java.

O integrante Edson Geovani dos Santos ficou responsável pela compra dos equipamentos, do teste dos mesmos, do desenvolvimento da placa PCI e do modelo mecânico.

O protótipo não fez a necessidade do uso de um display, sendo que toda a situação pode-se ser acompanhada pelo estado que se encontra os Led(s) dispostos na placa PCI desenvolvida.

No decorrer do projeto foi surgindo várias ideias e problemas, foram necessárias adaptações para o pleno funcionamento, entretanto, foi mantida a ideia e os componentes inicialmente projetados.

1.2 JUSTIFICATIVAS

Com o desenvolvimento do COFF, destaca-se vantagem no setor responsável por gerenciar o fluxo dos líquidos em um ambiente de trabalho, proporcionando precisão no controle de fluidos e principalmente gerando uma comunicação mais intuitiva, isso ocasiona feedbacks sobre o andamento do processo e relatórios mais preciso de gastos relacionados ao uso de um fluido. Como vantagem, também é possível destacar a possibilidade que o controlador tem de escolher a quantidade de fluidos que verterá pelo cano, essa opção abre o campo de atuação, e remete a qualquer aplicação que se deseja controlar o fluxo de um liquido em um ambiente específico.

Para o sistema é esperado um programa que tenha as opções de informar a quantidade de fluidos em litros que se deseja monitorar, a interrupção e abertura do fluxo. A abertura ainda pode fica em um modo continuo de vazão, o mesmo disposto de quantos litros foram utilizados.

1.3 OBJETIVOS

No contexto geral é preciso estabelecer objetivos a serem cumpridos no decorrer do projeto, para que isso ocorra é necessária uma visão de como será o projeto e suas principais características funcionais e mecânicas.

1.3.1 Objetivo Geral

Criar um protótipo funcional integrado com a aplicação desenvolvida em Java, que se comunica por meio de uma interface Wireless, a mesma controla os sensores responsáveis por controlar o fluxo do fluido.

1.3.2 Objetivos Específicos

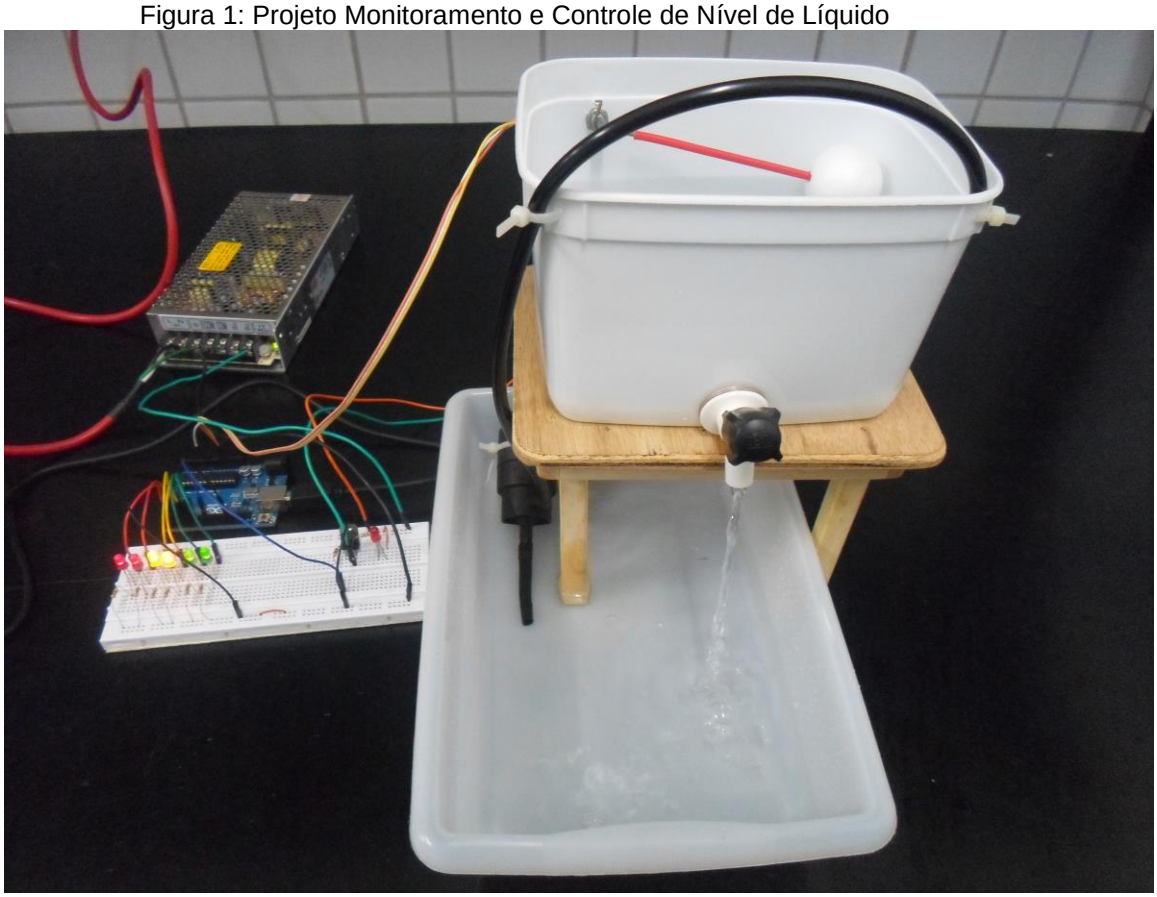
Os objetivos específicos do trabalho são:

a) Controlar a quantidade de fluidos que o usuário solicitar por meio do SGF;
b) Desenvolver botão com comandos de abertura e fechamento da vazão;
c) Mostrar na forma visual por meio de LED a situação do hardware;
d) Desenvolver um protótipo de tamanho e peso reduzido.

2 ESTADO DA ARTE

Um trabalho com características semelhantes ao COFF é o “Monitoramento e Controle de Nível de Líquido" que apresenta a utilização da placa Arduino no controle liga/desliga e monitoramento com LEDs o nível de líquido. Ele é um sistema de tanques em escala reduzida. O Arduino comanda o acionamento de uma bomba de água com o objetivo de controlar e monitorar com LEDs o nível de líquido dentro de uma faixa preestabelecida. Os resultados mostram que é possível desenvolver um controlador liga/desliga no Arduino para comandar e monitorar o nível de líquido em um sistema de tanques.

Figura 1: Projeto Monitoramento e Controle de Nível de Líquido



Fonte: http://www.diy.com.br/projeto/monitoramento-e-controle-de-nivel-de-liquido

Outro projeto semelhante é o ECO – Hidráulica desenvolvida pelos alunos Jessica Caroline Viana, Karoline Schulis, Gabriel Santos Quintão e o Rafael Dobriansky, todos são estudantes da Engenharia da computação da Pontifica Universidade Católica do Paraná. O projeto tem a finalidade de acabar com vazamentos evitando o desperdício desnecessário de água - e dinheiro - através da

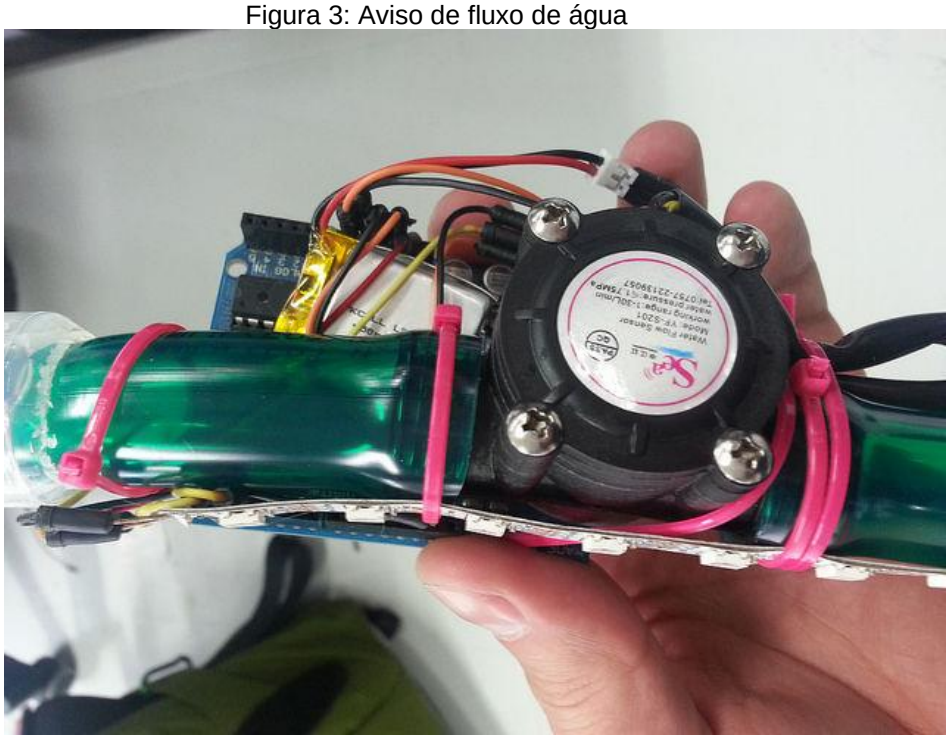
domótica (automatização residencial) relacionada ao total controle do sistema hidráulico.

Figura 2: Projeto ECO- Hidráulica

Fonte: http://www.afonsomiguel.com/content/eco-hidraulica.

Para o sistema é esperado um programa que tenha as opções de informar a quantidade de fluidos em litros que se deseja monitorar, a interrupção e abertura do fluxo. A abertura ainda pode fica em um modo continuo de vazão o mesmo disposto de quantos litros foram utilizados.

Figura 3: Aviso de fluxo de água



Fonte: http://www.usinainfo.com.br/module/csblog/detailpost/106-79-aviso-de-fluxo-deagua.html

O Projeto acima é o “Aviso de fluxo de água” que foi criado por Thomas Amberg, ele ajuda a economizar água acendendo LEDs vermelhos depois que a quantia de um litro for excedida. O aparelho é feito a partir de um Arduino, um sensor de fluxo e LEDs coloridos.

3 REFERENCIAL TEÓRICO

Para o desenvolvimento do projeto COFF, foi necessário o conhecimento sobre vários assuntos, entre eles:

a) Uso da linguagem adequada para o arduino.

b) Uso da linguagem de programação Java.

c) Uso do Eagle, MSprojetc, Netbeans.

d) Uso de conexão wireless entre os periféricos (computador e arduino).

e) Calculo de conversão de fluxo.

3.1 REDES DE COMPUTADORES

Quanto há a necessidade da troca de informações entre aparelhos eletrônicos é preciso à criação de uma rede, que Forouzan (2012, p.2) define como “a interligação de um conjunto de dispositivos capazes de se comunicar”.

Para a comunicação entre os periféricos usando interface de rede é preciso o uso de uma arquitetura, ou seja, uma lógica que especifica como os vários componentes devem se comunicar, essa estrutura é conhecido como o modelo OSI, em uma definição mais formal Forouzan (2012, p.21) define como “uma estrutura em camadas para a concepção de sistemas de rede que permitam a comunicação entre todos os tipos de sistemas computacionais”. Na Figura 4 apresenta o esboço de como é organizado o modelo OSI.

Figura 4: Estrutura do modelo OSI

Fonte: http://pplware.sapo.pt/tutoriais/networking/redes-sabe-o-que-e-o modelo-osi/

Uma rede é composta por várias classificações, para o projeto COFF foi usada uma conexão wireless, ela é baseada na norma (IEEE 802.11) que faz parte da família WLANs, ela se caracteriza por transportar ondas eletromagnéticas sem usar um condutor físico.

Figura 5: Esquemático do funcionamento da rede Wireless

Fonte: http://revistahometheater.uol.com.br/site/tec_artigos_02.php?id_lista_txt=8828

Para a conexão foi cogitado o uso de dois tipos de configurações: cliente-servidor (cliente-server) e ponto-a-ponto (peer-to-peer) a primeira é dispensável no projeto, já a segunda (empregada no projeto) tem como característica a não existência de um de servidor dedicado, assim não existe um computador que está dedicado à função única de oferecer algum tipo de serviço.

3.1.1 Internet das coisas

Com revolução tecnológica surgem equipamentos capazes de se conectar na rede mundial de computadores, essa característica é aplicada a um termo conhecido como Internet das Coisas.

A ideia principal do conceito “Internet das Coisas” é transformar cada vez mais o mundo físico e o digital em um só, isso é feito por meio de dispositivos que se

comunicam uns com os outros, para exemplificar, a Figura 6 mostra um exemplo de como é o modelo empregado pela Internet das Coisas.

Figura 6: Exemplo da conectividade da Internet das coisas

Fonte: http://tmdata.com.br/a-internet-das-coisas-a-arma-da-fidelidade/

O potencial para a Internet das Coisas é praticamente ilimitado, é possível imaginar um futuro próximo, carros que avisarão pelo sistema sonoro que algum problema foi detectado no veículo, ou até mesmo a intercomunicação entre os carros e os sinaleiros, ou o mais inimaginável, carros capazes de se auto dirigir, esses são exemplos do potencial que essa tecnologia nos proporciona.

3.2 SISTEMAS EMBARCADOS

Um sistema embarcado, conhecido também pelas denominações de: sistema embutido ou sistema embebido, ele se caracteriza por um sistema micro processado, ou seja, o computador é completamente encapsulado, em outras palavras ele é dedicado a um dispositivo ou um sistema em específico. Para explicar melhor Null e Lobur comentam no trecho abaixo sobre sistemas embarcados.

É difícil ter uma definição exata de sistemas embarcados, mas no geral eles são computadores reais, tendo uma UCP, memória e algum tipo de capacidade de E/S. Entretanto eles são diferentes dos computados pessoais, pois o proposito deles é realizar um numero limitado de tarefas dentro do domínio de um grande sistema (Null e Lobur. 2010. p. 530).

Nos sistemas embarcados existem restrições que fazem com que o seu desenvolvimento se torne complexo, essas restrições incluem velocidade limitada de UCP, memória limitada, restrições de peso, consumo de energia, ambientes de operação não controlados e algumas vezes hostis, espaço físico limitado etc. (NULL e LOBUR, 2010)

Como citado anteriormente, os sistemas embarcados são desenvolvidos para uma tarefa específica. Por motivos de segurança e para a melhor usabilidade, uma parte dos sistemas embarcados possuem restrições para a computação em tempo real. O software escrito para o funcionamento é conhecido como *firmware*, e as informações armazenadas geralmente em memória ROM ou memória flash, por suas limitações os sistemas embarcados geralmente funcionam sem o uso de teclado, mouse ou até mesmo display.

Figura 7: Dispositivos com Sistemas embarcados

Fonte: http://xxande.blogspot.com.br/2012/09/sistemas-embarcados.html

3.3 JAVA

Para o desenvolvimento do projeto usou uma linguagem de programação de alto nível, o Java, que é uma linguagem orientada a objetos, independente de plataforma e segura, projetada para ser mais fácil de aprender (Cadenhead. 2005).

Ainda segundo Cadenhead (2005. p. 4), a programação orientada a objeto (OOP) é uma metodologia de desenvolvimento de software em que um programa é percebido como um grupo de objetos que trabalham juntos. Os objetos são citados como modelos chamados classes, e contem os dados e as instruções necessárias para usar esses dados.

A principal diferença que faz com que essa linguem se torne uma plataforma extremamente potencial é o fato da neutralidade de plataforma, que é a capacidade de um programa executar sem modificações em diferentes ambientes de computação. Os programas Java são compilados para um formato chamado bytecode, que é executado por qualquer sistema operacional, software ou dispositivo com um interpretador Java (Cadenhead. 2005).

4 METODOLOGIA

A metodologia usada para desenvolvimento do projeto segue um cronograma desenvolvido com a finalidade de delegar atividades para os dois integrantes da equipe, primeiramente o integrante Daniel ficou com a parte de produção do firmware e do SGF, por confeccionar a PCI, o Edson ficou responsável pela compra e busca de informações de funcionamento dos equipamentos, e montar o diagrama da PCI, a montagem e teste de integração do software e o hardware foi desenvolvido pelos dois integrantes da equipe.

O protótipo tem como peça principal de gerenciamento, o arduino, ele é responsável por armazenas o firmware da aplicação e delegar as funções recebidas para os periféricos. É ligado nele uma shield desenvolvida especialmente para controlar os sensores de fluxo e a válvula solenoide.

A shield tem em sua composição 3 LED(s) os mesmos nas cores: verde, amarelo e vermelho, cada LED representa um estado do equipamento, se caso o LED verde estiver ativo, significa que a válvula está aberta, ocorrendo assim o fluxo do fluido sem nenhuma ordem de quantidade a ser escoada, se o LED vermelho estiver ativo, significa que a válvula está fechada, caracterizado assim o impedimento do fluxo, e se o LED verde e o amarelo estiverem ativo, significa que foi enviado uma quantidade delimitada de fluxo que deverá verter no cano, terminando essa quantidade a válvula irá fechar e desligar os LED(s) acionando ao termino o LED vermelho, que sinaliza o fechamento da válvula.

Ainda na shield está disposto o modulo Wi-Fi ESP8266 responsável por fazer a comunicação com o computador. A função dele é receber os comandos enviados pelo SGF e disponibiliza-los ao arduino, que por sua vez os encaminhará aos devidos periféricos.

Para que o controle do equipamento se torne mais simples, conexões rápidas nos pinos dos sensores foram implementados, assim facilitando todo o processo de ligação do protótipo.

Por ser um controle de fluxo, foi discutido desde o inicio formas de deixar o manuseio do protótipo cada vez mais usual, por esse motivo o SGF é feito em Java, assim possibilitou a criação de uma aplicação multiplataformas e muito usual, sendo composta por três botões, um responsável por abrir a válvula, outro por fechar a

válvula e o ultimo por enviar uma quantidade determinada para a vazão. No SGF ainda tem um campo responsável por informar a situação do equipamento.

5 O PROJETO

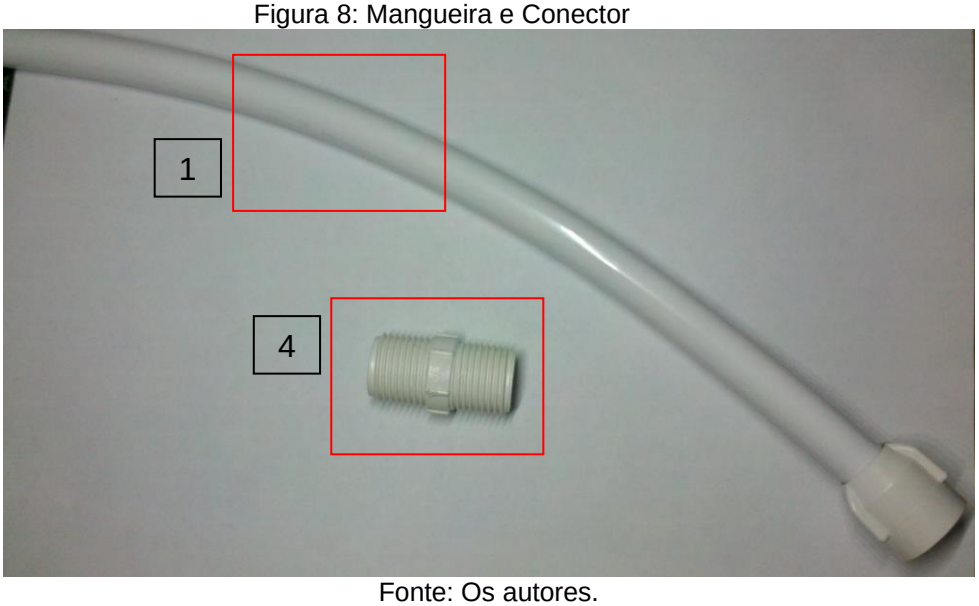
No inicio do fluxo, o usuário dá o comando para o notebook que tem um programa (SGF) que faz a comunicação por meio da rede sem fio (*Wireless*), a mesma é realizada por uma *shield* de *Wireless* que junto com o Arduino será responsável por fazer a comunicação com o roteador, o roteador por sua fez faz a interação entre o software e o hardware, é por meio dele que as informações serão enviadas e outras serão recebidas. Após as informações serem enviadas o Arduino aciona a válvula solenoide, a finalidade dela é o fechamento e a abertura da escotilha na qual o liquido passará, na mesma estrutura será disposto um sensor de fluxo responsável por calcular a quantidade de fluido que passará pelo cano. Ambos os equipamentos estão ligados no Arduino.

Junto ao Arduino é implantado LED que tem a finalidade de mostrar o estado do equipamento.

5.1 PROJETO MECÂNICO

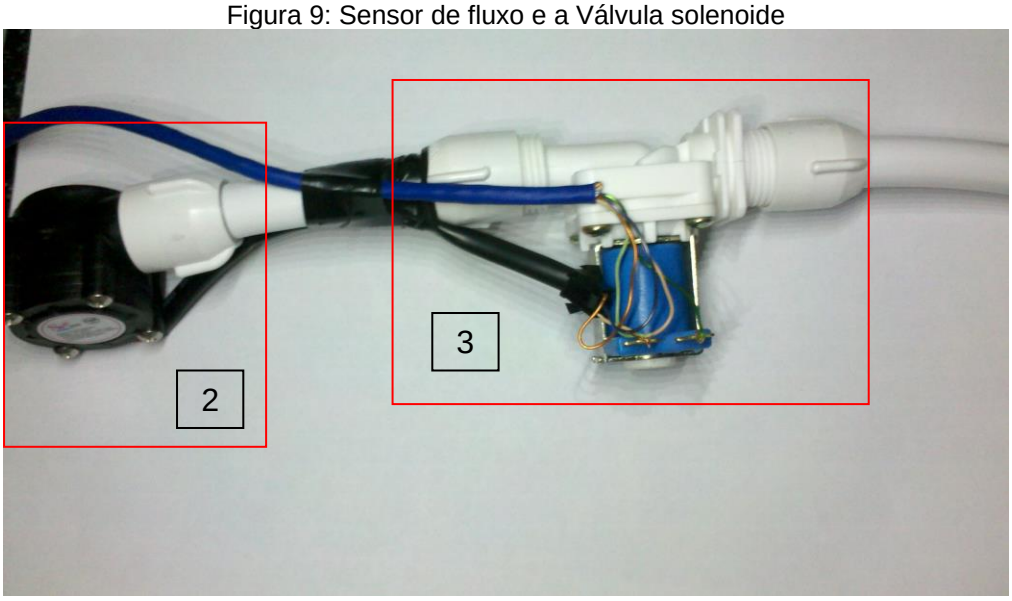
No projeto mecânico usa mangueiras ½ polegada (1) para interligar o sensor de fluxo (2) a válvula solenoide (3) e o ponto com o fluido que verterá, essa conexão do protótipo com o cano é feito por um conector macho macho (4).

Figura 8: Mangueira e Conector



Fonte: Os autores.

Figura 9: Sensor de fluxo e a Válvula solenoide

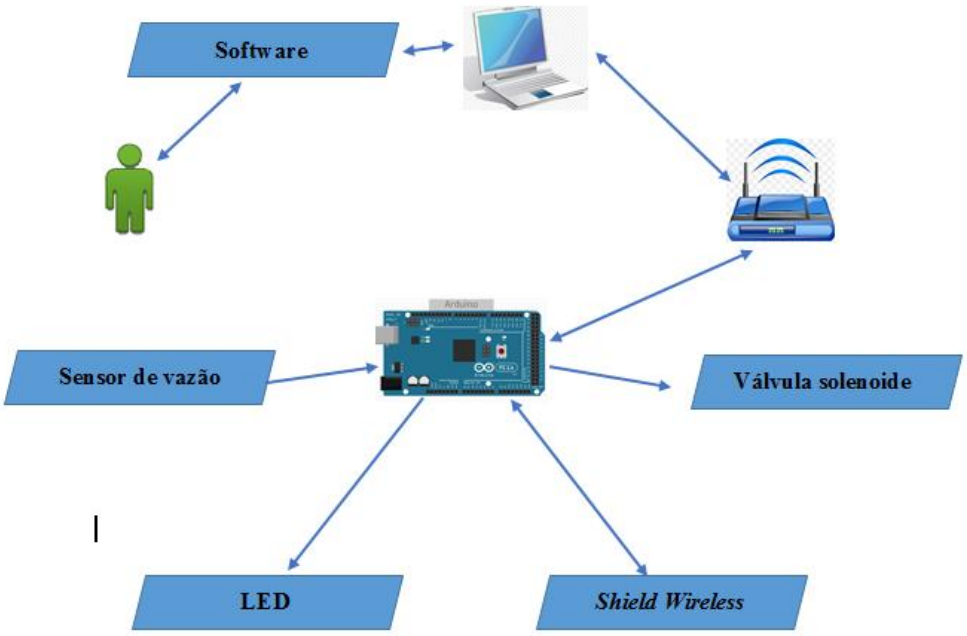


Fonte: Os autores.

5.1.1 Diagrama estrutural

O diagrama estrutural tem a finalidade de mostrar como vai ser o funcionamento de todo o conjunto de equipamentos empregados no protótipo.

Figura 10: Diagrama Estrutural

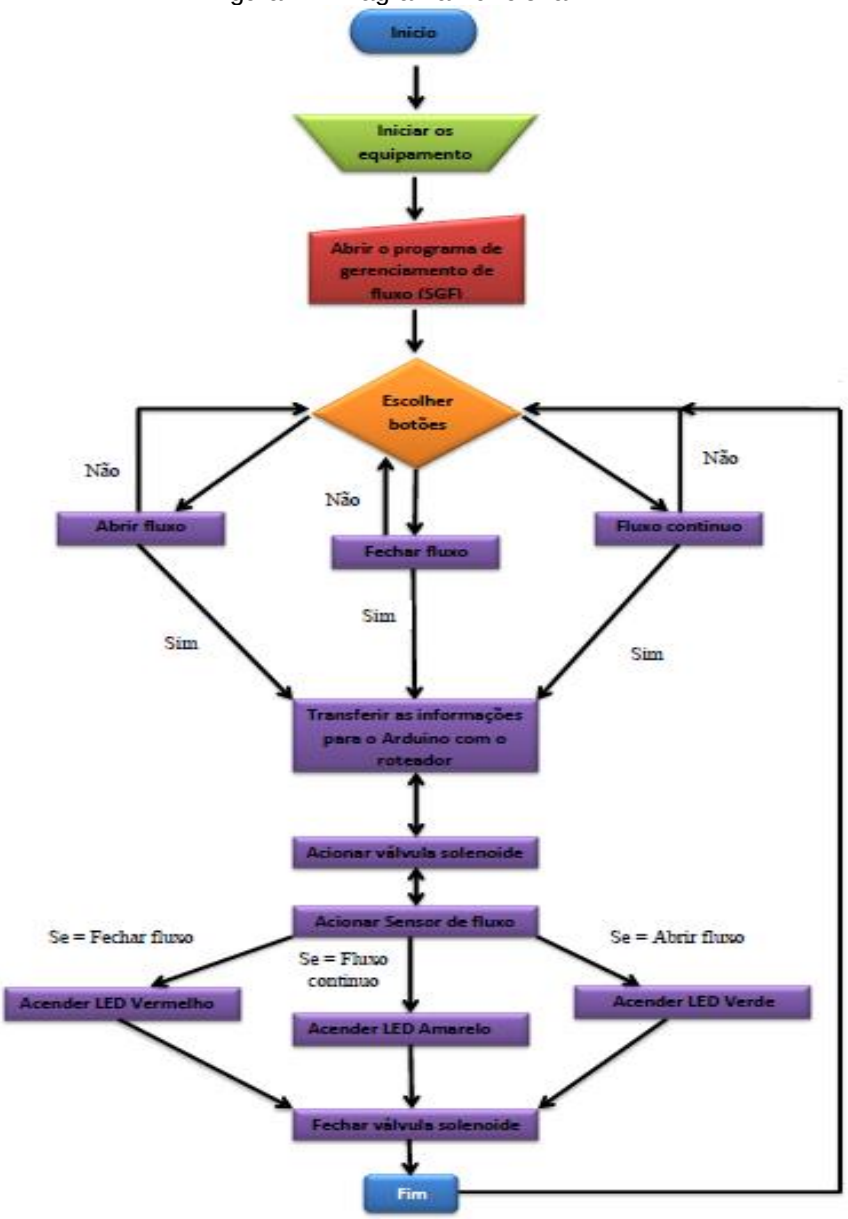


Fonte: Os autores.

5.1.2 Diagrama funcional

O diagrama funcional tem a finalidade representar o fluxo com que os equipamentos e o software irão funcionar.

Figura 11: Diagrama Funcional



Fonte: Os autores.

5.1.3 Lista de materiais

Os materiais utilizados para o desenvolvimento da parte mecânica do projeto são os seguintes:

a) Válvula solenoide 12V;
b) Controlador de fluxo de fluido 5V ½";
c) Canos;
d) Cabo par transado UTP Cat. 5e;
e) Fita isolante;
f) Fonte de alimentação de 12V 3A;
g) Barra de conexão Sindal.

5.1.4 Problemas e soluções encontrados

Foi encontrado um problema com a fonte de alimentação, pois ela teria que alimentar tanto o arduino como a válvula solenoide, então foi feita uma adaptação com a barra sindal para fazer uma ligação em "y" e alimentar tanto o arduino quanto a válvula solenoide.

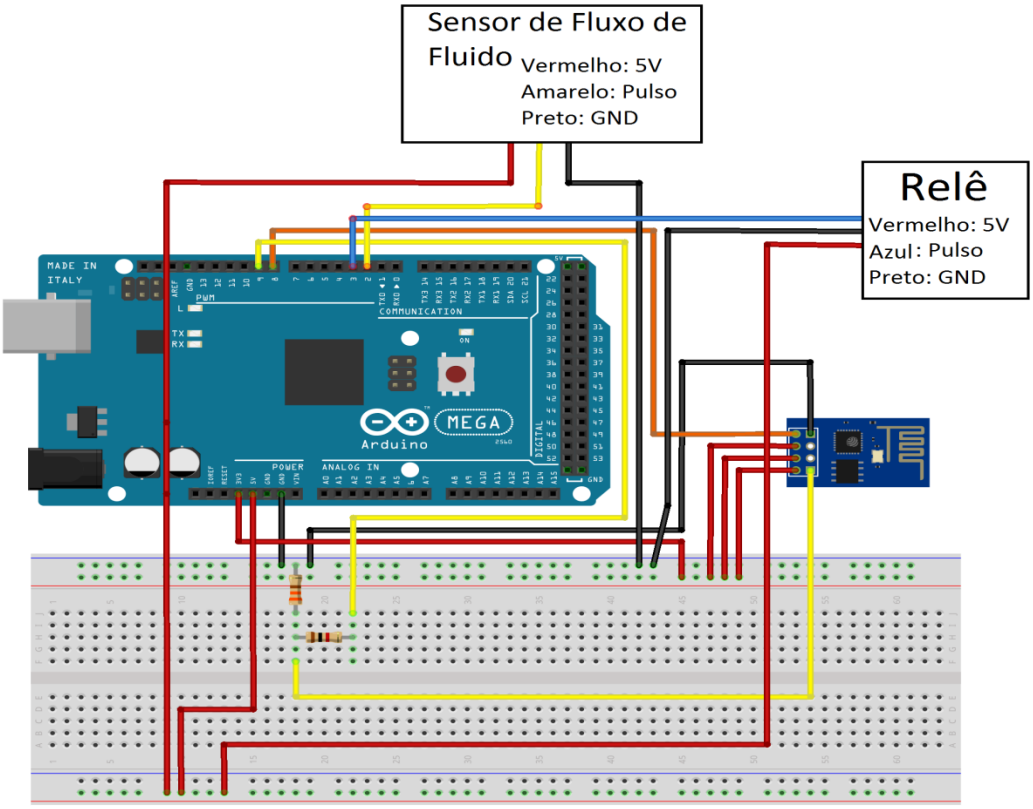
5.2 PROJETO ELETRO ELETRÔNICO (HARDWARE)

Para que o projeto mecânico entre em ação é preciso que o projeto eletrônico esteja funcionando perfeitamente, ele é responsável pelo acionamento dos componentes, pela comunicação entre os periféricos e por mostrar o status que o equipamento se encontra, faz isso por meio de LEDs.

5.2.1 Diagramas elétricos eletrônicos

Para a montagem exata do circuito é viável que se desenhe um esboço para verificar se a ligação entre os equipamentos está correta, essa função está representada pela figura abaixo;

Figura 12: Diagrama de conexão

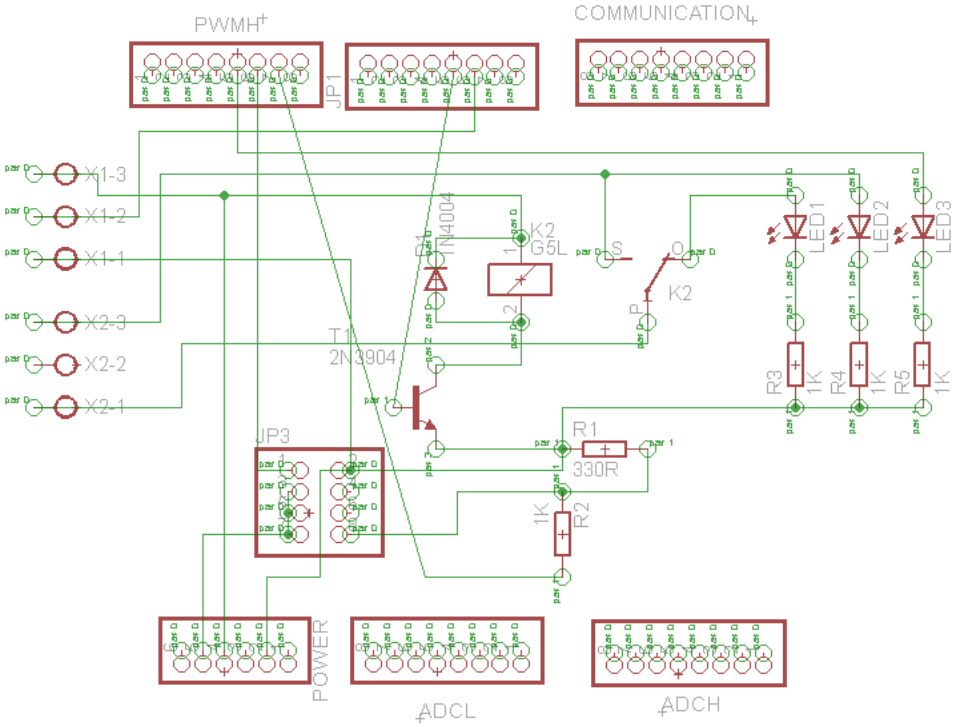


Fonte: Os autores.

Com o uso do diagrama de conexão foi possível desenvolver o protótipo sem ter o problema de errar quando for criar e montar a placa PCI.

Para a criação da PCI usou o software eagle, que é um software que possibilita a criação do esquemático de ligações e o esquemático final, usado para a fabricação da PCI.

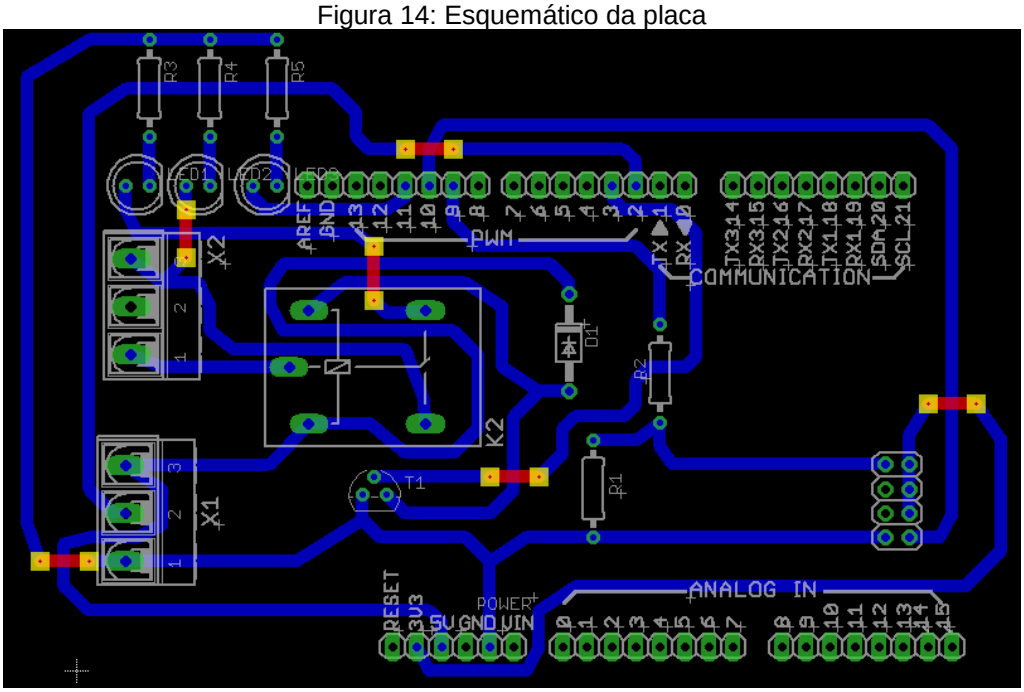
Figura 13: Esquemático das ligações



Fonte: Os autores.

Na figura abaixo representa a versão final da PCI, a mesma pronta para a corrosão.

Figura 14: Esquemático da placa



Fonte: Os autores.

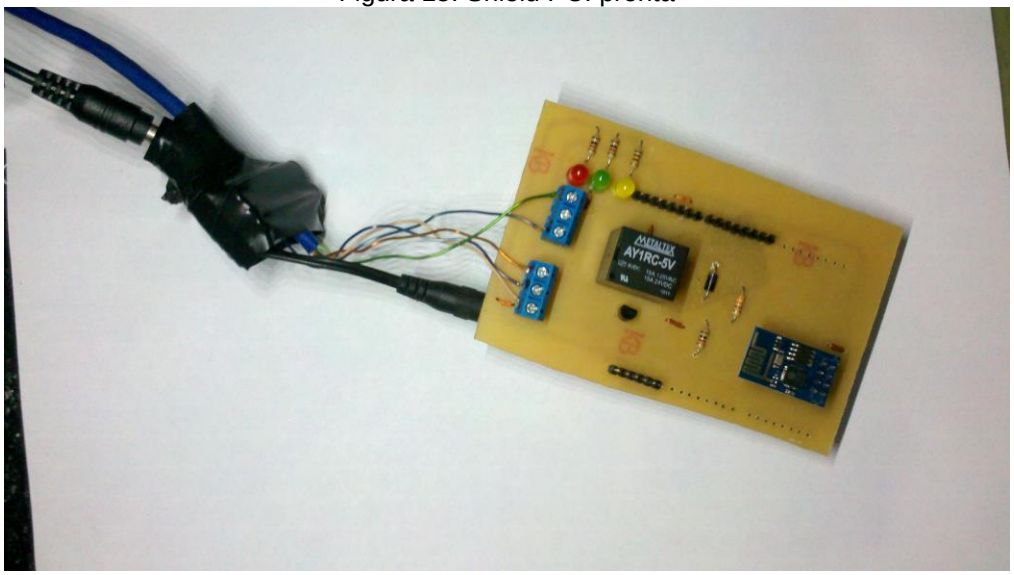
5.2.2 Lista de materiais

Para o desenvolvimento da PCI precisou dos seguintes compenentes;

a) Arduino mega 2560;
b) Modulo WI-FI ESP 8266 para arduino;
c) Placa de circuito impresso de fenolite;
d) 1 resistor de 330Ω;
e) 4 resistores de 1kΩ;
f) 1 diodo retificador;
g) 1 transistor;
h) 1 rele 5V;
i) 2 bornes BR0;
j) 3 LEDs (Verde, vermelho e amarelo);
k) Barra de pinos machos 180° para arduino;

Após a criação da PCI e ela virou uma *shield*, ela fica disposta acima do arduino, tornando um protótipo pequeno e fácil manuseio.

Figura 15: Shield PCI pronta



Fonte: Os autores.

5.2.3 Problemas e soluções adotados

Foi identificado um problema com as ilhas da primeira placa que foi feita, as ilhas estavam pequenas de mais para efetuar as soldas, então refizemos todo o esquemático da placa e adaptamos com ilhas maiores para conseguir efetuar a solda dos componentes.

5.3 SOFTWARE

Toda aplicação necessita de ferramentas para o seu desenvolvimento, as ferramentas utilizadas para o desenvolvimento do COFF está na lista abaixo.

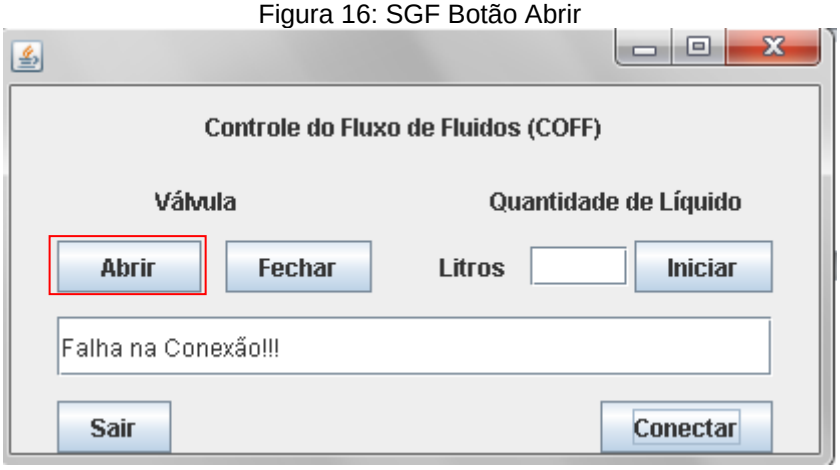
5.3.1 Lista de softwares

Para criação do projeto foram necessários os seguintes softwares:

a) Arduino IDE;
b) Microsoft Project 2013;
c) Netbeans versão 8.2;
d) Microsoft Power Point 2010;
e) Java 1.8;
f) Windows Movie Maker.

O SGF foi desenvolvido em Java, assim trouxe uma maior facilidade de acesso para o usuário, pois foi desenvolvido com uma interface baseada em botões isso facilita o usuário, nele contem três botões, o primeiro é de “Abrir” a válvula.

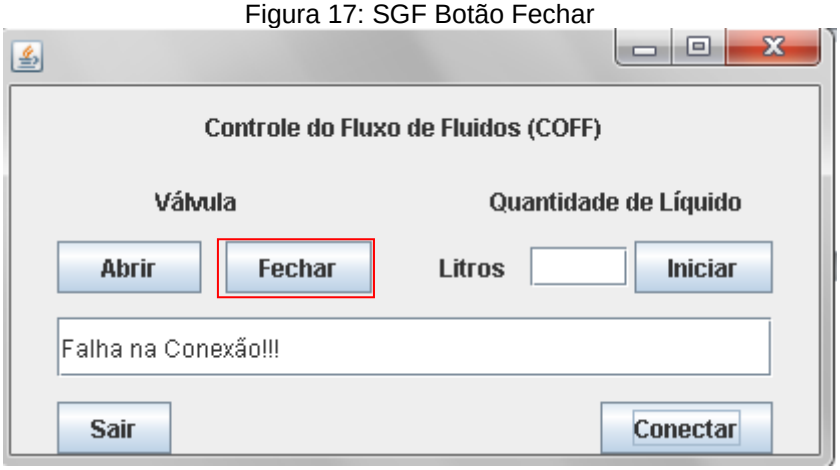
Figura 16: SGF Botão Abrir



Fonte: Os autores.

O segundo é de “Fechar” a válvula;

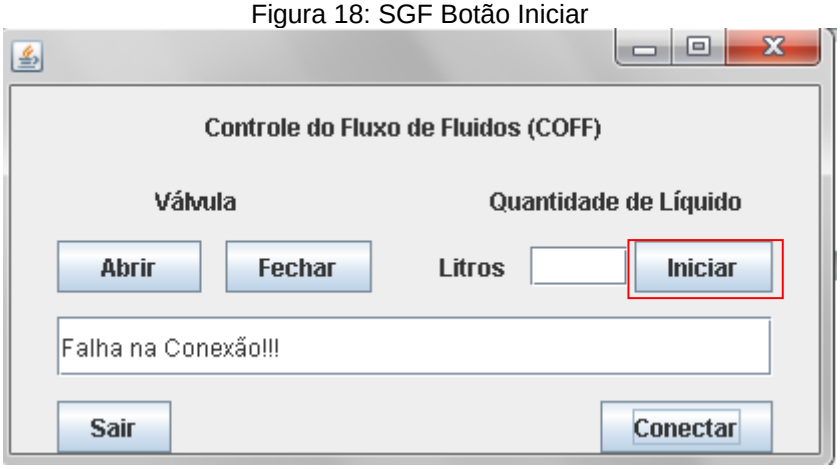
Figura 17: SGF Botão Fechar



Fonte: Os autores.

O terceiro é de “Iniciar” uma contagem da quantidade de litros a vazar;

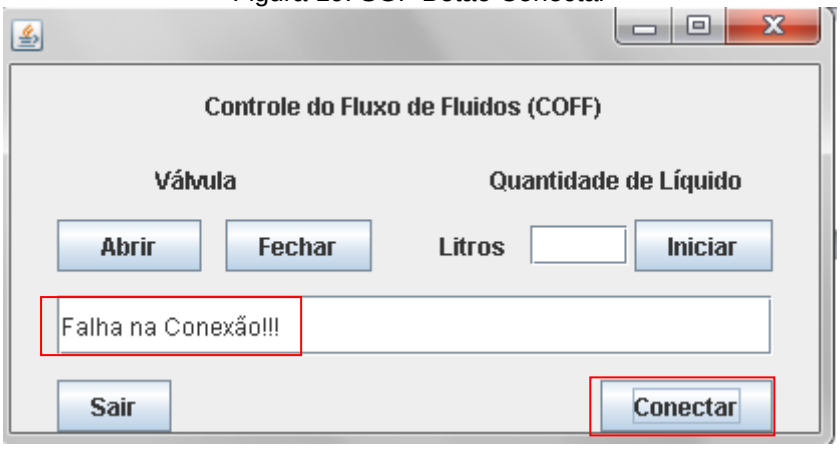
Figura 18: SGF Botão Iniciar



Fonte: Os autores.

Para a conexão tem o botão “Conectar”, se caso der problema ele mostra “Falha na Conexão”, conforme a Figura 19.

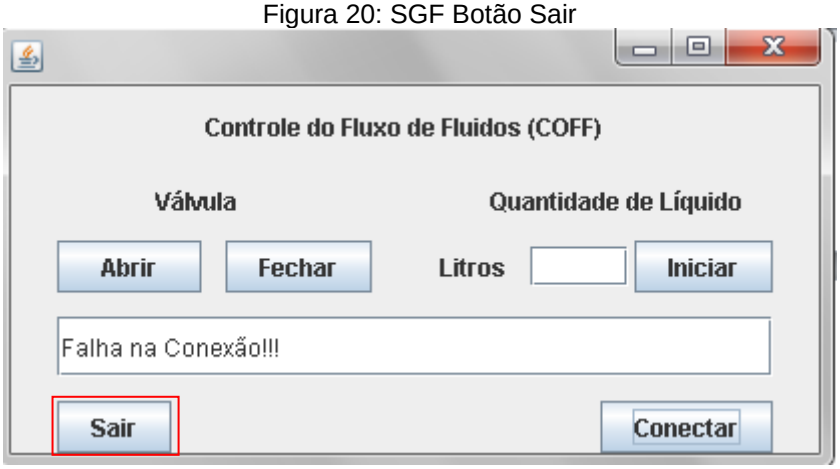
Figura 19: SGF Botão Conectar



Fonte: Os autores.

E por fim tem o botão “Sair” que como o próprio nome diz ele fecha o programa.

Figura 20: SGF Botão Sair



Fonte: Os autores.

No código tem tratamento de erros relacionado à conexão e a falha no envio do arquivo, assim não travando o programa e o mesmo fica sempre disponível e funcional. Para a execução é preciso ter o Java instalado na maquina, ele que é responsável por executar o código.

5.3.2 Problemas e soluções adotados

Existe varias restrições no programa, uma delas é o fato de ele passar apenas inteiros para a contagem do fluxo, isso acontece, pois o tratamento do arquivo é baseando em byte, e só é possível passar um por vez, inviabilizando assim o desenvolvimento do fluxo com números diferentes de inteiro, outro problema é o fato de necessitar de delay para envio das informações, isso gera um tempo de execução considerável do programa.

No arduino foi desenvolvido o programa que faz a comunicação com o Java, ele é responsável por fazer os cálculos e acionar as funcionalidades dos dispositivos, o grande problema no desenvolvimento foi o fato de ter muita comparação de String, isso gera um código extremamente grande e com um uma grande possibilidade de ocorrer erro na execução, sem contar que quanto maior o código e mais comparação o programa faz, mais lento será a execução das funcionalidades. Vale ressaltar que esse programa desenvolvido na linguagem nativa do arduino, tem limitações que complicaram o desenvolvimento do protótipo.

6 RESULTADOS

Com a divisão do projeto, cada integrante ficou responsável por determinadas partes, assim as atividades foram acontecendo de uma forma simultânea. Com essa divisão foi possível à obtenção de vários resultados.

Inicialmente ocorreram imprevistos com os equipamentos, os mesmos ficaram retidos por erro na confirmação do pagamento, todavia, o tempo de execução do projeto não foi afetado por esse atrapalho, assim foi possível o comprimento do cronograma inicialmente elaborado.

Para inicio do protótipo, foi criado o circuito completo na protobord, ele que foi a base para a criação da placa *shield* que acoplada diretamente no arduino, tornando-o um protótipo de pequeno tamanho e que não ocupa um grande espaço, obtendo assim o resultado esperado referente ao protótipo de tamanho e peso reduzido.

Ao termino da PCI, foi aprofundada toda a parte de desenvolvimento, a mesma feita para o *firmware* do arduino e para o computador, com o desenvolvimento do software foi possível analisar que a comunicação entre o arduino e o SGF é complicada, uma vez que os dados enviados para o arduino são de difícil tratamento, isso mostra que os resultados obtidos não tem a máxima exatidão.

Outra parte importante para ressaltar, é o fato de que o protótipo trabalha diretamente com a vazão de um liquido, a mesma pode variar conforme a pressão que está sendo exercida, e também pela quantidade de ar contida no cano, esses fatores alteram os resultados.

Em relação à quantidade de liquido que o usuário escolhe teve alguns problemas em seu desenvolvimento; o primeiro não é possível escolher valores sem que sejam inteiros, isso se dá ao fato que a informação que é passada para o arduino bit a bit, isso gera um tratamento de uma char, resultando em um programa muito complexo e com uma grande possibilidade de falha, outro problema é o fato que só é possível escolher uma quantidade de liquido entre os intervalos de 1 a 99, sem a presença do 0 ou acima de 99, isso acontece pelo mesmo motivo citado anteriormente, com essas restrições tem como resultado um sistema que envia comandos via *wireless*, esses comandos podem ser de: abrir fluxo, fechar fluxo e uma quantidade limitada entre 1 e 99 de litros que podem ser escolhido pelo usuário.

7 IMPACTO AMBIENTAL

Toda alteração realizado pelo ser humano ao meio em que ele vive é considerado como impacto ambiental, ele se caracteriza pela quebra do equilíbrio ecológico.

Pela definição formal (UNESP), o Artigo 1º da Resolução n.º 001/86 do Conselho Nacional do Meio Ambiente (CONAMA), Impacto Ambiental é “qualquer alteração das propriedades físicas, químicas, biológicas do meio ambiente, causada por qualquer forma de matéria ou energia resultante das atividades humanas que afetem diretamente ou indiretamente:

- A saúde, a segurança, e o bem estar da população;
- As atividades sociais e econômicas;
- A biota;
- As condições estéticas e sanitárias ambientais;
- A qualidade dos recursos ambientais".

O conceito que engloba toda a parte de preservação do meio ambiente está relacionado diretamente com a sustentabilidade, que é responsável por reproduzir um meio benéfico para tentar balancear as ações degradantes causados anteriormente, que segundo Bernardo (2011, p. 78) a definição mais usado nos dias atuais de sustentabilidade é proposta pela comissão Brundtland que diz: “Desenvolvimento sustentável é aquele que faz face ás necessidades de geração presente, sem comprometer a capacidade das gerações futuras de satisfazer suas próprias necessidades”.

O lixo eletrônico é dado como: todo resíduo material produzido pela inutilização de equipamentos eletrônicos, por exemplo, monitores de computadores, telefones celulares e baterias, computadores, televisores, câmeras fotográficas etc.

O descarte dos equipamentos eletrônicos acontece quando o mesmo se torna obsoleto ou sofre algum dando, o grande problema é que o descarte é feito de forma inadequada sendo geralmente depositados no meio ambiente, mas, é valido ressaltar que os equipamentos eletrônicos possuem substâncias químicas, como chumbo, cádmio, mercúrios etc., que provocam contaminação no solo e na água, e

ainda essas substanciam causam doenças graves em pessoas que coletam em lixões.

Para que todo esse problema seja amenizado é correto fazer o descarte do lixo eletrônico em locais apropriados, por exemplo, empresas que atuam no setor de reciclagem desses materiais tóxicos, devolver as baterias para as operadoras que tem o destino adequando para elas.

O COFF tem visado reduzir impacto ambiental que o homem causa a natureza por meio de um sistema de controle de água, diminuído assim o desperdício de água, e consequentemente gerando um protótipo de grande utilidade para sistemas que necessitem de um controle do fluxo.

8 CONSIDERAÇÕES FINAIS

Tendo em vista o uso do protótipo para o controle do fluxo em um ambiente específico o projeto se mostrou eficiente, o fator que mais chama a atenção é o fato dele ser uma aplicação multiplataforma que executa em qualquer maquina que esteja instalado o Java 1.8 e tenha um dispositivo que capte a rede *wireless*. Pelo SGF ser uma aplicação desenvolvida em Java, ela se tornou muito intuitiva e de fácil uso, tendo na sua composição botões, e indicativos de erro, se o mesmo acontecer.

O objetivo geral do projeto foi alcançado, com as ressalvas que existe um percentual de erro no controle do fluxo, pois a vazão na qual o sensor consegue captar é mostrada em litros por minuto, por ela ser uma grandeza que estima a media, no momento não se pode ter uma total exatidão no controle da quantidade a verter no cano, mas se a pressão e a vazão da agua forem constantes é possível ter um resultado razoavelmente preciso.

Uma sugestão de melhoria no protótipo seria o uso de um sensor com uma maior precisão, e juntamente com ele uma válvula responsável por retirar o ar do cano. Em relação ao SGF o desenvolvimento do software em tempo real que monitora a quantidade que está sendo vertida.

Os próximos passos para melhorar o COFF seria um estudo mais detalhando sobre técnicas de medição de fluidos, e a melhoria no próprio protótipo, isso inclui, um melhor acabamento e conectores fáceis, além da troca de toda a fiação. Com essas mudanças e adaptações o resultado seria de maior exatidão.

REFERÊNCIAS

BERNARDO, M. Políticas públicas e sociedade civil. In: BURSZTYN, M. **A difícil sustentabilidade: política energética e conflitos ambientais**. Rio de Janeiro: Garamond, 2001, p. 78.

CADENHEAD. Rogers. **Aprenda em 21 dias Java 2.** Rogers Cadenhead, Laura Lernay; tradução Daniel Vieira e Ana Beatriz Tavares. – Rio de Janeiro: Elisevier 2005. p. 4.

DIY. **Monitoramento e controle de nível de líquido**. Disponível em : http://www.diy.com.br/projeto/monitoramento-e-controle-de-nivel-de-liquido. Acesso em: 13 junho 2015.

FOROUZAN, B. A.; MOSHARRAF, F. **Redes de computadores: uma abordagem Top-Down**. Porto Alegre: AMGH, 2012. p(s). 2,21.

NULL, L.; LOBUR, J. **Princípios Básicos de Arquitetura e Organização de Computadores**. 2.ed. Porto Alegre: Bookman, 2010.p(s). 530, 531.

SAPO. **Redes – Sabe o que é o modelo OSI?**. Disponível em: <http://pplware.sapo.pt/tutoriais/networking/redes-sabe-o-que-e-o-modelo-osi/>. Acesso em: 30 maio 2015.

TMDATA. **Internet das Coisas**. Disponível em: <http://tmdata.com.br/a-internet-das-coisas-a-arma-da-fidelidade/>. Acesso em: 30 maio 2015.

UOL. **Wi-Fi, Bluetooth, ZigBee ou Z-Wave: uma briga sem fio**. Disponível em: <http://revistahometheater.uol.com.br/site/tec_artigos_02.php?id_lista_txt=8828>. Acesso em: 30 maio 2015.

USINAINFO. Disponível em: http://www.usinainfo.com.br/module/csblog/detailpost/106-79-aviso-de-fluxo deagua.html. Acesso em: 13 junho 2015.

UNESP. **Impacto Ambiental**. Disponível em: <http://www.rc.unesp.br/igce/aplicada/ead/estudos_ambientais/ea03.html>. Acesso em: 31 maio 2015.

XXANDE. **Sistemas embarcados**. Disponível em: <http://xxande.blogspot.com.br/2012/09/sistemas-embarcados.html>. Acesso em: 31 maio 2015.

ANEXO A – CÓDIGO FIRMWARE

```
#include <SoftwareSerial.h>

SoftwareSerial esp8266(10,9);
String srt;
String  srt1;
int ValorQt1;
int Sairloop = 2;
int Sairloop2 = 2;
int Sairloop3 = 2;
//controle da valvula
int sinalparaorele = 3;
char c;
int numero;
int composto1;
int composto2;
int numerototal;

//controle do sensor
double vazao; //Variável para armazenar o valor em L/min
int contaPulso; //Variável para a quantidade de pulsos
int i=0; //Variável para contagem
int cont = 0;
double media = 0;
double Litros = 0;
int dado; //variável que receberá os dados da porta serial
int cont1 = 0;
int ledVazao = 11;
double qutLitro = 0;
int tempo =0;

void setup() {

  Serial.begin(9600); // Serial
  esp8266.begin(9600); // Wirelles
  pinMode(sinalparaorele,OUTPUT);// pino de Rele
   pinMode(ledVazao,OUTPUT);
  pinMode(2, INPUT);
  attachInterrupt(0, incpulso, RISING);
  //Cria o Server
   esp8266.println("AT+RST");
   delay(4000);
  esp8266.println("AT+CIPMUX=1");
  delay(2000);
  esp8266.println("AT+CIPSERVER=1,9999");

  }

void loop() {
```

```
while(tempo <= 7){
  delay(1000);
  Serial.println(tempo);
  tempo = tempo +1;
}

 srt1 = "";
  srt = "";
  dado = 0;
 composto1 = 0;
 composto2 = 0;
 numero = 0;
 Sairloop = 2;
 Sairloop2 = 2;

   c = esp8266.read();
   srt = String(c);

   if (srt.substring(0,1) == "6") {
    digitalWrite(sinalparaorele,HIGH);
     Serial.print(c);
   }
   if (srt.substring(0,1) == "7") {
    digitalWrite(sinalparaorele,LOW);
    Serial.print(c);
    delay(500);
   }

    if (srt.substring(0,1) == "8") {

      while (Sairloop < 3){
          srt1 = "";
        char e = esp8266.read();
        srt1 = String(e);

     if (srt1.substring(0,1) == "g") {
     numero = 1;
     Serial.println("numero 1");
     srt = "";
     srt1 = "";
     Sairloop = 4;
     }
    if (srt1.substring(0,1) == "h") {
     numero = 2;
     Serial.println("numero 2");
     srt = "";
     srt1 = "";
     Sairloop = 4;
     }
      if (srt1.substring(0,1) == "j") {
```

```
numero = 3;
Serial.println("numero 3");
srt = "";
srt1 = "";
Sairloop = 4;
}
 if (srt1.substring(0,1) == "4") {
numero = 4;
Serial.println("numero 4");
srt = "";
srt1 = "";
Sairloop = 4;
}
 if (srt1.substring(0,1) == "5") {
numero = 5;
Serial.println("numero 5");
srt = "";
srt1 = "";
Sairloop = 4;
}
  if (srt1.substring(0,1) == "6") {
numero = 6;
Serial.println("numero 6");
srt = "";
srt1 = "";
Sairloop = 4;
}
  if (srt1.substring(0,1) == "7") {
numero = 7;
Serial.println("numero 7");
srt = "";
srt1 = "";
Sairloop = 4;
}
if (srt1.substring(0,1) == "8") {
numero = 8;
Serial.println("numero 8");
srt = "";
srt1 = "";
Sairloop = 4;
}
  if (srt1.substring(0,1) == "9") {
numero = 9;
Serial.println("numero 9");
srt = "";
srt1 = "";
Sairloop = 4;
}
}
}
```

```
// parte do numero composto

if (srt.substring(0,1) == "9") {

 while (Sairloop < 3){
    srt1 = "";
   char e = esp8266.read();
   srt1 = String(e);

if (srt1.substring(0,1) == "g") {
composto1 = 1;
Serial.println("numero 1");
srt = "";
srt1 = "";

}
if (srt1.substring(0,1) == "h") {
composto1 = 2;
Serial.println("numero 2");
srt = "";
srt1 = "";

}
 if (srt1.substring(0,1) == "j") {
composto1 = 3;
Serial.println("numero 3");
srt = "";
srt1 = "";

}
 if (srt1.substring(0,1) == "4") {
composto1 = 4;
Serial.println("numero 4");
srt = "";
srt1 = "";

}
 if (srt1.substring(0,1) == "5") {
composto1 = 5;
Serial.println("numero 5");
srt = "";
srt1 = "";

}
  if (srt1.substring(0,1) == "6") {
composto1 = 6;
Serial.println("numero 6");
srt = "";
srt1 = "";

}
```

```
  if (srt1.substring(0,1) == "7") {
 composto1 = 7;
 Serial.println("numero 7");
 srt = "";
 srt1 = "";

 }
 if (srt1.substring(0,1) == "8") {
 composto1 = 8;
 Serial.println("numero 8");
 srt = "";
 srt1 = "";

 }
  if (srt1.substring(0,1) == "9") {
 composto1 = 9;
 Serial.println("numero 9");
 srt = "";
 srt1 = "";

  }
  // SEGUNDA PARTE DO NUMERO COMPOSTO

 if (srt1.substring(0,1) == "m") {
  Sairloop = 2;

 while (Sairloop < 3){

     srt1 = "";
   char e = esp8266.read();
   srt1 = String(e);

    if (srt1.substring(0,1) == "g") {
 composto2 = 1;
 Serial.println("numero 1");
 srt = "";
 srt1 = "";
 Sairloop = 4;
 }

   if (srt1.substring(0,1) == "h") {
 composto2 = 2;
 Serial.println("numero 2");
 srt = "";
 srt1 = "";
 Sairloop = 4;
 }
  if (srt1.substring(0,1) == "j") {
composto2 = 3;
 Serial.println("numero 3");
 srt = "";
```

```
 srt1 = "";
 Sairloop = 4;
 }
  if (srt1.substring(0,1) == "4") {
 composto2 = 4;
 Serial.println("numero 4");
 srt = "";
 srt1 = "";
 Sairloop = 4;
 }
  if (srt1.substring(0,1) == "5") {
 composto2 = 5;
 Serial.println("numero 5");
 srt = "";
 srt1 = "";
 Sairloop = 4;
 }
   if (srt1.substring(0,1) == "6") {
 composto2 = 6;
 Serial.println("numero 6");
 srt = "";
 srt1 = "";
 Sairloop = 4;
 }
   if (srt1.substring(0,1) == "7") {
 composto2 = 7;
 Serial.println("numero 7");
 srt = "";
 srt1 = "";
 Sairloop = 4;
 }
 if (srt1.substring(0,1) == "8") {
 composto2 = 8;
 Serial.println("numero 8");
 srt = "";
 srt1 = "";
 Sairloop = 4;
 }
   if (srt1.substring(0,1) == "9") {
 composto2 = 9;
 Serial.println("numero 9");
 srt = "";
 srt1 = "";
 Sairloop = 4;
   }
   }
 }
}
numerototal = (composto1 * 10) +  composto2;

}
```

```
  // continua o codigo
  if (numero > 0){
    dado = numero;
  }

  if (numerototal > 0){
    dado = numerototal;
  }

  if (dado > 0){

    qutLitro = 0;

      // delay(100);

       digitalWrite(ledVazao,HIGH);
       qutLitro = dado;
       digitalWrite(sinalparaorele, HIGH);
    while (qutLitro >= 0 ){
 delay(1000);
contaPulso = 0;   //Zera a variável para contar os giros por segundos
sei();      //Habilita interrupção
delay (1000); //Aguarda 1 segundo
cli();      //Desabilita interrupção
vazao = contaPulso / 3.3 ; //Converte para L/min

media=media+vazao; //Soma a vazão para o calculo da media
i++;

if(i==2)
{

 media = (media/2)/60; //Tira a media dividindo por 60
 Litros = Litros + media;
 qutLitro = qutLitro - media;
 Serial.println(qutLitro);
 Serial.print("\nMedia por minuto = "); //Imprime a frase Media por minuto =
 Serial.print(Litros); //Imprime o valor da media
 Serial.println(" L/min - "); //Imprime L/min
 media = 0; //Zera a variável media para uma nova contagem
 i=0; //Zera a variável i para uma nova contagem

if (Litros >= dado){
  Serial.print(qutLitro);
    sei();      //Habilita interrupção
   digitalWrite(sinalparaorele, LOW);
  int valf= 0;
  while (valf <= 4){
    delay(1000);
    valf = valf +1;
```

```
    }
    Litros = 0;
  }
  }
 }
  digitalWrite(ledVazao,LOW);
    }
   }
   void incpulso ()
{
 contaPulso++; //Incrementa a variável de contagem dos pulsos
}
```

ANEXO B – CÓDIGO SGF

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package br.com.coff;

import java.io.IOException;
import java.io.OutputStream;
import java.net.Socket;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author Daniel
 */
public class Tela extends javax.swing.JFrame {

    Socket cliente = null;
    OutputStream os = null;

    public Tela() {
        initComponents();
        BT_Abrir.setEnabled(false);
        BT_Fechar.setEnabled(false);
        BT_Iniciar.setEnabled(false);
        L_Litros.setEnabled(false);

    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        BT_Conectar = new javax.swing.JButton();
        BT_Sair = new javax.swing.JButton();
        BT_Fechar = new javax.swing.JButton();
        BT_Iniciar = new javax.swing.JButton();
        NomeDoPrograma = new javax.swing.JLabel();
        TV = new javax.swing.JLabel();
        TQL = new javax.swing.JLabel();
        TLitros = new javax.swing.JLabel();
```

```
L_Litros = new javax.swing.JTextField();
L_Retorno = new javax.swing.JTextField();
BT_Abrir = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

BT_Conectar.setText("Conectar");
BT_Conectar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        BT_ConectarActionPerformed(evt);
    }
});

BT_Sair.setText("Sair");
BT_Sair.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        BT_SairActionPerformed(evt);
    }
});

BT_Fechar.setText("Fechar");
BT_Fechar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        BT_FecharActionPerformed(evt);
    }
});

BT_Iniciar.setText("Iniciar");
BT_Iniciar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        BT_IniciarActionPerformed(evt);
    }
});

NomeDoPrograma.setText("Controle do Fluxo de Fluidos (COFF)");

TV.setText("Válvula");

TQL.setText("Quantidade de Líquido ");

TLitros.setText("Litros");

BT_Abrir.setText("Abrir");
BT_Abrir.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        BT_AbrirActionPerformed(evt);
    }
});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
```

```
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(72, 72, 72)
                .addComponent(TV)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(TQL)
                .addGap(35, 35, 35))
            .addGroup(layout.createSequentialGroup()
                .addGap(23, 23, 23)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(BT_Abrir, javax.swing.GroupLayout.DEFAULT_SIZE, 72, Short.MAX_VALUE)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(BT_Fechar)
                        .addGap(34, 34, 34)
                        .addComponent(TLitros)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(L_Litros,                javax.swing.GroupLayout.PREFERRED_SIZE,                49,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(3, 3, 3)
                        .addComponent(BT_Iniciar))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(BT_Sair)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(BT_Conectar))
                    .addComponent(L_Retorno))
                .addGap(23, 23, 23))
            .addGroup(layout.createSequentialGroup()
                .addGap(97, 97, 97)
                .addComponent(NomeDoPrograma)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(NomeDoPrograma)
                .addGap(21, 21, 21)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(TV)
                    .addComponent(TQL))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(BT_Fechar)
                    .addComponent(BT_Iniciar)
                    .addComponent(TLitros)
                    .addComponent(L_Litros,                                javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                .addComponent(BT_Abrir))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(L_Retorno, javax.swing.GroupLayout.DEFAULT_SIZE, 30, Short.MAX_VALUE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(BT_Conectar)
                .addComponent(BT_Sair)))
    );

    pack();
}// </editor-fold>

private void BT_ConectarActionPerformed(java.awt.event.ActionEvent evt) {



    try {
        cliente = new Socket("192.168.4.1", 9999);
        L_Retorno.setText("Conectou com Sucesso!!!");
        os = cliente.getOutputStream();
    } catch (IOException ex) {
        L_Retorno.setText("Falha na Conexão!!!");
    }
    BT_Abrir.setEnabled(true);
    BT_Fechar.setEnabled(true);
    BT_Iniciar.setEnabled(true);
    L_Litros.setEnabled(true);
}

private void BT_SairActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        cliente.close();
    } catch (IOException ex) {
       L_Retorno.setText("Problema ao fechar a porta de comunicação!!!");
    }
    catch (NullPointerException exb){
     System.exit(0);
    }
    System.exit(0);
}

private void BT_AbrirActionPerformed(java.awt.event.ActionEvent evt) {
    try {

        os.write('6');
        L_Retorno.setText("Dado Enviado!!!");
        Thread.sleep(200);
    } catch (IOException e) {
       L_Retorno.setText("Falha no envio da informação!!!");
    } catch (InterruptedException ex) {
        Logger.getLogger(Tela.class.getName()).log(Level.SEVERE, null, ex);
    }
```

```
}

private void BT_FecharActionPerformed(java.awt.event.ActionEvent evt) {
     try {
        os.write('7');
        L_Retorno.setText("Dado Enviado!!!");
        Thread.sleep(200);
    } catch (IOException ee) {
       L_Retorno.setText("Falha no envio da informação!!!");
    } catch (InterruptedException ex) {
        Logger.getLogger(Tela.class.getName()).log(Level.SEVERE, null, ex);
    }
}

private void BT_IniciarActionPerformed(java.awt.event.ActionEvent evt) {
    int a= 1;
    int b= 1;
    int c= 1;

   String N_Litros = L_Litros.getText();
    int litro = Integer.parseInt(N_Litros);
    if (N_Litros == ""){
        L_Retorno.setText("Digite um numero!!!");
        a= 0;
    }else
    if (N_Litros.length() >= 3 ){
        L_Retorno.setText("Valor acima do permitido!!!");
        b= 0;
    }else

     if (litro  < 0 ){
        L_Retorno.setText("Valor abaixo do permitido!!!");
         c = 0;
    }

    if ((a == 1) && (b == 1) && (c ==1) ){

  int Litros  =  Integer.parseInt(N_Litros);

    char[] LitroPorPartes = String.valueOf( Litros ).toCharArray();
    try {

        int quantidade = LitroPorPartes.length;

        if (quantidade == 1){
            os.write('8');
          Thread.sleep(200);
          if(LitroPorPartes[0] == '1'){
              char v_numero = 'g';
              os.write(v_numero);
          }
```

```
  if(LitroPorPartes[0] == '2'){
     char v_numero = 'h';
     os.write(v_numero);
  }
   if(LitroPorPartes[0] == '3'){
     char v_numero = 'j';
     os.write(v_numero);
  }
  if (LitroPorPartes[0] >3 ){
  os.write(LitroPorPartes[0]);
  }

}
// numero composto
if (quantidade == 2){
    os.write('9');
    Thread.sleep(400);

  if(LitroPorPartes[0] == '1'){
     char v_numero = 'g';
     os.write(v_numero);
  }
   if(LitroPorPartes[0] == '2'){
     char v_numero = 'h';
     os.write(v_numero);
  }
   if(LitroPorPartes[0] == '3'){
     char v_numero = 'j';
     os.write(v_numero);
  }
  if (LitroPorPartes[0] >3 ){
  os.write(LitroPorPartes[0]);
  }

Thread.sleep(300);
os.write('m');
Thread.sleep(300);


    if(LitroPorPartes[1] == '1'){
     char v_numero = 'g';
     os.write(v_numero);
  }
   if(LitroPorPartes[1] == '2'){
     char v_numero = 'h';
     os.write(v_numero);
  }
   if(LitroPorPartes[1] == '3'){
     char v_numero = 'j';
     os.write(v_numero);
  }
```

```
            if (LitroPorPartes[1] >3 ){
            os.write(LitroPorPartes[1]);
            }

           }

       } catch (IOException e) {
          L_Retorno.setText("Falha no envio da informação!!!");
       } catch (InterruptedException ex) {
          L_Retorno.setText("Erro no tempo de envio!!!");
       }
       }
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(Tela.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(Tela.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(Tela.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(Tela.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new Tela().setVisible(true);
        }
    });
}

// Variables declaration - do not modify
private javax.swing.JButton BT_Abrir;
```

```
    private javax.swing.JButton BT_Conectar;
    private javax.swing.JButton BT_Fechar;
    private javax.swing.JButton BT_Iniciar;
    private javax.swing.JButton BT_Sair;
    public javax.swing.JTextField L_Litros;
    public javax.swing.JTextField L_Retorno;
    private javax.swing.JLabel NomeDoPrograma;
    private javax.swing.JLabel TLitros;
    private javax.swing.JLabel TQL;
    private javax.swing.JLabel TV;
    // End of variables declaration
}
```

Classe principal

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package br.com.coff;
import java.io.IOException;

/**
 *
 * @author Daniel
 */
public class Principal {

    public static void main(String[] args) {

        new Tela().setVisible(true);
    }
}
```